

Quest SQL Optimizer for SAP ASE 3.9

User Guide



Copyright 2018 Quest Software Inc. ALL RIGHTS RESERVED.

This guide contains proprietary information protected by copyright. The software described in this guide is furnished under a software license or nondisclosure agreement. This software may be used or copied only in accordance with the terms of the applicable agreement. No part of this guide may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording for any purpose other than the purchaser's personal use without the written permission of Quest Software Inc.

The information in this document is provided in connection with Quest Software products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Quest Software products. EXCEPT AS SET FORTH IN THE TERMS AND CONDITIONS AS SPECIFIED IN THE LICENSE AGREEMENT FOR THIS PRODUCT, QUEST SOFTWARE ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL QUEST SOFTWARE BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF QUEST SOFTWARE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Quest Software makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. Quest Software does not make any commitment to update the information contained in this document.

If you have any questions regarding your potential use of this material, contact:

Quest Software Inc.

Attn: LEGAL Dept

4 Polaris Way

Aliso Viejo, CA 92656

Refer to our Web site (www.quest.com) for regional and international office information.

Patents

Quest Software is proud of our advanced technology. Patents and pending patents may apply to this product. For the most current information about applicable patents for this product, please visit our website at www.quest.com/legal.

Trademarks

Quest, and the Quest logo are trademarks and registered trademarks of Quest Software Inc. in the U.S.A. and other countries. For a complete list of Quest Software trademarks, please visit our website at www.quest.com/legal. All other trademarks, servicemarks, registered trademarks, and registered servicemarks are the property of their respective owners.

Legend

-  **WARNING:** A WARNING icon indicates a potential for property damage, personal injury, or death.
-  **CAUTION:** A CAUTION icon indicates potential damage to hardware or loss of data if instructions are not followed.
-  **IMPORTANT, NOTE, TIP, MOBILE, or VIDEO:** An information icon indicates supporting information.

Contents

Introduction to Quest SQL Optimizer for SAP® ASE	24
Performance Diagnostics	24
Performance Optimization	24
Performance Management	24
Database Administration	24
Development	25
Functionality	25
New in This Release	25
Database Versions Supported	25
User Logon Privileges	25
Adaptive Server Configuration	28
SELECT permission on database objects	29
sa_role privilege	29
Create privileges	29
Adaptive Server monitoring tables	30
mon_role privilege	30
Configuration Parameters	30
Enter Authorization Key	32
Connect to the Database	32
Synchronize the Data Dictionary	33
About Product Improvement Program	33
SQL Inspector Tutorial	34
Create the Inspector for Monitor Tables	34
Create the Inspector for QP Metrics	34
Start the Inspector	35
View the results	35
SQL Collector for Monitor Server Tutorial	35
To use the collector	35
SQL Scanner Tutorial	36
Open a Scanner Group	36
Add Scanner Jobs	37
Scan Jobs	37
View scanning results	37
Performance Monitor Tutorial	38
Retrieve statistics from monitoring tables	38
SQL Optimizer Tutorial	38
Optimize a SQL statement	38
Compare SQL alternatives	39

Batch test SQL alternatives	39
Review test results	41
Index Advisor Tutorial	42
Create Index candidates	42
Test index candidates	42
Abstract Plan Manager Tutorial	43
Manage Abstract Plans	43
SQL Repository Tutorial	44
Add SQL to the SQL Repository	44
Save SQL to the SQL Repository from other modules Tutorial	44
Save SQL to SQL Repository	44
Configuration Analyzer Tutorial	45
Analyzer Page	45
Select SQL Page	45
Configuration Page	45
Reviewing Configuration Analyzer Results	46
Migration Analyzer Tutorial	46
Analyzer Page	46
Select SQL Tab	46
Migration Page	47
Destination Configuration Page	47
Reviewing Migration Analyzer Results	47
Index Impact Analyzer Tutorial	48
Analyzer Page	48
Select SQL Page	48
Index Page	48
Review Index Impact Analysis Results	49
Index Usage Analyzer Tutorial	49
Analyzer Page	50
Select SQL Page	50
Review Unused Index Analyzer Results	50
Database Explorer Tutorial	50
Browse Database Objects	50
Object Extractor Tutorial	51
Extract DDL	51
SQL Worksheet Tutorial	51
Executing SQL or Transact-SQL	51
Code Finder Tutorial	52
Search for Text	52
SQL Formatter Tutorial	52
User-Defined Temp Tables Tutorial	53
Create Temporary Tables	53

SQL History Tutorial	53
Find Previously Used SQL Statements	53
Preferences Overview	54
Optimization Settings for Pre-ASE 15	54
Intelligence (Optimization for Pre-ASE 15 Tab)	54
Optimization (Optimization Tab)	56
Forces (Pre-ASE 15)	58
Quota (Pre-ASE 15)	59
Abstract Plan (Optimization Tab)	60
Optimization Settings for ASE 15	61
Intelligence (Optimization for ASE 15 Tab)	62
Optimization (Optimization Tab)	63
Forces/Goal (ASE 15)	65
Criteria	67
Join	67
Union Distinct	67
Union All	68
Group By	68
Distinct	68
Individual	69
Quota (ASE 15)	70
Abstract Plan (Optimization Tab)	71
Index Advisor	72
Directory Setup	74
Directory (1)	74
Directory (2)	76
Activity Log	77
SQL History	78
Plan	79
SQL Worksheet	80
SQL Scanner	81
Options (SQL Scanner Tab)	82
Abstract Plan (SQL Scanner Tab)	84
General (SQL Scanner Tab)	84
SQL Classification	86
Problematic SQL	87
Complex SQL	88
Simple SQL	89
Invalid SQL	89
No privilege to tables or views	89
Using the wrong database or user	90
Dynamic SQL statements	90
General (General Tab)	90
Message (General Tab)	92

Result Set (General Tab)	93
Batch Run (General Tab)	93
Editor	93
Database Setting	95
DDL	96
Performance Monitor Preferences	97
General Tab	97
Refresh Data Setting	98
Preserve Data	99
Chart Data Normalization Factor (CNF)	99
Show confirmation before closing application (Default = checked)	100
Chart Setting Tab	100
Chart Setting	101
Information Tab	102
SQL statement	103
Highlight selected column	103
SQL Filter (executing SQL only)	103
Display Connection Information window only when insufficient privileges (Default = checked)	103
Top-N Display Tab	104
Syntax Highlight	105
Element options	105
Elements	105
Foreground color	106
Background color	106
Editor options	106
Reset Default button	106
Auto Correction	107
Member and Argument Lookup	108
Member lookup (Default)	109
Argument lookup (Default)	109
Delay (Default = 0.75 sec, Range = 0.5 to 1.5 sec)	109
Indent and Outdent Text	109
Comment and Uncomment Text	109
Change Text to Uppercase and Lowercase	110
Bracket Pairing	110
Customize Editable Panes	111
Editor Functions	111
Editor Panes Shortcuts	112
Send SQL Statement to SQL Optimizer	113
Supported SQL Statements	114
Setting Database and User	114
SQL from the SQL Scanner	114
SQL in a Cursor	114

SQL with Temporary Tables	114
Copy to SQL Worksheet	115
Send to Index Advisor	115
Open SQL from SQL Repository	115
Save SQL to SQL Repository	116
Create Benchmark Factory Import File	116
SQL Navigation	117
Modify Data	117
Navigate Data	117
Edit a Record	118
Insert a Record	118
Delete a Record	118
Refresh Data	118
Copy Data from a Single Cell	118
Comments	119
Parameters	119
Parameters Window	119
Filter Database Objects	121
SQL Information Pane	122
Query Plan	122
Abstract Plan	122
Trace On	122
Information	123
All Records	123
First Record	123
Scanner Temp Table (SQL Scanner)	124
Checked Details (SQL Scanner)	124
Index Advisor Window (Index Advisor)	124
Query Plan Overview	124
Review Query Plans	125
Query Plan Options	126
Animate Query Plans	126
Copy Query Plans	127
Plan Detail	127
Quick Find	129
Find	129
Options	129
Scope	129
Direction	130
Origin	130
Find Next	130
Replace	130

Goto Line	131
Performance Monitor Overview	131
Performance Monitor Privileges	132
Supported Monitor Tables	132
Performance Monitor Window	134
Left pane	135
Right Pane for Database Statistics Summary	135
Right Pane for Monitor Statistics	135
ASE Overall State Statistics	135
ASE Engines Statistics	135
Open Database Statistics	136
Open Objects	136
Cache Statistics	137
I/O Statistics	137
Device I/O	137
I/O Queue	137
Network I/O	138
Process Statistics	138
Right Pane for Information	138
Cache Information	138
Process Information	138
SQL Statement Information	139
Locks Information	139
Sys Worker Threads	139
Sys Waits	139
Error Log	139
Status Bar	140
Open the Performance Monitor	140
Connect and Disconnect from Database	140
View connection information	141
Start and Stop the Performance Monitor	141
Manual Refresh Data	142
Hide and Show Console Tree	142
Display View for Performance Monitor Charts	142
Chart Drill Down	143
Lock and Unlock Chart	143
Chart Zoom	143
Copy Chart	144
Print Chart	144
Save Chart	144
Performance Monitor Functions	144
Performance Monitor Preferences	145
General Tab	145
Refresh Data Setting	146

Preserve Data	147
Chart Data Normalization Factor (CNF)	147
Show confirmation before closing application (Default = checked)	148
Chart Setting Tab	148
Chart Setting	149
Information Tab	150
SQL statement	151
Highlight selected column	151
SQL Filter (executing SQL only)	151
Display Connection Information window only when insufficient privileges (Default = checked)	151
Top-N Display Tab	152
SQL Inspector Overview	153
SQL Inspector Privileges	153
Performance Statistics - Monitor Tables	154
Performance Statistics - QP Metrics	155
Data Directory Setting	155
Open SQL Inspector	156
SQL Inspector Window	156
Inspector List pane	157
Statistics Graphs pane	157
Statistics pane	157
SQL Text pane	158
Status Bar Information	158
Add/Modify Inspector	158
General Information	159
Monitor Process (Monitoring Tables)	159
Retrieval Criteria (Monitoring Tables)	160
Schedule Settings (Monitoring Tables)	162
Retrieval Criteria (QP Metrics)	163
Schedule Time (QP Metrics)	164
Inspect	165
Ad-hoc Inspect	166
Abort Inspect	166
Scan Inspector	167
Find Inspector	167
Delete Marked Inspector	168
View Inspector Properties	168
Place Bookmarks in SQL Inspector Window	169
Create Benchmark Factory Import File	169
View and Add Charts	169
Statistics	169

No. of Top Consumption SQLs and Percentage of Total Consumption sliders	170
Add another chart	170
Select the SQL Filtering Criteria	170
Show SQL Satisfying All Charts	170
Show SQL Satisfying Any Chart	170
View the SQL text	171
Scan Inspected SQL	171
Sum Statistics for a Stored Procedure	171
SQL Inspector Functions	172
SQL Collector for Monitor Server Overview	173
Adaptive Server Configuration Parameters for SQL Collector	174
Adaptive Server Version	174
Adaptive Server Enterprise Monitor Server	174
Monitor Server Name on Client SQL.INI and Server must match.	174
Monitor Server Name in Collector setup must match	174
Tempdb size	174
Logon Privilege	174
Adaptive Server parameter	175
Data Directory Setting	175
SQL Collector Window	176
Collector List	176
SQL Text	177
SQL Information	177
Status Bar	177
Open SQL Collector for Monitor Server	178
Add or Modify a Collector	178
General Information	178
Monitor Process	179
Collecting Criteria	180
Schedule Settings	182
Query Plan	183
Monitored SQL Statement Types	183
Problematic SQL	183
Complex SQL	183
Simple SQL	184
SQL without Plan	184
Monitor	184
Ad-hoc Monitor	185
Abort Monitor	185
Why use the SQL Scanner to Identify Problematic SQL Statements	186
Scan Collectors	186
Place Bookmarks in SQL Collector	186

View Particular Types of SQL Statement	187
Find Specific Information in Collectors	188
Find Tab	188
Text to find	188
Options	188
Advanced Tab	188
Text to find	188
Type	189
Search Results	189
Find SQL Statement	189
Generate a Report for Monitored SQL Statements	189
Collector Properties	190
SQL Collector Functions	191
SQL Scanner Overview	192
SQL Scanner Window	193
Job List Pane	193
SQL Text Pane	195
SQL Information Pane	195
Status Bar	195
Group Manager Window	196
View Group Summary	196
View Group Properties	197
General tab	197
SQL Summary tab	197
Embedded and Dynamic SQL Statements	198
Embedded SQL Statements	198
Dynamic SQL Statements	198
What files does SQL Scanner scan?	198
Database Objects	198
Abstract Plan Groups	198
SQL Collector for Monitor Server Files	198
SQL Inspector Files	199
Text/Binary Files	199
COBOL Files	199
Data Directory Setting	199
Open SQL Scanner	200
Add Jobs to SQL Scanner	200
Add Options	200
Summary Options	200
Add Database Objects (Add Jobs Wizard)	201
Add Abstract Plan Groups (Add Jobs Wizard)	202
Add SQL Collectors from the SQL Collector (Add Jobs Wizard)	202
Add SQL Inspectors (Add Jobs Wizard)	203

Add Source Code (Add Jobs Wizard)	204
Summary (Add Jobs Wizard)	205
Use Current DB & User	205
Scan	206
Scan Source Code with Temp Tables	206
Source Code	206
After scanning	207
SQL1	207
SQL2	207
Review Scanned SQL	207
Find Jobs	208
Find SQL Using a Text String	209
View Job Properties	209
General tab	209
SQL Summary tab	210
Open Group Manager while SQL Scanner is Open	210
Check Scanned SQL	210
Marking a Scanned SQL Statement as Checked.	211
Checked Details	212
Unmarking a Checked Scanned SQL Statement.	212
Adding the Scanned SQL Statement to Checked List Automatically	212
Preserving Checked SQL when rescanning	212
Mark and Unmark All Jobs	213
Move or Copy Jobs to Other Group	213
Switch Database and User	213
Modify a Job	214
Place Bookmarks in SQL Scanner	214
Open in SQL Collector	214
Open in SQL Inspector	215
Create Benchmark Factory Import File	215
View a Particular SQL Type	215
Generate Report for Scanned SQL	216
Save Abstract Plan	217
Create Benchmark Factory Import File	218
SQL Scanner Functions	218
SQL Conversion Overview	220
Trigger Conversion	220
Original SQL statement	220
After conversion	221
External Parameter Conversion	221
Original SQL statement	221
After conversion	222

Local Variable Conversion	222
Original SQL statement before scanning	222
After conversion	222
Cursor Query Plan Conversion	222
Into Clause Conversion	223
Original SQL statement	223
After conversion	223
COBOL Conversion	223
Conversion for variable name	223
Conversion for comment	223
Conversion for concatenate character	224
Original SQL statement	224
After conversion	224
Index Advisor Overview	224
Index Advisor Window	224
Top Pane	225
Left Bottom Pane	225
Right Bottom Pane	225
Bottom Left Pane	226
Bottom Right Pane - SQL Information Pane	227
SQL for Cursor Checkbox	227
Cursor Arguments	227
Run Time Tab	228
Privileges	229
Statistics Tab	229
Chart Tab	230
Filter the Run Time Results	232
Advise Index Alternatives	232
Index Advising Details	233
Add Index Alternatives	234
Add user-defined indexes	235
Create function-based indexes	235
Create Index Sets	235
Delete User-defined Indexes	236
Add Index Sets	236
Delete user-defined Index Set	236
Add and remove index from Index Set	236
Batch Show Plan	236
Create Indexes	237
Analyze Impact of New Indexes	237
Clear All Index Alternatives	237
Index Advisor Functions	238
Determine Best Performing Index Alternative	239

Create Indexes for Batch Run	240
Selected Index Set	240
SQL Termination (Options)	242
SQL Termination (Percentage Delay)	243
Batch Termination	244
No termination	244
Terminate Batch Run if the specified number of index sets fall in the criteria.	244
Run Time Mode	245
Batch Run Schedule	246
Schedule setting	247
View Batch Run Details	248
SQL Optimizer Overview	248
SQL Syntax Transformation	248
Applying Adaptive Server Optimization Techniques	249
Eliminating Duplicate Query Plan	249
Testing for Best Alternative	249
Intelligence Level Settings	249
Create Your Own Alternative SQL	249
Optimization Engine	250
Common Coding Errors in SQL Statements	250
What Function Should I Use to Retrieve the Run Time?	251
Unsatisfactory Performance Results	251
SQL Optimizer Functions	252
SQL Editor Functions	252
SQL Editor and Optimized SQL Functions	252
Optimized SQL Functions	253
SQL Optimizer Window	254
SQL Text	254
SQL Information	255
Run Time Information	255
Enter Original SQL Statement	255
Retrieve Query plan	256
Show Default Plan	256
Optimize Original SQL Statement	257
SQL for Cursor Checkbox	257
Cursor Arguments	258
Optimize Using Abstract Plan Only	258
Insert User-Defined SQL	259
Open Optimized SQL	259
Create Benchmark Factory Import File	260
Optimized SQL	261
SQL Text	261
SQL Information	262
Run Time Information	262

Optimization Details Window	262
Input Parameter Values	263
Analyze Time and Statistics Results	263
Analyze Run Time Results	263
Analyze Statistics Results	265
Analyze Chart Results	267
Filter Run Time Results	269
Find SQL Using a Text String	270
Generate a Report for Optimized SQL	270
Verify Correctness of Optimized SQL Statements	270
Run Result	271
Number of Records	271
First Record	271
Insert User-Defined SQL	271
Check Abstract Plan Compatibility	272
Save Abstract Plan	273
Test for Scalability	274
Save Optimized SQL	274
Open Optimized SQL	275
View Open Optimized SQL Details	276
Refresh Plan Details Window	277
Retrieve Run Time for a Group of SQL	278
Selected SQL	279
SQL Termination (Options)	280
SQL Termination (Percentage Delay)	281
Batch Termination	282
Run Time Mode	284
Batch Run Schedule	285
Commit or Rollback	287
View Batch Run Details	287
Run Time	287
Retrieve Run Time	288
Terminate a Run Time SQL Statement	289
Retrieve Run Result	289
Commit or Rollback	290
Terminate Retrieval of Run Result	290
SQL Comparer Overview	290
SQL Comparer Window	291
Open SQL Comparer Window	292
Start and Stop Comparison	292
Switch View in SQL Comparer Window	292
SQL Comparer Functions	293
Activity Log Overview	293
Start Recording Activities to the Activity Log	294
Activity to be logged	294
Information to be logged	294

View Activity Log Report	295
User Information	295
Activity Information	296
Purge Activity Log Data	297
Stop Recording Activities	297
SQL Worksheet Overview	298
SQL Worksheet Window	298
Command List	298
Object Explorer	299
Editor Pane	299
Information Pane	299
Message Log	299
Result	299
Status Bar	299
Progress Bar	299
Num, Insert, Caps, Scroll	299
Modified	300
n:n	300
nnn bytes	300
SPID	300
Execute SQL or Transact SQL	300
Retrieve Description of Tables and Views	301
Modify Data	301
New Database Session	302
Temporary Tables	302
Changes to the Data Dictionary	302
Command List	302
Save Commands	302
Load Commands	302
Clear Commands	303
Bookmarks	303
SQL Worksheet Functions	304
SQL Formatter Overview	305
SQL Formatter Window	305
Left Pane (Original SQL)	306
Right Pane (Formatted SQL)	306
Format a SQL statement	306
Parameters within SQL Formatter	307
SQL Formatter Functions	308
Database Explorer Overview	309
Database Explorer Window	309
Left Pane (Database Objects)	310
Right Pane (Object Information)	310

Database Explorer Privileges	310
Open Database Explorer	311
Modify Data	312
Copy column names to another window	312
Database Explorer Functions	313
Code Finder Overview	313
Code Finder Window	314
Search Criteria	314
Where to Search	315
Search Result	315
Search Database Objects and Files	316
For database objects	316
For directories and files	316
Regular Expression	316
Open Database Object or File in SQL Worksheet	318
Extract Object DDL to File	318
Save Search Results	319
Code Finder Functions	319
Object Extractor Overview	320
Object Extractor Window	320
Select objects to be extracted [left pane]	321
Objects to extract [right pane]	321
DDL [bottom pane]	322
Buttons	322
Extract Object DDL and Dependencies	322
Extraction Criteria	323
Place a DROP command before the object	323
Qualify object with database and user name	323
Include indexes for table	323
Place comments in script	323
Show on next extraction	323
Object Extractor Functions	324
SQL Repository Overview	324
SQL Repository Window	325
SQL Listing [Left pane]	325
SQL Listing Details [Top right pane]	325
SQL Information Details [Bottom right pane]	326
Properties button	326
SQL Information button	326
Save the Abstract Plan	326
Add SQL to SQL Repository	326

Add SQL Wizard: General Page	328
Add SQL Wizard: SQL Information Page	329
Left pane	329
Right Pane	329
Check SQL button	329
Add SQL Wizard: Settings Page	330
SQL Settings section	330
Abstract Plan section	330
Database Settings section	330
Refresh SQL Query Plan	331
Modify SQL	331
Rename SQL	332
Delete SQL	332
Create Benchmark Factory Import File	333
SQL Repository Functions	333
Index Impact Analyzer Overview	334
Index Impact Analyzer Window	334
Left pane - Analyzer List	335
Right Pane	335
Right Pane for Analyzers	336
Properties Button	336
Right Pane for Scenarios Analyzed Folder	336
Summary button	336
Chart button	336
Right Pane for Scenarios	336
Properties button	336
Prognosis button	337
Create/Drop Indexes button	337
Right Pane for SQL Analyzed Folder	337
Cost Summary button	337
Cost Classification button	337
Right Pane for SQL Statements	338
Properties button	338
SQL Information button	338
Saving the Abstract Plan	338
Create an Index Impact Analyzer	338
New Analysis Wizard: Analyzer Page	339
Creating a new Analyzer and select SQL to be analyzed	340
Continuing an existing Analyzer using the Analyzer's selected SQL	340
Do not drop the indexes after the analysis is finished	340
New Analysis Wizard: Select SQL Page	340
Select SQL to check for performance changes:	341
SQL query plans to be analyzed:	341
New Analysis Wizard: Index Page	342
Query plans will be grouped under a Scenario	342

Enter and select indexes to be analyzed	343
Add Index Wizard	343
New Analysis Wizard: Abstract Plan Page	345
Baseline Plan	345
Scenario	346
Drop Indexes	346
, View Index Impact Analysis Details	347
Add SQL to an Index Impact Analysis	347
Add a Scenario	348
Method One	348
Method Two	348
Delete an Index Impact Analysis	349
Modify an Index Impact Analyzer	349
Regenerate	349
Rename an Index Impact Analyzer	350
Index Impact Analyzer Functions	350
Index Usage Analyzer Overview	351
Index Usage Analyzer Window	351
Left pane - Analyzer List	351
Right Pane	351
Right Pane for Analyzers	352
Index Usage Chart [Top right pane]	352
Analysis Information [Bottom right pane]	352
Right Pane for Tables Analyzed	352
Summary button	352
Chart button	352
Right Pane for Analyzed Table Information	353
Prognosis Button	353
Unused Index Button	353
Table Information Button	353
Right Pane for SQL Analyzed Folder	354
Index Used	354
Right Pane for SQL Statements	354
Properties button	354
SQL Information button	354
Index Used button	354
Create an Index Usage Analysis	355
New Analysis Wizard: Analyzer Page	355
New Analysis Wizard: Select SQL Page	356
Select SQL to check for unused indexes:	357
Bottom Pane	357
Update an Index Usage Analysis	357
View Index Usage Analysis Details	358
Modify an Index Usage Analysis	359

Rename an Index Usage Analysis	359
Delete an Index Usage Analysis	360
Index Usage Analyzer Functions	360
Configuration Analyzer Overview	360
Configuration Analyzer Window	361
Left pane - Analyzer List	361
Right Pane	362
Right Pane for Analyzers	362
Properties Button	362
Right Pane for Scenarios Analyzed	362
Summary button	363
Chart button	363
Right Pane for Scenarios	363
Properties button	363
Prognosis button	363
Configuration Parameters button	363
Apply/Undo Changes button	364
Right Pane for SQL Analyzed Folder	364
Cost Summary button	364
Cost Classification button	364
Right Pane for SQL Statement	364
Properties button	364
SQL Information button	364
Saving the Abstract Plan	365
Create a Configuration Analysis	365
New Analysis Wizard: Analyzer Page	366
Creating a new Analyzer and select SQL to be analyzed	366
Continuing an existing Analyzer using the Analyzer's selected SQL	366
Do not rollback changes to the ASE configuration parameters	366
New Analysis Wizard: Select SQL page	367
Select SQL to check for performance changes:	368
SQL query plans will be analyzed:	368
New Analysis Wizard: Configuration Page	368
New Analysis Wizard: Abstract Plan	370
Regenerate a Configuration Scenario	372
Add a Scenario	372
Method One	373
Method Two	373
View Configuration Analysis Details	373
Rollback Changes to Configuration Parameters	374
Add SQL to a Configuration Analysis	374
Modify a Configuration Analysis	375
Rename a Configuration Analysis or Scenario	375
Delete a Configuration Analysis or Scenario	376

Modify a Configuration Scenario	376
Configuration Analyzer Functions	376
Migration Analyzer Overview	377
Migration Analyzer Window	377
Left pane - Analyzer List	378
Right Pane	378
Right Pane for Analyzers	379
Properties Button	379
Right Pane for Scenarios Analyzed	379
Summary button	379
Charts button	379
Right Pane for Scenarios	380
Properties button	380
Prognosis button	380
Database Properties button	380
Configuration Parameters button	380
Apply/Undo Changes button	380
Right Pane for SQL Analyzed Folder	381
Cost Summary button	381
Cost Classification button	381
Right Pane for SQL Statement	381
Properties button	381
SQL Information button	381
Saving the Abstract Plan	381
Create a Migration Analysis	382
New Analysis Wizard: Analyzer Page	383
Creating a new Analyzer and select SQL to be analyzed	383
Continuing an existing Analyzer using the Analyzer's selected SQL	383
Do not rollback changes to the ASE configuration parameters	383
New Analysis Wizard: Select SQL Page	384
Select SQL to check for performance changes:	385
SQL query plans will be analyzed:	385
New Analysis Wizard: Migration Page	386
New Analysis Wizard: Destination Configuration Page	387
New Analysis Wizard: Abstract Plan	389
Baseline Plan	389
Scenario	390
View Migration Analysis Details	390
Migration Analysis Results	391
Properties Button	391
Prognosis Button	391
Chart Tab	391
Summary Tab	392
Configuration Parameters Button	392
Apply Changes	392

Add a Scenario	392
Method One	392
Method Two	393
Rollback Changes to Configuration Parameters	393
Add SQL to a Migration Analysis	393
Modify a Migration Analysis	394
Regenerate a Migration Scenario	394
Delete a Migration Analysis or Scenario	395
Rename a Migration Analysis or Scenario	395
Modify a Migration Scenario	395
Migration Analyzer Functions	396
Abstract Plan Manager Overview	396
Abstract Plan Group	397
Why Save the Abstract Plan?	398
Abstract Plan Compatibility with Original SQL	399
Abstract Plan Manager Window	400
Left Pane	401
Right Pane	401
Open the Abstract Plan Manager	401
Abstract Plan Group Functions	402
Export a Group to a Table	402
Import an Abstract Plan for each User	403
Abstract Plan ID Functions	404
Viewing the text of the SQL statement	404
Use Saved Abstract Plans	405
Using Abstract Plans for Specific Applications	405
Using the Abstract Plans Server Wide	405
User-Defined Temp Table Overview	406
Create Temporary Tables	406
Parameters in Temporary Table SQL	408
View SQL scripts of temporary tables	409
Drop Temporary Tables	410
Example of Temporary Tables in SQL Scanner	410
Copy SQL with Temporary Tables to SQL Optimizer	412
Create Alternative SQL Which Uses Temporary Tables	412
Original SQL	413
SQL1 Alternative	413
Preference Settings for Handling Temporary Tables	413
SQL Scanner Tab	413
Optimization Tab	414
SQL History Overview	414

SQL History Window	414
SQL Text drop-down field	415
Last Connection drop-down field	415
Last Action drop-down field	415
Apply Filter Button	415
Button Bar	416
Grid	416
Filter SQL Statements	416
Recall a SQL Statement	417
SQL History Functions	418

Introduction to Quest SQL Optimizer for SAP® ASE

Quest SQL Optimizer for SAP ASE enables IT professionals to analyze, predict, preempt, diagnose, optimize and manage performance changes to ensure that mission-critical business applications run optimally. It maximizes application performance by focusing on those database factors that contribute up to 90% of performance problems: SQL statements, indexes, and database changes. Quest SQL Optimizer for SAP ASE allows for the management and prediction of performance changes; therefore, changes that will affect the database environment can be managed with the highest degree of certainty. It is the optimal performance change management solution for production and quality assurance teams.

SQL Optimizer has the following main modules and functionality:

Performance Diagnostics

- [SQL Inspector](#)
- [SQL Collector for Monitor Server](#)
- [SQL Scanner](#)
- [Performance Monitor](#)

Performance Optimization

- [SQL Optimizer](#)
- [Index Advisor](#)
- [Abstract Plan Manager](#)

Performance Management

- [Configuration Analyzer](#)
- [Migration Analyzer](#)
- [Index Impact Analyzer](#)
- [Index Usage Analyzer](#)

Database Administration

- [Database Explorer](#)
- [Object Extractor](#)
- [SQL Worksheet](#)

Development

- [SQL Formatter](#)
- [Code Finder](#)

Functionality

- [User-Defined Temp Table](#)
- [SQL History](#)
- [SQL Repository](#)
- [Editor Functions](#)
- [Activity Log](#)

New in This Release

Quest SQL Optimizer for SAP® ASE 3.9 is a maintenance release containing bug fixes and enhancements.

Database Versions Supported

Adaptive Server 15.0 or later is supported.

User Logon Privileges

Adaptive Server has database privileges that either allow or restrict the access and authority of each user. Some of the modules in SQL Optimizer need specific privileges. When you logon to SQL Optimizer, it checks to see if the logon satisfies the privileges. If your user account does not have a specific privilege, you can still use all the other modules in the program.

To see if your logon satisfies the privilege requirements for each module

Select **Database** | **View Connection Information**.

Privilege/Setting	Program Module	Operation	Description
"select on syscomments.text",1	Database Explorer and SQL Scanner	Retrieve the database objects' SQL text.	You must have privilege to access the system catalog table, syscomments, therefore the "select on syscomments.text" configuration parameter must be turned on.

<p>sa_role Adaptive Server 15 or later</p> <p>"event buffers per engine", 2000</p> <p>"max SQL text monitored", 4096</p>	<p>SQL Collector for Monitor Server</p>	<p>Capture currently running SQL statements</p>	<p>The SQL Monitor is not available if you do not have sa_role privileges. In addition, the following Adaptive Server parameters need to be set to use the Adaptive Server Enterprise Monitor Server. They are:</p>
<p>sa_role</p>	<p>SQL Optimizer</p>	<p>Abort Run Result for SQL statements with result set</p>	<p>If you do not have sa_role privileges, you will not be able to abort the Run Result operation.</p> <p>Note: The sa_role is only required for Run Result operation. You can terminate SQL statements during a Batch Run without the sa_role.</p> <p>For SQL statements without a result set (for example, INSERT, UPDATE, DELETE and SELECT with INTO clause) can be terminated without having the sa_role.</p>
<p>sa_role Adaptive Server 15.0 or later</p> <p>"allow resource limits", 1</p>	<p>SQL Optimizer, Migration Analyzer, Configuration Analyzer, Index Impact Analyzer, Index Advisor</p>	<p>Retrieve Estimated I/O Cost</p>	<p>The estimated I/O cost is not available for SQL statements during optimization under the following conditions:</p> <ol style="list-style-type: none"> Your logon account does not have sa_role privilege. You are using Adaptive Server 15.0 or later and "allow resource limits" configuration parameter is turned off. <p>If you do not have the privilege to retrieve the estimated I/O cost, then "N/A" is shown for the cost.</p>
<p>sa_role</p> <p>Turn on dbcc traceon (3604,302,310) option under the Database Settings tab of the Preferences setting.</p>	<p>SQL Information Panel</p>	<p>Trace on information</p>	<p>The "Trace on" feature shows the reason why the Adaptive Server optimizer has chosen a particular way to execute the source or alternative SQL statement and displaying the reasons for index and table joins selection. This information is located in the SQL Optimizer window or the SQL Scanner window. "Trace on" details are not available if you do not have sa_role privilege. In addition, the dbcc traceon option needs to be selected from the Preferences settings.</p>

			<p>a. Select Preferences from the File menu.</p> <p>b. Select the Database Settings tab page.</p> <p>c. Check ON the dbcc traceon (3604,302,310) option.</p>
Create Procedure	SQL Optimizer	Optimize for cursor	<p>The CREATE PROCEDURE privilege is required to optimize a SQL statement that runs inside a cursor.</p> <p>A procedure is created when the SQL for Cursor checkbox is selected in the SQL Optimizer window. The stored procedure is created to retrieve the query plan and run time. The stored procedure is executed and then dropped.</p>
Create/Drop Index	Index Adviser and Index Impact Analyzer	Retrieve query plan and in Index Adviser to retrieve the run time.	You must have the privilege to create and drop indexes when retrieving run time (Index Adviser only) and query plan.
"abstract plan cache",0 "abstract plan replace",0 Adaptive Server 15 and later	SQL Optimizer	Abstract Plans	Required for ASE 15.0 or later to generate and retrieve abstract plans.
mon_role Adaptive Server 15.0 or later	SQL Inspector	Retrieve SQL information from the Performance tables	You must have privilege to access the Adaptive Server monitoring tables, monSysSQLText and monSysStatement, therefore the mon_role is required.
"enable monitoring", 1 "wait event timing", 1 "process wait events", 1 "max SQL text monitored", 4096 "sql text pipe active", 1 "sql text pipe max messages", 1000 "statement pipe active", 1 "statement pipe max messages", 5000			You must also have several configuration parameters set to enable the monitoring

"SQL batch capture", 1

mon_role
Adaptive Server 15.0 or
later

Performance
Monitor

Retrieve
information
from the
Performance
tables

You must have privilege to access
the several of the Adaptive Server
monitoring tables, therefore the mon_
role is required.

"enable monitoring", 1
"sql text pipe active", 1
"sql text pipe max
messages", 100
"plan text pipe active", 1
"plan text pipe max
messages", 100
"statement pipe active", 1
"statement pipe max
messages", 5000
"errorlog pipe active", 1
"errorlog pipe max
messages", 1024
"deadlock pipe active", 1
"deadlock pipe max
messages", 1024
"wait event timing", 1
"process wait events", 1
"object lockwait timing", 1
"SQL batch capture", 1
"statement statistics active",
1
"per object statistics active",
1

You must also have several
configuration parameters set to
enable the monitoring.

Related Topic

[Adaptive Server Configuration Instructions](#)

Adaptive Server Configuration

Adaptive Server has database privileges that restrict the access and authority of each user. Although it is not required that the user logon have all of these privileges, certain functionality will not be available to the user without specific privileges.

When you logon to Quest SQL Optimizer for SAP ASE, it checks to see if the logon satisfies the privileges below.

To see if your logon satisfies the privilege requirements for each module

1. Select **Database | View Connection Information**.
2. Determine which of the following database setup and user privileges are needed for your environment and follow the steps under the following sections.
 - SELECT Permission on database objects
 - sa_role
 - CREATE PROCEDURE
 - Adaptive Server Monitor Tables
 - mon_role
 - Adaptive Server Configuration Parameters

SELECT permission on database objects

If the user is not the owner of the database objects, the user logon account must have the SELECT permission on the objects. If the user logon account does not have access to database objects in Adaptive Server, then these objects cannot be used or viewed in Quest SQL Optimizer for SAP ASE.

sa_role privilege

The sa_role privilege is needed for a few functions in Quest SQL Optimizer for SAP ASE. If the logon does not have this privilege, most of the functionality of Quest SQL Optimizer for SAP ASE is still available.

Assign 'sa_role' to the logins needing the following functionality:

- Use the SQL Collector for Monitor Server module.
- Abort Run Result function for SQL statements that return data.
- Retrieve Trace On (dbcc) information

Grant sa_role to the logon using the following command in the SQL Worksheet or ISQL:

```
sp_role 'grant', sa_role, logon_name
```

Create privileges

The CREATE PROCEDURE privilege is required to optimize a SQL statement that runs inside a cursor. A procedure is created when the SQL for Cursor checkbox is selected in the SQL Optimizer window (in the SQL Optimizer module) or when the SQL Scanner finds a SQL statement that is executed within a cursor declaration. The stored procedure is created to retrieve the query plan and run time. The stored procedure is executed and then dropped.

If the user logon does not have sa_role which includes this privilege, then grant this privilege to the logon using the following command in the SQL Worksheet or ISQL:

```
grant create procedure to logon_name
```

Adaptive Server monitoring tables

The SQL Inspector and Performance Monitor require installation of the Adaptive Server monitoring table. Later versions of Adaptive Server create the monitoring tables by default when you install Adaptive Server. So depending on the version of Adaptive Server you are running, you may find these tables already installed.

The tables are created using the *installmontables* script which is included in the Adaptive Server install and is located in the \$SYBASE/ASE-*version*/scripts directory (%SYBASE%/ASE-*version*/scripts for NT). This script requires that a server named "loopback" be included in *sys.servers*.

If the monitoring tables are not installed yet in your Adaptive Server, follow these steps to install the tables.

1. Create a loopback server entry either using ISQL or SQL Optimizer.

In SQL Optimizer:

- a. Logon to Quest SQL Optimizer for SAP ASE with **sa**.
- b. In the bottom left corner of the window, set the database to **Master**.
- c. In the SQL Worksheet module, execute: `sp_addserver 'loopback', null, server_name`

In ISQL

- a. Logon to Adaptive Server with **sa**.
 - b. Execute: `sp_addserver 'loopback', null, server_name go`
2. Execute the *installmontables* script located in the \$SYBASE/ASE-*version*/scripts directory (%SYBASE%/ASE-*version*/scripts for NT).

```
isql -Usa -P<sa password> -Sserver_name  
-n -i%SYBASE%/SYBASE_ASE%/scripts/installmontables
```

mon_role privilege

The *mon_role* privilege is needed to run the SQL Inspector and Performance Monitor modules. The *sa* logon does not have this role by default, so it needs to be added to the *sa* logon and any other user logon. Assign 'mon_role' to the logins using the SQL Worksheet module or ISQL with this command.

```
sp_role 'grant', mon_role, logon_name
```

Configuration Parameters

The following modules require specific settings for Adaptive Server configuration parameters. (You can copy, paste and execute the following script into the SQL Worksheet module).

```
/* parameters for SQL Inspector and Performance Monitor modules using monitoring tables */
```

```
/* requires Adaptive Server 12.5.0.3 and later */
```

```
sp_configure "enable monitoring", 1  
go  
sp_configure "sql text pipe active", 1
```

```

go
sp_configure "sql text pipe max messages",1024
go
sp_configure "plan text pipe active",1
go
sp_configure "plan text pipe max messages",5000
go
sp_configure "statement pipe active",1
go
sp_configure "statement pipe max messages",5000
go
sp_configure "errorlog pipe active",1
go
sp_configure "errorlog pipe max messages",1024
go
sp_configure "deadlock pipe active",1
go
sp_configure "deadlock pipe max messages",1024
go
sp_configure "wait event timing",1
go
sp_configure "process wait events",1
go
sp_configure "object lockwait timing",1
go
sp_configure "SQL batch capture",1
go
sp_configure "statement statistics active",1
go
sp_configure "per object statistics active",1
go
sp_configure "max SQL text monitored",4096
go
/* parameters for SQL Inspector module using the QP Metrics */
/* requires Adaptive Server 15.0 and later */
sp_configure 'enable metrics capture',1
go
/* parameters for Abstract Plan Manager module */
/* requires Adaptive Server 12.0 and later */
sp_configure "abstract plan cache",0
go
sp_configure "abstract plan replace",0
go
/* parameters for retrieving Estimated I/O Cost for query plans */

```

```

sp_configure "allow resource limits",1
go
/* parameters for SQL Collector for Monitor Server module*/
sp_configure "event buffers per engine",2000
go
sp_configure "max SQL text monitored",4096
go

/* parameter to access syscomments system table to retrieve DDL for database objects. */

/*Access to the system catalog table, syscomments, is needed to view SQL text for procedures,
triggers, views, default and rules objects. If you do not have access to the syscomments system
table, a message <SQL Text unavailable> will be presented in the Text tab of the Database
Explorer window, and you will be unable to scan database object in the SQL Scanner module. */
sp_configure "select on syscomments.text",1
go

```

Enter Authorization Key

SQL Optimizer requires an authorization key to use the program. When it is first installed it has a trial key that is good for thirty days.

To enter another trial key or to enter the production key

1. Select **Help | Register**.
2. Enter the key.
3. When you enter a production key after purchasing the product, you must also enter the Site Message information.
4. Click **OK**.

Connect to the Database

When SQL Optimizer starts, a User Logon dialog field displays for you to connect to the Adaptive Server database.

Item	Description
Login name	Enter the login name required to connect to the database.
Password	Enter the password associated with the login name.
Server name	Enter the Adaptive Server name as defined in the client configuration.

The Server list contains the database servers taken from the SQL.INI file.

Item	Description
Host name	Enter the name of the computer where the database resides.
Port number	Enter the port number that was assigned to Adaptive Server.

After selecting and entering the connection information, click **Connect**.

The Loading Data Dictionary window may be shown while connecting to the database to indicate that information from the data dictionary is being retrieved and loaded in the memory of the computer. This information is used when the SQL statement is parsed for functions such as scanning, optimization, and index generation. Depending on the number of objects in your database, the loading of all the information from the data dictionary may take considerable time. Therefore you may want to use the option in the [General Preferences](#) with does not load the data dictionary when the connection to the database is made, but does load the specific information that is needed when SQL statements are parsed. This saves time when the connection is made and adds a little more time to the parsing process.

Note: The first time the program is launched, the Language dialog field displays a list of available interface and character set languages. The character set allows you to view data in the chosen language.

Synchronize the Data Dictionary

Specific database information, such as tables, indexes, data volumes, and so on, from the data dictionary is used during the optimization process, SQL analysis, and other functions throughout program. This information can be loaded into memory of your PC each time you connect to a database. If your database has lots of database object, this process can take several moments, therefore you can choose to have the specific information loaded as it is needed in the program by clearing the **Load database dictionary after database connection** checkbox in the [Preferences](#) window.

If changes are made to the database while you are using the program, it is important to keep the information in the data dictionary up to date. Using the Synchronize Data Dictionary function will ensure that the changes to the database are directly reflected in program. This function does not break the connection to the database but updates the new database information in the memory on your computer.

To update the database information in the memory of the computer

Select **Database | Synchronize Data Dictionary**.

About Product Improvement Program

To prioritize enhancements in future releases, Quest SQL Optimizer for SAP ASE collects data about the use of its different features, and periodically, this data is communicated back to us. Initially, this usage data includes an IP address. Upon its receipt at a temporary server in the U.S.A., the IP address is removed, and then the anonymous data is aggregated before it is sent to our servers in California. Our product team analyses the aggregated data to understand our user community's preferences and common practices. This analysis influences our future releases. Click [here](#) for more information on the data we collect and on our privacy policy.

- No personal information is collected
- You can stop participating at any time

To initiate participation in Product Improvement Program

Select **Help | Product Improvement Program** and select **Yes, I want to participate**

To cancel participation in Product Improvement Program

Select **Help | Product Improvement Program** and select **No, thank you**

SQL Inspector Tutorial

The SQL Inspector monitors SQL statements and extracts SQL performance statistics from the Adaptive Server monitoring tables. The SQL Inspector also extracts SQL statements from the QP Metrics (sysquerymetrics view). The SQL Inspector graphically displays and compares SQL activity statistics to diagnose performance bottlenecks. With the SQL Inspector you can identify the Top-N most resource intensive SQL statements.

Create the Inspector for Monitor Tables

1. Click .
2. If this is the first time you have used the SQL Inspector, the Add Inspector wizard displays over the SQL Inspector window. Otherwise, click .
3. In the General Information page, enter an Inspector name and select **Monitor Tables**.
4. In the Monitor Process page, specify whether you want to monitor the entire database server, any specific process IDs, or a connection identity, which allows you to focus on a specific user or application.
5. In the Retrieval Criteria page, specify how to retrieve records from the Adaptive Server monitoring tables.
6. In the Schedule Setting page, specify the monitoring schedule.
7. Click **Finish**.

Note: To use the SQL Inspector to access the monitoring tables, your database logon must have mon_ role privilege.

Create the Inspector for QP Metrics

8. Click .
9. If this is the first time you have used the SQL Inspector, the Add Inspector wizard displays over the SQL Inspector window. Otherwise, click .
10. In the General Information page, enter an Inspector name and select **QP Metrics**.

11. In the Retrieval Criteria page, select the Users and Group IDs for the SQL you would like to extract and the number of SQL statements you would like.
12. In the Schedule Time page, specify when to start the inspection.
13. Click **Finish**.

Start the Inspector

14. Click .

View the results

15. Review the Top-N SQL statements chart used for filtering statements and a table with detailed SQL performance statistics.
16. Click  if you want to add multiple charts to filter captured SQL statements by specifying a Top-N SQL criteria of performance statistics.
17. To filter collected SQL statements, use the **No. of Top Consumption SQL** and **Percentage of Total Consumption** sliders.
18. Click the performance statistics line on the grid to view the SQL text in the bottom pane of the window.
19. Once you have identified potentially problematic SQL statements you can send the SQL using the **Edit** menu to the SQL Optimizer or Index Advisor.

Related Topic

[SQL Inspector Overview](#)

SQL Collector for Monitor Server Tutorial

The SQL Collector for Monitor Server allows you to capture from the ASE Monitor Server any currently executing SQL statements according to your user-defined criteria. Each SQL statement captured is categorized according to suspected levels of performance problem.

To use the collector

1. Create the Collector
 - a. Click . The SQL Collector window displays, followed by the Add Collector wizard if you have not created a Collector before. If the Add Collector wizard does not display, click .

- b. Enter a Collector name and the Adaptive Server Enterprise Monitor Server name. Check that the other settings satisfy your requirements. Define a monitoring end time on the **Schedule** page on the Add Collector wizard.
 - c. Click **Finish**.
 2. Start the Collector
 - a. Click  to begin capturing executing statements.
 - b. Details display in the SQL Collector grid as the information accumulates.
 - c. The monitoring process stops at the end time you define under the **Schedule** page in the Add Collector wizard.
 3. View the results
 - a. To view the retrieve SQL statements, select the newly added Collector row.
 - b. In the SQL Text pane, click the tabs, **SQL1**, **SQL2**, and so forth to see the captured SQL statements. The SQL Information pane displays the query plan information.
 - c. Select one SQL statement you want to analyze for performance improvement. Click  to optimize the SQL statement in the [SQL Optimizer](#). Alternatively, you can also send the SQL statement to the [Index Advisor](#). Click .

Related Topic

[SQL Collector for Monitor Server Overview](#)

SQL Scanner Tutorial

Use the SQL Scanner to analyze SQL statements embedded within database objects, text/binary files, Abstract Plan Groups, SQL Collector for Monitor Server files, SQL Inspector files, and application source code. The SQL Scanner extracts each SQL statement embedded within the scanned database objects and files, retrieves their respective query plans from Adaptive Server, and then performs an analysis that determines which of these SQL statements may be a performance bottleneck. You can copy the SQL statements analyzed as problematic (top priority) or complex (second priority) into the SQL Optimizer, or Index Advisor, and/or examine the extracted SQL statements with their query plans.

Open a Scanner Group

1. Click .
2. When you scan the database objects or the application files, you first create a Group to store the items you want to scan.

3. If this is the first time you have used the SQL Scanner, the Create Group window displays. Otherwise, click **Create** in the Group Manager window.
4. Enter a new Group name, e.g. "Test." Click **OK** to close the Create Group window.
5. Check that your new Group name is highlighted in the list field. Click **Open**.

Add Scanner Jobs

6. The selected group is opened in the SQL Scanner window. For a new Group, the Add Jobs wizard is automatically opened so you can select what files or database objects you want to scan.

Note: If you are using an existing Group, click .

7. In the Add Jobs wizard, select the page for the item that you want to scan. You can select Database Objects, Abstract Plan Groups, SQL Collector for Monitor Server files, SQL Inspector files, and application Source Code in text or binary format. Click **Finish**.

Scan Jobs

8. Click .
9. Details are filled in the Job List as the scanning process completes each job. It will show you how many SQL statements found in the Job and how each SQL statement is classified.

View scanning results

10. To view the scanned SQL statements, highlight the job by clicking the row.
11. The first SQL statement found is shown in the SQL Text pane. Click the tabs, e.g. **SQL1**, **SQL2**, **SQL3**, etc., at the bottom left of the pane to view the other SQL statements.
12. Notice the buttons on the top of the SQL Information pane. These buttons display in the, the query plan, the abstract plan, Trace On information, the SQL classification and connection information, the DDL for temporary tables used by the SQL, and the Checked SQL information about SQL statement that you are reviewing.
13. You can narrow the number of original SQL statements to view only the problematic and/or complex statements with **View | Problematic SQL** and/or **View | Complex SQL**.
14. Select one SQL statement you want to analyze for performance improvement. Click  to copy the SQL statement to the [SQL Optimizer](#) window and start the optimization process. Alternatively, you can also send the SQL statement to the Index Advisor. Click  to copy the SQL statement to the [Index Advisor](#) window and generate index options.

Related Topic

Performance Monitor Tutorial

The Performance Monitor monitors and retrieved database performance statistics and performance diagnostics metrics from the Adaptive Server (15.0 and later) monitoring tables. The collected statistics are presented graphically and provides a Top-N view that displays the overall database performance health. The graphical view makes it possible to drill down to get the details of different performance statistics such as engines, CPU, devices, processes, cache, lock, and others.

Retrieve statistics from monitoring tables

1. Click .
2. In the Performance Monitor window enable on the left tree view the detailed performance statistics that you want to monitor. By default, general performance statistics are enabled for monitoring.
3. Click  to start the monitoring process.
4. Click  to stop the monitoring process.

Note: Monitor options and refresh intervals can be changed by in **File | Preferences**.

Related Topic

[Performance Monitor Overview](#)

SQL Optimizer Tutorial

Due to complex nature SQL, there may be many SQL statements that return the same result set, but only a few that may be efficient. The SQL Optimizer applies advanced SQL transformation technology to generate a list of semantically equivalent SQL statements. To identify the most efficient SQL statement for your database environment you can benchmark test the SQL alternatives in your database.

Optimize a SQL statement

1. Open the SQL Optimizer window by clicking .
2. After entering a SQL statement in the SQL Editor pane of the window, click . This step launches the SQL Optimizer that automatically transforms the SQL statement.

Notes:

- The use of forces and other optimization options such as temp table generation, ANSI

JOIN syntax are optional and configurable in the Preferences.

- The intensity of the SQL transformation process is controlled by the Intelligence Level in the Preferences. The Intelligence Levels control how many forces are applied to transformed SQL and how many SQL alternatives are created.
 - If your SQL statement uses a temporary table, see section [User-Defined Temp Table](#) for the steps to create a temporary table in the User-Defined Temp Table module.
3. After optimization, the Optimization Details window shows the total number of semantically equivalent SQL statements, the number of alternative statements with query plans different from your original SQL statement, and a warning message if the number of SQL transformations reaches any of the optimization quotas set in the Preferences.
 4. Click **OK** to close the Optimization Details window.
 5. In the SQL Optimizer window, look at the tabs which are labeled `ALT1`, `ALT2`, `ALT3`, etc. By clicking the tabs you can see the alternative SQL statements that were created by the SQL optimization process. The query plan for each SQL statement displays beside the SQL text.
 6. At the bottom left of the SQL Optimizer window are three tabs, **Time**, **Statistics**, and **Charts**, which display the statistics for each SQL statement after it is executed. At this point since you have not yet run the SQL statements, it displays only the Estimated I/O Cost values. These are only estimations of how each statement will perform. You need to test each statement to obtain its actual run time statistics.
 7. In the SQL Optimizer window, look at the right pane to see the query plan for the SQL statement.

Compare SQL alternatives

8. To see how an alternative SQL statement differs from your original SQL, you can compare these statements side-by-side.
 - Click . Your original SQL statement displays in one pane of the window and an alternative statement in another pane. Blue highlighted items in one pane show the area where there is a difference from the SQL statement displayed in the other pane.
 - At the bottom of the SQL Comparer window, click the **Show query plan** checkbox to display the query plan for both SQL statements.
 - Select **File | Close SQL Comparer**.

Batch test SQL alternatives

9. To prepare to execute the original and the alternative SQL statements, click .
10. In the Batch Run Criteria window, select the SQL statements that you would like to run in batch. Notice the tabs at the top of the Batch Run Criteria window.
11. **Selected SQL Tab**

- Select which SQL statements are to be executed. The blue checkmark in the left column indicates that the SQL statement is selected. By default, all statements are executed.
- To deselect a statement, click that SQL statement in the SQL column, for instance click ALT1.
- To deselect all the SQL statements, right-click and select **Unselect All**.
- To save time when testing run all generated alternatives, a filter function is provided to help you to precisely select the alternatives to test run. Click the **Apply SQL Selection filter** checkbox. Click  to select and apply criteria to narrow down the SQL alternatives to test run.

12. SQL Termination Tab

Select the option for terminating the execution of your original and the alternative SQL statements. The SQL Optimizer generated all the alternative SQL statements in order to find the optimal SQL. Some of those alternatives may run faster than the original SQL, others may run longer. Therefore, you can set the termination criteria to cancel the longer running SQL statements and save database-processing time for the overall batch test. You have these options for terminating your SQL.

- **Original SQL:** Terminate the SQL statement when it has run as long as the original SQL.
- **Best running time SQL:** Run the first SQL statement and use the time from that statement as the termination time. When a SQL statement runs faster than this time, use the faster time as the new termination time, so you are always using the fastest run time as the termination time for the next SQL statement.
- **User-defined time:** Set your own termination time. If your original SQL statement takes a long time to execute and there are many alternative statements, executing all statements may take considerable time. In that event, consider setting an aggressive user-defined termination time. If the original takes 1 hour, try a 5-minute termination time. If no alternative statements execute in under that period, raise the termination time to 10 minutes, etc.
- You can also combine **User-defined time** with **Original SQL** or **Best running time SQL** by clicking the **Or User-defined time** checkbox next to each one.

13. Batch Termination Tab

Select the option for terminating the Batch Run.

- **No termination:** Specify to run the Batch Run to completion.
- **Terminate Batch Run if the specified number of SQL falls in the criteria:**

Specify to terminate the Batch Run when a specified number of SQL statements are found that meet the following requirements for terminating the Batch Run.

Number of SQL (excluding Original): Specify how many SQL statements must be found that show performance improvement over the Original SQL.

Count the SQL if it elapsed time is faster than: Specify one of the following criteria to determine how the performance improvement is determined.

Original SQL: Count all SQL statements that run faster than the run time from the Original SQL.

Original SQL with a percentage of improvement: Count all SQL statements where the run time for the alternative SQL statement is the specified percentage faster than time for the Original SQL statement.

User-defined time (mins/secs): Count all SQL statements that run faster than a specified number of minutes and/or seconds.

14. Run Time Mode Tab

- **Run to retrieve:** Select **First Record** to find the time to process the first record. Select **All Records** to find the time to process all records. You must run the Batch Run twice to get both times.
- **Retrieve the run time by executing:** Select the number of times to run each SQL alternative. SQL statements can be run more than once to eliminate data caching time and obtain more accurate run time statistics.

Note: The Batch Run function provides an efficient way of benchmarking SQL. It runs the selected SQL statements in the database and the SQL statements that exceed the termination time are cancelled. The Batch Run retrieves the time the SQL statement executes in the database and does not retrieve the result set from the database server to the client; so it does not create additional network traffic. For SQL statements such as SELECT...INTO, INSERT, DELETE and UPDATE, each statement is run in a transaction that is ROLLBACK, therefore maintaining the consistency of your data.

15. Batch Run Schedule Tab

- **Start:** Select the time you would like the Batch Run to starting executing.
 - **Until:** Select the option for when the Batch Run should finish.
16. To execute the original and the selected alternative SQL statements, click **OK**. The Batch Run window opens enabling you to view the results as each statement executes.
 17. At the completion of the entire job, the Batch Run Details window replaces the Batch Run window. This window provides greater detail about each SQL statement. Click **OK**.

Review test results

18. Click the **Time** tab in the SQL Optimizer window to see the columns in the SQL Run Time pane which contain the time to execute the entire statement (all records) and/or the time to retrieve only the first record for each SQL statement from the Batch Run.
19. Once you have identified the most-efficient alternative SQL statement to deploy it:
 - You can copy and paste it back in your application.
 - If your SQL statement comes from a database objects, you can open the database object source code in the SQL Worksheet and you can save the alternative SQL statement in a text file either individually or multiple SQL statements in the optimized SQL report. You can modify the SQL with the best alternative.
 - You can save your SQL optimization results for later review. Select **SQL | Saved Optimized SQL**.
 - If you want to use abstract plans to implement the most-efficient query plan, you should have enabled before optimization dump abstract plan in the Optimization Preferences. If abstract plans are displayed in the SQL Optimizer window, click . Select the abstract plan group and check that the abstract plan is correct. Only save abstract plans that are compatible. Select **Save**. This process saves the abstract plan to the database in the specified abstract plan group, so the next time you execute the same SQL statement, the saved abstract plan determines the query plan.

Related Topic

Index Advisor Tutorial

The Index Advisor analyzes the syntax of a SQL statement and the database structure and then proposes new index candidates to help improve performance. It provides detailed information on the suggested indexes, such as, space requirements and selectivity. The index recommendations can be benchmarked to identify which index yields the greatest performance gain. It also enables you to create your own indexes for testing.

Create Index candidates

1. Click .
2. In the top pane under the **SQL Editor** tab, enter the SQL statement for which you want to analyze for index recommendations.

Note: To copy a SQL statement from other windows such as SQL Scanner or SQL Optimizer, click .

3. Click  to see the current query plan and get a list of the indexes used in the current query plan. This index information displays in the **Used Index** tab of the bottom pane.
4. Click .
5. From the Select Tables to Provide Indexes window, select the tables on which you want recommendations for new indexes and specify the sampling size of each table to calculate selectivity. Click **OK**.
6. Once the advising process is completed, the Index Advising Details window displays detailing the index candidates. Click **OK**.
7. The index candidates are displayed on the bottom left pane on the tabs labeled **Index1**, **Index2**, ...**IndexN**. The **Used Index** tab displays the DDL for the index(es) currently used by the original SQL statement. The corresponding query plan, abstract plan, SQL classification, trace on and Sort Resource information are displayed in the SQL Information pane at the bottom right.
8. You can add your own index candidates using . This option displays a GUI for you to create indexes for analysis.

Test index candidates

9. To get the actual run time information for the SQL statement under every index scenario, click . Select your benchmarking options in the Batch Run Criteria window. Click **OK**.

Important Note: This process may impact your database server. Specify the index creation options such as the segment where the index is going to be physically created, and then dropped, and the number of consumers.

10. The Batch Run window opens enabling you to view the results as each statement executes.
11. At the completion of the entire job, the Batch Run Details window replaces the Batch Run window. This window provides greater detail about each SQL statement. Click **OK**.
12. Click the **Time** tab in the Index Advisor window to see the results of the Batch Run.
13. To analyze the impact of every index alternative on the query plans of other SQL statements, click .

Related Topic

[Index Advisor Overview](#)

Abstract Plan Manager Tutorial

The Abstract Plan Manager provides a window for you to easily view, create, delete and modify your abstract plan groups.

In Adaptive Server version 15 and later, the abstract plan enables you to influence the optimization of a SQL statement without having to modify the SQL statement syntax. If you cannot change the source code that contains your SQL statement, you can use the abstract plan to force Adaptive Server to use a specific query plan for a SQL statement. This is particularly useful if you have third party applications where you do not have access to the source code.

Manage Abstract Plans

1. Click to open the Abstract Plan Manager window.
2. You can navigate within the Abstract Plan Manager to work with abstract plan groups or individual plans within any database, and all functions within this module are available from the right-click menu.
3. To create a new abstract plan group, right-click and select **Group | Create**.
4. Select under which database you want this group created (or if you want this created in all databases, select the checkbox **Create in all databases**) and name your abstract plan group.
5. With already created groups, there are many functions you can employ by right-clicking a group including dropping or purging a plan group, performing an import or export of a plan group, or even comparing plan groups.
6. Locating an abstract plan can be done at any level in the tree structure by selecting **Find Abstract Plan | Text**, or **ID**.
7. Individually plans can be manipulated by drilling down to a specific Plan ID and specifying whether you would like to drop, copy, or edit an individual abstract plan.

Related Topic

[Abstract_Plan_Manager_Overview](#)

SQL Repository Tutorial

The SQL Repository stores the SQL statements that are used in the analysis of database performance. These may be SQL statements that you have identified as critical to the performance of your database application.

Add SQL to the SQL Repository

1. Click  to open the SQL Repository window. If no SQL exists in the SQL Repository, then the Add SQL wizard displays automatically. Otherwise, you can open the Add SQL wizard by clicking .
2. In the Add SQL wizard enter the SQL text in the **SQL Information** page.

The SQL syntax is checked and the query plan retrieved before adding a new node to the SQL tree view with the SQL name. Each SQL statement added to the SQL Repository contains a query plan, SQL classification type (Simple, Complex or Problematic) and the current connection information (login name, server name, database and user). The query plan stored with the SQL statement is important as it indicates the current performance of the SQL.

Related Topic

[SQL Repository Overview](#)

Save SQL to the SQL Repository from other modules Tutorial

You can save SQL statements to the SQL Repository from the SQL Inspector, SQL Collector for Monitor Server, SQL Scanner, SQL Optimizer, and SQL Worksheet.

Save SQL to SQL Repository

1. Click .
2. Select the location in which to save the SQL statements and click **OK**.

Note: If you are using this function from the SQL Scanner or SQL Inspector window you need to select which Job or Inspector to be added first. Only valid SQL statements are saved to the SQL Repository.

Related Topic

[SQL_Repository_Overview](#)

Configuration Analyzer Tutorial

The Configuration Analyzer evaluates the effect on SQL performance when changing Adaptive Server parameter settings. It enables you to analyze whether the database performance may improve before you make configuration parameter changes permanent.

1. Click .
2. Click . If this is your first time in the Configuration Analyzer, the New Analysis wizard automatically opens.

Analyzer Page

3. In the New Analysis wizard, under the **Analyzer** page, specify if you want to check the effects of configuration changes by either **Creating a new Analyzer** or **Continuing an existing Analyzer**.
4. Give the analysis a name and description for easy reference.
5. If you would like to create a folder for better organization of your analysis, click .

Select SQL Page

6. Select the source of the SQL statements for Analysis: **SQL Repository** or **SQL Scanner**.
7. Select the SQL statement(s) to add to the Analysis from your predefined SQL statements, or you may add a statement to this Analysis by clicking .
8. Under the SQL Query Plans will be analyzed section, select the options for retrieving your query plan.
 - **Using existing query plan saved with the SQL**
This option uses the query plan that was saved with the SQL statement at the time that statement was saved to the SQL Repository, or scanned in the SQL Scanner.
 - **Obtaining a new query plan under the current connection**
This option retrieves the query plan with the current database logon. This current query plan is compared to the query plan that is retrieved after executing the configuration changes.

Configuration Page

9. Select to view the configuration parameters for the various options within Adaptive Server.
10. Make any parameter changes by inserting a value in the **New Value** column.
11. Click **Finish** to perform the analysis.

Reviewing Configuration Analyzer Results

12. After the analysis, you can see the overall results by clicking the **Analyzer**, and its related information, in the tree structure in the left pane and viewing the corresponding information in the panes to the right.
13. In the right pane for the Scenarios, click **Prognosis** to see overall performance changes and SQL details.

Related Topic

[Configuration_Analyzer_Overview](#)

Migration Analyzer Tutorial

The Migration Analyzer helps you to preempt performance degradation when performing database migrations, upgrades, and application rollouts. It ensures reliable database performance by tracking query plan and cost changes. You have an option to integrate abstract plan management to help stabilize SQL performance during migrations.

1. Click .
2. Click . If this is your first time in the Migration Analyzer, the New Analysis wizard automatically opens.

Analyzer Page

3. In the New Analysis wizard, under the **Analyzer** page, specify if you want to check the effects of database migration by either **Creating a new Analyzer** or **Continuing an existing Analyzer**.
4. Give your analysis a name and description for easy reference.
5. If you want to create a folder for better organization of your analysis, click .

Select SQL Tab

6. Select the source of the SQL statements for Analysis: **SQL Repository** or **SQL Scanner**.
7. Select the SQL statement(s) to add to the Analysis from your predefined SQL statements, or you may add a statement to this Analysis by clicking .

8. Under the SQL Query Plans will be analyzed section, select the options for retrieving your query plan.
 - **Using existing query plan saved with the SQL**
This option uses the query plan that was saved with the SQL statement at the time that statement was saved to the SQL Repository, or scanned in the SQL Scanner.
 - **Obtaining a new query plan under the current connection**
This option retrieves the query plan with the current database logon that is assumed to be the source Adaptive Server instance you are migrating from. This current query plan is compared to the query plan that is retrieved from the destination Adaptive Server instance for the migration.

Migration Page

9. Give a name and description to your Analysis Scenario (i.e. "Migration to ASE 15.0").
10. Specify the connection information of the destination Adaptive Server instance including login name, password, and server name. If you want to test the specified connection, click .
11. Select whether to use the default database and user or whether you want to login with different database and user information.

Destination Configuration Page

The **Destination Configuration** page is used to change the database parameters on the migration database and only available after you have entered your migrating database connection information on the **Migration** page.

12. You can switch the list of configuration parameters for Adaptive Server available in the parameter grid by selecting the parameter category from the Show configuration parameters for drop-down list
13. Make any parameter changes by inserting a value in the **New Value** column.
14. Click **OK** to perform the analysis.

Reviewing Migration Analyzer Results

15. After the analysis, you can see the overall results by clicking the **Analyzer**, and its related information, in the tree structure in the left pane and viewing the corresponding information in the panes to the right.
16. In the right pane for the scenario, click **Prognosis** to see overall performance changes and SQL details.

Related Topic

[Migration Analyzer Overview](#)

Index Impact Analyzer Tutorial

Index Impact Analyzer allows you to analyze the impact of new indexes on other SQL statements in your database.

Note: The indexes that are used in these scenarios will be physically created in order to retrieve the query plans for analysis and then dropped.

1. Click .
2. Click **New Analysis**. If this is your first time in the Index Impact Analyzer, the New Analysis wizard automatically opens.

Analyzer Page

3. In the New Analysis wizard, under the **Analyzer** page, select to check the effects of index creation by either **Creating a new Analyzer ...** or **Continuing an existing Analyzer ...**.
4. Give the analysis a name and description for easy reference.
5. If you would like to create a folder for better organization of your analysis, click .

Select SQL Page

6. Choose the source of the SQL statements for Analysis: **SQL Repository** or **SQL Scanner**.
7. Select the SQL statement(s) to add to the Analysis from your predefined SQL statements, or you may add a statement to this Analysis by selecting .
8. Under the **SQL Query Plans will be analyzed** section, select the options for retrieving your query plan.
 - **Using existing query plan saved with the SQL**
This option uses the query plan that was saved with the SQL statement at the time that statement was saved to the SQL Repository, or scanned in the SQL Scanner.
 - **Obtaining a new query plan under the current connection**
This option retrieves the query plan with the current database logon. This current query plan is compared to the query plan that is retrieved after creating the new index(es).

Index Page

9. Name the Index Scenario and enter an optional description for easy reference.
10. Selecting the segment for the index scenario to be created in and the number of consumers.

11. If the Index Impact Analysis was called from the Index Advisor, the index scenarios are automatically display and you can choose the index scenarios to include for analysis.
12. To manually add indexes click .
 - Give the index a name for easy reference.
 - In the **Index based on** section of this page, specify in which database to create this index and specify the owner to use, and then specify the table that contains the column(s) to include in the index.
 - The columns of the table display in the **Table columns** page and can be selected to use in the index scenario, under the **Index columns** page, by double clicking them or using the right arrow button.
 - Specify the Index type by checking the option to have the index *Unique* and whether to create the index as **Clustered** or **NonClustered**.
13. Click **OK** to perform the Index Impact Analysis.

Note: The indexes are physically created in the database, and then the query plans for the selected SQL statements are retrieved. After the query plan retrieval, the indexes are dropped.

Review Index Impact Analysis Results

14. After the analysis, you can see the overall results by clicking the **Analyzer**, and its related information, in the tree structure in the left pane and viewing the corresponding information in the panes to the right.
15. In the right pane for the scenarios, click **Prognosis** to see overall performance changes and SQL details.

Related Topic

[Index Impact Analyzer Overview](#)

Index Usage Analyzer Tutorial

The Index Usage Analyzer identifies unused indexes by analyzing query plans from SQL statements in your database applications. It examines their query plans and reports any indexes in the database that are not used. You can use this module to quickly identify the indexes in databases that are not contributing to the performance of the database applications. These unused indexes can then be deleted to free up space and improve the speed of the database applications and maintenance.

1. Click .
2. Click . If this is your first time in the Index Usage Analyzer, the New Analysis wizard automatically opens.

Analyzer Page

3. Give the analysis a name and description for easy reference.
4. For better organization, create a folder for the analysis you are creating by clicking .

Select SQL Page

5. Select the source of the SQL statements for Analysis: **SQL Repository** or **SQL Scanner**.
6. Select the SQL statement(s) to add to the Analysis from your predefined SQL statements or you may add a statement to this Analysis by selecting .
7. Click **OK**.

Review Unused Index Analyzer Results

8. After the analysis, you can see the overall results by clicking the **Analyzer**, and its related information, in the tree structure in the left pane and viewing the corresponding information in the panes to the right.
9. In the right pane, click **Index Summary** to see a list of indexes used and unused.

Related Topic

[Index Usage Analyzer Overview](#)

Database Explorer Tutorial

The Database Explorer displays detailed information about database objects including columns, indexes, keys, statistics, DDL, procedure text, and data from the tables.

Browse Database Objects

To browse database objects

1. Click .
2. The left pane of the window displays database objects that relate to your user logon. The right pane displays information about a selected database object.
Note: Access to database objects depends on your database privileges.

3. In the left pane, expand the branches to select a database object.
4. Use the tabs at the bottom of the right pane to view information about the selected database object.

Related Topic

[Database Explorer Overview](#)

Object Extractor Tutorial

The Object Extractor extracts the DDL for creating database objects from the database. As an option, object dependencies can also be retrieved ordering them by the lowest dependencies first.

Extract DDL

1. Click .
2. Select the objects to extract from the left pane [Select objects to be extracted]. Click **Add** to add the selected objects to the right pane [Objects to extract].
3. Click **Extract DDL** to display the DDL script on the bottom pane. Or, click **Extract to file** to save the DDL script directly to a file.
4. In the Extraction Criteria window, select options to generate the DDL script.
5. Click **OK**.
6. Extraction preferences can be changed by selecting **File | Preferences** and selecting the **DDL** tab.

Related Topic

[Object_Extractor_Overview](#)

SQL Worksheet Tutorial

The SQL Worksheet enables you to create and execute Transact-SQL code, database objects and SQL commands.

Executing SQL or Transact-SQL

1. Click .
2. Enter your SQL statement or Transact-SQL code in the Editor pane.

3. Click .
4. Use the drop down field at the top of the Editor pane to review and reuse any code executed in the SQL Worksheet during your session.

Related Topic

[SQL_Worksheet_Overview](#)

Code Finder Tutorial

The Code Finder identifies database objects and files that contain specific text.

Search for Text

1. Click .
2. Enter the text string to search for in the **Text** field in the **Search Criteria** section.
3. Use the **Quick Find**, **Database Objects** and **Directory/Files** tabs to select what to search. You can search in database objects (Tables, Views, Procedures, Triggers, Rules, Defaults, etc.) and/or files.
4. Click **Search**.
5. Double-click an item in the Search Result pane to open the matched database object or file in the [SQL Worksheet](#).

Related Topic

[Code Finder Overview](#)

SQL Formatter Tutorial

The SQL Formatter formats SQL statements, verifies syntax, and color-codes variables, invalid field or table names, optimization forces and comments. The use of indenting and highlighting gives SQL statements a standard of formatting that is easy to read.

To format a SQL statement

1. Click .
2. After entering a SQL statement in the left pane of the window, click .

Comments, bind variables, optimizer forces, invalid column or table names, and variables are highlighted in different colors. If there is a syntax error, an error message from Adaptive Server displays.

Related Topic
[SQL Formatter Overview](#)

User-Defined Temp Tables Tutorial

When your SQL statement uses a temporary table, you must create a temporary table before you use the SQL statement in several modules. When you exit from the program or connect to another session, all the temporary tables you created are dropped.

Create Temporary Tables

1. Click .
2. On the **Creation** tab, type in the statements for creating the temp table. You may include CREATE, SELECT INTO, INSERT, UPDATE and DELETE statements.
3. Click **Execute**.

Related Topic
[User-Defined Temp Tables Overview](#)

SQL History Tutorial

Throughout the use of the product, SQL statements are saved in the SQL History so that you can use them again. They are stored in a file so that they are available even if you exit the program.

Find Previously Used SQL Statements

1. Click .
2. To lookup SQL statements, enter the text string to search for in the **SQL Text** field. You can specify if you want to search in the last connection or last action. In the **Last Action** field, select an action from the drop down field. Only actions that have occurred in the present session will be available.
3. Select the SQL statement and click one of the buttons in the window to copy the SQL statement to another module.

Related Topic
[SQL History Overview](#)

Preferences Overview

Preferences allow the alteration of the default settings used in several of the modules.

To open the Preferences window

Click .

When you change the settings, your latest settings are always saved and restored each time to exit and restart the program. You can also save several sets of preference settings either by saving settings from one or more tab pages or by saving all the preference settings.

To save specific settings so that you can use them later

1. Select the Preferences settings.
2. Click **Save**.
3. Enter a filename for the saved settings. Click **Save**.
4. From the Save Preferences Profile window, select the tab pages for the setting you want to save. Click **OK**.

To restore the saved settings

1. In the Preferences window, click **Load**.
2. In the Open File window, select the filename.
3. Click **Open**.

Optimization Settings for Pre-ASE 15

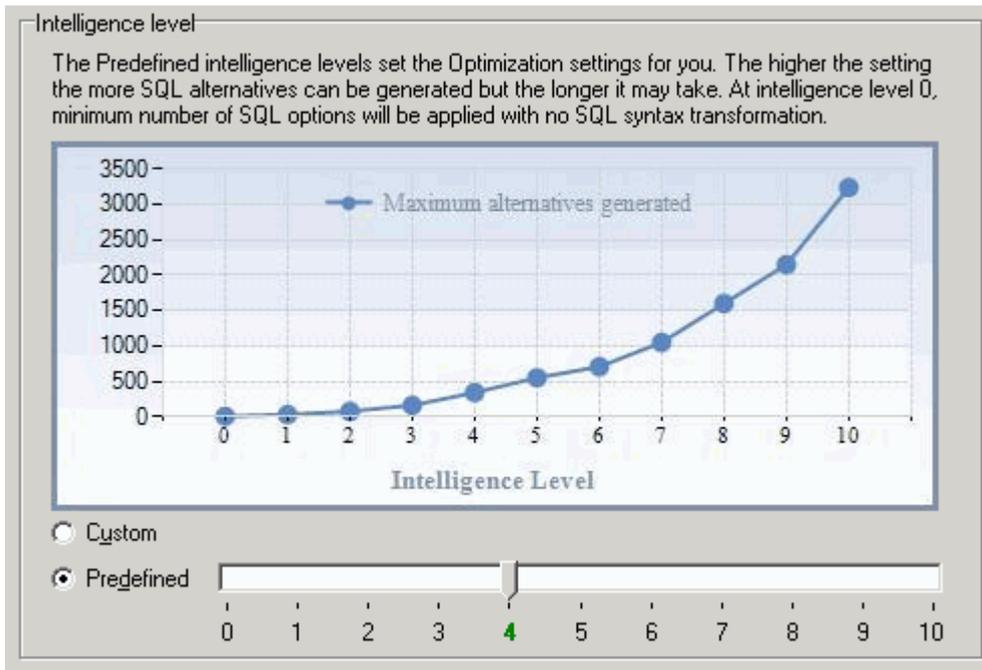
The **Optimization for Pre-ASE 15** tab on the **Preferences** window consists of 5 buttons that allow users to select the level of optimization intelligence and options for optimization while using the SQL Optimizer module when connected to Adaptive Server 12.5.2 or earlier.

Related Topic

[SQL Optimizer Overview](#)

Intelligence (Optimization for Pre-ASE 15 Tab)

View Optimization for Pre-ASE 15--Intelligence



The Intelligence page on the Optimization tab of the Preferences window allows you to select your optimization intelligence settings.

Optimization intelligence settings enables you to either choose the settings used during optimization or allows you to select the predefined "Intelligence Level" settings.

Custom

Customize the settings on the Optimization, Forces, and Quota pages.

Predefined

Use the predefined optimization intelligence levels. The items selected on the Forces and Quota pages and the **Temp table generation** option on the Optimization page change according to the level selected. Levels range from 0 to 10. The higher the level the more intelligent the SQL Optimizer is and the more likely of finding a better SQL alternative.

The Abstract Plan settings on the Abstract Plan page and the **Join table** and **Advanced SQL transformation** options on the Optimization page is adjusted independent of the optimization intelligence level.

Intelligence Level for Pre-ASE15											
	Level 0	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Level 7	Level 8	Level 9	Level 10
Temp Table											
Temp table generation					<input checked="" type="checkbox"/>						
Apply Forces									<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Join Tables											
Rewrite SQL using the Ansi-92 JOIN syntax											
Rewrite SQL without using the Ansi-92 JOIN syntax	<input checked="" type="radio"/>										
Both									<input checked="" type="radio"/>	<input checked="" type="radio"/>	<input checked="" type="radio"/>
Forces											
Set Forceplan On	<input checked="" type="checkbox"/>										
Set Sort_Merge On	<input checked="" type="checkbox"/>										
Set JTC On	<input checked="" type="checkbox"/>										
Set Table Count	<input checked="" type="checkbox"/>										
Set Table Count - 1	<input checked="" type="checkbox"/>										
Set Table Count - 2		<input checked="" type="checkbox"/>									
Set Table Count - 3				<input checked="" type="checkbox"/>							
Set Table Count - 4					<input checked="" type="checkbox"/>						
Set Table Count - 5						<input checked="" type="checkbox"/>					
Set Table Count - 6							<input checked="" type="checkbox"/>				
Set Table Count - 7								<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Set Table Count - 8									<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Parallel									<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Index					<input checked="" type="checkbox"/>						
Quota											
Syntax Transformation Quota	1	20	50	100	200	250	300	350	400	500	600
Parallel Quota	0	0	0	0	0	0	0	0	100	200	400
Forces Quota Ratio	100	10	10	10	10	15	15	20	20	30	40
Table Join Permutation Quota	0	20	50	100	200	250	300	350	400	500	600

Related Topics

[SQL Optimizer Overview](#)

[Optimization Preferences](#)

[Forces](#)

[Quotas](#)

[Abstract Plan Preferences](#)

Optimization (Optimization Tab)

The Optimization page on the Optimization tab of the Preferences window allows you to decide which SQL syntax to use for the table joins, whether to apply some the advanced SQL transformation rules, and if you would like the SQL Optimizer to generate alternatives that use temporary tables.

Temp table

Temp table generation

If selected, alternative [SQL statements that use temporary tables](#) to obtain the exact same results may be generated during the optimization process.

Apply selected forces to temp table SQL

If selected, forces are applied to the SQL statement that create and use the temporary table. This option is only available if the **Temp table generation** checkbox is selected.

Join tables (for ASE 15 or later)

Rewrite SQL using the same JOIN syntax as the original SQL

Specify that the alternative SQL statements join the tables in the FROM clause using the same SQL syntax that is used in the original SQL statement. If the original SQL statement contains both syntax types, then the optimization process will rewrite the syntax using the Ansi-92 JOIN syntax. The outer join is not including in this conversion.

Rewrite SQL using the Ansi-92 JOIN syntax

Specify to use the JOIN clause from the Ansi-92 SQL standard when generating the SQL alternatives. During the optimization, the SQL statement is converted to the Ansi-92 SQL standard and then SQL syntax transformation rules are applied to rewrite the converted SQL statement. Next, the ASE optimization forces, goals, and criteria are applied to the original SQL and the transformed SQL. So you may see SQL alternatives that use the JOIN syntax from the original SQL and not the Ansi-92 JOIN syntax, but these SQL alternatives are simply the original SQL with the ASE optimization options applied.

The outer join is not including in this conversion because Ansi-92 OUTER JOIN syntax does not always retrieve the same result set as the outer join using the (+) operator. So to avoid producing the wrong result set, the conversion of the outer join syntax cannot be applied.

For example:

```
SELECT DPT_ID
FROM EMPLOYEE
INNER JOIN DEPARTMENT
ON EMP_DEPT = DPT_ID
```

Rewrite SQL without using the Ansi-92 JOIN syntax

Specify to join tables in the FROM clause without the JOIN syntax or using comma. The join analysis occurs in the WHERE clause which specifies the column in one table that is compared to a column in another table. During the optimization, the SQL statement is converted from the Ansi-92 SQL standard and then SQL syntax transformation rules are applied to rewrite the converted SQL. Next, the ASE optimization forces, goals, and criteria are applied to the original SQL and the transformed SQL. So you may see SQL alternatives that use the JOIN syntax from the original SQL, but these SQL alternatives are simply the original SQL with the ASE optimization options applied.

The outer join is not including in this conversion because Ansi-92 OUTER JOIN syntax does not always retrieve the same result set as the outer join using the (+) operator. So to avoid producing the wrong result set, the conversion of the outer join syntax cannot be applied.

For example:

```
SELECT DPT_ID
FROM EMPLOYEE,
DEPARTMENT
WHERE DPT_ID = EMP_DEPT
```

Rewrite SQL with and without using the ANSI-92 JOIN syntax

Specify to join tables using either one of the JOIN syntax methods. The OUTER JOIN is not including in this conversion because Ansi-92 JOIN syntax is needed to define an OUTER JOIN.

Advanced SQL transformation

Enable transformation that adds COALESCE.

Specify to apply the SQL syntax transformation rule that adds COALESCE to a column. When the data is retrieved, the COALSECE function, which in this case is not actually doing anything to change the value of the column, causes a full table scan or the database to pick another index to use. For example:

```
SELECT *
```

```
FROM EMPLOYEE,  
DEPARTMENT  
WHERE COALESCE(DPT_ID, DPT_ID) = EMP_DEPT
```

Related Topics

[SQL Optimizer Overview](#)

[Intelligence Level \(Pre-ASE 15\)](#)

[Intelligence Level \(ASE 15\)](#)

[Forces \(Pre-ASE 15\)](#)

[Forces \(ASE 15\)](#)

[Criteria \(ASE 15\)](#)

[Quotas \(Pre-ASE 15\)](#)

[Quotas \(ASE 15\)](#)

[Abstract Plan Preferences](#)

Forces (Pre-ASE 15)

The Forces page on the Optimization for Pre-ASE 15 tab of the Preferences window allows you to decide if you would like to have the Adaptive Server forces applied to alternative SQL statements. By adding an Adaptive Server force, you can force the Adaptive Server's internal optimizer to choose a particular query plan.

Set

Set Forceplan On

Specify whether to enforce the tables to be joined in the order specified in the FROM clause.

Set Sort_Merge On/Off

{Adaptive Server 15 or later}

Specify whether to consider using merge joins. If selected, the parameter will toggle against the server sort_merge setting.

Set JTC On/Off

{Adaptive Server 15 or later}

Specify whether to use join transitive closure. If selected, the parameter will toggle against the server JTC setting.

Set Table Count

{Adaptive Server 15 or later}

Specify the number of tables that the optimizer considers at one time while determining table join order.

Parallel force

PARALLEL

Specify whether to enforce the use of a particular parallelism degree to access the table if parallel query is enabled in the database server.

Index force

INDEX

Specify whether to enforce the use of a particular index to access the table.

Note: The forces that are marked with an asterisk are counted in the **Number of forces selected** on the [Quota Preferences](#) page.

Related Topics

[SQL Optimizer Overview](#)

[Intelligence Level](#)

[Optimization Preferences](#)

[Quotas](#)

[Abstract Plan Preferences](#)

Quota (Pre-ASE 15)

View Optimization for Pre-ASE 15 - Quota

Syntax Transformation Quota :	<input type="text" value="200"/>	<input type="text" value="200"/>
Parallel Quota :	<input type="text" value="0"/>	<input type="text" value="0"/>
Syntax Transformation Quota + Parallel Quota :		<input type="text" value="200"/>
Forces Quota Ratio (%) :	<input type="text" value="10"/>	
Number of forces selected (forces with * only) :	<input type="text" value="7"/>	<input type="text" value="140"/>
Total Quota :		<input type="text" value="340"/>
Table Join Permutation Quota :		<input type="text" value="200"/>

The Quota page on the Optimization tab of the Preferences window allows you to restrict the number of SQL transformations produced during optimization process. The optimization quotas can only be changed when the optimizer intelligence level is set to custom.

Syntax Transformation Quota (Range = 1 to 99,999)

The maximum number of SQL statements generated by applying rules in the Feedback-Searching engine. The default quota of 100 is normally sufficient for most of the complicated SQL statements. However you can increase the quota to tackle exceptionally complicated SQL statements with very high levels of table joins or multiple levels of nested sub-queries.

This value controls the number of SQL statements that are generated during the process of rewriting the syntax of the SQL statement.

Parallel Quota (Range = 10 to 99,999)

The maximum number of SQL statements generated by applying the degree of parallelism in the Feedback-Searching engine. Only applicable if the **MAXDOP** checkbox is selected.

Forces Quota Ratio (Range = 1% to 100%)

The percentage used to calculate the maximum number of SQL statements to apply forces.

This value determines the number of SQL alternatives that are generated by applying the Forces to the original SQL and the SQL alternatives that were created by transforming the SQL syntax.

Number of forces selected (forces with * only)

A read only field indicating the number of forces selected. Only the forces that have an * after the name on the Forces page are counted. This figure is used to calculate the maximum number of SQL statements generated by applying forces, $((\text{Syntax Transformation Quota} + \text{Parallel Quota}) * \text{Forces Quota Ratio}) * \text{Number of Forces Selected} = \text{Forces Quota}$.

Total Quota

The maximum number of SQL statements generated during optimization, this figure consists of: Syntax Transformation Quota + Parallel Quota + Forces Quota.

Table Join Permutation Quota (Range = 50 to 999,999)

The maximum number of table joins rearrangements to generate a new table join access path during optimization.

Note: You should bear in mind that the higher the quota, the longer it might take to optimize a complicated SQL statement.

Related Topics

[SQL Optimizer Overview](#)

[Intelligence Level](#)

[Optimization Preferences](#)

[Forces](#)

[Abstract Plan Preferences](#)

Abstract Plan (Optimization Tab)

The Abstract Plan page on the Optimization tab of the Preferences window allows you to include the retrieval of the abstract plan for the SQL statements. This requires Adaptive Server 15 or later.

Dump abstract plan

Specify whether to retrieve the abstract plan for the SQL statement whenever the query plan is retrieved. The abstract plan displays in the SQL Optimizer window. The abstract plan is not saved on the database until you deliberately save it.

When this checkbox is selected, the selected abstract plan icon  and the abstract plan group name display on the bottom right of the main status bar.

Note: For third-party package applications, you are able to optimize SQL statements without changing the source code by saving the abstract plan.

Group name

Specify the abstract plan group name where the abstract plans are saved. The default group names in Adaptive Server are: `ap_stdout` and `ap_stdin`. These groups are usually used by the Database Administrator to enable server-wide abstract plan capturing and retrieving.

`ap_stdout` is used by default to capture an abstract plan.

`ap_stdin` is used by default to retrieve the abstract plan associated with a SQL statement during the execution of the SQL statement.

Abstract Plan Manager button

Opens the Abstract Plan Manager window to view, create, and modify abstract plan group.

Check compatibility with Original SQL

Specify whether you want to check the compatibility of the optimized SQL statements with the original SQL. When this option is selected, the AP Compatibility column displays on the SQL Run Time pane of the SQL Optimizer window and in the Batch Run Criteria window.

Note: A SQL statement may have many different "semantically equivalent" SQL statements. Each statement has an abstract plan. Even though SQL statements are "semantically equivalent", the abstract plans may not be compatible with the original SQL statement. When you check this option, the alternative abstract plans that are compatible with your original SQL are identified.

Related Topics

[SQL Optimizer Overview](#)

[Intelligence Level \(Pre-ASE 15\)](#)

[Intelligence Level \(ASE 15\)](#)

[Optimization Preferences](#)

[Forces \(Pre-ASE 15\)](#)

[Forces \(ASE 15\)](#)

[Criteria \(ASE 15\)](#)

[Quotas \(Pre-ASE 15\)](#)

[Quotas \(ASE 15\)](#)

Optimization Settings for ASE 15

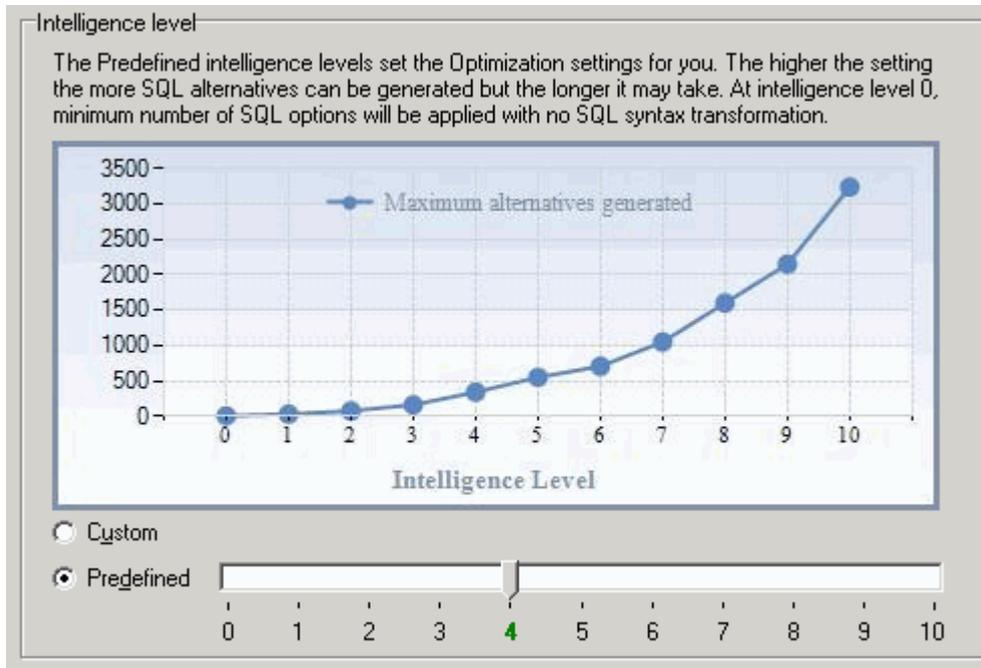
The Optimization for ASE 15 tab on the Preferences window consists of six buttons that allow users to select the level of optimization intelligence and options for optimization while using the SQL Optimizer module when connected to Adaptive Server 15.

Related Topic

[SQL Optimizer Overview](#)

Intelligence (Optimization for ASE 15 Tab)

View Optimization for ASE 15--Intelligence page



The Intelligence page on the Optimization tab of the Preferences window allows you to select your optimization intelligence settings.

Optimization intelligence settings enables you to either choose the settings used during optimization or allows you to select the predefined "Intelligence Level" settings.

Custom

Customize the settings on the Optimization, Forces/Goal, Criteria, and Quota pages.

Predefined

Use the predefined optimization intelligence levels. The items selected on the Forces and Quota pages and the **Temp table generation** option on the Optimization page change according to the level selected. Levels range from 0 to 10. The higher the level the more intelligent the SQL Optimizer is and the more likely of finding a better SQL alternative.

The Abstract Plan settings on the Abstract Plan page and the **Join table** and **Advanced SQL transformation** options on the Optimization page is adjusted independent of the optimization intelligence level.

Intelligence Level for ASE15

	Level 0	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Level 7	Level 8	Level 9	Level 10
Temp Table											
Temp table generation						<input checked="" type="checkbox"/>					
Apply Forces							<input checked="" type="checkbox"/>				
Join Tables											
Rewrite SQL using the Ansi-92 JOIN syntax											
Rewrite SQL without using the Ansi-92 JOIN syntax	<input checked="" type="checkbox"/>										
Both									<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Forces											
Set Forceplan On	<input checked="" type="checkbox"/>										
Parallel									<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Index					<input checked="" type="checkbox"/>						
Goal											
Allows_mix					<input checked="" type="checkbox"/>						
Allows_dss						<input checked="" type="checkbox"/>					
Criteria											
merge_union_distinct								<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
hash_union_distinct								<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Append_union_all								<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Merge_union_all						<input checked="" type="checkbox"/>					
group_sorted						<input checked="" type="checkbox"/>					
group_hashing						<input checked="" type="checkbox"/>					
Hash_join					<input checked="" type="checkbox"/>						
NL_join					<input checked="" type="checkbox"/>						
Merge_join					<input checked="" type="checkbox"/>						
Distinct sorted						<input checked="" type="checkbox"/>					
Distinct sorting						<input checked="" type="checkbox"/>					
Opportunistic distinct_view						<input checked="" type="checkbox"/>					
Distinct hashing						<input checked="" type="checkbox"/>					
Store index											<input checked="" type="checkbox"/>
Paralle_query					<input checked="" type="checkbox"/>						
Index_intersection					<input checked="" type="checkbox"/>						
Multi table store ind											<input checked="" type="checkbox"/>
Bushy space search											<input checked="" type="checkbox"/>
Quota											
Syntax Transformation Quota	1	20	50	80	100	150	200	300	400	500	600
Parallel Quota	0	0	0	0	0	0	0	0	100	200	400
Forces Quota Ratio	100	10	10	10	20	25	35	40	50	60	80
Table Join Permutation Quota	0	20	50	100	200	250	300	350	400	450	500

Related Topics

[SQL Optimizer Overview](#)

[Intelligence Level](#)

[Optimization Preferences](#)

[Forces](#)

[Criteria](#)

[Quotas](#)

[Abstract Plan Preferences](#)

Optimization (Optimization Tab)

The Optimization page on the Optimization tab of the Preferences window allows you to decide which SQL syntax to use for the table joins, whether to apply some the advanced SQL transformation rules, and if you would like the SQL Optimizer to generate alternatives that use temporary tables.

Temp table

Temp table generation

If selected, alternative [SQL statements that use temporary tables](#) to obtain the exact same results may be generated during the optimization process.

Apply selected forces to temp table SQL

If selected, forces are applied to the SQL statement that create and use the temporary table. This option is only available if the **Temp table generation** checkbox is selected.

Join tables (for ASE 15 or later)

Rewrite SQL using the same JOIN syntax as the original SQL

Specify that the alternative SQL statements join the tables in the FROM clause using the same SQL syntax that is used in the original SQL statement. If the original SQL statement contains both syntax types, then the optimization process will rewrite the syntax using the Ansi-92 JOIN syntax. The outer join is not including in this conversion.

Rewrite SQL using the Ansi-92 JOIN syntax

Specify to use the JOIN clause from the Ansi-92 SQL standard when generating the SQL alternatives. During the optimization, the SQL statement is converted to the Ansi-92 SQL standard and then SQL syntax transformation rules are applied to rewrite the converted SQL statement. Next, the ASE optimization forces, goals, and criteria are applied to the original SQL and the transformed SQL. So you may see SQL alternatives that use the JOIN syntax from the original SQL and not the Ansi-92 JOIN syntax, but these SQL alternatives are simply the original SQL with the ASE optimization options applied.

The outer join is not including in this conversion because Ansi-92 OUTER JOIN syntax does not always retrieve the same result set as the outer join using the (+) operator. So to avoid producing the wrong result set, the conversion of the outer join syntax cannot be applied.

For example:

```
SELECT DPT_ID
FROM EMPLOYEE
INNER JOIN DEPARTMENT
ON EMP_DEPT = DPT_ID
```

Rewrite SQL without using the Ansi-92 JOIN syntax

Specify to join tables in the FROM clause without the JOIN syntax or using comma. The join analysis occurs in the WHERE clause which specifies the column in one table that is compared to a column in another table. During the optimization, the SQL statement is converted from the Ansi-92 SQL standard and then SQL syntax transformation rules are applied to rewrite the converted SQL. Next, the ASE optimization forces, goals, and criteria are applied to the original SQL and the transformed SQL. So you may see SQL alternatives that use the JOIN syntax from the original SQL, but these SQL alternatives are simply the original SQL with the ASE optimization options applied.

The outer join is not including in this conversion because Ansi-92 OUTER JOIN syntax does not always retrieve the same result set as the outer join using the (+) operator. So to avoid producing the wrong result set, the conversion of the outer join syntax cannot be applied.

For example:

```
SELECT DPT_ID
FROM EMPLOYEE,
DEPARTMENT
WHERE DPT_ID = EMP_DEPT
```

Rewrite SQL with and without using the ANSI-92 JOIN syntax

Specify to join tables using either one of the JOIN syntax methods. The OUTER JOIN is not including in this conversion because Ansi-92 JOIN syntax is needed to define an OUTER JOIN.

Advanced SQL transformation

Enable transformation that adds COALESCE.

Specify to apply the SQL syntax transformation rule that adds COALESCE to a column. When the data is retrieved, the COALSECE function, which in this case is not actually doing anything to change the value of the column, causes a full table scan or the database to pick another index to use. For example:

```
SELECT *
FROM EMPLOYEE,
DEPARTMENT
WHERE COALESCE(DPT_ID, DPT_ID) = EMP_DEPT
```

Related Topics

[SQL Optimizer Overview](#)

[Intelligence Level \(Pre-ASE 15\)](#)

[Intelligence Level \(ASE 15\)](#)

[Forces \(Pre-ASE 15\)](#)

[Forces \(ASE 15\)](#)

[Criteria \(ASE 15\)](#)

[Quotas \(Pre-ASE 15\)](#)

[Quotas \(ASE 15\)](#)

[Abstract Plan Preferences](#)

Forces/Goal (ASE 15)

[View Optimization for ASE 15 - Forces/Goals page](#)



Set

Set Forceplan On *

Parallel force

PARALLEL

Index force

INDEX

Optimization Goal

allows_mix *

allows_dss *

allows_oltp *

fastfirstrow *

The Forces/Goal page on the Optimization for ASE 15 tab of the Preferences window allows you to decide if you would like to have the Adaptive Server forces applied to alternative SQL statements. By adding an Adaptive Server force, you can force the Adaptive Server's internal optimizer to choose a particular query plan.

Set

Set Forceplan On

Specify whether to enforce the tables to be joined in the order specified in the FROM clause.

Parallel force

PARALLEL

Specify whether to enforce the use of a particular parallelism degree to access the table if parallel query is enabled in the database server.

Index force

INDEX

Specify whether to enforce the use of a particular index to access the table.

Optimization Goal

allows_mix

Specify to set the optimization goal for a database environment that is a mixture of online transactions and database warehouse transactions.

allows_dss

Specify to set the optimization goal for a database environment that is purely for database warehouse transactions.

allows_oltp

Specify to set the optimization goal for a database environment that is purely for oltp (online transaction processing) transactions.

fastfirstrow

Specify to set the optimization goal to retrieve the first row as fast as possible.

Note: The forces that are marked with an asterisk are counted in the **Number of forces selected** on the [Quota Preferences](#) page.

Related Topics

[SQL Optimizer Overview](#)

[Intelligence Level](#)

[Optimization Preferences](#)

[Criteria](#)

[Quotas](#)

[Abstract Plan Preferences](#)

Criteria

The Optimization Criteria are session level settings. Each criteria represents a specific algorithm or relational techniques that the Adaptive Server Optimizer may choose to use when it is retrieving the query plan. They are used to fine-tune the Adaptive Server optimizer to provide the best performance for your database environment.

Although you may specify that Adaptive Server optimizer use a specific algorithm or relational technique, the optimizer may not choose what you have specified.

Join

Optimization Criteria	Description
hash_join	Specify that the Adaptive Server optimizer uses the hash join algorithm. This is valuable when a large number of rows satisfy the join condition or when the joining columns do not have useful indexes. However, the hash join algorithm may consume more run time resources than other join algorithms.
nl_join	Specify that the Adaptive Server optimizer uses the nested-loop-join algorithm.
merge_join	Specify that the Adaptive Server optimizer uses the merge join algorithm. This relies on ordered input and is most valuable when the input is ordered on the merge key.

Union Distinct

Optimization Criteria	Description
merge_union_distinct	Specify that the Adaptive Server optimizer uses the merge algorithm for the UNION ALL. It is similar to merge_union_all except that duplicate rows are eliminated.

hash_union_distinct	Specify that the Adaptive Server optimizer uses the hash distinct algorithm. This will be inefficient if most of the rows are distinct.
---------------------	---

Union All

Optimization Criteria	Description
append_union_all	Specify that the Adaptive Server optimizer uses the append union all algorithm.
merge_union_all	Specify that the Adaptive Server optimizer uses the merge algorithm for the UNION ALL. This means that it will maintain the ordering of the result row from the union input.

Group By

Optimization Criteria	Description
group_sorted	Specify that the Adaptive Server optimizer uses an on-the-fly algorithm. This algorithm relies on an input stream sorted on the grouping column and preserves this order in its output.
group_hashing	Specify that the Adaptive Server optimizer uses the use a group hashing algorithm to process aggregates.

Distinct

Optimization Criteria	Description
distinct_sorted	Specify that the Adaptive Server optimizer uses a single-pass algorithm to eliminate duplicate rows. This algorithm relies on an ordered input stream.
distinct_sorting	Specify that the Adaptive Server optimizer uses the sorting algorithm to eliminate duplicate rows. This algorithm is useful when the input is not ordered.
opportunistic_distinct_view	Specify that the Adaptive Server optimizer uses a more flexible algorithm when enforcing distinctness.
distinct_hashing	Specify that the Adaptive Server optimizer uses the hashing algorithm to eliminate duplicates, which is very efficient when there are only a few distinct values in comparison to the number of rows.

Individual

Optimization Criteria	Description
store_index	Specify that the Adaptive Server optimizer uses the reformatting, which may cause an increase in the number of worktables used.
parallel_query	Specify that the Adaptive Server optimizer uses parallel query optimization.
index_intersection	Specify that the Adaptive Server optimizer uses the intersection of multiple index scans as part of the query plan in search space.
multi_table_store_ind	Specify that the Adaptive Server optimizer uses reformatting on the result of a multiple table join.
bushy_space_search	Specify that the Adaptive Server optimizer uses bushy-tree-shaped query plans, which may cause an increase in the search space, but may provide more query plan options to improve performance.
advanced_aggregation	Specify that the Adaptive Server optimizer uses eager aggregation.
replicated_partition	Specify that the Adaptive Server optimizer uses replicated partitioning when multiple scans of the same table in different threads can improve the performance of nested loop joins.
group_inserting	Specify that the Adaptive Server optimizer uses the group by aggregation algorithm. This algorithm generates a clustered index work table on the grouping columns and inserts rows into the table in order to evaluate the aggregate.

Note: The criteria that are marked with an asterisk are counted in the **Number of forces selected** on the [Quota Preferences](#) page.

Optimization Time and Parallel	Description
plan_optimeoutlimit	Specify that the Adaptive Server optimizer uses the optimization timeout limit to restrict the amount of time it takes to optimize a query. Note: The default value is 60 percent. You can specify a value from 0 to 1000.
resource_granularity	Specify that the Adaptive Server optimizer uses max resource granularity to set the amount of total memory allocated to a single query.
repartition_degree	Specify that the Adaptive Server optimizer uses repartition degree to suggest the maximum number of worker processes the query processor uses to partition a data stream.

Related Topics

[SQL Optimizer Overview](#)

[Intelligence Level](#)

Quota (ASE 15)

View Optimization for ASE 15 - Quota page

Syntax Transformation Quota :	200	200
Parallel Quota :	0	0
Syntax Transformation Quota + Parallel Quota :		200
Forces Quota Ratio (%) :	10	
Number of forces selected (forces with * only) :	7	140
Total Quota :		340
Table Join Permutation Quota :	200	

The Quota page on the Optimization tab of the Preferences window allows you to restrict the number of SQL transformations produced during optimization process. The optimization quotas can only be changed when the optimizer intelligence level is set to custom.

Syntax Transformation Quota (Range = 1 to 99,999)

The maximum number of SQL statements generated by applying rules in the Feedback-Searching engine. The default quota of 100 is normally sufficient for most of the complicated SQL statements. However you can increase the quota to tackle exceptionally complicated SQL statements with very high levels of table joins or multiple levels of nested sub-queries.

This value controls the number of SQL statements that are generated during the process of rewriting the syntax of the SQL statement.

Parallel Quota (Range = 10 to 99,999)

The maximum number of SQL statements generated by applying the degree of parallelism in the Feedback-Searching engine. Only applicable if the **MAXDOP** checkbox is selected.

Forces Quota Ratio (Range = 1% to 100%)

The percentage used to calculate the maximum number of SQL statements to apply forces.

This value determines the number of SQL alternatives that are generated by applying the Forces to the original SQL and the SQL alternatives that were created by transforming the SQL syntax.

Number of forces selected (forces with * only)

A read only field indicating the number of forces, goals, and criteria selected. Only the items that have an * after the name on the Forces/Goals and the Criteria pages are counted. This figure is used to calculate the maximum number of SQL statements generated by applying these selections, $((\text{Syntax Transformation Quota} + \text{Parallel Quota}) * \text{Forces Quota Ratio\%}) * \text{Number of Forces Selected} = \text{Forces Quota}$.

Total Quota

The maximum number of SQL statements generated during optimization, this figure consists of: Syntax Transformation Quota + Parallel Quota + Forces Quota.

Table Join Permutation Quota (Range = 50 to 999,999)

The maximum number of table joins rearrangements to generate a new table join access path during optimization.

Note: You should bear in mind that the higher the quota, the longer it might take to optimize a complicated SQL statement.

Related Topics

[SQL Optimizer Overview](#)

[Intelligence Level](#)

[Optimization Preferences](#)

[Forces](#)

[Criteria](#)

[Quotas](#)

[Abstract Plan Preferences](#)

Abstract Plan (Optimization Tab)

The Abstract Plan page on the Optimization tab of the Preferences window allows you to include the retrieval of the abstract plan for the SQL statements. This requires Adaptive Server 15 or later.

Dump abstract plan

Specify whether to retrieve the abstract plan for the SQL statement whenever the query plan is retrieved. The abstract plan displays in the SQL Optimizer window. The abstract plan is not saved on the database until you deliberately save it.

When this checkbox is selected, the selected abstract plan icon  and the abstract plan group name display on the bottom right of the main status bar.

Note: For third-party package applications, you are able to optimize SQL statements without changing the source code by saving the abstract plan.

Group name

Specify the abstract plan group name where the abstract plans are saved. The default group names in Adaptive Server are: `ap_stdout` and `ap_stdin`. These groups are usually used by the Database Administrator to enable server-wide abstract plan capturing and retrieving.

`ap_stdout` is used by default to capture an abstract plan.

`ap_stdin` is used by default to retrieve the abstract plan associated with a SQL statement during the execution of the SQL statement.

Abstract Plan Manager button

Opens the Abstract Plan Manager window to view, create, and modify abstract plan group.

Check compatibility with Original SQL

Specify whether you want to check the compatibility of the optimized SQL statements with the original SQL. When this option is selected, the AP Compatibility column displays on the SQL Run Time pane of the SQL Optimizer window and in the Batch Run Criteria window.

Note: A SQL statement may have many different "semantically equivalent" SQL statements. Each statement has an abstract plan. Even though SQL statements are "semantically equivalent", the abstract plans may not be compatible with the original SQL statement. When you check this option, the alternative abstract plans that are compatible with your original SQL are identified.

Related Topics

[SQL Optimizer Overview](#)

[Intelligence Level \(Pre-ASE 15\)](#)

[Intelligence Level \(ASE 15\)](#)

[Optimization Preferences](#)

[Forces \(Pre-ASE 15\)](#)

[Forces \(ASE 15\)](#)

[Criteria \(ASE 15\)](#)

[Quotas \(Pre-ASE 15\)](#)

[Quotas \(ASE 15\)](#)

Index Advisor

[View Index Advisor page](#)

Option

Default prefix of index name :

Calculate selectivity of indexes using a default sampling size of :

Top percentage of rows from table:

Minimum number of rows:

Maximum number of rows:

Do not advise indexes for tables with row count less than:

Evaluate columns in SELECT list

Maximum number of columns in a composite index:

Maximum number of indexes to advise per table usage:

Default prefix of index name: (Default: *QUEST_SX_IDX*)

Enter the prefix that is placed on the index name when the Index Advisor automatically generates index candidates.

Calculate selectivity of indexes using a default sampling size of:

The selectivity is calculated from the number of rows that are selected from the table. The number of rows selected is determined by the following settings:

Top percentage of rows from table (Default = 10)

This number is used as a percentage to calculate the number of rows in the table that are retrieved in the sample which determines the selectivity of the data. You see the number of row used in the sample in the Select tables window.

Minimum number of rows (Default = 1000)

This is the minimum number of rows that are used in the sample which determines the selectivity of the data. This number is used if the calculation using the percentage of the table is below this value.

Maximum number of rows (Default = 10000)

This is the maximum number of rows that are used in the sample which determines the selectivity of the data. This number is used if the calculation using the percentage of the table is above this value.

Do not advise indexes for tables with row count less than (Default = 500)

Specify the number of rows that a table must have before index candidates are generated for any columns in that table.

Evaluate columns in SELECT list (Default = *cleared*)

For SELECT SQL statements, specify whether the columns in the SELECT list are considered when generating the alternative indexes.

Maximum number of columns in a composite index (Default = 4)

Specify the maximum number of columns that are placed in a composite index.

Maximum number of indexes to advise per table usage (Default = 3)

Specify the maximum number of indexes that the Index Advisor creates for each table in the SQL statement.

Related Topic

[Index Advisor Overview](#)

Directory Setup

The **Directory Setup** tab of the **Preferences** window allows you to define the directory for saving and opening files used in the different modules.

[Directory \(1\)](#)

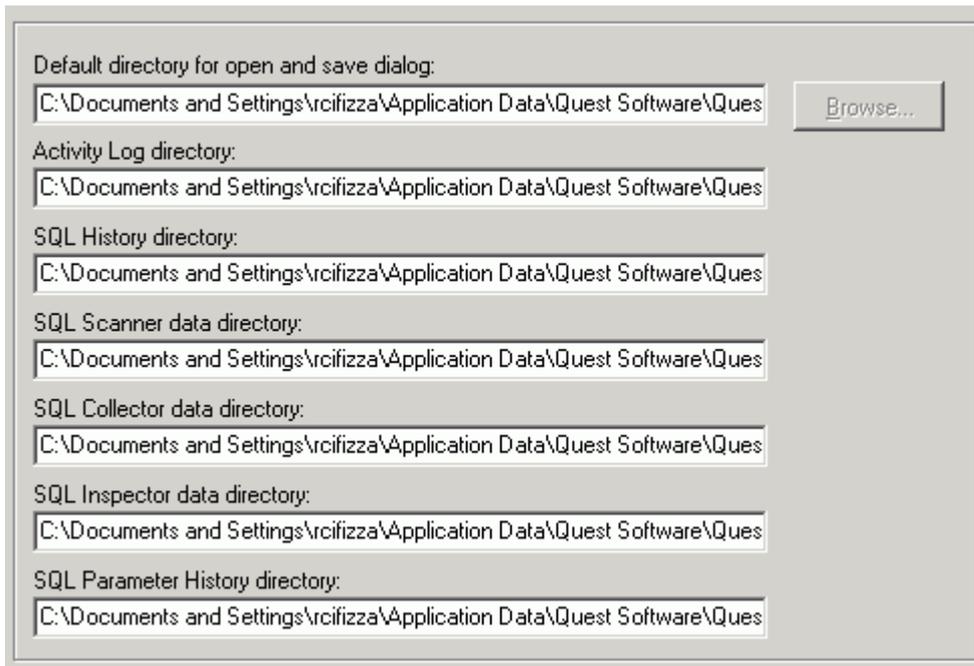
Set up directories for the Activity Log, the SQL History, the SQL Scanner, the SQL Collector for Monitor Server, and SQL Inspector modules. Also, set-up the directory for opening and saving files in several modules.

[Directory \(2\)](#)

Set up the directories for all the Analyzer modules and SQL Repository.

Directory (1)

[View Directory \(1\) page](#)



Default directory for open and save dialog

This directory is the default for opening and saving files with the Open or Save commands from various locations through the modules. By default, it is C:\Documents and Settings\User\Application Data\Quest Software\SQL Optimizer.

Activity Log Directory

This directory specifies the location where the Activity Log file is created. By default, it is C:\Documents and Settings\User\Application Data\Quest Software\SQL Optimizer.

SQL History directory

This directory is used to store the data file that stores the SQL history information. The default is C:\Documents and Settings\User\Application Data\Quest Software\Quest SQL Optimizer\SQL_History.

SQL Scanner data directory

This directory is used to store the data files created while scanning. The default is the sub-directory DATA of the installed directory, for example: C:\Documents and Settings\User\Application Data\Quest Software\Quest SQL Optimizer\DATA. Changes to this directory cannot be made while SQL Scanner is active.

SQL Collector data directory

This directory is used to store the data files created while monitoring. The default is the sub-directory DATA of the installed directory, for example: C:\Documents and Settings\User\Application Data\Quest Software\Quest SQL Optimizer\DATA. Changes to this directory cannot be made while SQL Collector for Monitor Server is active.

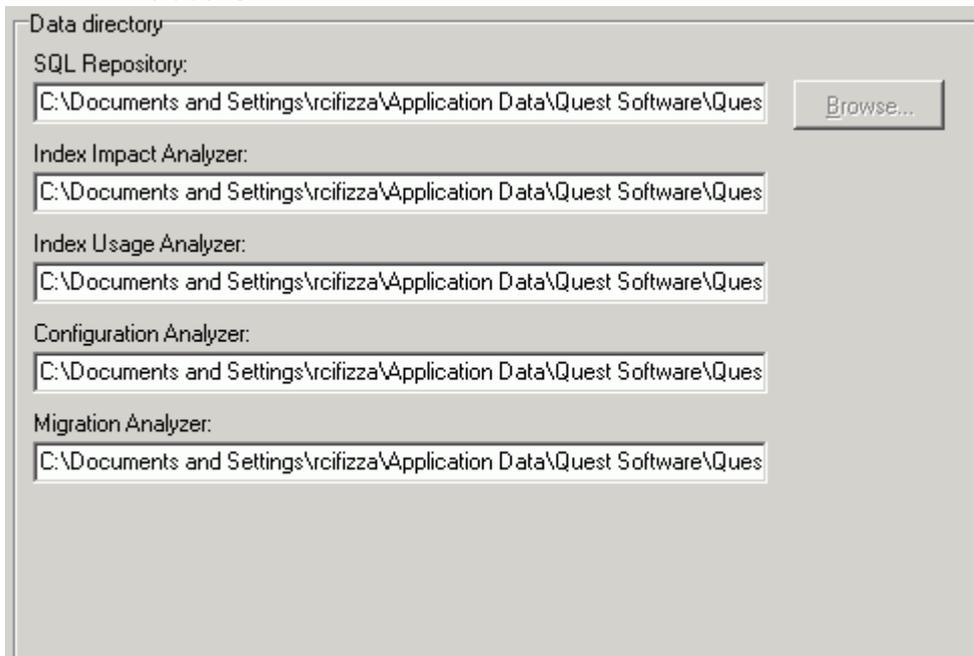
SQL Inspector data directory

This directory is used to store the data files created while capturing SQL from the system tables. The default is the sub-directory DATA of the installed directory, for example: C:\Documents and Settings\User\Application Data\Quest Software\SQL Optimizer\DATA. Changes to this directory cannot be made while SQL Inspector is active.

Note: It is advisable not to change the data directory after selection, as files already created are kept in the original directory and are not moved to the new directory.

Directory (2)

View Directory (2) page



The screenshot shows a dialog box titled "Data directory" with the following fields:

- SQL Repository: C:\Documents and Settings\rcifizza\Application Data\Quest Software\Ques
- Index Impact Analyzer: C:\Documents and Settings\rcifizza\Application Data\Quest Software\Ques
- Index Usage Analyzer: C:\Documents and Settings\rcifizza\Application Data\Quest Software\Ques
- Configuration Analyzer: C:\Documents and Settings\rcifizza\Application Data\Quest Software\Ques
- Migration Analyzer: C:\Documents and Settings\rcifizza\Application Data\Quest Software\Ques

A "Browse..." button is located to the right of the first text box.

SQL Repository data directory

By default, it is C:\Documents and Settings\User\Application Data\Software\Quest Software\SQL Optimizer.

Index Impact Analyzer data directory

By default, it is C:\Documents and Settings\User\Application Data\Software\Quest Software\SQL Optimizer.

Index Usage Analyzer data directory

By default, it is C:\Documents and Settings\User\Application Data\Software\Quest Software\SQL Optimizer.

Configuration Analyzer data directory

By default, it is C:\Documents and Settings\User\Application Data\Software\Quest Software\SQL Optimizer.

Migration Analyzer data directory

By default, it is C:\Documents and Settings\User\Application Data\Software\Quest Software\SQL Optimizer.

Activity Log

View Abstract Plan Manager Window

Activities to be logged

- SQL optimization
- Query plan generation

Information to be logged

- SQL text
- Query plan
- Abstract plan

Housekeeping

Show warning message when log file size exceeds (Mbytes) : 5

Purge activity log

- Dated from (DD-MM-YYYY) [] to 11-08-2005
- Whole log

Purge Now

Activity to be logged

The activity log records activities obtained from SQL optimization and query plan retrieval. If neither the **SQL optimization** nor **Query plan generation** checkbox is selected, then no activities are recorded in the activity log. By default, both of these checkboxes are unselected.

Information to be logged

To record the SQL text, the query plan information and the abstract plan information, select the desired checkboxes. By default the following information is recorded automatically: login user, OS user, database, all records and/or first record time, and SQL type.

Housekeeping

Show warning message when log file size exceeds (Default 5MB, Range = 1 to 500MB)

Use to indicate the maximum size in MB of the activity log file; the default is 5 MB. If the size of the activity log file exceeds the maximum value a warning message displays to inform you of this.

Purge activity log

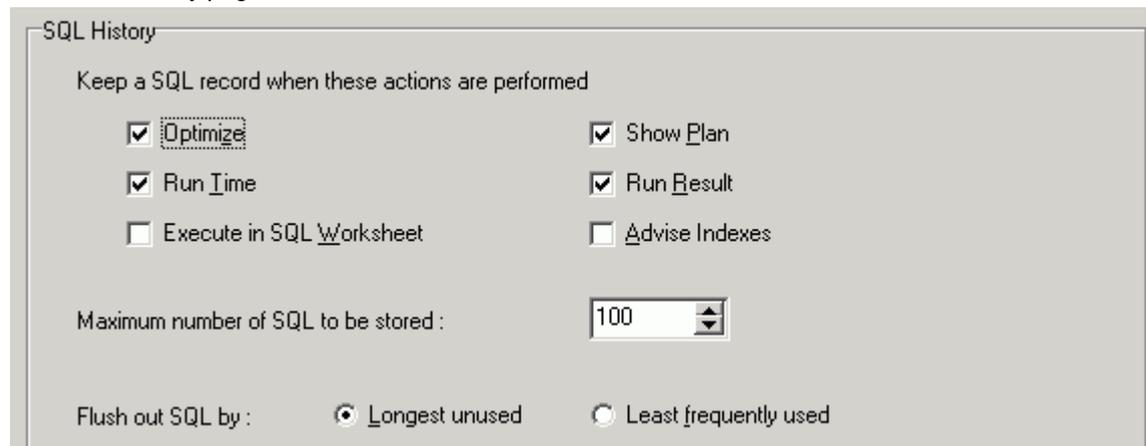
Use to remove information from the activity log. Select either **Whole Log** to remove all information or **Dated from** and specify a date range to remove logs between these dates. Click **Purge Now** to remove the desired logs.

Related Topic

[Activity Log Overview](#)

SQL History

View SQL History page



SQL History

Keep a SQL record when these actions are performed

<input checked="" type="checkbox"/> Optimize	<input checked="" type="checkbox"/> Show Plan
<input checked="" type="checkbox"/> Run Time	<input checked="" type="checkbox"/> Run Result
<input type="checkbox"/> Execute in SQL Worksheet	<input type="checkbox"/> Advise Indexes

Maximum number of SQL to be stored :

Flush out SQL by : Longest unused Least frequently used

The SQL History tab of the Preferences window allows users to specify which functions save the SQL statements.

Keep a SQL record when these actions are performed

Optimize (Default = *checked*)

Save the SQL statement every time it is optimized in the SQL Optimizer module.

Run Time (Default = *checked*)

Save the SQL statement when the run time is retrieved using the Run Time function.

Execute in SQL Worksheet (Default = *cleared*)

Save the SQL statement when it is executed in the SQL Worksheet.

Show Plan (Default = *checked*)

Save the SQL statement every time the query plan is retrieved using the Show Plan function.

Run Result (Default = *checked*)

Save the SQL statement when it is executed using the Run Result function.

Advise Indexes (Default = *cleared*)

Save the SQL statement every time indexes are generated for it the Index Advisor module.

Maximum number of SQL to be stored (Default = 100)

Specify the maximum number of SQL statements that are stored in the SQL History.

Flush out SQL by

When the number of SQL statement in the SQL History reaches its maximum, the SQL statements remove SQL statements based on one of the following criteria:

Longest unused (Default)

Removes the SQL statement with the oldest date and time in the **Last Used Datetime** column.

Least frequently used

Removes the SQL statement with the lowest value in the **Times Used** column.

Related Topic

[SQL History Overview](#)

Plan

View Plan page

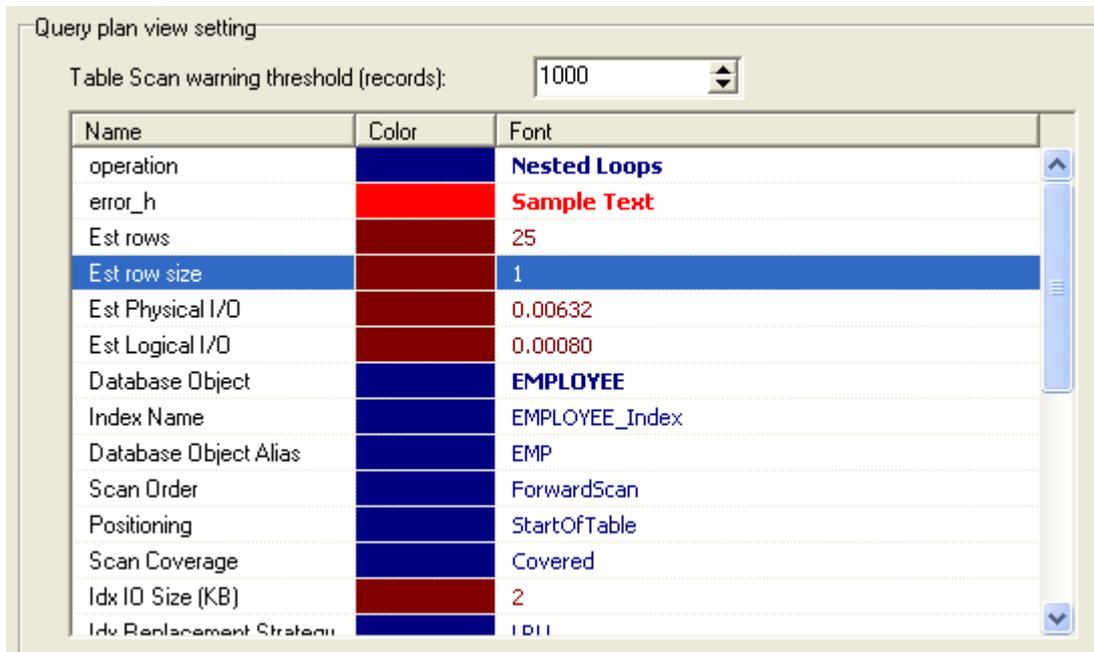


Table Scan warning threshold (records)

Enter a threshold warning value for table scans between 1 and 2147483647. If the estimated rows for a table scan is larger than this value, the TableScan icon turns red. Otherwise, the TableScan icon is green.

Color

Specify the color of the individual items in the query plan by clicking the **Color** column in the row for the item and selecting the new color from the dialog.

Font

Specify the font settings of the individual items in the query plan by clicking the **Font** column in the row for the item and selecting the new settings from the dialog.

Note: To select which elements of the query plan that display when you are viewing the query plan, right-click and select **Plan Options**.

Related Topic

[Query Plan Overview](#)

SQL Worksheet

View SQL Worksheet page

The screenshot shows a preferences dialog box for the SQL Worksheet. It is divided into three sections: Execution, Result buffer size, and Display results. In the Execution section, the 'Clear commands upon execution' checkbox is checked, and the 'Batch separator' is set to 'GO'. In the Result buffer size section, the 'Unlimited' radio button is selected, and the 'Fixed size (records)' is set to 1000. In the Display results section, the 'Maximum characters per column' is set to 256, the 'Discard results' checkbox is unchecked, and the 'Display results in' radio buttons are set to 'Text'. The 'Output format' is set to 'Column aligned', the 'Right aligned numerics' checkbox is checked, the 'Delimiter' is empty, and the 'Display headers' checkbox is checked.

The SQL Worksheet tab of the Preferences window allows users to determine if the commands are clear upon execution, the batch separator, the buffer size for the result set, and to specify how the results sets are displayed.

Execution

Clear commands upon execution (Default = *checked*)

Specify to clear the SQL statement or Transact-SQL code in the Editor pane after the command is successfully executed. If the execution is not successful, then the text remains in the Editor pane.

If this option is not selected, the text also remains in the Editor pane.

Note: After a successful execution, the text is always saved in the Command drop-down field at the top of the window.

Batch Separator (Default: GO)

Specify the separator used to signal the end of a batch of Transact-SQL statements.

Result buffer size

Unlimited or Fixed size (records)

Select **Unlimited** to return all records. Select **Fixed size** and then in the text field, set the maximum number of records to be returned. (Default fixed size: 1000)

Display results

Maximum characters per column: (Default: 256)

Specify how wide each column should be in the *Resultn* tab in the lower pane of the SQL Worksheet window.

Discard results: (Default: *cleared*)

Specify whether to show the results on the *Resultn* tab, or to just show the number of affected records on the Message Log tab.

Display results in

Grid or Text: Select between displaying the results in a grid or in plain text.

Under the **Text** option you can also select:

Output format (Default: *Column aligned*)

Select between Column aligned, Comma separated (CSV), Tab delimited, or Other delimiter.

Right aligned numerics: (Default: *cleared*)

When you select Column aligned for the Output format, specify whether or not to right align number columns.

Delimiter

When you select Other delimiter for the Output format, specify the character to be used as the delimiter between the columns.

Display headers:

Specify whether or not the column names are displayed.

Related Topic

[SQL Workshop Overview](#)

SQL Scanner

The SQL Scanner settings are used to define the requirements for the SQL Scanner module.

[SQL Scanner Options Settings](#)

[SQL Scanner Abstract Plan Settings](#)

[SQL Scanner General Settings](#)

The settings in the SQL Classification tab are used by the SQL Scanner to help you identify which SQL statements are likely to be causing performance problems by classifying the SQL as Problematic, Complex, or Simple.

[SQL Classification Settings](#)

Related Topic

[SQL Scanner Overview](#)

Options (SQL Scanner Tab)

View SQL Scanner - Options page

Scanning options

- Skip SQL within comments
- Skip SQL that does not involve tables
- Whole word matching for the first SQL keyword
- Always use current database and user
- Automatically switch to current database and user when unable to set user to dbo
- Create Scanner Temp Table
 - Include data
 - Override previous Scanner Temp Table
 - Override User-Defined Temp Table

Number of characters to be skipped at the beginning of every line for all files:

Skip end of line continuation character:

Skip SQL within comments (Default = *cleared*)

Specify whether the scanning algorithm will ignore any SQL statements within comments enclosed by `/* */`, `//` and `--` found in the source code. By default, the scanning algorithm will search for any SQL statements contained in comments.

Skip SQL that does not involve tables (Default = *cleared*)

Specify whether the scanning algorithm will ignore any SQL statements that do not involve tables.

Whole word matching for the first SQL keyword (Default = *cleared*)

Specify to search for SELECT, INSERT, UPDATE or DELETE as a whole word, the keyword must be preceded and followed by a space. The Scanner therefore will not find something like PROCEDUREINSERT and attempt to build a SQL statement from it.

Always use current database and user (Default = *cleared*)

Specify to always use the current database and user when the SQL Scanner is retrieving the query plan.

Automatically switch to current database and user when unable to set user to dbo

Specify to allow the SQL Scanner to switch the database and user for a job to the current database and current user when the login is unable to 'setuser dbo' during scanning. You will notice that the Job name is changed

from "[xx] [dbo] yyy" to "[current db][current user] yyy" when the 'setuser dbo' fails. Also, in the information pane in the SQL Scanner window under "Connection Information," this line is added "Scanned using the current database and user setting".

When this option is not selected and the login is unable to 'set user to dbo', the SQL Scanner displays the 'cannot set dbo' status message in the Status column and it skips the Job.

Create Scanner Temp Table (Default = *checked*)

If selected, the scanning algorithm automatically creates a temporary table during scanning when a CREATE TABLE #TEMP or a SELECT INTO #TEMP statement is found. Tables created during the scanning process are dropped at the end of the process.

Include data (Default = *cleared*)

If selected, the SQL Scanner automatically executes any INSERT, DELETE, UPDATE, and SELECT INTO SQL statement that modifies a temporary table that is created by the SQL Scanner.

If this option is not selected, then a WHERE 0 =1 clause is added to the SELECT INTO statement so that no data is included when the temporary table is created.

Note: Selecting the Include data option may affect the total scanning time.

Override previous Scanner Temp Table (Default = *checked*)

If this option is selected, a temporary table is created with the first 'CREATE TABLE #TEMP' statement. If another 'CREATE TABLE #TEMP' with the same name is found, it will drop the current table and create a new one for every time if find a new 'CREATE TABLE #TEMP'.

This option will not override any temporary tables created in **User-Defined Temp Table** window.

Override User-Defined Temp Table (Default = *cleared*)

If selected, a previous User-Defined Temp Table is overwritten if the SQL Scanner finds another create temporary table statement using the same table name. This option will not override any temporary tables created in the SQL Scanner.

Number of characters to be skipped at the beginning of every line for all files (Default = 0)

When a file is scanned, the SQL Scanner skips the number of characters specified at the beginning of every line.

Skip end of line continuation character (Default = *<do not use>*)

If the text of the SQL statement is on more than one physical line and there is a continuation character at the end of each physical line of the SQL text, then specify what the continuation character is so that it will be skipped. If this character is not skipped, it may cause the SQL Scanner to miss part of the SQL statement. Three options are included: **<Do not use>**, / (forward slash character), and _ (underscore character). In addition, you may add your own character. This character will be saved in the field as long as it is the selected character. Once you make another selection, your own character will not be remembered.

Related Topics

[SQL Scanner Overview](#)

[Example of Temporary Tables in SQL Scanner](#)

[Options \(SQL Scanner Tab\)](#)

[Abstract Plan \(SQL Scanner Tab\)](#)

[General \(SQL Scanner Tab\)](#)

Abstract Plan (SQL Scanner Tab)

The Abstract Plan page on the SQL Scanner tab of the Preferences window allows you to include the retrieval of the abstract plan for the SQL statements. This requires Adaptive Server 15 or later.

Dump abstract plan (Default = *cleared*)

Specify whether to retrieve the abstract plan for the SQL statement whenever the query plan is retrieved. The abstract plan displays in the SQL Scanner window. The abstract plan is not saved on the database.

When this checkbox is selected, the selected abstract plan icon  and the abstract plan group name display on the bottom right of the main status bar.

Note: For third-party package applications, you are able to optimize SQL statements without changing the source code by saving the abstract plan.

Group name (Default = *blank*)

Specify the abstract plan group name where the abstract plans are saved. The default groups names in Adaptive Server are: `ap_stdout` and `ap_stdin`. These groups are usually used by the Database Administrator to enable server-wide abstract plan capturing and retrieving.

`ap_stdout` is used by default to capture an abstract plan.

`ap_stdin` is used by default to retrieve the abstract plan associated with a SQL statement during the execution of the SQL statement.

Abstract Plan Manager button

Opens the Abstract Plan Manager window to view, create, and modify abstract plan group.

Related Topics

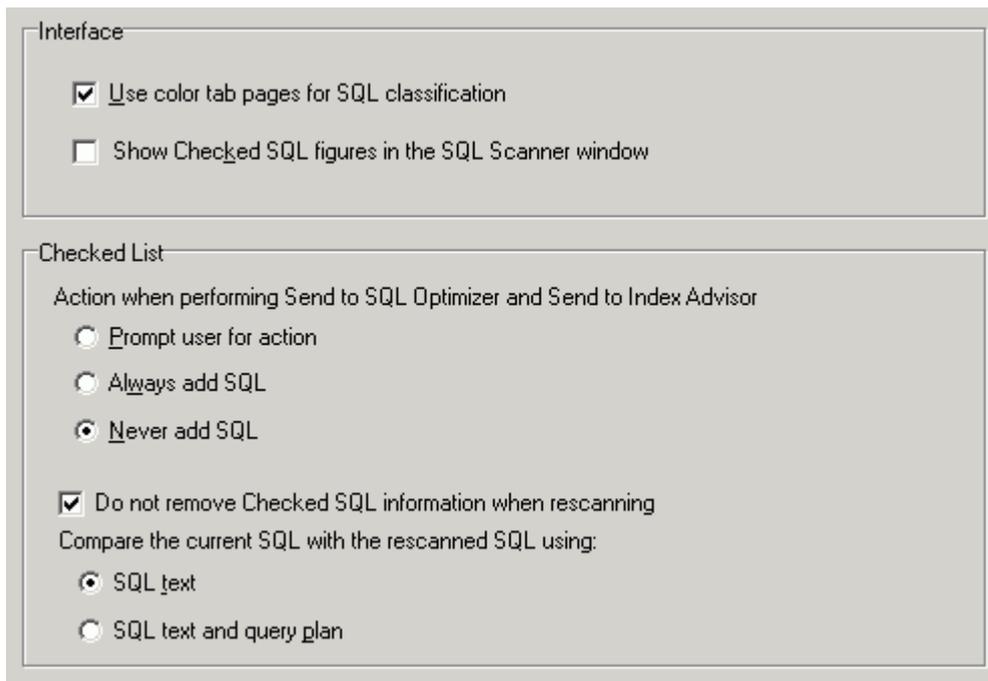
[SQL Scanner Overview](#)

[Options \(SQL Scanner Tab\)](#)

[General \(SQL Scanner Tab\)](#)

General (SQL Scanner Tab)

[View SQL Scanner - General page](#)



Interface

Use color tabs for SQL classification (Default = *checked*)

When checked, the SQL statements are displayed with color-coded tabs representing the SQL's classification. Red for Problematic, purple for Complex, green for Simple, and blue for Invalid. If the SQL statement is checked, the tab is grey.

Show Checked SQL figures on the SQL Scanner window (Default = *cleared*)

When checked, an additional column of numbers is added to the SQL classification columns to indicate how many SQL statements have been checked for each Job. When a SQL statement is added to the Checked SQL list, it has a value in the appropriate SQL classification column in the SQL Scanner window.

Checked List

Action when performing Send to SQL Optimizer and Send to Index Advisor

Prompt user for action

If selected, the user is prompted to be added to the Checked List.

Always add SQL

If selected, all SQL sent is added to the Checked List.

Never add SQL (Default)

If selected, all SQL sent is not automatically added to the Checked List.

Do not remove Check SQL information when rescanning (Default = *cleared*)

Specify to keep the checked SQL information when the Job is rescanned. If unchecked, the checked SQL information is removed when the Job is scanned again.

Compare the current SQL with the rescanned SQL using

SQL text (Default)

Specify to match the SQL statements from the previous time the Job was scanned with the current scanning using only the SQL text as a comparison for the match. The checked SQL information is preserved for those SQL statements where the SQL text matches.

SQL text and query plan

Specify to match the SQL statements from the previous time the Job was scanned with the current scanning using the SQL text and the query plan as a comparison for the match. The checked SQL information is only preserved for those SQL statements where the query plan and the SQL text match.

Related Topics

[SQL Scanner Overview](#)

[Options \(SQL Scanner Tab\)](#)

[Abstract Plan \(SQL Scanner Tab\)](#)

SQL Classification

Based on the query plan, a SQL statement is classified according to the characteristics of the query plan. If the query plan is not retrieved successfully, then these SQL statements are classified as Invalid SQL. A SQL statement can be [invalid](#) if the database object it references does not exist, the database user does not have privileges to access it, or the incorrect database and user are used to scan the Job.

All valid SQL statements are further classified as Simple, Complex, or Problematic SQL statements.

Problematic SQL Statements

[Problematic SQL](#) statements are potentially under performing SQL statements that should be optimized.

Problematic SQL satisfy one of the following criteria:

- The number of tables referenced in the query plan of a SQL statement exceeds the upper limit of the Complex table scan operations range.
- Single full table scan with table size larger than the threshold size.
- Single full table scan in a nested loop with table size larger than the threshold size.
- The number of worktables is greater than or equal to the defined value if the **With Worktable** checkbox is selected in the Preferences window.
- The number of reformatting.

Complex SQL Statement

[Complex SQL](#) statements are complicated SQL statements where there is room for improvement. Complex SQL satisfy one of the following criteria:

- If the number of tables referenced in the query plan of an SQL statement falls into the Complex table scan operations range defined in the Preferences window.
- SQL statements with full index scan.

Simple SQL Statement

Simple SQL statements are direct and straightforward SQL statements with minimal probability of improvement. SQL statements are defined as Simple SQL statements when number of tables referenced in the query plan is less than the lower limit of the Complex table scan operations range; the default value is 1 table.

Problematic SQL

View SQL Classification-Problematic page

Problematic SQL

Number of table operations greater than: 3

With full table scan

Table size (KBytes): 16

Number of rows: 1000

Including inserted simulation temp table

Including deleted simulation temp table

With full table scan iterated by nested loop

Table size (KBytes): 16

Number of rows: 1000

Including inserted simulation temp table

Including deleted simulation temp table

With number of worktables greater than or equal to: 1

With number of Reformattings greater than or equal to: 1

SQL statements are classified as Problematic SQL if they meet any one of the following requirements:

Number of table operations greater than (Default = 3)

Read-only field indicating the number of table operations references in the query plan. If the total number of table operation is greater than this value, then this SQL statement is classified as Problematic. This value is the same as the upper limit of the Complex table operations range.

With full table scan

Specify that SQL statements with full table scan are classified as Problematic SQL statements. This classification depends upon the size of the table or the number of row.

Table size (KBytes): (Default = 16, Range= 8 to 9,999,996)

Number of rows: (Default = 1000)

In the case of a temporary table where the size is unknown when the query plan is retrieved, any full table scan will be classified as problematic.

Note: Table size is calculated using sp_spaceused.

You can specify to include or exclude inserted or deleted simulation temp table created in the SQL Scanner [Trigger Conversion](#) by using the **Including inserted simulation temp table** and **Including deleted simulation temp table** checkboxes.

With full table scan iterated by nested loop

Specify whether SQL statements with a full table scan inside a nested loop are classified as Problematic SQL statements. This classification depends upon the size of the table or the number of row.

Table size (KBytes): (Default = 16, Range= 8 to 9,999,996)

Number of rows: (Default = 1000)

You can specify to include or exclude inserted or deleted simulation temp table created in the SQL Scanner [Trigger Conversion](#) by using the **Including inserted simulation temp table** and **Including deleted simulation temp table** checkboxes.

With number of worktables greater than or equal to (Default = 1, Range = 1 to 99)

Specify that SQL statements involving the number of worktables greater than or equal to the defined number are classified as Problematic SQL statements.

With number of Reformattings greater than or equal to (Default = 1, Range = 1 to 99)

Specify that SQL statements whose query plan uses reformatting are classified as Problematic SQL. Reformatting is the process of generating a worktable with a clustered index and performing a nested-loop join. The Adaptive Server optimizer may choose this strategy when the table is large and does not have any useful index for a join.

Related Topics

[SQL Classification](#)

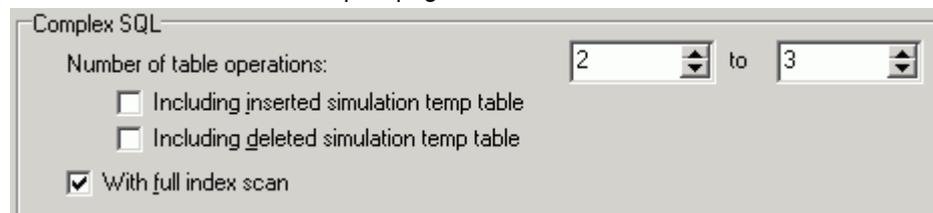
[Complex SQL](#)

[Simple SQL](#)

[Invalid SQL](#)

Complex SQL

View SQL Classification - Complex page



Complex SQL

Number of table operations: 2 to 3

Including inserted simulation temp table

Including deleted simulation temp table

With full index scan

SQL statements are classified as Complex SQL if they meet any one of the following requirements and do not meet any requirement for being a [Problematic SQL](#) statement:

Number of table operations (Default = 2 /3, Range 2 to 99)

Specify the number of table references in the query plan for Complex SQL statements. Checkboxes are available to indicate whether you want the inserted and deleted simulated temp tables to be included in this table range.

You can specify to include or exclude inserted and deleted simulation temporary tables created in the SQL Scanner Trigger Conversion by using the **Including inserted simulation temp table** and **Including deleted simulation temp table** checkboxes.

With full index scan (Default = *checked*)

Specify whether SQL statements with full index scan are classified as Complex SQL statements.

Related Topics

[SQL Classification](#)

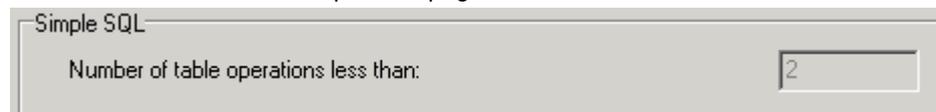
[Problematic SQL](#)

[Simple SQL](#)

[Invalid SQL](#)

Simple SQL

View SQL Classification - Simple SQL page



SQL statements are classified as Simple SQL if they meet the following requirement and do not meet any requirement for being a [Problematic SQL](#) statement or a [Complex SQL](#) statement:

Number of table operations less than (Default = 2)

Read-only field indicating the number of table operations references in the query plan. If the total number of table operation is less than this value, then this SQL statement is classified as Simple. This value is the same as the lower limit of the Complex table operations range.

Related Topics

[SQL Classification](#)

[Problematic SQL](#)

[Complex SQL](#)

[Invalid SQL](#)

Invalid SQL

In some cases the scanned SQL statements are classified as invalid. They may fall into one of the following:

No privilege to tables or views

The logon user does not have privilege to the tables or views referenced even though the syntax of the SQL statement is correct.

Using the wrong database or user

You can declare the database and user used to scan for SQL statements. If these do not match with the SQL statement, it may be classified as invalid.

Dynamic SQL statements

The SQL Scanner is unable to identify SQL statements that are dynamically created if the text of the SQL statement is not all on one command line.

Related Topics

[SQL Classification](#)

[Problematic SQL](#)

[Complex SQL](#)

[Simple SQL](#)

General (General Tab)

View General - General page

The screenshot shows the 'General' tab settings page. It contains the following sections and options:

- Character set:** A dropdown menu set to 'English (DOS)'.
- Code Finder and Object Extractor:** A checkbox for 'Show extraction options' which is checked.
- Display process description animation during optimization:** A checkbox for 'Show process description' which is checked.
- Maximize window:** A checkbox for 'Maximize the first window opened' which is checked.
- Load Data Dictionary:** A checkbox for 'Load data dictionary after database connection' which is unchecked.
- SQL Parameters:** A checkbox for 'Enable SQL parameter history' which is checked, and a numeric spinner for 'Maximum number of parameters' set to 100.

The General page of the General tab of the Preferences window allows you to specify the general settings.

Character set (Default = *English*)

Character set

Specify the language set used to enter and display SQL and data. However, this setting does not affect the user interface language which is always in English.

Code Finder and Object Extractor (Default = *checked*)

Show extraction options

Specify whether to show the Extraction Criteria window when extracting object DDL in the Code Finder module and Object Extractor module. The Extraction Criteria window allows the specification of the script information.

Display process description animation during optimization

Show process description (Default = *checked*)

Specify whether to show during the optimization process the animated explanation (displayed in the Information pane of the SQL Optimizer window) describing the optimization process.

Maximize window

Maximize the first window opened (Default = *cleared*)

Specify that the first window you open is maximized. Every window that you open after that also opens maximized.

If you "restore down" a window, then all open windows are "restored down" and any other window that is open is not maximized as it is open. Once all windows are closed, the next window that is open is maximized as it is opened.

Load Data Dictionary

Load database dictionary after database connection (Default = *cleared*)

Specify that the database dictionary is not automatically loaded into the memory of the computer every time you connect to a database. With this option unselected, the specific information that you need from the database is loaded when the SQL statement is parsed for functions such as scanning, optimization, and index generation. With this option unselected, the [member lookup](#) function used in text editing will not have all of the information about every database object.

SQL parameters

Enable SQL parameter history (Default = *checked*)

Specify to save the parameter information that you enter into the Parameters window so that next time that you are prompted to enter the values for the parameters, the data type and value that you last used for a parameter name are automatically entered for you. The parameter names are case sensitive, so `dpt_id` and `DPT_ID` are treated as two different parameters.

Maximum number of parameters (Default = *100*, Range 10 - 9999)

Specify the maximum number of parameters that will be saved. When the maximum number is reached, the parameter name that is the longest unused will be removed from the file when a new parameter name is added.

Related Topics

[Synchronizing the data dictionary](#)

[Input Parameter Values](#)

Message (General Tab)

View General - General page

Display confirmation message

- Show confirmation before closing Application
- Show confirmation before clearing optimized SQL statements
- Show confirmation before saving data
- Show confirmation for Run Result

Display description message

- Show message when SQL Collector is opened

The Message page on the General tab of the Preferences window allows you to decide which confirmation messages to show.

Display confirmation messages

Show confirmation before closing Application (Default = *checked*)

Brings up a dialog field to confirm that you really want to exit the program.

Show confirmation before clearing optimized SQL statements (Default = *checked*)

Brings up a dialog field to confirm that you want to remove all the alternative SQL statements and reminds you that you can save them.

Show confirmation before saving data (Default = *checked*)

Specify whether to show a confirmation message when saving data (for example from the **Run Result** and **Database Explorer** [Data tab] windows).

Show confirmation for Run Result (Default = *checked*)

Specify whether to show the confirmation message to continue to retrieve records from the first result set before getting the second result set.

In the Run Result window, you can display multiple result sets. If there is a query already opened from the last time you executed the Run Result function, you are prompted to continue to retrieve all records from the first result set before getting the second result set.

Note: The time it takes to retrieve all records from the last query will depend on the SQL performance.

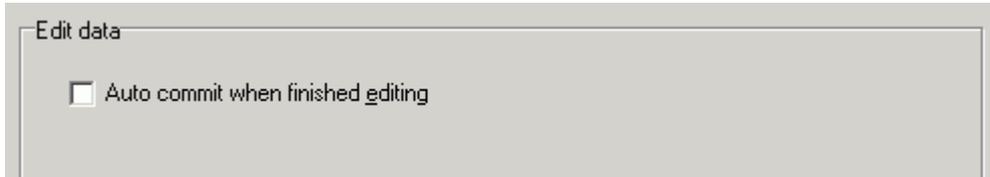
Display description message

Show message when SQL Collector is opened (Default = *checked*)

Specify whether to show a message each time the SQL Collector module is opened that tells you to use the SQL Inspector module if you have Adaptive Server 15.0 or later.

Result Set (General Tab)

View General - Result Set page



The Result Set page on the General tab of the Preferences window allows you to set preferences for editing data.

Edit data

Auto commit when finished editing (Default = *cleared*)

Specify to automatically commit the data when [editing the data](#) in the data grids.

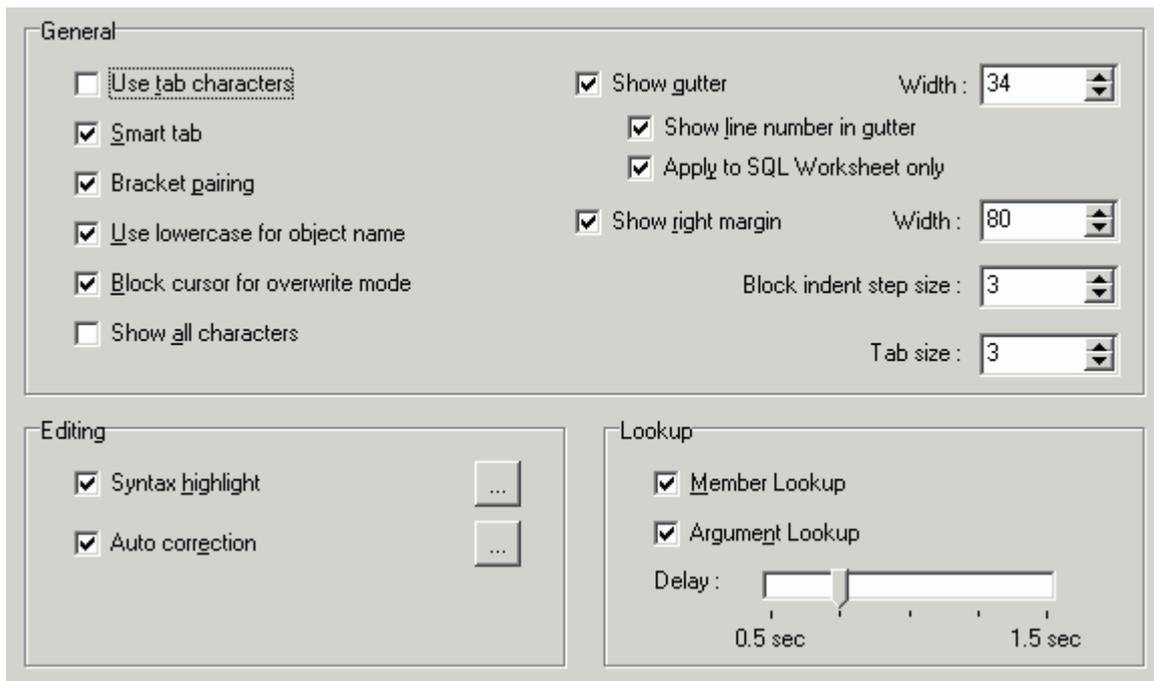
Batch Run (General Tab)

The Batch Run page of the General tab of the Preferences window allows you to set preferences for the Batch Run.

Batch Run for Optimization	Description
Automatically start Batch Run after Optimization	If selected, batch run starts automatically after Optimization.
Batch Run for Index Advisor	Description
Automatically start Batch Run after Index Advisor	If selected, batch run starts automatically after Index Advisor.

Editor

View Editor page



The Editor tab of the Preferences window allows you to define the editable panes control and layout.

General

Use tab characters (Default = *cleared*)

Specify whether to use the tab character (ASCII 9) instead of spaces.

Smart tab (Default = *checked*)

Specify whether the tab aligns with the next non-whitespace character of the previous line.

Bracket pairing (Default = *checked*)

Specify whether to match the corresponding opening bracket when the ending bracket is typed.

Use lowercase for object name (Default = *checked*)

Specify whether to use lowercase for database object names or have them displayed in uppercase.

Block cursor for overwrite mode (Default = *checked*)

Specify whether to use the block cursor when in overwrite mode. Overwrite mode is when text entered at the cursor overwrites the existing text.

Show all characters (Default = *cleared*)

Specify whether to display all characters including spaces, carriage return and tabs on the screen.

Show gutter (Default = *checked*)

Specify whether to show a non-editable boarder on the left of the Editor pane.

Width (Default = 34, Range = 1 to 100 pixels)

Specify the gutter width in pixels.

Show line numbers in gutter (Default = *checked*)

Specify whether to show line numbers in the gutter.

Apply to SQL Worksheet only (Default = *checked*)

Specify to display the line number in the gutter only in the SQL Worksheet module when the *Show line numbers in gutter* option is selected. If not selected, the line numbers are displayed in all windows where you can enter and edit SQL text.

Show right margin ((Default = *checked*)

Specify whether to show a vertical line at the right margin.

Width (Default = *80*, Range = 1 to 1000 characters)

Specify the width of the page or the right margin position.

Block indent size (Default = *3*, Range = 1 to 20)

Specify the block indent size in characters.

Tab size (Default = *3*, Range = 1 to 20)

Specify the character length of a tab.

Editing

Syntax highlight (Default = *checked*)

Specify whether to use syntax highlight. If selected, it is possible to define the format for the following syntax: reserved word, SQL function, comment, identifier, quoted identifier, string, number, symbol, data type, global variable, local variable, temporary table, system procedure, invalid object, force, difference highlight (for SQL Comparer), text selection, member lookup and argument lookup.

Auto correction (Default = *checked*)

Specify whether to automatically correct typing errors and abbreviations as characters are typed. Allows the addition, editing and deletion of the auto correction entries. Also allows the specification of the action key that triggers the automatic replacement.

Lookup

Member Lookup (Default = *checked*)

Specify whether to show the lookup hint for database object members. For example, displays a list column names when a table name or alias name is followed by a period.

Argument Lookup ((Default = *checked*)

Specify whether to show the argument parameters hint for database functions and procedures. These hints display when the opening bracket is typed.

Delay (Default = *0.75* sec, Range = 0.5 to 1.5 sec)

Specify the delay time before the lookup hint displays.

Related Topic

[Editor Functions Overview](#)

Database Setting

The Database settings are Adaptive Server configuration parameter settings. These settings should correspond to how you have these parameters set in Adaptive Server.

Set Ansinull (Default = *Use database default*)

Specify the option with regards to comparison of NULL values. Select **Use database default** to have the program check how the **Set Ansinull** parameter is set on Adaptive Server and set this parameter the same as it is set on the database.

Set Quoted_identifier (Default = *Use database default*)

Specify to allow the use of the delimited identifier (" ") for table names. This setting needs to be turned on if the space character is used in the name of tables or other database objects. Select **Use database default** to have the program check how the **Set Quoted identifier** parameter is set on Adaptive Server and set this parameter the same as it is set on the database.

Set Statistics Subquerycache On (Default = *cleared*)

Specify whether to display the number of cache hits, misses, and the number of rows in the subquery cache for each subquery in the All Records or First Record statistic information in the SQL Optimizer window.

Set Statistics Simulate On (Default = *cleared*)

{SAP Adaptive Server 15 or later}

Specify whether to load simulated statistics into the database. Simulated statistics can be generated using optdiag command and can be used to optimize SQL statements using the simulated statistics rather than the actual statistics.

Note: This can be used to simulate the production database statistics and optimize SQL statements in the development environment.

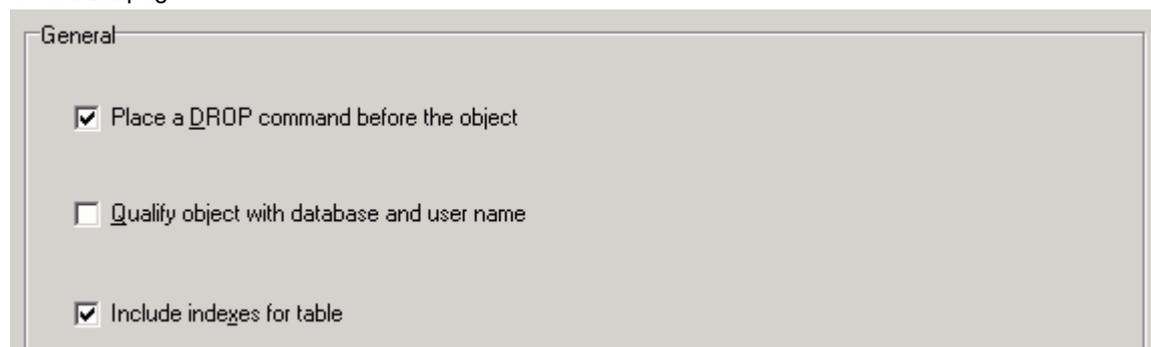
dbcc traceon (3604, 302, 310) (Default = *cleared*)

{sa_role privilege only}

Specify whether to display the trace on information in the SQL Optimizer and SQL Scanner windows. The trace on information displays the reasons why the Adaptive Server optimizer chooses to resolve the SQL statement in a particular way. This option is applicable only if you have sa_role privileges.

DDL

View DDL page



The screenshot shows a window titled "General" with three checkboxes:

- Place a DROP command before the object
- Qualify object with database and user name
- Include indexes for table

The DDL tab of the Preferences window allows you define the way the DDL is extracted for the database objects in the Database Explorer and the Object Extractor.

Place a DROP command before the object (Default = *checked*)

Specify whether to include the **DROP** command before the **CREATE** object command.

Qualify object with database and user name (Default = *cleared*)

Specify whether to place the database and user name in front of every object in the DDL.

Include indexes for table (Default = *checked*)

Specify whether to include the **CREATE INDEX** command when the DDL for a table is extracted.

Note: The settings for this tab are also changed from the Extraction Criteria window that can be optionally displayed at the beginning of the extraction process in the Object Extractor.

Performance Monitor Preferences

Preferences allow the alteration of the default settings used by the Performance Monitor.

To change the preferences

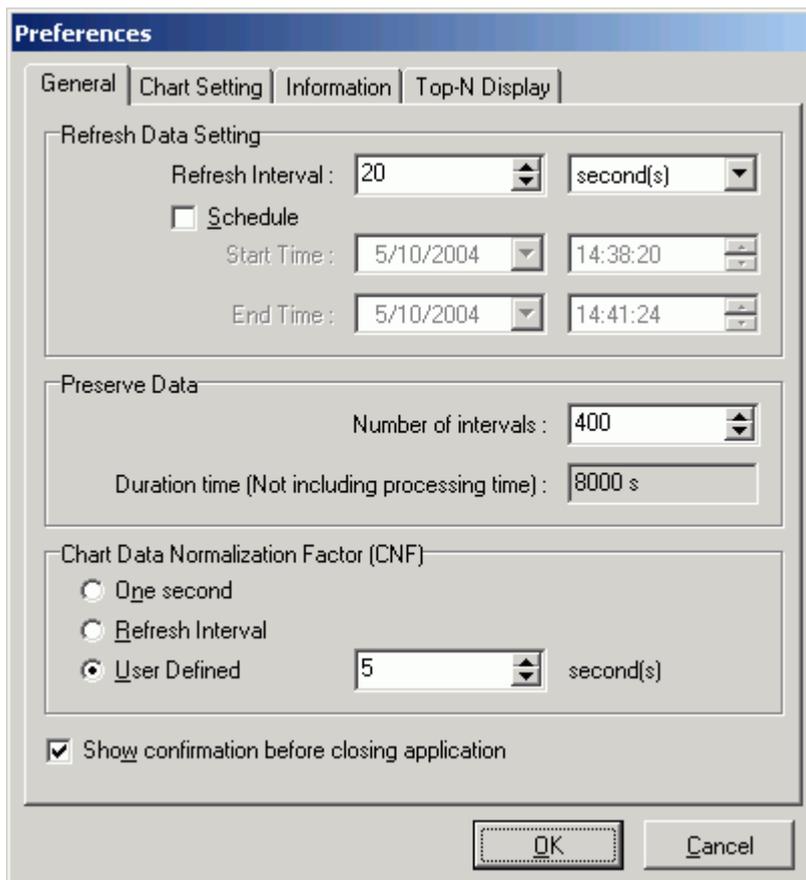
In the Performance Monitor module, click .

Related Topic

[Performance Monitor Overview](#)

General Tab

View Performance Monitor - General page



The **General** tab on the Preferences window for the Performance Monitor is for setting the statistics retrieval frequency for how often the statistics are refreshed, how many refresh intervals are saved and the normalization factor for accurate comparisons of the statistics.

Refresh Data Setting

The Refresh setting allows you to collect the database performance statistics at regular intervals once you have [started the monitoring process](#). If you have selected the **Schedule** option, then the monitoring starts and ends at the specified time. If the **Schedule** option is not selected, then the monitoring starts immediately and does not end until you stop it.

Refresh Interval (Default = 5 seconds)

The refresh interval specifies how often the database performance statistics are collected. It can be set to seconds, minutes, hours, or days. Since most of the statistics for Adaptive Server are cumulative counters stored in the columns of monitoring tables, the Performance Monitor keeps track of those statistics changes and presents the delta values for each time interval in the line charts, bar charts, and pie charts. This makes it easy to pinpoint the resource consumption peak time by reviewing those graphical presentations.

For example, you can check for database performance by taking snapshots of the database statistics for each 15 minutes by setting a Refresh Interval at 15 minutes with Preserve Data number of intervals set at 200. The Performance Monitor starts monitoring the database statistics until you manually stop monitoring. The Performance Monitor keeps up to 200 * 15 minutes = 3000 minutes (50 hours) of statistics data in memory for

review. If the monitoring time is longer than the reserve interval data time, the most current 50 hours of data is retained.

Schedule (Default = *cleared*)

The Schedule monitor function starts the monitor at a specified **Start Time** enabling the tracking of the performance of the database at a specific time. The scheduled monitoring uses the Refresh Interval setting to collect the statistics at regular intervals. The monitoring stops once the **End Time** is reached.

Preserve Data

When monitoring, the statistics are stored in memory for each Refresh Interval. All the statistics for each interval are kept in memory to facilitate the graphical plotting and data retrieval.

Number of intervals (Default = 200)

Specify the number of Refresh Intervals that are stored by the monitoring process for your review.

Duration time (Not including processing time) (Default = 1000 s)

This is a calculation presented in seconds using this formula: **Number of intervals * Refresh Interval**

Chart Data Normalization Factor (CNF)

Since the Refresh Interval time is adjustable by users for each monitor task, the statistical data is proportional to the selected Refresh Interval time. For example; a 10 minutes Refresh Interval may collect 100 writes on Master device, but 30 minutes refresh interval may have 300 writes on Master device. It would be easy to be misled by just looking at the result of 300 writes to judge them in comparison to the 100 writes. The Performance Monitor provides a normalization factor to help standardize the result statistics scale. The following is an example of using 5 minutes as a normalization factor to calculate the results for different Refresh Interval settings:

Monitor Time	Refresh Interval (RI)	Master Device Writes (MDW)	CNF	CNF Result = RI / MDW * CNF
09:00AM-11:00AM	10 minutes	100 writes	5 minutes	50 writes
01:00PM-03:00PM	30 minutes	360 writes	5 minutes	60 writes
04:00PM-06:00PM	60 minutes	540 writes	5 minutes	45 writes

With the Chart Data Normalization Factor, the result is standardized to 5 minutes and the comparison is more accurate. This example shows that the Monitor Time from 1:00PM to 3:00PM was more Write intensive on the Master device with 60 writes for every 5 minutes than the other time slices.

One second

Specify to use 1 second as the Chart Data Normalization Factor. This means that the statistics are shown as an average per second calculation. For example, using an Refresh Interval of 1 minute with 240 reads, the CNF for Reads is $240/60\text{seconds} = 4 \text{ Reads/per second}$.

Refresh Interval (Default)

Specify to use the Refresh Interval as the Chart Data Normalization Factor.

User Defined

Specify in seconds the Chart Data Normalization Factor.

The range of this setting is dependent upon the Refresh Interval measurement you select. The measurement is always in seconds.

Range for seconds: 1 - 60 (1 minute)

Range for minutes: 1 - 3600 (1 hour)

Range for hours: 1 - 216000 (60 hours)

Range for days: 1 - 4665600 (54 days)

Chart Data Normalization Factor text field

If the **User Defined** option is not selected, this seconds text field is dimmed and the calculation from the **One second** or **Refresh Interval** displays in this field.

Show confirmation before closing application (Default = *checked*)

Brings up a dialog field to confirm that you really want to exit the Performance Monitor.

Related Topics

[Performance Monitor Overview](#)

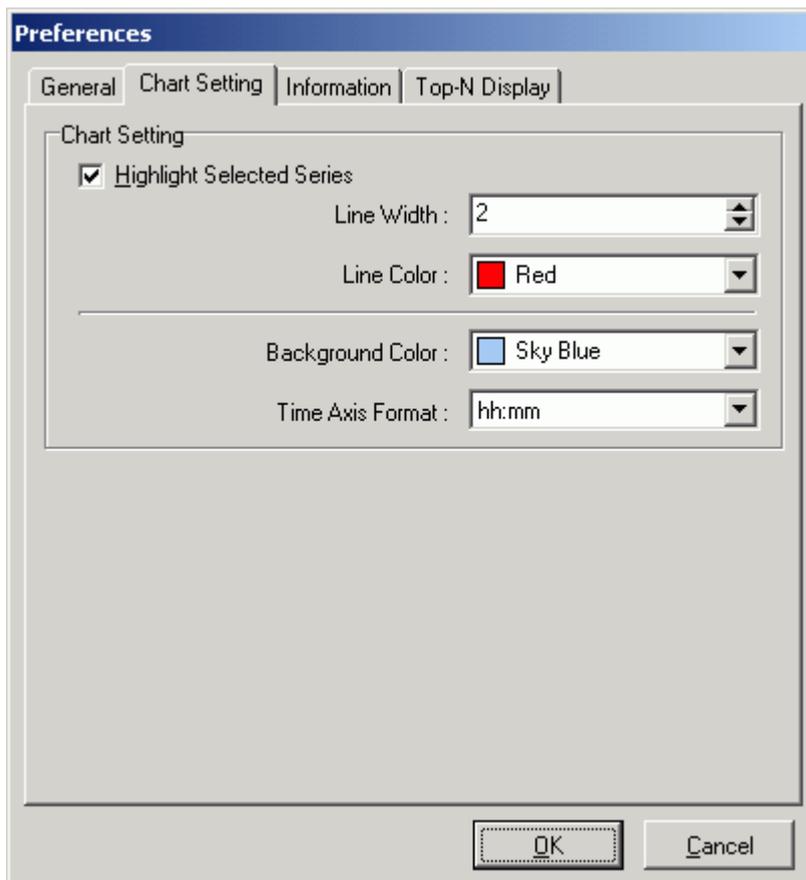
[Chart Setting Tab](#)

[Information Tab](#)

[Top-N Display Tab](#)

Chart Setting Tab

View Performance Monitor - Chart Setting page



The **Chart Setting** tab on the Preferences window for the Performance Monitor is for controlling the display of the charts.

Chart Setting

Highlight Selected Series (Default = *checked*)

The following settings affect the chart display of the selected statistic when you have drilled down to the bottom level of information.

Line Width (Default: 2 Range: 2 - 10)

Specify the width of the line for the statistic that you have selected in the left pane. This is used to enable the statistic selected in the left pane to stand out from the other statistics in the chart.

Line Color (Default = *Red*)

Select the color for the selected statistic from the drop-down list.

Background Color (Default = *Sky Blue*)

Select the color for the chart background from the drop-down list.

Time Axis Format (Default = *hh:mm*)

Select the format for displaying the date and time on the chart from the drop-down list.

Related Topics

[Performance Monitor Overview](#)

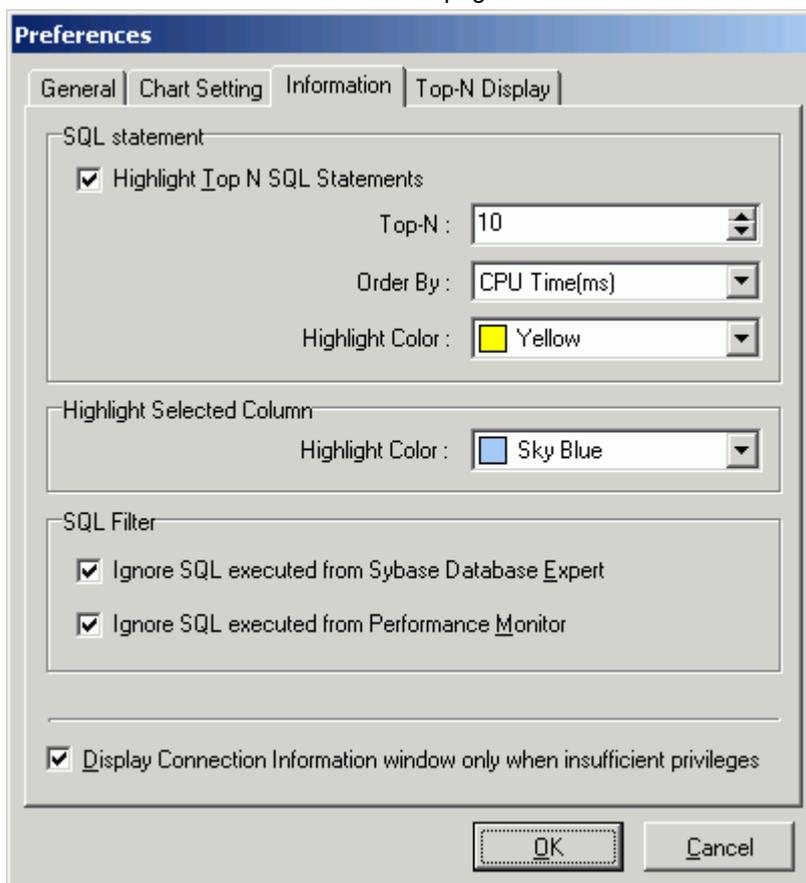
[General Tab](#)

[Information Tab](#)

[Top-N Display Tab](#)

Information Tab

View Performance Monitor - Information page



The **Information** tab on the Preferences window for the Performance Monitor determines which of the top SQL statements are highlighted, sets highlighting colors, and sets the automatic display of the connection information when logging on.

SQL statement

In the left pane of the Performance Monitor window, individual SQL statements are displayed when you drill down from Information to Process Information to the Executing SQL Statements. The following settings determine how many SQL statements are highlighted, by what criteria they are sorted, and the color that is used for highlighting.

Highlight Top N SQL Statements (Default = *checked*)

The top SQL statements are highlighted for you to quickly identify which SQL statements are consuming the most resources.

Top-N (Default = *10*)

Specify the number of SQL statements to be highlighted.

Order By (Default = *CPU Time(ms)*)

Select the statistic to sort the SQL statements from the drop-down field.

Highlight Color (Default = *Yellow*)

Select the color from the drop-down field.

Highlight selected column

When you drill down to the lowest level of the charts, the data from the table displays in a grid underneath the chart. The column for the statistic that is selected in the left pane is highlighted.

Highlight Color (Default = *Sky Blue*)

Select the color from the drop-down field.

SQL Filter (executing SQL only)

Ignore SQL executed from SQL Optimizer (Default = *checked*)

Specify to exclude the SQL statements that are generated by any of the modules in SQL Optimizer but the Performance Monitor.

Ignore SQL executed from Performance Monitor (Default = *checked*)

Specify to exclude the SQL statements that are generated by the Performance Monitor.

Display Connection Information window only when insufficient privileges (Default = *checked*)

Specify to display the Connection Information window each time you logon if the user logon does not have the database privileges needed for the modules.

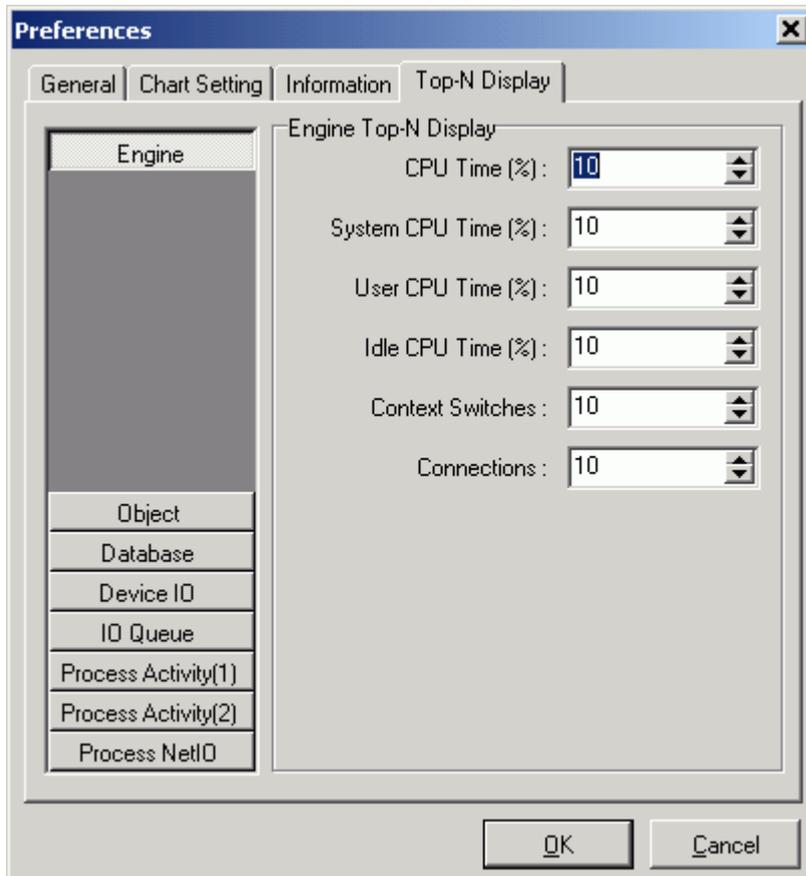
Related Topics

[Performance Monitor Overview](#)

[General Tab](#)
[Chart Setting Tab](#)
[Top-N Display Tab](#)

Top-N Display Tab

View Performance Monitor - Top-N-Display page



The **Top-N Display** tab on the Preferences window for the Performance Monitor is for determining the percent of each statistics that is retrieved from the monitoring tables.

To select only the top statistics

Select the button on the left for the statistics information and fill in the values on the right for each individual statistic.

Please reference the *Adaptive Server Performance and Tuning: Monitor and Analyzing* manual for the meaning of each statistic.

Related Topics

[Performance Monitor Overview](#)

[General Tab](#)

[Chart Setting Tab](#)

[Information Tab](#)

Syntax Highlight

Syntax highlighting changes the colors and attributes of the text in the SQL statement display panes, making it easier to quickly identify parts of the code. The text colors and attributes for syntax highlights are defined within the Preferences window.

To change display attributes of the SQL text

1. Click **Preferences** 
2. Select the **Editor** tab.
3. Under the Editing section, the Syntax highlight checkbox is used to apply the syntax highlight to the code. By default this option is selected.
4. To modify the colors and attributes, click the corresponding  button to open the Syntax Highlight Settings window.

Element options

Elements

It is possible to define the font format and color for the following elements:

Element	Description
Reserved Word	Adaptive Server SQL reserved words. For example: ADD, ALL, ALTER, AND, ANY, AS, ASC, BACKUP, BEGIN
Function	Adaptive Server build-in functions. For example: ABS, ACOS, ASCII, ASIN, ATAN, ATAN2, CEILING, CHARINDEX
Comment	Comment delimiters (--).
Identifier	Any text that does not fall in the above categories.
Quoted Identifier	Database object identifier within quotes.
String	Text enclosed in single quotes (' ').
Number	Numbers 0 to 9 and floating point.
Symbol	Symbols. For example: !, %, ^, &, *, (,), -, =, +, {, }, ;, ~, <, >, ,, ,, /, <=, >=
Data Type	Adaptive Server data type.

	For example: BIT, INT, SMALLINT, DEC, CHAR, VARCHAR, BINARY
Global Variable	@@variable accessed by a any program block.
Local Variable	@variable accessed by a single program block.
Temporary Table	User-Defined Temporary Tables created for the current session.
System Procedure	Procedures from the System catalog.
Invalid object	Unrecognized database object under the current database and user.
Force	Optimizer hints that influence choices of the SQL Server optimizer.
Difference highlight	The differences between two SQL for the SQL Comparer window.
Text Selection	Text selection highlighting.
Member Lookup	Member lookup field.
Argument Lookup	Argument lookup field.

Foreground color

Specify the foreground color of the selected element; use the drop-down list or  to select the desired color. Select the **Use default foreground color** checkbox to use the Windows default text color.

Background color

Specify the background color of the selected element; use the drop-down list or click  to select the desired color. Select the **Use default background color** checkbox to use the Windows default background color. By default, this checkbox is selected for all elements.

Editor options

Specify the font style and size of all editable panes.

Reset Default button

Reset the default settings for syntax highlight for all settings.

To change the color and attribute of a particular element

1. Select the syntax name from the **Element option** list-box or select the text from the **Preview** section and change the attributes accordingly.
2. Changes are reflected on the **Preview** section, click **Apply** to implement the changes.

Related Topic

[Editor Functions Overview](#)

Auto Correction

Auto correction automatically corrects any typo and spelling errors, if the auto correction option is turned on. For example, if "teh" is typed followed by a space, it will automatically be corrected to "the". You can use this to create your own abbreviations of text you frequently type. For example, if you set up your own auto correction to have "si" replace with "SELECT INTO".

Auto correction entries are user-defined and can be modified in the Preferences window.

To open the Auto Correction window

1. Click **Preferences** .
2. Select the Editor tab.
3. Under the Editing section, the Auto correction checkbox is used to enable auto correction. By default, this option is selected.
4. To modify the auto correction entries, click the corresponding  button to open the Auto Correction Settings window.

Available auto correction entries are listed on the list-box. Each entry has a REPLACE text (the original text typed) and WITH text (the replacement text).

Action keys are the keystrokes that activate the auto correction. When you type the REPLACE text and follow it by any action key, the replacement takes place. No whitespaces are needed to separate each action key. By default, the action keys are ".,:=[\n\t\s" representing the following:

Action Key	Keyboard Function
;	semi-colon
,	comma
:	colon
=	equal
[]	right and left brackets
\n	enter key
\t	tab key
\s	spacebar

To add an auto correction entry

1. Click **Add**
2. Enter the original [Replace] and replacement text [With]
3. Click **OK**.

To modify an auto correction entry

1. Select the entry to be modified from the Auto correction list-box and click **Edit**.
2. Modify the text
3. Click **OK**.

To delete an auto correction entry

- Select the entry to delete from the Auto correction field and click **Delete**.

To implement all changes

- Click **Apply** on the Auto Correction Settings window.

Related Topic

[Editor Functions Overview](#)

Member and Argument Lookup

Member lookup enables you to select the database objects or column names from a list as you create a SQL statement. Argument lookup provides hints for functions and procedures to help construct SQL statements. Both of these functions automatically display as you type the appropriate corresponding text in the SQL statement.

Member lookup displays the list of database object members for databases, users, tables, views and alias. For example, if a table name or alias is typed followed by "." then the corresponding lookup hint is the column names. You select the item from the list and it is entered in to the text. The **[Ctrl+Space]** can be used a short-cut key to bring up the member lookup.

Argument lookup displays the next argument parameter for functions and procedures. For example, if a procedure name is typed followed by "(" then the list of arguments is shown highlighting the next enterable argument. You cannot select items from the Argument lookup list. It provides a guide to help you determine what you should enter as a value for the function or procedure. The **[Ctrl+Shift+Space]** can be used as a short-cut key to bring up the argument lookup.

To enable or disable the member and argument lookup

1. Click **Preferences** .
2. Select the **Editor** tab.

3. Under the Lookup section, specify whether to use the lookups hints.

Member lookup (Default)

Specify whether to show the lookup hint for database object members.

Argument lookup (Default)

Specify whether to show the argument parameters hint for functions and procedures.

Delay (Default = 0.75 sec, Range = 0.5 to 1.5 sec)

Specify the minimum delay time before the lookup hint displays.

Related Topic

[Editor Functions Overview](#)

Indent and Outdent Text

The text can be shifted from left to right or right to left using the Indent and Outdent functions. The number of characters shifted is dependent on the Block indent size (default = 3 characters) parameter in the Preferences window.

To indent text

Highlight the text or place the cursor on the line to be indented and click **Indent** .

To outdent text

Highlight the text or place the cursor on the line to be outdented and click **Outdent** .

Related Topic

[Editor Functions Overview](#)

Comment and Uncomment Text

You can comment or uncomment a line of text by selecting the line or highlighting the text and using the Comment and Uncomment functions. These two functions use the single line comment delimiter "--".

Commenting out a line means that the line is ignored during execution.

To comment a line of text

Select the text or place the cursor on the line to be commented and click **Comment** . Notice that the beginning of a text or line prefix is marked with the "--" delimiter.

To uncomment a line of text

Select the text or place the cursor on the line to be uncommented and click **Uncomment** . Notice the first occurrence of the "--" comment delimiter is removed.

Related Topic

[Editor Functions Overview](#)

Change Text to Uppercase and Lowercase

Text can be changed from uppercase and lowercase by selecting the text and using the Uppercase and Lowercase functions.

To change text to uppercase

Select the text and click **Uppercase** .

To change text to lowercase

Select the text and click **Lowercase** .

Related Topic

[Editor Functions Overview](#)

Bracket Pairing

Simple mistakes are often made during SQL construction because of uneven left and right brackets. With bracket pairing, the corresponding open bracket is matched with the closing bracket as you type the closing bracket.

To enable bracket pairing

1. Click **Preferences** .
2. Select the **Editor** tab.
3. Under the General section, select **Bracket Pairing**.

Bracket pairing matches brackets for the parentheses (), the curly brackets { }, and the square brackets [].

Related Topic

[Editor Functions Overview](#)

Customize Editable Panes

You can customize the behavior of editable panes in many ways. The editor options can be set in the **Preferences** window under the **Editor** tab. These options include:

- [Use tab characters](#)
- [Smart tab](#)
- [Use lowercase for object name](#)
- [Block cursor for overwrite](#)
- [Show all characters](#)
- [Right margin](#)
- [Gutter](#)
- [Block indent size](#)
- [Tab Size](#)

Related Topic

[Editor Functions Overview](#)

Editor Functions

Below is a list of available functions within the editable panes.

Button or Menu	Function
	Syntax Highlight
	Auto Correction
	Bracket Pairing
	Argument Lookup
	Member Lookup
Edit Menu 	Indent

Edit Menu 	Outdent
Edit Menu 	Comment
Edit Menu 	Uncomment
Edit Menu 	Uppercase
Edit Menu 	Lowercase

Related Topic

[Editor Functions Overview](#)

Editor Panes Shortcuts

The following list is the quick reference to some common shortcut keys provided to quickly accomplish tasks that are performed frequently within the Editor panes.

Function	Shortcut Keys
Delete next word	CTRL+DELETE
Delete previous word	CTRL+BACKSPACE
Delete next character	DELETE
Delete previous character	BACKSPACE
Delete current line	CTRL+Y
Highlight a word	SHIFT+CTRL+LEFT or RIGHT ARROW
Highlight a character	SHIFT+LEFT or RIGHT ARROW
Cursor forward a word	CTRL+RIGHT ARROW
Cursor backward a word	CTRL+LEFT ARROW
Insert mode	INSERT

Bring up Member Lookup	CTRL+SPACEBAR
Bring up Argument Lookup	SHIFT+CTRL+SPACEBAR
Comment	SHIFT+CTRL+C
Uncomment	SHIFT+CTRL+N
Indent	SHIFT+CTRL+I
Outdent	SHIFT+CTRL+O
Uppercase	SHIFT+CTRL+U
Lowercase	SHIFT+CTRL+L
Quick Find	CTRL+Alt+F
Find	CTRL+F
Find Next	F3
Replace	CTRL+Alt+R
Go to Line	CTRL+Alt+G
Select All	CTRL A
Cut	CTRL X
Copy	CTRL C
Paste	CTRL V
Undo	CTRL Z
Save	CTRL S

Related Topic

[Editor Functions Overview](#)

Send SQL Statement to SQL Optimizer

Performance improvement could be obtained through the SQL Optimizer window by transforming the SQL syntax into equivalent statements with different query plans.

If you are in a module that has a SQL Text pane, you can send a SQL statement to the SQL Optimizer.

To send a SQL statement to the SQL Optimizer and automatically start the optimization process

1. Select the SQL statement you want to optimize.

2. Click **Send to SQL Optimizer** .

Supported SQL Statements

The SQL Optimizer only supports a single SELECT, SELECT INTO, INSERT, UPDATE or DELETE SQL statement.

Setting Database and User

When you optimize a SQL statement, create a temp table, or execute a SQL statement, you must have the right database selected and your logon must have access to the tables used in the SQL statement.

In the bottom left corner of the main window are find two drop-down lists: one for databases and the other for users. Select the appropriate database and user. This is equivalent to perform a **USE** and **SETUSER** commands.

SQL from the SQL Scanner

You receive the following warning message if your current database and user do not match the one previously used to retrieve the query plan of the SQL statement. Click Yes to have the database and user automatically changed for you and continue with the optimization process.

"Your current connection is not the same as the scanned SQL statement. Do you want to change the database to 'database_name' and user to 'username' now to continue with the optimization?"

SQL in a Cursor

If the SQL statement is used within a cursor declaration the **SQL for cursor** checkbox is automatically selected.

SQL with Temporary Tables

If the SQL statement uses a temporary table and the SQL Scanner found the CREATE TABLE or SELECT INTO statement for creating the table, then the User-Defined Temp Table window is automatically opened and the DDL for creating and populating is entered ready for you to execute it.

Related Topic

[SQL Optimizer Overview](#)

Copy to SQL Worksheet

SQL statements can be executed in the SQL Worksheet. If you are in a module that has a SQL Text pane, you can copy a SQL statement to the SQL Worksheet.

To copy a SQL statement to the SQL Worksheet

1. Select the SQL statement you want to execute.

2. Click **Copy to SQL Worksheet** .

The selected SQL statement is copied to Editor pane of the SQL Worksheet window.

Related Topic

[SQL Worksheet Overview](#)

Send to Index Advisor

Performance improvement for a SQL statement could be obtained by creating indexes to reduce I/O consumption. By sending a SQL statement to the Index Advisor module, you can obtain recommendations for new indexes that could change SQL performance.

If you are in a module that has a SQL Text pane, you can send a SQL statement to the Index Advisor.

To send a SQL statement to the Index Advisor and automatically start the index recommendation process

1. Select the SQL statement you want to generate indexes for.

2. Click **Send to Index Advisor** .

Note: The Index Advisor only supports a single SELECT, INSERT, UPDATE and DELETE SQL statement.

Related Topic

[Index Advisor Overview](#)

Open SQL from SQL Repository

SQL statements saved in the SQL Repository can be copied into the SQL Optimizer, SQL Formatter, or SQL Worksheet.

To open a SQL statement from the SQL Repository

1. In the SQL Optimizer, SQL Formatter, or SQL Worksheet., select **Edit | Open SQL from SQL Repository**.
2. Navigate through the tree structure and select the SQL statement.
3. Either double-click the SQL statement, or click **Open**.

Related Topic

[SQL Repository Overview](#)

Save SQL to SQL Repository

If you are using this function from the SQL Scanner, you can send multiple SQL statements to the SQL Repository. If you are using this function from the other modules, you can send one SQL statement to the SQL Repository.

To save multiple SQL statements to the SQL Repository module

1. In the SQL Scanner, select the Job which contains the SQL statements. In the SQL Inspector, click in the SQL Text pane.
2. Click **Save SQL to SQL Repository** .
3. Select the location in which to save the SQL statements and click **OK**.

To save a SQL statement to the SQL Repository module

1. In the modules with a SQL Text window, select a SQL statement.
2. Click **Save SQL to SQL Repository** .
3. Select the location in which to save the SQL statements and click **OK**.

Only valid SQL statements are added to the SQL Repository.

Note: The SQL Repository only supports a single SELECT, INSERT, UPDATE or DELETE SQL statement.

Related Topic

[SQL Repository Overview](#)

Create Benchmark Factory Import File

All the SQL statements can be saved in a text file from the SQL Scanner, the SQL Repository, the SQL Inspector, and the SQL Optimizer windows. These SQL statements can then be imported into Benchmark Factory 4.6 or later.

To create a file to import into Benchmark Factory

1. Right-click and select **Create Benchmark Factory Import File**.
2. Select the specific SQL statements that you want to save.
3. Enter the filename and select the file location.

SQL Navigation

You can navigate through multiple SQL statements tabs using the following methods.

- Click the tab set located at the bottom of the window.
- Use the **SQL | Go to SQL** function which navigates to a specify SQL number.
- Use the navigation buttons on the toolbar or select the corresponding menu item.

Menu	Button
Navigate First SQL	
Navigate Previous SQL	
Navigate Next SQL	
Navigate Last SQL	

Modify Data

You can edit the data in the Database Explorer, the SQL Worksheet, the Run Result, and the Parameters windows. When the data is display, a button bar displays at the bottom of the pane when you can edit the data. This button bar offers functions to view, edit, insert and delete data.



Navigate Data

To navigate through data you can either use the scroll bar or use the **Navigator**  buttons (**First Record**, **Previous Record**, **Next Record**, and **Last Record**). Depending on the amount of data, scrolling to

the bottom of the page or executing the **Last Record** function may take a long time as all records are retrieved from the database.

Edit a Record

To edit the record's column, place the cursor on the record that you want to edit and click **Edit Record** . You can change the data in the grid. Press the **tab** key or use the mouse to move to the next column. Once you complete the modification, click **Commit**  or **Rollback** .

Insert a Record

To insert a new record, place the cursor on the record that you want the new record inserted before and click **Insert Record** . Enter data for each column, press the **tab** key or use the mouse to move to the next column. Once you have completed entering the data, click **Commit**  or **Rollback** .

Delete a Record

To delete a record, place the cursor on the record that you want to delete and click **Delete Record** . Click **OK** on the confirmation dialog to confirm that the record you want to delete is correct. The **Delete Record** is automatically committed on your database.

Refresh Data

To refresh the data, click **Refresh** .

Note: Not all Views are updateable; a general rule is that any UPDATE, DELETE, or INSERT statement on a join view can be modified if only one underlying base table exists.

Related Topic

[Result Set \(General Tab\)](#)

Copy Data from a Single Cell

You can copy the data from a single cell into the windows clipboard from the Database Explorer, the SQL Worksheet, the Run Result, and the Parameters windows.

To copy data

1. Highlight the cell with data.
2. Right-click and select **Copy**.

Comments

The three most popular comment delimiters used within SQL statements are recognized as valid SQL syntax:

- A single line comment using --
- A single line comment using //
- A block comment starting with /* and ending with */

Other comment delimiters are not accepted as valid SQL syntax. Due to the lack of uniform standards in the presentation of comments within source code, it is not practical to implement each and every commenting indicator.

Parameters

Parameters, also called variables, can be embedded in a SQL statement without pre-defining the data type and value. All variable names are highlighted in red (by default) after the formatting. Variables with or without a "@" sign are recognized. When you optimize the SQL statement or retrieve the query plan, run time, the run result for a SQL statement that uses a parameter, you are prompted to enter the variables using the Parameters window.

Note: You can retrieve the query plan for a SQL statement that has a variable without the need to enter the data type or value by executing the [Show Default Plan](#) function from the SQL menu.

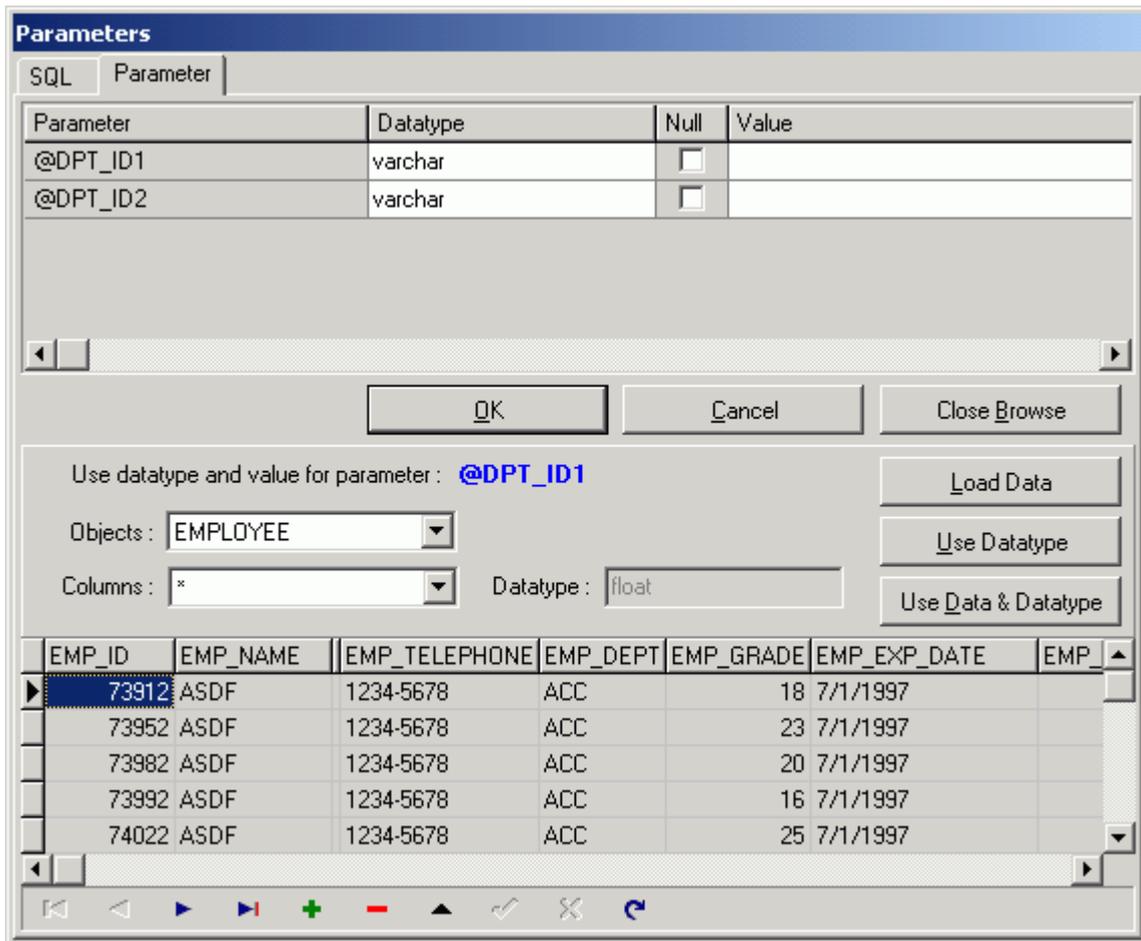
Related Topic

[Parameters Window](#)

[Input Parameter Values](#)

Parameters Window

[View Parameters Window](#)



The data type and value of parameters are entered using the Parameters window. If parameters are embedded in the SQL statement, the Parameters window displays each time you optimize the SQL statement, execute the **Show Plan, Run Result, Run Time, Batch Run** functions, or create a temporary table through the User-Defined Temp Table window.

The parameter values that you have previously entered can be saved by selecting the [Enable SQL parameter history](#) Preferences setting. If the same parameter is used in a SQL statement again, the value and the data type from the last time it was used is automatically entered for you.

To help identify the data type and value, click **Browse** to expand the Parameters window. The bottom pane displays displaying object and data information corresponding to the selected objects from the SQL statement. Select the object to browsed using the **Objects** and **Columns** drop-down list. If a column is selected (asterisk * selects all columns from the selected object), the corresponding data type is shown.

Click **Load Data** and highlight the cell with the variable value from the data grid. Then click **Use Data & Datatype** or double-click the cell to copy the compatible data type and value to the parameter selected at the top of the window.

Click **Use Datatype** to copy a compatible data type to the parameter selected at the top of the window.

After the data type and value of all the parameters have been selected, click **OK**.

Notes:

- The values entered in the Parameters window has a direct effect the on the query plan, run time and run result retrieved and on the resulting SQL alternatives that are generated during the optimization process.
- You can [edit the data](#) using the Button Bar at the bottom of the Parameter window.

- If the Parameters window displays when you do not have variables in the SQL statement, this may be caused by incorrect spelling of column or table names, having the wrong database or user selected, or the tables or column do not exist in the database.

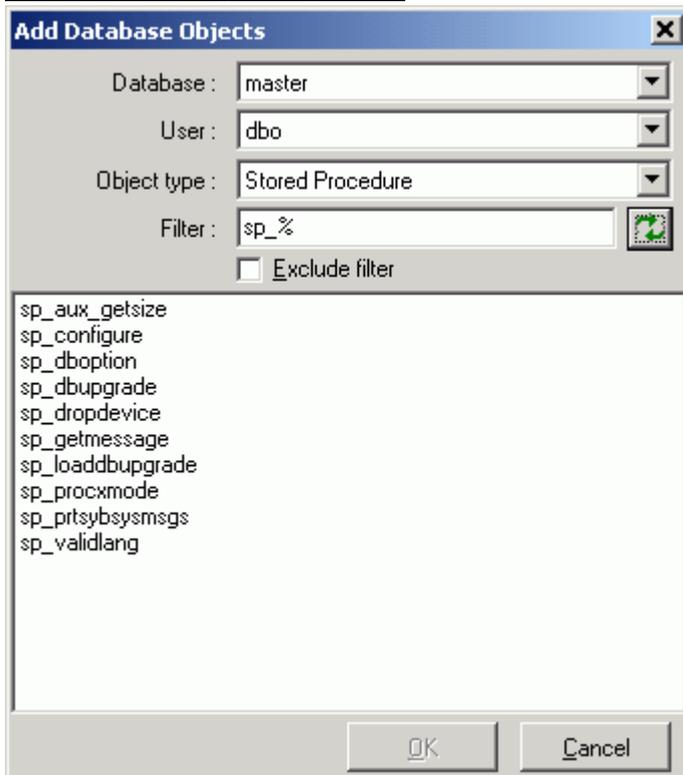
Related Topics

[Parameters](#)

[Input Parameter Values](#)

Filter Database Objects

View Add Database Objects Window



When you are selecting database objects from the list of all objects, you can filter the list in order to more quickly locate specify objects.

To open the Add Database Objects window.

1. Click **Browse**.
2. Enter the following information to filter the database objects.

Field	Description
-------	-------------

Database	Specify the database name where the objects reside.
User	Specify the owner of the database objects.
Object type	Select the database object type from the list.
Filter	Using the % wildcard that replaces multiple characters, enter the filtering criteria. The filter is case sensitive, so you must match the upper or lowercase characters of the database object name.

Related Topic

[Add Database Objects \(Add Jobs Wizard\)](#)

SQL Information Pane

The SQL Information pane displays in several modules. It provides a variety of information about the SQL statement like the query plan, abstract plan, trace on and run time information. The specific information is dependent upon the function of the module.

The SQL Information pane displays SQL information according to the button you select from the button bar



at the top of this pane. The exact buttons that you find at the top of this pane varies from one module to the next. These buttons include: query plan, abstract plan, trace on, other information (SQL classification, connection information, warning messages, etc), all records statistics, first record statistics, Scanner temp table and checked details.

Query Plan

Displays the SQL [query plan](#). When two SQL Information panes are displayed side-by-side, as in the SQL Comparer module and the Analyzer modules, the differences between the operations are highlighted in green.

Abstract Plan

The Abstract Plan button is only available for Adaptive Server 15 or later and you have selected **Dump abstract plan** option in the Preferences window.

Displays the [abstract plan](#) which describes the query plan using a language created for that purpose. This language contains operators to specify the choices and actions that can be generated by the optimizer.

Trace On

The Trace On button is only available if you have selected **dbcc traceon (3604, 302, 310)** option in the [Preferences window](#) and have sa_role privilege.

Trace on shows the reason why the Adaptive Server optimizer chose a particular way of executing the original SQL statement, displaying the reasons for index and table joins selection.

Information

The exact information depends on the module. It always displays the [SQL classification](#) type. This classification is dependent on the parameters set in the Preferences window to identify whether the original SQL statement is potentially Problematic. It may also display:

- Warning or alert information about the SQL statement.
- SQL statement type classification: Problematic, Complex, Simple, or Invalid.
- Database error message if SQL is classified as Invalid.
- SQL conversion applied, if conversions have been added to the scanned SQL text in order for it to generate a query plan, this information is also displayed.
- Start position of the SQL statement (SQL Scanner only, for database object, TXT and SQL files).
- Connection information (Login name, Server name, Database and User)
- Database Settings from the Preferences window.
- Session settings for the optimization timeout limit and optimization goals. (Available in Adaptive Server 15 or later) The optimization timeout limit is an Adaptive Server configuration parameter that specifies the amount of time Adaptive Server can spend optimizing a query as a fraction of the total time spent processing the query. The [optimization goals](#) are selected by you in the Preferences settings.

All Records

Displays the run time information that is collected when retrieving all records of the query.

First Record

Displays the run time information that is collected when retrieving the first record of the query.

Both **All Records** and **First Record** buttons display the run time statistical information:

- The amount of disk activities needed to execute the SQL statement.
 - Number of scans performed
 - Number of pages read from the data cache
 - Number of pages placed into the cache
 - Number of pages read from disk

Number of scans performed

- Number of milliseconds required to parse, compile, and execute each SQL statement.
- SQL script used to test run the SQL statement.

Scanner Temp Table (SQL Scanner)

Displays the temporary table SQL statement assumed to create or modify the temporary table used on the scanned SQL statement if the SQL Scanner finds it in the source.

Checked Details (SQL Scanner)

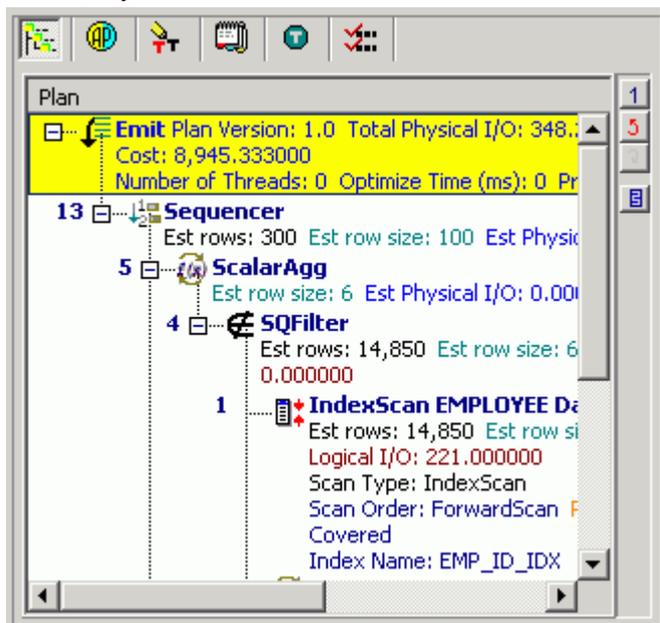
If the selected SQL statement is checked, then this page displays the date time of when the SQL statement was checked, the name of person who checked the SQL, the status, and the checked description.

Index Advisor Window (Index Advisor)

Displays information about how the index will be created and the estimated amount of space required for the index.

Query Plan Overview

View Query Plan Pane



The query plan is the steps the Adaptive Server optimizer follows as it executes a SQL statement. Each line represents how the Adaptive Server optimizer will physically retrieve rows of data from the database or how the data is prepared. The Adaptive Server optimizer performs the child step before the parent step. Depending on the SQL statement, the parent step may be executed when a single row has been returned from the child step. Some query plans may require all rows to be returned from the child step before executing the parent step.

By examining the query plan, you can see exactly how the database executes your SQL statement, helping you judge whether the SQL statement is the most efficient.

You can use the Query Plan page of the SQL Information pane to view the steps Adaptive Server follows to execute the SQL statement. You can animate the plan to show you the progression of the plan steps. You can look up the explanation of the query plan operations.

Related Topics

[Review Query Plans](#)

[Query Plan Options](#)

[Animate Query Plans](#)

[Copy Query Plans](#)

[Plan Detail](#)

Review Query Plans

The query plan can be displayed in different ways to help you get a clear picture of or detailed information about the steps that Adaptive Server is taking to execute the SQL statement.

Note: These different display modes are only available when you are connected to Adaptive Server 15.0 or later.

To change how the query plan displays

1. Right-click the Query Plan window and select **View Plan**.
2. Select one of the following display options.
 - **As Tree Plan**
 - **As Plain Language Plan**
 - **As Graphic Plan**
 - **As MS Graphic Plan**
 - **As Text**
 - **As XML**

When you are viewing the query plan as Tree Plan or as Plain Language Plan, the following functions are available from the button pane at the right edge of the query plan.

Button	Action	Description
	Go to First Step	Highlights the first execution step.
	Go to Previous Step	Highlights the previous execution step.
	Go to Next Step	Highlights the next execution step.
	Show Plan Detail	Displays the Plan Detail window. The information displayed is dependent on the highlighted row.

Related Topics

[Query Plan Overview](#)
[Query Plan Options](#)
[Animate Query Plans](#)
[Copy Query Plans](#)
[Plan Detail](#)

Query Plan Options

The display of the Query Plan page can be customized for each module in which it is included. You can select which elements of the execution plan you would like displayed in the text of the query plan. You can also have the element displayed in a separate column.

To customize the Query Plan page

1. Right-click the query plan and select **Plan Options**.
2. In the **Visible** column, select the elements you want displayed in the execution plan text.
3. In the **As Column** column, select the elements for which you want a column added to the display in the Query Plan page.
4. Click **OK**.

Related Topics

[Query Plan Overview](#)
[Review Query Plans](#)
[Animate Query Plans](#)
[Copy Query Plans](#)
[Plan Detail](#)

Animate Query Plans

You can have the steps in the execution plan highlighted one-by-one in the order they are executed.

To animate the plan or cancel the animation

1. Right-click the Query Plan window and select **View Plan | As Tree Plan** or **As MS Graphic Plan**.
2. Right-click the Query Plan window and select **Animate Plan Steps**.

Related Topics

[Query Plan Overview](#)
[Review Query Plans](#)
[Query Plan Options](#)
[Copy Query Plans](#)

Copy Query Plans

You can copy the query plan to the clipboard.

To copy a query plan

Right-click the Query Plan window and select **Copy**.

When the query plan displays as **Tree Plan** and Plan Text Language the query plan can be pasted as text or as a bitmap picture. Graphic applications, such as MS Paint, only paste the query plan as a bitmap. Text applications, such as Notepad, only paste it in a text format. Applications that handle both text and graphics, such as Microsoft Word, allow you to choose the format if you select the **Paste Special** option to paste the image. This option is usually found on the **Edit** menu.

Related Topics

[Query Plan Overview](#)

[Review Query Plans](#)

[Query Plan Options](#)

[Animate Query Plans](#)

[Plan Detail](#)

Plan Detail

The Plan Detail window provides detailed information on the selected branch of the query plan. This window is opened in two different ways depending on which display view you have selected for the query plan.

To view the Plan Detail – if you are viewing the query plan as Tree Plan, Plain Language Plan, Graphic Plan, or Microsoft Graphic Plan

1. Select the query plan step.
2. Click **Plan Detail** .

To view the Plan Detail – if you are viewing the query plan as Text

Click the query plan step.

Note: **Help | Show Plan Detail** must be selected. If you clear this menu option, then the Plan Detail will not be displayed in the Text view of the query plan.

The Plan Detail provides the following information on each branch of the query plan:

Item	Description
StmtText	Text of the transact-SQL statement.
PhysicalOp	Physical implementation algorithm for the node.

LogicalOp	Relational algebraic operator for the node.
Argument	Supplemental information about the operation being performed, the content depends on the physical operator.
DefinedValues	A list of values introduced by the operator.
EstimateRows	Estimated number of output rows by the operator.
EstimateIO	Estimated I/O cost for the operator.
EstimateCPU	Estimated CPU cost for the operator.
AvgRowSize	Estimated average row size (in bytes) of the row being passed through the operator.
TotalSubtreeCost	Cumulated estimated cost of the operation and all child operation.
OutputList	List of columns being projected by the operation.
Warnings	List of warning messages related to the operation.
Type	Node type.
Parallel	Indicator of whether the operator is running in parallel (0 represents the operator is not running in parallel, while 1 represents the operator is running in parallel).
EstimateExecutions	Estimated number of times the operator will be executed while running the current query.

Table/View Detail

The Plan Detail also provides detailed information about the table and views used by the SQL statement.

To see the table or view details

1. Click the table or view name in the left pane of the Plan Detail window.
2. Click one of the five tabs at the bottom of the right pane to see specific information about the definition, columns, indexes, constraints/key or data for the table or view.

Related Topics

[Query Plan Overview](#)

[Review Query Plans](#)

[Query Plan Options](#)

[Animate Query Plans](#)

[Copy Query Plans](#)

Quick Find

The Quick Find search function searches for the text that is highlighted and moves the line containing the search string to the top of the text pane.

To search with Quick Find

With text highlighted in the active window, select **Search | Quick Find [Ctrl + Alt + F]**

Related Topics

[Search Functions](#)

[Find](#)

[Find Next](#)

[Replace](#)

[Goto Line](#)

Find

The Find search function brings up the Find Text window where you can select the search criteria.

To search with Find

Select **Search | Find [Ctrl + F]**

The search is refined using the following settings:

Options

- Case sensitive
- Searches for the text in exact case that you enter it.
- Whole word only
- Searches for words only, or in other words, searches for a text string that is preceded and followed by a space or punctuation mark.
- Regular expression
- Specify whether to recognize [regular expressions](#) in the search string.

Scope

- Global
- Searches the entire text pane.
- Selected text
- Searches only the highlighted text.

Direction

- Forward
- Searches from the cursor to the end of the text pane.
- Backward
- Searches from the cursor to the top of the text pane.

Origin

- From cursor
- Searches from the cursor either forward or backward depending on the Direction setting.
- Entire scope
- Searches the entire text pane. If the Direction setting is forward, it starts at the top of the pane. If the Direction setting is backward, it starts at the bottom of the pane.

Related Topics

[Search Functions](#)

[Quick Find](#)

[Find Next](#)

[Replace](#)

[Goto Line](#)

Find Next

The Quick Find search function finds the next occurrence of the text search string that was previously searched for. The function is not available until after you have done an initial search.

To search with Find Next

Select **Search | Find Next [F3]**

Related Topics

[Search Functions](#)

[Quick Find](#)

[Find](#)

[Replace](#)

[Goto Line](#)

Replace

The Replace search function brings up the Replace Text window where you can enter

To replace text

Select **Search | Replace** [Ctrl + Alt + R]

Related Topics

[Search Functions](#)

[Quick Find](#)

[Find](#)

[Find Next](#)

[Goto Line](#)

Goto Line

The Goto Line search function brings up the Goto Line window where you can enter the line number that you want to move the cursor to. It is not necessary to have the line numbers displayed in the [gutter](#) to use this function, but it would certainly be helpful to see the line numbers in order to select the line number to move the cursor to.

To go to a specific line

Select **Search | Goto Line** [Ctrl + Alt + G]

Related Topics

[Search Functions](#)

[Quick Find](#)

[Find](#)

[Find Next](#)

[Replace](#)

Performance Monitor Overview

The Performance Monitor is a module to facilitate monitoring the performance of Adaptive Server. The Performance Monitor is based on the sophisticated build-in monitoring tables in Adaptive Server (15.0 or later), which provides database statistical and diagnostic information. You can easily understand the performance behavior of Adaptive Server by monitoring its run time statistics by means of snapshots, real time monitoring, and scheduled monitoring. The monitor result is presented graphically from the overall database performance down to the statistics of an individual process.

The Performance Monitor displays what is happening on the database. It helps to identify a CPU or I/O intensive server that may require some configuration tuning in the operating system or Adaptive Server. In addition, it also helps to illustrate that the tuning of application code and database design are where the most performance improvement can normally be gained. A missing index on a table or a poorly constructed SQL statement may cause poor performance on the database causing a chain reaction from high CPU, I/O, and Lock Waits to even more Dead Locks. The Performance Monitor can easily help to locate the symptom, but you still need the other modules for solving the performance problems, using the Index Advisor to locate a missing index or the SQL Optimizer to rewrite a poorly performing SQL statement.

The Performance Monitor monitors for the statistics that are consuming the most resources. It identifies where the top resource consumption occurs through various performance statistics and thereby filtering out unnecessary information so that performance problems are easier to pin point.

A daily monitoring of the performance of the database can be achieved by setting up a periodic capture time for retrieving the performance statistics. For example; to review the database performance on daily basis, a 15 minutes to 60 minutes refresh interval can provide a pretty clear picture of the system performance and resource consumption status; such as CPU, Devices IO, Network I/O, Lock/Dead Lock, etc. For troubleshooting on an extremely slow database, a frequent polling to the database may be too intrusive, so the manual refresh option which takes snapshots from the database may be more appropriate.

Related Topics

[Performance Monitor Window](#)

[Open the Performance Monitor](#)

[Start and Stop the Performance Monitor](#)

[Performance Monitor Functions](#)

[Performance Monitor Preferences](#)

Performance Monitor Privileges

Only users with the **mon_role** role have access to monitoring tables in Adaptive Server. You can provide extra role-based security by modifying the CIS proxy table definitions provided with the monitoring tables. For information about acquiring roles, see Chapter 11, "Managing User Permissions," in the Adaptive Server *Performance and Tuning: Monitor and Analyzing* manual. Because the Performance Monitor collects the performance statistics from the monitoring tables in Adaptive Server, version 15.0 or later is required.

Supported Monitor Tables

The Performance Monitor supports both dynamic performance information and static information from the following Adaptive Server monitoring tables:

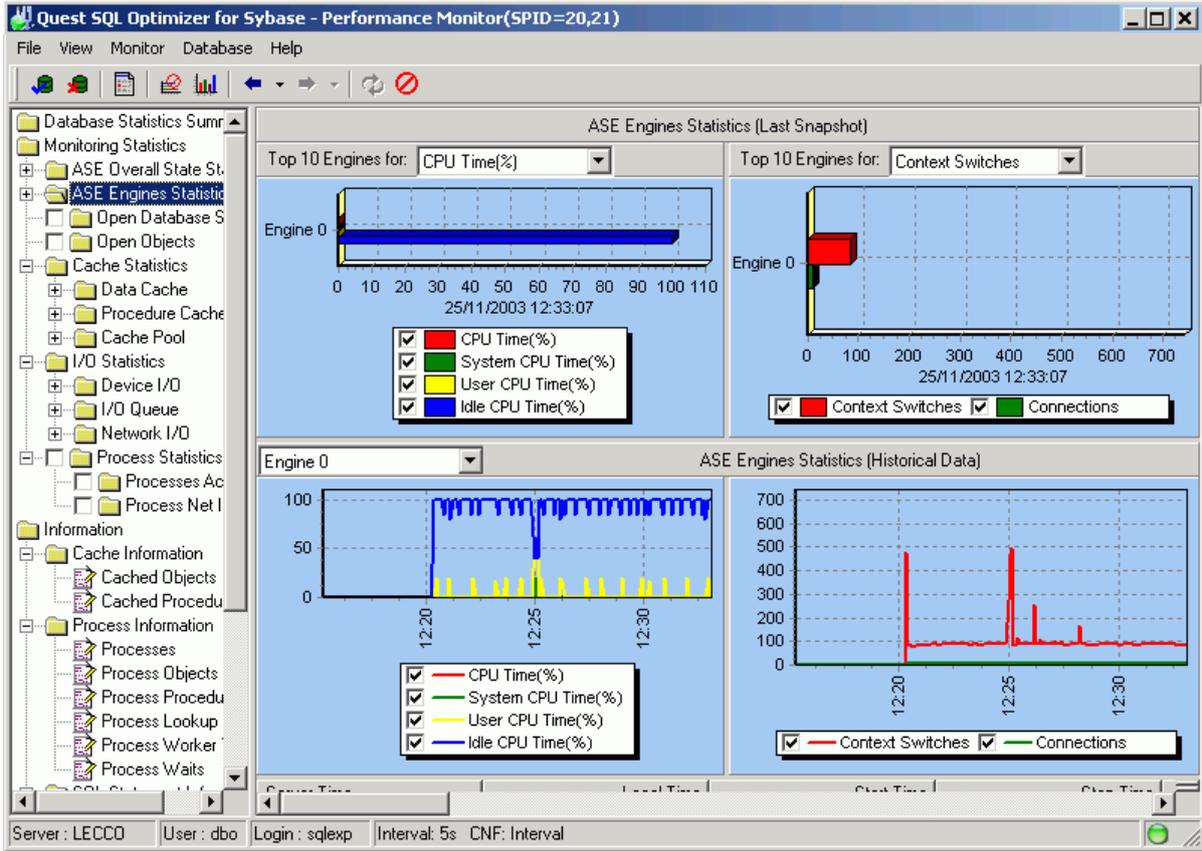
Table Name	Description
monState	Provides information regarding the overall state of the Adaptive Server.
monEngine	Provides statistics regarding Adaptive Server engines.
monDataCache	Provides statistics relating to data cache usage.
monProcedureCache	Provides server wide information related to cached procedures.
monOpenDatabases	Provides state and statistical information for databases that are currently in use (i.e. open databases).
monSysWorkerThread	Provides server-wide statistics about worker threads.
monNetworkIO	Provides server-wide statistics about network I/O.

monErrorLog	Provides the most recent error messages raised by Adaptive Server. The maximum number of messages returned can be tuned by use of the "errorlog pipe max messages" configuration option.
monLocks	Provides information for all locks that are being held, and those that have been requested, by any process, for every object.
monDeadLock	Provides information about the most recent deadlocks that have occurred. The maximum number of messages returned can be tuned by use of the "deadlock pipe max messages" configuration option.
monWaitClassInfo	Provides a textual description for all of the wait classes, e.g. "waiting for a disk read to complete". All wait events (see monWaitEventInfo table) have been grouped into the appropriate wait class.
monWaitEventInfo	Provides a textual description for every possible situation where a process is forced to wait for an event, e.g. "wait for buffer read to complete".
monCachedObject	Provides statistics for all objects and indexes that currently have pages cached within a data cache.
monCachePool	Provides statistics for all pools allocated for all caches.
monOpenObjectActivity	Provides statistics for all open objects.
monIOQueue	Provides device IO statistics broken down into data and log IO, for normal and temporary databases on each device.
monDeviceIO	Provides statistical information about devices.
monSysWaits	Provides a server-wide view of events that processes are waiting for.
monProcess	Provides information about processes that are currently executing or waiting.
monProcessLookup	Provides information enabling processes to be tracked to an application, user, client machine, etc.
monProcessActivity	Provides statistics about process activity.
monProcessWorkerThread	Provides information about process use of worker threads.
monProcessNetIO	Provides statistics about process network I/O activity.
monProcessObject	Provides statistical information about process object access.
monProcessWaits	Provides information about processes currently waiting for an event.
monProcessStatement	Provides statistics for currently executing statements.
monSysStatement	Provides statistics for the most recently executed statements. The maximum number of statement statistics returned can be tuned by use of the "statement pipe max messages" configuration option.
monProcessSQLText	Provides the SQL text that is currently being executed. The maximum size of

monSysSQLText	Provides the most recently executed SQL text. The maximum number of messages returned can be tuned by use of the "max SQL text monitored" configuration option.
monCachedProcedures	Provides statistics about all procedures currently stored in procedure cache.
monProcessProcedures	Provides information about procedures that are being executed.
monSysPlanText	Provides the most recently generated plan text. The maximum number of messages returned can be tuned by use of the "plan text pipe max messages" configuration option.

Performance Monitor Window

View Performance Monitor Window



The Performance Monitor window displays charts and information about the status of Adaptive Server and the performance of your database. It has two panes. The left pane is for selecting the database statistic that you would like to view. The right pane displays the charts and information about the selected statistic.

Left pane

The top-level headings for selecting the information and statistics to review are Database Statistics Summary, Monitor Statistics, and Information.

The charts and information displayed in the right pane are dependent upon what is selected in the left pane:

[Right pane for Database Statistics Summary](#)

[Right pane for Monitor Statistics](#)

[Right pane for Information](#)

The status information for the Performance Monitor is display on the [status bar](#) at the bottom of the window.

Related Topic

[Performance Monitor Overview](#)

Right Pane for Database Statistics Summary

The Database Statistics Summary in the Performance Monitor provides an overview of database performance. Important information is displayed on this screen and provides a high level view of the overall performance on CPU utilization for Top N Engines, Device I/O, Network I/O, Data Cache status, and Lock Waits information. Bottlenecks of those statistics can be spotted and you can drill down to and individual process, device, or SQL statement. You can drill down to specific items on the chart by double-clicking on a data point.

Related Topic

[Performance Monitor Window](#)

Right Pane for Monitor Statistics

The Monitor Statistics in the Performance Monitor displays various charts that correspond to the statistic selected in the left pane. You can drill down and [zoom in or out](#) on these charts.

ASE Overall State Statistics

Provides information regarding the overall performance state of Adaptive Server. Information includes number of lock waits which exceeded the lock wait threshold, number of dead locks, and number of connections.

At the bottom of the right pane is the Information Box that has some static information of overall performance state of Adaptive Server. To see this information, move the cursor right above the word "Information" until the cursor changes and you can drag and expand the Information Box.

ASE Engines Statistics

In a system with multiple CPUs, you can enhance performance by configuring Adaptive Server to run using multiple Adaptive Server engines. Each engine is a single operating system process that yields high performance when you configure one engine per CPU. In the Adaptive Server Engines Statistics, multiple engines can be monitored at the same time. Various CPU statistics are displayed on the latest snapshot and

historical line chart format. A number of definitions are introduced by the Performance Monitor that are not documented in the *Adaptive Server Performance and Tuning: Monitor and Analyzing* manual.

Statistic Definition

Statistic	Definition
CPU (%)	$(\text{SystemCPUTime} + \text{UserCPUTime}) / \text{CPUTimeEngine} * 100\%$
System CPU Time (%)	$\text{SystemCPUTime} / \text{CPUTimeEngine} * 100\%$
User CPU Time (%)	$\text{UserCPUTime} / \text{CPUTimeEngine} * 100\%$
Idle CPU Time (%)	$\text{IdleCPUTime} / \text{CPUTimeEngine} * 100\%$

Variables Definition

Variable	Definition
SystemCPUTime	Time (in seconds) the engine has executed system database services in last interval.
UserCPUTime	Time (in seconds) the engine has executed user commands in last interval.
CPUTimeEngine	Total time (in seconds) the engine has run in last interval.
IdleCPUTime	Time (in seconds) the engine was in idle spin mode in last interval.

Open Database Statistics

Provides state and statistical information pertaining to databases that are currently in use. Two dynamic statistical items are monitored for individual databases in the Open Database Statistics section: Append Log Requests and Append Log Waits. These items are displayed on a snapshot bar chart and a historical line chart. These statistics are not collected unless the checkbox in front of the option is checked. Since the enabling of this monitor option may result in the capture of a large amount of information from the database, it is not recommended to turn it on if you do not need to troubleshoot a current database performance problem.

Open Objects

For open database objects such as tables, indexes, views, and stored procedures, statistical information is consolidated by database, statistics item, and object type.

These statistics are not collected unless the checkbox in front of the option is checked. Since the enabling of this monitor option may result in the capture of a large amount of information from the database, it is not recommended to turn it on if you do not need to troubleshoot a current database performance problem.

Cache Statistics

Cache Statistics are used to monitor the Data Cache, Procedure Cache, and Cache Pool information. Most of the statistics displayed in this section are directly extracted from the monitoring tables, but the following items are calculations from these retrieved statistics:

Data Cache Hit Ratio for the data cache for the life of Adaptive Server.

Data Cache Hit Ratio(%) = LogicalReads * 100 / CacheSearches

Data Cache Hit Ratio for the data cache in the last interval.

Data Cache Delta Hit Ratio(%) = Delta_LogicalReads * 100 / Delta_CacheSearches

Procedure Cache Hit Ratio for the procedure cache for the life of Adaptive Server.

Procedure Cache Hit Ratio(%) = (Requests - Loads) * 100 / Requests

Procedure Cache Hit Ratio for the procedure cache in last interval

Procedure Cache Delta Hit Ratio(%) = (Delta_Requests - Delta_Loads) * 100 / Requests

Note: The "Delta_" in the above equations denotes that the difference in the statistics values for the last polling interval.

Normally, a Hit Ratio greater than 90% is considered healthy and indicates that no extra memory is necessary. But there are some situations that a large Hit Ratio is only contributed by a small piece of a procedure or from Data Pages resident in memory that have been executed million times during the day. So, carefully review these figures to understand the behavior your applications. Do not rely only on these figures to judge the healthiness of memory usage on your systems.

I/O Statistics

The I/O Statistic provides a compressive view of the I/O activity on Adaptive Server. It is broken down into Device I/O, I/O Queue and Network I/O.

Device I/O

The Top-N I/O devices are displayed on this monitor section to keep track of individual device I/O trends or the I/O devices which are most active on a real time basis. An intensive I/O device may need further investigation or reallocation of hard disks. Reads for user devices and temporary database reads/writes may result from poorly performing SQL statements.

I/O Queue

The I/O Queue statistics are broken down into data and log I/O for normal (User) and temporary databases on each device. I/O statistics include the information of I/O Operations and I/O Time. Too many I/O Operations and I/O Time on temporary databases means that there may be many temporary table operations in your application processes; or your SQL statements are using too many worktable operations or sorting operations. An Index or SQL syntax restructure may help to improve the problem. SQL Optimizer provides the SQL Optimizer and Index Advisor to help identify missing indexes and to optimize SQL.

Network I/O

Network IO statistics for both Received and Sent information are displayed in this monitor section. For a high Packet/Byte Received situation, you should consider converting your dynamic SQL or Transact-SQL into Stored Procedure calls to Adaptive Server. Thousands of dynamic SQL calls within a loop is normally the problem in most applications, it not only increases the network traffic, but also introduces extra workloads to Adaptive Server. If there are high values for Packet/Byte Sent, look to see if the applications are sending unnecessary data to the Client application. For example; in some online queries, the users may need only the first few records to review, a filtering criteria should be added to those applications to narrow the result set being retrieved from the database. Furthermore, some development tools, such as PowerBuilder, provide options to retrieve necessary data only if users want to see more by pressing a next-page button. Stored Procedures can be created in the SQL Worksheet module in SQL Optimizer.

Process Statistics

The Process Statistics provides very comprehensive information for all executing processes.

These statistics are not collected unless the checkbox in front of the option is checked. Since the enabling of this monitor option may result in the capture of a large amount of information from the database, it is not recommended to turn it on, if you do not need to troubleshoot a current database performance problem. When you are troubleshooting use **Manual Refresh Data** to take a snapshot, which provides very good insight on processes running status, such as Top-N CPU intensive processes. Processes that are blocked by other processes are marked with a special icon and the related blocking processes can also be displayed.

Related Topic

[Performance Monitor Window](#)

Right Pane for Information

The Information section in the Performance Monitor provides information about Cache, Processes, SQL Statements, Locks, Sys Worker Threads, Sys Waits and the Error Log.

Cache Information

The Cache Information section has the following information:

Statistic	Description
Cached Objects	Provides statistics for all objects and indexes with pages currently in a data cache.
Cached Procedures	Provides statistics for all procedures currently stored in the procedure cache.

Process Information

The Process Information section has the following information:

Statistic	Description
Processes	Provides detailed statistics about processes that are currently executing or waiting.
Process Objects	Provides statistical information regarding database objects that have been accessed by processes.
Process Procedures	Provides a list of all procedures that are being executed by processes.
Process Lookup	Provides information enabling processes to be tracked to an application, user, client machine, and so on.
Process Worker Thread	Provides information about process use of worker threads.
Process Waits	Provides a server-wide view of where processes are waiting for an event.

SQL Statement Information

This SQL Statement Information section provides information about currently executing SQL statements and recently executed SQL statement.

Statistic	Description
Executing SQL	Provides information for currently executing statements.
Executed SQL	Provides statistics pertaining to the most recently executed statements. The maximum number of statement statistics returned can be set with statement pipe max messages Adaptive Server configuration parameter.

Locks Information

Returns a list of all locks that are being held, and those that have been requested, by any process, for every object.

Sys Worker Threads

Provides server-wide statistics related to worker threads.

Sys Waits

Provides a server-wide view of where processes are waiting for an event.

Error Log

Returns the most recent error messages from the Adaptive Server error log. The maximum number of messages returned can be set with the **errorlog pipe max messages** Adaptive Server configuration parameter.

Related Topic

[Performance Monitor Window](#)

Status Bar

The status bar at the bottom of the Performance Monitor window has the following information.

Item	Description
Server	Adaptive Server name
User	Default user for the login account
Login	Login account name
Interval	Interval setting for retrieving the statistics from the monitoring tables from the Preferences settings.
CNF	The Chart Data Normalization Factor from the Preferences settings.
Monitor Status Icon	When the Monitor Status icon at the far right of the status bar is green  , the Performance Monitor is idle. When the Status icon at the far right of the status bar is red  , it is retrieving the statistics from the monitoring tables.

Related Topic

[Performance Monitor Window](#)

Open the Performance Monitor

To open the Performance Monitor window

Click .

Related Topic

[Performance Monitor Window](#)

Connect and Disconnect from Database

When the Performance Monitor window is opened from within SQL Optimizer, the current SQL Optimizer logon is used.

If you open Performance Monitor window outside of SQL Optimizer, you are prompted to logon to a server. You can disconnect from or reconnect to a server without closing the Performance Monitor window.

To connect to a different server

Click .

To disconnect from a server

Click .

View connection information

To find out if the server has the necessary parameter settings and your current logon has the necessary privileges needed to run the Performance Monitor, select **Database | View Connection Information**.

The Connection Information window confirms that your logon has the necessary the privileges to run the Performance Monitor or tells which privileges your logon is lacking under the **Privilege Information** tab.

Certain Adaptive Server configuration parameters are needed to run the Performance Monitor. The **Configuration Information** tab in the Connection Information window shows the current settings for these parameters and the parameter values that are necessary. If the current parameter value does not meet the required value, the line for configuration parameter is highlighted in red.

Related Topic

[Performance Monitor Overview](#)

Start and Stop the Performance Monitor

The Performance Monitor provides three ways to monitor the database performance: immediate monitoring, scheduled monitoring, and manual refresh.

The settings on the **General** tab in Preferences window determine whether the monitoring process begins immediately or at a scheduled time. The monitoring continues to collect the statistics at the [refresh interval](#) until you stop the monitoring or until the scheduled end time.

To start the monitoring processing for collecting the database performance statistics

Click .

To stop the monitoring process

Click .

The [manual refresh](#) of the statistics collects the statistics immediately and only collects them once.

Related Topic

[Performance Monitor Overview](#)

Manual Refresh Data

The **Manual Refresh Data** option is used to check the database performance on ad-hoc basis. The performance snapshot is taken once. This function provides a more flexible statistics retrieval method for troubleshooting a performance problem that is currently happening on the database. It only retrieves the statistics from the monitoring tables when you specifically do the refresh. It retrieves the statistics once and then waits until you manual refresh the data again.

To manually refresh data

Click .

You can set up the monitoring to automatically poll the monitoring tables at [regular intervals](#).

The collection of the Open Database, Open Objects, and the Process statistics are not collected unless the checkbox in front of the statistic is checked in the left pane. The enabling of the retrieval of the statistics for these monitoring options may result in the capture of a large amount of information from the database. Therefore it is not recommended to turn it on when you are using Auto Refresh to continuously monitor the activity of Adaptive Server. But when you need to troubleshoot a current database performance problem, enabling the retrieval of these statistics can help you to quickly identify the problem. Using **Manual Refresh Data** to take a snapshot can provide good insight on processes running, such as Top N CPU intensive processes information. Processes that are blocked by other processes are marked with a special icon and the related blocking processes can also be displayed.

Related Topic

[Performance Monitor Overview](#)

Hide and Show Console Tree

The display of the Console Tree [left pane] in the Performance Monitor window can be toggled between show and hide.

To hide/show the Console Tree

Click .

Display View for Performance Monitor Charts

You can switch the x and y axis for the charts for the Monitor Statistics in the Performance Monitor.

To display horizontal bar charts

Click .

To display vertical bar charts

Click .

Chart Drill Down

In the Performance Monitor, you can click a data point in a higher-level chart to drill down to a chart for that specific statistic.

Lock and Unlock Chart

When the Performance Monitor is capturing statistics, the display is updated every time the statistics are gathered. This means that you could be reviewing some statistics and they would change while you are viewing them. The **Lock Chart** function freezes the display with the current time slice but the Performance Monitor still captures the new information from the database.

To lock the current display

Click .

To unlock the display

Click .

Chart Zoom

You can zoom in on a selected range of data on a chart in the Performance Monitor.

To zoom-in on a specific portion of the chart

Click and hold down the left-hand mouse button to draw a rectangular field from top left to bottom right of the selected area of the chart. You can see a rectangle around the selected area. Release the left-hand mouse button. You can zoom-in more than once.

To zoom-out

Hold down the left-hand mouse button to draw a reverse rectangular field from bottom right to top left. Click hold down the left-hand mouse button anywhere in the chart and drag to the left. Release the left-hand mouse button to restore the chart to the last zoom-in scale.

Copy Chart

To copy a chart from the Performance Monitor

Right-click the chart and select **Copy**.

Print Chart

To print a chart from the Performance Monitor

Right-click the chart and select **Print**.

Save Chart

To save a chart to a file from the Performance Monitor

Right-click the chart and select **Save**.

Performance Monitor Functions

Below is a list of available functions within the Performance Monitor window.

Button or Menu	Function
View Menu 	Show Console Tree/Hide Console Tree
View Menu 	Lock Chart/Unlock Chart
View Menu 	Vertical Chart/Horizontal Chart
View Menu 	Previous Chart/Next Chart
Monitor Menu 	Manual Refresh Data

Monitor Menu	Start Monitor/Stop Monitor
	
Database Menu	Connect/Disconnect
	
Right-click Menu	Copy Chart
Right-click Menu	Print Chart
Right-click Menu	Save Chart

Related Topics

[Performance Monitor Overview](#)

[Performance Monitor Window](#)

Performance Monitor Preferences

Preferences allow the alteration of the default settings used by the Performance Monitor.

To change the preferences

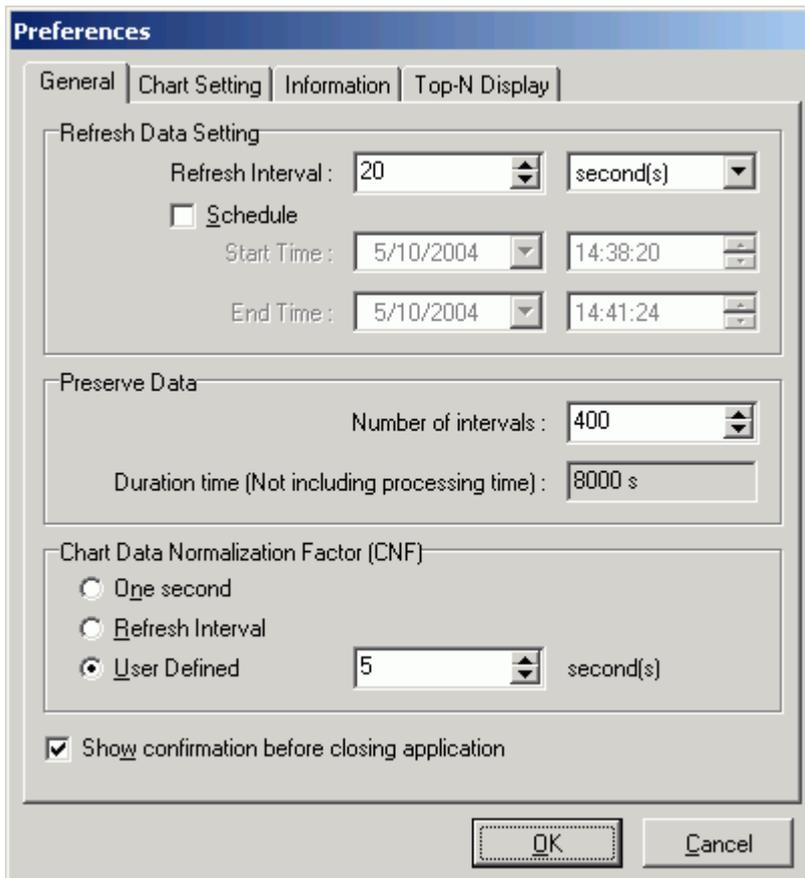
In the Performance Monitor module, click .

Related Topic

[Performance Monitor Overview](#)

General Tab

View Performance Monitor - General page



The **General** tab on the Preferences window for the Performance Monitor is for setting the statistics retrieval frequency for how often the statistics are refreshed, how many refresh intervals are saved and the normalization factor for accurate comparisons of the statistics.

Refresh Data Setting

The Refresh setting allows you to collect the database performance statistics at regular intervals once you have [started the monitoring process](#). If you have selected the **Schedule** option, then the monitoring starts and ends at the specified time. If the **Schedule** option is not selected, then the monitoring starts immediately and does not end until you stop it.

Refresh Interval (Default = 5 seconds)

The refresh interval specifies how often the database performance statistics are collected. It can be set to seconds, minutes, hours, or days. Since most of the statistics for Adaptive Server are cumulative counters stored in the columns of monitoring tables, the Performance Monitor keeps track of those statistics changes and presents the delta values for each time interval in the line charts, bar charts, and pie charts. This makes it easy to pinpoint the resource consumption peak time by reviewing those graphical presentations.

For example, you can check for database performance by taking snapshots of the database statistics for each 15 minutes by setting a Refresh Interval at 15 minutes with Preserve Data number of intervals set at 200. The Performance Monitor starts monitoring the database statistics until you manually stop monitoring. The Performance Monitor keeps up to 200 * 15 minutes = 3000 minutes (50 hours) of statistics data in memory for

review. If the monitoring time is longer than the reserve interval data time, the most current 50 hours of data is retained.

Schedule (Default = *cleared*)

The Schedule monitor function starts the monitor at a specified **Start Time** enabling the tracking of the performance of the database at a specific time. The scheduled monitoring uses the Refresh Interval setting to collect the statistics at regular intervals. The monitoring stops once the **End Time** is reached.

Preserve Data

When monitoring, the statistics are stored in memory for each Refresh Interval. All the statistics for each interval are kept in memory to facilitate the graphical plotting and data retrieval.

Number of intervals (Default = 200)

Specify the number of Refresh Intervals that are stored by the monitoring process for your review.

Duration time (Not including processing time) (Default = 1000 s)

This is a calculation presented in seconds using this formula: **Number of intervals * Refresh Interval**

Chart Data Normalization Factor (CNF)

Since the Refresh Interval time is adjustable by users for each monitor task, the statistical data is proportional to the selected Refresh Interval time. For example; a 10 minutes Refresh Interval may collect 100 writes on Master device, but 30 minutes refresh interval may have 300 writes on Master device. It would be easy to be misled by just looking at the result of 300 writes to judge them in comparison to the 100 writes. The Performance Monitor provides a normalization factor to help standardize the result statistics scale. The following is an example of using 5 minutes as a normalization factor to calculate the results for different Refresh Interval settings:

Monitor Time	Refresh Interval (RI)	Master Device Writes (MDW)	CNF	CNF Result = RI / MDW * CNF
09:00AM-11:00AM	10 minutes	100 writes	5 minutes	50 writes
01:00PM-03:00PM	30 minutes	360 writes	5 minutes	60 writes
04:00PM-06:00PM	60 minutes	540 writes	5 minutes	45 writes

With the Chart Data Normalization Factor, the result is standardized to 5 minutes and the comparison is more accurate. This example shows that the Monitor Time from 1:00PM to 3:00PM was more Write intensive on the Master device with 60 writes for every 5 minutes than the other time slices.

One second

Specify to use 1 second as the Chart Data Normalization Factor. This means that the statistics are shown as an average per second calculation. For example, using an Refresh Interval of 1 minute with 240 reads, the CNF for Reads is $240/60\text{seconds} = 4 \text{ Reads/per second}$.

Refresh Interval (Default)

Specify to use the Refresh Interval as the Chart Data Normalization Factor.

User Defined

Specify in seconds the Chart Data Normalization Factor.

The range of this setting is dependent upon the Refresh Interval measurement you select. The measurement is always in seconds.

Range for seconds: 1 - 60 (1 minute)

Range for minutes: 1 - 3600 (1 hour)

Range for hours: 1 - 216000 (60 hours)

Range for days: 1 - 4665600 (54 days)

Chart Data Normalization Factor text field

If the **User Defined** option is not selected, this seconds text field is dimmed and the calculation from the **One second** or **Refresh Interval** displays in this field.

Show confirmation before closing application (Default = *checked*)

Brings up a dialog field to confirm that you really want to exit the Performance Monitor.

Related Topics

[Performance Monitor Overview](#)

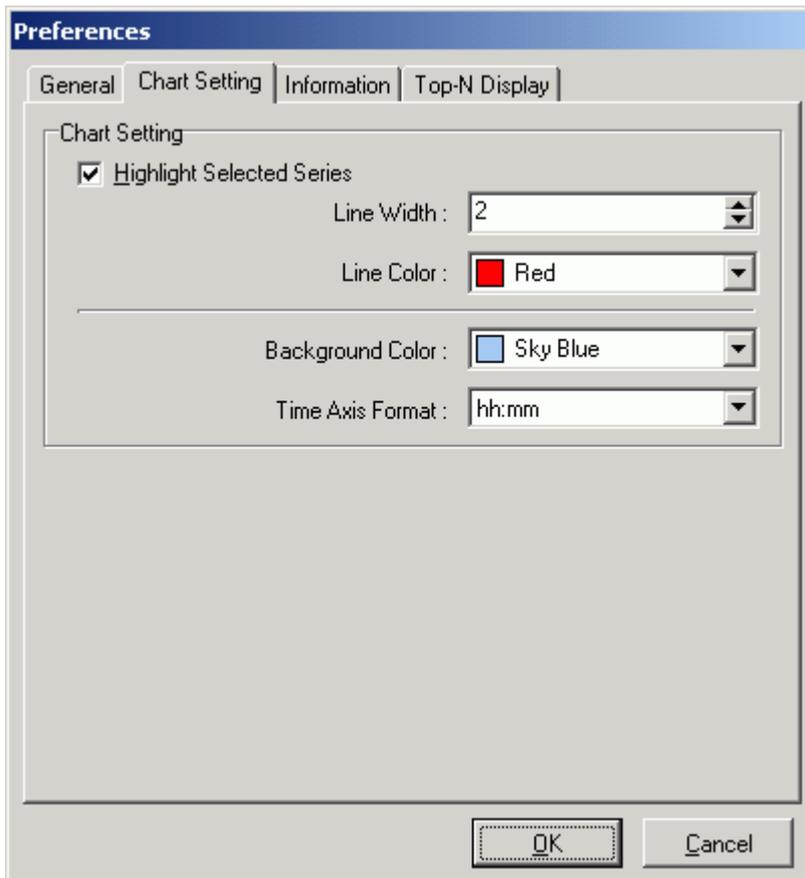
[Chart Setting Tab](#)

[Information Tab](#)

[Top-N Display Tab](#)

Chart Setting Tab

View Performance Monitor - Chart Setting page



The **Chart Setting** tab on the Preferences window for the Performance Monitor is for controlling the display of the charts.

Chart Setting

Highlight Selected Series (Default = *checked*)

The following settings affect the chart display of the selected statistic when you have drilled down to the bottom level of information.

Line Width (Default: 2 Range: 2 - 10)

Specify the width of the line for the statistic that you have selected in the left pane. This is used to enable the statistic selected in the left pane to stand out from the other statistics in the chart.

Line Color (Default = *Red*)

Select the color for the selected statistic from the drop-down list.

Background Color (Default = *Sky Blue*)

Select the color for the chart background from the drop-down list.

Time Axis Format (Default = hh:mm)

Select the format for displaying the date and time on the chart from the drop-down list.

Related Topics

[Performance Monitor Overview](#)

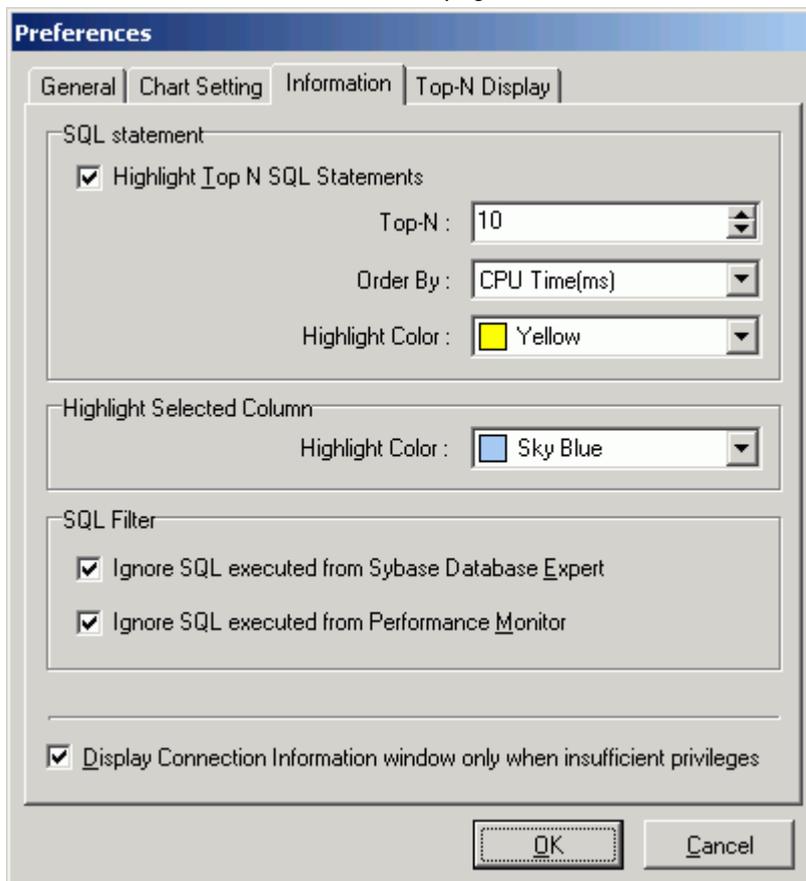
[General Tab](#)

[Information Tab](#)

[Top-N Display Tab](#)

Information Tab

View Performance Monitor - Information page



The **Information** tab on the Preferences window for the Performance Monitor determines which of the top SQL statements are highlighted, sets highlighting colors, and sets the automatic display of the connection information when logging on.

SQL statement

In the left pane of the Performance Monitor window, individual SQL statements are displayed when you drill down from Information to Process Information to the Executing SQL Statements. The following settings determine how many SQL statements are highlighted, by what criteria they are sorted, and the color that is used for highlighting.

Highlight Top N SQL Statements (Default = *checked*)

The top SQL statements are highlighted for you to quickly identify which SQL statements are consuming the most resources.

Top-N (Default = *10*)

Specify the number of SQL statements to be highlighted.

Order By (Default = *CPU Time(ms)*)

Select the statistic to sort the SQL statements from the drop-down field.

Highlight Color (Default = *Yellow*)

Select the color from the drop-down field.

Highlight selected column

When you drill down to the lowest level of the charts, the data from the table displays in a grid underneath the chart. The column for the statistic that is selected in the left pane is highlighted.

Highlight Color (Default = *Sky Blue*)

Select the color from the drop-down field.

SQL Filter (executing SQL only)

Ignore SQL executed from SQL Optimizer (Default = *checked*)

Specify to exclude the SQL statements that are generated by any of the modules in SQL Optimizer but the Performance Monitor.

Ignore SQL executed from Performance Monitor (Default = *checked*)

Specify to exclude the SQL statements that are generated by the Performance Monitor.

Display Connection Information window only when insufficient privileges (Default = *checked*)

Specify to display the Connection Information window each time you logon if the user logon does not have the database privileges needed for the modules.

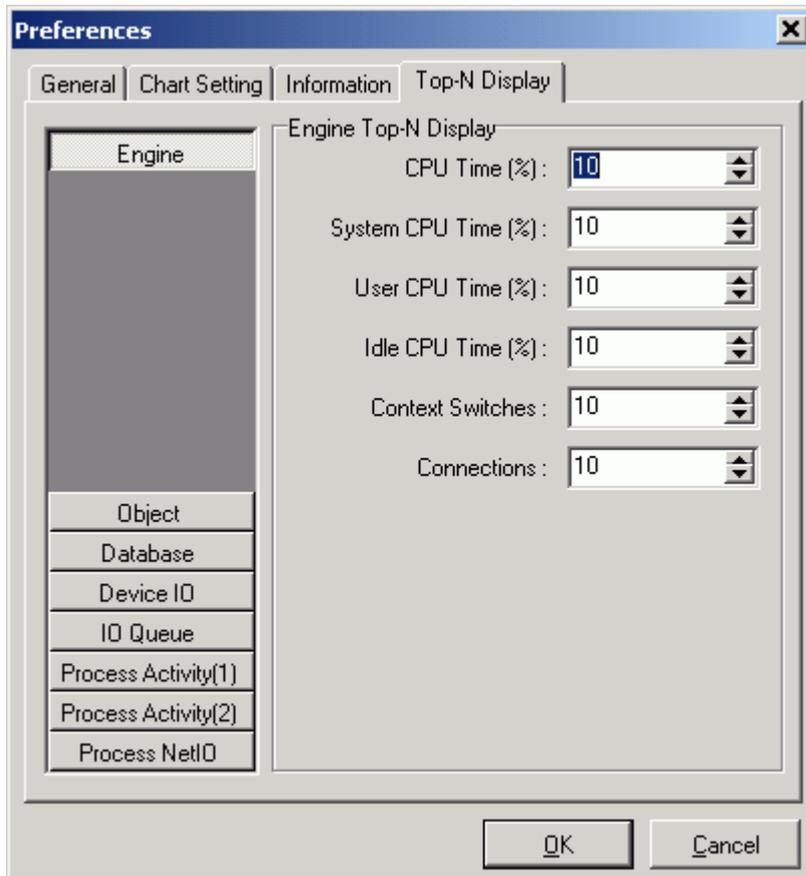
Related Topics

[Performance Monitor Overview](#)

[General Tab](#)
[Chart Setting Tab](#)
[Top-N Display Tab](#)

Top-N Display Tab

View Performance Monitor - Top-N-Display page



The **Top-N Display** tab on the Preferences window for the Performance Monitor is for determining the percent of each statistics that is retrieved from the monitoring tables.

To select only the top statistics

Select the button on the left for the statistics information and fill in the values on the right for each individual statistic.

Please reference the *Adaptive Server Performance and Tuning: Monitor and Analyzing* manual for the meaning of each statistic.

Related Topics

[Performance Monitor Overview](#)

[General Tab](#)

[Chart Setting Tab](#)

[Information Tab](#)

SQL Inspector Overview

The SQL Inspector identifies SQL statements with performance problems by extracting SQL statements and SQL performance statistics from the QP Metrics (Adaptive Server 15 or later). The SQL Inspector graphically displays and compares SQL activity statistics to diagnose performance bottlenecks. With the SQL Inspector you can identify the top N most resource intensive SQL statements.

With the SQL Inspector, you can schedule the date and time and the frequency that the SQL statistics are collected in order to identify the impact of SQL activities on database performance. By displaying the statistics on a graph you can see which SQL statements are consuming the most database resources, hence identifying problematic SQL statements quickly and efficiently.

The SQL Inspector window provides a work area for you to add and delete Inspectors, storing information on the Inspector name and description, number of SQL captured, start time, end time, and duration. It offers a way to view the statistics from the SQL statements captured in charts and a grid format. It provides a view of the top N worst performing SQL statements. SQL statements can be sorted by different AND/OR conditions of SQL performance statistics such as number of executions, CPU time, logical reads, physical reads, and others.

SQL statements collected in Inspector files can be also be analyzed with the [SQL Scanner](#) module. Once you have identified potentially problematic SQL statements you can send the SQL from the SQL Inspector to the [SQL Optimizer](#), [Index Advisor](#), the [SQL Worksheet](#), or save them to the [SQL Repository](#).

Note: This module is available for Adaptive Server 15 and later and requires that the [Adaptive Server monitoring tables are installed](#).

Or, if you have Adaptive Server 15.0 or later, you can retrieve the SQL statements and performance statistics using QP Metrics if you have access to the sysquerymetrics view.

Related Topics

[SQL Inspector Privileges](#)

[Add/Modify Inspector](#)

[Performance Statistics - Monitoring Tables](#)

[Performance Statistics - QP Metrics](#)

[Inspect](#)

[View and Add Charts](#)

[SQL Inspector Functions](#)

SQL Inspector Privileges

To use the SQL Inspector module you must have access to the sysquerymetrics view in Adaptive Server 15.0 or later.

You must also set up specific [configuration parameters](#) in Adaptive Server.

Related Topic

[SQL Inspector Overview](#)

Performance Statistics - Monitor Tables

Each SQL statement captured from the monitoring tables has the following performance statistics:

Statistics	Description
CPU Time	Number of milliseconds of CPU used by the statement.
CPU Time Per Execution	Ratio between the number of milliseconds of CPU used by the statement and the number of executions.
Executions	Number of times the statement was executed during the monitoring period.
Logical Reads	Number of buffers read from cache.
Logical Reads Per Execution	Ratio between the number of buffers read from cache and the number of executions.
Mem Usage (KB)	Number of kilobytes of memory used for the execution of the statement.
Mem Usage (KB) Per Execution	Ratio between the number of kilobytes of memory used for the execution of the statement and the number of executions
Packets Received	Number of network packets received by Adaptive Server.
Packets Received Per Execution	Ratio between the number of network packets received by Adaptive Server and the number of executions.
Packets Sent	Number of network packets sent by Adaptive Server.
Packets Sent Per Execution	Ratio between the number of network packets sent by Adaptive Server and the number of executions.
Pages Modified	Number of pages modified by the statement.
Pages Modified Per Execution	Ratio between the number of pages modified by the statement and the number of executions.
Physical Reads	Number of buffers read from disk.
Physical Reads Per Execution	Ratio between the number of buffers read from disk and the number of executions.
Wait Time	Number of milliseconds the task has waited during execution of the statement.
Wait Time Per Execution	Ratio between the number of milliseconds the task has waited during execution of the statement and the number of executions.

Related Topics

Performance Statistics - QP Metrics

Each SQL statement captured from the QP Metrics has the following performance statistics:

Statistics	Description
Execution time	How long it takes to execute the SQL statement on the CPU displayed in milliseconds.
Elapsed time	This is a total of the Execution time and the time it takes to parse, compile, and optimize the SQL statement displayed in milliseconds.
Logical IO	Total logical IO reads during the execution of the SQL statement.
Physical IO	Total physical IO reads during the execution of the SQL statement.
Execution count	Number of times the SQL statement has been executed.
Abort count	Number of times the resource governor terminated the SQL statement because a resource limit was surpassed.

The execution time, elapsed time, logical IO, and physical IO statistics each of have three values taken from the number of times the SQL statements was executed: the minimum, the maximum, and the average.

Related Topics

Data Directory Setting

The SQL Inspector data directory is where the data files are created while executing the Inspect function. The data directory path is set in the Preferences window.

1. Click .
2. Select the **Directory Setup** tab.

The default setting for this directory is:

C:\Documents and Settings\User\Application Data\Quest Software\Quest SQL Optimizer for SAP ASE\DATA
Changes to this directory cannot be made while the SQL Inspector is active.

Note: It is advisable not to change the data directory after you have used this function, as files created during inspecting are kept in this directory.

Related Topic
[SQL Inspector Overview](#)

Open SQL Inspector

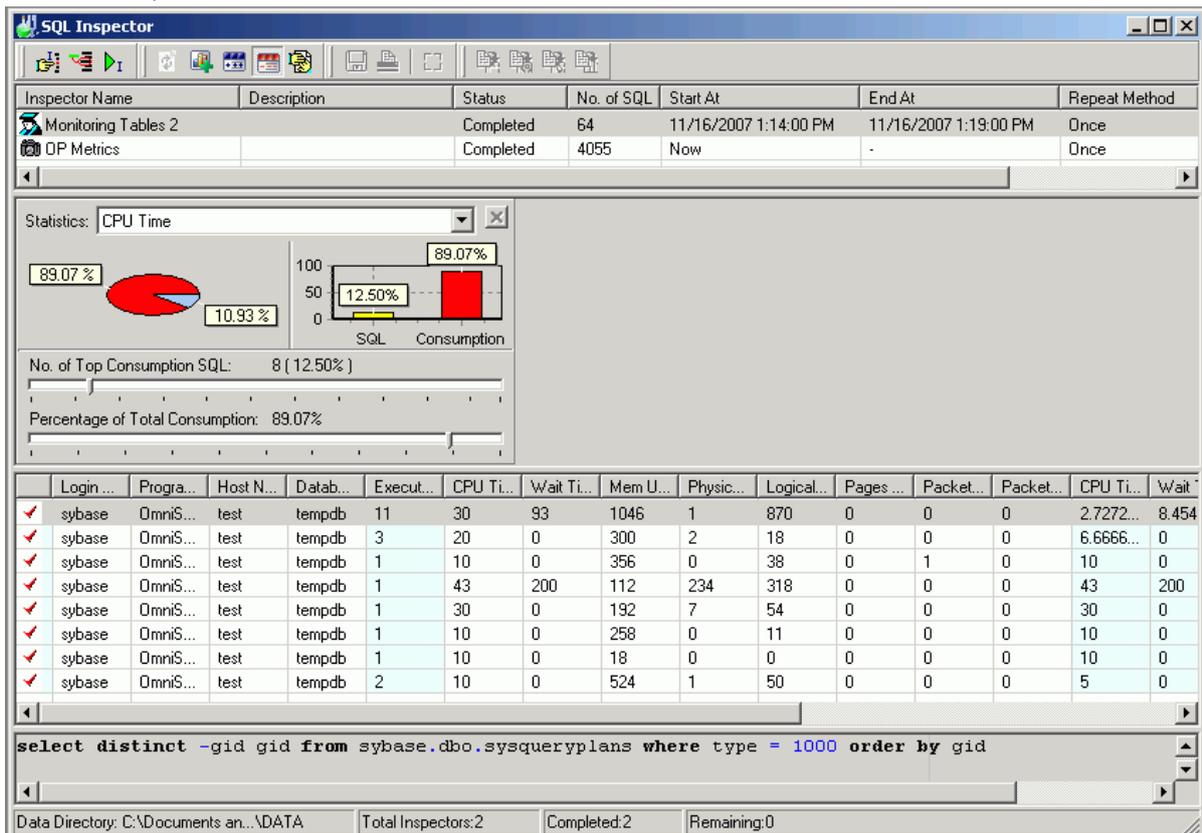
To open the SQL Inspector window

Click 

Related Topic
[SQL Inspector Window](#)

SQL Inspector Window

View SQL Inspector Window



The screenshot shows the SQL Inspector window with the following components:

- Inspector List:** A table listing monitoring tasks.
- Statistics Graphs:** A pie chart and a bar chart showing CPU Time and SQL Consumption.
- Statistics pane:** Text showing 'No. of Top Consumption SQL: 8 (12.50%)' and 'Percentage of Total Consumption: 89.07%'.
- SQL Text pane:** A query: `select distinct -gid gid from sybase.dbo.sysqueryplans where type = 1000 order by gid`
- Status bar:** Shows 'Data Directory: C:\Documents and Settings\... \DATA', 'Total Inspectors: 2', 'Completed: 2', and 'Remaining: 0'.

Inspector Name	Description	Status	No. of SQL	Start At	End At	Repeat Method
Monitoring Tables 2		Completed	64	11/16/2007 1:14:00 PM	11/16/2007 1:19:00 PM	Once
OP Metrics		Completed	4055	Now	-	Once

Login	Progra...	Host N...	Datab...	Execut...	CPU Ti...	Wait Ti...	Mem U...	Physic...	Logical...	Pages ...	Packet...	Packet...	CPU Ti...	Wait
✓ sybase	OmniS...	test	tempdb	11	30	93	1046	1	870	0	0	0	2.7272...	8.454
✓ sybase	OmniS...	test	tempdb	3	20	0	300	2	18	0	0	0	6.6666...	0
✓ sybase	OmniS...	test	tempdb	1	10	0	356	0	38	0	1	0	10	0
✓ sybase	OmniS...	test	tempdb	1	43	200	112	234	318	0	0	0	43	200
✓ sybase	OmniS...	test	tempdb	1	30	0	192	7	54	0	0	0	30	0
✓ sybase	OmniS...	test	tempdb	1	10	0	258	0	11	0	0	0	10	0
✓ sybase	OmniS...	test	tempdb	1	10	0	18	0	0	0	0	0	10	0
✓ sybase	OmniS...	test	tempdb	2	10	0	524	1	50	0	0	0	5	0

The SQL Inspector window consists of the Inspector List pane, Statistics Graphs panes, Statistics pane, SQL Text pane, and status bar.

Inspector List pane

The Inspector List pane stores information on each Inspector. Information includes:

Item	Description
Inspector Name	User-defined unique Inspector name.
Description	Description of the Inspector defined by the user.
Status	Current status on the Inspector. This column will remain blank until inspecting starts.
No. of SQL	Total number of SQL statements retrieved from the monitoring tables.
Start At	Date and time of when the inspecting process starts.
End At	Date and time of when the inspecting process ends.
Repeat Method	The frequency at which the inspection process captures SQL.
Last Modified	Date and time of when the Inspector was last modified.

Statistics Graphs pane

Displays charts of SQL performance according to the selected statistic in the drop-down list. Use the 2 sliders to select the SQL with the top resource consumption. The **No. of Top Consumption SQL** slider selects the number of SQL and the **Percentage of Total Consumption** slider selects the SQL statements based on the selected percentage of top resource consumption.

Statistics pane

Displays the statistics for each SQL statement retrieved from the chart selections.

The red checkmarks ✓ in the left column indicate those SQL statements that fall in the criteria of the selected chart. If you selected AND as the mode to join multiple charts, the checkmarks are displayed in every row because every single SQL statement satisfies all the charts. If you selected OR as the mode to join multiple charts, when you click one of the charts, the red checkmarks indicate the SQL statements that correspond to the selection criteria for that chart only.

Monitor Table statistics

The system resource statistics extracted from the monitoring tables include: executions, CPU time, wait time, memory usage, physical reads, logical reads, pages modified, packets sent, packets received, CPU time per execution, wait time per execution, memory usage per execution, physical reads per execution, logical reads per execution, pages modified per execution, packets sent per execution, and packets received per execution. The login, program, host, and database names are included.

QP Metrics statistics

The system resource statistics extracted from the QP Metrics include: execution time, elapsed time, logical IO, physical IO, Execution count, and abort count.

SQL Text pane

Displays the SQL text for the statistic that is selected in the Statistics pane.

Status Bar Information

The status bar consists of the following information:

Item	Description
Data Directory	Directory path shows where the data files produced during inspecting are saved. You can define the path of your data directory in the Preferences window. While inspecting, a progress bar will be shown to indicate the inspecting progress.
Total Inspectors	Total number of Inspectors.
Completed	Total number of Inspectors already executed.
Remaining	Total number of Inspectors that have not been executed.

Related Topic

[SQL Inspector Overview](#)

[Open SQL Inspector](#)

[Add/Modify Inspector](#)

[Inspect](#)

[SQL Inspector Functions](#)

Add/Modify Inspector

To add an Inspector

Click .

To Modify an Inspector

1. Select the Inspector row.

2. Click .

Note: If you have already executed the Inspector, modifying the Inspector criteria erases any existing information.

General Information

The General Information page of the Add/Modify Inspector wizard consists of the following settings:

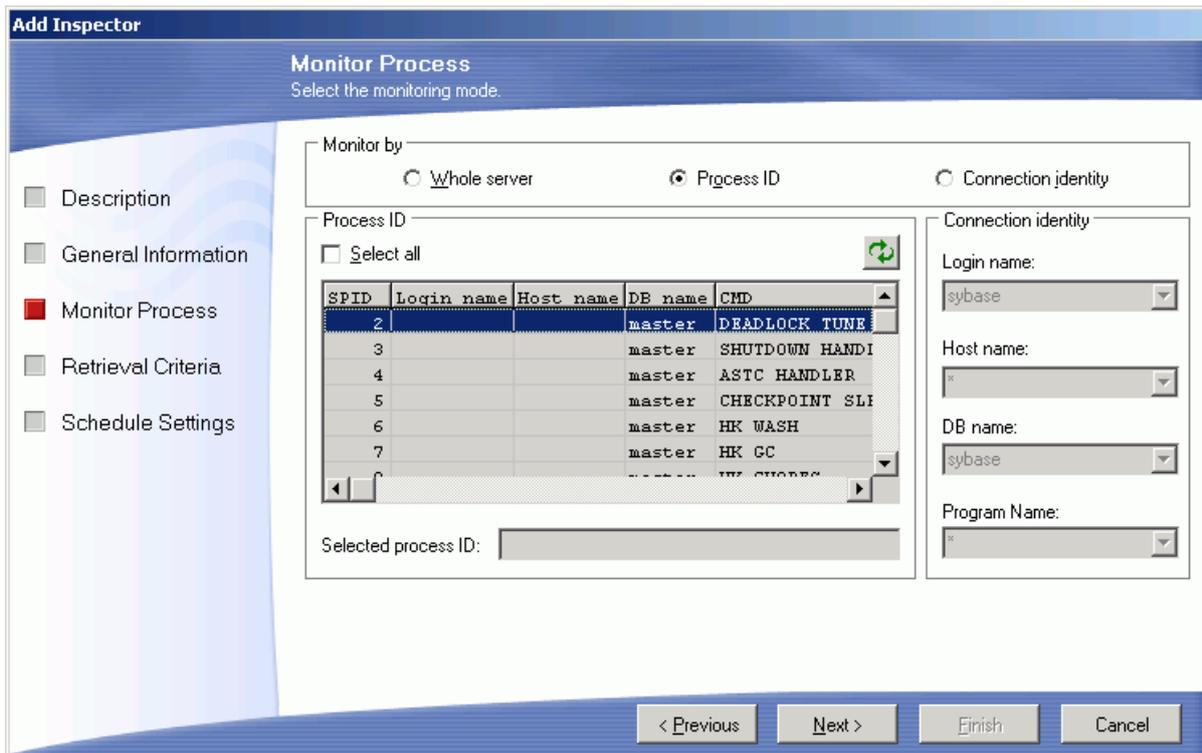
Item		Description
Inspector name		Enter a name for the Inspector.
Description		Enter a description for the Inspector.
Last modified		The date and time the Inspector was last modified.
Inspecting Approach	Monitor Tables (ASE 15 or later)	Specify to extract the SQL statements and transact SQL from the Adaptive Server monitoring table. The monitoring tables are available in Adaptive Server 15 or later.
	QP Metrics (ASE 15 or later)	Specify to extract the SQL statement from QP Metrics (sysquerymetrics view). The QP Metrics are available in Adaptive Server 15 or later.

Related Topic

[Add/Modify Inspector](#)

Monitor Process (Monitoring Tables)

View Add Inspector - Monitor Process Page



The Monitor Process page of the Add/Modify Inspector wizard consists of the following settings:

Item	Description
Whole server	Monitor all SQL statements currently running from the database server.
Process ID	Monitor all SQL statements currently running which have the specified process ID (spid). The Process ID grid displays information about all current users and processes of the database server, displaying the process ID (spid), login name, server name, database name and the command type currently executing. Click the row of the grid to select the process ID you want to monitor. To select all the process IDs, click the Select all checkbox.
Connection identity	Monitors all currently running SQL statements with the specified login name, host name, database name, and/or program name.

Related Topic

[Add/Modify Inspector](#)

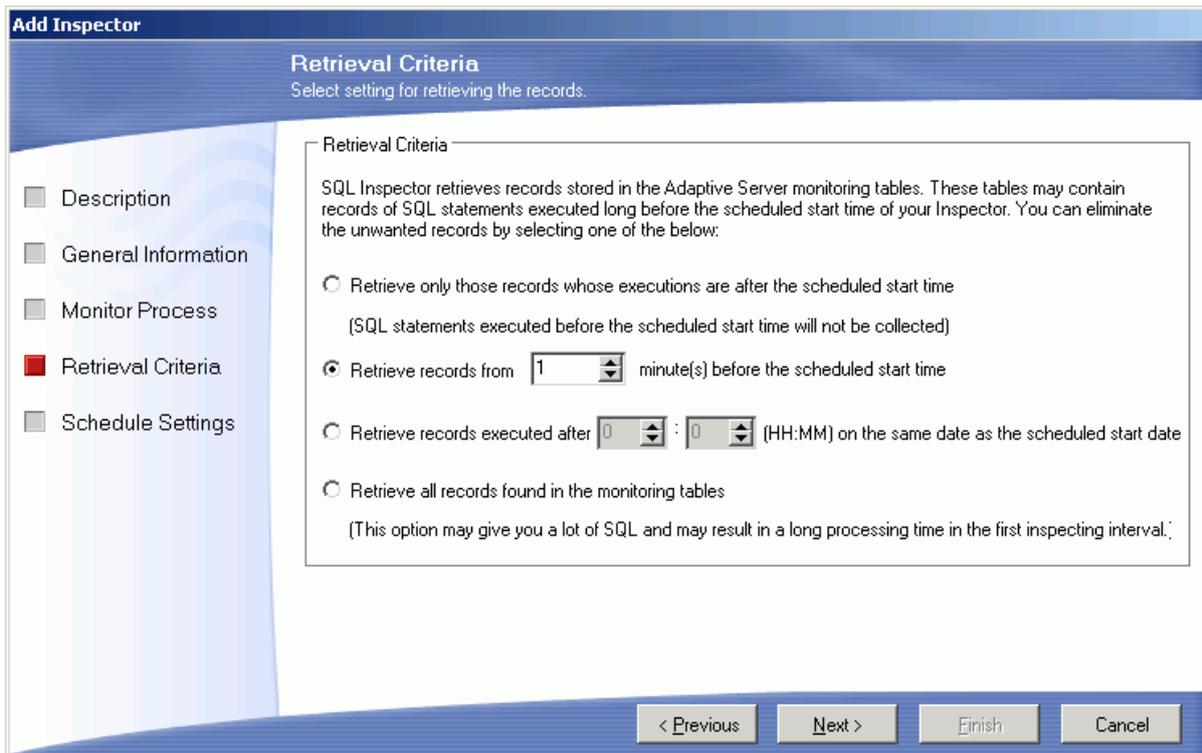
[Monitor Process \(Monitoring Tables\)](#)

[Retrieval Criteria \(Monitoring Tables\)](#)

[Schedule Settings \(Monitoring tables\)](#)

Retrieval Criteria (Monitoring Tables)

View Add Inspector - Retrieval Criteria Page



When using the monitoring tables, the Retrieval Criteria page of the Add/Modify Inspector wizard consists of the following settings:

Item	Description
Retrieve only those records whose executions are after the scheduled start time	Specify that SQL statements executed before the Inspector starts will not be retrieved. All the records are retrieved from the start time and forward.
Retrieve records from <i>nn</i> minute(s) before the scheduled start time (Range 1 to 99999 minutes (1666 hours or 69 days))	Specify that all records from a certain number of minutes before the Inspector starts are retrieved.
Retrieve records executed after (HH:MM) on the same date as the scheduled start date	Specify that only records from a specific time be retrieved.
Retrieve all records found in the monitoring tables	Specify that all records are retrieved when the Inspector starts. If you have a large volume of SQL statements in the monitoring tables, this option can take a long time to process the first inspecting interval.

The Adaptive Server monitoring tables may contain a large number of records when the Inspector starts. Therefore you may not want to retrieve all the existing records from the tables when the Inspector starts. The Retrieval Criteria page of the Add/Modify Inspector wizard allows you to specify which records are retrieved as the Inspector starts.

Related Topic

Schedule Settings (Monitoring Tables)

(View Add Inspector - Schedule Settings Page)

The Schedule Settings page of the Add/Modify Inspector wizard consists of the following settings:

Item	Description
Start date	Specifies the inspection start date.
End date	Specifies the inspection end date.
Repeat method	Inspect only once.
Once	
Day(s) of the Week	Inspect on the selected days of every week. Select one or more days
Week(s) of Month	Inspect on the selected week. Select the week or weeks of the month and the day of the week for the Inspector to run.
Day of Month	Select the day of the month for the Inspector to run.
Run at (HH:MM) (Default = current time)	Specify the starting time in hours and minutes.

Until	Time (Default = Start Time + 5 minutes)	Specify the ending time.
	Duration (Default = 5 minutes)	Specify the length of time the Inspector runs in hours and minutes.
	Interval (Default = 15000 milliseconds which is 15 seconds)	Specify how frequently the Inspector collects the statistics in milliseconds.

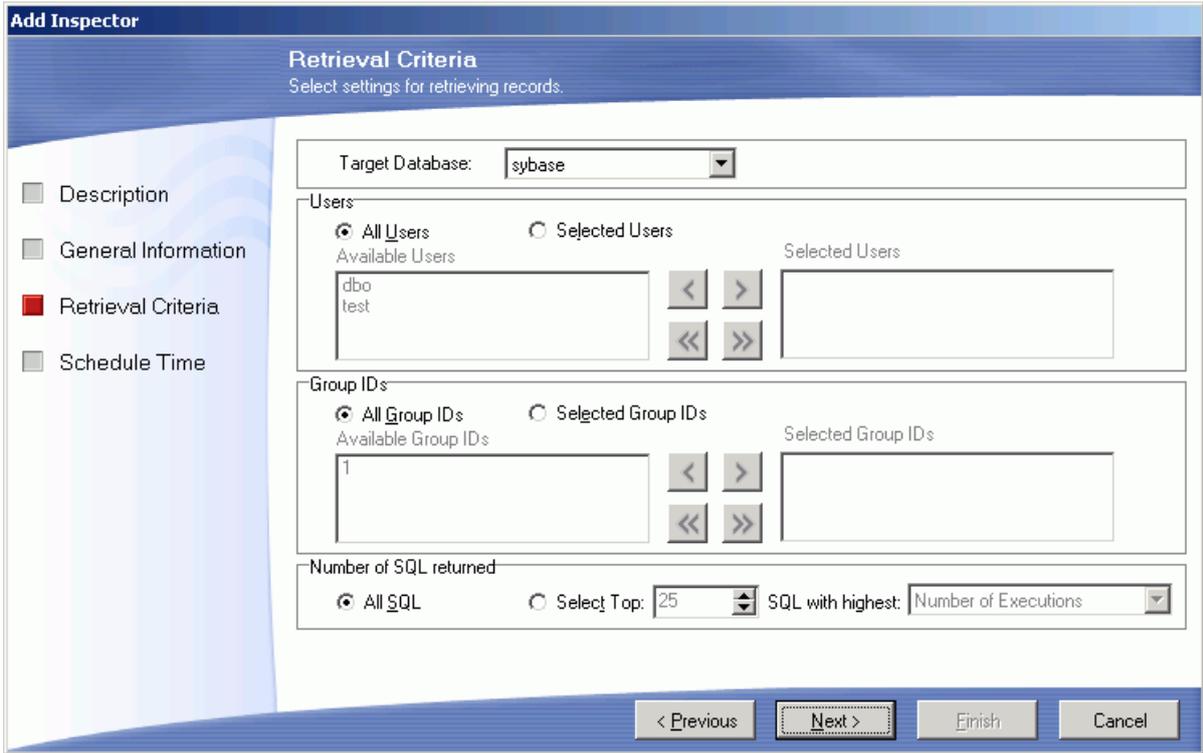
Notes:

- Decreasing the interval time may affect database server performance and network traffic.
- [Ad-hoc inspecting](#) does not use the start, until or duration time. The Inspector process starts instantly and is terminated by clicking **Abort Inspector** .

Related Topic
[Add/Modify Inspector](#)

Retrieval Criteria (QP Metrics)

View Add Inspector - Retrieval Criteria Page



When using the QP Metrics, the Retrieval Criteria page of the Add/Modify Inspector wizard consists of the following settings:

Item		Description
Target Database		Specify the database from which to retrieve the SQL statements and the run time statistics. This information is retrieved from the <i>database.dbo.sysquerymetrics</i> view.
User	All User	Specify to retrieve the SQL statements and Transact SQL executed by all users.
	Selected User	Select the specific user from the Available User ID column
Group IDs	All Group IDs	Specify to retrieve the SQL statements and transact SQL saved in all groups.
	Selected Group IDs	Select the specific group IDs from the Available Group ID column.
Number of SQL returned	All SQL	Retrieve all SQL from the sysquerymetrics view.
	Select Top "value" SQL with highest "criteria"	Select the number of SQL you want to be retrieved and the criteria for select those SQL statements from the drop-down boxes.

Related Topic

[Add/Modify Inspector](#)

[General Information](#)

[Schedule Time \(QP Metrics\)](#)

Schedule Time (QP Metrics)

[View Add Inspector - Schedule Time Page](#)

The Schedule Time page of the Add/Modify Inspector wizard consists of the following settings:

Item	Description
Start inspecting now	Specify to start the inspection as soon as you click the Inspect  button.
Start inspecting at date Time (HH:MM)	Specify the date and time you want the inspection to run.
Start inspecting in (HH:MM)	Specify the time to start the inspection in hours and minutes.
Schedule on	Shows the date that you scheduled to run the inspection.

Related Topic

[Add/Modify Inspector](#)

[Retrieval Criteria \(QP Metrics\)](#)

Inspect

After adding or modifying an Inspector in the SQL Inspector window, SQL statements and statistics from the monitoring tables are retrieved by executing the **Inspect** function.

To start the inspecting process

Click .

If the start time of the Inspector has not been reached, the SQL Inspector waits until it is time to begin. During and at the end of the inspecting, information is updated on the SQL Inspector window. Inspect terminates automatically once the end time is reached, except for ad-hoc inspecting. The ad-hoc inspecting process has no ending time so it must be terminated manually.

To abort the inspecting

Click .

Note: If you have already executed the Inspect function for an Inspector, re-executing the Inspect function will erase all existing information.

Related Topics

[Ad-hoc Inspect](#)

[SQL Inspector Overview](#)

[SQL Inspector Window](#)

Ad-hoc Inspect

Ad-hoc inspect allows the retrieval of the SQL performance statistics without considering or setting the scheduled start date and time.

To start an ad-hoc inspection

1. Mark an Inspector by clicking the Inspector row. A green checkmark  is prefixed to the Inspector Name column to indicate that the Inspector is marked.
2. Right-click and select **Ad-hoc Inspect**.

The process will not stop until you click .

Abort Inspect

After the Inspect function has been executed from the SQL Inspector window, it can be aborted at any time.

To abort the inspecting

Click .

It may take several seconds for all the events to close down.

Related Topics

[Inspect](#)

[Ad-hoc Inspect](#)
[SQL Inspector Overview](#)
[SQL Inspector Window](#)

Scan Inspector

The SQL statements collected by the Inspector can be scanned in the SQL Scanner to classify the SQL statements to help identify which SQL statements are likely to be causing performance problems.

To scan an Inspector from the SQL Inspector window:

1. Mark an Inspector by clicking the Inspector row.
2. Right-click and select **Scan**.

The selected Inspector is copied to the SQL Scanner window for scanning. If the SQL Scanner window is not opened then you will need to first select or create a job group from the Group Manager window. Once added, the scanning starts automatically.

Related Topics

[SQL Scanner Overview](#)
[Scan Inspected SQL](#)
[SQL Inspector Overview](#)
[SQL Inspector Window](#)

Find Inspector

To search for a specified text in all Inspectors

1. Select **Inspector | Find Inspector**.
2. Enter the search text string
3. Click **Find**.

Related Topics

[SQL Inspector Overview](#)
[SQL Inspector Window](#)

Delete Marked Inspector

To delete an Inspector

1. Click the Inspector row.
2. Press **Delete**.

Note: Only one Inspector can be deleted at a time.

Related Topics

[SQL Inspector Overview](#)

[SQL Inspector Window](#)

View Inspector Properties

To view the Inspector properties

Right-click an Inspector and select **Properties**.

Review the following for additional information:

Item	Description
Name	Inspector name.
Description	Inspector description provided by user.
Creation date time	Date and time the Inspector was created.
Repeat method	The frequency that the SQL statements are retrieved from the Adaptive Server monitoring tables.
Inspecting Approach	Indicates whether the SQL was retrieved from the Monitoring Tables or the QP Metrics.
Size	Inspector data file size.
Status	Inspector status.
Version	Version number for the program.
No. of SQL	Number of SQL retrieved.
Data directory	The directory where the Inspector files are kept.

Related Topics

[SQL Inspector Overview](#)

Place Bookmarks in SQL Inspector Window

To place or remove a bookmark on the SQL Inspector window:

1. Mark an Inspector by clicking the Inspector row.
2. Right-click and select **Add/Remove Bookmark**.

Multiple bookmarks can be placed in the SQL Inspector window.

Related Topics

[SQL Inspector Overview](#)

[SQL Inspector Window](#)

Create Benchmark Factory Import File

All the SQL statements can be saved in a text file from the SQL Scanner, the SQL Repository, the SQL Inspector, and the SQL Optimizer windows. These SQL statements can then be imported into Benchmark Factory 4.6 or later.

To create a file to import into Benchmark Factory

1. Right-click and select **Create Benchmark Factory Import File**.
2. Select the specific SQL statements that you want to save.
3. Enter the filename and select the file location.

View and Add Charts

When you first select an Inspector in the SQL Inspector window, one chart group showing the CPU time displays in the top pane with the following:

Statistics

Select the statistic to display on the chart from the drop-down field.

No. of Top Consumption SQLs and Percentage of Total Consumption sliders

These two sliders work in conjunction with each other. If you select the number of SQL statements and Transact-SQL using the No. of Top Consumption SQLs slider, the Percentage of Total Consumption slider is adjusted to show the percentage of the SQL statements that correspond to the number of selected SQL.

If you select the Percentage of Total Consumption, the corresponding number of SQL is shown in the No. of Top Consumption SQL's slider.

The SQL statements and Transact-SQL that are shown in the statistics pane are determined by the statistics that is selected for each chart and by the selection of the AND or OR criteria for joining the SQL statements based on the selected statistics.

Add another chart

Multiple charts are used to narrow or increase the number of SQL statements and statistics that can be viewed.

To add another chart

1. Click .
2. Select the statistic from the field at the top of the chart.
3. Adjust the No. of Top Consumption SQLs or Percentage slider.

Select the SQL Filtering Criteria

Two options are provided at the left of the window for toggling the filtering criteria for displaying SQL statements and Transact-SQL.

Show SQL Satisfying All Charts

The SQL statements can be selected for display using the AND operator to require that all SQL statements must be in the No. of Top Consumption SQLs for each charted performance statistic.

To select only those SQL statements in all charts

Click .

Show SQL Satisfying Any Chart

The SQL statements are selected for display using the OR operator to allow any SQL statement in the No. of Top Consumption SQLs for each charted performance statistic.

To select all SQL statements from all charts

Click .

View the SQL text

Click the performance statistics line in the Statistics pane to view the SQL text for those statistics.

Related Topics

[SQL Inspector Overview](#)

[SQL Inspector Window](#)

Scan Inspected SQL

The SQL statements that you have filtered in the Statistic pane of the SQL Inspector window can be scanned in the SQL Scanner to classify the SQL statements to help identify which SQL statements are likely to be causing performance problems.

To scan the SQL statements currently showing in the Statistics pane

Right-click the Statistics pane and select **Scan Filtered SQL**.

If the SQL Scanner window is not opened then the Group Manager window opens so you can first select or create a group. Once you open the group, the scanning starts automatically.

Related Topics

[SQL Scanner Overview](#)

[Scan Inspector](#)

[SQL Inspector Overview](#)

[SQL Inspector Window](#)

Sum Statistics for a Stored Procedure

In the SQL Inspector, when you have extracted the SQL statements from the monitoring tables, the SQL statements may have been executed in a stored procedure. You can tell whether or not the SQL statement was executed in a store procedure by looking at the SQL text. If the SQL statement was executed in a stored procedure, the whole procedure will be displayed as the SQL text.

You have the option to display the statistics for each SQL statement in a stored procedure individually or to display the statistics as a sum of the SQL statement executed in the entire procedure.

To toggle between display the statistics for each SQL statement or to sum all SQL statements in a stored procedure

Click **Sum Statistics for a Stored Procedure** .

When you are displaying the statistics for the individual SQL statements, the line number column will be shown in the statistics grid. This number indicates the line in the stored procedure where the text of the SQL statement begins. This column is not displayed when you are displaying the statistics for the entire procedure.

Related Topics

[SQL Inspector Overview](#)

[SQL Inspector Window](#)

SQL Inspector Functions

Below is a list of available functions within the SQL Inspector window.

Button or Menu	Function
Schedule Menu 	Inspect/Abort Inspect
Inspector & Right-click Menu 	Add Inspector
Inspector & Right-click Menu 	Modify Inspector
Inspector & Right-click Menu	Delete Marked Inspector
Inspector Menu	Find Inspector
File Menu 	Save SQL to SQL Repository
Right-click Menu	Add/Remove Bookmark
Right-click Menu	Ad-hoc Inspect
Right-click Menu	Scan
Right-click Menu	Properties
Right-click Menu	Create Benchmark Factory Import File
Right-click Menu	Save
	Add Chart
	Refresh

	Show SQL Satisfying All Charts
	Show SQL Satisfying Any Chart
	Sum Statistics for a Stored Procedure
Right-click Menu	Scan Filtered SQL
Right-click Menu	Create Benchmark Factory Import File

Related Topics

[SQL Inspector Overview](#)

[SQL Inspector Window](#)

SQL Collector for Monitor Server Overview

The SQL Collector for Monitor Server allows you to capture from the Adaptive Server Enterprise Monitor Server any currently executing SQL statements according to your user-defined criteria. Each SQL statement captured is categorized according to suspected levels of performance problem.

The SQL Collector for Monitor Server provides an easy way of viewing and analyzing any SQL statements currently running on the database server. You can specify your own criteria to use in selecting the SQL statements you want to view. Each SQL statement monitored is analyzed to see whether it is potentially problematic so that you can quickly determine which SQL statements may be causing the performance problem.

The SQL Collector window provides a work area for you to add and delete Collectors, storing information on the Collector name and description, number of SQL analyzed, start time, end time, and duration. It offers a way to view the monitored SQL statements. It displays the monitored SQL statement, corresponding query plan and related information such as server process ID (SPID), login name, capture time, and database name. You are able to copy problematic SQL statements to the SQL Optimizer for optimization.

SQL statements collected in Inspector files can be also be analyzed with the [SQL Scanner](#) module. Once you have identified potentially problematic SQL statements you can send the SQL to the [SQL Optimizer](#), [Index Advisor](#), the [SQL Worksheet](#), or save them to the [SQL Repository](#).

The SQL Collector for Monitor Server retrieves the SQL statements using the Adaptive Server Enterprise Monitor Server.

Related Topics

[SQL Collector Window](#)

[SQL Collector Functions](#)

Adaptive Server Configuration Parameters for SQL Collector

The SQL Collector for Monitor Server uses the Adaptive Server Enterprise Monitor Server to capture SQL statements that are currently running on the database. The following items need to be set before using the SQL Collector for Monitor Server:

Adaptive Server Version

You must have Adaptive Server 15.0 or later in order to have the Adaptive Server Enterprise Monitor Server.

Adaptive Server Enterprise Monitor Server

The Monitor Server must be currently running. It must be configured on the same machine as the database server you want to monitor. With Dsedit, you can PING the remote Monitor Server to ensure that a connection is available from the PC.

Monitor Server Name on Client SQL.INI and Server must match.

The Adaptive Server and Monitor Server are defined in the `\sybase\ini\sql.ini` file on the client PC. The names in the SQL.INI file must match exactly the name of the Adaptive Server and the Monitor Server on the Database Server.

Monitor Server Name in Collector setup must match

When adding a new Collector in the Add Collector window, the name for the Monitor Server must match exactly the name of the Monitor Server in the SQL.INI file.

Tempdb size

The tempdb system segment should be at least 20 MB.

Logon Privilege

You must have `sa_role` privilege assigned to your logon.

Adaptive Server parameter

ASE Parameter	Description	Recommended setting
event buffers per engine	Specifies the number of events per database server that can be simultaneously monitored.	sp_configure "event buffers per engine", 2000
max SQL text monitored	Specifies the maximum size in Kbytes of the SQL text monitored per connection.	sp_configure "max SQL text monitored", 4096
enable monitoring	Disables the collection of performance statistics through the monitoring tables.	sp_configure "enable monitoring", 0

Note: For details of how to configure Adaptive Server Enterprise Monitor Server to capture SQL statements, please refer to the Adaptive Server Manuals.

Related Topics

[User Logon Privileges](#)

[Adaptive Server Configuration Instructions](#)

[SQL Collector for Monitor Server Overview](#)

Data Directory Setting

The SQL Collector for Monitor Server data directory is where the data files are created while executing the **Inspect** function. The data directory path is set in the Preferences window.

To review or change a directory

1. Click .
2. Select the **Directory Setup** tab.

The default setting for this directory is:

C:\Documents and Settings\User\Application Data\Quest Software\Quest SQL Optimizer\DATA

Changes to this directory cannot be made while the SQL Collector for Monitor Server is active.

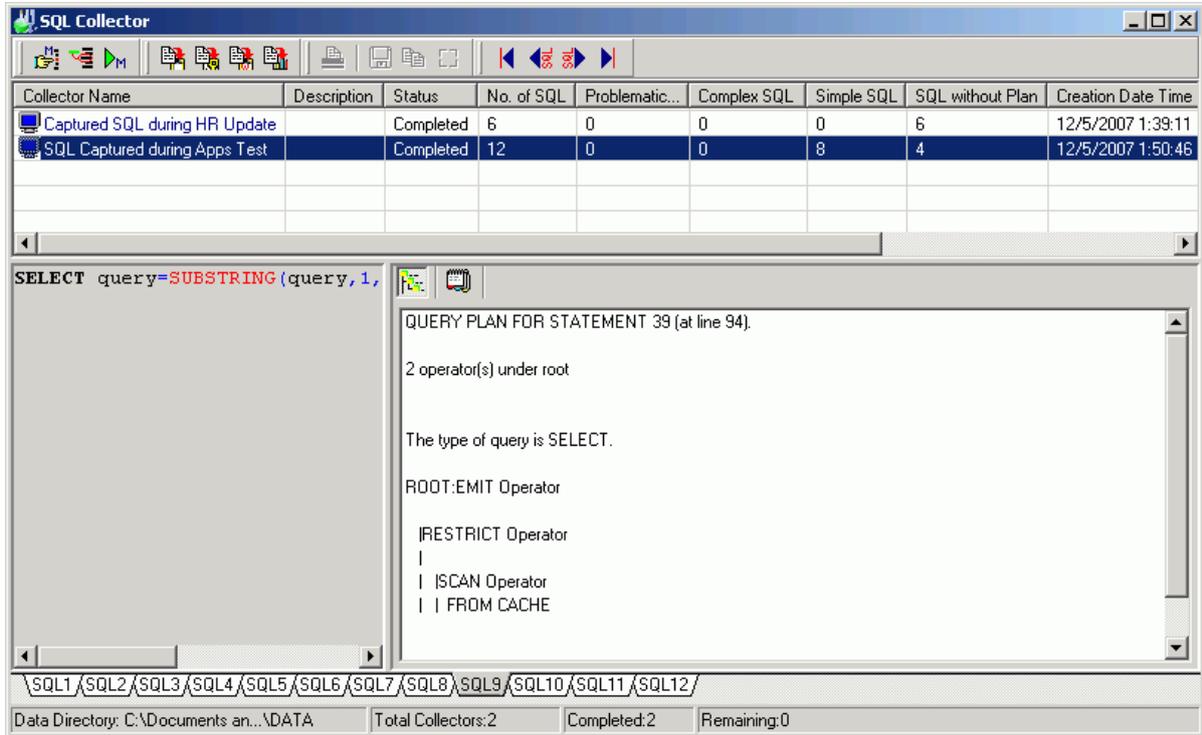
Note: It is advisable not to change the data directory after you have used this function, as files created during monitoring are kept in this directory.

Related Topics

[SQL Collector for Monitor Server Overview](#)

SQL Collector Window

View SQL Collector Window



The SQL Collector window consists of the toolbar, Job List pane, the SQL Text pane, the SQL Information pane and status bar.

Collector List

The Collector List pane in the SQL Collector window stores information on each Collector. Information includes:

Item	Description
Collector Name	The unique Collector name defined by the user.
Description	Description of the Collector defined by the user.
Status	The current status of the Collector. This column will remain blank until the monitoring process has started.
Number of SQL	Total number of SQL statements collected
Problematic SQL	Number of Problematic SQL statements collected.
Complex SQL	Number of Complex SQL statements collected.

Simple SQL	Number of Simple SQL statements collected.
SQL without Plan	Number of SQL statements collected without a query plan.
Creation Date Time	Date and time of when the Collector was created.
Start Time	Start time of when the monitoring process starts.
End Time	End time of when the monitoring process ends.
Server Name	Monitor Server name.

Note: The definitions of Problematic, Complex, and Simple SQL statements are provided in the SQL Collector Criteria tab of the Add Collector wizard.

SQL Text

The SQL Text pane displays the monitored SQL statements for a particular Collector. The layout of the SQL statement is unformatted and displayed according to how it is retrieved from the Adaptive Server Enterprise Monitor Server. SQL statements captured from the Monitor Server may consist of transact-SQL and stored procedures. Therefore to identify the SQL statements that you may want to optimize, you should use the SQL Scanner to identify the SELECT, UPDATE, INSERT and DELETE SQL statements that are contained with the stored procedures or transact-SQL .

SQL Information

If the query plan is retrieved for the monitored SQL statement the query plan and other details will be displayed in the SQL Information Pane

Status Bar

The status bar at the bottom of the SQL Collector window shows the following details:

Item	Description
Data Directory	Path showing where all the data files produced during monitoring are saved. While monitoring, a progress bar displays to show the monitoring progress.
Total Collectors	Total number of Collectors.
Completed	Total number of Collectors already monitored.
Remaining	Total number of Collectors that have not been monitored.

Note: The SQL Collector window must remain open during the monitoring process. If you have started the monitoring process, but the scheduled time has not arrived, this window must remain open for the monitoring process to start and run to completion.

Related Topics

[SQL Collector for Monitor Server Overview](#)
[Add or Modify a Collector](#)
[SQL Collector Functions](#)

Open SQL Collector for Monitor Server

To display the SQL Collector window

Click .

Related Topics

[SQL Collector for Monitor Server Overview](#)
[SQL Collector Window](#)

Add or Modify a Collector

To add a Collector to the SQL Collector window

Click  to display the Add Collectors wizard.

To modify a Collector

1. Highlight the Collector row.
2. Click .

Note: If you have already monitored the Collector, by modifying the Collector criteria you will erase any existing information.

Related Topics

[SQL Collector for Monitor Server Overview](#)
[SQL Collector Window](#)

General Information

[View Add Collector - General Information Page](#)

The General Information page consists of the following settings:

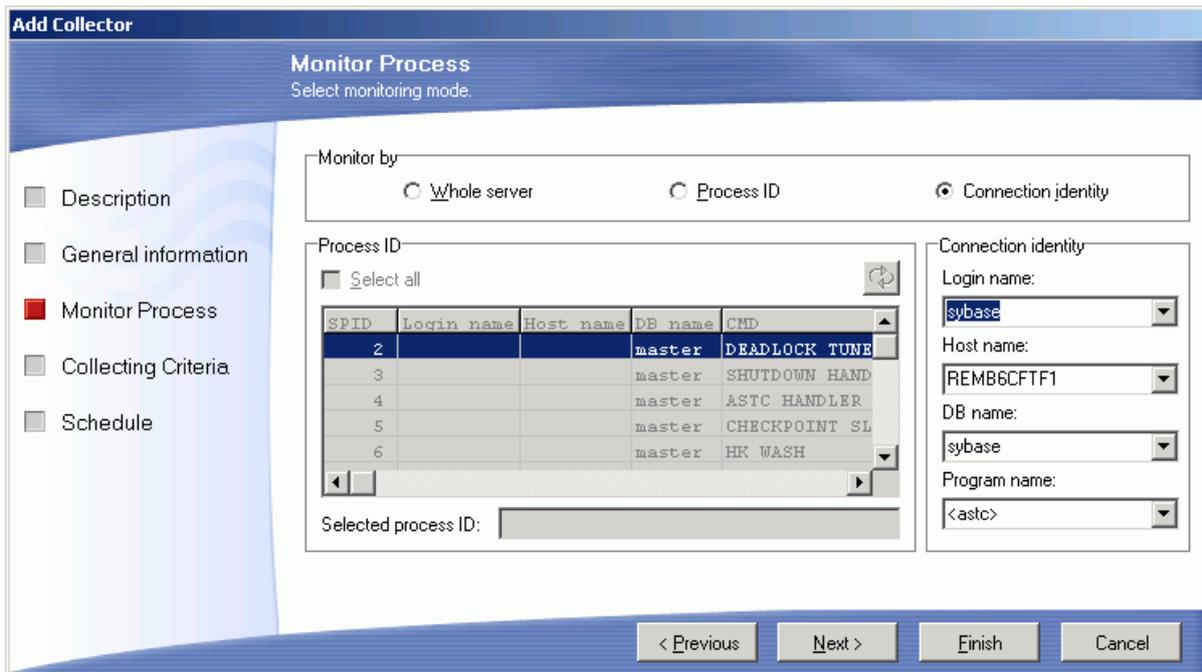
Item	Description
Collector Name	Name of Collector.
Description	Description of Collector.
Created Date Time	The date and time the Collector was created.
Monitor Server Name	The Adaptive Server Enterprise Monitor Server name (defined in the SQL.INI file).
Host name	Name of the host computer.
Port	Number of the port used for the Monitor Server.

Related Topic

[Add or Modify a Collector](#)

Monitor Process

View Add Collector - Monitor Process Page



The Monitor Process page consists of the following settings:

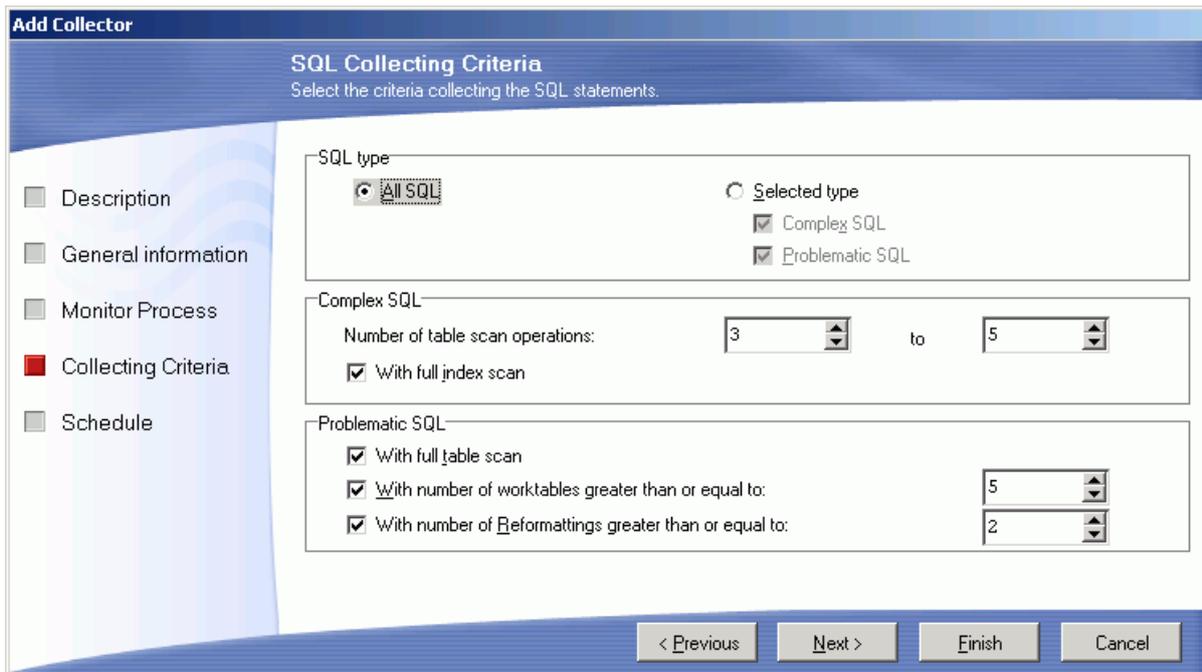
Item	Description
Whole Server	Monitor all SQL statements currently running from the database server.
Process ID	Monitor all SQL statements currently running which have the specified process ID (spid). The Process ID grid displays information about all current users and processes of the database server, displaying the process ID (spid), login name, host name, database name and the command currently executed (cmd). Click the row of the grid to select the process ID you want to monitor. To select all the process IDs, simply click the Select All checkbox.
Connection Identity	Monitor all currently running SQL statements with specified login name, host name, database name and program name.

Related Topic

[Add or Modify a Collector](#)

Collecting Criteria

View Add Collector - Collecting Criteria Page



The Collecting Criteria page has the following settings:

Section	Item	Description
SQL Type	All SQL	Monitor all SQL statements currently running.
	Selected Type	To monitor the selected SQL classification type (Problematic or Complex).
Complex Range	Number of table scan operations	Specify the range of numbers for the table references in the query plan for Complex SQL statement definition.
	With full index scan	Specify whether SQL statements with full index scan will be classified as Complex SQL statements.
Problematic Range	With full table scan	Specify whether SQL statements with full table scan will be classified as Problematic SQL statements.
	With number of worktables greater than or equal to (Range = 1 to 99)	Specify that SQL statements involving the number of worktables greater than or equal to the defined number are classified as Problematic SQL statements.
	With number of Reformattings greater than or equal to (Range = 1 to 99)	Specify that SQL statements whose query plan uses reformatting are classified as Problematic SQL. Reformatting is the process of generating a worktable with a clustered index and performing a nested-loop join. The Adaptive Server optimizer may choose this strategy when the table is large and does not have any useful index for a join.

Note: The default settings for the values are taken from the SQL Classification tab in the Preferences window.

Related Topic
[Add or Modify a Collector](#)

Schedule Settings

View Add Collector - Schedule Settings Page

Add Collector

Schedule Setting
Select settings for when to start and end the collecting and how often to collect.

Description
 General information
 Monitor Process
 Collecting Criteria
 Schedule

Polling setting

Start time (HH:MM): 14 : 35

Until: Time (HH:MM): 14 : 40
 Duration (HH:MM): 0 : 5

Interval (secs): 3

< Previous Next > Finish Cancel

The Schedule Settings page consists of the following settings:

Item	Description
Start Time	Monitor start time. Default is the current time.
Until (time)	Monitor end time. Default is 5 minutes after the start time.
Until (Duration)	Monitoring duration in hours and minutes. Default is 5 minutes.
Interval	Monitoring interval time in seconds. Default is 3 seconds.

Notes:

- Decreasing the interval time may affect the database server performance and network traffic.
- The [ad-hoc monitoring](#) from the right-click menu does not use the start, until, and duration time. Monitor will start immediately and is terminated by clicking **Abort Monitor**.

Related Topic
[Add or Modify a Collector](#)

Query Plan

SQL statements captured in the SQL Collector for Monitor Server modules are displayed with corresponding query plan using the `sp_showplan` stored procedure. If the query plan is not available, then the Query Plan pane window remains blank and the SQL is classified as SQL without Plan

Note: It is not guarantee that using the `sp_showplan` stored procedure to retrieve the query plan will obtain a query plan that corresponds to the SQL statement.

Related Topic

[SQL Collector Window](#)

Monitored SQL Statement Types

SQL statements captured in the SQL Collector for Monitor Server are classified as Problematic, Complex, Simple, or SQL without Plan. The classification settings are found on the SQL Collecting Criteria of the [Add Collector wizard](#).

Problematic SQL

Problematic SQL statements are potentially problem SQL statements that should be optimized. Problematic SQL must satisfy one of the following criteria:

- The number of tables referenced in the query plan exceeds the upper limit of the Complex SQL Table Range.
- SQL statements with a full table scan if **With Full Table Scan** option is selected.
- SQL statements with the number of worktables greater than or equal to the value defined in the **With number of worktables greater than or equal to** option.
- SQL statements whose number of reformattings is greater than or equal to the value defined in the **With number of Reformattings greater than or equal to** option.

Complex SQL

Complex SQL statements are complicated SQL statements where there is room for improvement. Complex SQL must satisfy one of the following criteria:

- The number of tables referenced in the query plan falls into the Complex SQL Table Range.
- SQL statements with full index scan if **With Full Index Scan** selected.

Simple SQL

Simple SQL statements are direct and straight forward SQL statements with minimal probability of improvement. Simple SQL are SQL statements with the number of tables referenced in the query plan less than the lower limit of the Complex SQL Table Range.

SQL without Plan

SQL statements for which the query plan cannot be retrieved using `sp_showplan` stored procedure.

For SQL statements without Plan you can use the SQL Scanner module to scan the Collector and obtain a query plan for all the captured SQL statements.

Note: The parameter settings for Problematic, Complex, and Simple SQL statements may be different between SQL Collector for Monitor Server and SQL Scanner modules. The settings for the SQL Classification for the SQL Collector for Monitor Server are set in the Add/Modify Collector window and settings for the SQL Classification for the SQL Scanner are set in the Preferences window.

Related Topics

[SQL Collector for Monitor Server Overview](#)

[SQL Collector Window](#)

[Query Plan](#)

Monitor

For the Monitor function to be enabled, you must first mark the Collector you want to monitor in the SQL Collector window. To mark a Collector, simply click the row you want to monitor.

To start monitoring

Click .

Warning: If you have already monitored the Collector, re-monitoring erases your existing information.

If the start time of the Collector has not been reached, SQL Collector for Monitor Server will wait until it is time to begin. During and after monitoring, information acquired is updated in the SQL Collector window.

The SQL Collector window must remain open during the monitoring process. If you have started the monitoring process, but the scheduled time has not arrived, this window must remain open for the monitoring process to start and run to completion.

Monitor is terminated automatically once the end time is reached.

At the end of the monitoring process, duplicated SQL statements are eliminated. This may take a few minutes to complete.

Note: The ad-hoc monitoring from the right-click menu does not use the start, until, and duration time. The ad-hoc monitoring starts immediately and is only terminated by clicking **Abort Monitor**.

Related Topics

[Ad-hoc Monitoring](#)

[Abort Monitor](#)
[SQL Collector for Monitor Server Overview](#)
[SQL Collector Window](#)

Ad-hoc Monitor

Ad-hoc monitoring allows you to analyze the SQL statements currently running without taking consideration of start, until, and duration time.

To start ad-hoc monitoring

1. Mark an existing Collector
2. Right-click and select **Ad-hoc Monitor** to start monitoring.

To stop ad-hoc monitoring

Click .

Warning: If you have already monitored the Collector, ad-hoc monitoring erases any existing information.

Related Topics

[Monitor](#)
[SQL Collector for Monitor Server Overview](#)
[SQL Collector Window](#)

Abort Monitor

To abort the monitoring process

Click .

Note: Duplicated SQL statements are eliminated. This may take a few minutes to complete.

Related Topics

[Monitor](#)
[Ad-hoc Monitoring](#)
[SQL Collector for Monitor Server Overview](#)
[SQL Collector Window](#)

Why use the SQL Scanner to Identify Problematic SQL Statements

The SQL Collector for Monitor Server offers you a way of identifying problematic SQL statements by analyzing the query plan retrieved using the `sp_showplan` procedure. However, Adaptive Server does not guarantee that the query plan retrieved from `sp_showplan` corresponds to the SQL statement. Therefore, it is recommended that you use the SQL Scanner to retrieve the query plan and identify which ones are problematic.

Note: The settings for the SQL classification for Problematic, Complex and Simple SQL statements may differ between SQL Collector for Monitor Server module and SQL Scanner module. The settings for the SQL Collector for Monitor Server module are found SQL Collecting Criteria tab in the Add Collector window. The settings for the SQL Scanner are found SQL Classification tab in the Preferences window.

Related Topics

[SQL Scanner Overview](#)

[Scan Collectors](#)

[SQL Collector for Monitor Server Overview](#)

Scan Collectors

The SQL statements captured by the SQL Collector for Monitor Server can be scanned in the SQL Scanner to classify the SQL statements to help identify which SQL statements are likely to be causing performance problems.

To scan a Collector from the SQL Collector window

1. Mark a Collector by clicking the Collector row. A green checkmark ✓ is prefixed to the Collector Name column to indicate that the Collector is marked.
2. Right-click and select **Scan**.

Related Topics

[Why use the SQL Scanner to Identify Problematic SQL Statements](#)

[SQL Collector for Monitor Server Overview](#)

[SQL Collector Window](#)

Place Bookmarks in SQL Collector

Bookmarks are used to mark a Collector in the SQL Collector window to indicate such things as:

- You are currently working on a Collector.
- You have finished reviewing a Collector.
- You have determined the critical Collectors to be reviewed first.

If the row contains a bookmark, the complete row text will be displayed in fuchsia color. You can have multiple bookmarks.

To place or remove a bookmark

Right-click and select **Add/Remove Bookmark** on the selected row.

Related Topics

[SQL Collector for Monitor Server Overview](#)

[SQL Collector Window](#)

View Particular Types of SQL Statement

After the SQL statements are captured, all the SQL statements found in the selected Collector are displayed. However, you can restrict the SQL statements shown by selecting the type of SQL statement by selecting the menu items available under the **View** menu. The SQL Collector window's title bar displays your chosen option. You can view one or more types of SQL statement by selecting or clearing the menu items.

To display the simple SQL statements

Select **View | Simple SQL**.

To display the complex SQL statements

Select **View | Complex SQL**.

To display the problematic SQL statements

Select **View | Problematic SQL**.

To display the SQL statements without a query plan

Select **View | SQL without Plan**.

To display all SQL statements

Select **View | All SQL**.

Related Topics

[SQL Collector for Monitor Server Overview](#)

[SQL Collector Window](#)

Find Specific Information in Collectors

From the SQL Collector window, you can search for SQL statements in Collectors that have specific text in the SQL text, the query plan, and/or a certain SQL classification.

To open the Find Collectors window

Select **Collector | Find Collector**.

The Find Collector window has two tabs to specify what text to search for. If you select multiple items, the search is combined as an AND search so that it searches for SQL statements that include all the selected items. After you have selected the search criteria, click **Find** to search for the SQL statements which contains all search criteria.

Find Tab

Text to find

- Under the Find tab, specify the text to search for in the SQL text.
- SQL
- Specify the text to search for in the SQL text.

Options

- Case sensitive
- Specify whether to search for the text in exact case that you enter it.
- Whole word only
- Specify whether to search for words only.

Advanced Tab

Text to find

- Under the Advanced tab, specify the text to search for in the [SQL Information Pane](#).
- Query plan
- Specify the text to search for in the query plan.
- Abstract plan
- Specify the text to search for in the abstract plan.
- Trace On Information
- Specify the text to search for in the dbcc trace on information.

- Information
- Specify the text to search for in the SQL classification Monitored SQL information.

Type

Check the [SQL Classification](#) type defined for the Collector.

Search Results

The search results are displayed at the bottom of the Find Collectors window. It displays the Collector name and the SQL name for the SQL statements which contain the search criteria.

Related Topics

[Find SQL Statement](#)

[SQL Collector for Monitor Server Overview](#)

[SQL Collector Window](#)

Find SQL Statement

The **Find SQL** function is available in the SQL Collector window. It locates the SQL statements that contain a specified word for a particular Collector taking into account the [View criteria](#).

To find a text string

1. Select **SQL | Find SQL**.
2. Enter the text you want to search for.
3. Click **Find**.

To continue searching for the same text string

Select **SQL | Find Next SQL [Ctrl + F3]**.

Related Topics

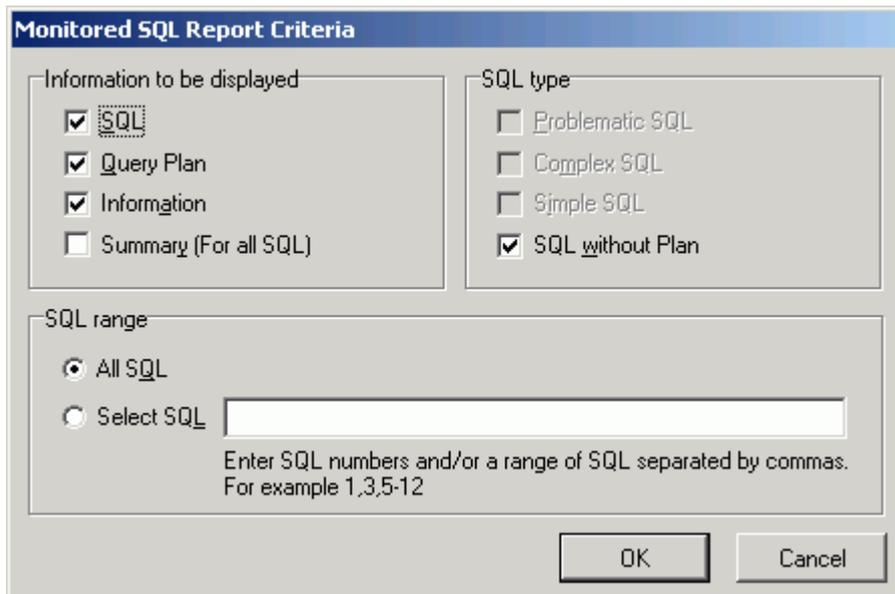
[Find Specific Information in Collectors](#)

[SQL Collector for Monitor Server Overview](#)

[SQL Collector Window](#)

Generate a Report for Monitored SQL Statements

[View Monitored SQL Report Criteria Window](#)



You can generate a report with the SQL statements in the SQL Collector window. The contents of the report depend on the components you select.

To generate a report

1. Select **Report | Monitored SQL** to open the Monitored SQL Report Criteria window.
2. Select the components for the report.
3. Select **All SQL** or **Select SQL** and enter the specific SQL statement numbers.
4. Click **OK** to generate the report.

The information in the report can be Saved and Printed.

Note: A few minutes may be needed to generate a long report.

Related Topics

[SQL Collector for Monitor Server Overview](#)

[SQL Collector Window](#)

Collector Properties

The Collector Properties window displays the name, description, creation date, file size and location, status, program version number, number of SQL monitored.

To view the properties of a Collector

Right-click and select **Properties**.

Related Topics

[SQL Collector for Monitor Server Overview](#)

SQL Collector Functions

Below is a list of available functions within the SQL Collector.

Button or Menu	Function
Collector & Right-click Menu 	Add Collector
Collector & Right-click Menu 	Modify Collector
Monitor Menu 	Monitor/Abort Monitor
Right-click Menu	Add/Remove Bookmark
Right-click Menu	Ad-hoc Monitor
Right-click Menu	Delete (Collector)
Right-click Menu	Scan
Right-click Menu	Properties
Right-click Menu	Save
	First SQL / Previous SQL / Next SQL / Last SQL
SQL Menu	Find SQL
SQL Menu	Find Next SQL
Edit Menu 	Send to SQL Optimizer
Edit Menu 	Send to Index Advisor
Edit Menu 	Copy to SQL Worksheet
File Menu	Save SQL to SQL Repository

View Menu	All SQL
View Menu	Simple SQL
View Menu	Complex SQL
View Menu	Problematic SQL
View Menu	SQL without Plan
Report Menu	Monitored SQL

Related Topics

[SQL Collector for Monitor Server Overview](#)

[SQL Collector Window](#)

SQL Scanner Overview

The SQL Scanner is a unique module which extracts SQL statements embedded in applications, database objects or files without any execution of programs. It retrieves and analyzes in batch the query plans for the extracted SQL to categorize the SQL statements according to the complexity of the query plan and suspect levels of performance problems. The SQL Scanner allows you to quickly review SQL in existing code and detect potential SQL performance problems. With this approach, the SQL Scanner allows you to be proactive in the detection of performance problems. The task of extracting, reviewing and analyzing many SQL statements is simplified and automated with the SQL Scanner.

Typically database applications contain thousands of SQL statements. Without the SQL Scanner, you have to extract and review each SQL statement manually. This is a very tedious and time-consuming task. Once these SQL statements have been extracted, you need to manually analyze each SQL statement's query plan to see if the query plan represents a potential performance problem. The SQL Scanner does this task for you.

Once you have identified potentially problematic SQL statements you can send the SQL to the [SQL Optimizer](#), [Index Advisor](#), the [SQL Worksheet](#), or save them to the [SQL Repository](#).

Related Topics

[SQL Scanner Window](#)

[Group Manager Window](#)

[Open SQL Scanner](#)

[Add Jobs to SQL Scanner](#)

[Scan](#)

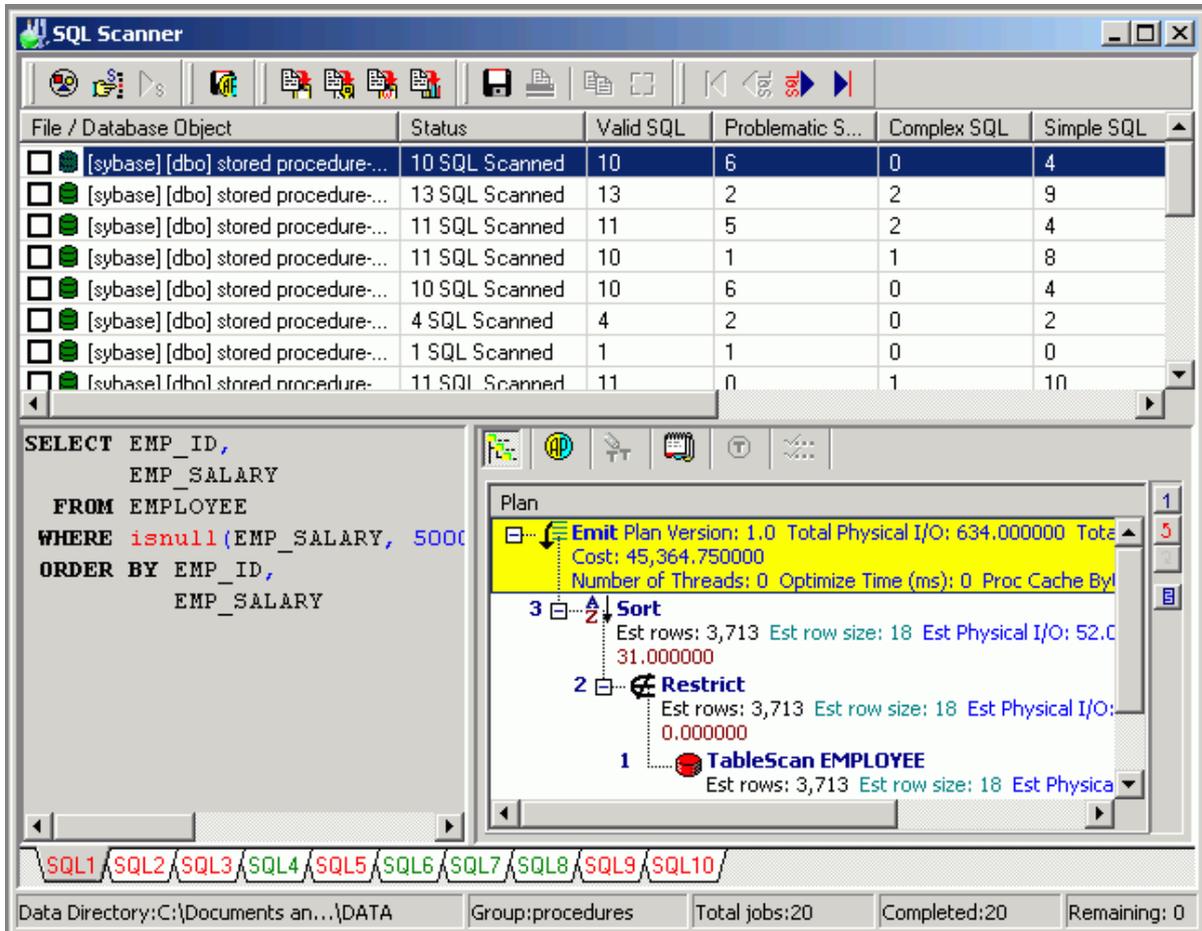
[Scan Source Code with Temp Tables](#)

[SQL Scanner Functions](#)

[SQL Conversion Overview](#)

SQL Scanner Window

View SQL Scanner Window



The SQL Scanner window has 3 panes, a status bar and a toolbar.

Job List Pane

The grid from the SQL Scanner window stores information on each individual Job in the Group. It displays the following:

Column Heading	Description
----------------	-------------

File / Database Object	<p>Checkbox</p> <p>Type of job icon and text</p>	<p>Indicates if the job is selected</p> <p> [Database] [User] Object type -> Object type</p> <p> [Database] [User] Text/Binary -> Path/File Name</p> <p> [Database] [User] COBOL -> Path/COBOL File Name</p> <p> [Database] [User] Group -> [Database] [User] APGroup -> Abstract Plan Group</p> <p> Collector -> Collector Name</p> <p> Inspector -> Inspector Name</p>
Status		Displays the current Job status. This column is blank until the Job is scanned. After the Job is scanned it displays the number of SQL statements found.
Valid SQL		Displays the number of valid SQL statements found in the Job. Valid SQL statements are syntactically correct statements recognized by the SQL Scanner, and for which Adaptive Server can provide a query plan. Valid SQL statements are further classified as Simple, Complex and Problematic SQL statements.
Problematic SQL		<p>Displays the number of Problematic SQL statements found.</p> <p>If the Show Checked SQL figures on the SQL Scanner window option is selected within Preferences you will notice that there are two figures, first one indicates the number of Problematic SQL that have not been checked and the other (enclosed in brackets) is the number of Problematic SQL that have been checked. The total of these two figures represents the total number of Problematic SQL statements.</p>
Complex SQL		<p>Displays the number of Complex SQL statements found.</p> <p>If the Show Checked SQL figures on the SQL Scanner window option is selected within Preferences you will notice that there are two figures, first one indicates the number of Complex SQL that have not been checked and the other (enclosed in brackets) is the number of Complex SQL that have been checked. The total of these two figures represents the total number of Complex SQL statements.</p>
Simple SQL		<p>Displays the number of Simple SQL statements found.</p> <p>If the Show Checked SQL figures on the SQL Scanner window option is selected within Preferences you will notice that there are two figures, first one indicates the number of Simple SQL that have not been checked and the other (enclosed in brackets) is the number of Simple SQL that have been checked. The total of these two figures represents the total number of Simple SQL statements.</p>
File Size		Displays the size of the Job in bytes.
Started At		Displays the date and time when the scanning the Job began.

Processing Time	Displays the total time taken to scan the Job. The time displays in the HH24:MI:SS format.
-----------------	--

SQL Text Pane

The left pane of the SQL Scanner window displays the scanned SQL statement for a particular scan Job. The SQL statement is formatted according to SQL Formatter's indentation algorithm. If more than one SQL statement is found in a Job, there are multiple tabs at the bottom of this pane for selecting the SQL. The tabs are color-coded to the SQL type classification if the **Use color tabs for SQL classification** option is selected in the Preferences window. Problematic SQL is red. Complex SQL is purple. Simple SQL is green. Invalid SQL is blue. Checked SQL is grey.

SQL Information Pane

The [SQL Information Pane](#) provides detailed information about the SQL statement.

Status Bar

The Status Bar at the bottom of the SQL Scanner window contents the following information:

Item	Description
Data Directory	Directory path where all the data files produced during scanning are saved. The data directory path can be changed in the Preferences window. While scanning, a progress bar displays to show scanning progress. The upper bar shows the current job progress, while the lower bar shows the total Job status.
Group	Name of the currently opened Scanner Group.
Total Jobs	Total number of Jobs.
Completed	Total number of Jobs already scanned.
Remaining	Total number of Jobs that have not been scanned.

Related Topics

[SQL Scanner Overview](#)

[Group Manager Window](#)

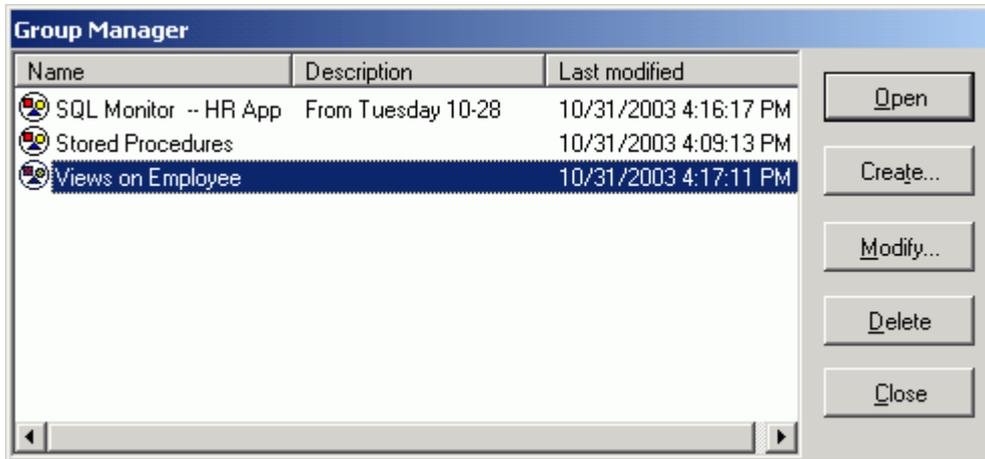
[Add Jobs to SQL Scanner](#)

[Scan](#)

[SQL Scanner Functions](#)

Group Manager Window

View Group Manager Window



A scanner Group is used to store Jobs (files, database objects, abstract plan groups, Inspectors, or Collectors from the SQL Collector for Monitor Server). The Group Manager window is used to create, open, delete, and rename these Groups. It also displays a list of existing Groups. A Group is opened in the SQL Scanner window. The Group Manager provides the following functions:

Function	Description
Open	Opens the selected Group in SQL Scanner window.
Create	Creates a new Group.
Modify	Modifies selected Group name or description.
Delete	Deletes the selected Group and all associated files.

Note: All Group information is stored as files in the SQL Scanner data directory defined in the Preferences window.

Related Topics

[View Group Summary](#)

[View Group Properties](#)

[Open Group Manager while SQL Scanner is Open](#)

View Group Summary

The Group summary report provides a statistical list and charts of all the Jobs and SQL statements scanned within the Group.

To view the report

With the **SQL Scanner** window as the active window, select **Report | Group Summary**.

Related Topics

[Group Manager Window](#)

[View Group Properties](#)

View Group Properties

To view the Group properties In the Group Manager window

Right-click a Group and select **Properties**.

Review the following for additional information:

General tab

Item	Description
Name	Group name.
Description	Group description.
Last modified	Last modified date and time.
Size	Number of bytes of disk space used to store the Job information.
Data Directory	Directory where the files are stored.
Version	Version number of SQL Optimizer when the Group was created.
Total Job	Total number of Jobs.
Completed	Total number of Jobs that were scanned.
Remaining	Total number of Jobs that had not been scanned.

SQL Summary tab

Displays a pie chart showing the number and percentage of Problematic, Complex, Simple and invalid SQL statements.

Related Topics

[Group Manager Window](#)

Embedded and Dynamic SQL Statements

Generally speaking, there are two types of SQL statements found in source code.

Embedded SQL Statements

Embedded SQL statements refer to SQL statements that are placed within the source programs and are constructed at compilation time.

Dynamic SQL Statements

Dynamic SQL statements are SQL statements constructed at run time.

Related Topic

[SQL Scanner Overview](#)

What files does SQL Scanner scan?

The SQL Scanner extracts SQL statements from the following:

Database Objects

Database objects with embedded SQL statements. This includes: stored procedures, triggers, views, rules and defaults. Depending on your database privileges, the available database objects can be selected by object name, object type, or user name.

Abstract Plan Groups

SQL statements saved in abstract plans.

SQL Collector for Monitor Server Files

SQL statements captured in the SQL Collector for Monitor Server module during execution.

SQL Inspector Files

SQL statements retrieved in the SQL Inspector module from the Adaptive Server monitoring tables.

Text/Binary Files

Text and binary files that contain embedded SQL statements.

COBOL Files

COBOL source code that contain embedded SQL statements.

Note: If your SQL statements are dynamic SQL statement generated on the fly or at run time, you can use the SQL Collector for Monitor Server to capture the SQL statements. Then the SQL Scanner can be used to analyze the captured SQL statements.

Related Topics

[SQL Scanner Overview](#)

[Add Jobs to SQL Scanner](#)

Data Directory Setting

The SQL Scanner data directory is used to store the data files created while executing the Scan function. The data directory path is stored in the **Preferences** window.

To change the data directory

1. Click .
2. Select the **Directory Setup** tab.

The default setting for this directory is:

C:\Documents and Settings\User\Application Data\Quest Software\SQL Optimizer\DATA

Changes to this directory cannot be made while the SQL Scanner is active.

Note: It is advisable not to change the data directory after selection, as files created during scanning are kept in this directory.

Related Topic

[SQL Scanner Overview](#)

Open SQL Scanner

To open the SQL Scanner

1. Click .
2. After creating a new Group or selecting an existing Group, click **Open**.

The SQL Scanner window displays after opening the working Group. If it is an empty group, the Add Jobs wizard also displays.

Related Topics

[SQL Scanner Overview](#)

[SQL Scanner Window](#)

Add Jobs to SQL Scanner

The Add Jobs wizard enables you to select the database objects, abstract plan groups, Inspectors from the SQL Inspector, Collectors from the SQL Collector for Monitor Server module, SQL Inspector files, and application source or binary files that you want to scan. The Summary page enables you to control the user and database used.

To add Jobs to the SQL Scanner window

Click  to display the Add Jobs wizard. The Add Jobs wizard displays automatically when the SQL Scanner window is opened if no Job exists in the SQL Scanner window.

Add Options

The following pages are for selecting which database objects, abstract plan groups, or files to scan.

[Database Objects](#)

[Abstract Plan Group](#)

[SQL Collectors from the SQL Collector for Monitor Server](#)

[SQL Inspectors](#)

[Source Code](#)

Summary Options

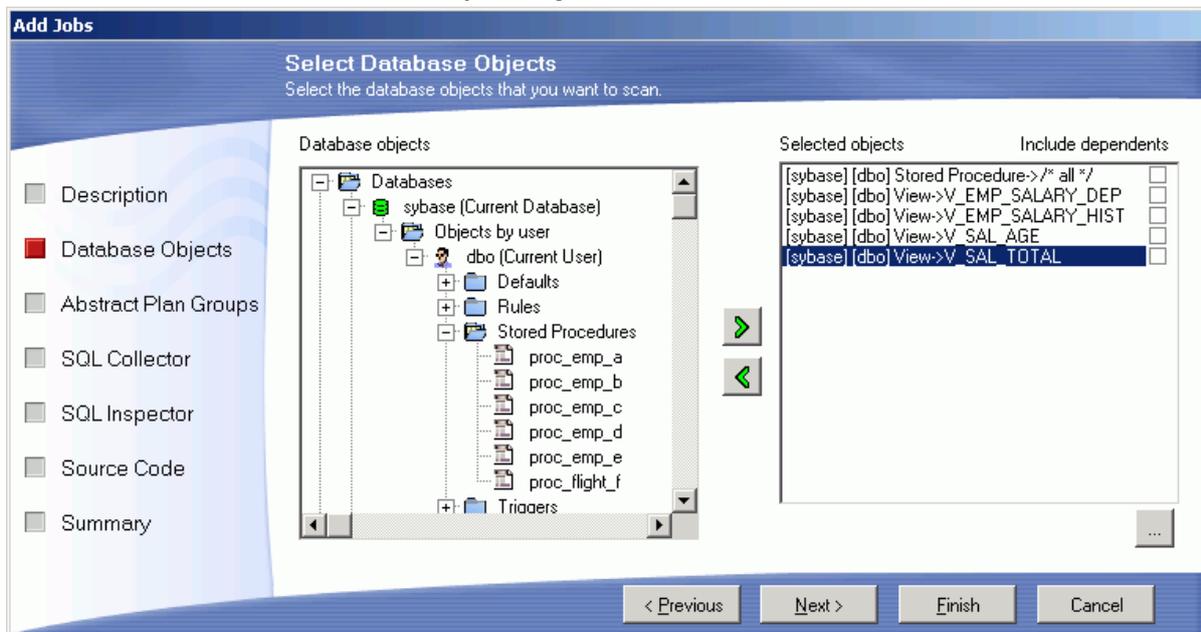
The summary page is for selecting to scan the jobs with the current database and user.

[Use Current DB & User](#)

When you have selected a large number of database objects, it may take a long to insert all of them into the SQL Scanner window. If you click **Abort**, the objects and files that have already been inserted to the SQL Scanner window remain in the window.

Add Database Objects (Add Jobs Wizard)

View Add Jobs Wizard - Add Database Objects Page



The SQL Scanner can scan all SQL statements within database objects. These include views, procedures, triggers, rules, defaults and tables. Use the Database Objects page on the Add Jobs wizard. The database objects can be selected by their object name, object type or user name.

Note: You must have privilege to access the system catalog table, syscomments, therefore the **select on syscomments.text** configuration parameter must be turned on. Also, your access to the specific database objects depends upon the database access privileges on your logon.

By clicking each branch name, the associated list of sub-items can be expanded or collapsed. You can add a

database object to a Group by highlighting the name, object type or object name, followed by clicking , double-clicking the item, or dragging the item to the list on the right [Selected objects]. To include the objects dependencies select the corresponding checkbox from the list on the right [Selected objects].

Alternatively, you can click  to open the [Add Database Objects](#) window. Select the database, user, and object type. Using the % wildcard, enter the filtering criteria. The filter is case sensitive, so you must match the upper or lowercase of the database object. Select the database objects and click **OK** to add the selected objects to the Add Jobs wizard. If a database object is already listed in the list on the right [Selected objects] of the Add Jobs wizard, it will not be entered again.

Note: By default, the scanning process uses the object's database and the user who owns the object. You can change this to the current database and user on the Summary page of the Add Jobs wizard.

Related Topic

Add Abstract Plan Groups (Add Jobs Wizard)

(For Adaptive Server 15 and later)

Use the Abstract Plan Group page of the Add Jobs wizard to scan SQL statements belonging to a particular user and abstract plan group. The list of available abstract plan group depends on whether the **Show only groups with SQL** checkbox is selected.

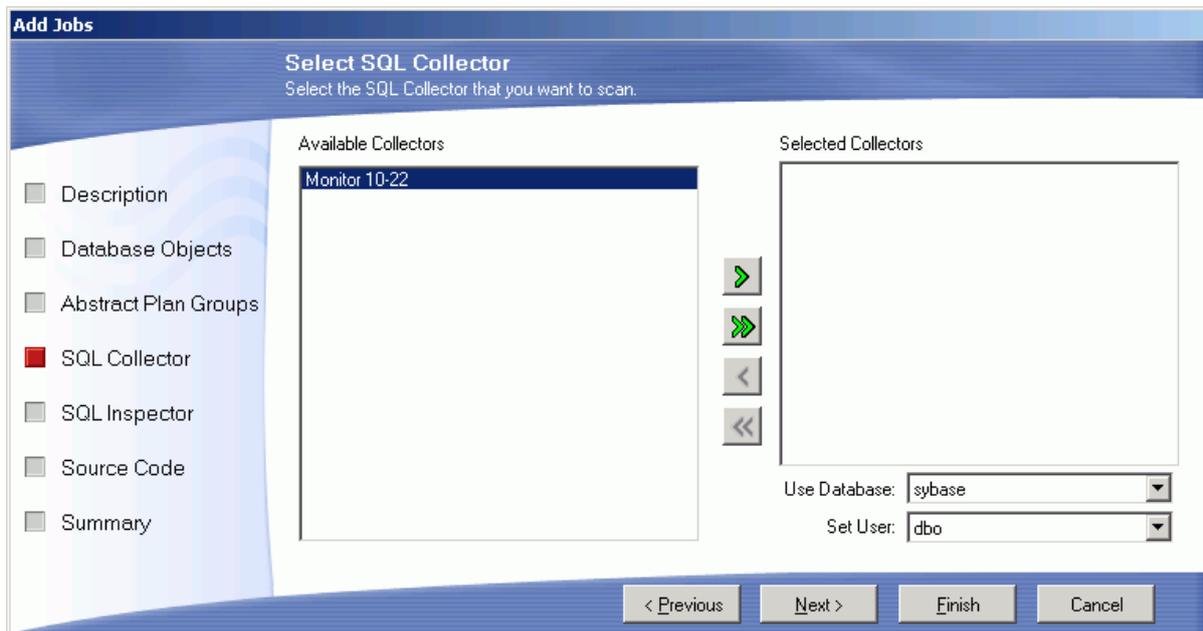
Select the abstract plan group and click  or drag the item to the right-hand list field. If you want to remove an item from the right-hand list field, click  or drag that item out of the list field back to the left-hand field. Select the **Specify database and user** checkbox and use the **Use Database** and **Set User** drop-down lists to select the database and user that corresponds with the SQL that you are scanning.

Related Topic

[Add Jobs to SQL Scanner](#)

Add SQL Collectors from the SQL Collector (Add Jobs Wizard)

View Add Jobs Wizard--Add SQL Collector



You may want to scan the Collectors from the SQL Collector for Monitor Server module to identify the SELECT, INSERT, UPDATE and DELETE SQL statements with their query plan and trace on (if available). This gives you a better understanding of which SQL statement is causing the performance problem within the database server.

To add Collectors

1. Select the SQL Collector page.
2. Select a Collector and click .
3. Use the **Use Database** and **Set User** drop down lists to select the database and user that corresponds with the SQL that you are scanning.

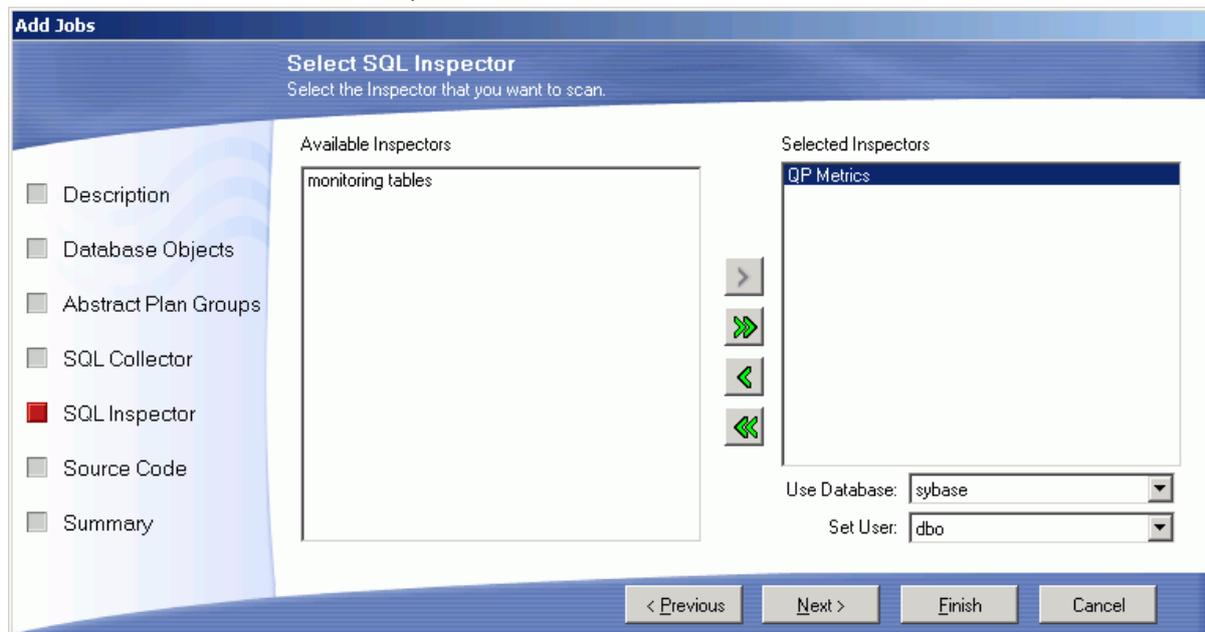
Note: The parameter settings for Problematic, Complex, and Simple SQL statements may differ between SQL Collector for Monitor Server and SQL Scanner modules.

Related Topic

[Add Jobs to SQL Scanner](#)

Add SQL Inspectors (Add Jobs Wizard)

View Add Jobs Wizard - Add SQL Inspector



(For Adaptive Server 15.0 and later)

You may want to scan the SQL Inspector to identify the SELECT, INSERT, UPDATE and DELETE SQL statements with their query plan, abstract plan and trace on information (if available). This enables you to review the query plan and helps to give a better understanding of which SQL statement is causing the performance problem within the database server.

To add a SQL Inspector

1. Click the **SQL Inspector** page. The left-hand list field displays a list of available Inspectors. If you have not yet created any Inspectors, this list field is blank.
2. Select an Inspector and click  or drag the Inspector you want to scan over to the right-hand list field.
3. Use the **Use Database** and **Set User** drop-down lists to select the database and user that corresponds with the SQL that you are scanning.

Related Topic

[Add Jobs to SQL Scanner](#)

Add Source Code (Add Jobs Wizard)

View Add Jobs Wizard - Add Source Code Page

File name	Type
C:\Documents and Settings\rwoody\My Documents\Scanner Files\Sybase\SY T...	Text/Binary

The SQL Scanner can scan through any source code stored as a text or a binary file to identify SQL statements.

To add sources code files

1. Select the Source Code page.

2. Select the file type and click .

Files Types	Description
Text/Binary	Source code saved in text or binary format.
COBOL	COBOL source files.

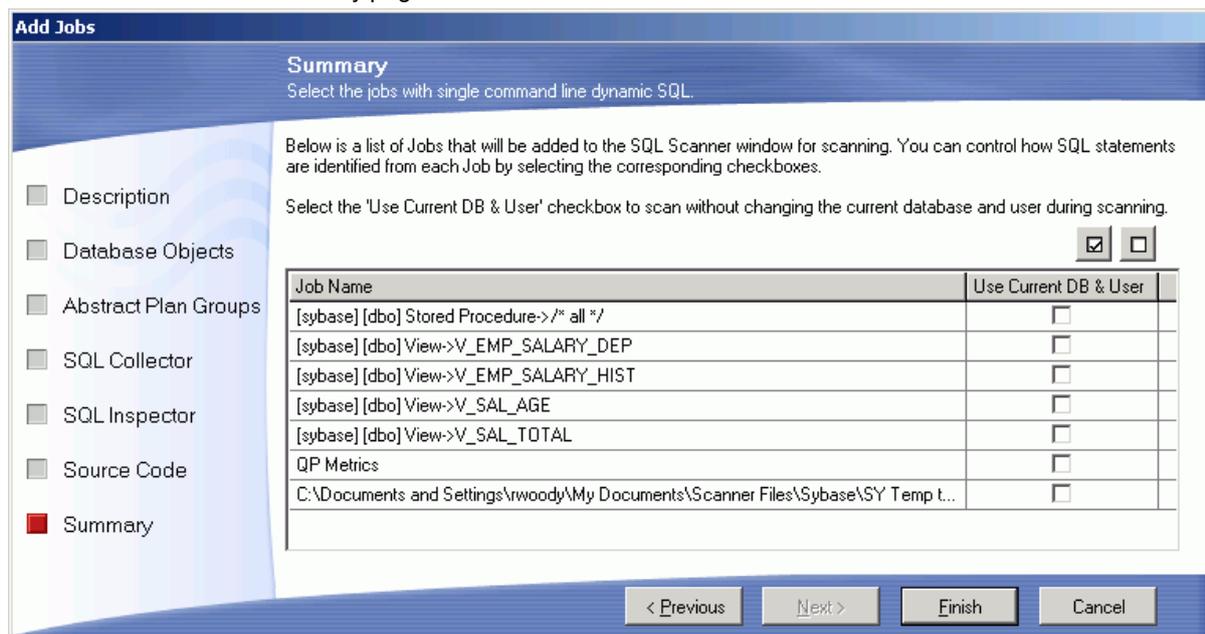
3. Use the **Use Database** and **Set User** drop-down lists to select the database and user that corresponds with the SQL that you are scanning.

Related Topic

[Add Jobs to SQL Scanner](#)

Summary (Add Jobs Wizard)

View Add Jobs Wizard Summary page



Use Current DB & User

Enables you to select specific Jobs to scan using the current database and user connection

Note: When you close the Add Jobs wizard, the newly added Jobs are automatically marked in the SQL Scanner window with a black checkmark in the far left of the row.

Related Topic

[Add Jobs to SQL Scanner](#)

Scan

To prepare a Job for scanning

Mark the Job on the SQL Scanner window by clicking the row to be scanned. The black checkmark prefixes the File / Database Object column. Multiple Jobs can be marked.

To start scanning

Click .

During scanning, information is updated on the SQL Scanner window. The time required to scan a Job depends on the processor speed of your computer, the file size, the number of valid and invalid SQL statements identified, and the time it takes to retrieve the query plan for each SQL statement.

To abort the scan

Click .

Related Topics

[Scan Source Code with Temp Tables](#)

[SQL Scanner Overview](#)

[SQL Scanner Window](#)

Scan Source Code with Temp Tables

During scanning, if the scanned SQL statement is used to create or modify a temp table these SQL statements are automatically executed if the **Create Scanner Temp Table** option in the Preferences is selected. The created Scanner Temp Tables are dropped after the Job finishes scanning.

For example:

Source Code

```
select EMP_ID,  
EMP_NAME  
into #a  
from EMPLOYEE
```

```
select *  
from #a
```

After scanning

SQL1

```
select EMP_ID,  
EMP_NAME  
into #a  
from EMPLOYEE
```

SQL2

```
select *  
from #2
```

The DDL for creating the temporary tables displays under **Scanner Temp Table** in the bottom pane. This includes the DDL found by the SQL Scanner or the DDL used to create the User-Defined Temp Table.

```
select EMP_ID,  
EMP_NAME  
into #a  
from EMPLOYEE
```

Related Topics

[Scan](#)

[SQL Scanner Overview](#)

[SQL Scanner Window](#)

Review Scanned SQL

The **SQL Text** pane and the **SQL Information** pane in **SQL Scanner** window enables the viewing and analysis of the SQL statements scanned. The following information is provided:

- Formatted scanned SQL statement
- Query plan
- Abstract plan
- Trace on
- SQL Information
 - SQL statement **type classification**: Problematic, Complex, Simple, or Invalid.
 - Database error message if SQL is classified as Invalid.

- Information about any SQL conversion the SQL Scanner applied to the SQL statement in order for it to generate a query plan.
- Line and column where the SQL statement was found in the source (for database object and source code files only)
- SQL statement used to create and populate any temporary tables the SQL Scanner created
- Checked SQL Information (if the SQL has been marked as checked)
 - Date time of when the SQL statement was checked
 - Name of who checked the SQL
 - Status
 - Checked description are displayed

Note: When the SQL Scanner finds a SQL statement that contains a variable, it will assign the BINARY data type to the variable as it retrieves the execution plan.

To assign a different data type to the variable

1. Copy the SQL statement to the SQL Editor pane in the SQL Optimizer window.
2. Click **Show Plan** .
3. Select the data type from the Parameters window which automatically displays when you have a variable in a SQL statement.
4. Click **OK**.

Related Topic

[SQL Scanner Window](#)

Find Jobs

In the SQL Scanner window, you can find a contain that contains a specific text string,

To search for a text string in all Jobs

1. Select **Job | Find Job**.
2. Enter the text to be located.
3. Click **Find**.

All Jobs and SQL number that satisfy the search criteria are listed.

Related Topic

[SQL Scanner Window](#)

Find SQL Using a Text String

The Find SQL function is available in the SQL Scanner window. This enables the location of SQL statements that contain a specified text string within a particular Job taking the View criteria into account.

To find SQL with specific text

1. Select **SQL | Find SQL** to open the Find SQL window.
2. Enter the text string to be found.
3. Click **Find**.

To continue searching for the same text

Select **SQL | Find Next SQL [Ctrl + F3]**.

Related Topic

[SQL Scanner Window](#)

View Job Properties

To view the Job properties in the SQL Scanner window

Right-click the Job and select **Properties**.

Review the following for additional information:

General tab

Item	Description
Name	Job name.
Type	Type of Job scanned (text/binary file, COBOL file, abstract plan group, Collectors from SQL Collector for Monitor Server, SQL Inspector file, database object).
Description	Job description.
Last modified	Last modified date and time.
Size	The amount of disk space required to store the job.
Data Directory	Directory where the SQL Scanner files are stored.
Status	Job status.

Version

Version number of SQL Optimizer when the Job was last scanned.

SQL Summary tab

Displays a pie chart showing the number and percentage of Problematic, Complex, Simple and invalid SQL statements.

Related Topics

[SQL Scanner Overview](#)

[SQL Scanner Window](#)

Open Group Manager while SQL Scanner is Open

To open the Group Manager

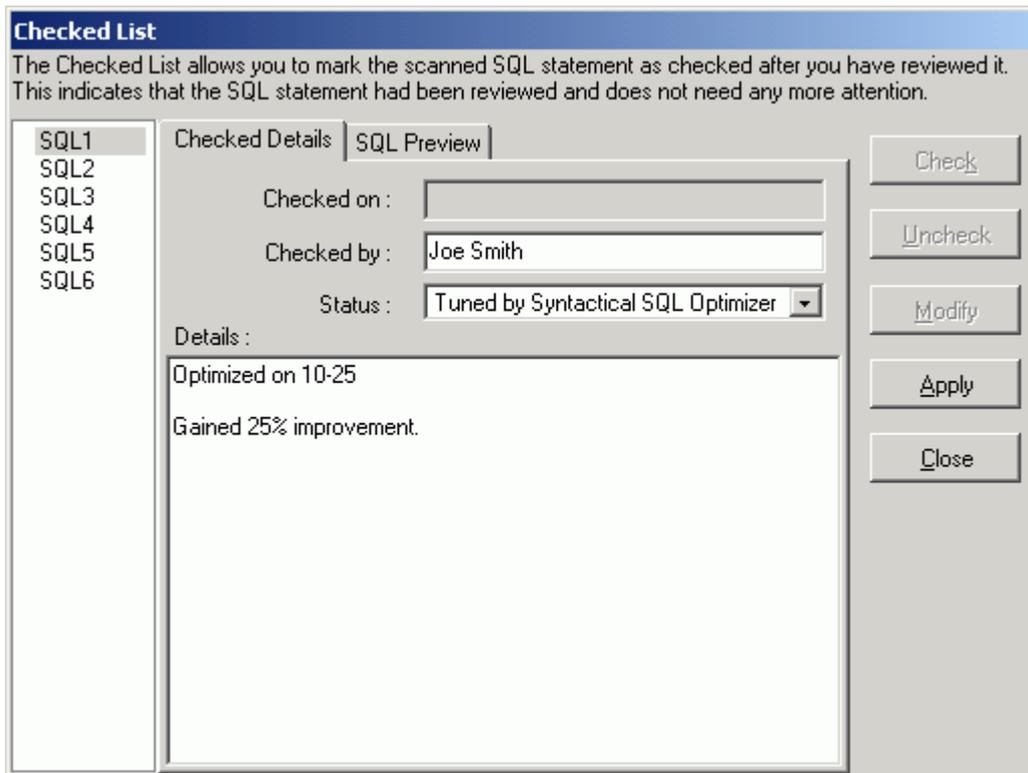
Click .

Related Topic

[Group Manager Window](#)

Check Scanned SQL

View Checked Scanner SQL Window



After scanning is completed, you can review the scanned SQL statement in the SQL Scanner window. The [SQL classification](#) types (Problematic, Complex and Simple) highlight which SQL needs attention first. It is recommended that you look at Problematic SQL, followed by Complex SQL. After you have reviewed a scanned SQL statement you can mark the SQL as checked. This indicates that the SQL statement had been reviewed and does not need any more attention.

To mark a SQL statement as checked

- From the SQL Scanner window, right-click and select **Checked List**.

Marking a Scanned SQL Statement as Checked.

All the valid SQL statements are listed on the left pane (e.g.) SQL1, SQL2 ... SQLn. Select the SQL you want to check. You can view the SQL text by selecting the **SQL Preview** tab. After you have ensured that the SQL statement is the one you want to check, select the **Checked Details** tab and click **Check**. All editable fields are changed to edit mode, which you can modify accordingly.

Checked Details

To mark a SQL statement as checked

1. Enter the following:

Item	Description
Checked on	Read-only field display the current date and time.
Checked by	Enter the user name that checked the SQL. By default, the PC user name is entered.
Status	Select the reason why you checked the SQL.
Details	Enter a short comment.

2. Click **Apply**. A blue checkmark ✓ displays next to the SQL name at the left pane to indicate that the SQL statement is checked.

Unmarking a Checked Scanned SQL Statement.

To unmark a checked Scanned SQL statement, select the SQL statement you want to unmark from the left pane of the Checked List window and click **Uncheck**. The ✓ blue checkmark is removed from the SQL name on the left pane.

Adding the Scanned SQL Statement to Checked List Automatically

You can automatically add a scanned SQL statement to the Checked List when you use the **Send to SQL Optimizer** function. In the Preferences window on the SQL Scanner tab under **General** button, select **Always Add SQL**.

Preserving Checked SQL when rescanning

You can specify to preserve the Checked SQL information when you rescan a Job by selecting the **Do not remove Checked SQL information when rescanning** option in the [General SQL Scanner Preferences](#).

Related Topic

[View a Particular SQL Type](#)

Mark and Unmark All Jobs

The black checkmark is used to indicate that the job is marked; any appended symbol is removed to indicate that the job is unmarked in the SQL Scanner window.

To mark or unmark all Jobs

Select **Group | Mark All [Ctrl + M]** or select **Group | Unmark All [Ctrl + U]**.

Related Topic

[SQL Scanner Window](#)

Move or Copy Jobs to Other Group

To move or copy Jobs in the SQL Scanner to another Group

1. Mark the Jobs to be moved or copied.
2. Select **Group | Move to Other Group** or select **Group | Copy to Other Group**.
3. Select Group where the Jobs are being moved or copied from the list.
4. Click **OK**.

Related Topic

[SQL Scanner Window](#)

Switch Database and User

To change the database used to scan a particular Job in the SQL Scanner window

1. Right-click the job and select **Use Database**.
2. Select the database name.

To change the user used to scan a particular Job in the SQL Scanner window

1. Right-click the job and select **Set User**.
2. Select the user name.

Related Topic

[SQL Scanner Window](#)

Modify a Job

To modify a Job in the SQL Scanner window

Right-click and select **Modify**.

The Modify Job dialog field allows you to do the following:

- Modify Job description
- Change the file type
- Eliminate duplicate SQL while scanning.

When modifications are made to a Job, it must be rescanned.

Related Topic

[SQL Scanner Window](#)

Place Bookmarks in SQL Scanner

Bookmarks are used to mark a Job in the **SQL Scanner** window to indicate such things as:

- You are currently working on a Job.
- You have finished reviewing a Job.
- You have determined the critical Jobs to be reviewed first.

If the row contains a bookmark, the complete row text is displayed in fuchsia color. You can have multiple bookmarks.

To place or remove a bookmark in the SQL Scanner window

Right-click on a job and select **Add/Remove Bookmark**.

Related Topic

[SQL Scanner Window](#)

Open in SQL Collector

In the SQL Scanner if the Job is a Collector file, you can open the Collector file in the SQL Collector window to view the SQL statements.

To open the SQL Collector window

Right-click the job that want to open in the Job List pane and select **Open in SQL Collector**.

Related Topics

[SQL Scanner Window](#)

[SQL Collector for Monitor Server Overview](#)

Open in SQL Inspector

In the SQL Scanner if the selected Job is an Inspector file or the selected SQL statement, you can open the Inspector file in the SQL Inspector window to view the statistics and charts.

To open the SQL Inspector and display the statistics

Right-click the job that want to open in the Job List pane and select **Open in SQL Inspector**.

Related Topics

[SQL Scanner Window](#)

[SQL Inspector Overview](#)

Create Benchmark Factory Import File

All the SQL statements can be saved in a text file from the SQL Scanner, the SQL Repository, the SQL Inspector, and the SQL Optimizer windows. These SQL statements can then be imported into Benchmark Factory 4.6 or later.

To create a file to import into Benchmark Factory

1. Right-click and select **Create Benchmark Factory Import File**.
2. Select the specific SQL statements that you want to save.
3. Enter the filename and select the file location.

View a Particular SQL Type

In the SQL Scanner window, initially all SQL statements in the selected Job are shown. Customization of the view allows the displaying of: All, Simple, Problematic, Complex, and/or Invalid SQL statements. In addition, you can view the Checked and/or Unchecked SQL statements. The title bar of the SQL Scanner window displays the chosen criteria. One or more types of SQL statements can be viewed by selecting or de-selecting the menu items.:

To display SQL statements

Select **View | option**.

Related Topics

[Check Scanned SQL](#)

[Problematic SQL](#)

[Complex SQL](#)

[Simple SQL](#)

[Invalid SQL](#)

Generate Report for Scanned SQL

View Scanner SQL Report Criteria Window

Scanned SQL Report Criteria

Information to be displayed

- SQL
- Query Plan
- Scanner Temp Table
- Trace On
- Information
- Checked Details
- Summary (For all SQL)

SQL type

- Problematic SQL
- Complex SQL
- Simple SQL
- Invalid SQL
- Checked SQL
- Unchecked SQL

SQL range

All SQL

Select SQL

Enter SQL numbers and/or a range of SQL separated by commas.
For example 1,3,5-12

OK Cancel

You can generate a report with the SQL statements in the SQL Scanner window. The contents of the report depend on the components you select.

1. Select **Report | Scanned SQL** to open the Scanned SQL Report Criteria window.
2. Select the components for the report.
3. Select **All SQL** or **Select SQL** and enter the specific SQL statement numbers.
4. Click **OK** to generate the report. The information in the report can be saved and printed.

Note: A few minutes may be needed to generate a long report.

Related Topic

[SQL Scanner Window](#)

Save Abstract Plan

To save the abstract plan from the SQL Scanner to the database

1. Click .
2. In the Save to group drop-down field, select the group where you want to store your abstract plan.
3. Click **Save**.

When the Abstract Plan is saved, it is only saved for the user that you are logged on as. In order for another user to use this abstract plan, you must export/import the plan to another user.

At the prompt "**Plan has been created successfully. The id is nnnnnnnnnn**". Click **OK**. The abstract plan is saved in Adaptive Server.

Notes:

- Adaptive Server saves the abstract plan in the sysqueryplans system table. When a query is executed, Adaptive Server looks in the sysqueryplans table for a stored SQL text that matches the query. If a match is found, the saved abstract plan is used to execute the query.
Saving the abstract plan onto the database means that when the same SQL statement is executed, the query plan is based on the abstract plan.
- When saving the abstract plan, Adaptive Server automatically trims the white-spaces from the SQL text replacing it with one space. You need to make sure the SQL statement you execute in your application is the same as the original SQL text that you saved with the abstract plan. If the SQL text does not match, then the abstract plan will not be used.

Here are some examples you need to be aware of:

Spaces in between functions

```
where substring ( EMP_NAME, 1, 5 ) = 'SMITH'
```

This is not the same as

```
where substring(EMP_NAME,1,5) = 'SMITH'
```

Spaces in between database, scheme and object name

```
from sqlexp . sqlexp . EMPLOYEE
```

This is not the same as

```
from sqlexp.sqlexp.EMPLOYEE
```

Parameter replacement

```
where EMP_ID = @var_a
```

This is not the same as

```
if @var_a = 56
```

```
where EMP_ID = 56
```

Comments

```
where EMP_ID =123 /* comment */
```

This is not the same as

```
where EMP_ID = 123
```

Related Topics

[Abstract Plan Manager Overview](#)

[Why Save the Abstract Plan?](#)

[Use Saved Abstract Plans](#)

[SQL Scanner Window](#)

Create Benchmark Factory Import File

All the SQL statements can be saved in a text file from the SQL Scanner, the SQL Repository, the SQL Inspector, and the SQL Optimizer windows. These SQL statements can then be imported into Benchmark Factory 4.6 or later.

To create a file to import into Benchmark Factory

1. Right-click and select **Create Benchmark Factory Import File**.
2. Select the specific SQL statements that you want to save.
3. Enter the filename and select the file location.

SQL Scanner Functions

Below is a list of available functions within the SQL Scanner window.

Button or Menu	Function
Group & Right-click Menu 	Add Jobs
Group & Right-click Menu	Delete Marked Jobs
Group Menu	Mark All
Group Menu	Unmark All
Group Menu	Copy to Other Group
Group Menu	Move to Other Group
Group Menu 	Group Manager
Job Menu 	Scan/Abort Scan

Job Menu	Find Job
Job Menu	Checked List
SQL & Right-click Menu	Open in SQL Inspector
SQL & Right-click Menu	Open in SQL Collector
SQL & Right-click Menu	Checked List
SQL Menu	Save Abstract Plan
	
SQL Menu	Find SQL
SQL Menu	Find Next SQL
Report Menu	Scanned SQL Report
Report Menu	Group Summary
Navigate Menu	First SQL / Previous SQL / Next SQL / Last SQL
	
Navigate Menu	Go to SQL
View Menu	All SQL
View Menu	Simple SQL
View Menu	Complex SQL
View Menu	Problematic SQL
View Menu	Invalid SQL
View Menu	Checked SQL
View Menu	Unchecked SQL
Edit Menu	Send to SQL Optimizer
	
Edit Menu	Send to Index Advisor
	
Edit Menu	Copy to SQL Worksheet
	

File Menu	Save SQL to SQL Repository
Right-click Menu	Add/Remove Bookmark
Right-click Menu	Modify
Right-click Menu	Use Database
Right-click Menu	Set User
Right-click Menu	View (Large Icons/ Small Icons/ List/ Details)
Right-click Menu	Mark Selected Jobs
Right-click Menu	Unmark Selected Jobs
Right-click Menu	Properties
Right-click Menu	Save
Right-click Menu	Create Benchmark Factory Import File

Related Topics

[SQL Scanner Overview](#)

[SQL Scanner Window](#)

SQL Conversion Overview

In order to render the SQL statement as a valid standalone SQL a number of conversions may be applied to the SQL statement.

Note: If a conversion is applied, it may be necessary to reverse the changes after optimization when an alternative SQL is pasted back to the original source code.

Related Topic

[SQL Scanner Overview](#)

Trigger Conversion

During a trigger operation, two logical tables store deleted and inserted records. The deleted and inserted logical tables cannot be referenced outside the trigger body. Therefore to be able to optimize a SQL statement used in a trigger two temporary tables are used to simulate the inserted and deleted tables.

For example:

Original SQL statement

```
INSERT INTO EMP_SMALL (EMP_ID,
```

```

EMP_NAME,
EMP_SALARY)
SELECT A.EMP_ID,
A.EMP_NAME,
B.EMP_SALARY
FROM EMPLOYEE A,
Inserted B
WHERE A.EMP_ID = B.EMP_ID

```

After conversion

```

SELECT *
INTO #inserted_simulation_table
FROM dbo.EMPLOYEE
WHERE 1 = 2
INSERT INTO EMP_SMALL (EMP_ID,
EMP_NAME,
EMP_SALARY)
SELECT A.EMP_ID,
A.EMP_NAME,
B.EMP_SALARY
FROM EMPLOYEE A,
#inserted_simulation_table B
WHERE A.EMP_ID = B.EMP_ID

```

Related Topic

[SQL Conversion Overview](#)

External Parameter Conversion

For some source code, ? is used to define external parameters, therefore to enable unique referencing, the SQL Scanner adds a number so that each parameter has a unique name within the SQL statement.

For example:

Original SQL statement

```

SELECT EMP_ID
FROM EMPLOYEE
WHERE EMP_ID = ?
AND EMP_NAME = ?

```

After conversion

```
SELECT EMP_ID
FROM EMPLOYEE
WHERE EMP_ID = ?1
AND EMP_NAME = ?2
```

Related Topic

[SQL Conversion Overview](#)

Local Variable Conversion

The local variable conversion converts SQL statements that are found in the application source code on one command line and also contain at least one "local variable" which will be replaced by the application before the SQL statement is sent to the server. The SQL Scanner encloses the variable name with `@[variablename]` and removes the concatenate character and the quotes surrounding the SQL text.

For example:

Original SQL statement before scanning

```
"" FROM EMPLOYEE WHERE EMP_ID > 100" + VEMPID + "SELECT
```

After conversion

```
SELECT FROM EMPLOYEE WHERE EMP_ID > 100@[VEMPID]
```

Note: The local variables in a scanned SQL statement should be treated as replacement or substitute variables rather than parameters. Therefore, you should hard code the values before you optimize the SQL statement. The reason for hard coding the values is that the local variables may be literals and when the application is run, these values are replaced before the SQL is sent to the database. That is why the SQL Scanner puts the variable within bracket to differentiate the local variables from the parameters.

Related Topic

[SQL Conversion Overview](#)

Cursor Query Plan Conversion

When a SQL statement is used inside a cursor declaration, the query plan that Adaptive Server generates is different than the query plan that Adaptive Server generates for the same SQL statement used without a cursor declaration.

When the SQL Scanner finds that a SQL statement that has a FOR READ ONLY or FOR UPDATE clause and is used within a cursor declaration, the retrieval of the query plan is embedded in a cursor declaration so that the query plan matches the query plan that is used when the SQL statement is executed.

No change is made to the SQL statement.

Note: When you use the Send to SQL Optimizer function, the **SQL for Cursor** checkbox is automatically selected in the SQL Optimizer window.

Related Topic

[SQL Conversion Overview](#)

Into Clause Conversion

For some front-end tools, the INTO clause of an SQL statement may be used for variable assignment which violates the syntax of the INTO clause. Therefore, an INTO clause in the SQL statement with more than one variable will be commented.

For example:

Original SQL statement

```
select EMP_ID,  
EMP_NAME  
into a, b  
from EMPLOYEE
```

After conversion

```
select EMP_ID,  
EMP_NAME /* into a, b */ /* Commented by SQL Optimizer*/  
from EMPLOYEE
```

Related Topic

[SQL Conversion Overview](#)

COBOL Conversion

The COBOL conversion searches for three items within the syntax of a SQL statement that are allowed in the COBOL, but are not valid SQL syntax: 1) a dash or minus in a variable name, 2) comments in the middle of the SQL statement, and 3) the]] (double right square bracket) as the concatenate symbol.

This conversion is only applied when the Scanner Job is added to the Job List in the SQL Scanner window using the **COBOL option** under the Source Code page in the Add Jobs wizard.

Conversion for variable name

If a variable name contains "-" minus sign, then it is replaced with "_".

Conversion for comment

If the 7th column of the line is an asterisk (*) then the complete line is recognized as a line comment.

Conversion for concatenate character

If]] (two right square brackets) are used to concatenate column names, they are replaced with a +.
For example:

Original SQL statement

```
SELECT *
FROM EMPLOYEE
* Get the department number
WHERE EMP_ID > :employee-id
AND :name-job = ENAME]]JOB
```

After conversion

```
SELECT *
FROM EMPLOYEE -- * Get the department number
WHERE EMP_ID > @employee_id
AND ENAME + JOB = @name_job
```

Note: If your COBOL file has tags at the beginning of the lines of code, you need to use the **Number of characters to be skipped at the beginning of every line for all files** option found on the SQL Scanner tab in the Preferences window.

Related Topic

[SQL Conversion Overview](#)

Index Advisor Overview

The Index Advisor provides the index candidates by analyzing the structure of a SQL statement and identifying all the related tables and indexes currently used by the SQL statement. By looking into the search arguments and table join conditions of the SQL statement, different index candidates are generated for you to evaluate the effect new indexes may have on database performance. You can further investigate index possibilities by creating your own candidates and grouping the candidates into Index Sets.

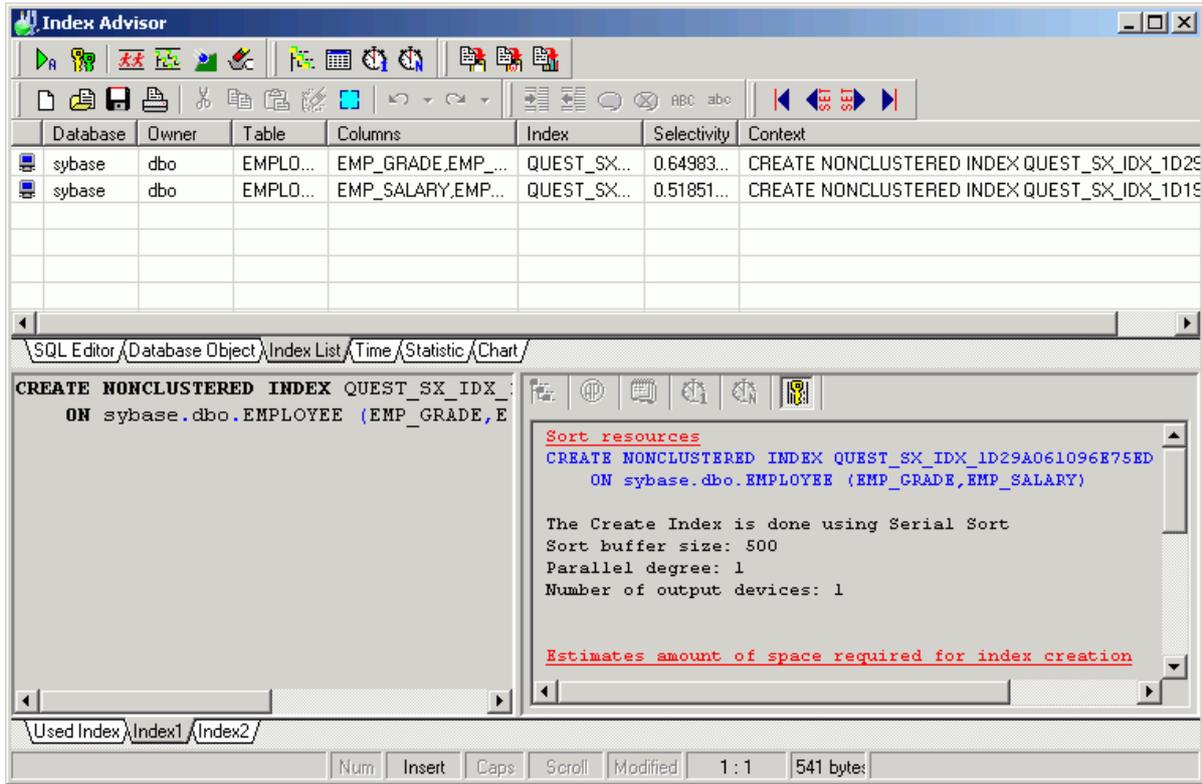
Related Topic

[Index Advisor Window](#)

[Index Advisor Functions](#)

Index Advisor Window

[View Index Advisor Window](#)



The Index Advisor window is divided into the following sections:

Top Pane

SQL Editor tab

Used to enter the SQL statement you wish to analyze for index recommendations.

Database Objects tab

Displays information about the tables used in the SQL statement.

Left Bottom Pane

Displays the table names accessed by the SQL Statement

Right Bottom Pane

Displays specific information about the selected table on different tabs: Definition, Columns, Indexes, Constraints/Keys, and Data.

Index List tab

Displays a list of the individual index candidates proposed by the Index Advisor. These are the indexes used in the Index alternatives displayed in the bottom left pane. The following information displays on this page:

Item	Description
Database	Database where index resides
Owner	Owner of the index
Table	Indexed table
Columns	Indexed column(s)
Index	Index name. All indexes generated by SQL Optimizer will be prefixed with QUEST_SX_IDX_.
Selectivity	Selectivity value. The selectivity is the percentage of rows in the table that the SQL statement selects. A SQL statement that selects a small percentage of the rows from a table has good selectivity, while a query that selects a large percentage of rows has poor selectivity. The maximum selectivity value is 1 and it corresponds to the best selectivity. Note: A Sample Block of the actual table data is used to calculate the selectivity value therefore it is possible that the value may vary according to what Sample Block is used.
Context	Index DDL for creating the index.

Time tab

Displays the [run time information](#) for the SQL statement using the alternative indexes. The values are filled in after the Show Plan, Batch Run, Run for First Record, or Run for All Records functions are executed.

Statistics tab

Displays the [run time statistics](#) for the SQL statement using the alternative indexes. These values are filled in after the Batch Run, the Run for First Record, or Run for All Record functions are executed. An **✗** is placed in the far left column if for some reason the index is not created when the SQL statement is executed.

Chart tab

[Charts the run time statistics](#) for the SQL statement using the alternative indexes.

Bottom Left Pane

Displays the Used Index which is the index(es) that are currently used by the SQL statement, the Index candidates which are generated by the Index Advisor, or the Index Sets that are created using the User-Defined Index window.

Bottom Right Pane - SQL Information Pane

Displays [information](#) about the SQL statement when it is using a specific index.

Related Topics

[Index Advisor Overview](#)

[Index Advisor Functions](#)

SQL for Cursor Checkbox

Adaptive Server uses a different query plan for a SQL statement that is embedded in a cursor declaration from the query plan when the SQL statement is not embedded in a cursor. This needs to be taken into account when retrieving the query plan or run time.

Therefore, if the SQL statement comes from or will be embedded in a cursor declaration then you need to select **SQL for Cursor** in the SQL Editor pane of Index Advisor window. This enables cursor simulation when retrieving the query plan and run time information.

This checkbox is automatically selected in the SQL Editor pane when you use the [Send to Index Advisor](#) function from the SQL Scanner if the SQL was extracted from within a cursor declaration.

In Adaptive Server 15 or later, the Declare Cursor Setting window is also available to select specific cursor settings.

View Declare Cursor Settings Window



Cursor Arguments

The cursor arguments should match the settings used for the SQL statement in your application code.

Item	Description
SEMI-SENSITIVE	Specify that the worktable which holds the result set is populated only as the rows are fetched. Therefore changes to the data that occur while the cursor is opened may be visible in the result set.
INSENSITIVE	Specify that the data is copied to a worktable when the cursor is open which makes the data insensitive to changes in the data that may occur while the cursor is opened.

SCROLL	Specify that the cursor is scrollable meaning that you can position the cursor anywhere in the cursor result set for as long as the cursor is open. All scrollable cursors are read only.
NO SCROLL	Specify that the rows are retrieved one row at a time. All update cursors are non-scrollable.

Related Topics

[Index Advisor Overview](#)

[Index Advisor Window](#)

Run Time Tab

The Run Time tab in the Index Advisor window displays the run time information for the SQL statement using the alternative indexes. The values are filled in after the Show Plan, Batch Run, Run for First Record, or Run for All Records functions are executed.

For Index candidates proposed by the Index Advisor, an icon is added in the far left column to indicate if the proposed index is used in the query plan. A key with a green checkmark  indicates that the query plan is changed by the proposed index. A key with a red exclamation mark  indicates that the key plan changed, but that the proposed index was not used by the new query plan. A key with a red x  indicates that the query plan is not changed by the addition of the proposed index.

Item	Description
Scenario Name	The index advising process numbers the SQL alternatives.
Estimated I/O cost	This is a cost estimation calculated by Adaptive Server for the query plan.
Actual I/O Cost (All Records and First Record)	This is the Actual I/O Cost of the SQL statement after it was executed in the Batch Run or the Run Time functions.
Elapsed Time (All Records and First Record)	This is the run time calculated from the clock start and finish times for the SQL statement.
Times of Improvement (All Records and First Record)	If the original SQL statement run time is available, this indicates how much faster the SQL statement is under the alternative index(es) than it is under the original index(es).
Records (All Records and First Record)	This is the number of records processed by the SQL statement.
Remarks	Information from the Batch Run is included in this column. It includes: <ul style="list-style-type: none"> 1. If the SQL was terminated by the termination criteria.

2. If the SQL was run more than once.
3. If a database error occurred.

Privileges

Estimated I/O cost, All Records Actual I/O Cost and First Record Actual I/O Cost are not available for logon users without sa_role privileges or with the "allow resource limits" configuration parameter turned off for the Adaptive Server. To retrieve the Estimated I/O cost, All Records Actual I/O Cost and First Record Actual I/O Cost estimations, you should either grant yourself sa_role, logon as another user with sa_role privileges, or turn on the "allow resource limits" parameter:

```
sp_configure "allow resource limits", 1
```

Related Topics

[Index Advisor Overview](#)

[Index Advisor Window](#)

Statistics Tab

The Statistics tab of the Run Time pane in the Index Advisor window shows the accumulative totals for the CPU Time, Scan Count, Logical Reads, Physical Reads, and Read-Ahead Reads from the execution of the SQL statement with each index alternative. To see the individual values for the statistics by table, click **All Records** or **First Record** on the top right pane or the Statistics tab at the bottom of the Index Advisor window.

For Index candidates proposed by the Index Advisor, an icon is added in the far left column to indicate if the proposed index is used in the query plan. A key with a green checkmark  indicates that the query plan is changed by the proposed index. A key with a red exclamation mark  indicates that the query plan changed, but that the proposed index was not used by the new query plan. A key with a red x  indicates that the query plan is not changed by the addition of the proposed index. In this case, the SQL statement is not executed by the Batch Run.

For more information about this statistics see the Adaptive Server Performance Tuning: Monitor and Analyzing manual, Chapter 4.

Item	Description
Elapsed Time (All Records and First Record)	The Elapsed Time is the time (in clock time) that it takes the SQL statement to select all records or the first record. This figure may include time that Adaptive Server spent on processing other tasks or waiting for disk or network I/O to complete. This is the accumulative run time for the SQL statement. The individual values for the Elapsed Time can be view by clicking All Records or First Record on the top right pane of the SQL Optimizer window.
CPU Time (All Records and First Record)	The CPU Time is accumulative total for the CPU time to execute the SQL statement. The individual values for the CPU time can be view

Actual I/O Cost (All Records and First Record)	by clicking All Records or First Record on the top right pane of the SQL Optimizer window.
Writes (All Records and First Record)	This Actual I/O Cost is a calculation from Adaptive Server after the SQL statement was executed in the Batch Run or the Run Time function.
Scan Count (All Records and First Record)	The Writes is the total number of buffers written to the disk.
Logical Reads (All Records and First Record)	The Scan Count represents the number of times a query accessed a particular table.
Physical Reads (All Records and First Record)	The Logical Reads represents the accumulative total for logical read for each table and index used in the SQL statement.
APF IOs used (All Records and First Record)	The Physical Reads represents the accumulative total for logical read for each table and index used in the SQL statement.
	The APF (Asynchronous-PreFetch) IOs used is the I/O the server does in advance anticipating which pages will need to be read next, so that the pages will be in cache when the process actually tries to access them.

Related Topics

[Index Advisor Overview](#)

[Index Advisor Window](#)

Chart Tab

Charts are available for all the statistics and run times. The Charts tab of the Run Time pane in the Index Advisor window displays two chart panes showing the run time information and statistics.

To change the information in the chart

Click the Chart Title and select one of the following functions:

Function	Description
Performance Projection	A performance forecast based on the retrieved Estimated I/O Cost value. If the Performance Projection value is greater than 1, then it is forecasted that the recommended Index candidate may give a better performance.
Estimated I/O cost	Charts the estimated I/O cost from the SQL statement with the corresponding indexes.
All Records Elapsed Time	Charts the execution time (in clock time) that it takes the SQL

and		statement to select all records or the first record.
First Record	Times of Improvement	How many times faster the alternative SQL statement is compared to the original SQL statement.
	CPU Time	Charts the execution time (in CPU time) that it takes the SQL statement to select all records or the first record.
	Actual I/O Cost	Charts the Actual I/O Cost of executing the SQL statement from Adaptive Server
	Elapsed Time + CPU Time	Charts side by side the Elapsed Time and the CPU Time.
	Writes	Charts the accumulative number of disk writes for the SQL statement from the run time statistics.
	Scan Count	Charts the accumulative number of scans for the SQL statement from the run time statistics
	Logical Reads	Charts the accumulative number of logical reads for the SQL statement from the run time statistics
	Physical Reads	Charts the accumulative number of physical reads for the SQL statement from the run time statistics.
	APF IOs used	Charts the accumulative number of APF IOs (Asynchronous-PreFetch IOs) for the SQL statement from the run time statistics.
Other	All Records	Charts the Elapsed Time + CPU Time, Actual I/O Cost, and Times of Improvement for all records.
	First Record	Charts the Elapsed Time + CPU Time, Actual I/O Cost, and Times of Improvement for the first record.
	All Records vs. First Record	Charts the Elapsed Time for all records and the first record.
	All Records vs. First Record with Cost	Charts the Estimated I/O Cost, the Elapsed Time + CPU Time (for all records), and Elapsed Time + CPU Time (for the first record).
Print		Prints the chart.
Save		Saves the selected chart.
Hide Chart		Hides the selected chart.

Related Topics

[Index Advisor Overview](#)

[Index Advisor Window](#)

Filter the Run Time Results

You can filter the Index Sets displayed on the Time tab and the Statistics tab of the Index Advisor window by any of the run time statistics after you have executed the SQL statement with the Index Sets.

The current filter displays in the bottom left corner of the Time or Statistics pane.

To filter the Index Sets

1. Click the downward pointing arrow to the right of a column heading.
2. Select one of the following:

Item	Description
All	Displays all the SQL statements.
Custom...	Opens the Custom AutoFilter window where you can select the options for filtering the SQL statements based on specific values for the selected column.
Blanks	Select all the SQL statements which do not have a value in the selected column.
NonBlanks	Select all the SQL statements which do have a value in the selected column.
Specific Value	Select all SQL statements that have one specific value.

You can repeat steps 1 and 2 to add as many filters as you would like.

To clear the filter

Click  at the bottom left corner of the Time or Statistics pane.

Related Topics

[Index Advisor Overview](#)

[Index Advisor Window](#)

Advise Index Alternatives

To have the Index Advisor propose Index candidates for a SQL statement

1. Click .
2. Enter the original SQL statement for which you want to analyze for new indexes that could be created to change SQL performance. You can also copy the SQL statement directly from another window using the **Send to Index Advisor** function.
3. Click  to generate new Index candidates for the original SQL statement.

4. In the Select tables window, select the tables for which you want the index candidates generated in the Selected column.

The Sampling Size (Rows) column displays the number of rows that will be used in the data sample that is used to determine the selectivity of the data. These values are calculated using the settings on the [Index Advisor tab](#) in the Preferences window. You can adjust these values to change the size of the sample.

The Index Advisor identifies columns as index candidates after performing an analysis of the SQL syntax, relation between tables, and selectivity of the data. It displays the DDL for the index candidates in the bottom left pane of the Index Advisor window.

To terminate the index generation process

Click .

It may take a few seconds to terminate all processes.

Once the advising process is completed, the Index Advising Details window displays detailing the indexes proposed. The Index candidates are displayed on the bottom left pane on the tabs labeled **Index1**, **Index2** ... **IndexN**. The Used Index tab displays the DDL for the index(es) currently used by the original SQL statement. The corresponding query plan, abstract plan, SQL classification, trace on and Sort Resource information are displayed in the [SQL Information Pane](#) at the bottom right.

Summarized index information for the entire index candidates is shown on the Index List tab on the top pane of the Index Advisor window.

Related Topics

[Analyze Impact of New Indexes](#)

[Create Index\(es\)](#)

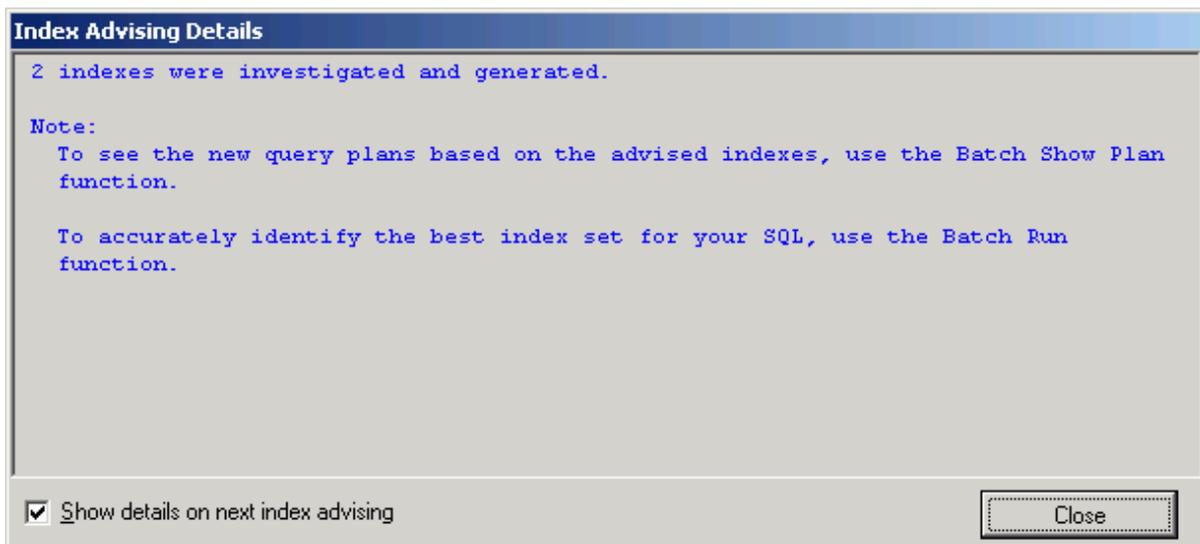
[Clear All Index Alternatives](#)

[Index Advisor Overview](#)

[Index Advisor Window](#)

Index Advising Details

[View Index Advising Details Window](#)



The Index Advising Details window is optional and can be displayed after index candidates are generated. This window displays the number of index candidates generated.

The Index Advising Details window is shown every time the index generation is finished unless it is disabled. You can disable it by clearing the **Show details on next index advising** option in the Index Advising Details window.

To open the Index Advising Details window

Select **View | Show Index Advise Details**.

If no alternative index candidates are generated,  is available for you to directly copy the original SQL statement from the Index Advisor to the SQL Optimizer for optimization. The optimization process will start automatically. It applies advanced SQL transformation technology to generate a list of semantically equivalent SQL statements that you can test in your environment to find the best performing SQL statement.

Related Topics

[Advise Index Alternatives](#)

[Index Advisor Overview](#)

[Index Advisor Window](#)

Add Index Alternatives

In addition to generating indexes from the Index Advisor window, you can also add your own Index candidates and combine them into index sets manually.

1. Click .
2. Enter the original SQL statement you want to analyze for new indexes that could be created to change SQL performance.

3. You can also copy the SQL statement directly from another window using the **Send to Index Advisor** function.

It is recommended that you use the [Advise Indexes](#) function first to generate Index candidates before creating your own.

4. Click .

This window enables you to add new indexes or to construct index sets by combining two or more indexes together.

You need to first create your own index candidates and then group them into Index Sets. If you have generated Index candidates using the **Advise Indexes** function, notice that they are displayed in the User-Defined Index window and can be used to create Index sets.

Add user-defined indexes

If one or no index candidate exists, then the Add Index window displays automatically. Otherwise, click  to the right of the list of indexes to add a new index.

The Add Index window enables you to construct your index using a graphical user interface without the need to construct the DDL for the index manually. Enter the index name, select the database, owner, table and columns on which you are defining the index and select the index type (Unique and Clustered and Non-clustered).

Create function-based indexes

Select the **Function-based index** checkbox to create the index as a function-base index. Computed columns and function-based indexes is a new feature in ASE 15.0. This feature allows developers to define a column as an expression and create indexes on expressions. Use  to select the column to add the operation to it. Use  to add the modified column to the selected columns for the function-based index.

Create Index Sets

Three options are available for whether to include the new index in an Index Set:

- Add the index and add it to a new Index Set (Create new Index Set for this index)
- Add the index to an existing User-defined Index Set (Add to existing index set), or
- Add the index without adding it to any Index Set (Do not assign to any index set).

Click **OK** to add the index to the list of indexes on the left pane list [Create Index here]. If you choose to add a new Index Set, notice that the Index Set is added to right pane list [Create Index Set here]. You can add multiple User-defined Indexes and Index Sets.

Delete User-defined Indexes

Select the index you want to delete from the top left pane list [Create Index here] and click . You can only delete indexes that are not used by any Index Sets.

Add Index Sets

You can create new Index Sets by clicking  from the top right pane [Create Index Set here]. In the Add Index Set window, enter a name for the Index Set and select the indexes you want in the Set. Click **OK**.

Delete user-defined Index Set

Select the Index Set you want to delete from the top right pane list [Create Index Set here] and click .

Add and remove index from Index Set

The bottom pane of the User-Defined Index window allows you to add and remove indexes from the Index Set by double-clicking the corresponding cell under the Index Set name.

Once you have finish creating your index(es) and Index Set(s), click **OK**. The new Index Sets are added and displayed on the tabs in the bottom left pane of the Index Advisor window.

Related Topics

[Advise Index Alternatives](#)

[Index Advisor Overview](#)

[Index Advisor Window](#)

Batch Show Plan

The index candidates are either generated with the Advise Index function or manually created by the User-Defined Index function. To evaluate the impact of the index candidates on the SQL statement, the query plan is needed. You can obtain the query plans for all the index candidates by using the Batch Show Plan.

Click .

Note: The indexes are physically created on the database, the query plan is retrieved, and then the indexes are dropped. This process may affect other SQL statements executing on the database during this period.

Related Topics

[Index Advisor Overview](#)

[Index Advisor Window](#)

Create Indexes

Once the index candidates have been reviewed, you can create an index on the database.

To create an index

1. In the bottom right pane of the Index Advisor window, select the Index Set you want to create.
2. Select **Index | Create Index(es)**.
3. The Create Indexes window displays with the CREATE INDEX statement for the selected Index Set. Click **Create**.

Related Topics

[Advise Index Alternatives](#)

[Index Advisor Overview](#)

[Index Advisor Window](#)

Analyze Impact of New Indexes

Analyzing the impact of new indexes on other SQL statements before permanently creating them on your database is essential. While improving the performance of one SQL statement, new indexes may degrade the performance of others.

First, generate or create the Index candidates for your original SQL statement in the Index Advisor window and determined which Index Set(s) give you the most performance for your SQL statement. Before performing the Index Impact Analysis function, you must save the SQL statements that may be impacted by the creation of new indexes in the SQL Repository or SQL Scanner.

To analyze the impact of creating the new indexes

1. In the Index Advisor window, select the bottom-left pane of the Index Set tab for the indexes that you want to analyze.
2. Click .

Related Topic

[Create an Index Impact Analyzer](#)

[Index Impact Analyzer Overview](#)

Clear All Index Alternatives

You can clear all the Index Sets that have been generated in the Index Advisor window.

To clear all the Index Sets

Click .

Related Topics

[Index Advisor Window](#)

Index Advisor Functions

Below is a list of available functions within the Index Advisor window.

Button or Menu	Function
Index Menu 	Advise Indexes / Abort Advise Indexes
Index Menu 	User-Defined Index
Index Menu 	Show Plan
Index Menu	Show Default Plan
Index Menu 	Batch Show Plan
Index Menu 	Run Result
Index Menu 	Run for First Record
Index Menu 	Run for All Records
Index Menu 	Batch Run / Abort Batch Run
Index Menu	Create Indexes
Index Menu	Clear All Index Sets

	
Navigate Menu	First Set, Previous Set, Next Set, Last Set
	
Edit Menu	Send to SQL Optimizer
	
Edit Menu	Copy to SQL Worksheet
	
View Menu	Show Index Advising Details
View Menu	Show Batch Run Details

Related Topics

[Index Advisor Overview](#)

[Index Advisor Window](#)

Determine Best Performing Index Alternative

After you generated Index candidates or created Index Sets for your original SQL statement in the Index Advisor window, you need to determine which indexes give you the best performance. The Batch Run function is used to retrieve the run time of the original SQL statement for different index candidates or set.

To setup and execute the Batch Run

1. Click .
2. The [Create Indexes for Batch Run](#) window displays. Select the segment where the index will be created and specify the number for the consumers. Click **OK**.
3. In the Batch Run Criteria window, select the Batch Run criteria from the following tabs and click **OK**.
 - [Selected Index Set](#)
 - [SQL Termination](#)
 - [Batch Termination](#)
 - [Run Time Mode](#)
 - [Batch Run Schedule](#)

The Batch Run dialog field displays the run time criteria and the current run time of the SQL statement for each Index Set as it is retrieved.

Note: The indexes are physically created on the database, the query plan is retrieved, and then the indexes are dropped. This process may affect other SQL statements executing on the database during this period.

To terminate the running of the SQL statement under the current Index set
Select **SQL | Stop Current**.

To terminate the Batch Run process

Click .

Related Topic
[Index Advisor Window](#)

Create Indexes for Batch Run

The Create Indexes for Batch Run window displays when you execute the [Batch Run](#) or [Batch Show Plan](#) function. It enables you to specify how you would like the indexes created:

Item	Description
Segment	Select the segment where you would like the index created from the drop-down list.
Consumer (Default 1 Range 1-20)	Specify the number of consumer processes that should perform the sort operation for creating the index. The actual number of consumer processes used to sort the index may be smaller than the specified number, if fewer worker processes are available when Adaptive Server executes the sort.

Related Topic
[Determine Best Performing Index Alternative](#)

Selected Index Set

The Select Index Set tab of the Batch Run Criteria window is used to select or deselect which Index Sets are applied before retrieving the run time for the original SQL statement. All Index Sets are selected by default. By clicking a row, an Index Set can be selected or deselected. The selected Index Sets are prefixed with a Blue Checkmark .

Note: The higher the Estimated I/O value does not necessarily mean lower performance. If possible, it is recommended that all alternatives be tested.

To select or unselect all the index sets

Right click and select Unselect All or Select All.

The Used Index can be de-selected only if the **Used Index** checkbox is not selected on the SQL Termination and Batch Termination tab and the **Always run Used Index first** option in the Selected Index tab is not selected.

To change the order the Index Sets are tested

1. Click the row.
2. Click **Move Up**  or **Move Down** .

To change the order of the Index Sets by sorting

The Index Sets are ranked by Est. I/O Cost by default, with the exception for the Used Index which is placed at the top. You can sort either the Est. I/O column or the Index Set column by clicking the column heading.

To always run the Used Index first

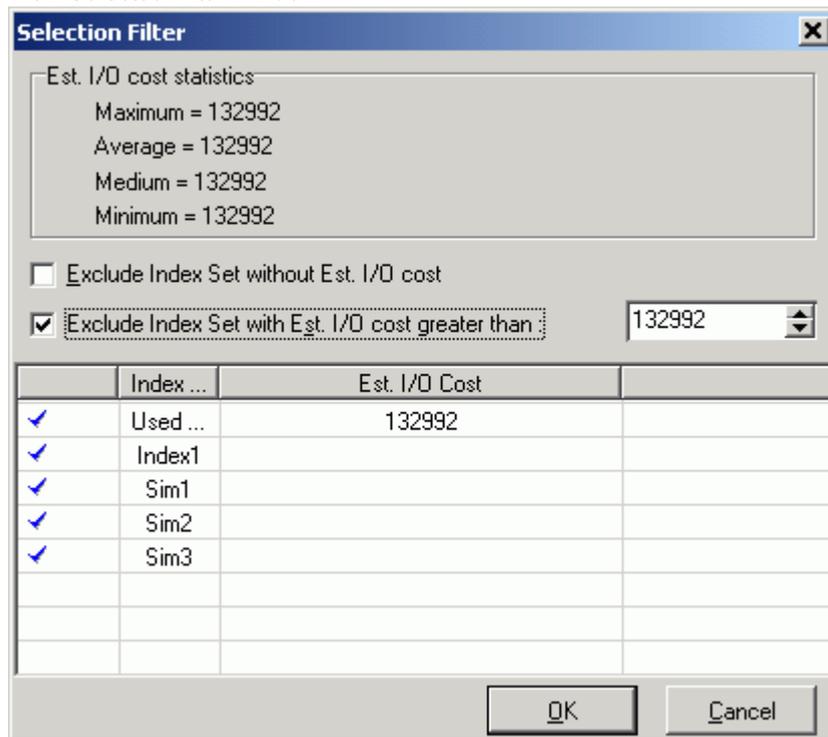
Select the **Always run Used Index first** checkbox.

Note: This checkbox is dimmed if Used Index is selected as the SQL Termination or Batch Termination criteria.

To select or unselect a group of Index Sets according to the Est. I/O Cost values

1. Select the **Apply Filtering** checkbox.
2. Click **SQL Selection Filter** .

View Selected Filter Window



Note: The Est. I/O Cost is only available if you have already obtained the query plans.

Related Topic

[Determine Best Performing Index Alternative](#)

SQL Termination (Options)

The Index Advisor provides several index candidates for the SQL statement. The SQL statement will run faster with some of these index alternatives and it may run longer with other index alternatives. Therefore, you will want to terminate SQL statement when it is running longer. The SQL Termination tab is used to set the termination criteria for the SQL statement with each index set. If the current run time for the SQL statement exceeds the termination time, then it is terminated automatically.

Note: The termination time has a [percentage delay](#) added to it.

To define your termination criteria select one of the following options

Used Index

The **Used Index** option enables you to retrieve the run time of SQL statements that run faster than the original SQL statement. It terminates the SQL statement when it runs longer using an Index alternative than it ran with the current or "used" indexes. If you choose this option, the Used Index is automatically selected on the **Selected Index Set** tab. During the Batch Run, you cannot terminate the SQL statement when it is running with the Used Indexes because this run time is needed to determine when to terminate the Index alternatives.

Best running time SQL

The **Best running time SQL** option allows you to retrieve the run time of the SQL statement that is faster than the current best run time. With this option, the run time from the first execution of the SQL statement is used as the termination time. When a faster run is found, the faster time is used as the new termination time. So you are always using the current fastest run time as the termination time for the next execution.

Note: The first index is either the Used Index or the index with the lowest Est. I/O Cost. This depends on whether the Always run Used Index first option is checked on the Selected Index Set tab.

User-defined time

The **User-defined time** option retrieves the run time of the SQL statement when it is less than the defined time.

Combining Criteria

You can also combine **User defined time** with **Used Index** or **Best running time SQL** by clicking the **Or user-defined time** checkbox next to each one.

Run without Termination

The **Run without termination** option allows you to retrieve the run time of the SQL statements without any termination criteria. The SQL statement is run to completion with each Index Set.

Related Topic

[Determine Best Performing Index Alternative](#)

SQL Termination (Percentage Delay)

View Batch Run Criteria--SQL Termination Window

The screenshot shows the 'Batch Run Criteria' dialog box with the 'SQL Termination' tab selected. The dialog is divided into three main sections: 'Batch Termination', 'Run Time Mode', and 'Batch Run Schedule'. The 'SQL Termination' section is active and contains the following options and settings:

- Termination Criteria:** Three radio buttons are present: 'Used Index', 'Best run time SQL' (which is selected), and 'User-defined time (mins/secs)'. To the right of these are two checkboxes, both of which are unchecked: 'Or user-defined time'.
- Time Settings:** Below the radio buttons are three spinners: 'User-defined time (mins/secs)' with values '1' and '0', 'Percentage delay (%)' with a value of '5', 'Minimum delay time (secs)' with a value of '5', and 'Maximum delay time (mins)' with a value of '60'.
- No termination:** A section at the bottom contains a radio button for 'Run without termination', which is currently unselected.
- Buttons:** At the bottom right are 'OK' and 'Cancel' buttons. At the bottom left is a checkbox for 'Save settings for next index advising', which is unchecked.

The SQL Termination tab is used to determine the percentage delay calculation and then adds the additional time to the termination time. It is used to account for the time it takes the SQL statement to travel from the PC to the database server over the network. It also enables you to find other SQL statement that are executing close to the termination time.

Percentage delay (%)

The value entered into this field is used as a percentage to calculate the additional time that is added to the termination time. For example, if the termination time is 10 minutes and the percentage delay is 5%, then all SQL statements executed are terminated if the run time exceeds 10.5 minutes. $(10 + (10 * 5\%))$

Minimum delay time (secs) - Range 1 - 99 (Default: 5)

This is the minimum number of seconds that is added to the termination time. It is necessary to factor into the overall termination time the time needed for the SQL statement to be sent to the database server from the PC before it starts to run. This number is only used if the Percentage delay calculation is lower than this value.

Maximum delay time (mins) - Range 1 - 9999 (Default 60)

This is the maximum number of minutes that can be added to the termination time. This number is only used if the Percentage delay calculation is higher than this value.

Related Topic

[Determine Best Performing Index Alternative](#)

Batch Termination

View Batch Run Criteria--Batch Termination Window

The screenshot shows the 'Batch Run Criteria' dialog box with the 'Batch Termination' tab selected. The dialog has three main sections: 'Selected Index Set', 'SQL Termination', and 'Batch Run Schedule'. The 'Batch Run termination criteria' section contains the following options:

- No termination
- Terminate Batch Run if the specified number of index sets fall in the criteria
 - Number of index sets (excluding Used Index): 3
 - Count the index set if its elapsed time is faster than:
 - Used Index
 - Used Index with a percentage of improvement: 10
 - User-defined time (mins/secs): 1 min, 0 secs

At the bottom, there is a checkbox for 'Save settings for next index advising' (unchecked), and 'OK' and 'Cancel' buttons.

The Batch Termination page of the Batch Run Criteria window is used to determine if and when to terminate the Batch Run. It enables you to find alternative SQL statements that give you good performance improvement without having to execute every SQL alternative.

No termination

Specify to run the Batch Run to completion.

Terminate Batch Run if the specified number of index sets fall in the criteria.

Specify to terminate the Batch Run when a specified number of index set are found that meet the following requirements for terminating the Batch Run.

Number of index sets (excluding the Used Index)

Specify how many index sets must be found which show performance improvement over any index that is currently being used.

Count the index set if its elapsed time is faster than

Specify one of the following criteria to select how the performance improvement is determined.

Used Index

Count each index set where the run time for the SQL statement is faster than the time using the indexes currently on the database.

Used Index with a percentage of improvement

Count all index sets where the run time for the SQL statement is the specified percentage faster than the time using the indexes currently on the database.

User-defined time (mins/secs)

Count all index sets where the run time for the SQL statement is faster than a specified number of minutes and/or seconds.

Related Topic

[Determine Best Performing Index Alternative](#)

Run Time Mode

The Batch Run is designed to give the most accurate result by providing options for obtaining the most accurate run time taking into account the effects of caching the data, indexes and the SQL statements. This section allows you to select one of the four options best suited to your SQL statement.

Run to retrieve	Description
All records	Specify to retrieve the run time for processing All Records.
First n Records(s)	Specify to retrieve the run time for processing n records where you specify the number of records retrieved.
Retrieve the run time by executing	Description
Run SQL options	Select one of the following options: <ul style="list-style-type: none">Run all SQL twice if original SQL runs faster than (seconds)—Combines the Original SQL twice and all others once and the All SQL twice options into one option and allows you to determine (by the number of seconds a SQL statement runs) which option to use. The original SQL statement always runs twice. The SQL alternatives run twice if the original SQL statement runs in less time than the value specified. Otherwise, the SQL alternatives all run once.

- First one twice using the second run time and others once—The first time you access data from table, the data is cached into memory. This process takes a few moments. The next time you access that data, it is already in memory so the following SQL statements run faster. So to have a comparable test, the first SQL statement is run twice and the time from the second run is compared to the time from the other statements.
- All once—For long running SQL statements, there is no need to run them twice since the effect from caching diminishes over time.
- All twice—Running all SQL statements twice enables you to eliminate some factors that can affect the accuracy of the results. If some SQL statements have been recently executed, then the SQL information is likely to be resident in cache and it may execute faster because of that. Also, if the SQL statements use different indexes, some indexes may be resident in cache and the others are not.

Five methods of calculating the run time are available if you select this option: Excluding the first run time, Average, Sum, Maximum, and Minimum.

- Excluding the first run time—Specify to throw out the first run time and use the second one when all SQL statements should have the necessary items in cache. This minimizes the effect that caching of the SQL statement and the indexes may have on the accuracy of the run time results.
- All multiple times—This option is suitable for SQL statements with very short run times. It executes the SQL statements the selected number of times. It calculates the run time based upon the selected calculation method. This option helps to negate the factor of other activity running on the CPU at the same time as the SQL is executed. Since the run time is calculated from the moment the SQL statement starts running to the time it finishes, if SQL statement shares the CPU with other activity, the run time includes this other activity. Four methods of calculating the run time are available if you select this option: Average, Sum, Maximum, and Minimum.

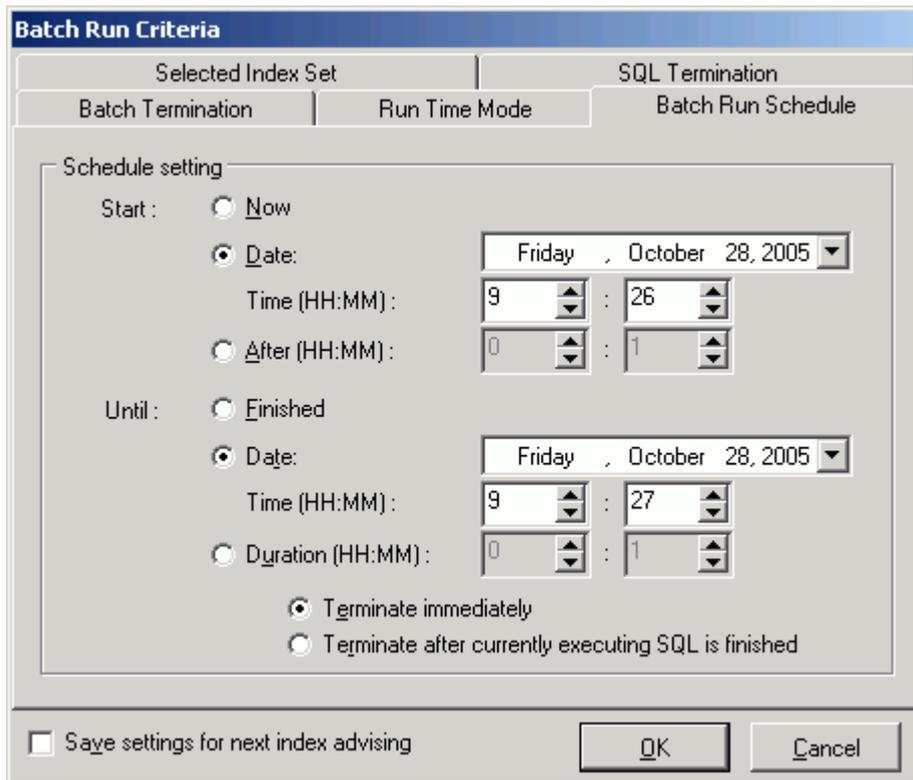
This option is only available for SELECT statements since the INSERT, UPDATE, and DELETE statements would need a rollback between each of the multiple executions.

Related Topic

[Determine Best Performing Index Alternative](#)

Batch Run Schedule

View Batch Run Criteria--Batch Run Schedule Window



The Batch Run Schedule criteria enable you to schedule when to start and stop the Batch Run.

Schedule setting

Start

Now

Specify to start the Batch Run immediately.

Date and Time

Specify to start the Batch Run at a specific date and time.

After

Specify to start the Batch Run after it has been running for a specified number of hours and minutes.

Until

Finished

Specify to run the entire Batch Run until all SQL statements are executed.

Date and Time

Specify to terminate the Batch Run on the specified date and time.

Duration

Specify to terminate the Batch Run after it has executed a specified number of hours and minutes.

Terminate immediately (Date/Time and Duration only)

Specify to terminate the currently executing SQL statement and all remaining unexecuted SQL statements.

Terminate after currently executing SQL is finished (Date/Time and Duration only)
Specify to finish running the currently executing SQL statement and do not create the remaining indexes and benchmark the SQL statement against them.

Related Topic

[Determine Best Performing Index Alternative](#)

View Batch Run Details

The Batch Run Details window displays a summary of the run time information of the SQL statements with the alternative indexes. The Batch Run Details window displays after the batch run process is completed unless the **Show details on next batch run** checkbox in the Batch Run Details window is unchecked.

You can review the Batch Run Details window after a Batch Run by selecting **View | Show Batch Run Details** when the Index Advisor window is active.

If a SQL statement has a database error during the Batch Run, an explanation of the error is included in the Batch Run details.

Related Topic

[Determine Best Performing Index Alternative](#)

SQL Optimizer Overview

The SQL Optimizer module automates the optimization of SQL statements. It employs a unique engine that uses Artificial Intelligence to generate all the possible SQL alternatives that can be mathematically proven to be "semantically equivalent" to the original SQL statement which guarantees that the SQL alternatives will produce the exact same results as the original SQL statement. After the alternatives are generated, you can compare each SQL statement to any other SQL statements to see the different SQL coding techniques for achieving the same results. You can then test these alternative SQL statements in your environment to find the best one for your database environment.

SQL Syntax Transformation

The first step of this engine transforms the original SQL statement and produces a group of alternative SQL statements where the syntax was rewritten. Then, the SQL Optimizer rewrites each newly created SQL statement to produce another group of alternatives. The engine continues rewriting each alternative until all the SQL statements cannot be rewritten any further or until the user-defined [quota](#) for the number of SQL statements generated by syntax transformation is reached.

In the [Optimization settings](#), you can also control whether the optimization process will generate SQL statements make use of temporary tables. You can also select to use the JOIN clause (INNER JOIN, CROSS JOIN) from the Ansi-92 SQL standard or to use the original SQL syntax for joining tables.

Applying Adaptive Server Optimization Techniques

After the SQL Optimizer has exhausted rewriting the syntax of the SQL statement, the [various techniques available in Adaptive Server for optimizing a SQL statement](#) are applied to the original SQL statement and each of the SQL alternatives until all selected options have been applied to all the SQL alternatives or until the user-defined quota is reached.

Eliminating Duplicate Query Plan

For each rewritten SQL statement, the query plan is compared to all the other query plans. One SQL alternative is selected for each unique query plan. Although the optimization process may generate hundreds of SQL alternatives, you will see only some of the alternatives since the alternatives with a duplicate query plan are eliminated.

Testing for Best Alternative

Although all the SQL alternative statements produce the same result, Adaptive Server will likely use a different path to retrieve the data for each one. It is difficult to decide which SQL statement will run faster without taking into account the database structure, indexes, and data volume, so it is important to test the SQL alternatives in your database environment using the [Batch Run](#) to determine the best SQL alternative from the run time statistics.

Intelligence Level Settings

The settings in the [Optimizer options](#) affect the amount of time it takes and the number of alternatives that are generated by the optimization process. You can quickly select to increase or decrease the intensity of the optimization process using the [Intelligence Level settings](#) to automatically select more or less options.

Create Your Own Alternative SQL

You also have the ability to test and compare your own SQL alternatives using the [Insert User-Defined SQL](#) function.

Related Topic

[Optimization Engine](#)

[SQL Optimizer Window](#)

[SQL Optimizer Functions](#)

[Optimize Original SQL Statement](#)

Optimization Engine

During optimization, the SQL Optimizer employs a unique optimization engine that uses multiple SQL syntax transformation rules to produce a list of semantically equivalent SQL statements. This engine first transforms the original SQL statement, hence producing a group of optimized SQL statements. It then goes on rewriting each optimized SQL statement to produce another group of alternatives. The engine continues rewriting until the SQL statements cannot be rewritten any further or a set of user-defined quotas are reached.

One of the searching rules is illustrated in the following example:

```
SELECT *
FROM table_a
WHERE table_a.key in (SELECT table_b.key
FROM table_b)
```

If `table_b.key` is an indexed column, the following transformation is executed:

```
SELECT *
FROM table_a
WHERE EXISTS (SELECT 'x'
FROM table_b
WHERE table_b.key = table_a.key)
```

Although the above two SQL statements produce the same result, the database may evolve two different query plans. It is difficult to decide which SQL statement will run faster without taking into account the database structure, indexes, and data volume, so testing the SQL alternatives in your database environment is important to the process of selecting the best SQL alternative.

Related Topic

[SQL Optimizer Overview](#)

Common Coding Errors in SQL Statements

The performance of database applications can be downgraded simply because common coding errors existing in SQL statements introduced by database changes such as adding a new index. Therefore it is important to review every SQL statement existing on the database server and source code using the SQL Scanner module. Performance of SQL statements can normally be dramatically improved by correcting common coding errors based on database information obtained during login.

The SQL Optimizer not only optimizes the input SQL statements but also corrects common coding errors based on database information obtained during login.

The followings are some examples of common mistakes found in SQL statements:

Enable index search (Built-in De'Morgan Law Engine is used to solve this problem)

```
SELECT *
FROM table_a
WHERE NOT(table_a.key + 5) > 15
```

```
SELECT *
FROM table_a
WHERE table_a.key <= 15 - 5
```

Remove unnecessary function calls (If A.key is a not null column detected from database)

```
SELECT *
FROM table_a
WHERE ISNULL(table_a.key,0) = 10
```

```
SELECT *
FROM table_a
WHERE table_a.key = 10
```

Related Topic

[Optimization Engine](#)

What Function Should I Use to Retrieve the Run Time?

The SQL Optimizer provides two different measurements of performance; All Records and First Record. Both measurements give you an indication on the fastest running SQL statement but with two different aims. The All Records is the time it takes to retrieve all the records from the query. The First Record is the time it takes to retrieve the first record from the query. You must understand the intention of the SQL statement and what type of source code it is be embedded in. Generally, if the SQL statement is used for reports, then you should use the **Run for All Records** or **Batch Run** with **Run Time Mode** set to **All Records**. If the SQL statement is used for on-line query, then use the **Run for First Record** or **Batch Run** with **Run Time Mode** set to **First n Record(s)**.

Note: If the aim of the SQL statement is unknown, then use All Records as a performance indication.

Related Topics

[Retrieve Run Time for a Group of SQL](#)

[Analyze Time and Statistics Results](#)

[Analyze Run Time Results](#)

[Analyze Statistics Results](#)

Unsatisfactory Performance Results

After optimization, you may discover that the performance of the optimized SQL statements is still not satisfactory. To remedy this, first check that the searching quota has not been reached in the [Optimization Details](#) window. If it has, then you should increase the intelligence level or optimization options in the Preferences window and optimize again to ensure all transformed SQL statements are given. Review the query plan of the optimized SQL statement to check if there should be any alterations to the database structure, for example adding a new index.

Rerun the SQL statement optimization after the review.

SQL Optimizer Functions

Below is a list of available functions within the SQL Optimizer window.

SQL Editor Functions

These functions are available only in the SQL Editor.

Button or Menu	Function
SQL Menu 	Optimize/Abort Optimize
SQL Menu 	Optimize Using Abstract Plan/Abort Optimize Using Abstract Plan
SQL Menu 	Show Plan
SQL Menu	Show Default Plan
File Menu	Open SQL from SQL Repository

SQL Editor and Optimized SQL Functions

These functions are available for both the SQL Editor and Optimized SQL.

Button or Menu	Function
SQL Menu 	Run Result/Abort Run Result
SQL Menu 	Run for First Record/Abort Run for First Record
SQL Menu 	Run for All Records/Abort Run for All Records
SQL & Right-click Menu	Testing for Scalability

	SQL Menu	Open Optimized SQL
	Edit Menu	Send to Index Advisor
	Edit Menu	Copy to SQL Workshop
	File Menu	Save to SQL Repository
	Right-click Menu	Create Benchmark Factory Import File

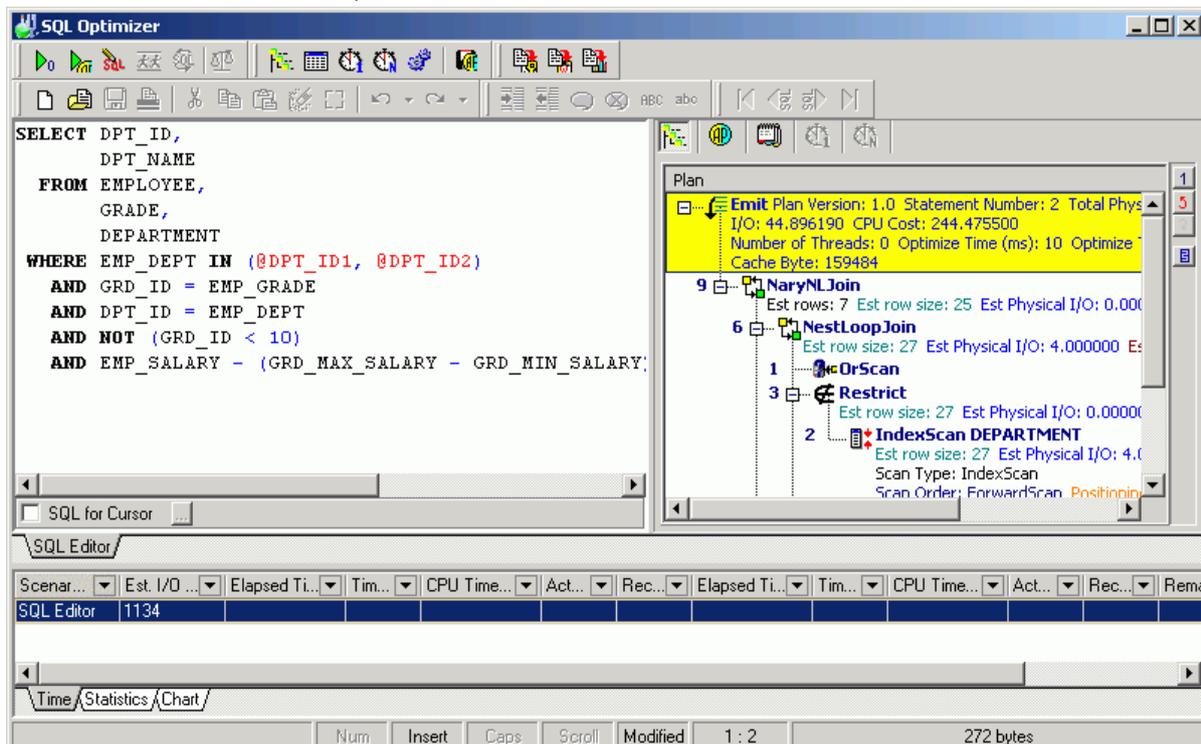
Optimized SQL Functions

These functions are available only for the Optimized SQL.

Button or Menu	Function
SQL Menu	Batch Run/Abort Batch Run
	
SQL Menu	Abstract Plan Matrix
	
SQL Menu	Save Abstract Plan
	
SQL Menu	SQL Comparer
	
SQL Menu	Save Optimized SQL
SQL Menu	Find SQL.../Find Next SQL
Navigate Menu	First SQL/ Previous SQL/ Next SQL/ Last SQL
	
Navigate Menu	Go to SQL
Report Menu	Optimized SQL
View Menu	Show Optimization Details
View Menu	Show Batch Run Details

SQL Optimizer Window

View SQL Editor Pane in SQL Optimizer Window



The SQL Optimizer window allows you to enter a SQL statement for optimization. The optimization process analyzes the original SQL statement entered in the SQL Editor pane and uses the built-in Feedback-Searching engine to reproduce a list of new, semantically equivalent SQL statements.

The SQL Optimizer supports only a single SELECT, SELECT INTO, DELETE, UPDATE, or INSERT SQL statement. To optimize SQL within Transact SQL, use the SQL Scanner module to identify potential problematic SQL statements within the Transact SQL code. Then optimize each SQL statement in the SQL Optimizer module.

The SQL Editor pane of the SQL Optimizer window is where you input your original SQL statement so that you can optimize it.

SQL Text

The top left pane is used to enter and display SQL statements. The SQL Editor tab allows you to enter the SQL statement you want to optimize. When a SQL statement is copied to this pane from other modules, the optimization process automatically begins.

After optimization, multiply tabs are added to display the optimized SQL statements.

SQL Information

The top right pane [SQL Information] displays SQL information according to the button you select from the button bar at the top of this pane. These buttons include: query plan, abstract plan, and Information (SQL classification and warning messages), All Records, and First Record. The [SQL Information Pane](#) provides detailed information about the SQL statement.

Run Time Information

The Run Time information displays in three tabs in the bottom pane.

- [Time](#)
- [Statistics](#)
- [Charts](#)

Related Topics

[Enter Original SQL Statement](#)

[Optimize Original SQL Statement](#)

[SQL Optimizer Overview](#)

[SQL Optimizer Functions](#)

Enter Original SQL Statement

The SQL statement that you want to optimize is referred to as the original SQL statement.

To optimize a SQL statement

1. Click .
2. Enter the SQL statement by typing it in, opening an existing file, or pasting it. Parameters within the SQL statement can be prefixed with a "@" sign or without.

To help construct a SQL statement use the [member lookup](#), [argument lookup](#), [auto correction](#), [indent](#), [outdent](#), [comment](#), and [uncomment](#) functions.

For SQL statements that involve temporary tables, you need to create the temporary table first before optimizing the SQL statement using the User-Defined Temp Table window.

Only one SQL Optimizer window can be opened at any time. The system will prompt you to save the original SQL statement if you attempt to open another file.

You can perform the following functions from the SQL Optimizer window:

- [Retrieve the Query plan](#)
- [Retrieve the Run Time for All Records and/or First Record](#)
- [Retrieve the Run Result](#)
- [Optimize a SQL Statement](#)

Note: The SQL Optimizer supports only a single SELECT, SELECT INTO, DELETE, UPDATE and INSERT SQL statement. To optimize SQL within Transact SQL, use the SQL Scanner module to identify potential problematic SQL statements (e.g. database objects such as Procedures.)

Related Topics

[Insert User-Defined SQL](#)

[SQL Optimizer Overview](#)

[SQL Optimizer Window](#)

Retrieve Query plan

By examining the query plan, you can see exactly how the database executes your SQL statement, helping you judge whether the SQL statement is the most efficient or any changes to the table structure are needed such as adding a new index.

To view the query plan of the original SQL statement in the SQL Optimizer window

Click .

If parameters exist in the original SQL statement, you are required to define the parameters and data types of the parameters before the query plan is retrieved. Or, alternatively use the **SQL | Show Default Plan** to retrieve the query plan without the need to enter parameter details. [Show Default Plan](#) allows you to quickly view the query plan without the need to enter the data type variables.

For more specific information about each step of the query plan, click any text in the step and the [Plan Detail](#) window displays.

Related Topics

[Show Default Plan](#)

[SQL Optimizer Window](#)

Show Default Plan

When the original SQL statement has parameters, you need to determine the data type of the parameters when executing the Show Plan, Optimize, Run Result, Run for all Records, or Run for First Record functions.

If you would like to view the query plan without needing to enter all the data types and values for the parameters, use the **SQL | Show Default Plan** to retrieve the query plan. In this case, all the parameters are assumed to have a BINARY data type when executing the Show Plan function. This is useful when you want to quickly investigate the query plan of the original SQL statement.

Related Topic

[Retrieve Query plan](#)

[SQL Optimizer Window](#)

Optimize Original SQL Statement

If sending a SQL statement from the SQL Scanner, SQL Worksheet, Database Explorer, or SQL Formatter to the SQL Optimizer (**Edit | Send to SQL Optimizer**), these statements are automatically optimized.

To optimize a SQL statement

1. In the SQL Optimizer window, enter the SQL statement in the SQL Editor pane of the SQL Optimizer.
2. Click .

The time it takes to optimize is dependent on the complexity of the original SQL statement and the quota values set in the Preferences window.

During optimization, the unique optimization engine uses multiple SQL syntax transformation rules to produce a list of semantically equivalent SQL statements. Selected optimization forces from the Preferences window are also applied to produce the list of optimized SQL statements.

To stop the optimization process

Click .

It may take a few seconds to terminate all processes.

Note: When setting the searching quota values that, the higher the quota, the longer it may take to optimize a complicated SQL statement.

Related Topic

[Optimize Using Abstract Plan Only](#)

[SQL for Cursor Checkbox](#)

[Open Optimized SQL](#)

[SQL Optimizer Window](#)

SQL for Cursor Checkbox

Adaptive Server uses a different query plan for a SQL statement that is embedded in a cursor declaration from the query plan when the SQL statement is not embedded in a cursor. This needs to be taken into account when retrieving the query plan or run time and also when generating SQL alternatives.

Therefore, if the original SQL statement comes from or will be embedded in a cursor declaration then you need to select **SQL for Cursor** in the SQL Editor pane of SQL Optimizer window. This enables cursor simulation when retrieving the query plan and run time information.

This checkbox is automatically selected in the SQL Editor pane when you use the **Send to SQL Optimizer** function from the SQL Scanner if the SQL was extracted from within a cursor declaration.

In Adaptive Server 15 or later, the Declare Cursor Setting window is also available to select specific cursor settings.

[View Declare Cursor Settings Window](#)



Cursor Arguments

The cursor arguments should match the settings used for the SQL statement in your application code.

Cursor Type	Description
SEMI-SENSITIVE	Specify that the worktable which holds the result set is populated only as the rows are fetched. Therefore changes to the data that occur while the cursor is opened may be visible in the result set.
INSENSITIVE	Specify that the data is copied to a worktable when the cursor is open which makes the data insensitive to changes in the data that may occur while the cursor is opened.
SCROLL	Specify that the cursor is scrollable meaning that you can position the cursor anywhere in the cursor result set for as long as the cursor is open. All scrollable cursors are read only.
NO SCROLL	Specify that the rows are retrieved one row at a time. All update cursors are non-scrollable.

Optimize Using Abstract Plan Only

This function is only available if you are connected to Adaptive Server 15 or later.

The Optimize using abstract plan function optimizes the original SQL with the objective of producing an optimal abstract plan. Therefore the transformed SQL statements are not shown. After optimization, the alternative abstract plans are shown with the original SQL statement in the left pane of the SQL Optimizer window. All abstract plans are compatible with the original SQL statement.

To optimize using only the abstract plan

1. In the SQL Optimizer window, enter the SQL statement in the SQL Editor pane of the SQL Optimizer.
2. Click .

The time it takes to optimize is dependent on the complexity of the original SQL statement and the quota values set in the Preferences window.

To stop the optimization process

Click .

It may take a few seconds to terminate all processes.

Note: After the optimization, the Abstract Plan page remains blank since the abstract plan displays with the original SQL statement.

Insert User-Defined SQL

Once you have entered the original SQL statement in the SQL Optimizer module, you can add your own alternative SQL statement. You can do this either before or after you have optimized the original SQL statement. With this feature, you can benchmark test your own SQL alternatives with the alternatives created by the SQL Optimizer. Or, you can simply test your own alternatives against the original SQL statement.

To insert your own SQL alternative

1. Select the original SQL statement or the alternative SQL statement most like the one you want to insert.
2. Click .
3. Create your SQL statement.

The query plan for your SQL statement is checked to see if it matches any of the query plans for the SQL alternatives or the original SQL. If it does, you will be prompted to decide whether to insert your alternative.

Note: The User-Defined SQL statements are not checked to see if they are semantically equivalent to the original SQL. When you include a User-Defined SQL in a Batch Run, be sure to check the Remarks column of Run Time pane to see if the record count for the User-Defined SQL matches the record count for the original SQL.

Related Topic

[Optimized SQL](#)

Open Optimized SQL

After you have saved the optimized SQL statements to a file, you can load them back through the SQL Optimizer window:

To load a saved optimized SQL statements and alternatives

1. Select **SQL | Open Optimized SQL**.
2. Select the file you want to load and click **Open**. This loads the saved SQL statements to the SQL Optimizer window. The Open Optimized SQL Details window displays the following:

Summary tab

Optimization Information

Displays the original connection and optimization settings information.

Last Saved Query plan Information

Displays the saved and current connection information and whether there are any changes in SQL structure and query plans.

User-Defined Temp Table Tab

If the SQL statement uses a temporary table, the User-Defined Temp Table tab displays in this window. It displays the DDL used to create the temporary table.

Changes Tab

If there are any changes to the SQL structure or the access plans, the Changes tab displays in this window. It displays the SQL text along with the saved and current query plan.

If there are any changes in either SQL structure or query plans it is advisable that you refresh the query plans so that the reloaded image is a truth reflection of the current environment before any further testing is done. Click **Refresh Plan** from the Open Optimized SQL Details window. If there are changes in the query plan the corresponding SQL statements run time and statistics information are deleted. All invalid SQL statements are removed, except for the original SQL statement. You have an option to eliminate SQL statements with duplicate query plans.

After refresh, the Refresh Plan Details window can be displayed. This window displays the number of query plans refreshed, the total eliminated, and invalid plans.

The Open Optimized SQL Details and Refresh Plan Details windows can be reviewed at a later stage.

To view the details

Select **View | Show Open Optimized SQL Details** and **View | Refresh Plan Details**.

Related Topics

[View Open Optimized SQL Details](#)

[Refresh Plan Details Window](#)

[Save Optimized SQL](#)

[Optimized SQL](#)

Create Benchmark Factory Import File

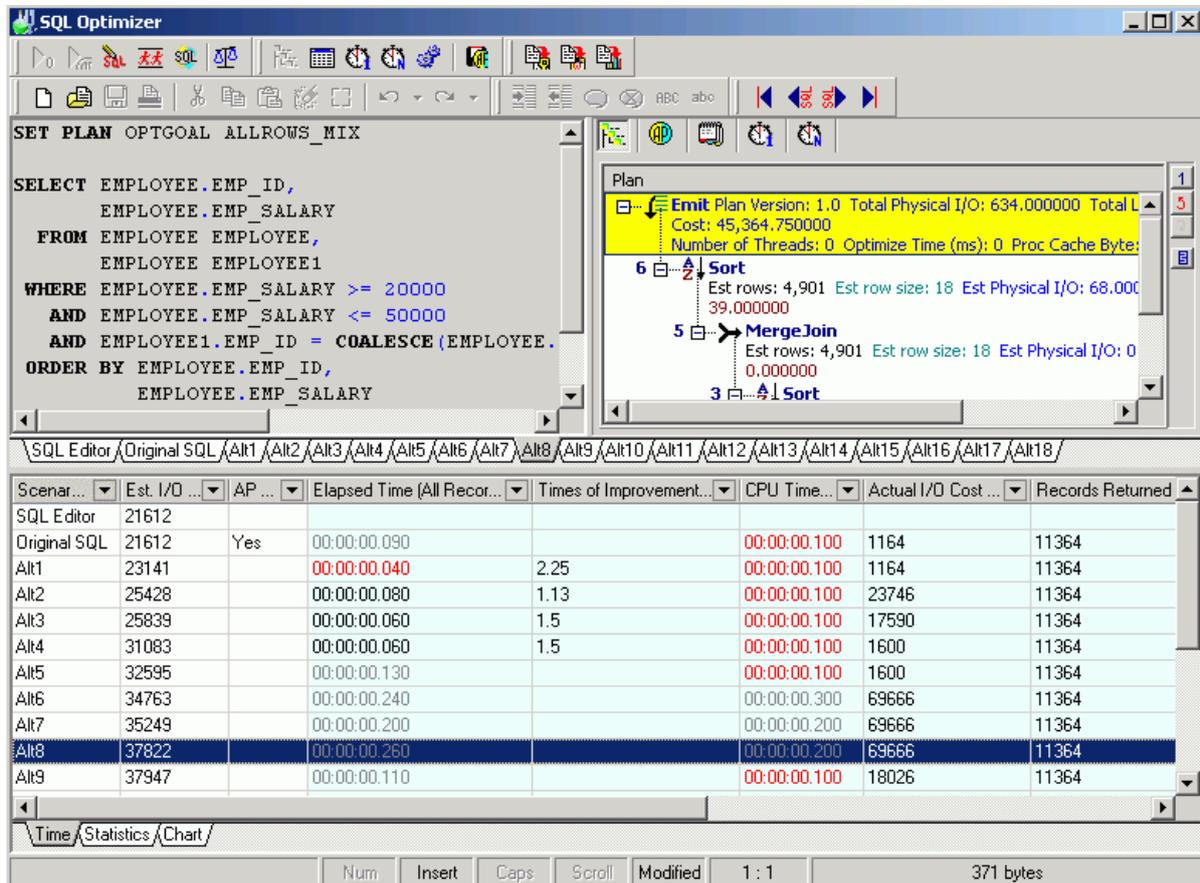
All the SQL statements can be saved in a text file from the SQL Scanner, the SQL Repository, the SQL Inspector, and the SQL Optimizer windows. These SQL statements can then be imported into Benchmark Factory 4.6 or later.

To create a file to import into Benchmark Factory

1. Right-click and select **Create Benchmark Factory Import File**.
2. Select the specific SQL statements that you want to save.
3. Enter the filename and select the file location.

Optimized SQL

View Optimized SQL in the SQL Optimizer Window



The Optimized SQL information displays in the SQL Optimizer window after performing the Optimize function:

SQL Text

The top left pane allows you to select the of semantically equivalent SQL statements produced after optimization from the tabs at the bottom of the pane. The arrangement of the SQL statements on these tabs from left to right is the **SQL Editor**, then the **Original**, followed by optimized SQL statements starting with **Alt1**, **Alt2**, etc. The optimized SQL statements are ranked according to increasing Estimated I/O cost.

Note: The lower the Estimated I/O cost the better the estimated performance of the SQL statement. However, the cost value should not be used as the actual indication of performance.

Auto indentation format

All alternative SQL statements are transformed into a more readable format by automatically indenting and aligning.

Text color

All parameters are in red (default color), indicating that a value and data type needs to be defined upon execution. Other highlights are according to parameters set for syntax highlights under the Preferences window.

Comments

Comments can be entered in the original SQL statement but are not included in the SQL alternatives.

SQL Information

The top right pane, the [SQL Information pane](#), displays detailed information about the SQL statement.

Run Time Information

The Run Time information displays in three tabs in the bottom pane:

[Time](#)

[Statistics](#)

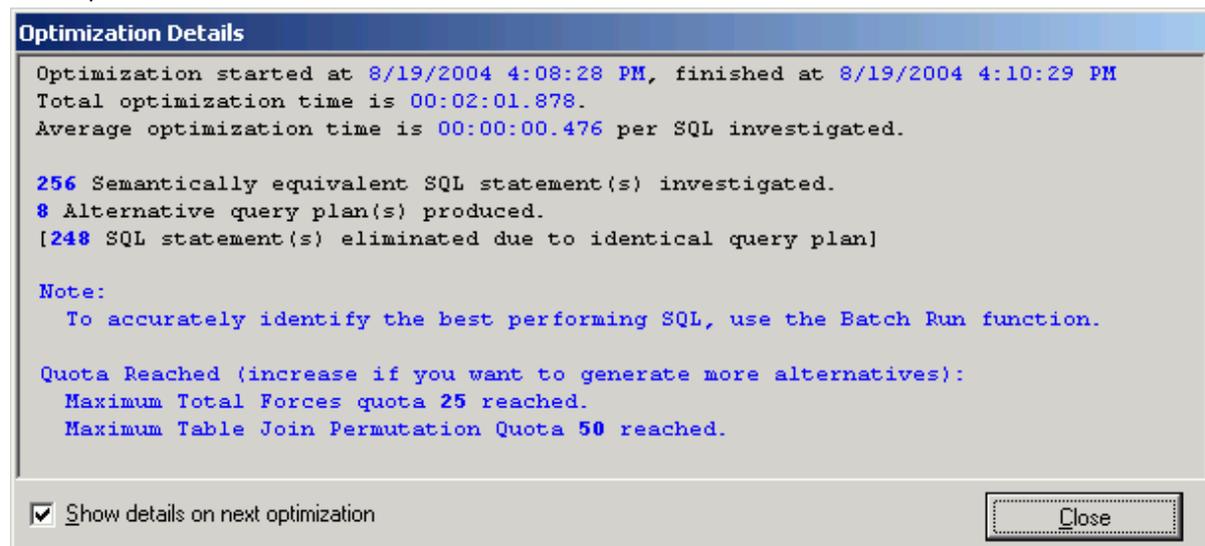
[Charts](#)

Related Topic

[SQL Optimizer Window](#)

Optimization Details Window

View Optimization Details Window



The Optimization Details window is optional and can be displayed after optimization. This window displays the following:

- Time calculations for the optimization process.
- Number of semantically equivalent SQL statements investigated
- Number of alternative query plans produced.
- How many SQL statements were eliminated because they had identical query plans.

- A warning message if the Syntax Transformation Quota, Total Forces Quota, Parallel Quota, Total Quota, or Join Path Permutation Quota is reached.

Note: Generally speaking, SQL statements with the same query plan will produce the same run time because the database executes the SQL statement in the same way. Therefore, newly generated SQL statements with equivalent query plans are eliminated.

The Optimization Details window is shown every time optimization is finished unless it is disabled. This is done by unselecting the **Show details on next optimization** option in the Optimization Details window.

The Optimization Details window can be shown anytime after the optimization process by selecting **View | Show Optimization Details** when the SQL Optimizer window is the active window.

If no alternatives are available, the **Advise Indexes** button is shown for you to directly copy the original SQL statement from the SQL Optimizer to Index Advisor for generation index candidates. The generate process will start automatically.

Related Topic

[Optimize Original SQL Statement](#)

Input Parameter Values

The SQL Optimizer identifies variables defined within a SQL statement and highlights them in red in the SQL Optimizer window. When the test run process begins, the SQL Optimizer prompts you to enter a value for the variable and to select the data type from a list through the [Parameters window](#).

Note: The value and data type entered on the Parameters window have a direct affect of the run time and query plan retrieved.

Analyze Time and Statistics Results

The results from the [Run for All Records](#), the [Run for First Record](#), and the [Batch Run](#) are placed in the bottom Pane of the SQL Optimizer window for you to analyze and pick the SQL statement that is the best performing in your database environment. These results are displayed in the Time, Statistics, and Charts tab.

- [Time](#)
- [Statistics](#)
- [Charts](#)

Analyze Run Time Results

[View Analyze Run Time Results Pane](#)

Scenar...	Est. I/O ...	AP Com...	Elapsed Time (All Re...	Tim...	CPU Time...	Act...	Rec...	Elapsed Time (First n Rec...	Ti...	CPU Time...
SQL Editor	9682									
Source SQL	9682	Yes	00:00:00.070		00:00:00.100	19078	21	00:00:00.000		00:00:00.000
Alt1	4718		00:00:00.060	1.17	00:00:00.000	2688	21	00:00:00.040		00:00:00.000
Alt2	4718		00:00:00.030	2.33	00:00:00.000	2248	21	00:00:00.030		00:00:00.000
Alt3	4764		00:00:00.030	2.33	00:00:00.100	2202	21	00:00:00.030		00:00:00.000
Alt4	5556		00:00:00.030	2.33	00:00:00.000	2376	21	00:00:00.030		00:00:00.000
Alt5	6506		00:00:00.030	2.33	00:00:00.000	2292	21	00:00:00.030		00:00:00.000
Alt6	9646		00:00:00.070		00:00:00.100	19556	21	00:00:00.010		00:00:00.000
Alt7	9646		00:00:00.090		00:00:00.100	19542	21	00:00:00.010		00:00:00.000
Alt8	13164		00:00:00.050	1.4	00:00:00.100	7796	21	00:00:00.000		00:00:00.000
Alt9	13164		00:00:00.070		00:00:00.100	15516	21	00:00:00.010		00:00:00.000
Alt10	13164		00:00:00.050	1.4	00:00:00.100	7880	21	00:00:00.010		00:00:00.000
Alt11	14192		00:00:00.090		00:00:00.100	26916	21	00:00:00.010		00:00:00.000
Alt12	15038		00:00:00.080		00:00:00.100	19252	21	00:00:00.020		00:00:00.000
Alt13	15918		00:00:00.080		00:00:00.100	19254	21	00:00:00.010		00:00:00.000
Alt14	16988		00:00:00.100		00:00:00.100	26916	21	00:00:00.000		00:00:00.000
Alt15	17446		00:00:00.070		00:00:00.100	19078	21	00:00:00.000		00:00:00.000
Alt16	17528		00:00:00.070		00:00:00.100	19196	21	00:00:00.000		00:00:00.000
Alt17	17686		00:00:01.553		00:00:01.500	658156	21	00:00:00.090		00:00:00.100
Alt18	17852		00:00:00.090		00:00:00.000	26798	21	00:00:00.000		00:00:00.000

Scenario Name

The Optimization process numbers the SQL alternatives according to the Estimated I/O cost.

Estimated I/O cost

This is a cost estimation calculated by Adaptive Server. It is only an estimation and should not be used to determine the best performing SQL.

AP Compatibility

This shows the SQL statements that have abstract plans that are compatible with the original SQL statement.

Elapsed Time (All Records and First Record)

The execution time (in clock time) that it takes the SQL statement to select all records or the first record. This figure may include time that Adaptive Server spent on processing other tasks or waiting for disk or network I/O to complete. This is the accumulative run time for the SQL statement. The individual values for the Elapsed Time can be view by clicking **All Records** or **First Record** on the top right pane of the SQL Optimizer window.

Times of Improvement (All Records and First Record)

If the original SQL statement run time is available, the Times of Improvement indicates how much faster the alternative SQL is than the original SQL.

CPU Time (All Records and First Record)

The CPU Time is accumulative total for the CPU time to execute the SQL statement. The individual values for the CPU time can be view by clicking **All Records** or **First Record** on the top right pane of the SQL Optimizer window.

Actual I/O Cost (All Records and First Record)

The Actual I/O Cost is a calculation from Adapter Server after the SQL statement is executed in the **Batch Run** or the **Run Time** function.

Records (All Records and First Record)

The Records column shows the number of records processed by the SQL statement. This number should be constant. It is an indication that the alternative SQL statements are semantically equivalent to the original SQL statement.

Remarks

Information from the Batch Run is included in this column. It includes:

- If the SQL was terminated by the termination criteria.
- If the SQL was run more than once.
- If a database error occurred.

Note: Estimated I/O cost, All Records Actual I/O Cost and First Record Actual I/O Cost are not available for logon users without sa_role privileges or with the "allow resource limits" configuration parameter turned off for the Adaptive Server. To retrieve the Estimated I/O Cost, All Records Actual I/O Cost, and First Record Actual I/O Cost estimations, you should either grant yourself sa_role, logon as another user with sa_role privileges, or turn on the "allow resource limits" parameter:
sp_configure "allow resource limits", 1

Related Topic

[Analyze Time and Statistics Results](#)

Analyze Statistics Results

View Analyze Statistics Window

Scenar...	Elapsed Time (All Rec...	CPU Time...	Act...	Writ...	Sca...	Logi...	Phy...	Apf...	Elapsed Time (First n Rec...	CPU Time...	Act...
Source SQL	00:00:00.070	00:00:00.100	19078	1	3762	9539	0	0	00:00:00.000	00:00:00.000	288
Alt1	00:00:00.060	00:00:00.000	2688	4	78	1146	22	0	00:00:00.040	00:00:00.000	2048
Alt2	00:00:00.030	00:00:00.000	2248	4	79	1124	0	0	00:00:00.030	00:00:00.000	2004
Alt3	00:00:00.030	00:00:00.100	2202	1	78	1101	0	0	00:00:00.030	00:00:00.000	1958
Alt4	00:00:00.030	00:00:00.000	2376	4	99	1188	0	0	00:00:00.030	00:00:00.000	2052
Alt5	00:00:00.030	00:00:00.000	2292	4	78	1146	0	0	00:00:00.030	00:00:00.000	2048
Alt6	00:00:00.070	00:00:00.100	19556	1	3807	9598	20	12	00:00:00.010	00:00:00.000	1098
Alt7	00:00:00.080	00:00:00.100	19542	1	3808	9627	16	8	00:00:00.010	00:00:00.000	1154
Alt8	00:00:00.050	00:00:00.100	7796	1	1925	3898	0	0	00:00:00.000	00:00:00.000	446
Alt9	00:00:00.070	00:00:00.100	15516	1	3828	7758	0	0	00:00:00.010	00:00:00.000	886
Alt10	00:00:00.050	00:00:00.100	7880	1	1946	3940	0	0	00:00:00.010	00:00:00.000	450
Alt11	00:00:00.090	00:00:00.100	26916	1	5710	13458	0	0	00:00:00.010	00:00:00.000	1538
Alt12	00:00:00.080	00:00:00.100	19252	1	3808	9626	0	0	00:00:00.020	00:00:00.000	1152
Alt13	00:00:00.080	00:00:00.000	19254	2	3808	9627	0	0	00:00:00.010	00:00:00.000	1154
Alt14	00:00:00.100	00:00:00.100	26916	1	5710	13458	0	0	00:00:00.000	00:00:00.000	1538
Alt15	00:00:00.070	00:00:00.100	19078	1	3762	9539	0	0	00:00:00.000	00:00:00.000	288
Alt16	00:00:00.070	00:00:00.100	19196	1	3807	9598	0	0	00:00:00.000	00:00:00.000	1098
Alt17	00:00:01.553	00:00:01.500	658156	1	130528	329078	0	0	00:00:00.090	00:00:00.100	3744
Alt18	00:00:00.090	00:00:00.000	26798	1	5665	13399	0	0	00:00:00.000	00:00:00.000	404
Alt19	00:00:00.093	00:00:00.000	26798	1	5665	13399	0	0	00:00:00.010	00:00:00.000	404

The Statistics tab of the Run Time pane in the SQL Optimizer window shows the statistics the accumulative totals for the CPU Time, Scan Count, Logical Reads, Physical Reads, and APF I/Os from the execution of the SQL statement. To see the individual values for the statistics by table, click **All Records** or **First Record** on the top right pane or the Statistics tab at the bottom of the SQL Optimizer window.

For more information about this statistics see the Adaptive Server Performance Tuning: Monitor and Analyzing manual, Chapter 4.

Statistics	Description
(All Records and First Record)	
Elapsed Time	The execution time (in clock time) that it takes the SQL statement to select all records or the first record. This figure may include time that Adaptive Server spent on processing other tasks or waiting for disk or network I/O to complete. This is the accumulative run time for the SQL statement. The individual values for the Elapsed Time can be view by clicking All Records or First Record on the top right pane of the SQL Optimizer window.
CPU Time	The CPU Time is accumulative total for the CPU time to execute the SQL statement. The individual values for the CPU time can be view by clicking All Records or First Record on the top right pane of the SQL Optimizer window.
Actual I/O Cost	This Actual I/O Cost is a calculate combining the Logical and Physical I/O statistics after it was executed in the Batch Run or the Run Time function.
Writes	The Writes is the total number of buffers written to the disk.
Scan Count	The Scan Count represents the number of times a query accessed a particular table.
Logical Reads	The Logical Reads represents the accumulative total for logical read for each table and index used in the SQL statement.
Physical Reads	The Physical Reads represents the accumulative total for logical read for each table and index used in the SQL statement.

APF IOs used

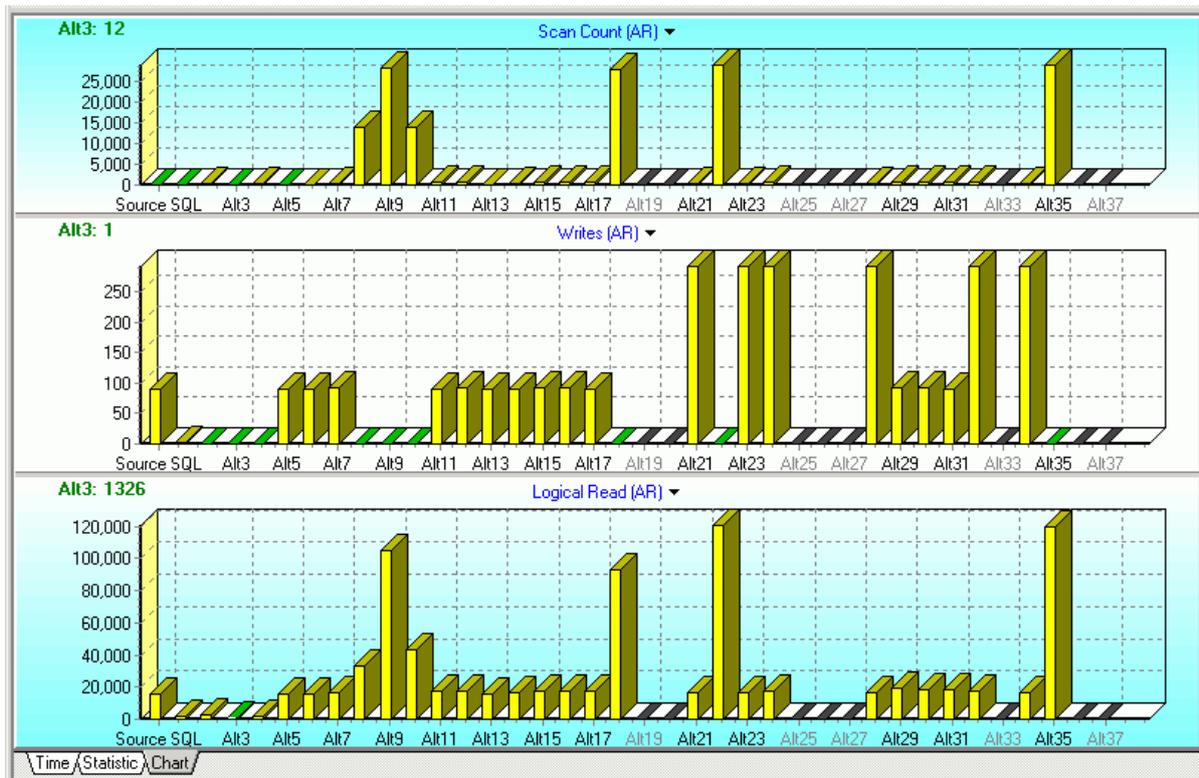
The APF (Asynchronous-PreFetch) IOs used is the I/O the server does in advance anticipating which pages will need to be read next, so that the pages will be in cache when the process actually tries to access them.

Related Topic

[Analyze Time and Statistics Results](#)

Analyze Chart Results

View Chart Results Window



Charts are available for all the statistics and run times. Three chart panes display the run time information.

To change the information in the chart

Click the Chart Title and select the following functionality:

Performance Projection

A performance forecast based on the retrieved Estimated I/O Cost value. If the Performance Projection value is greater than one, then it is forecasted that the SQL alternative may give a better performance.

Estimated I/O cost

Charts the estimated I/O cost from the SQL statement with the corresponding indexes.

All Records and First Record

Charts the following statistics for All Records or First Record after the run time is retrieved for a SQL statement using the Batch Run, Run for All Records, or Run for First Record.

Statistics	Description
Elapsed Time	Charts the execution time (in clock time) that it takes the SQL statement to select all records or the first record. This figure may include time that Adaptive Server spent on processing other tasks or waiting for disk or network I/O to complete.
Times of Improvement	How many times faster the alternative SQL statement is compared to the original SQL.
CPU Time	Charts the execution time (in CPU time) that it takes the SQL statement to select all records or the first record.
Actual I/O Cost	Charts the Actual I/O Cost of executing the SQL statement from Adaptive Server.
Elapsed Time + CPU Time	Charts side by side the Elapsed Time and the CPU Time.
Writes	Charts the accumulative number of disks writes for the SQL statement run time statistics.
Scan Count	Charts the accumulative number of scans for the SQL statement from the run time statistics.
Logical Read	Charts the accumulative number of logical reads for the SQL statement from the run time statistics.
Physical Read	Charts the accumulative number of physical reads for the SQL statement from the run time statistics.
Apf IOs used	Charts the accumulative number of Apf IOs for the SQL statement from the run time statistics.

Other

Charts the following statistics.

Statistics	Description
All Records	Charts the Elapsed Time + CPU Time, Actual I/O Cost, and Times of Improvement for all records.
First Records	Charts the Elapsed Time + CPU Time, Actual I/O Cost, and Times of Improvement for the first record.

All Records vs. First Record

Charts the Elapsed Time for all records and the first record.

All Records vs. First Record
with Cost

Charts the Estimated I/O Cost, the Elapsed Time + CPU Time (for all records), and Elapsed Time + CPU Time (for the first record).

Print

Print the chart.

Save

Save the selected chart.

Hide Chart

Hide the selected chart.

Related Topic

[Analyze Time and Statistics Results](#)

Filter Run Time Results

You can filter the SQL statements displayed on the [Time](#) tab and the [Statistics](#) tab of the SQL Optimizer window by any of the run time statistics after you have executed the SQL statements so that the run time statistics are displayed on the Time tab and the Statistics tab.

The current filter displays in the bottom left corner of the Time or Statistics pane.

To filter the SQL statements

1. Click the downward pointing arrow to the right of a column heading.
2. Select one of the following:

Item	Description
All	Displays all the SQL statements.
Custom...	Opens the Custom AutoFilter window where you can select the options for filtering the SQL statements based on specific values for the selected column.
Blanks	Select all the SQL statements which do not have a value in the selected column.
NonBlanks	Select all the SQL statements which do have a value in the selected column.
Specific Value	Select all SQL statements that have one specific value.

3. You can repeat steps 1 and 2 to add as many filters as you would like.

To clear the filter

Click  at the bottom-left corner of the Time or Statistics pane.

Find SQL Using a Text String

The Find SQL function is available in the SQL Optimizer window. This enables the location of SQL statements that contain a specified text string.

To find text in a SQL statement

1. Select **SQL | Find SQL** to open the Find SQL window.
2. Enter the text string to be found and click **Find**.
3. To continue searching for the same text, select **SQL | Find Next SQL [Ctrl + F3]**.

Related Topic

[SQL Optimizer Window](#)

Generate a Report for Optimized SQL

The details of the SQL statements and run time results in the SQL Optimizer window can be generated into a report.

To generate the Optimized SQL report

1. Select **Report | Optimized SQL** to open the Optimized SQL Report Criteria window.
2. Select the components for the report.
3. Select **All SQL** or **Select SQL** and enter the specific SQL statement numbers.

Note: A few minutes may be needed to generate a long report.

Related Topics

[Optimized SQL](#)

[Optimization Details Window](#)

Verify Correctness of Optimized SQL Statements

You can verify the correctness of optimized SQL statements by comparing it with the original SQL statement using the Run Result function and by looking at the **No. of Records** column in the Time tab in the SQL Optimizer window after retrieving the run time. This information enables you to see whether the optimized SQL statement provides the same results as the original SQL statement.

Run Result

The [Run Result](#) function retrieves the queried records from the connected database.

Number of Records

In the [Run Time](#) pane of the SQL Optimizer window, the **No. of Records** displays the total number of records influenced by the SQL statement. This figure should remain constant for both the original and optimized SQL statements.

First Record

In the [Run Time](#) pane of the SQL Optimizer window, the **First Record** indicates the first record retrieved either 0 or the number of records that was selected for First n Records in the [Run Time Mode](#) page in the Batch Run Criteria window . This figure should remain constant for both the original and optimized SQL statements.

Related Topic

[Optimized SQL](#)

Insert User-Defined SQL

Once you have entered the original SQL statement in the SQL Optimizer module, you can add your own alternative SQL statement. You can do this either before or after you have optimized the original SQL statement. With this feature, you can benchmark test your own SQL alternatives with the alternatives created by the SQL Optimizer. Or, you can simply test your own alternatives against the original SQL statement.

To insert your own SQL alternative

1. Select the original SQL statement or the alternative SQL statement most like the one you want to insert.
2. Click .
3. Create your SQL statement.

The query plan for your SQL statement is checked to see if it matches any of the query plans for the SQL alternatives or the original SQL. If it does, you will be prompted to decide whether to insert your alternative.

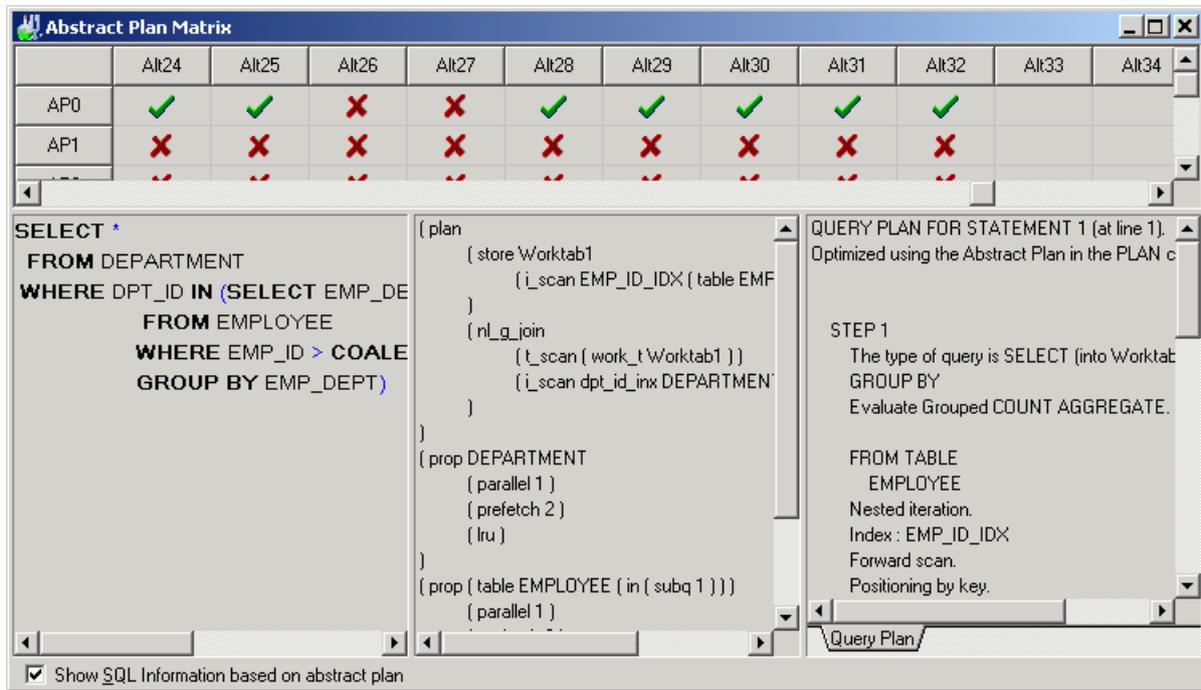
Note: The User-Defined SQL statements are not checked to see if they are semantically equivalent to the original SQL. When you include a User-Defined SQL in a Batch Run, be sure to check the Remarks column of Run Time pane to see if the record count for the User-Defined SQL matches the record count for the original SQL.

Related Topic

[Optimized SQL](#)

Check Abstract Plan Compatibility

View Abstract Plan Matrix Window



To check the compatibility between the original SQL statement and the abstract plan from the optimized SQL statements

Click .

The Abstract Plan Matrix window is mainly divided into a top and bottom section. The top section shows a matrix of the SQL (source, Alt1, Alt2 ...) and the abstract plan (AP0, AP1, AP2 ...). The bottom section has 3 panes which show the SQL text, abstract plan, XML plan, query plan, and trace on details. To display the third pane with the XML plan query plan, and trace on information (the right-most pane), select the **Show SQL Information based on abstract plan** checkbox.

Note: The trace on information is only displayed if the **dbcc traceon** option is selected on the **Database Setting** tab in the Preferences window. The XML Plan is only displayed if you are connected to Adaptive Server 15.0 or later.

The Abstract Plan Matrix provides the following functions:

Function	Description
Check Selected	Check the compatibility of the selected abstract plan and SQL.
Check All	Check the compatibility of all the abstract plans and SQL statements.
Check Column	Check the compatibility of the selected SQL statement with all the abstract plans.
Check Row	Check the compatibility of the selected abstract plan with all the SQL statements.

Related Topics

[Optimize Using Abstract Plan Only](#)

[Abstract Plan Manager Overview](#)

[Abstract Plan Compatibility with Original SQL](#)

[Save Abstract Plan](#)

Save Abstract Plan

To save the abstract plan from the SQL Optimizer window to the database

1. Open the Save Abstract Plan window by clicking .
2. In the **Save to group** drop-down field, select the group where you want to store your abstract plan.
3. In the **SQL** drop-down field, select the SQL statement you want to associate with an abstract plan. This is usually your original SQL statement.
4. In the **Abstract plan from SQL** drop-down field, select the SQL statement that you determined is the best abstract plan for your application.
5. Click **Save**.

Note: Only save an [abstract plan that is compatible](#) with the original SQL text. Otherwise, the abstract plan will not be used the next time you execute the original SQL.

When the Abstract Plan is saved, it is only saved for the user that you are logged on as. In order for another user to use this abstract plan, you must export/import the plan to another user.

At the prompt "**Plan has been created successfully. The id is nnnnnnnnnn**". Click **OK**. The abstract plan is saved in Adaptive Server.

Note: Adaptive Server saves the abstract plan in the sysqueryplans system table. When a query is executed, Adaptive Server looks in the sysqueryplans table for a stored SQL text that matches the query. If a match is found, the saved abstract plan is used to execute the query.

Saving the abstract plan onto the database means that when the same SQL statement is executed, the query plan is based on the abstract plan.

Warning: When saving the abstract plan, Adaptive Server automatically trims the white-spaces from the SQL text replacing it with one space. You need to make sure the SQL statement you execute in your application is the same as the original SQL text that you saved with the abstract plan. If the SQL text does not match, then the abstract plan will not be used.

Here are some examples you need to be aware of:

Spaces in between functions

where substring (EMP_NAME, 1, 5) = 'SMITH'

This is not the same as

where substring(EMP_NAME,1,5) = 'SMITH'

Spaces in between database, scheme and object name

from sqlexp . sqlexp . EMPLOYEE

This is not the same as

from sqlexp.sqlexp.EMPLOYEE

Parameter replacement

```
where EMP_ID = @var_a
```

This is not the same as

```
if @var_a = 56
```

```
where EMP_ID = 56
```

Comments

```
where EMP_ID =123 /* comment */
```

This is not the same as

```
where EMP_ID = 123
```

Related Topics

[Check Abstract Plan Compatibility](#)

[Abstract Plan Manager Overview](#)

[Why Save the Abstract Plan?](#)

[Abstract Plan Manager Window](#)

[Abstract Plan Compatibility with Original SQL](#)

[Use Saved Abstract Plans](#)

Test for Scalability

You can test a single SQL statement to find out how it will perform under different amounts of data in your database in Quest's Benchmark Factory program (version 4.6 or later).

To send SQL statement to Benchmark Factory:

1. Optimize the original SQL statement.
2. Select the original or one of the SQL alternatives
3. Click .

Save Optimized SQL

You can save the Optimized SQL statements. Unlike a report, this file is used to reload the SQL back to the SQL Optimizer window using the **Open Optimized SQL** function.

To save the original SQL and the SQL alternatives

Select **SQL | Save Optimized SQL**.

Related Topics

[Optimized SQL](#)

[Open Optimized SQL](#)

Open Optimized SQL

After you have saved the optimized SQL statements to a file, you can load them back through the SQL Optimizer window:

To load a saved optimized SQL statements and alternatives

1. Select **SQL | Open Optimized SQL**.
2. Select the file you want to load and click **Open**. This loads the saved SQL statements to the SQL Optimizer window. The Open Optimized SQL Details window displays the following:

Summary tab

Optimization Information

Displays the original connection and optimization settings information.

Last Saved Query plan Information

Displays the saved and current connection information and whether there are any changes in SQL structure and query plans.

User-Defined Temp Table Tab

If the SQL statement uses a temporary table, the User-Defined Temp Table tab displays in this window. It displays the DDL used to create the temporary table.

Changes Tab

If there are any changes to the SQL structure or the access plans, the Changes tab displays in this window. It displays the SQL text along with the saved and current query plan.

If there are any changes in either SQL structure or query plans it is advisable that you refresh the query plans so that the reloaded image is a truth reflection of the current environment before any further testing is done. Click **Refresh Plan** from the Open Optimized SQL Details window. If there are changes in the query plan the corresponding SQL statements run time and statistics information are deleted. All invalid SQL statements are removed, except for the original SQL statement. You have an option to eliminate SQL statements with duplicate query plans.

After refresh, the Refresh Plan Details window can be displayed. This window displays the number of query plans refreshed, the total eliminated, and invalid plans.

The Open Optimized SQL Details and Refresh Plan Details windows can be reviewed at a later stage.

To view the details

Select **View | Show Open Optimized SQL Details** and **View | Refresh Plan Details**.

Related Topics

[View Open Optimized SQL Details](#)

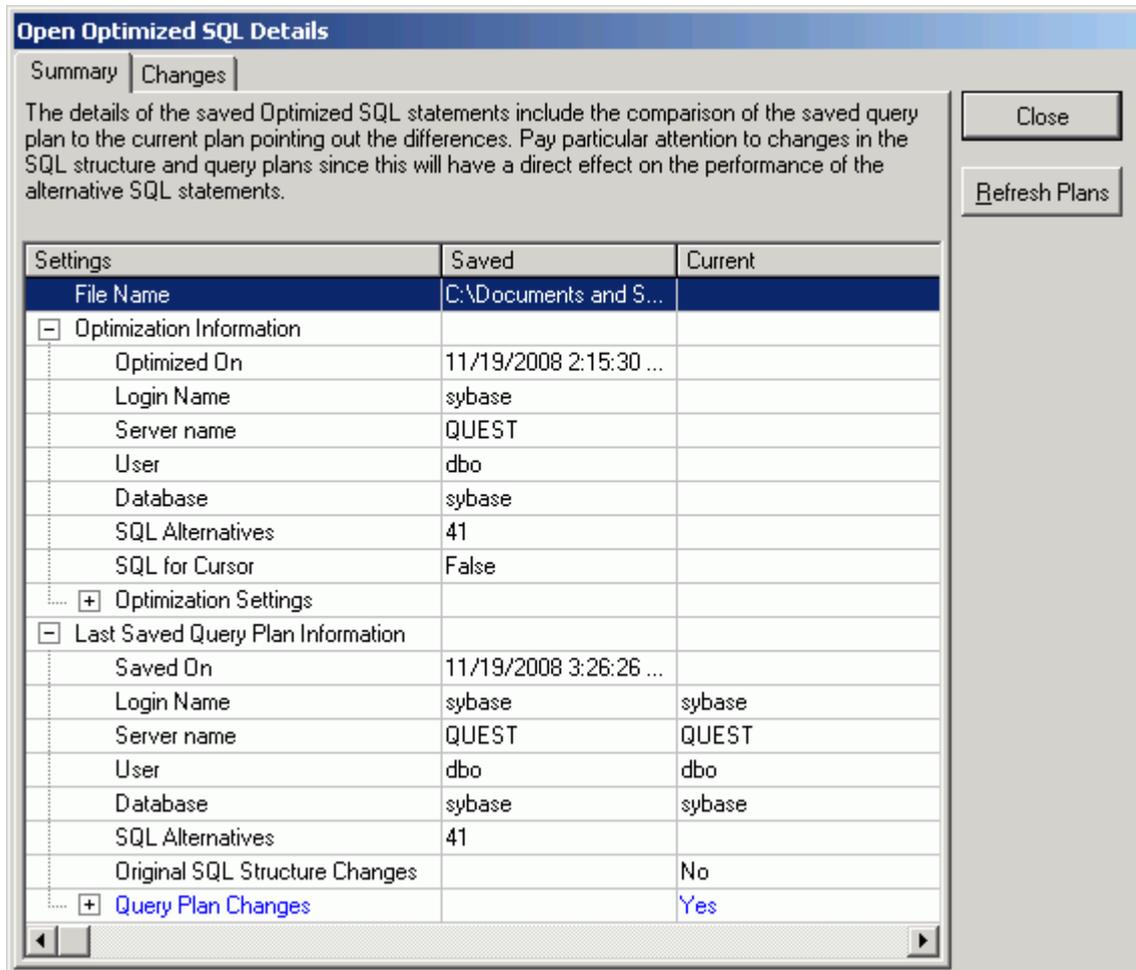
[Refresh Plan Details Window](#)

[Save Optimized SQL](#)

[Optimized SQL](#)

View Open Optimized SQL Details

View Open Optimized SQL Details Window



The Open Optimized SQL Details window displays information about the optimization process and the resulting query plans. The Open Optimized SQL Details window displays after the SQL statements are reloaded into the SQL Optimizer window. You can review the Open Optimized SQL Details window once it has been closed by selecting **View | Show Open Optimized SQL Details** when the SQL Optimizer window is active.

Note: The Optimized SQL Details cannot be viewed if the SQL Editor pane is the active pane in the SQL Optimizer window.

The Open Optimized SQL Details window has the following tabs:

Summary Tab

The Summary tab provides detailed information about the optimization of the SQL statements and changes that may have occurred in the query plans of the original SQL or alternative SQL statements since the optimization process was done.

Optimization Information

The optimization information includes the date and time of the optimization, how many alternatives were created, the connection information, and the optimization settings from the Preferences window at the time of this optimization.

Last Saved Query Plan Information

The query plan information includes the date and time of the SQL statements that were saved, how many alternatives were created, the connection information, and if any of the query plans have changed since the time they were saved.

During the process of opening the Save Optimized SQL, the current query plan is retrieved and compared to the saved query plan. The changes are noted in the Current column as follows:

Current column item	Description
Cost only	The only change to the query plan was a change in the Estimated I/O cost.
Structure and cost	Both the structure and Estimated I/O cost of the query plan changed.
Structure only	The structure of the query plan changed but the Estimated I/O cost remained the same.
No	The current query plan is the same as the query plan that was saved with the SQL statement.

Refresh Plan Button

The **Refresh Plan** button enables you to replace old query plans with the current plans. This process retrieves the current query plans, deletes any SQL statement that is now invalid, removes any run time information. You can select to eliminate SQL alternatives that now have duplicate query plans. After the query plans are refreshed, the [Refresh Plan Details](#) window displays.

Note: The **Refresh Plan** button is only enabled when there is a difference between the current query plan and the saved plan.

Changes Tab

The Changes tab displays the text of the SQL statement in the top pane. The bottom pane displays the saved query plan and the current query plan side by side for comparison.

User-Defined Temp Table Tab

The User-Defined Temp Table tab is available only if the SQL statements use a temporary table. When the Optimized SQL statements were saved, the DDL for creating the temporary table is saved with them and when the SQL statements are reloaded the DDL displays.

Refresh Plan Details Window

The Refresh Plan Details window tells how many SQL statements were refreshed and if any of the SQL alternatives were eliminated due to duplicate query plans or invalid SQL statements. It also displays the SQL for

Cursor setting that was used and tells whether a temporary table was used.

This window is only available after the saved query plans were replaced with the current query plans by clicking **Refresh Plan** on the Open Optimized SQL Details window.

Retrieve Run Time for a Group of SQL

The Batch Run function is used to retrieve the run time of a group of optimized SQL statements.

To start the Batch Run

Click  to open the **Batch Run Criteria** window.

The Batch Run Criteria window is divided into the following tabs:

- Selected SQL
- SQL Termination
- Batch Termination
- Run Time Mode
- Batch Run Schedule

Select your batch run criteria and click **OK**. The Batch Run window displays the run time criteria and the run time of the SQL statements as it is retrieved. **SQL | Stop Current** and **SQL | Abort Batch Run** functions are available to terminate the currently running SQL statement and stop the batch run process. Each selected SQL statement is execute sequentially retrieving the run time unless terminated.

To terminate the currently running of the SQL statement

Select **SQL | Stop Current**.

To terminate the batch run process

Click .

Note: For UPDATE, INSERT, and DELETE SQL statements, while retrieving the run time and run result you may encounter the following Adaptive Server error message:

```
allocate space for object 'syslogs' in database 'sqlexp' because the 'logsegment' segment is full. If
you ran out of space in syslogs, dump the transaction log. Otherwise, use ALTER DATABASE or
sp_extendsegment to increase the size of the segment
```

This is due to the lack of space in system table (syslogs) in which all changes to the database are recorded. Empty the transaction log in the database and re-execute. Use the following command in the SQL Workshop module:

```
DUMP TRANSACTION database_name WITH TRUNCATE_ONLY
go
```

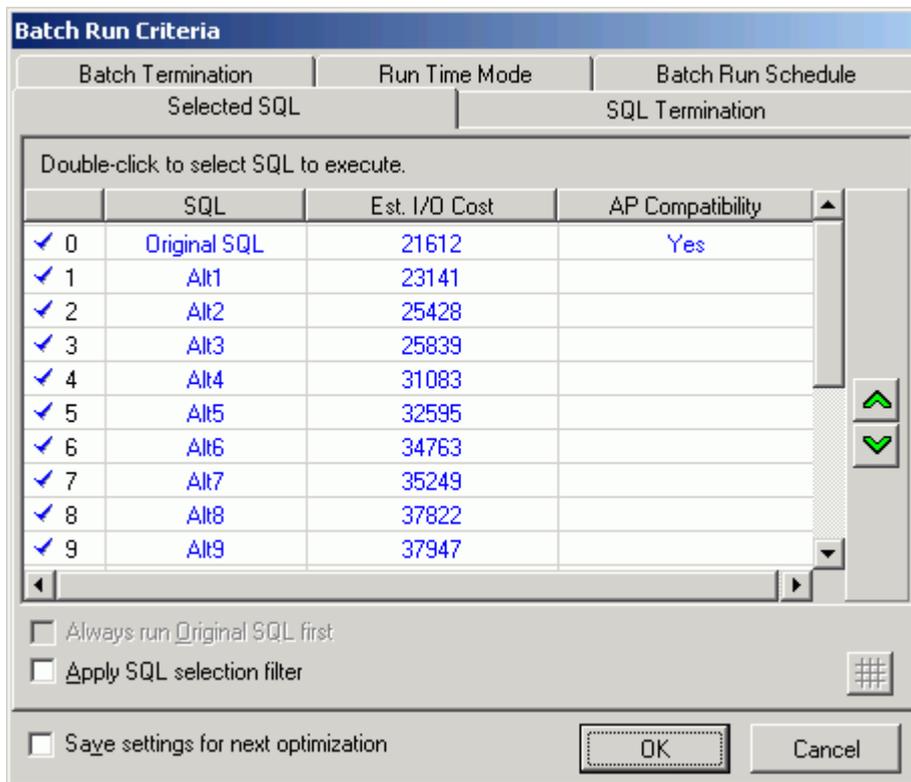
Related Topics

[Run Time](#)

Commit or Rollback
View Batch Run Details
Retrieve Run Time

Selected SQL

View Batch Run Criteria Window - Selected SQL



The Selected SQL tab is used to select or unselect SQL statements to be executed. All SQL statements are selected by default. The selected SQL statements are displayed in blue. You can select or de-select SQL

statements you want to execute by clicking a row. The ✓ blue checkmark at the left of the row indicates the SQL statement is selected.

Selecting or unselecting all SQL statements

To select or unselect all the SQL statements

- Right-click and select Unselect All or Select All.

The original SQL can be de-selected from the list only if the **Original SQL** checkbox is not selected on the SQL Termination and Batch Termination tab and the **Always run Original SQL first** option in the Selected Index tab is not selected.

Changing the order the SQL statements are executed

The SQL statements are ranked by Est. I/O Cost by default, with the exception for the original SQL which is placed at the top. You can sort either the Est. I/O column or the SQL column by clicking the column heading.

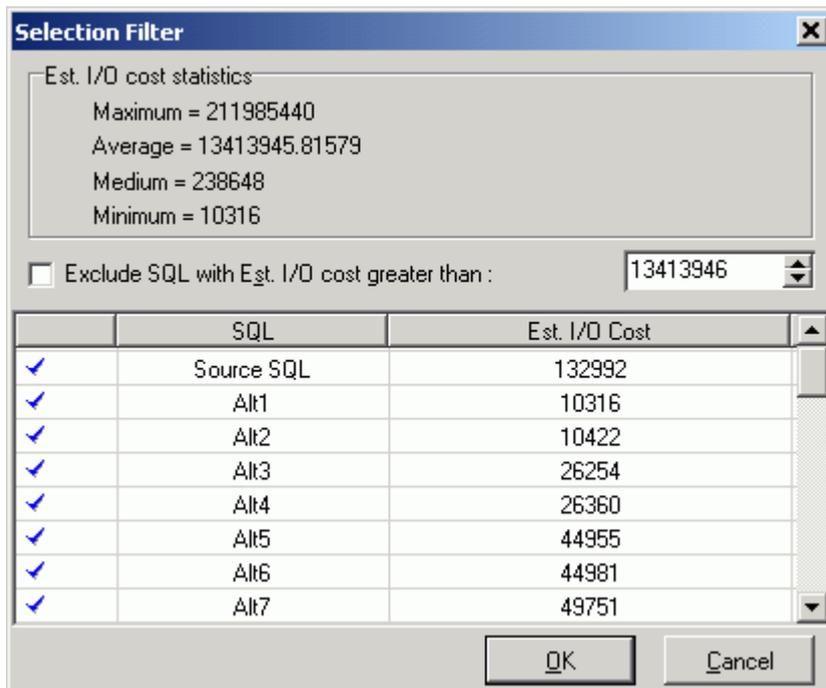
You can change the order of any SQL statement, by clicking the row and then click  or .

Always run Original SQL first

Specify to always run the original SQL statement first despite sort order of the Est. I/O cost. This checkbox is dimmed if the **Original SQL** option is selected in the Termination Criteria or Batch Termination tab of the Batch Run Criteria window.

Apply SQL selection filter

The **SQL selection filter** unselects the SQL statements that have an Estimated I/O cost greater than a specified value. When you select this option,  is enabled at the bottom right of the window. View Selection Filter Window



Related Topic

[Retrieve Run Time for a Group of SQL](#)

SQL Termination (Options)

The SQL Optimizer finds all the alternative SQL statements that produce the equivalent results to your SQL. These statements may run faster than your source. They also may run longer than your source. Therefore, you will want to terminate the longer running SQL statements. The SQL Termination page is used to set the termination criteria for each SQL statement while retrieving the run time. If the current run time for a particular SQL statement exceeds the termination time, that SQL statement is terminated automatically.

Note: The termination time has a [percentage delay](#) added to it.

To define your termination criteria select one of the following options.

Original SQL

The **Original SQL** option enables you to retrieve the run time of SQL statements that run faster than the original SQL statement. It terminates all SQL statements that run longer than the run time from the original SQL. If you choose this option, the original SQL statement is automatically selected on the Selected SQL tab and during the Batch Run, you cannot terminate the original SQL because its run time is needed to determine when to terminate the SQL alternatives.

Best running time SQL

The **Best Running Time SQL** option allows you to retrieve the run time of SQL statements that run faster than the current best run time. With this option, the first SQL statement is run and the time from that statement is used as the termination time. When a SQL statement runs faster than this time, the faster time is used as the new termination time. So you are always using the current fastest run time as the termination time for the next SQL statement.

Note: The first SQL statement is either the original or the SQL with the lowest Estimated I/O cost. This depends on whether the Always run Original SQL first option is checked on the Selected SQL tab.

User-defined time

The **User Defined Time** option retrieves the run time of the SQL statements that are less than the defined time. If your original statement takes a long time to execute and there are many alternative statements, executing all statements may take considerable time. In that event, consider setting an aggressive user defined termination time. If the original SQL takes 1 hour, try a 5-minute termination time. If no alternative statements execute in under that period, raise the termination time to 10 minutes, etc.

Combining Criteria

You can also combine **User Defined Time** with **Original SQL** or **Best Running Time SQL** by clicking the **Or user-defined time** checkbox next to each one.

Run without termination

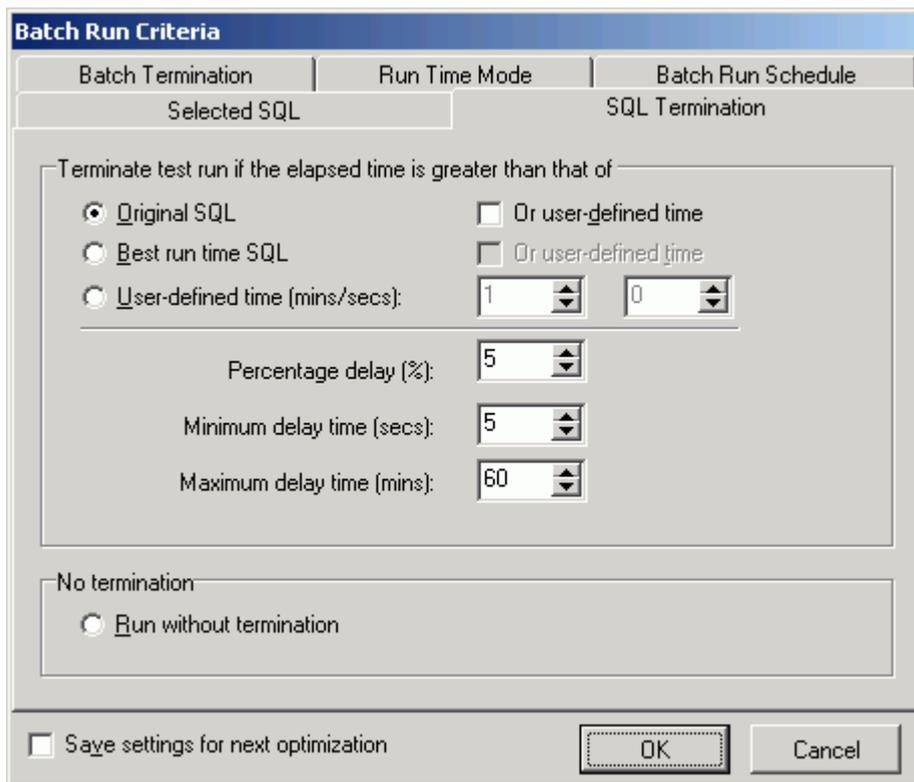
The **Run without termination** option allows you to retrieve the run time of the SQL statements without any termination criteria. All SQL statements run to completion.

Related Topic

[Retrieve Run Time for a Group of SQL](#)

SQL Termination (Percentage Delay)

View Batch Run Criteria Window--SQL Termination



The percentage delay calculation adds additional time to the termination time. It is used to account for the time it takes the SQL statement to travel from the PC to the database server over the network and to find all the SQL statements that terminate very close the fastest SQL statement.

Percentage delay (%) - Range 1 - 200 (Default: 5)

The value entered into this field is used as a percentage to calculate the additional time that is added to the termination time. For example, if the termination time is 10 minutes and the percentage delay is 5%, then all SQL statements executed are terminated if the run time exceeds 10.5 minutes. (10+ (10 * 5%))

Minimum delay time (secs) - Range 2 - 99 (Default: 5)

This is the minimum number of seconds that is added to the termination time. It is necessary to factor into the overall termination time the time needed for the SQL statement to be sent to the database server from the PC before it starts to run. This number is only used if the Percentage delay calculation is lower than this value.

Maximum delay time (mins) - Range 1 - 9999 (Default 60)

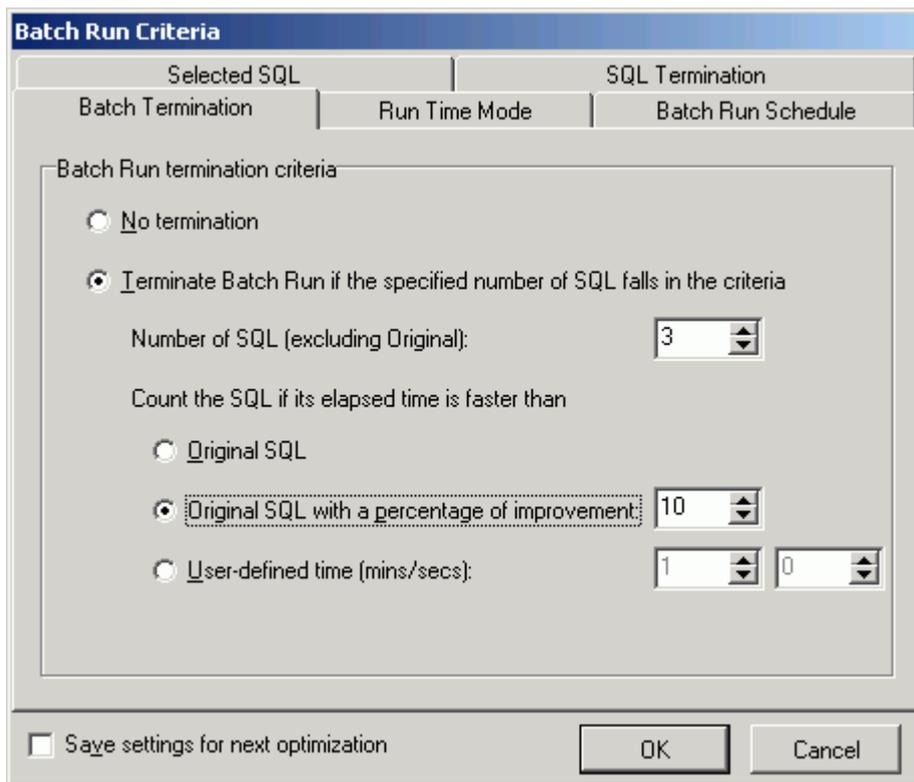
This is the maximum number of minutes that can be added to the termination time. This number is only used if the Percentage delay calculation is higher than this value.

Related Topic

[Retrieve Run Time for a Group of SQL](#)

Batch Termination

View Batch Run Criteria Window - Batch Termination



The Batch Termination page of the Batch Run Criteria window is used to determine if and when to terminate the Batch Run. It enables you to find alternative SQL statements that give you good performance improvement without having to execute every SQL alternative.

No termination

Specify to run the Batch Run to completion.

Terminate Batch Run if the specified number of SQL falls in the criteria.

Specify to terminate the Batch Run when a specified number of SQL statements are found that meet the following requirements for terminating the Batch Run.

Number of SQL (excluding the Source)

Specify how many SQL statements must be found that show performance improvement over the original SQL statement.

Count the SQL if its elapsed time is faster than

Specify one of the following criteria to determine how the performance improvement is determined.

Original SQL

Count all SQL statements that run faster than the run time from the original SQL.

Original SQL with a percentage of improvement

Count all SQL statements where the run time for the alternative SQL statement is the specified percentage faster than time for the original SQL statement.

User-defined time (mins/secs)

Count all SQL statements that run faster than a specified number of minutes and/or seconds.

Related Topic

[Retrieve Run Time for a Group of SQL](#)

Run Time Mode

The Batch Run is designed to give the most accurate result by providing options for obtaining the most accurate run time taking into account the effects of caching the data, indexes and the SQL statements. This section allows you to select one of the four options best suited to your SQL statement.

Run to retrieve	Description
All records	Specify to retrieve the run time for processing All Records.
First <i>n</i> Records(s)	Specify to retrieve the run time for processing <i>n</i> records where you specify the number of records retrieved.
Retrieve the run time by executing	Description
Run SQL options	Select one of the following options: <ul style="list-style-type: none">• Run all SQL twice if original SQL runs faster than (seconds)—Combines the Original SQL twice and all others once and the All SQL twice options into one option and allows you to determine (by the number of seconds a SQL statement runs) which option to use. The original SQL statement always runs twice. The SQL alternatives run twice if the original SQL statement runs in less time than the value specified. Otherwise, the SQL alternatives all run once.• First one twice using the second run time and others once—The first time you access data from table, the data is cached into memory. This process takes a few moments. The next time you access that data, it is already in memory so the following SQL statements run faster. So to have a comparable test, the first SQL statement is run twice and the time from the second run is compared to the time from the other statements.• All once—For long running SQL statements, there is no need to run them twice since the effect from caching diminishes over time.• All twice—Running all SQL statements twice enables you to eliminate some factors that can affect the accuracy of the results. If some SQL statements have been recently executed, then the SQL information is likely to be resident in cache and it may execute faster because of that. Also, if the SQL statements use different indexes, some indexes may be resident in cache and the others are not. Five methods of calculating the run time are available if you select this option: Excluding the first run time, Average, Sum, Maximum, and Minimum.

- Excluding the first run time—Specify to throw out the first run time and use the second one when all SQL statements should have the necessary items in cache. This minimizes the effect that caching of the SQL statement and the indexes may have on the accuracy of the run time results.
- All multiple times—This option is suitable for SQL statements with very short run times. It executes the SQL statements the selected number of times. It calculates the run time based upon the selected calculation method. This option helps to negate the factor of other activity running on the CPU at the same time as the SQL is executed. Since the run time is calculated from the moment the SQL statement starts running to the time it finishes, if SQL statement shares the CPU with other activity, the run time includes this other activity. Four methods of calculating the run time are available if you select this option: Average, Sum, Maximum, and Minimum.
This option is only available for SELECT statements since the INSERT, UPDATE, and DELETE statements would need a rollback between each of the multiple executions.

Include trace statistics

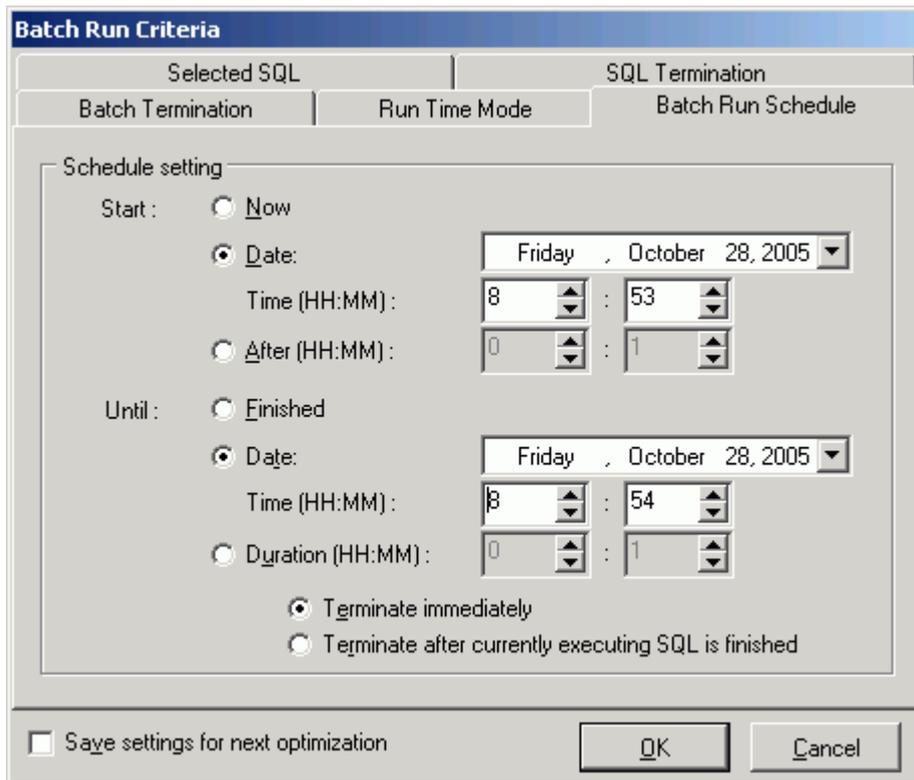
Select this checkbox to include trace statistics in the retrieved SQL.

Related Topic

[Retrieve Run Time for a Group of SQL](#)

Batch Run Schedule

View Batch Run Criteria Window--Batch Run Schedule



The Batch Run Schedule criteria enable you to schedule when to start and stop the Batch Run.

Schedule setting

Start

Now

Specify to start the Batch Run immediately.

Date and Time

Specify to start the Batch Run at a specific date and time.

After

Specify to start the Batch Run after it has been running for a specified number of hours and minutes.

Until

Finished

Specify to run the entire Batch Run until all SQL statements are executed.

Date and Time

Specify to terminate the Batch Run on the specified date and time.

Duration

Specify to terminate the Batch Run after it has executed a specified number of hours and minutes.

Terminate immediately (Date/Time and Duration only)

Specify to terminate the currently executing SQL statement and all remaining unexecuted SQL statements.

Terminate after currently executing SQL is finished (Date/Time and Duration only)

Specify to finish running the currently executing SQL statement and do not run the remaining unexecuted SQL statements.

Related Topic

[Retrieve Run Time for a Group of SQL](#)

Commit or Rollback

In order to obtain the correct result during the Batch Run or Run Time of alternative SQL statement, the SQL statement executes against the database. However, when executing UPDATE, INSERT, DELETE statements, any changes to the database are rolled back.

Note: For UPDATE, INSERT and DELETE SQL statements, while retrieving the run time and run result you may encounter the following Adaptive Server error message:

Can't allocate space for object 'syslogs' in database 'database_name' because the 'logsegment' segment is full. If you ran out of space in syslogs, dump the transaction log. Otherwise, use ALTER DATABASE or sp_extendsegment to increase the size of the segment

This is due to the lack of space in system table (syslogs), in which all changes to the database are recorded. Empty the transaction log in the database and re-execute. Use the following command in the SQL Workshop module:

```
DUMP TRANSACTION database_name WITH TRUNCATE_ONLY  
go
```

Related Topic

[Retrieve Run Time for a Group of SQL](#)

View Batch Run Details

The Batch Run Details window displays a summary of the run time information for all SQL statements executed. The Batch Run Details window displays after the batch run process is completed unless the **Show details on next batch run** checkbox in the Batch Run Details window is unchecked.

To review the Batch Run Details window after a Batch Run

Select **View | Show Batch Run Details** when the SQLOptimizer window is active.

If a SQL statement has a database error during the Batch Run, an explanation of the error is included in the Batch Run details.

Related Topic

[Retrieve Run Time for a Group of SQL](#)

Run Time

Run time can be generated for the original and optimized SQL statements to help you choose the best performing SQL statement that is most suitable for your application. All run times are calculated to the nearest milliseconds in format HH:MM:SS.MS. The actual execution time of each SQL statement can be obtained by executing the [Run for First Record](#), [Run for All Records](#) and [Batch Run](#) functions.

Excludes network traffic time

The calculation of the run time is based on the CPU time of the database server. Thus network traffic is excluded from the time.

Fluctuation of All Records - or - First Record Time

If you attempt to execute the SQL statement more than once, you will notice a slight fluctuation in the run time. This fluctuation is due to data cache in the memory and other processes on the CPU and database server. Therefore, for a more stable result, it is advisable to repeat the test run process more than once.

Retrieve run time for cursor

For SQL statements embedded in the cursor, the run time is obtained by creating a stored procedure, executing it, and then dropping it.

Note: For UPDATE, INSERT and DELETE SQL statements, while retrieving the run time and run result you may encounter the following Adaptive Server error message:

```
Can't allocate space for object 'syslogs' in database 'database_name' because the 'logsegment'
segment is full. If you ran out of space in syslogs, dump the transaction log. Otherwise, use
ALTER DATABASE or sp_extendsegment to increase the size of the segment
```

This is due to the lack of space in system table (syslogs), in which all changes to the database are recorded. Empty the transaction log in the database and re-execute. Use the following command in the SQL Workshop module:

```
DUMP TRANSACTION database_name WITH TRUNCATE_ONLY
go
```

Retrieve Run Time

Two run time functions are available for retrieving how long it takes to run the SQL statement: **Run for First Record** and **Run for All Records**. Both of these functions return run time information. You should first determine the aim of the SQL statement before retrieving the run time information.

If the SQL statement is used for batch job, you want the best time for retrieving all records. Use .

If the SQL statement is used for on-line inquiry, then you may also want to know how long it takes to retrieve the first record. Use .

The time displays in the bottom section of the SQL Optimizer window in the Time tab.

Note: For UPDATE, INSERT, and DELETE SQL statements, retrieving the First Record information by using the Run for First Record function is irrelevant as these SQL statements are processed all at once. Therefore, retrieving the First Record for UPDATE, INSERT and DELETE SQL statements will give the same result as retrieving the All Records.

Related Topics

[Commit or Rollback](#)

[Retrieve Run Time for a Group of SQL](#)

Terminate a Run Time SQL Statement

To terminate the run time process for Run for All Records

Select **SQL** option.

The termination of the run time is not instantaneous because it takes time to rollback the transaction and close all opened processes.

Related Topic

[Retrieve Run Time](#)

[Run Time](#)

Retrieve Run Result

View Retrieve Run Result Window

The screenshot shows a 'Run Result' window with two panes, SQL1 and SQL2. Each pane contains a SQL query and a table of results. The SQL1 query filters for EMP_ID = 73712, and the SQL2 query filters for EMP_ID between 73712 and 73712. Both result sets show three rows of data with columns EMP_ID, EFFECT_DATE, and SALARY.

EMP_ID	EFFECT_DATE	SALARY
73712	1/1/1990	63666
73712	1/1/1991	64666
73712	1/1/1993	65666

To retrieve Run Result data from the database

Click  in the SQL Optimizer window.

If the original SQL statement contains a result set then the Run Result window displays the data retrieved; otherwise it will display an information dialog field showing the number of rows affected.

You can display up to 4 result sets in the Run Result window.

Notes:

- When you are in the SQL Editor, you are prompted to commit or rollback all affected records for UPDATE, INSERT and DELETE SQL statements.
- For UPDATE, INSERT and DELETE SQL statements, while retrieving the run time and run result you may encounter the following Adaptive Server error message:
Can't allocate space for object 'syslogs' in database 'sqlexp' because the 'logsegment' segment is full. If you ran out of space in syslogs, dump the transaction log. Otherwise, use ALTER DATABASE or sp_extendsegment to increase the size of the segment.

This is due to the lack of space in system table (syslogs), in which all changes to the database are recorded. Empty the transaction log in the database and re-execute. Use the following command in the SQL Workshop module:

```
DUMP TRANSACTION database_name WITH TRUNCATE_ONLY

go
```

Related Topic

[Terminate Retrieval of Run Result](#)

Commit or Rollback

If you execute Run Result from within the SQL Editor pane in the SQL Optimizer window, two types of action - commit or rollback - can be made after the Run Result is executed. In this case, you are prompted to commit or rollback for UPDATE, INSERT, DELETE and SELECT INTO SQL statements.

For the optimized SQL, all records affected by the UPDATE, INSERT, DELETE, and SELECT INTO SQL statements are rolled back automatically.

Terminate Retrieval of Run Result

You must logon with sa to terminate SQL statements with a result set, i.e., SELECT SQL statements. Run Result for SQL statements without result set (for example: INSERT, UPDATE, DELETE and SELECT with INTO clause) can be terminated by any logon account.

To terminate the Run Result process

Click .

SQL Comparer Overview

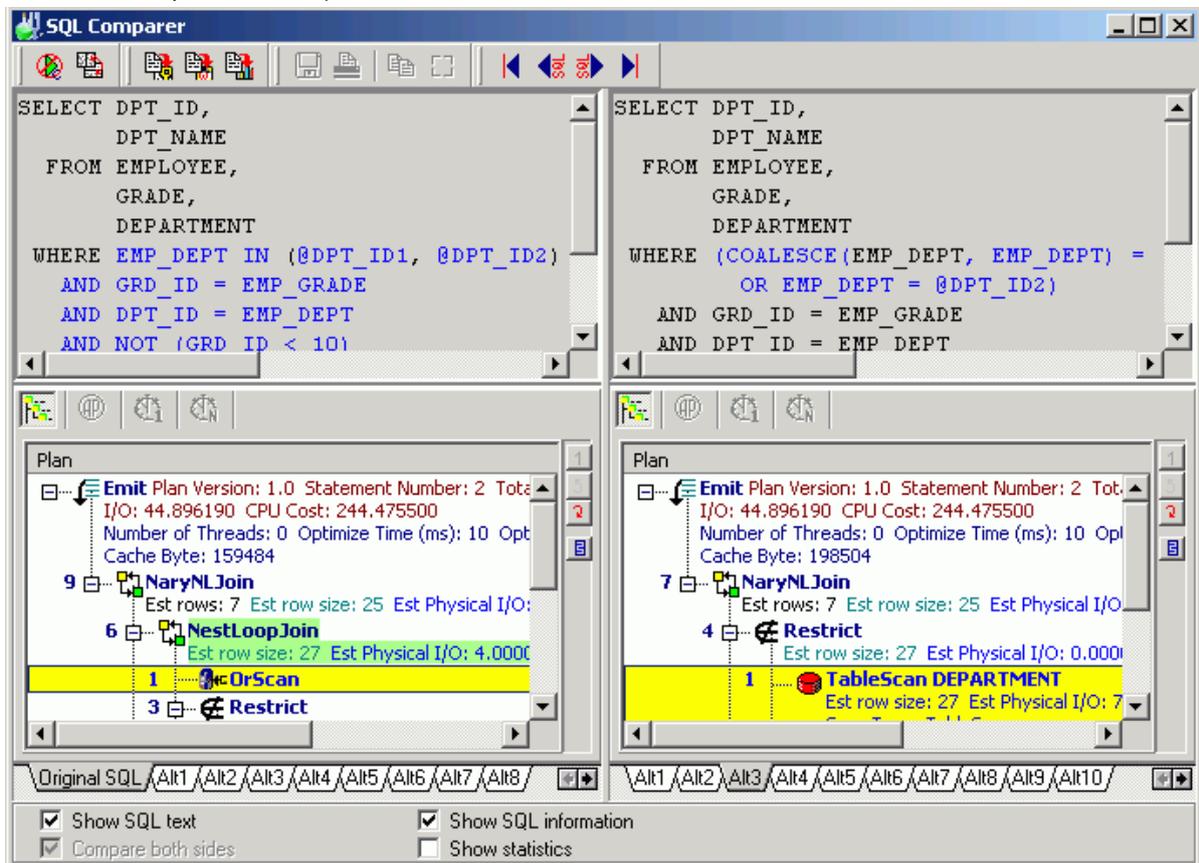
After optimization, the SQL Comparer window provides a way for you to easily compare two SQL statements, query plans, statistics, and abstract plan between the original SQL and optimized alternative SQL statements.

Related Topics

[Open SQL Comparer Window](#)

SQL Comparer Window

View SQL Comparer Window)



The SQL Comparer window is divided into 4 sections: SQL text, query plan, statistics, and abstract plan.

Source of Comparison

A tab showing a list of SQL statements, the selected tab will be the source of comparison.

Show SQL text checkbox

By default, the **Show SQL text** checkbox is selected to display the comparisons between two SQL texts.

Show SQL Information checkbox

You can display the query plans and the abstract plans by selecting the **Show SQL Information** checkbox which opens the [SQL Information pane](#).

Show statistics checkbox

Displays the run time information for the original SQL statement and the selected alternative. The statistics are displayed for both the First Record and All Records if you have retrieved both times.

Compare both sides checkbox

Specify whether to highlight the differences in both SQL statements by selecting the **Compare both sides** checkbox.

Note: You can control the color of the highlighting in the Preferences window under the Syntax Highlight on the Editor tab.

Related Topics

[Open SQL Comparer Window](#)

[SQL Comparer Overview](#)

[SQL Comparer Functions](#)

Open SQL Comparer Window

After optimization and when the SQL Optimizer window is active, click . By default, the original SQL statement is selected as the source of comparison and the differences are **highlighted in blue** (default) on the right or bottom pane of the window.

Note: To use the SQL Comparer, you must have at least one alternative SQL statement from the optimization process.

Related Topics

[SQL Comparer Overview](#)

[SQL Comparer Window](#)

Start and Stop Comparison

When SQL Comparer is launched, it automatically highlights the differences between the two SQL statements in SQL text panes. Click the **Compare both sides** checkbox to see the differences highlighted in both SQL text

panes. If you want to remove the highlighting of the SQL text, click .

To display the comparison

Click .

Related Topics

[SQL Comparer Overview](#)

[SQL Comparer Window](#)

Switch View in SQL Comparer Window

The sections of the SQL Comparer window can be switched from horizontal to vertical and vice versa.

To switch the display orientation

Click .

Related Topic

[SQL Comparer Window](#)

SQL Comparer Functions

Below is a list of available functions within the SQL Comparer window.

Menu or Button	Function
Compare Menu 	Start Compare / Stop Compare
Compare Menu 	Rotate
Navigate Menu 	First SQL / Previous SQL / Next SQL / Last SQL
Navigate Menu	Go to SQL
Edit Menu 	Send to Index Advisor
Edit Menu 	Copy to SQL Worksheet
File Menu	Save SQL to SQL Repository

Related Topics

[SQL Comparer Overview](#)

[SQL Comparer Window](#)

Activity Log Overview

The Activity Log is designed to provide management reports on the effective usage of SQL Optimizer on each individual client PC. The Activity Log records the activities during the SQL optimization process and query plan generation. The information recorded can be used to review the number of SQL statements analyzed, number of SQL statements optimized, percentage of performance improvement and also the original SQL text and the best alternative SQL.

Related Topics

[Start Recording Activities to the Activity Log](#)

[View Activity Log Report](#)

[Purge Activity Log Data](#)

[Stop Recording Activities](#)

[Activity Log Preferences](#)

Start Recording Activities to the Activity Log

By default, the Activity Log is disabled during the SQL optimization process and query plan retrieval,

To start recording the activities

1. Click .
2. Select the Activity Log tab.

Activity to be logged

Specify the activity to be logged by selecting the **SQL optimization** and/or the **Query plan generation** checkboxes. If no option is selected, the activity log is disabled.

Information to be logged

Specify the information to be collected by selecting the **SQL text** and/or **Query plan** checkboxes.

Note: Only activities executed in the SQL Optimizer window are logged.

The following information is recorded automatically for each activity logged:

PC user

PC user name.

Logon name

Database logon name.

Server

Adaptive Server name.

Activity Log directory

The Activity Log directory is used to store the log file created while recording the activities. Select the **Directory Setup / Linkage** tab to change the directory where the log is written. The default is the installed directory.

For example C:\Documents and Settings\User\Application Data\Quest Software\SQL Optimizer

Note: It is advisable not to change the Activity Log directory after selection, as log file is saved to this directory.

Related Topics

[Activity Log Overview](#)

[View Activity Log Report](#)

[Purge Activity Log Data](#)

[Stop Recording Activities](#)

[Activity Log Preferences](#)

View Activity Log Report

View Activity Log Report Criteria Window

The screenshot shows the 'Activity Log Report Criteria' dialog box. It is divided into three main sections. The top-left section, 'Information to be displayed', contains five checkboxes: 'SQL text', 'Query plan', 'Abstract plan', 'Remark', and 'Summary only'. The top-right section, 'Order by', has a 'Descending' label and a list of six items with checkboxes: 'Connection host string', 'Logon name', 'Database', 'User', 'Date', and 'Activity'. The bottom section, 'Range', has two radio buttons: 'Whole log' (which is selected) and 'From (DD-MM-YYYY)'. To the right of the second radio button are two text input fields for dates, with the second field containing the date '12-11-2003'. At the bottom of the dialog are 'OK' and 'Cancel' buttons.

You can save or print the information in the report.

To print or view information stored in the activity log

1. Select **Report | Activity Log** to open the Activity Log Report Criteria window.
2. Select the information to view.
3. Click **OK** to generate the report.

The following information may be displayed depending on the criteria you select.

User Information

Item	Description
PC User	Windows user name used to logon to the PC.

Logon Name	Database logon name.
Server Name	Adaptive Server name.
User	Database user.
Database	Database name.

Activity Information

Item	Description														
Date	Date of when the activity was executed.														
Activity	Activity type SQL optimization (OP) Query plan generation (EP)														
Original SQL Type	The SQL Classification of SQL the original SQL (Simple, Complex or Problematic).														
<i>The information below is repeated for ALL RECORDS and FIRST RECORD.</i>															
Status	The current status of the activity. <table> <tr> <td>PGC</td> <td>Plan generation completed</td> </tr> <tr> <td>OPA</td> <td>Optimization aborted</td> </tr> <tr> <td>OWT</td> <td>Optimization without any test run</td> </tr> <tr> <td>TAI</td> <td>Improvement found (after test running all SQL alternatives)</td> </tr> <tr> <td>TAN</td> <td>No improvement (after test running all SQL alternatives)</td> </tr> <tr> <td>WAI</td> <td>Improvement found (without testing running all SQL alternatives)</td> </tr> <tr> <td>WAN</td> <td>No improvement (without testing running all SQL alternatives)</td> </tr> </table>	PGC	Plan generation completed	OPA	Optimization aborted	OWT	Optimization without any test run	TAI	Improvement found (after test running all SQL alternatives)	TAN	No improvement (after test running all SQL alternatives)	WAI	Improvement found (without testing running all SQL alternatives)	WAN	No improvement (without testing running all SQL alternatives)
PGC	Plan generation completed														
OPA	Optimization aborted														
OWT	Optimization without any test run														
TAI	Improvement found (after test running all SQL alternatives)														
TAN	No improvement (after test running all SQL alternatives)														
WAI	Improvement found (without testing running all SQL alternatives)														
WAN	No improvement (without testing running all SQL alternatives)														
Original SQL Time	The run time of the original SQL statement.														
Best Alternative Time	Best run time from the alternative SQL statements.														
Times of Improvement	Calculates how much faster the alternative is in comparison to the original SQL statement.														

Note: The activity log information is retrieved from the Activity Log directory; therefore if any changes are made to this directory the information may not be retrieved.

Related Topics

[Activity Log Overview](#)

[Start Recording Activities to the Activity Log](#)

[Purge Activity Log Data](#)

[Stop Recording Activities](#)

[Activity Log Preferences](#)

Purge Activity Log Data

To purge the activity log data

1. Click the .
2. Select the Activity Log tab.
3. Under the [Housekeeping section](#), select either Whole Log to remove all information or specify a date range to remove logs between these dates.
4. Click **Purge Now**.

Related Topics

[Activity Log Overview](#)

[Start Recording Activities to the Activity Log](#)

[View Activity Log Report](#)

[Stop Recording Activities](#)

[Activity Log Preferences](#)

Stop Recording Activities

To stop recording activities to the activity log:

1. Click .
2. Select the Activity Log tab.
3. Under the Activity to be logged, de-select the SQL optimization and **Query plan** generation checkboxes to disable the activity log.

Related Topics

[Activity Log Overview](#)

[Start Recording Activities to the Activity Log](#)

[View Activity Log Report](#)

[Purge Activity Log Data](#)

[Activity Log Preferences](#)

SQL Worksheet Overview

The SQL Worksheet allows you to enter and execute multiple SQL statements, database objects, Transact SQL scripts. Supported SQL statements include DML and DDL commands. Local temporary table operations are allowed.

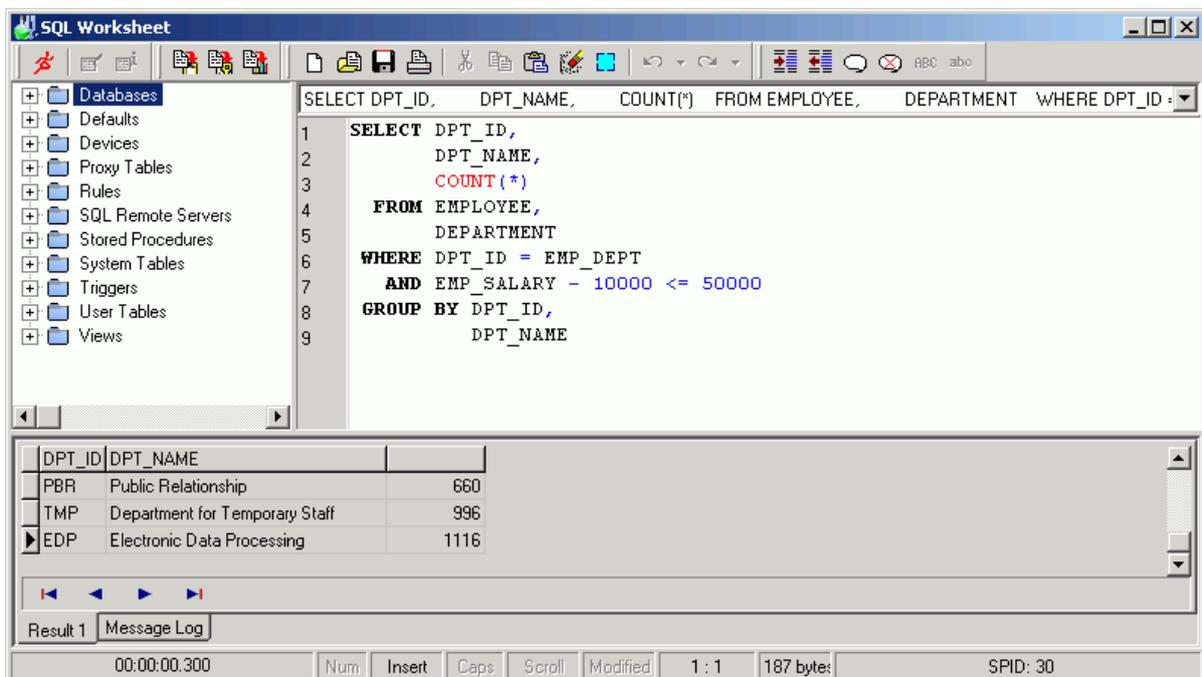
Related Topics

[SQL Worksheet Window](#)

[SQL Worksheet Functions](#)

SQL Worksheet Window

View SQL Worksheet Window



The SQL Worksheet window consists of the following: Command List, Object Explorer, Editor Pane, Information Pane, and Status Bar.

Command List

Each time a command is successfully executed, the command is saved in the [Command List](#) drop-down field at the top of the window.

Object Explorer

Provides information on database objects. It is possible to drag drop object text and name to the Editor Pane.

Editor Pane

The top right pane is the Editor pane which is used as a work area to create, modify, and execute SQL statements or Transact SQL commands.

Information Pane

The bottom pane has 2 tabs: Message Log and Resultn.

Message Log

Displays information, alert and error messages from each execute. This includes the date and time, Adaptive Server messages, and whether the execution was successful or not. This information is accumulative.

To clear the Message Log information

Right-click and select **Clear All**.

Result

Displays the result set after executing the command in the Editor pane. There can be multiple Result tabs, one for each result set from the code that was executed.

Status Bar



Progress Bar

Displays execution time while code is executing.

Num, Insert, Caps, Scroll

Specifies the state of the key functions for the Number Lock, Insert Mode, Caps Lock, and Scroll Lock.

Modified

Specifies that a modification was made to the text in the Editor pane.

n:n

Specifies the line and column position of the cursor.

nnn bytes

Specifies the number of bytes of text in the Editor pane.

SPID

Specifies the SPID for the database session used by the SQL Worksheet. The session is not connect until the first time you execute a command.

Related Topics

[SQL Worksheet Overview](#)

[SQL Worksheet Functions](#)

Execute SQL or Transact SQL

To open the SQL Worksheet window

Click .

You can enter text into the SQL Worksheet by

- Copying from the Database Explorer, the SQL Scanner, or the SQL Optimizer. (Click )
- Typing the text.
- Opening a file. (**File | Open**).
- Opening a file or database object found through the Code Finder. (Right-click and select **Open in SQL Worksheet**)

To execute your SQL statement or Transact SQL code

Click .

To abort the execution of SQL statement or Transact SQL code

Click .

Note: In Adaptive Server when you complete a transaction, it is committed to the database.

Related Topic
[SQL Worksheet Window](#)

Retrieve Description of Tables and Views

You can retrieve the description for a table or view which gives you the column definitions, data type, and length of the column. This function is not available until text is highlighted in the Editor pane.

To retrieve the table description

1. In the Editor pane of the SQL Worksheet window, highlight a table or view name.
2. Click .

Note: The information is retrieved from the database using the `sp_help` procedure. Whatever text you highlight is sent to Adaptive Server as the parameter to `sp_help`. If you highlight an entire SQL statement, Adaptive Server executes `sp_help` without a parameter and executes the SQL statement. The SQL Worksheet then displays the result of both executions.

Related Topic
[SQL Worksheet Window](#)

Modify Data

In the SQL Worksheet, you can modify the data from a table. The Edit button is not enabled until text is highlighted in the Editor pane.

To modify the data in a table

1. Highlight a table from the text in the Editor pane in the SQL Worksheet window.
2. Click .
3. Use the [Edit button bar](#) at the bottom of the pane to modify the data.

Note: If you create a table and enter data in the SQL Worksheet, you must synchronize the data dictionary by selecting **Database | Synchronize Data Dictionary** before you can use the **Edit** command or **Edit** button to modify the data.

Related Topic
[SQL Worksheet Window](#)

New Database Session

The SQL Worksheet uses a different database session to execute the SQL and Transact SQL from the one that was created when you initially logged on. It is important to understand how using a different session affects your work in other modules.

Temporary Tables

Temporary tables created in the SQL Worksheet are not available throughout the rest of the modules. For example, if you create #temp1 in the SQL Worksheet, you are not able to use that temporary table in SQL Optimizer. Also, if you create a temporary table in the User-Defined Temp Table window, you will not be able to use it in the SQL Worksheet.

Changes to the Data Dictionary

Each time you logon, information from the data dictionary is loaded into the memory of the local PC. If you make changes to the data dictionary in the SQL Worksheet, these changes are not reflected in the information that was initially stored on the local PC. For example, if you create a new table in the SQL Worksheet and then optimize a SQL statement that uses the new table, you will get a message saying that the table does not exist. This is because the data dictionary on the local PC was not updated.

To update the data dictionary

Select **Database** | **Synchronize Data Dictionary**.

Related Topic

[SQL Worksheet Window](#)

Command List

The Command List stores each command that is successfully executed in a drop-down field at the top of the SQL Worksheet window. The following functions are available from the right-click menu in the Command List.

Save Commands

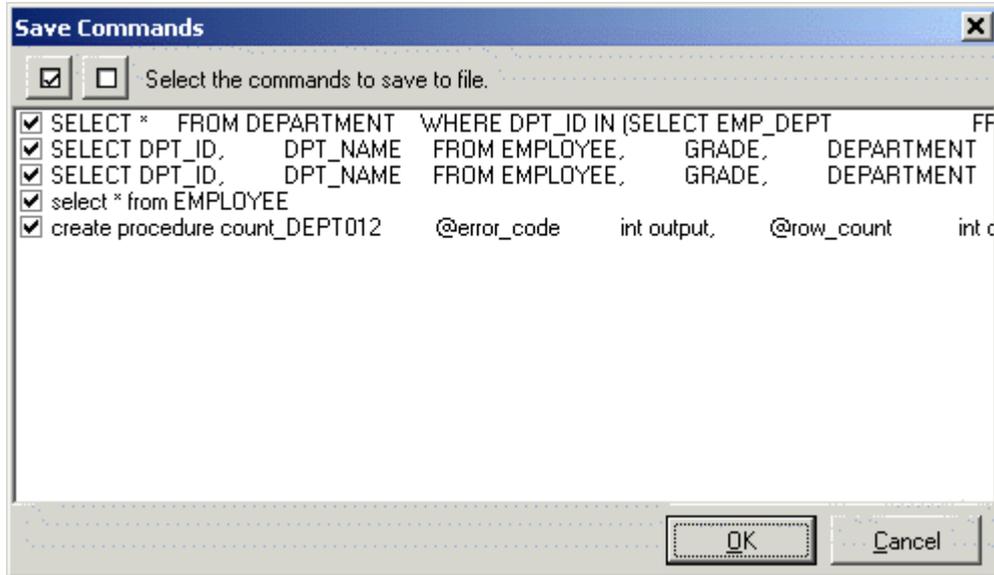
Brings up the Save Commands window so you can select which commands you would like saved to a file.

Load Commands

Brings up the Open Commands window so you can select which commands to load in the Command List from a list that you saved previously.

Clear Commands

Brings up the Clear Commands window so you can select which commands you would like remove from the Command List.



Related Topic

[SQL Worksheet Window](#)

Bookmarks

Bookmarks can be set to quickly return to a place in your code. To set a bookmark, right-click, select Toggle Bookmark or select **Set | Toggle Bookmark** and then pick a bookmark number between 0 and 9. The bookmark number displays in a field in the gutter in place of the line number.

To move the cursor to a bookmark

1. Right-click and select **Goto Bookmark**.
2. Select your bookmark number.

To move to the next or previous bookmark

Select **Set | Previous/Next Bookmark**.

Related Topic

[SQL Worksheet Window](#)

SQL Worksheet Functions

Below is a list of available functions within the Editor pane of the SQL Worksheet.

Button or Menu	Function
Run Menu 	Execute/Abort Execute
Run Menu 	Edit
Run Menu 	SP_HELP
	Syntax Highlight
	Pairing Brackets
	Argument Lookup
	Member Lookup
Edit Menu 	Indent
Edit Menu 	Outdent
Edit Menu 	Comment
Edit Menu 	Uncomment
Edit Menu 	Uppercase
Edit Menu 	Lowercase
Set & Right-click Menu	Toggle Bookmark

Set & Right-click Menu	Goto Bookmark
Set Menu	Previous Bookmark
Set Menu	Next Bookmark
Edit Menu	Send to SQL Optimizer
	
Edit Menu	Send to Index Advisor
	

All the settings for the Editor functions can be changed within the Preferences window under the **Editor** tab. Below is the list of functions available in the Message Log in the SQL Worksheet window.

Menu	Function
Right-click Menu	Save
Right-click Menu	Clear All

Related Topics

[SQL Worksheet Window](#)

[SQL Worksheet Overview](#)

SQL Formatter Overview

The SQL Formatter transforms a SQL statement into a more readable format by automatically indenting and aligning the text, keeping a consistent SQL layout throughout the source code, and therefore making the SQL statement easier to read and understand. The SQL Formatter not only formats the SQL statement but also checks its syntax, and highlights parameters, forces, and comments.

Related Topics

[SQL Formatter Window](#)

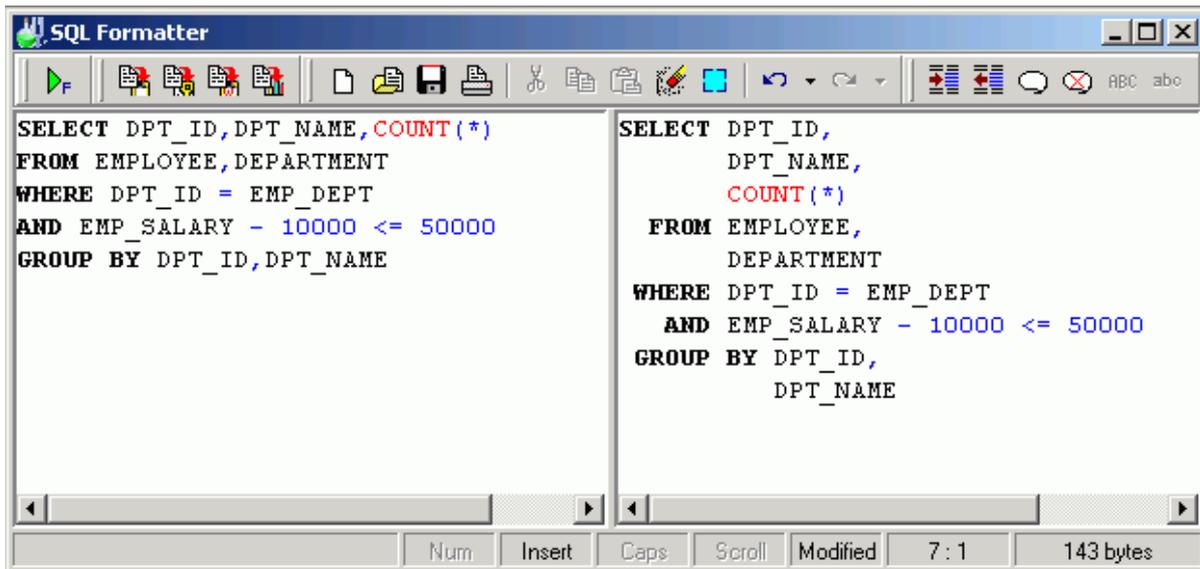
[Format a SQL statement](#)

[Parameters within SQL Formatter](#)

[SQL Formatter Functions](#)

SQL Formatter Window

[View SQL Formatter Window](#)



The SQL Formatter window is divided into two sections:

Left Pane (Original SQL)

Provides an area for entering the SQL statement to be formatted.

Right Pane (Formatted SQL)

Displays the formatted SQL statement after the Format function is executed. Parameters, forces and comments are highlighted for ease of recognition. The color of the text is dependent on the syntax highlight definition set under the Editor tab in the Preferences.

Related Topics

[SQL Formatter Overview](#)

[Format a SQL statement](#)

[Parameters within SQL Formatter](#)

[SQL Formatter Functions](#)

Format a SQL statement

The SQL Formatter checks the syntax of the SQL statement. The SQL statement is formatted only if the syntax is correct. The right pane [Formatted SQL] of the SQL Formatter window shows the formatted SQL statement. If the SQL statement has an invalid object, that object is highlighted in maroon (default color).

If you open the SQL Formatter with the SQL Optimizer as the active window, the SQL statement from the SQL Optimizer window is automatically copied to the SQL Formatter window. Otherwise, enter a SQL statement into the left pane of the SQL Formatter window.

To format a SQL statement

1. Click .
2. Enter the SQL text.
3. Click  to format the SQL statement.

To help in the construction of SQL statements

Use these functions:

- [Auto Correction](#)
- [Member and Argument Lookup](#)
- [Indent](#)
- [Outdent](#)
- [Uppercase](#)
- [Lowercase](#)
- [Comment](#)
- [Uncomment](#)

To terminate the format process

Click .

Note: The SQL Format supports only a single SELECT, SELECT INTO, DELETE, UPDATE, and INSERT SQL statement.

Related Topics

[SQL Formatter Overview](#)

[SQL Formatter Window](#)

[Parameters within SQL Formatter](#)

[SQL Formatter Functions](#)

Parameters within SQL Formatter

SQL statements can have parameters either prefixed with a "@" sign or without the "@". If parameters are recognized, they are highlighted in red (default color) after executing the Format function.

Note: If table names are in maroon (default color) it indicates that the table and corresponding columns are not recognized. Check that:

- The login user has sufficient privilege to access the table.
- You are connected to the correct database or set user.

- Make sure you have the most up-to-date database information in memory. If not, execute the **Synchronize Data Dictionary** function from the **Database** menu.

Related Topics

[SQL Formatter Overview](#)

[SQL Formatter Window](#)

[Format a SQL statement](#)

[SQL Formatter Functions](#)

SQL Formatter Functions

Below is a list of available functions available in the SQL Formatter window.

Button or Menu	Function
SQL Menu 	Format/Abort Format
Edit Menu 	Send to SQL Optimizer
Edit Menu 	Send to Index Advisor
Edit Menu 	Copy to SQL Worksheet
File Menu	Save SQL to SQL Repository
	Syntax Highlight
	Auto Correction
	Bracket Pairing
	Argument Lookup
	Member Lookup
Edit Menu 	Indent
Edit Menu 	Outdent

Edit Menu 	Comment
Edit Menu 	Uncomment
Edit Menu 	Uppercase
Edit Menu 	Lowercase

Related Topics

[SQL Formatter Overview](#)

[SQL Formatter Window](#)

[Format a SQL statement](#)

[Parameters within SQL Formatter](#)

[SQL Formatter Functions](#)

Database Explorer Overview

The Database Explorer facilitates the viewing of database objects within the database server. It saves you from having to use specific SQL commands to access information from the data dictionary, making it an easy job to obtain the most up-to-date database information, simply by point and click.

Related Topics

[Database Explorer Window](#)

[Database Explorer Privileges](#)

[Open Database Explorer](#)

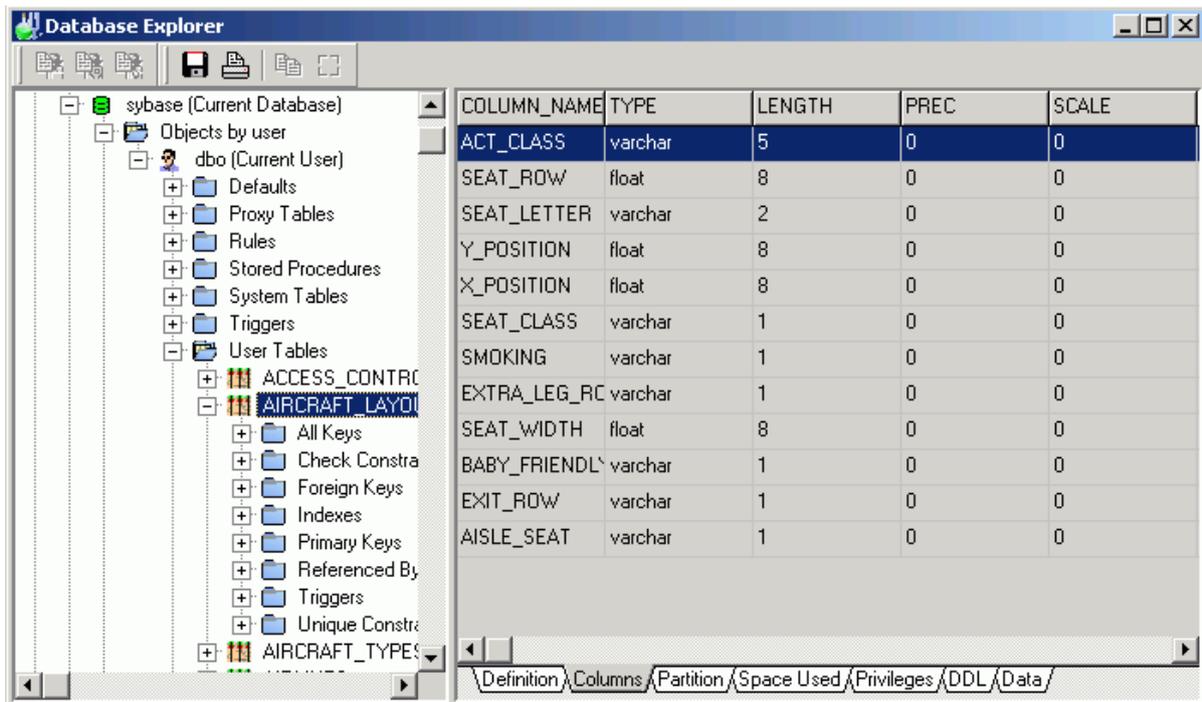
[Modify Data](#)

[Copy column names to another window](#)

[Database Explorer Functions](#)

Database Explorer Window

[View Database Explorer Window](#)



The Database Explorer window is divided into two sections:

Left Pane (Database Objects)

Displays a tree diagram of the database objects.

Right Pane (Object Information)

Displays relevant information of the selected database object.

Related Topics

[Database Explorer Overview](#)

[Database Explorer Privileges](#)

[Open Database Explorer](#)

[Modify Data](#)

[Copy column names to another window](#)

[Database Explorer Functions](#)

Database Explorer Privileges

Your user privileges within the database server are reflected in Database Explorer. If you do not have access to a database, the database icon will be dimmed and no sub-items will be available.

When you expand the first component of the tree you will notice that the database icon is either gray or green, to indicate which databases you have access to.

Access to the system catalog table, syscomments, is needed to view SQL text for procedures, triggers, views, default and rules objects. If you do not have access to the syscomments system table, a message "SQL Text unavailable" will be presented in the Text tab.

Note: For Adaptive Server, you are able to access the syscomments system catalog table by turning the "select on syscomments.text" parameter on, use the command:

```
sp_configure "select on syscomments.text", 1
```

Related Topics

[Database Explorer Overview](#)

[Database Explorer Window](#)

[Open Database Explorer](#)

[Modify Data](#)

[Copy column names to another window](#)

[Database Explorer Functions](#)

Open Database Explorer

To open the Database Explorer window

Click .

The tree diagram displays database objects in the form of a tree layout. The behavior of this module is very similar to Windows-based applications. To expand or collapse the associated list of sub-items, click each branch name. To display the details of a database object, click the object name and the corresponding information will be displayed in the right pane of the window.

The Object Information displays information about the selected database object. By clicking a tab located at the bottom of the right pane, related details about the database object are displayed.

Related Topics

[Database Explorer Overview](#)

[Database Explorer Window](#)

[Database Explorer Privileges](#)

[Modify Data](#)

[Copy column names to another window](#)

[Database Explorer Functions](#)

Modify Data

To modify table or view data

1. Select the table or view you want to modify from the left pane of the Database Explorer window.
2. Select the Data tab from the bottom of the right pane
3. Use the [button bar](#) at the bottom of the pane to select an editing option.

Note: Not all Views are updateable; a general rule is that any UPDATE, DELETE, or INSERT statement on a join view can be modified if only one underlying base table exists.

Related Topics

[Database Explorer Overview](#)

[Database Explorer Window](#)

[Database Explorer Privileges](#)

[Open Database Explorer](#)

[Copy column names to another window](#)

[Database Explorer Functions](#)

Copy column names to another window

To facilitate the construction of a SQL statement, copy the selected column names to the SQL Optimizer or SQL Formatter windows.

To copy column names

1. Select the table or view name displayed on the tree diagram in the left pane [Database Objects] of the Database Explorer window.
2. Make sure the Column tab is active and select the column names by clicking the left-hand mouse button with the Shift key for multiple selections.
3. Right-click and select **Copy columns to SQL Optimizer** to copy the column names to the SQL Optimizer window or select **Copy columns to SQL Formatter** to copy the column names to the SQL Formatter window.

A new line and a comma separate each column name after the copy.

Related Topics

[Database Explorer Overview](#)

[Database Explorer Window](#)

[Database Explorer Privileges](#)

[Open Database Explorer](#)

[Modify Data](#)

Database Explorer Functions

Below is a list of functions available in the **Database Explorer** window.

Button or Menu	Function
Edit Menu 	Send to SQL Optimizer
Edit Menu 	Send to Index Advisor
Edit Menu 	Copy to SQL Worksheet
Right-click Menu	Copy columns to SQL Optimizer
Right-click Menu	Copy columns to SQL Formatter
Data Tab	Modify Data

Related Topics

- [Database Explorer Overview](#)
- [Database Explorer Window](#)
- [Database Explorer Privileges](#)
- [Open Database Explorer](#)

Code Finder Overview

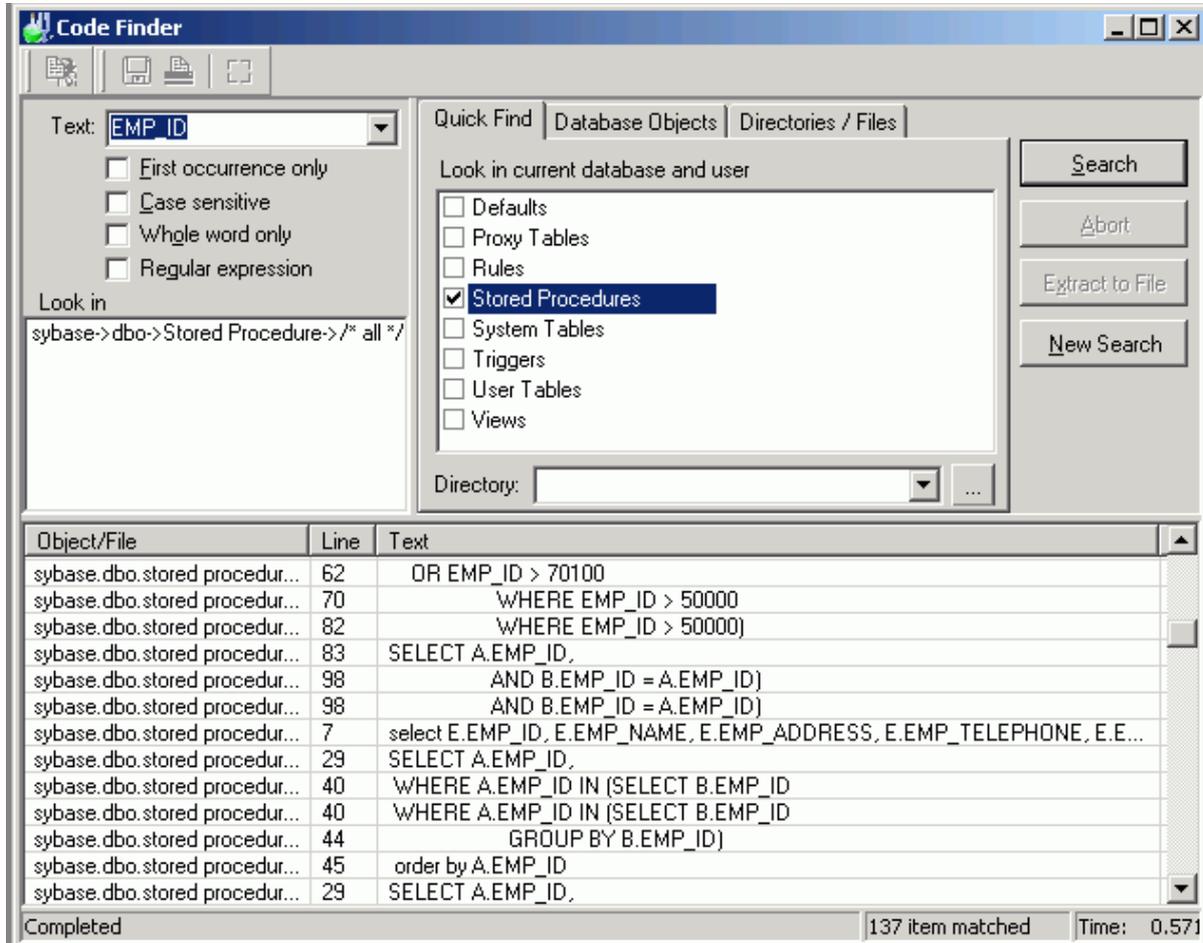
The Code Finder searches database objects and files for a specific text string. Each occurrence of the specified text is listed and the database objects or files where it is found.

Related Topics

- [Code Finder Window](#)
- [Search Database Objects and Files](#)
- [Regular Expression](#)
- [Open Database Object or File in SQL Worksheet](#)
- [Extract Object DDL to File](#)
- [Save Search Results](#)
- [Code Finder Functions](#)

Code Finder Window

The Code Finder window is divided into three sections: search criteria, where to search, and the search result.



Search Criteria

The search criteria displays on the top left pane. Review the following for additional information:

Item	Description
Text	Specify text.
First occurrence only	Specify whether to search for only the first occurrence of the text in the database object or file. If this option is not selected, all occurrences of the text displays.
Case sensitive	Specify whether to search for the text in exact case that you enter it.

Whole word only	Specify whether to search for words only
Regular Expression	Specify whether to recognize regular expressions in the search string.

Where to Search

Select where you want to search. Review the following for additional information:

Item	Description
Look in	Displays the list of selected objects and files to search.
Quick Find	Allows the selection of the database objects belonging to the current database and user (Tables, Views, Procedures, Triggers, Rules, Defaults) and/or the selection of all files belonging to the selected directory.
Database Objects	Allows the browsing and selection of database objects from different databases and users.
Directories/Files	Allows the browsing and selection of directories or files. .

Search Result

The lower pane displays the list of items that matches the search criteria. Review the following for additional information:

Item	Description
Name	Database object or file name.
Line	Line number of matched specified text.
Text	Line of text showing the matched specified text.

Related Topics

[Code Finder Window](#)

[Search Database Objects and Files](#)

[Open Database Object or File in SQL Worksheet](#)

[Extract Object DDL to File](#)

[Save Search Results](#)

[Code Finder Functions](#)

Search Database Objects and Files

To display the Code Finder window

Click .

Enter a search text or use the down arrow next to the input field to select from a list of previously entered search text. The **First occurrence only**, **Case sensitive**, **Whole word only** and **Regular expression** checkboxes lets you refine the search.

For database objects

Select the **Quick Find** tab to select objects belonging to the current database and user. Or, select the **Database Objects** tab. Expand the object and select user, object type or object name and click **Add**. The selected object displays in the **Look in** pane on the top left indicating that the selected objects will be searched. No duplicate objects can be selected.

For directories and files

Select the **Directories/Files** tab. Select the directory and enter a file mask or click the down arrow to select from a list of previously entered file marks. If blank, *.* is used indicating that all files will be searched. To search in the sub-directory, select **Include sub-directories** checkbox. Click **Add** to select the directory and files. No duplicate files can be selected.

To start searching

Click **Search**.

To stop the search

Click **Abort**.

Related Topics

[Code Finder Overview](#)

[Code Finder Window](#)

[Open Database Object or File in SQL Worksheet](#)

[Extract Object DDL to File](#)

[Save Search Results](#)

[Code Finder Functions](#)

Regular Expression

Regular expressions are characters that customize a text search string. They formulate a more complex search. You must select the **Regular expression** checkbox to use these advanced search features.

Type of Search	Search Characters	Location of character in the search string	Explanation
Text at the beginning of line	^ (circumflex) Example:^select	Beginning	A circumflex at the beginning of the text search string finds the text only if it is the first non white space text on a line
Text at the end of line	\$ (dollar sign) Example:error\$	End	A dollar sign at the end of the expression matches the end of a line.
Single character wildcard	. (period) Example:f.r matches "for" and "far"	Any place	A period represents any character.
Class of character	: (colon) Example:SQL:a finds SQLSTATE or SQLTABLE SQL:d Finds SQL1 or SQL3	Any place	A colon matches a class of characters described by the character following the colon. :a matches any alphabetic character :d matches a digit :n matches an alphanumeric character ": matches a space, tab, or other control character or punctuation mark (ASCII 0x01 - 0x40)
Search for Regular Expression characters	\ (backslash) Example: \^ searches for ^ \. searches for \.	Any place	A backslash before a wildcard character searches for actual character and does not use the character as a wildcard.
Search for multiple characters in one position	[...] (characters within square brackets) Example:r[au]n finds run or ran	Any place	Characters enclosed in square brackets matches any one character that displays in the brackets, but no others. Nesting of brackets is not supported.
Search for any character but the specified characters in one character position	[^] (circumflex within square brackets) Example: r[^oa]n finds run, r n, rin, etc but not ron or ran.	Any place	A circumflex at the start of the string in square brackets means NOT. The expression matches any character except the characters in the string and the carriage return (ASCII 0x0D) or line feed (ASCII 0x0A).
Search for a	[-] (hyphen within the square brackets)	Any place	A hyphen within the square brackets signifies a range of characters.

range of
characters in
one character
position

Example: SQL[1-3]
finds SQL1, SQL2 and
SQL3

Note: the range must be in a progressive
order, i.e. "[a-x]" matches any character
from a through x while "[x-a]" does not
match anything.

Related Topics

[Code Finder Overview](#)

[Code Finder Window](#)

[Search Database Objects and Files](#)

[Open Database Object or File in SQL Worksheet](#)

[Extract Object DDL to File](#)

[Save Search Results](#)

[Code Finder Functions](#)

Open Database Object or File in SQL Worksheet

You can open the matched database objects and files in the SQL Worksheet after performing the Search function in the Code Finder window.

To open the database object or file in the SQL Worksheet

Right-click the object or file and select **Open in SQL Worksheet**.

The database object's DDL or the file opens on the SQL Worksheet window and the cursor displays at the beginning of the matching text.

Related Topics

[Code Finder Overview](#)

[Code Finder Window](#)

[Search Database Objects and Files](#)

[Regular Expression](#)

[Extract Object DDL to File](#)

[Save Search Results](#)

[Code Finder Functions](#)

Extract Object DDL to File

In the Code Finder window, you can extract the DDL to a file for the database objects that contain the search string.

To extract the DDL from an object

1. Perform a search.
2. Select the object from the lower pane.
3. Click **Extract to file**.
4. Enter a name for the DDL file.

Related Topics

[Code Finder Overview](#)

[Code Finder Window](#)

[Search Database Objects and Files](#)

[Regular Expression](#)

[Open Database Object or File in SQL Worksheet](#)

[Extract Object DDL to File](#)

[Save Search Results](#)

[Code Finder Functions](#)

Save Search Results

You can save the search results from the Code Finder in a file.

To save the search results

1. Right-click and select **Save**.
2. Select to format the file as **Text**, **HTML**, or **Excel Convertible (Spreadsheet)**.

Related Topics

[Code Finder Overview](#)

[Code Finder Window](#)

[Search Database Objects and Files](#)

[Regular Expression](#)

[Open Database Object or File in SQL Worksheet](#)

[Extract Object DDL to File](#)

[Code Finder Functions](#)

Code Finder Functions

Below is a list of available functions within the Code Finder window.

Menu	Function
	Search
	Abort
	Extract to File
	New Search
Right-click Menu	Open in SQL Worksheet
Right-click Menu	Save

Related Topics

[Search Database Objects and Files](#)

[Extract Object DDL to File](#)

[Open Database Object or File in SQL Worksheet](#)

[Save Search Results](#)

[Code Finder Overview](#)

[Code Finder Window](#)

[Regular Expression](#)

Object Extractor Overview

The Object Extractor allows the retrieval of the DDL text for creating the database objects. You can also retrieve the DDL for the object dependencies ordered by the lowest dependencies first.

Related Topics

[Object Extractor Window](#)

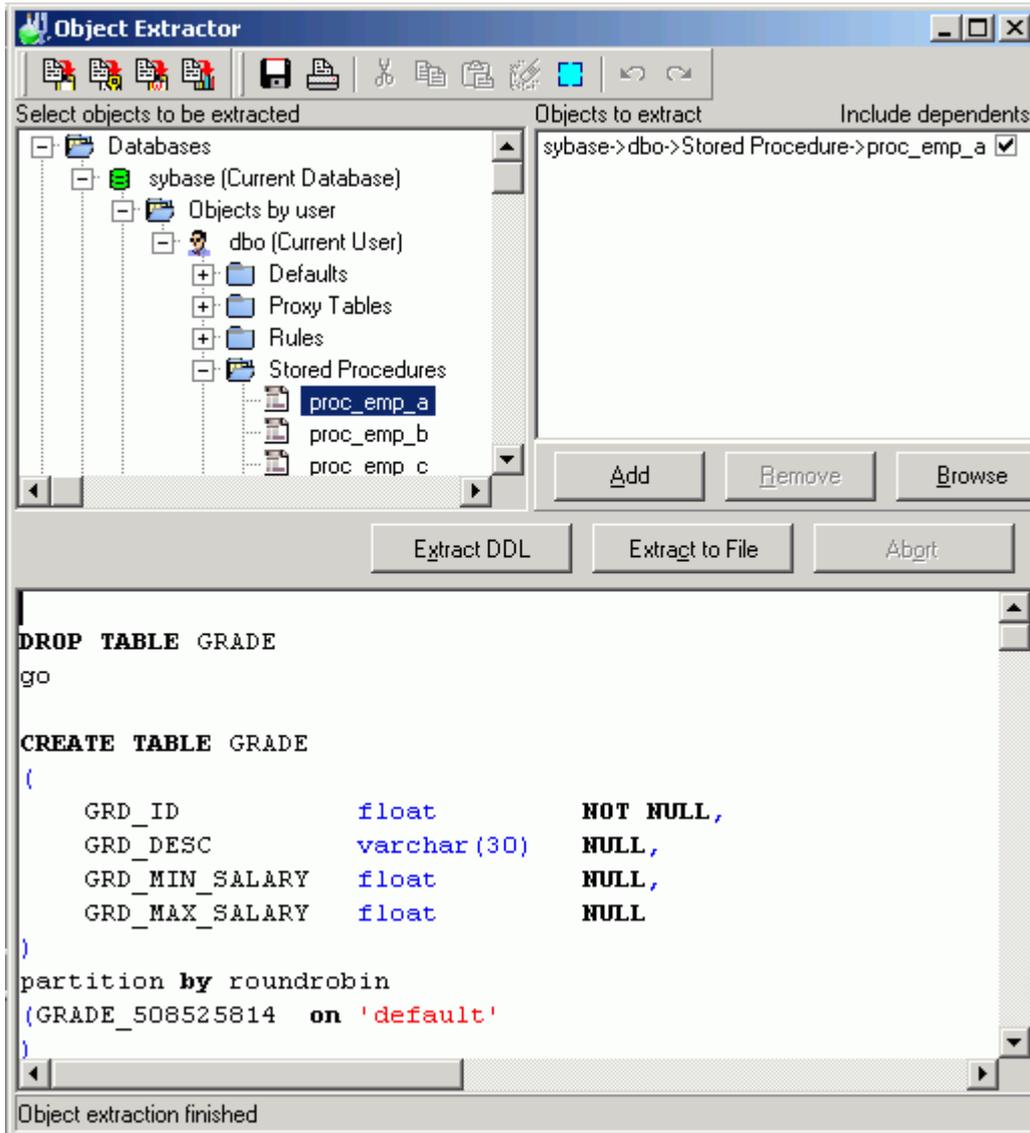
[Extract Object DDL and Dependencies](#)

[Extraction Criteria](#)

[Object Extractor Functions](#)

Object Extractor Window

[View Object Extractor Window](#)



The Object Extractor window has the following panes and buttons:

Select objects to be extracted [left pane]

Displays a tree view of all database objects available for extraction.

Objects to extract [right pane]

Displays the list of selected objects to be extracted, duplicate objects cannot be added. A checkbox is available to include extraction of the object's dependents.

DDL [bottom pane]

Displays the DDL script when **Extract DDL** is selected.

Buttons

Button	Function
Extract DDL	Extract the DDL of the selected object(s) to the bottom pane.
Extract to file	Extract the DDL of the selected object(s) directly to a file.
Abort	Abort the extraction process.

Related Topics

[Object Extractor Overview](#)

[Extract Object DDL and Dependencies](#)

[Extraction Criteria](#)

[Object Extractor Functions](#)

Extract Object DDL and Dependencies

To open the Object Extractor window

Click .

From the left pane [Select objects to be extracted], select the objects to extract. You can select the user name (all objects belonging to the selected user), object type (all objects of the selected type belonging to the selected object) and a specific object name.

To select the objects for retrieving the DDL, use one of the following

- Select the tree node from the left pane and click **Add**.
- Click **Browse** and use the Add Database Objects window to select the database objects.
- Double-click the database object or tree node.
- Drag and drop the object over to the right pane [Objects to extract].

By default, all selected objects are extracted with dependents. Deselect the corresponding checkbox if you do not want to include dependents in the [Objects to extract] pane.

To display the DDL script on the bottom pane

Click **Extract DDL**.

To save the DDL script directly to file

Click **Extract to file**.

Before the object DDL is extracted, the Extraction Criteria window displays. This window allows you to select how the object DDL displays. If you disable this window, the DDL script is generated according to the selected parameters on the DDL tab in the Preferences window.

To abort the extraction of the DDL

Click **Abort**.

Related Topics

[Object Extractor Overview](#)

[Object Extractor Window](#)

[Extraction Criteria](#)

[Object Extractor Functions](#)

Extraction Criteria

The Extraction Criteria window provides options on how the object DDL displays. These options include:

Place a DROP command before the object

Specify whether to include the DROP command before the CREATE object command.

Qualify object with database and user name

Specify whether to place the database and user name in front of every object in the DDL.

Include indexes for table

Specify whether to include the CREATE INDEX command when the DDL for a table is extracted.

Place comments in script

Specify whether to place a comment before and after the DDL script for each object. The comment includes information on the database, user, object name and date and time of generation.

Show on next extraction

Specify whether to show the Extraction Criteria window next time DDL is extracted. If this window is not shown then the extraction option will be according to last selection.

Related Topics

- [Object Extractor Overview](#)
- [Object Extractor Window](#)
- [Extract Object DDL and Dependencies](#)
- [Object Extractor Functions](#)

Object Extractor Functions

Below is a list of available functions within the Object Extractor window.

Button or Menu	Function
	Extract DDL
	Extract to File
	Abort
Edit Menu 	Send to SQL Optimizer
Edit Menu 	Send to Index Advisor
Edit Menu 	Copy to SQL Worksheet

Related Topics

- [Object Extractor Overview](#)
- [Object Extractor Window](#)
- [Extraction Criteria](#)

SQL Repository Overview

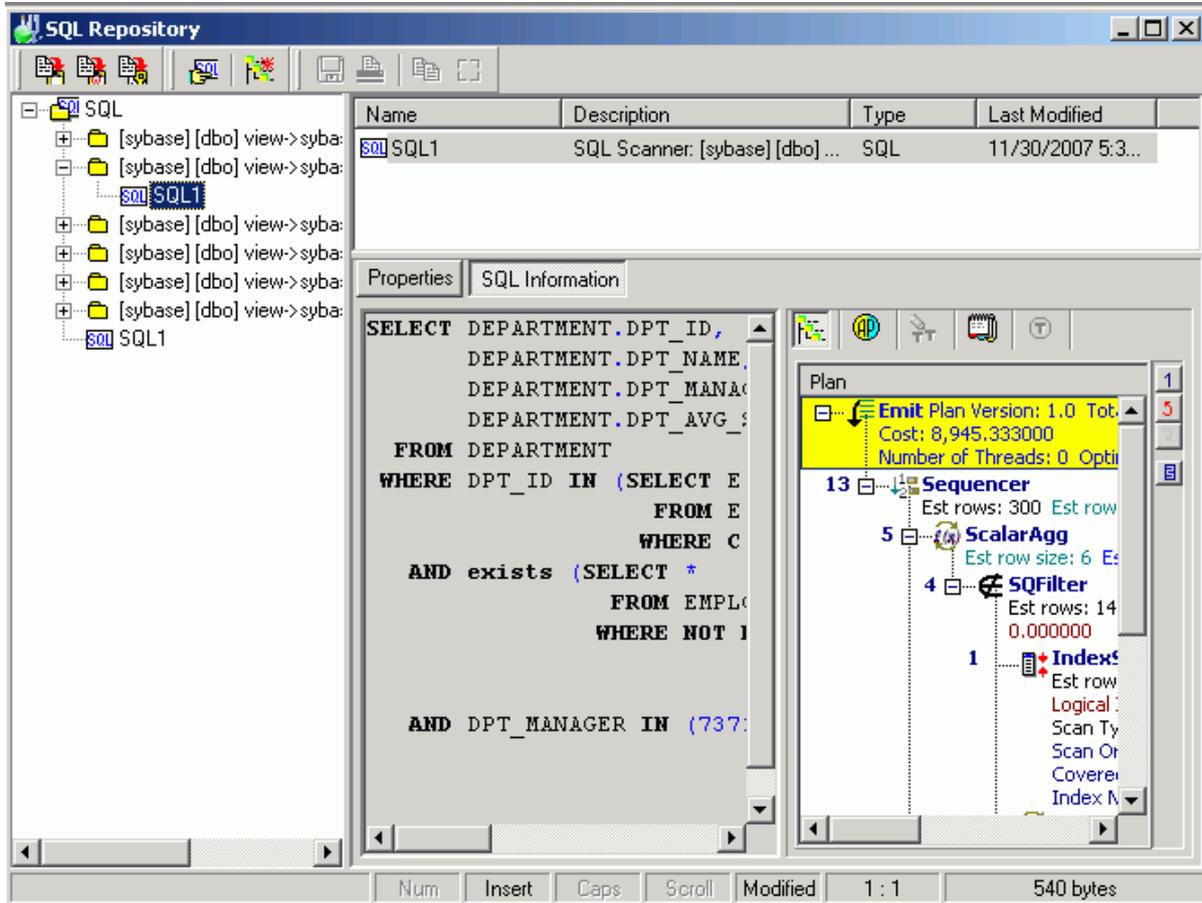
The SQL Repository stores the SQL statements that are used in the analysis of database performance. These may be SQL statements that you have identified as critical to the performance of your database application.

Related Topics

- [SQL Repository Window](#)
- [SQL Repository Functions](#)
- [Add SQL to SQL Repository](#)

SQL Repository Window

View SQL Repository Window



The SQL Repository window displays the information about the SQL statements that are saved in the SQL Repository. It is divided into three panes.

SQL Listing [Left pane]

Displays a tree diagram of the SQL statements and the folders they are stored in.

SQL Listing Details [Top right pane]

Displays detailed information about the folder, the SQL statement, or the folder with its contents depending on which item in the tree in the left pane is selected.

SQL Information Details [Bottom right pane]

The detailed information about the SQL statement displays in the button pages. The buttons for displaying specific information are found at the top of this pane.

Properties button

Displays general information, the SQL name, where the SQL statement was copied from, the logon connection, and database and other settings.

SQL Information button

Displays the SQL text, the query plan, the abstract plan, trace on information, SQL classification, and DDL for any temporary tables used in the SQL statement.

Save the Abstract Plan

The abstract plan for a SQL statement can be saved from this pane. In order to save the abstract plan, it must have been retrieved when the SQL statement was created in the SQL Repository. Before the SQL statement is created, you must enable the dumping of the Abstract Plan in the [Optimization](#) or the [Scanner](#) Preferences. If you are connected to Adaptive Server 15, use the Abstract Plan settings in the Optimization for ASE 15 tab. Otherwise, use the Abstract Plan settings in the Optimization for Pre-ASE 15 tab.

To save the abstract plan

1. Select the SQL statement in the left pane.
2. Click in the right pane.
3. Click .
4. In the **Save to group** field, select the abstract plan group.
5. Click **Save**.

Related Topics

[SQL Repository Overview](#)

[SQL Repository Function](#)

Add SQL to SQL Repository

SQL statements are stored in the SQL Repository for use in an Index Impact Analysis, an Index Usage Analysis, a Migration Analysis, or a Configuration Analysis. From within the SQL Repository window, you can add a single SQL statement, some or all SQL statements found by the SQL Scanner, and SQL statements that reference specific tables or views. SQL statements can also be added to the SQL Repository from [other modules](#).

To add a single SQL statement

1. Click .
2. If no SQL exists in the SQL Repository, then the Add SQL wizard displays automatically. Otherwise, click **Add SQL** .
3. The Add SQL wizard has three pages to select the options for saving the SQL statement to the SQL Repository.
 - [General](#)
 - [SQL Information](#)
 - [Database Settings](#)
4. After making the selections from the three tabs, click **OK** to save the SQL to the SQL Repository.

The SQL syntax is checked and the query plan retrieved before adding a new node to the SQL tree view with the SQL name. Each SQL statement added to the SQL Repository contains a query plan, SQL classification type (Simple, Complex or Problematic) and the current connection information (login name, server name, database and user). The query plan stored with the SQL statement is important as it indicates the current performance of the SQL.

To add SQL from the SQL Scanner

1. From the SQL Repository window, select **SQL | Add SQL from SQL Scanner**.
2. On the General page of the wizard, select the location to save the SQL statements.
3. On the SQL Scanner page, select the Groups or Jobs that contain the SQL you want to add.
4. Click **Finish**.

To add all SQL statements that references a specific table or view

1. From the SQL Repository window, select **SQL | Add SQL by Database Object**.
2. In the Database object pane, select the tables or views that you want the SQL to reference.
3. Click **Extract**. The Save SQL to SQL Repository wizard displays.
4. On the General page, select the location to save the SQL statements.
5. Click **Finish**.

Related Topics

[SQL Repository Overview](#)

[SQL Repository Functions](#)

Add SQL Wizard: General Page

View Add SQL Wizard - General Page

The General page of the Add SQL wizard is used to name the SQL statement and to select the folder where you want the SQL statement saved.

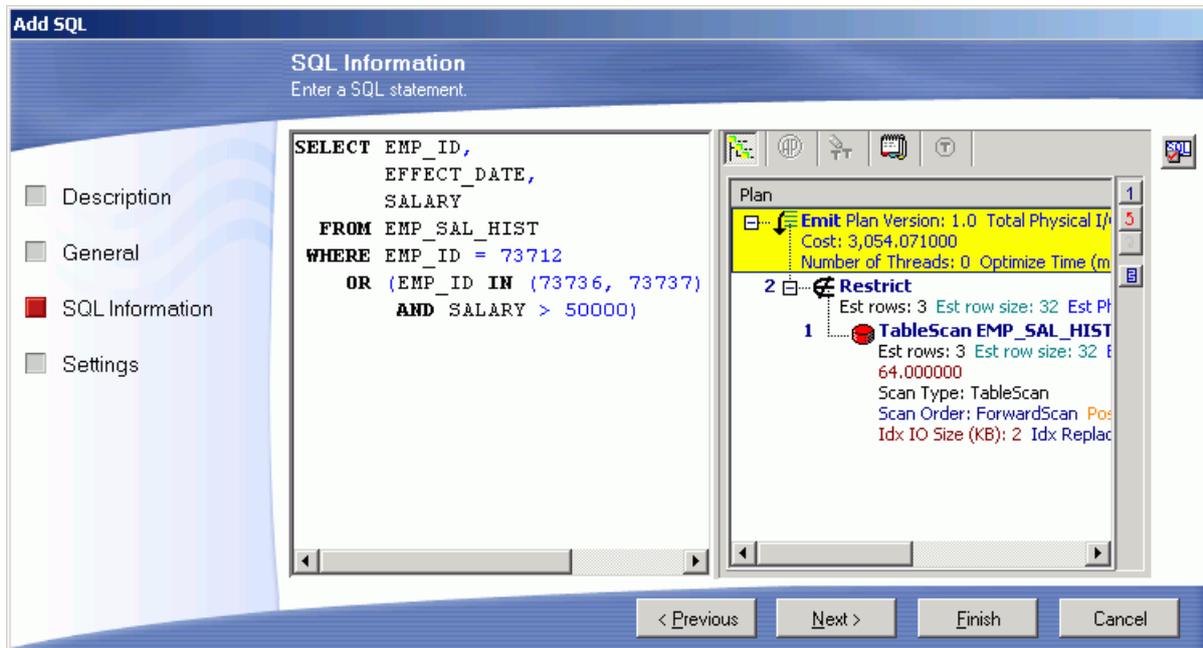
Item	Description
Name	Enter the name for the SQL to be saved in the SQL Repository.
Description	Enter the description for the SQL.
Last modified	Displays the last modified date and time.
Login name	Displays the connected login name.
Server name	Displays the connected server name.
Database	Displays the connected database.
User	Displays the connected user name.
Save SQL to location	Displays the tree location in which the SQL will be saved. In the bottom pane, select the folder where you want to save the SQL. To create a new folder, use  .

Related Topic

[Add SQL to SQL Repository](#)

Add SQL Wizard: SQL Information Page

View Add SQL Wizard - SQL Information Page



The SQL Information page of the Add SQL wizard is where to enter the SQL text and view various information about the SQL statement, such as the query and abstract plans.

Left pane

Enter the SQL text.

Right Pane

At the top of the right pane are buttons that display the query plan, the abstract plan, Trace On information, the SQL classification and connection information, and the DDL for creating any temporary tables used by the SQL statement.

Check SQL button

Checks the SQL syntax using the current database connection and retrieves the query plan, SQL type classification and other information relating to the SQL statement. To get the query plan most appropriate for your SQL statement, make sure that you have made the corresponding selections from the Settings tab before retrieving the checking the SQL.

Related Topic

[Add SQL to SQL Repository](#)

Add SQL Wizard: Settings Page

The Settings page of the Add SQL wizard is used to set various parameters that affect the retrieval of the query plan for the SQL statement.

SQL Settings section

SQL for Cursor

Adaptive Server uses a difference query plan for a SQL statement that is embedded in a cursor declaration from the query plan when the SQL statement is not embedded in a cursor. This needs to be taken into account when retrieving the query plan or run time and also when generating SQL alternatives.

Therefore, if the original SQL statement comes from or will be embedded in a cursor declaration then you need to select the **SQL for Cursor** checkbox. This enables cursor simulation when retrieving the query plan.

Use Default Plan

This option uses the BINARY data type when executing the Show Plan function for all variables in the SQL statement. This is useful when you want to quickly investigate the query plan of the original SQL statement without having to select the data type for each variable.

Abstract Plan section

Dump abstract plan

Specify whether to retrieve the abstract plan for the SQL statement whenever the query plan is retrieved. The abstract plan is not saved on the database until you deliberately save it. The default group names in Adaptive Server are: ap_stdout and ap_stdin. These groups are usually used by the Database Administrator to enable server-wide abstract plan capturing and retrieving.

Group name

Specify the abstract plan group name where the abstract plan for this SQL statement is saved.

Abstract Plan Manager button

Opens the Abstract Plan Manager window to view, create, and modify abstract plan group.

Database Settings section

Set Ansinull On

Specify the option with regards to comparison of NULL values.

Set Quoted_identifier On

Specify whether to allows the use of delimited identifier (" ") for table names.

Set Statistics Simulate On SAP ASE 15 or later)

Specify whether to load simulated statistics into the database. Simulated statistics can be generated using `optdiag` command and can be used to optimize SQL statements using the simulated statistics rather than the actual statistics.

dbcc traceon (3604, 302, 310) (sa_role privilege only)

Specify whether to retrieve the trace on information which displays the reasons why the Adaptive Server optimizer chooses to resolve the SQL statement in a particular way. This option is applicable only if you have `sa_role` privileges.

Related Topic

[Add SQL to SQL Repository](#)

Refresh SQL Query Plan

If you feel the query plan stored with the SQL statement in the SQL Repository does not represent the current indication of performance due to database changes, you can retrieve the current query plan from the database.

To refresh the query plan in the SQL Repository

1. Select the SQL.
2. Select **SQL | Refresh Plan** or right-click the tree view and select **Refresh Plan** to open the Refresh Plan window.
3. The original query plan displays on the left pane while the newly retrieved query plan is displayed on the right pane for comparison. You can view and analyze the newly retrieved query plan, abstract plan, and SQL classification before saving the new query plan with the SQL statement.
4. Click **Save** to replace the original query plan with the newly retrieved one.

Related Topic

[SQL Repository Window](#)

Modify SQL

You can only modify the SQL name and description of the SQL statement stored in the SQL Repository.

To modify SQL name or description

1. From the SQL Repository window, select the SQL you want to modify from the left pane.
2. Right-click and select **Modify**.

3. Modify the name and/or description. Click **OK** to save changes to the SQL Repository. The Last modified field will automatically be updated once the information is saved.

Note: SQL text cannot be modified. You must delete the current SQL and recreating a new one to change the SQL text.

Related Topic

[SQL Repository Window](#)

Rename SQL

You can rename a SQL statement stored in the SQL Repository window in two ways:

To rename SQL

1. Right-click the SQL and select **Rename**. A highlighted field is placed around the SQL name in the tree to indicate that you can edit the name.
2. Change the name and press **Enter** or click any where outside the field.

Note: You can also right-click the SQL and select **Modify** to open the Modify SQL window. Modify the name. Click **OK** to save changes to the SQL Repository.

Related Topic

[SQL Repository Window](#)

Delete SQL

When you delete a SQL statement from the SQL Repository, it also deletes all references to this SQL statement in the Analyzer modules.

To delete a SQL statement stored in the SQL Repository

1. Right-click the SQL and select **Delete** or highlight the SQL statement and press **DELETE**.
2. If the SQL statement is in an Index Impact Analysis, an Index Usage Analysis, a Migration Analysis, or a Configuration Analysis, you will be prompted before it is deleted.

Related Topic

[SQL Repository Window](#)

Create Benchmark Factory Import File

All the SQL statements can be saved in a text file from the SQL Scanner, the SQL Repository, the SQL Inspector, and the SQL Optimizer windows. These SQL statements can then be imported into Benchmark Factory 4.6 or later.

To create a file to import into Benchmark Factory

1. Right-click and select **Create Benchmark Factory Import File**.
2. Select the specific SQL statements that you want to save.
3. Enter the filename and select the file location.

SQL Repository Functions

Below is a list of available functions within the SQL Repository window.

Button or Menu	Function
SQL & Right-click Menu 	Add SQL
SQL Menu	Add SQL from SQL Scanner
SQL & Right-click Menu	Add SQL by Database Object
SQL & Right-click Menu 	Refresh Plan
SQL & Right-click Menu	Add Folder
Right-click Menu	Modify
Right-click Menu	Delete
Right-click Menu	Rename
SQL & Right-click Menu	Save Abstract Plan
Right-click Menu	Create Benchmark Factory Import File

Related Topic

[SQL Repository Overview](#)

Index Impact Analyzer Overview

The Index Impact Analyzer analyzes the performance impact of new indexes on SQL statements before the indexes are permanently created on your database. New indexes may improve the performance of one SQL but downgrade the performance of several others. Index Impact Analyzer evaluates the performance effect that given indexes might have on SQL statements, thereby removing the uncertainty associated with index creation.

The Index Impact Analyzer analyzes the performance impact of index creation on SQL statements retrieved from the SQL Repository or the SQL Scanner. To analyze the impact the creation of new indexes might have on SQL performance, the proposed indexes are created on the database. Then new query plans are retrieved showing the impact the new indexes have on the query plans of individual SQL statements. The new indexes are dropped after the retrieval of the query plans.

After the analysis is complete, the Index Impact Analyzer provides metrics of the overall performance changes and allows you to identify which SQL statements experienced query plan changes with the proposed indexes.

Related Topics

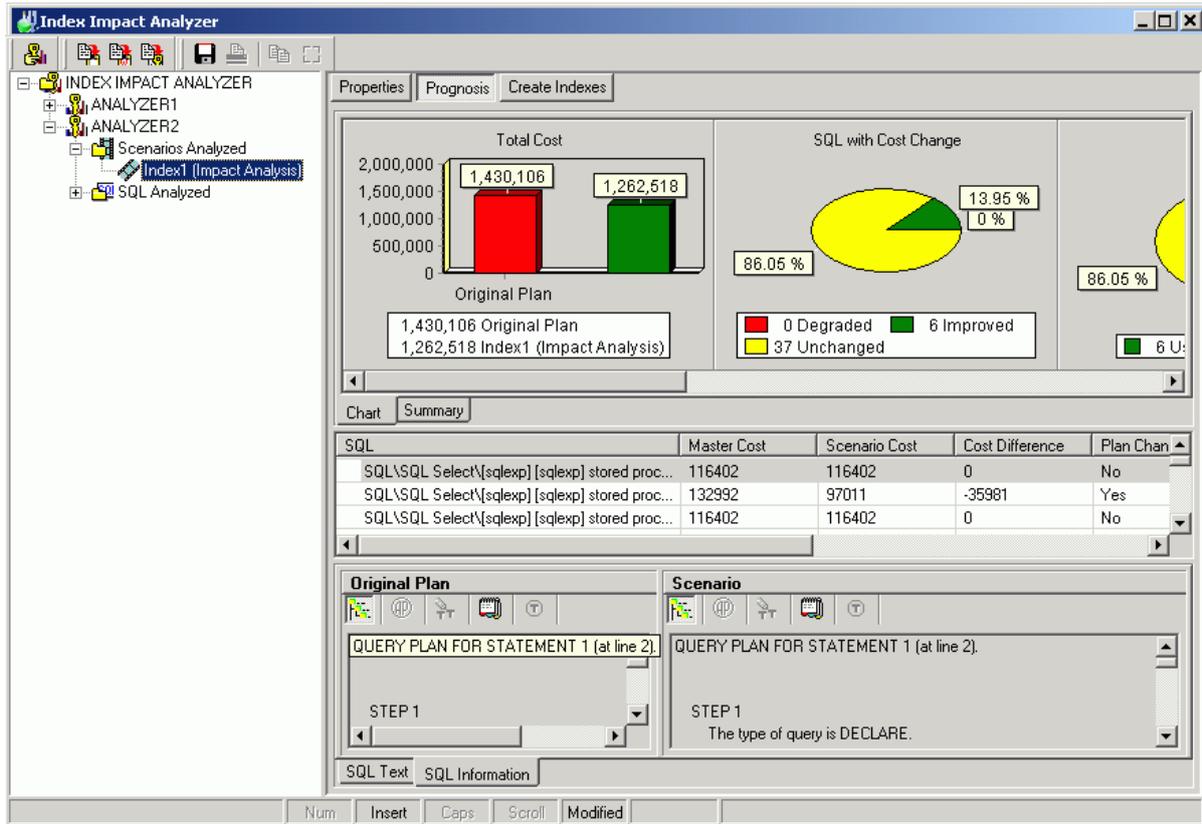
[Create an Index Impact Analyzer](#)

[Index Impact Analyzer Window](#)

[Index Impact Analyzer Functions](#)

Index Impact Analyzer Window

[View Index Impact Analyzer Window](#)



The Index Impact Analyzer window displays the information about the SQL statements that are saved in the Index Impact Analyzer. The display in the right section of the window depends on what is selected in the left pane.

Left pane - Analyzer List

Always displays a tree diagram of the Analyzers and the folders they are stored in.

Right Pane

The right pane displays a variety of different information depending upon what is selected from the tree diagram in the left pane.

You can display information for the following:

- [Analyzers](#)
- [Scenarios Analyzed Folder](#)
- [Scenario](#)
- [SQL Analyzed Folder](#)
- [SQL Statements](#)

Related Topics

[Create an Index Impact Analyzer](#)

Right Pane for Analyzers

In the Index Impact Analyzer window, when an Analyzer is selected in the left pane, the following information displays in the right pane.

Properties Button

Displays general information about the analysis, information on the connection used to retrieve the query plans and the SQL statements used in the analysis.

Related Topic

[Index Impact Analyzer Window](#)

Right Pane for Scenarios Analyzed Folder

In the Index Impact Analyzer window, when the **Scenarios Analyzed** Folder is selected in the left pane, the following information displays in the right pane.

Summary button

Provides a summary of the Estimated I/O Cost and SQL classification for the SQL statements before and after the proposed indexes were created.

Chart button

Charts the total Estimated I/O Cost of the SQL statements with and without the proposed indexes.

Related Topic

[Index Impact Analyzer Window](#)

Right Pane for Scenarios

In the Index Impact Analyzer window, when the Scenario folder is selected in the left pane, the following information displays in the right pane.

Properties button

Displays general information about the analysis and the index name.

Prognosis button

Displays 3 panes in the right section of the window.

Charts and Summary tabs [Top right pane]

The **Chart** tab displays five charts of the comparison information for the query plans from before the analysis and during the analysis.

The **Summary** tab displays summarized statistics for the Scenario,

SQL Classification and Plan Change Comparisons [Middle right pane]

A grid displays the information about each query plan from before the analysis and during the analysis in a comparison format.

SQL Text and SQL Information tabs [Bottom right pane]

The SQL statement that is highlight in the middle right pane is the one that displays in this pane.

The **SQL Text** tab displays the text of the SQL statement.

The **SQL Information** tab displays two [SQL Information panes](#), one for the SQL information before the analysis and one for the SQL Information with the parameter changes.

Create/Drop Indexes button

Displays the script for creating and dropping the index sets. Click **Execute**  located at the right of the **Index Impact Analyzer** window to create the new index or drop the index.

Related Topic

[Index Impact Analyzer Window](#)

Right Pane for SQL Analyzed Folder

In the Index Impact Analyzer window, when the SQL Analyzed folder is selected in the left pane, the following information displays in the right pane.

Cost Summary button

Displays all the SQL statements and shows the Estimated I/O Cost with and without the proposed indexes.

Cost Classification button

Displays all the SQL statements and shows the [SQL Classification](#) (Simple, Complex, and Problematic) with and without the proposed indexes.

Related Topic

[Index Impact Analyzer Window](#)

Right Pane for SQL Statements

In the Index Impact Analyzer window, when a SQL statement is selected in the left pane, the following information displays in the right pane.

Properties button

Displays general information about the analysis, information on the connection used to retrieve the query plans and the [database settings](#) at the time of the analysis.

SQL Information button

Displays the SQL statement selected, as well as, two [SQL Information panes](#) that show the query plans for the statement before and after the index Scenario is in place. In the query plan, the operations that are different between the two plans are highlighted in green.

Saving the Abstract Plan

The abstract plan for a SQL statement can be saved from this pane. In order to save the abstract plan, it must be retrieved when the SQL was selected to be analyzed. This is done from the Abstract Plan page of the New Analysis wizard.

1. Select the SQL statement in the left pane.
2. Click SQL Information section of the right pane. This enables the **Save Abstract Plan** button.
3. Click **Save Abstract Plan** .
4. In the **Save to group** field, select the abstract plan group.
5. Click **Save**.

Related Topic

[Index Impact Analyzer Window](#)

Create an Index Impact Analyzer

To create an Index Impact Analyzer

1. Click  to open the Index Impact Analyzer window.
2. If no analysis exists in the Index Impact Analyzer then the New Analysis wizard displays automatically.
Otherwise, click  to open the New Analysis wizard.

3. The New Analysis wizard the following pages to select the options for creating an analysis.
 - [Analyzer](#)
 - [Selected SQL](#)
 - [Index](#)
 - [Abstract Plan](#)
4. Name the Analyzer, select the SQL statements for analysis, and specify the DDL for the proposed indexes in the New Analysis window.

Note: The indexes are physically created and then dropped after the query plans are obtained. This may affect other SQL statements executing on the database during this period.

Related Topics

[Index Impact Analyzer Overview](#)

[Index Impact Analyzer Window](#)

New Analysis Wizard: Analyzer Page

View New Analysis Wizard--Analyzer Page

New Analysis

Analyzer
Select settings for analyzer.

Check the effects of adding new indexes by

- Creating a new Analyzer and select SQL to analyze
- Continuing an existing Analyzer using the Analyzer's selected SQL

Do not drop the indexes after the analysis is finished

Name: ANALYZER1

Description:

Last modified: 11/21/2008 3:10:52 PM

Save location: INDEX IMPACT ANALYZER

Select save location

INDEX IMPACT ANALYZER

< Previous Next > Finish Cancel

In the New Analysis wizard, the Analyzer page is used to name the analysis and to select the folder where the Analyzer information will be stored.

Creating a new Analyzer and select SQL to be analyzed

Select this option to create a new analysis using a different set of SQL statements.

Continuing an existing Analyzer using the Analyzer's selected SQL

Select this option to use the same set of SQL statements that is stored in an existing Analyzer. This option adds another Scenario under an existing Analyzer and does not retrieve a new set of query plans for the SQL statements. It uses the query plans that were originally saved with the SQL statements.

Do not drop the indexes after the analysis is finished

Specific to leaves the new indexes that you create under the Index tab after the analysis is finished. Otherwise, these changes are dropped at the end of the analysis processing. After the analysis is complete, the **Create/Drop Indexes** button displays at the top of the Index Impact Analyzer window which displays the Create Indexes, Drop Indexes, and the Drop All Indexes panes. These panes contain the scripts to create or drop the indexes.

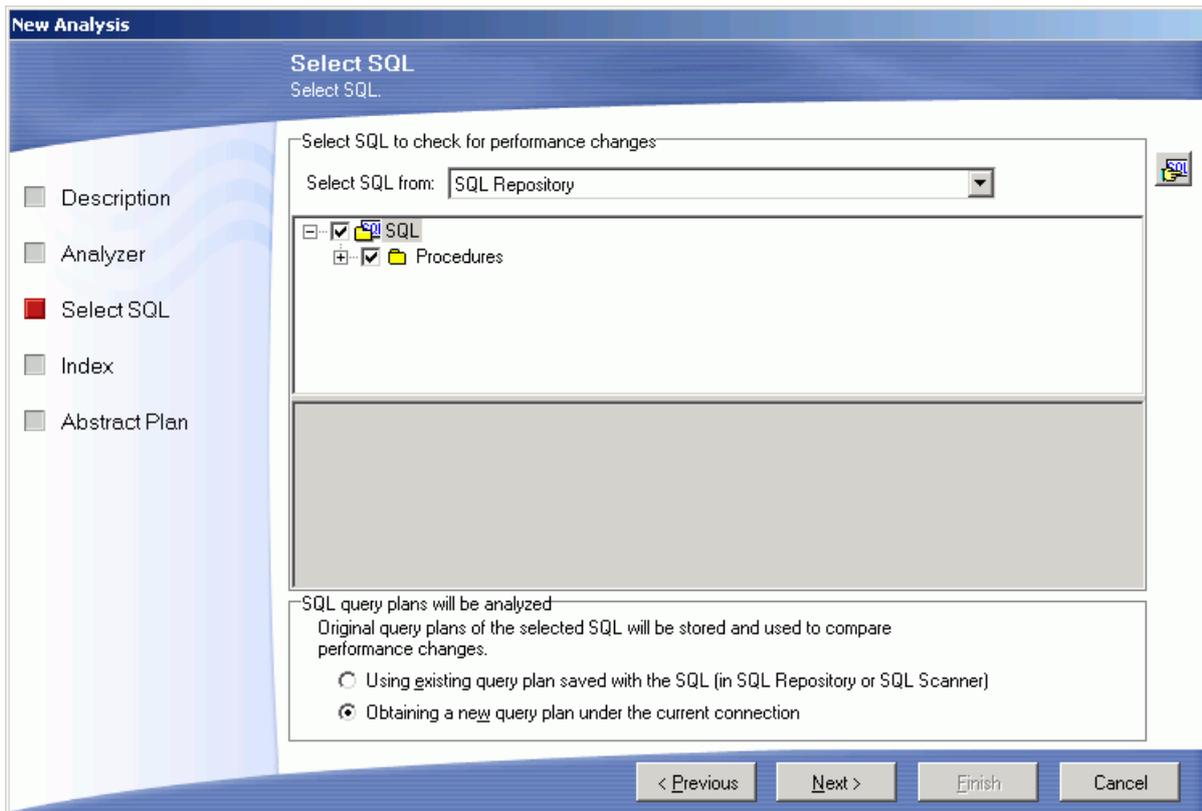
Item	Description
Name	Enter the name for the analysis.
Description	Enter the description for the analysis.
Last modified	Displays the last modified date and time.
Save location	Displays the folder in the tree where the Analyzer is saved.
Folder Tree	In the bottom pane, select the folder where you want to save the Analyzer.
	To create a new folder, click Add Folder .

Related Topic

[Create an Index Impact Analyzer](#)

New Analysis Wizard: Select SQL Page

View New Analysis Wizard - Select SQL Page



In the New Analysis wizard, the Select SQL page is used to select the SQL statements whose query plans you want to analyze in order to review the impact on their performance that the creation of new indexes would have.

Select SQL to check for performance changes:

Select SQL from

The SQL statements used in an analysis can be taken from the SQL Repository or the SQL Scanner. If you select SQL statements from the SQL Scanner, these statements are added to the SQL Repository.

Select the SQL statements for the analysis. If you select SQL statements from the SQL Scanner, the bottom portion of this section displays the SQL Repository folder so that you can select the folder where they will be stored.

SQL query plans to be analyzed:

In order to analyze the impact the proposed indexes might have on the performance of the SQL statements, the Index Impact Analyzer compares the query before the indexes are created to the plans after the indexes are created. The query plans from before the analysis can be obtained two ways:

Using existing query plan saved with the SQL

This option uses the query plan that was saved with the SQL statement when it was saved in the SQL Repository or SQL Scanner.

Obtaining a new query plan under the current connection

This option retrieves the query plan with the current logon and the current database settings. This current query plan is compared to the query plan that is retrieved after the new indexes are created.

Related Topic

[Create an Index Impact Analyzer](#)

New Analysis Wizard: Index Page

View New Analysis Wizard - Index Page

New Analysis

Index
Select the index.

Query plans will be grouped under a Scenario

Name: Scenario1
Description:

Enter and select indexes to analyze

Please note that the indexes are physically created on your database in order to retrieve the query plan. After the query plan is retrieved, these indexes are dropped.

Please specify how you would like the indexes created

Segment: default Consumers: 1

Index Name	Database	User
------------	----------	------

< Previous Next > Finish Cancel

In the New Analysis wizard, the Index page is used to specific the index(es) that are used for this analysis.

Query plans will be grouped under a Scenario

An individual Index Impact Analyzer consists of a group of stored SQL statements with the query plans called the "Baseline Plan." The actual Impact Analysis displays in a "Scenario." A Scenario shows the comparison information for the stored SQL statements identifying the impact on the query plan that the creation of new index will have.

Name

Enter a name for the Scenario.

Description

Enter a description.

Enter and select indexes to be analyzed

Use **Add Index** to add the indexes for this Scenario.

Segment

Select the segment where you would like the index created from the drop-down list.

Consumer (Default 1 Range 1-20)

Specify the number of consumer processes that should perform the sort operation for creating the index. The actual number of consumer processes used to sort the index may be smaller than the specified number, if fewer worker processes are available when Adaptive Server executes the sort.

To add an index

Click **Add Index**  to bring up the [Add Index](#) wizard.

To remove an index

Select the index and click **Remove Index** .

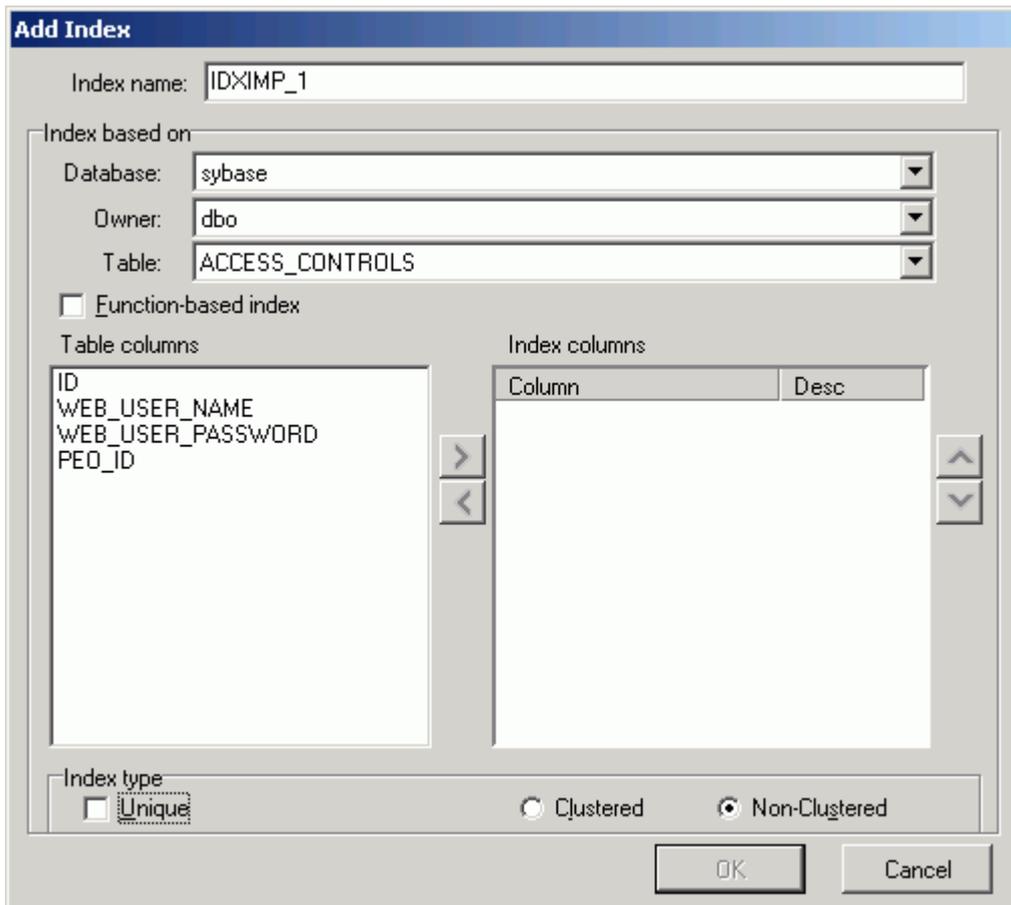
After you have created the first Scenario, you can [add more Scenarios](#).

Related Topic

[Create an Index Impact Analyzer](#)

Add Index Wizard

[View Add Index Wizard](#)



The Add Index wizard is used to create the DDL for creating of the indexes for the Index Impact Analysis. This wizard is available on the [Index page](#) in the Index Impact Analyzer's New Analysis wizard.

To open this Add Index wizard

Click .

Item	Description
Index name	Enter a name for the index.
Database	Select the database where the index is to be created from the drop-down list.
Owner	Select the owner of the index from the drop-down list.
Table	Select the table where the index is to be created from the drop-down list.
Table columns	List of all the columns from the selected table.
Index columns	Move the columns to be indexed from the Table columns list by either double-clicking the column or highlighting the column and clicking  .

Unique	Specify to create a unique index.
Clustered/Nonclustered	Select to create either a clustered or a nonclustered index

Related Topic

[Create an Index Impact Analyzer](#)

New Analysis Wizard: Abstract Plan Page

View New Analysis Wizard: Abstract Plan Page

In the New Analysis wizard, the Abstract Plan page is used to include the abstract plan with SQL statements.

Baseline Plan

Dump abstract plan

Specify whether to retrieve the abstract plan for the SQL statement for the before new indexes are created. The abstract plan displays in the [right pane for SQL statements](#). This retrieves a new abstract plan even if one was saved with the original SQL. It is important to use this option if you originally saved the SQL statement and abstract plans in version 12.5.0.3 or earlier of Adaptive Server and are now using Adaptive Server 15.0 or later since the format of the abstract plan has changed.

Note: This option is disabled if you have selected **Using existing query plan saved with the SQL** on the [Select SQL](#) page.

Group name

Specify the abstract plan group name where the abstract plans are saved. The default group names in Adaptive Server are: `ap_stdout` and `ap_stdin`. These groups are usually used by the Database Administrator to enable server-wide abstract plan capturing and retrieving.

`ap_stdout` is used by default to capture an abstract plan.

`ap_stdin` is used by default to retrieve the abstract plan associated with a SQL statement during the execution of the SQL statement.

Abstract Plan Manager button

Opens the Abstract Plan Manager window to view, create, and modify abstract plan group.

Scenario

Dump abstract plan

Specify whether to retrieve the abstract plan for the SQL statement after the indexes are created. The abstract plan is displayed in the [right pane for SQL statements](#).

Group name

Specify the abstract plan group name where the abstract plans are saved. The default group names in Adaptive Server are: `ap_stdout` and `ap_stdin`. These groups are usually used by the Database Administrator to enable server-wide abstract plan capturing and retrieving.

`ap_stdout` is used by default to capture an abstract plan.

`ap_stdin` is used by default to retrieve the abstract plan associated with a SQL statement during the execution of the SQL statement.

Abstract Plan Manager button

Opens the Abstract Plan Manager window to view, create, and modify abstract plan group.

Related Topic

[Create an Index Impact Analyzer](#)

Drop Indexes

The Index Impact analysis first creates the indexes on your database. Then it retrieves the query plan for each SQL statement that you have selected for your analyzer. When you set up the individual scenarios, you determine if you would like the indexes dropped or to leave them on your database.

If you do not have the analysis process drop the indexes, you can drop them in any one of the following ways:

- In the tree pane (left pane), right-click the scenario name and select **Go To Drop Indexes**.
- When you close the Index Impact Analyzer window, you are prompted to drop the indexes. Click **Yes**.
- In the tree pane (left pane), click the scenario name and select the **Create/Drop Indexes** button at the top of the right pane. Click  next to the Drop Indexes or Drop All Indexes pane.

You can make the changes permanent, in one of the following ways:

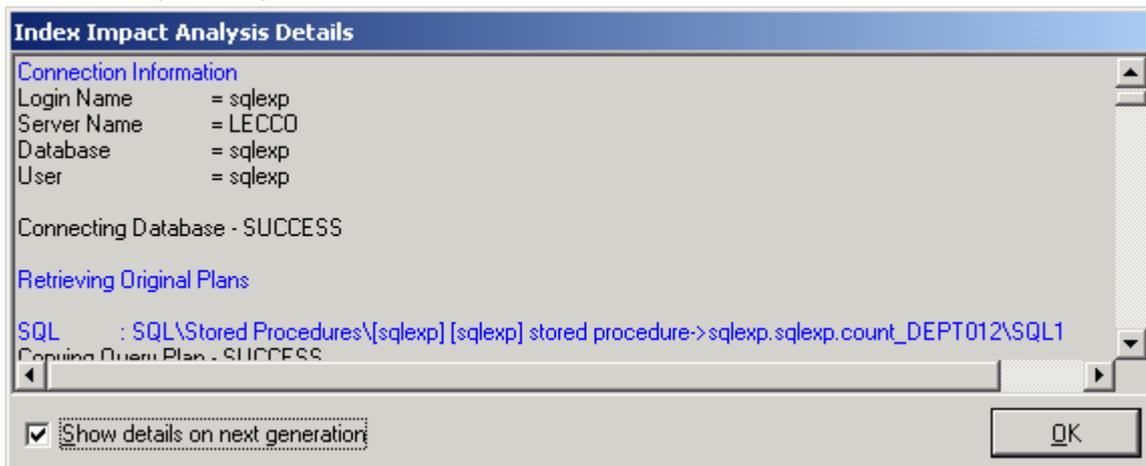
- In the tree pane (left pane), right-click the scenario name and select **Confirm Index Changes**. Click **Yes**.
- When you close the Configuration Analyzer window, you are prompted to rollback the changes. Click **No**.

Related Topic

[Index Impact Analyzer Window](#)

, View Index Impact Analysis Details

View Index Impact Analysis Details Window



The Index Impact Analysis Details window displays information about the creation of the indexes and the retrieval of the query plans. The Index Impact Analysis Details window displays after the Index Impact Analysis process is completed unless the **Show details on next generation** checkbox in the Index Impact Analysis Details window is unchecked.

To review the Index Impact Analysis Details window after completing Index Impact Analysis

1. Select **View | Last Index Impact Analysis Details**.
2. Review this information to see if an error occurred during the retrieval of the query plan. Errors may occur if the selected SQL statements are from one database and user and you have set another database and user in the drop-down boxes at the bottom left of the main window.

Related Topic

[Create an Index Impact Analyzer](#)

Add SQL to an Index Impact Analysis

After creating an Index Impact Analysis, you can add more SQL statements from the SQL Repository to it.

To add more SQL statements to an Analysis

1. From the left pane, select the Analyzer in which you want to add the SQL.
2. Right-click and select **Add SQL**.
3. In the Add SQL window under the **Select SQL to be added** pane, check the SQL statements you want added.
4. Under the **Select Scenario to include SQL** pane, check the Scenarios you want the SQL added to.
5. Click **OK** to add the SQL, retrieve the query plans and re-execute the Scenario.

Related Topic

[Index Impact Analyzer Window](#)

Add a Scenario

After creating an Index Impact Analysis, you can add more Scenarios to it.

To add a Scenario

Method One

1. In the left pane, right-click the Analyzer and select **Add Scenario**.
2. In the Add Scenario wizard, select the [Index](#) page and specify the information for the proposed new indexes. Click **OK**.

Method Two

1. In the left pane, right-click on an Analyzer and select **New Analysis**.
2. In the New Analysis wizard, select the **Continuing an existing Analyzer using the Analyzer's selected SQL** option.
3. From the **Select Analyzer** field, click the **Analyzer** name.
4. Select the [Index](#) page and specify the information for the proposed new indexes. Click **OK**.

Related Topic

[Index Impact Analyzer Window](#)

Delete an Index Impact Analysis

To delete an analysis

1. Select the analysis.
2. Right-click and select **Delete** or press **Delete**.

Related Topic

[Index Impact Analyzer Window](#)

Modify an Index Impact Analyzer

You can only modify the analysis name and description.

To modify an Analysis

1. From the Index Impact Analyzer window, select the Analyzer you want to modify from the left pane.
2. Right-click and select **Modify** to open the Modify Analyzer window.
3. Modify the name and/or description. Click **OK** to save the changes.

Related Topic

[Index Impact Analyzer Window](#)

Regenerate

The Index Impact Analyzer Scenario can be re-executed using the same indexes. You can choose to retrieve the query plans for all SQL statements in the Scenario or select specific SQL to retrieve the query plan again using the same indexes. This function is useful if changes have occurred in the database environment, such as data volume changes, that may affect query plans.

To regenerate an analysis

1. Select the Scenario from the left pane.
2. Right-click and select **Regenerate**.
3. From the Regenerate Scenario window, check the SQL statements whose current query plans you would like retrieved again.

Related Topic

Rename an Index Impact Analyzer

To rename an analysis

1. Select the Analysis you want to modify from the left pane.
2. Right-click and select **Rename**. A highlighted field is placed around the SQL name in the tree to indicate that you can edit the name. Change the name and press **Enter** or click any where outside the field.

Note: You can also right-click and select **Modify** to open the Modify Analyzer window. Modify the name. Click **OK** to save changes to the analysis.

Related Topic

[Index Impact Analyzer Window](#)

Index Impact Analyzer Functions

Below is a list of available functions within the Index Impact Analyzer window.

Button or Menu	Function
Analysis & Right-click Menu 	New Analysis/Abort Analysis
Right-click Menu	Add Folder
Right-click Menu	Add Scenario
Right-click Menu	Add SQL
Right-click Menu	Regenerate
Right-click Menu	Modify
Right-click Menu	Rename
Right-click Menu	Delete
Right-click Menu	Go To Drop Indexes
Right-click Menu	Confirm Index Changes

Related Topic

[Index Impact Analyzer Overview](#)

Index Usage Analyzer Overview

The Index Usage Analyzer identifies unused indexes by analyzing query plans from SQL statements in your database applications. It examines their query plans and reports any indexes in the database that are not used. You can use this module to quickly identify the indexes in databases that are not contributing to the performance of the database applications. These unused indexes can then be deleted to free up space and improve the speed of the database applications and maintenance.

The Index Usage Analyzer identifies:

- Tables that are referenced in the SQL statements
- Indexes in each table that are used in the query plans, and the number of referenced SQL for each index
- Indexes in each table that are not used in the query plans

The selected SQL statements that the Index Usage Analyzer analyzes are from SQL that you have saved in the SQL Repository or SQL statements from the SQL Scanner.

Related Topics

[Index Usage Analyzer Window](#)

[Index Usage Analyzer Functions](#)

Index Usage Analyzer Window

The Index Usage Analyzer window displays the information about the SQL statements that are saved in the Index Usage Analyzer. The display in the right section of the window depends on what is selected in the left pane.

Left pane - Analyzer List

Always displays a tree diagram of the Analyses and the folders they are stored in.

Right Pane

The right pane displays a variety of different information depending upon what is selected from the tree diagram in the left pane.

You can display information for the following:

- [Analyzers](#)
- [Scenarios Analyzed](#)
- [Table Information](#)
- [SQL Analyzed Folder](#)
- [SQL Statements](#)

Related Topics

[Create an Index Usage Analysis](#)

[Index Usage Analyzer Overview](#)

[Index Usage Analyzer Functions](#)

Right Pane for Analyzers

In the Index Usage Analyzer window, when an Analyzer is selected in the left pane, the following information displays in the right pane.

Index Usage Chart [Top right pane]

Displays the total number of indexes in the tables used in all the selected SQL statements and the number of used and unused indexes. Charts the percentage of the indexes that are used by the query plans for the selected SQL statements.

Analysis Information [Bottom right pane]

The detailed information about the Analysis displays in the button pages. The buttons for displaying specific information are found at the top of this pane.

Properties button

Displays general information about the analysis, information on the connection used to retrieve the query plans and the SQL statements used in the analysis.

Index Summary button

Displays the indexes that are used by the selected SQL statements in black text and highlights in red the indexes that are not used. Use the **Right-click menu** to **Save** the list of indexes or **Drop Index**.

Related Topic

[Index Usage Analyzer Window](#)

Right Pane for Tables Analyzed

In the Index Usage Analyzer window, when the **Tables Analyzed** is selected in the left pane, the following information displays in the right pane.

Summary button

Provides a summary of the use of indexes for each table. It gives the total number of SQL that use the table and if the SQL statement accesses the table with a full table scan or uses an index to access the table.

Chart button

Charts the number of SQL statements that use the table.

Related Topic

[Index Usage Analyzer Window](#)

Right Pane for Analyzed Table Information

In the Index Usage Analyzer window, when a table is selected under the Tables Analyzed section is selected in the left pane, the following information displays in the right pane.

Prognosis Button

Item	Description
Top Pane - Summary Information	Displays a summary of the indexes in the table that are used or unused.
Middle Pane - Index Used Information	Displays the indexes used by the SQL statement.
Bottom Left Pane - SQL Text	Displays the text of the SQL statement.
Bottom Right Pane - SQL Information	Displays the query plan, abstract plan, traceon information, SQL Classification, and the temporary table DDL for the selected SQL statement in the SQL Information Pane .

Unused Index Button

Lists the indexes and the key for all indexes in the table that are not used. Use the **Right-click menu** to **Save** the list of indexes or **Drop Index**.

Table Information Button

Item	Description
Definition Tab	Displays when and where the table was created.
Columns Tab	Displays the column information for the table.
Index Tab	Displays the index information for the table.
DDL Tab	Displays the DDL for creating the table and indexes.

Related Topic

[Index Usage Analyzer Window](#)

Right Pane for SQL Analyzed Folder

In the Index Usage Analyzer window, when the SQL Analyzed folder is selected in the left pane, the following information displays in the right pane.

Index Used

Displays all the SQL statements and shows the tables and indexes used.

Related Topic

[Index Usage Analyzer Window](#)

Right Pane for SQL Statements

When a SQL statement is selected in the left pane, the following information displays in the right pane.

Properties button

Displays general information about the analysis, information on the connection used to retrieve the query plans and the database settings at the time of the analysis.

SQL Information button

Displays the query plan and the [SQL Information pane](#).

Index Used button

Displays the table name, the index name, and the index key for the indexes used by the SQL statement. Use the **Right-click menu** to **Save** the list of indexes.

Related Topic

[Index Usage Analyzer Window](#)

Create an Index Usage Analysis

To create an Index Usage Analysis

1. Click  to open the Index Usage Analyzer window.
2. If no analysis exists in the Index Usage Analyzer then the New Analysis wizard displays automatically.
Otherwise, click .
3. The New Analysis wizard has two pages to select the options for creating an analysis.
 - [Analyzer](#)
 - [Selected SQL](#)
4. Make the selections from these two tab page. Click **OK**.

When the analysis is finished, the [Index Usage Analysis Details window](#) displays with information about the retrieval of the query plan for each SQL statement.

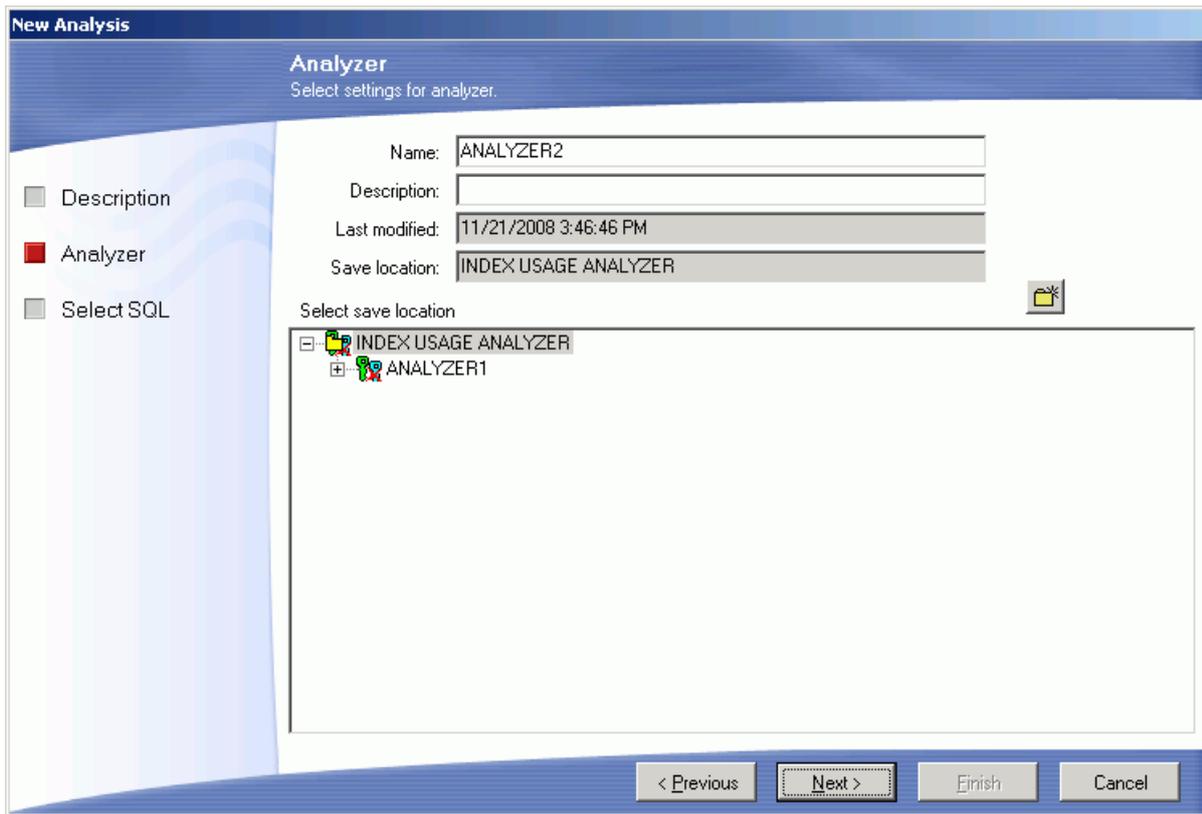
Related Topics

[Index Usage Analyzer Overview](#)

[Index Usage Analyzer Window](#)

New Analysis Wizard: Analyzer Page

[View New Analysis Wizard: Analyzer Page](#)



The Analyzer page is used to name the analysis and to select the folder where the Analyzer information will be stored.

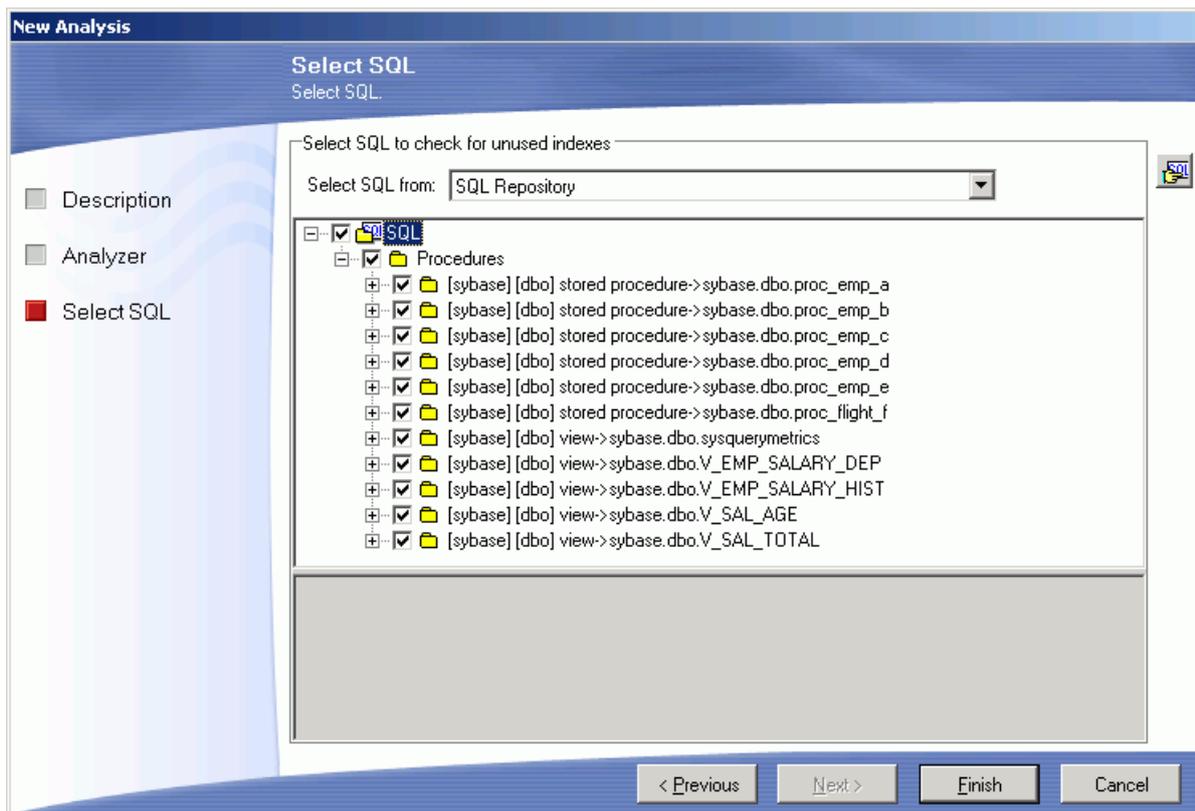
Item	Description
Name	Enter the name for the analysis.
Description	Enter the description for the analysis.
Last modified	Displays the last modified date and time.
Save location	Displays the folder in the tree where the analysis is saved. In the bottom pane, select the folder where you want to save the analysis.
<p>To create a new folder</p> <ul style="list-style-type: none"> Click Add Folder  	

Related Topic

[Create an Index Usage Analysis](#)

New Analysis Wizard: Select SQL Page

View New Analysis Wizard: Select SQL Page



The Select SQL page is used to select the SQL statements whose query plans you want to analyze to see which Indexes are used and which are not used by these SQL statements.

Select SQL to check for unused indexes:

Select SQL from

The SQL statements used in an analysis can be taken from the SQL Repository or the SQL Scanner. If you select SQL statements from the SQL Scanner, these statements are added to the SQL Repository.

Bottom Pane

Select the SQL statements for the analysis.

Related Topic

[Create an Index Usage Analysis](#)

Update an Index Usage Analysis

You can update an analysis for unused indexes by rerunning the analysis with the same SQL statements from the original analysis or you can add or delete SQL statements from the analysis.

To update an analysis

1. Right-click the analysis name in the left pane of the Index Usage Analysis window and select **Update Analysis**. The Update Analysis window has two tabs to select the options for changing an analysis before it is updated. It is not necessary to make any changes on these tabs before rerunning the analysis.

[Analyzer tab](#)

[Selected SQL tab](#)

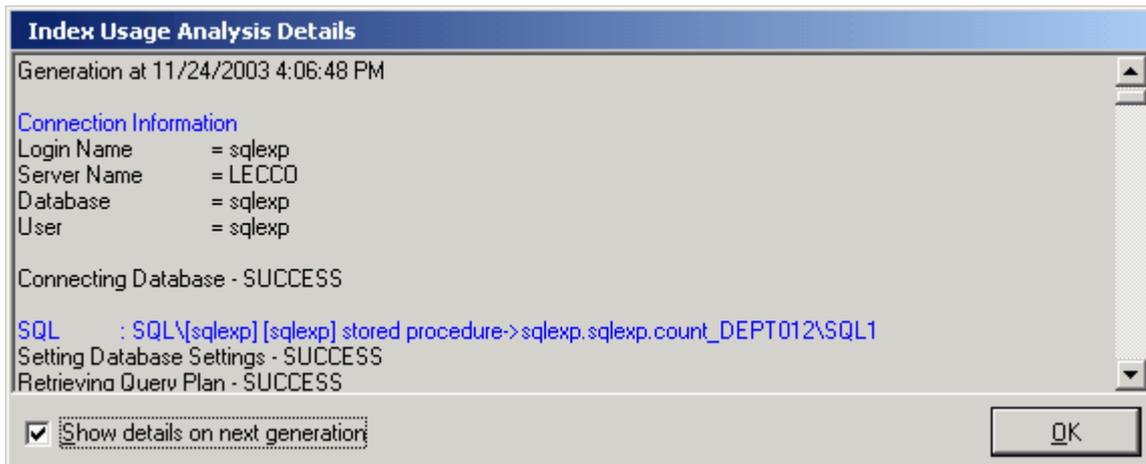
2. On the Analyzer tab, you can change the name and description of the analysis.
3. On the Selected SQL tab, you can add or delete SQL statements to the analysis.
4. After making the selections from the two tabs, click **OK**.
5. When the analysis is finished, the [Index Usage Analysis Details window](#) displays with information the retrieve of the query plan for each SQL statement.

Related Topic

[Create an Index Usage Analysis](#)

View Index Usage Analysis Details

View Index Usage Analysis Details Window



The Index Usage Analysis Details window displays information about the retrieve of the query plan for each SQL statement.

The Index Usage Analysis Details window displays after the batch run process is completed unless the **Show details on next batch run** checkbox in the Index Usage Analysis Details window is unchecked.

To review the Index Usage Analysis Details window after an analysis

1. Select **View | Show Index Usage Analysis Details** when the Index Usage Analysis window is active.
2. Review this information to see if an error occurred during the retrieval of the query plan. Errors may occur if the selected SQL statements are from one database and user and you have set another database and user in the drop-down boxes at the bottom left of the main window.

Related Topic

[Index Usage Analyzer Window](#)

Modify an Index Usage Analysis

You can only modify the analysis name and description.

To modify an Analysis

1. Select the Analyzer you want to modify from the left pane.
2. Right-click and select **Modify**.
3. Modify the name and/or description.

Related Topic

[Index Usage Analyzer Window](#)

Rename an Index Usage Analysis

You can rename an analysis in the Index Usage Analyzer window in two ways.

To rename an Analysis

1. Right-click the Analysis you want to modify from the left pane and select **Rename**. A highlighted field is placed around the SQL name in the tree to indicate that you can edit the name.
2. Change the name and press **Enter** or click any where outside the field.

Note: You can also right-click and select **Modify**. Modify the name.

Related Topic

[Index Usage Analyzer Window](#)

Delete an Index Usage Analysis

To delete an analysis

1. Select the analysis you want to delete in the left pane.
2. Right-click and select **Delete**.

Related Topic

[Index Usage Analyzer Window](#)

Index Usage Analyzer Functions

Below is a list of available functions within the Index Usage Analyzer window.

Menu	Function
Analysis & Right-click Menu	New Analysis/Abort Analysis
Right-click Menu	Update Analysis
Right-click Menu	Add Folder
Right-click Menu	Modify
Right-click Menu	Rename
Right-click Menu	Delete
View Menu	Last Index Usage Analysis Details
Right-click Menu	Drop Index
Right-click Menu	Save

Related Topics

[Index Usage Analyzer Overview](#)

[Index Usage Analyzer Window](#)

Configuration Analyzer Overview

The Configuration Analyzer evaluates the effect on SQL performance when changing sp_configure Adaptive Server parameter settings. It enables you to analyze whether the database performance may improve before you make any Adaptive Server parameter changes permanent.

The Configuration Analyzer compares the query plans from your selected SQL statement to the query plans that are retrieved during the configuration analysis using the `sp_configure` parameter changes. The analysis changes the specified database parameters for your session, retrieves a new query plans, and then sets the parameter values back to the current value.

The selected SQL statements are from SQL that you have saved in the SQL Repository.

Note: Changing the configuration parameters affects other connections to the database, i.e. SQL statements from other users.

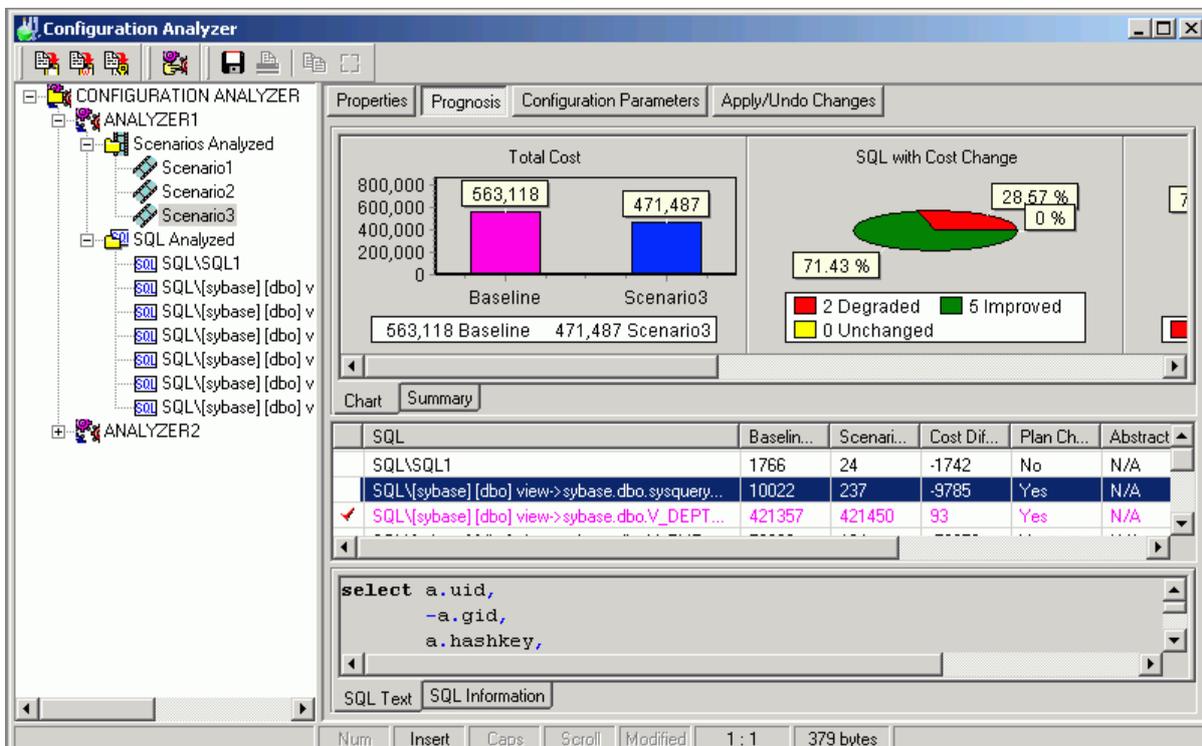
Related Topic

[Configuration Analyzer Window](#)

[Configuration Analyzer Functions](#)

Configuration Analyzer Window

The Configuration Analyzer window displays the information about the SQL statements that are saved in the Configuration Analyzer. The display in the right section of the window depends on what is selected in the left pane.



Left pane - Analyzer List

Provides a tree view of the Analyzers that provide performance analysis information for different configuration Scenarios. For each Analyzer two folders are provided. Review the following for additional information:

Migration Displays performance analysis information for configuration Scenarios

Information

SQL Analyzed Displays the SQL statements selected from the SQL Repository for analysis and the query plans for each of the Scenarios.

Right Pane

The right pane displays a variety of different information depending upon what is selected from the tree diagram in the left pane. Review the following for additional information:

Analyzers	Right Pane for Analyzers
Scenarios Analyzed Folder	Right Pane for Scenarios
Scenario	Right Pane for Scenarios
SQL Analyzed Folder	Right Pane for SQL Analyzed Folder
SQL Statements	Right Pane for SQL Statement

Related Topics

[Create a Configuration Analysis](#)
[Configuration Analyzer Overview](#)
[Configuration Analyzer Functions](#)

Right Pane for Analyzers

In the Configuration Analyzer window, when an Analyzer is selected in the left pane, the following information displays in the right pane.

Properties Button

Displays general information about the analysis, information on the connection used to retrieve the query plans and the SQL statements used in the analysis.

Related Topic

[Configuration Analyzer Window](#)

Right Pane for Scenarios Analyzed

In the Configuration Analyzer window, when Scenarios Analyzed is selected in the left pane, the following information displays in the right pane.

Summary button

Provides a summary of the Estimated I/O Cost and SQL classification for the SQL statements before and after the configuration parameters were changed.

Chart button

Charts the total Estimated I/O Cost of the SQL statements before and after the configuration parameters were changed.

Related Topic

[Configuration Analyzer Window](#)

Right Pane for Scenarios

In the Configuration Analyzer window, when a specific **Scenario** is selected in the left pane, the detailed information about the Scenario displays in the button pages in the right pane. The buttons for displaying specific information are found at the top of this pane.

Properties button

Displays general information about the SQL statement and the database settings at the time of the analysis.

Prognosis button

Displays 3 panes in the right section of the window.

Charts and Summary tabs [Top right pane]

The Chart tab displays five charts of the comparison information for the query plans from before the analysis and during the analysis.

The Summary tab displays summarized statistics for the Scenario,

SQL Classification and Plan Change Comparisons [Middle right pane]

A grid displays the information about each query plan from before the analysis and during the analysis in a comparison format.

SQL Text and SQL Information tabs [Bottom right pane]

The SQL statement that is highlight in the middle right pane is the one that displays in this pane.

The **SQL Text** tab displays the text of the SQL statement.

The **SQL Information** tab displays two [SQL Information panes](#), one for the SQL information before the analysis and one for the SQL Information with the parameter changes.

Configuration Parameters button

Displays the database parameter changes used in this analysis displaying the parameter name, the value during the analysis and the old value for the parameters that were changed.

Apply/Undo Changes button

Displays the scripts for changing the configuration parameter on Adaptive Server. Click **Execute**  located at the right of the Configuration Analyzer window to run the script.

Related Topic

[Configuration Analyzer Window](#)

Right Pane for SQL Analyzed Folder

In the Configuration Analyzer window, when the SQL Analyzed folder is selected in the left pane, the following information displays in the right pane.

Cost Summary button

Displays all the SQL statements and shows the Estimated I/O Cost before and after the configuration parameters were changed.

Cost Classification button

Displays all the SQL statements and shows the [SQL Classification](#) (Simple, Complex, and Problematic) before and after the configuration parameters were changed.

Related Topic

[Configuration Analyzer Window](#)

Right Pane for SQL Statement

The detailed information about the SQL statements used in the Analysis displays in the button pages. The buttons for displaying specific information are found at the top of this pane.

Properties button

Displays general information about the SQL statement and the database settings at the time of the analysis.

SQL Information button

Displays the SQL text in the top portion of the right pane and two [SQL Information panes](#), one for the SQL information before the analysis and one for the SQL information with the parameter changes, in the bottom right pane.

Saving the Abstract Plan

The abstract plan for a SQL statement can be saved from this pane. In order to save the abstract plan, it must be retrieved when the SQL was selected to be analyzed. This is done from the Abstract Plan page of the New Analysis wizard.

1. Select the SQL statement in the left pane.
2. Click SQL Information section of the right pane. This enables the Save Abstract Plan button.
3. Click **Save Abstract Plan** .
4. In the **Save to group** field, select the abstract plan group.
5. Click **Save**.

Related Topic

[Configuration Analyzer Window](#)

Create a Configuration Analysis

To create a Configuration Analysis:

1. Click  to open the Configuration Analyzer window.
2. If no analysis exists in the Configuration Analyzer then the New Analysis wizard displays automatically. Otherwise, click  to open the New Analysis window.
3. The New Analysis wizard has three pages for selecting the options for creating an analysis:

Page	See...
Analyzer	New Analysis Wizard: Analyzer Page
Selected SQL	New Analysis Wizard: Select SQL page
Configuration	New Analysis Wizard: Configuration Page
Abstract Plan	New Analysis Wizard: Abstract Plan

4. After making your selection, click **OK**.

To abort the Configuration analysis

Select **Analysis** | **Abort Analysis**.

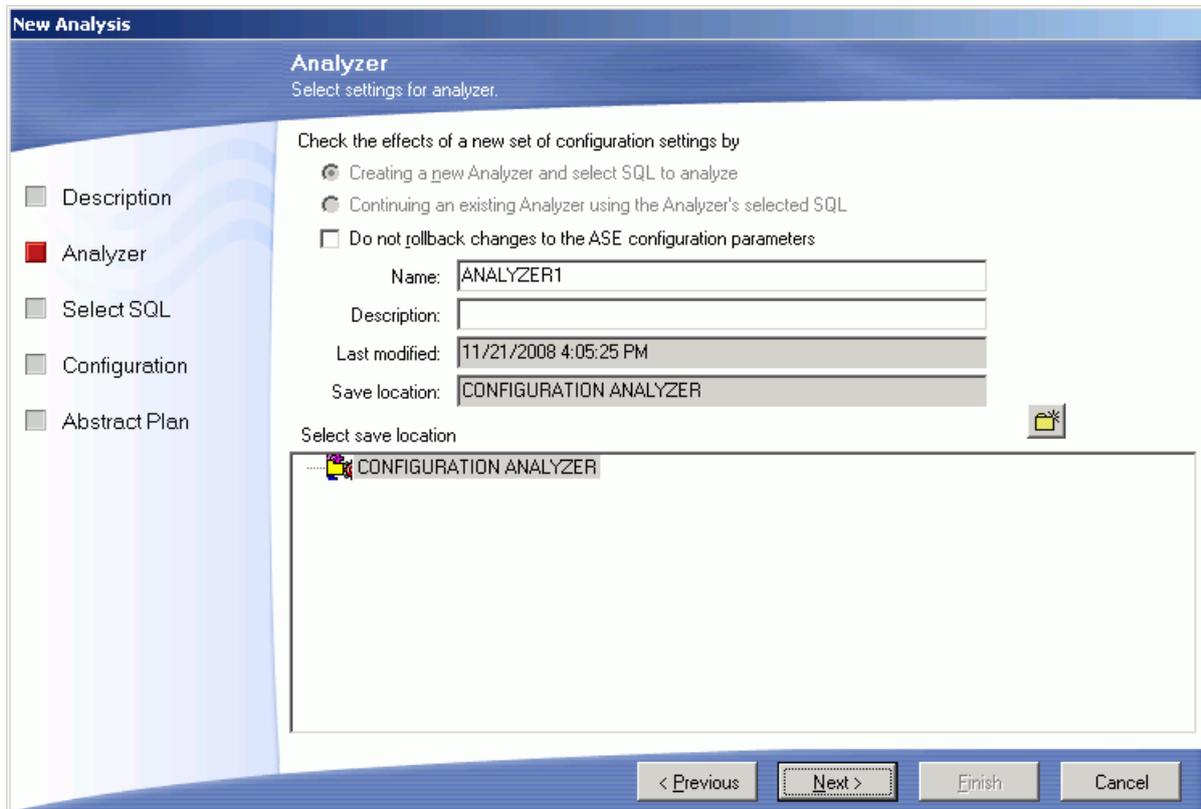
Related Topics

[Configuration Analyzer Overview](#)

[Configuration Analyzer Window](#)

New Analysis Wizard: Analyzer Page

View New Analysis Wizard - Analyzer Page



The Analyzer page is used to name the analysis and to select the folder where the SQL statements that you want to analyze are stored.

Creating a new Analyzer and select SQL to be analyzed

Select this option to create a new analysis using a different set of SQL statements.

Continuing an existing Analyzer using the Analyzer's selected SQL

Select this option to use the same set of SQL statements that is stored in an existing Analyzer. This option adds another Scenario under an existing Analyzer and does not retrieve a new set of query plans for the SQL statements. It uses the query plans that were originally saved with the SQL statements.

Do not rollback changes to the ASE configuration parameters

Specific to leaves the database configuration changes that you select under the Configuration tab after the analysis is finished. Otherwise, these changes are rolled back at the end of the analysis processing. After the

analysis is complete, the **Apply/Undo Changes** button displays at the top of the Configuration Analyzer window which displays the Apply Changes and the Undo Changes panes. These panes contain the scripts to make changes to the Adaptive Server configuration parameters.

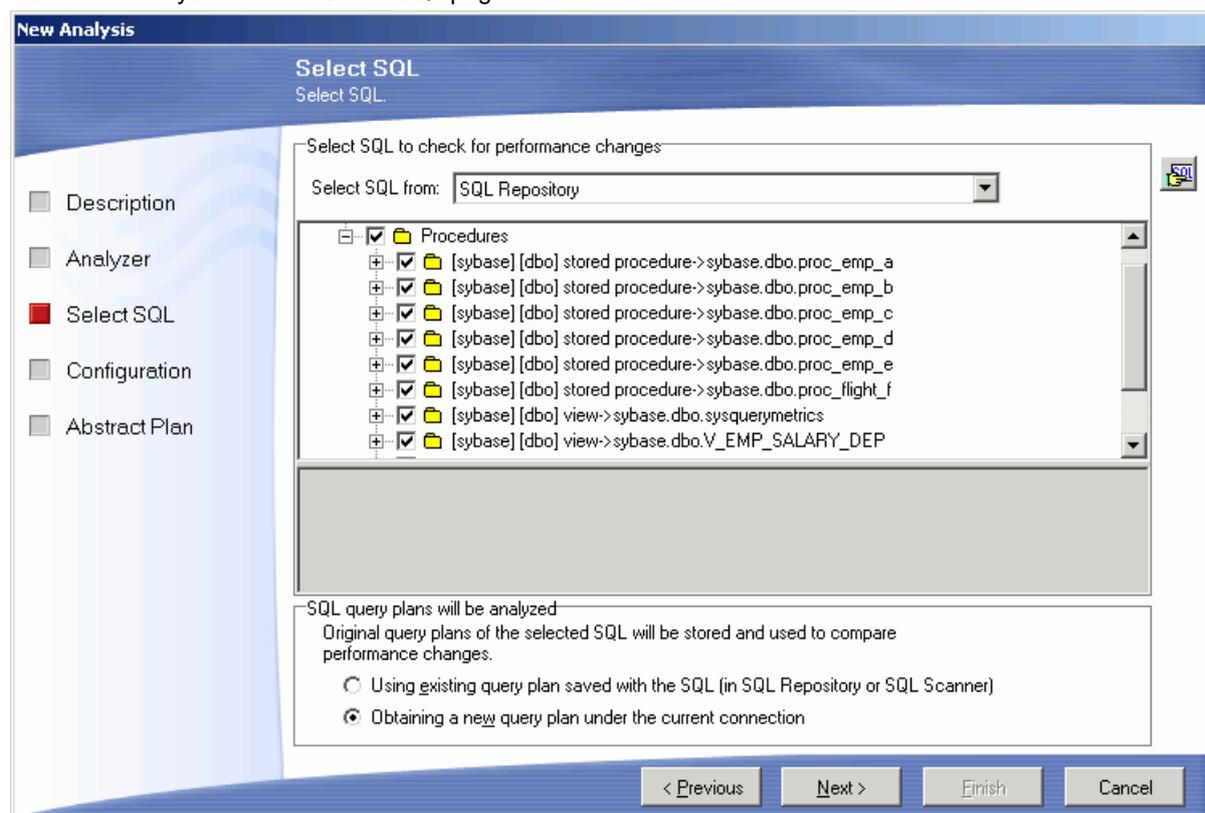
Item	Description
Name	Enter the name for the analysis.
Description	Enter the description for the analysis.
Last modified	Displays the last modified date and time.
Save location	Displays the folder in the tree where the analysis is saved. In the bottom pane, select the folder where you want to save the analysis. To create a new folder, click Add Folder  .

Related Topic

[Create a Configuration Analysis](#)

New Analysis Wizard: Select SQL page

View New Analysis Wizard - Select SQL page



The Select SQL page is used to select the SQL statements whose query plans you want to analyze when various database parameters are changed.

Select SQL to check for performance changes:

Select SQL from

The SQL statements used in an analysis can be taken from the SQL Repository or the SQL Scanner. If you select SQL statements from the SQL Scanner, these statements are added to the SQL Repository.

SQL query plans will be analyzed:

In the Configuration Analyzer, the query plans for the selected SQL statements before the configuration change are compared to the query plans after the change. The following options allow you to determine the "before" query plans.

Using existing query plan saved with the SQL

This option uses the query plan that was saved with the SQL statement when it was saved in the SQL Repository or SQL Scanner.

Obtaining a new query plan under the current connection

This option retrieves the query plan with the current logon and the current database settings. This current query plan is compared to the query plan that is retrieved after the configuration changes are made.

Related Topic

[Create a Configuration Analysis](#)

New Analysis Wizard: Configuration Page

[View New Analysis Wizard - Configuration Page](#)

New Analysis

Configuration
Enter new values for the Adaptive Server configuration parameters.

Query plans under your specified configuration settings will be grouped under a Scenario

Name:
Description:

Please specify your new configuration values in the New Value column
Show configuration parameters for:

For some parameter, changes will take effect only when you restart Adaptive Server

Name	Default	Current Value	New Value	Unit	Restart/Read-Only
extended cache size	0	0		memory pa	
global async prefetch limit	10	10		percent	
global cache partition number	1	1		number	Restart
memory alignment boundary	2048	2048		bytes	Restart
number of index trips	0	0		number	
number of oam trips	0	0		number	
total data cache size	0	8774		kilobytes	Read-Only

Minimum value: Maximum value:

Monitoring Information:

Description: Specifies the alignment of the data cache with the memory address boundary on certain hardware platforms. Do not change this parameter unless asked to do so by Sybase Technical Support.

< Previous Next > Finish Cancel

The Configuration page is used to change the database parameters. The query plans for the selected SQL statements are retrieved after the parameters are changed in order to analysis the effect changing database parameters will have on the performance of the SQL statements.

Each set of configuration parameters used in an analysis is called a Scenario. When you create an analysis, you can create one Scenario. Additional Scenarios can be created after the analysis is created.

Item	Description
Name	Enter a name for this Scenario.
Description	Enter a description for this Scenario.
Show configuration parameters for	You can narrow the selection of parameters available in the parameter grid by selecting the parameter category from this drop-down list.
Parameter settings grid	Enter the new settings for the parameters you want to analyze and what effect changing them will have on the performance of your SQL statements. This grid displays the parameter name, the default parameter value, the current value, unit of measurement for the parameter and if you must restart the database before the parameter change can take affect. Enter the new parameter setting in the New Value column.
Minimum and Maximum value	Displays the lowest and the highest value that you can enter for the selected parameter.

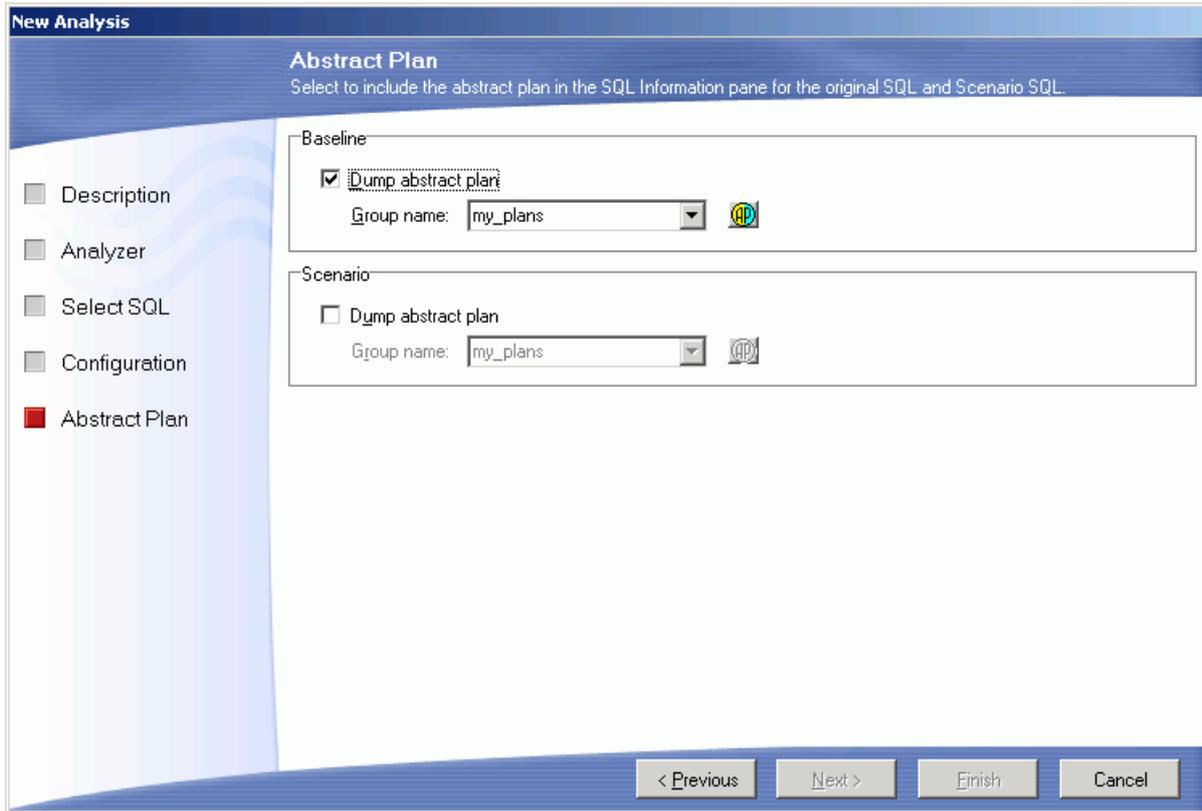
Monitor Information	Displays the following information. This information is only available for some of the Adaptive Server configuration parameters.
num_free	Amount of scan descriptors, both metadata and auxiliary, which are not currently in use.
num_active	Amount of scan descriptors, both metadata and auxiliary, which are installed in cache and therefore are in use.
pct_act	Percentage of the total number of scan descriptors that are installed in cache and therefore are in use.
Max_Used	Maximum amount of scan descriptors, both metadata and auxiliary, which were in use at any one time since starting the server was started.
Reuse	Number of scan descriptors reused.
Description	Provides an explanation of the function of the selected Adaptive Server configuration parameter.

Related Topic

[Create a Configuration Analysis](#)

New Analysis Wizard: Abstract Plan

In the New Analysis wizard, the Abstract Plan page is used to include the abstract plan with SQL statements.



Review the following for additional information:

Baseline Plan

Dump abstract plan

Description

Specify whether to retrieve the abstract plan for the SQL statement for the before the configuration parameters are changed. The abstract plan displays in the right pane for SQL statements. This retrieves a new abstract plan even if one was saved with the original SQL.

It is important to use this option if you original saved the SQL statement and abstract plans in version 12.5.0.3 or earlier of Adaptive Server and are now using Adaptive Server 15.0 or later since the format of the abstract plan has changed.

Note: This option is disabled if you have selected **Using existing query plan saved with the SQL** on the Select SQL page.

Group name

Specify the abstract plan group name where the abstract plans are saved. The default group names in Adaptive Server are: *ap_stdout* and *ap_stdin*. These groups are usually used by the Database Administrator to enable server-wide abstract plan capturing and retrieving.

- *ap_stdout* is used by default to capture an abstract plan.
- *ap_stdin* is used by default to retrieve the abstract plan associated with a SQL statement during the execution of the SQL statement.



Opens the Abstract Plan Manager window to view, create, and modify abstract plan group.

Scenario

	Description
Dump abstract plan	Specify whether to retrieve the abstract plan for the SQL statement after the configuration parameters are changed. The abstract plan displays in the right pane for SQL statements.
Group name	Specify the abstract plan group name where the abstract plans are saved. The default group names in Adaptive Server are: <code>ap_stdout</code> and <code>ap_stdin</code> . These groups are usually used by the Database Administrator to enable server-wide abstract plan capturing and retrieving. <ul style="list-style-type: none"> • <code>ap_stdout</code> is used by default to capture an abstract plan. • <code>ap_stdin</code> is used by default to retrieve the abstract plan associated with a SQL statement during the execution of the SQL statement.
	Opens the Abstract Plan Manager window to view, create, and modify abstract plan group.

Related Topic

[Create a Configuration Analysis](#)

Regenerate a Configuration Scenario

The Scenario can be re-executed using the exact same parameter settings. You can choose to retrieve the query plans from using the same parameter settings for all SQL statements in the Scenario or select specific SQL to retrieve the query plan using the parameter setting.

The function is useful if changes have occurred in the database environment, such as data volumes, that may change the query plans.

To regenerate

1. Select the Scenario you want to modify from the left pane.
2. Right-click and select **Regenerate**.
3. From the Regenerate Scenario window, check the SQL statements whose current query plans you would like retrieved using the parameter changes.

Related Topic

[Configuration Analyzer Window](#)

Add a Scenario

After creating a configuration analysis, you can add more Scenarios to it. A Scenario can be added in two ways.

To add a Scenario

Method One

1. In the left pane, right-click the analysis and select **Add Scenario**.
2. In the Add Scenario wizard, click the Configuration page and set the new database parameters'

Method Two

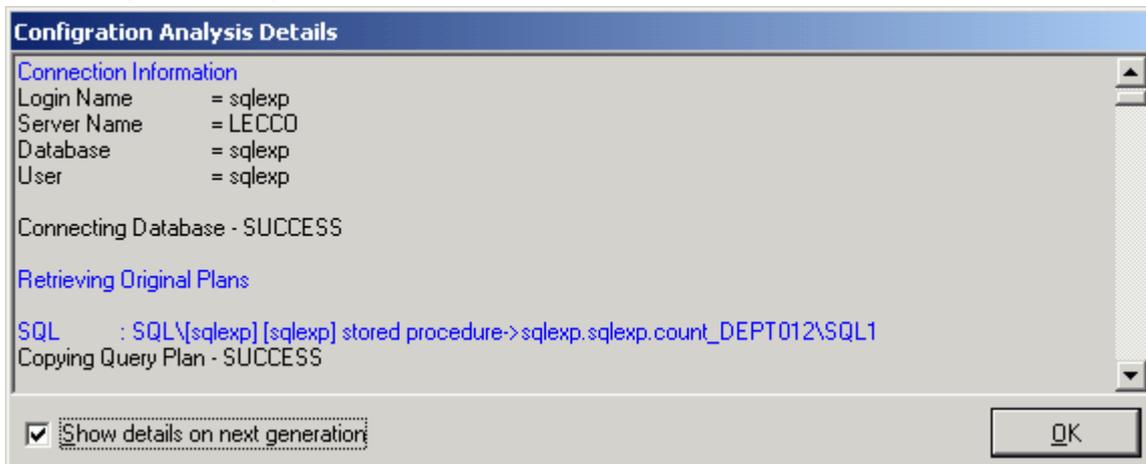
1. In the left pane, right-click and select **New Analysis**.
2. In the New Analysis wizard, select the **Continuing an existing Analyzer using the Analyzer's select SQL** option.
3. From the **Select Analyzer** field, click **Analyzer**.
4. Click the Configuration page and set the new database parameters.

Related Topic

[Configuration Analyzer Window](#)

View Configuration Analysis Details

View Configuration Analysis Details Window



The Configuration Analysis Details window displays information about the retrieve of the query plan for each SQL statement.

The Configuration Analysis Details window displays after the batch run process is completed unless the **Show details on next batch run** checkbox in the Configuration Analysis Details window is unchecked.

To view the Configuration Analysis Details window after an analysis

1. Select **View | Show Configuration Analysis**.
2. Review this information to see if an error occurred during the retrieval of the query plan. Errors may occur if the selected SQL statements are from one database and user and you have set another database and user in the drop-down boxes at the bottom left of the main window.

Related Topic

[Create a Configuration Analysis](#)

Rollback Changes to Configuration Parameters

The Configuration analysis first executes the configuration parameter changes that you have selected on your database. Then it retrieves the query plan for each SQL statement that you have selected for your analyzer. When you set up the individual scenarios, you determine if you would like the configuration parameter changes rolled back or to leave them set on Adaptive Server.

If you do not have the analysis process rollback the configuration changes, you can roll them back in any one of the following ways:

- In the tree pane (left pane), right-click the scenario name and select **Go To Undo Changes**.
- When you close the Configuration Analyzer window, you are prompted to rollback the changes. Click **Yes**.
- In the tree pane (left pane), click the scenario name and select the **Apply/Undo Changes** button at the top of the right pane. Click  next to the Undo Change pane.

You can make the changes permanent, in one of the following ways:

- In the tree pane (left pane), right-click the scenario name and select **Confirm Changes**.
- When you close the Configuration Analyzer window, you are prompted to rollback the changes. Click **No**.

Related Topic

[Configuration Analyzer Window](#)

Add SQL to a Configuration Analysis

After creating a configuration analysis, you can add more SQL statements from the SQL Repository to it.

To add SQL to an existing configuration analysis

1. From the left pane, select the analysis in which you want to add the SQL.
2. Right-click and select **Add SQL**.
3. From the Add SQL window under the **Select SQL to be added** pane, check the SQL statements you want added.
4. Under the **Select Scenario to include SQL** pane, check the Scenarios you want the SQL added to.
5. Click **OK** to add the SQL, retrieve the query plans and re-execute the Scenario.

Related Topic

[Configuration Analyzer Window](#)

Modify a Configuration Analysis

You can only modify the analysis name and description of a configuration analysis.

To modify a Configuration Analysis

1. From the Configuration Analyzer window, select the Analysis name you want to modify from the left pane.
2. Right-click and select **Modify**.
3. Modify the name and/or description

Related Topic

[Configuration Analyzer Window](#)

Rename a Configuration Analysis or Scenario

You can rename an Analysis or Scenario in the Configuration Analyzer window in two ways.

To rename an Analysis or Scenario

1. Right-click the Analysis or Scenario and select **Rename**. A highlighted field is placed around the name in the tree to indicate that you can edit the name.
2. Change the name and press **Enter** or click any where outside the field.

or

1. Right-click the Analysis or Scenario and select **Modify**.
2. Modify the name.
3. Click **OK** to save the changes.

Related Topic

[Configuration Analyzer Window](#)

Delete a Configuration Analysis or Scenario

To delete an analysis or a Scenario

1. Select the item you want to delete in the left pane.
2. Right-click and select **Delete**.

Related Topic

[Configuration Analyzer Window](#)

Modify a Configuration Scenario

You can only modify the Scenario name and description of the Scenario.

To modify a Configuration Scenario

1. Select the Scenario name you want to modify from the left pane.
2. Right-click and select **Modify** to open the Modify Scenario window.
3. Modify the name and/or description.

Related Topic

[Configuration Analyzer Window](#)

Configuration Analyzer Functions

Below is a list of available functions within the Configuration Analyzer window.

Menu	Function	See...
------	----------	--------

Right-click Menu	New Analysis/Abort Analysis	Create a Configuration Analysis
Right-click Menu	Add Folder	
Right-click Menu	Add Scenario	Add a Scenario
Right-click Menu	Add SQL	Add SQL to a Configuration Analysis
Right-click Menu	Regenerate	Regenerate a Configuration Scenario
Right-click Menu	Modify	Modify a Configuration Analysis
Right-click Menu	Rename	Rename a Configuration Analysis or Scenario
Right-click Menu	Delete	Delete a Configuration Analysis or Scenario
Right-click Menu	Go To Undo Changes	Rollback Changes to Configuration Parameters
Right-click Menu	Confirm Changes	Rollback Changes to Configuration Parameters
View Menu	Last Configuration Analysis Details	View Configuration Analysis Details

Related Topic

[Configuration Analyzer Overview](#)

[Configuration Analyzer Window](#)

Migration Analyzer Overview

The Migration Analyzer evaluates the effect on SQL performance when migrating to another database or Adaptive Server version. It enables you to analyze whether the SQL performance is going to improve or degrade by simply making a connection to the new database and viewing how query plan information will change for your selected SQL statements without physically having to execute the database applications in the new Adaptive Server version. The Migration Analyzer compares the query plans from the current database to the query plans that are retrieved from the destination Adaptive Server instance during the Migration analysis to project performance improvement or degradation.

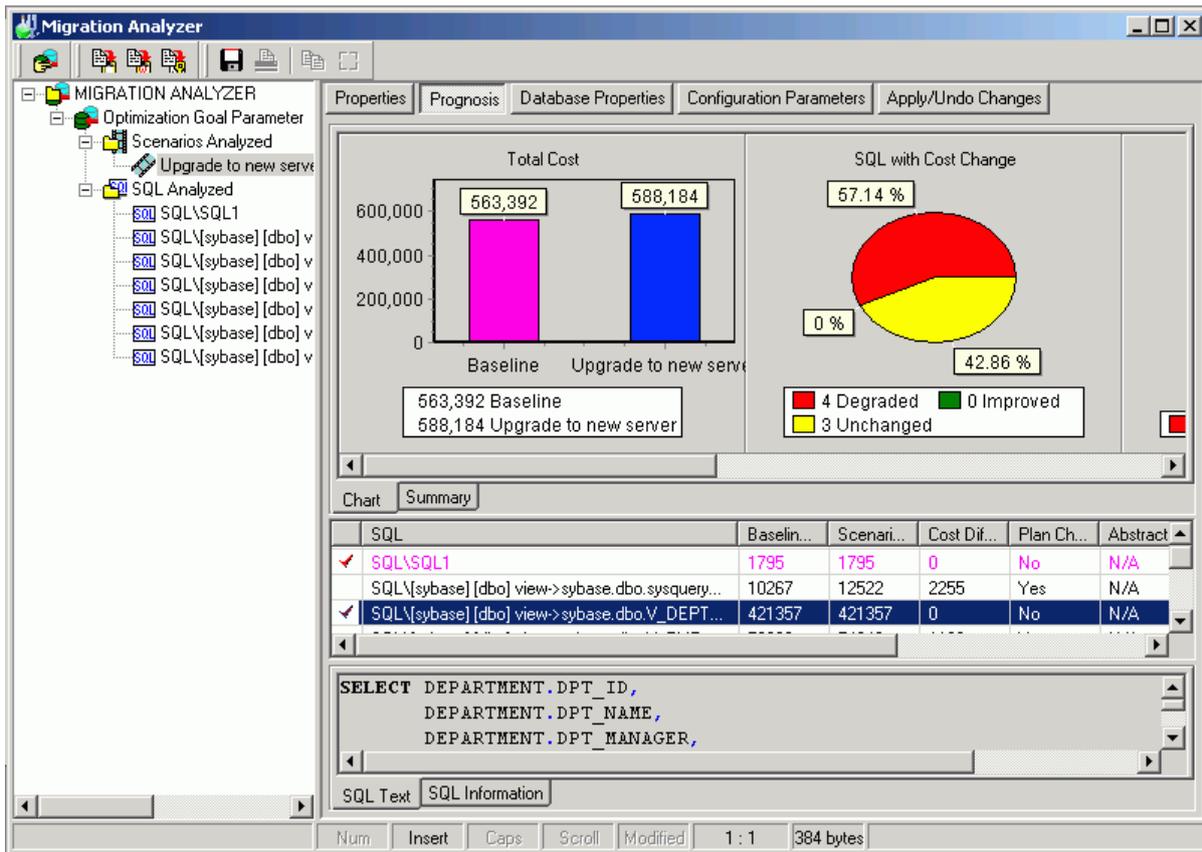
Related Topic

[Migration Analyzer Window](#)

[Migration Analyzer Functions](#)

Migration Analyzer Window

[View Migration Analyzer Window](#)



The Migration Analyzer window displays performance analysis information about the SQL statements and query plans analyzed between two Adaptive Server versions. For each performance analysis selected from the tree view in the left pane, overall performance change metrics and details on query plan changes are provided in the right pane.

Left pane - Analyzer List

Provides a tree view of the Analyzers that provide performance analysis information for different migration Scenarios. For each Analyzer two folders are provided:

Migration Information

This folder displays performance analysis information for migration Scenarios.

SQL Analyzed

Displays the SQL statements selected from the SQL Repository for analysis and the query plans for each of the Scenarios.

Right Pane

The right pane displays a variety of different information depending upon what is selected from the tree diagram in the left pane.

You can display information for the following:

[Analyzers](#)

[Scenarios Analyzed Folder](#)

[Scenario](#)

[SQL Analyzed Folder](#)

[SQL Statements](#)

Related Topics

[Create a Migration Analysis](#)

[Migration Analyzer Overview](#)

[Migration Analyzer Functions](#)

Right Pane for Analyzers

In the Migration Analyzer window, when an Analyzer is selected from the tree in the left pane, the following information displays in the right pane.

Properties Button

Displays general information about the analysis, information on the connection used to retrieve the query plans and the SQL statements used in the analysis.

Related Topic

[Migration Analyzer Window](#)

Right Pane for Scenarios Analyzed

In the Migration Analyzer window, when **Scenarios Analyzed** is selected in the left pane, the following information displays in the right pane.

Summary button

Provides a summary of the Estimated I/O Cost and SQL classification for the SQL statements from the current Adaptive Server instance and from the destination Adaptive Server instance for the migration.

Charts button

Charts the total Estimated I/O Cost of the SQL statements from the current Adaptive Server instance and from the destination Adaptive Server instance for the migration.

Related Topic

[Migration Analyzer Window](#)

Right Pane for Scenarios

In the Migration Analyzer window, when a specific Scenario is selected in the left pane, the detailed information about the Scenario displays in the button pages in the right pane. The buttons for displaying specific information are found at the top of this pane.

Properties button

Displays general information about the SQL statement and the database settings at the time of the analysis.

Prognosis button

Displays 3 panes in the right section of the window.

Charts and Summary tabs [Top right pane]

The Chart tab displays charts of the comparison information for the query plans from before the analysis and during the analysis.

The Summary tab displays summarized statistics for the Scenario.

SQL Classification and Plan Change Comparisons [Middle right pane]

A grid displays the comparison information about the query plan from the current Adaptive Server instance and the query plan on the destination Adaptive Server instance for the migration.

SQL Text and SQL Information tabs [Bottom right pane]

The SQL statement that is highlight in the middle right pane is the one that displays in this pane.

The **SQL Text** tab displays the text of the SQL statement.

The **SQL Information** tab displays two [SQL Information panes](#), one for the SQL information on the current Adaptive Server instance and the query plan on the destination Adaptive Server instance for the migration.

Database Properties button

Displays the Adaptive Server database options and the devices from the current Adaptive Server instance and from the destination Adaptive Server instance for the migration.

Configuration Parameters button

Displays the Adaptive Server configuration changes used on the destination Migration Adaptive Server instance, showing the parameter name, the value applied during the analysis, and the old value for the parameters that were changed.

Note: The Configuration Parameters only displays if configuration changes were made for this analysis.

Apply/Undo Changes button

Displays the script that changes the configuration parameters for the migration database. Click **Execute**  located at the right of the Migration Analyzer window to run the script.

Related Topic

[Migration Analyzer Window](#)

Right Pane for SQL Analyzed Folder

In the Migration Analyzer window, when the SQL Analyzed folder is selected in the left pane, the following information displays in the right pane.

Cost Summary button

Displays all the SQL statements and shows the Estimated I/O Cost from the current Adaptive Server instance and from the destination Adaptive Server instance for the migration.

Cost Classification button

Displays all the SQL statements and shows the [SQL Classification](#) (Simple, Complex, and Problematic) from the current Adaptive Server instance and from the destination Adaptive Server instance for the migration.

Related Topic

[Migration Analyzer Window](#)

Right Pane for SQL Statement

Displays detailed information about the SQL statements used in the Analysis. The buttons for displaying specific information are found at the top of this pane.

Properties button

Displays general information about the SQL statement and the database settings on the current database at the time of the analysis.

SQL Information button

Displays the SQL statement information for comparison of possible performance changes between the current Adaptive Server instance and on the destination Adaptive Server instance for the migration. The SQL text is in the top portion of the right pane. The bottom portion of the window has two [SQL Information panes](#), for comparison of the query plans, abstract plans, Trace On information, SQL classification, and DDL for any temporary tables that are required by the SQL statement.

Saving the Abstract Plan

The abstract plan for a SQL statement can be saved from this pane. In order to save the abstract plan, it must be retrieved when the SQL was selected to be analyzed. This is done from the [Abstract Plan page](#) of the New Analysis wizard.

To save an abstract plan

1. Select the SQL statement in the left pane.
2. Click SQL Information section of the right pane. This enables the Save Abstract Plan button.
3. Click **Save Abstract Plan** .
4. In the **Save to group** field, select the abstract plan group.
5. Click **Save**.

Related Topic

[Migration Analyzer Window](#)

Create a Migration Analysis

To create a Migration Analysis

1. Click  to open the Migration Analyzer window.
2. If no analysis exists in the Migration Analyzer then the New Analysis wizard displays automatically.
Otherwise, click .
3. The New Analysis wizard has four pages for selecting the options for creating an analysis.
 - [Analyzer](#)
 - [Select SQL](#)
 - [Migration](#)
 - [Destination Configuration](#)
 - [Abstract Plan](#)
4. After making the selections from these four tabs, click **OK**.

To abort the Migration analysis

Select **Analysis | Abort Analysis**.

When the analysis is finished, the [Migration Analysis Details window](#) displays with information the retrieve of the query plan for each SQL statement.

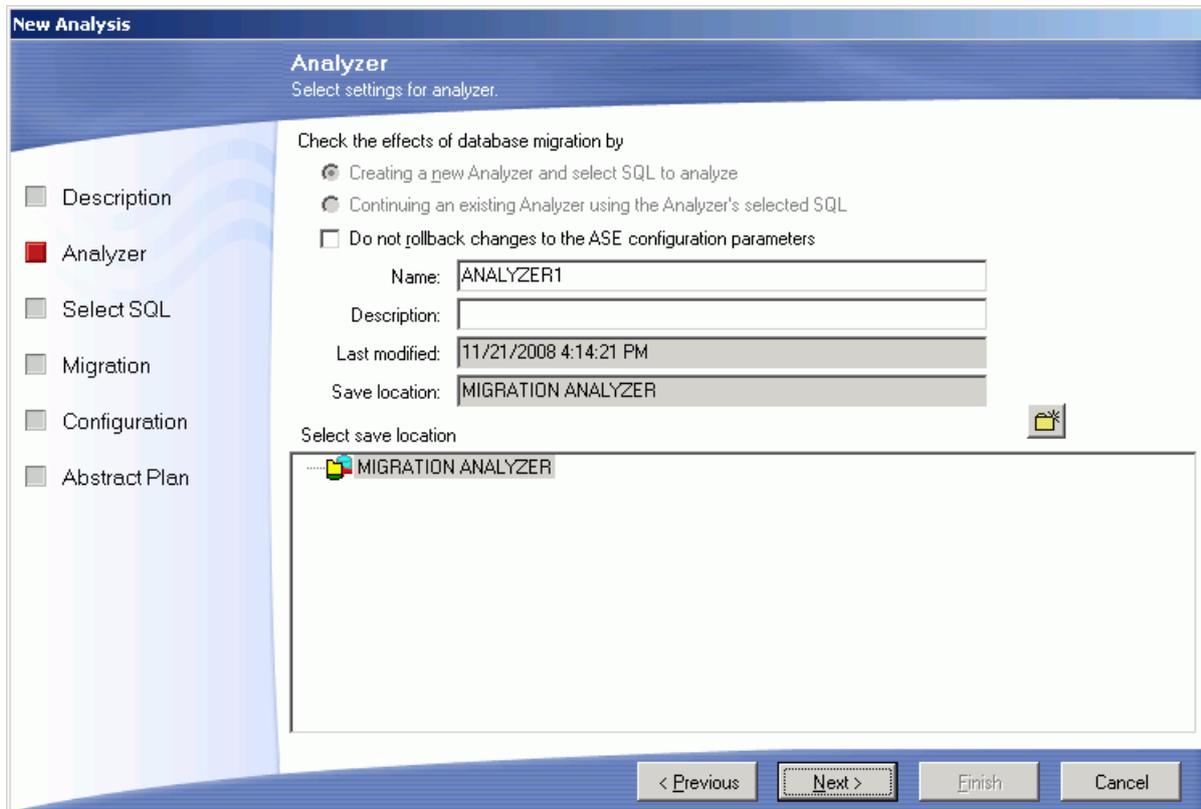
Related Topics

[Migration Analyzer Overview](#)

[Migration Analyzer Window](#)

New Analysis Wizard: Analyzer Page

View New Analysis Wizard - Analyzer Page



The Analyzer tab is used to name the analysis and to select the folder where the SQL statements that you want to analyze are stored.

Creating a new Analyzer and select SQL to be analyzed

Select this option to create a new analysis using a different set of SQL statements.

Continuing an existing Analyzer using the Analyzer's selected SQL

Select this option to use the same set of SQL statements that is stored in an existing Analyzer. This option adds another Scenario under an existing Analyzer and does not retrieve a new set of query plans for the SQL statements. It uses the query plans that were originally saved with the SQL statements.

Do not rollback changes to the ASE configuration parameters

Specific to leaves the database configuration changes that you select under the Configuration tab after the analysis is finished. Otherwise, these changes are rolled back at the end of the analysis processing. After the

analysis is complete, the **Apply/Undo Changes** button displays at the top of the Migration Analyzer window which displays the Apply Changes and the Undo Changes panes. These panes contain the scripts to make changes to the Adaptive Server configuration parameters.

Item	Description
Name	Enter the name for the analysis.
Description	Enter the description for the analysis.
Last modified	Displays the last modified date and time.
Save location	Displays the folder in the tree where the analysis is saved. In the bottom pane, select the folder where you want to save the analysis.

To create a new folder

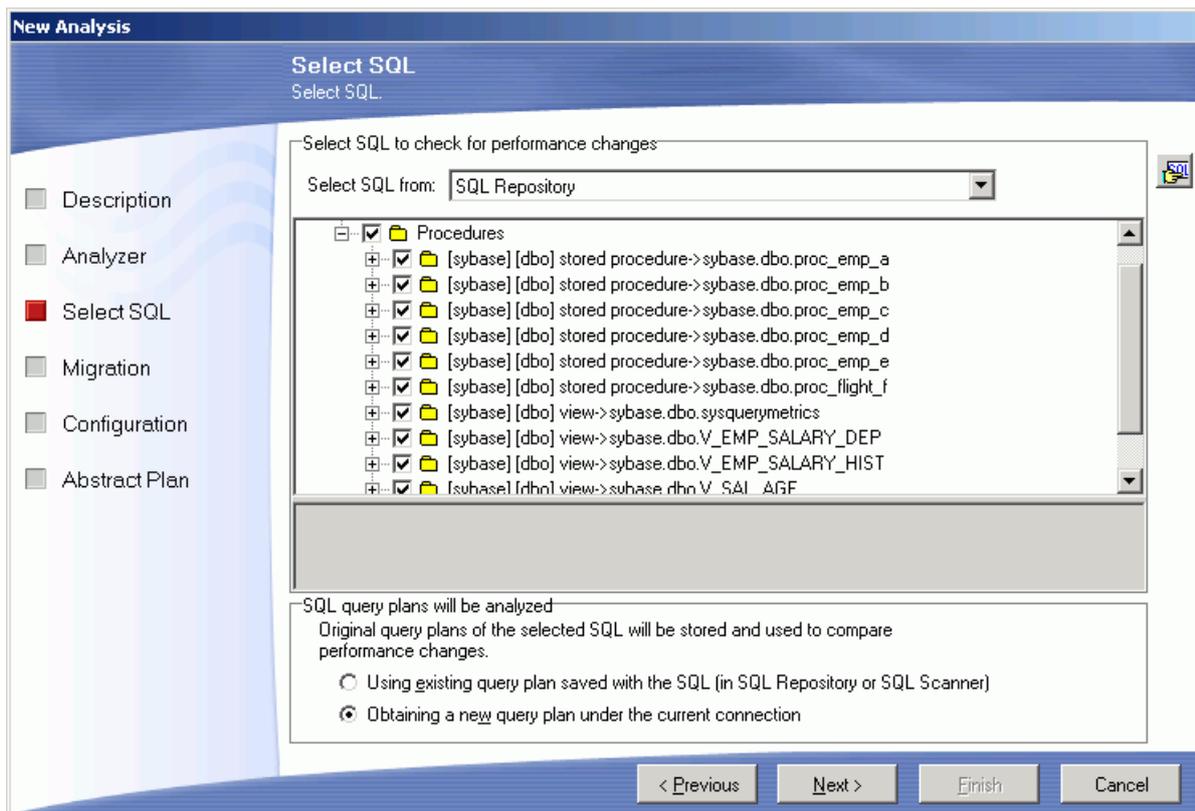
Click **Add Folder** .

Related Topic

[Create a Migration Analysis](#)

New Analysis Wizard: Select SQL Page

View New Analysis Wizard - Select SQL Page



The Select SQL page is used to select the SQL statements whose query plans you want to analyze when various database parameters are changed.

Select SQL to check for performance changes:

Select SQL from

The SQL statements used in an analysis can be taken from the SQL Repository and the SQL Scanner. If you select SQL statements from the SQL Scanner, these statements are added to the SQL Repository.

SQL query plans will be analyzed:

In the Migration Analyzer, the query plans for the selected SQL statements current database are compared to the query plans in the database for the migration. The following options allow you to determine the "before" query plans.

Using existing query plan saved with the SQL

This option uses the query plan that was saved with the SQL statement when it was saved in the SQL Repository or SQL Scanner.

Obtaining a new query plan under the current connection

This option retrieves the query plan with the current database logon that is assumed to be the source Adaptive Server instance you are migrating from. This current query plan is compared to the query plan that is retrieved from the destination Adaptive Server instance for the migration.

Related Topic

New Analysis Wizard: Migration Page

View New Analysis Wizard - Migration Page

The Migration tab is used to specify the database connection to retrieve the query plans for the selected SQL statements from the Adaptive Server instance for the migration in order to analysis the changes in the query plans that result from migrating to a new Adaptive Server instance.

Item	Description
Name	Enter a name for this Scenario.
Description	Enter a description for this Scenario.
Login name	Enter the name of the login for the destination Adaptive Server instance.
Password	Enter the password of the login.
Server name	Enter the database server you are logging into from the drop down field, or enter the name directly.
Use default database and user	Select this option if you would like to login with the default database and user information and privileges.

Specify database and user

Select this option if you would like to set the user to one with different credentials or privileges. Enter the database and user.

Note: The new query plans from the Adaptive Server instance for the migration are compared to the query plans that were retrieved either when the selected SQL was saved or that are retrieved using the current logon.

Related Topic

[Create a Migration Analysis](#)

New Analysis Wizard: Destination Configuration Page

New Analysis Wizard - Destination Configuration Page

New Analysis

Configuration
Enter new values for the Adaptive Server configuration parameters.

Please specify your new configuration values in the New Value column

Show configuration parameters for:

For some parameter, changes will take effect only when you restart Adaptive Server

Name	Default	Current Value	New Value	Unit	Restart/Read-Only
extended cache size	0	0		memory pa	
global async prefetch limit	10	10		percent	
global cache partition number	1	1		number	Restart
memory alignment boundary	2048	2048		bytes	Restart
number of index trips	0	0		number	
number of oam trips	0	0		number	
total data cache size	0	8774		kilobytes	Read-Only

Minimum value: Maximum value:

Monitoring Information:

Description: Specifies the default setting for the number of cache partitions that are used for the data cache. This global parameter setting is used if the number of cache partitions for a particular data cache has not been explicitly set with sp_cacheconfig.

< Previous Next > Finish Cancel

The Destination Configuration page is used to change the database parameters on the migration database and only available after you have entered your migrating database connection information on the Migration tab. The query plans for the selected SQL statements are retrieved after the parameters are changed in order to analyze the effect changing Adaptive Server configuration parameters will have on the performance of the SQL statements on the destination Adaptive Server instance.

Item

Description

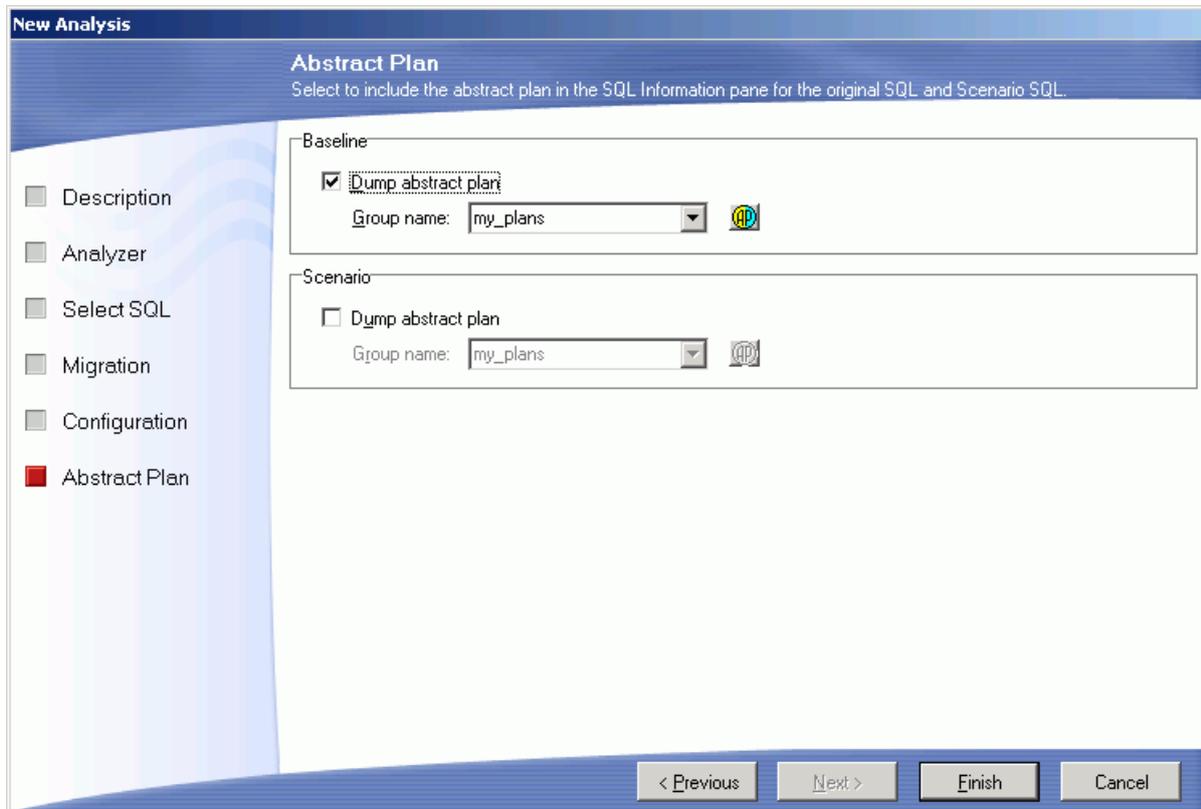
Name	Enter a name for this Scenario.										
Description	Enter a description for this Scenario.										
Show configuration parameters for	You can narrow the selection of parameters available in the parameter grid by selecting the parameter category from this drop-down list.										
Show configuration parameters for	You can narrow the list of Adaptive Server configuration parameters available in the parameter grid by selecting the parameter category from this drop-down list.										
Parameter settings grid	Enter the new settings for the parameters you want to analyze and what effect changing them will have on the performance of your SQL statements on the destination Adaptive Server instance. This grid displays the parameter name, the default parameter value, the current value, unit of measurement for the parameter and if you must restart the database before the parameter change can take affect. Enter the new parameter setting in the New Value column.										
Minimum and Maximum value	Displays the lowest and the highest value that you can enter for the selected parameter.										
Monitor Information	<p>Displays the following information. This information is only available for some of the Adaptive Server configuration parameters.</p> <table border="0"> <tr> <td>num_ free</td> <td>Amount of scan descriptors, both metadata and auxiliary, which are not currently in use.</td> </tr> <tr> <td>num_ active</td> <td>Amount of scan descriptors, both metadata and auxiliary, which are installed in cache and therefore are in use.</td> </tr> <tr> <td>pct_ act</td> <td>Percentage of the total number of scan descriptors that are installed in cache and therefore are in use.</td> </tr> <tr> <td>Max_ Used</td> <td>Maximum amount of scan descriptors, both metadata and auxiliary, which were in use at any one time since starting the server was started.</td> </tr> <tr> <td>Num_ Reuse</td> <td>Number of scan descriptors reused.</td> </tr> </table>	num_ free	Amount of scan descriptors, both metadata and auxiliary, which are not currently in use.	num_ active	Amount of scan descriptors, both metadata and auxiliary, which are installed in cache and therefore are in use.	pct_ act	Percentage of the total number of scan descriptors that are installed in cache and therefore are in use.	Max_ Used	Maximum amount of scan descriptors, both metadata and auxiliary, which were in use at any one time since starting the server was started.	Num_ Reuse	Number of scan descriptors reused.
num_ free	Amount of scan descriptors, both metadata and auxiliary, which are not currently in use.										
num_ active	Amount of scan descriptors, both metadata and auxiliary, which are installed in cache and therefore are in use.										
pct_ act	Percentage of the total number of scan descriptors that are installed in cache and therefore are in use.										
Max_ Used	Maximum amount of scan descriptors, both metadata and auxiliary, which were in use at any one time since starting the server was started.										
Num_ Reuse	Number of scan descriptors reused.										
Description	Provides an explanation of the function of the selected Adaptive Server configuration parameter.										

Related Topic

[Create a Migration Analysis](#)

New Analysis Wizard: Abstract Plan

View New Analysis Wizard - Abstract Plan



In the New Analysis wizard, the Abstract Plan page is used to include the abstract plan with SQL statements.

Baseline Plan

Dump abstract plan

Specify whether to retrieve the abstract plan for the SQL statement from the system you are migrating from. The abstract plan displays in the [right pane for SQL statements](#). This retrieves a new abstract plan even if one was saved with the original SQL. It is important to use this option if you original saved the SQL statement and abstract plans in version 12.5.0.3 or earlier of Adaptive Server and are now using Adaptive Server 15.0 or later since the format of the abstract plan has changed.

Note: This option is disabled if you have selected **Using existing query plan saved with the SQL** on the [Select SQL](#) page.

Group name

Specify the abstract plan group name where the abstract plans are saved. The default group names in Adaptive Server are: `ap_stdout` and `ap_stdin`. These groups are usually used by the Database Administrator to enable server-wide abstract plan capturing and retrieving.

ap_stdout is used by default to capture an abstract plan.

ap_stdin is used by default to retrieve the abstract plan associated with a SQL statement during the execution of the SQL statement.

Abstract Plan Manager button

Opens the Abstract Plan Manager window to view, create, and modify abstract plan group.

Scenario

Dump abstract plan

Specify whether to retrieve the abstract plan for the SQL statement from the system you are migrating to. The abstract plan displays in the [right pane for SQL statements](#).

Group name

Specify the abstract plan group name where the abstract plans are saved. The default group names in Adaptive Server are: ap_stdout and ap_stdin. These groups are usually used by the Database Administrator to enable server-wide abstract plan capturing and retrieving.

ap_stdout is used by default to capture an abstract plan.

ap_stdin is used by default to retrieve the abstract plan associated with a SQL statement during the execution of the SQL statement.

Abstract Plan Manager button

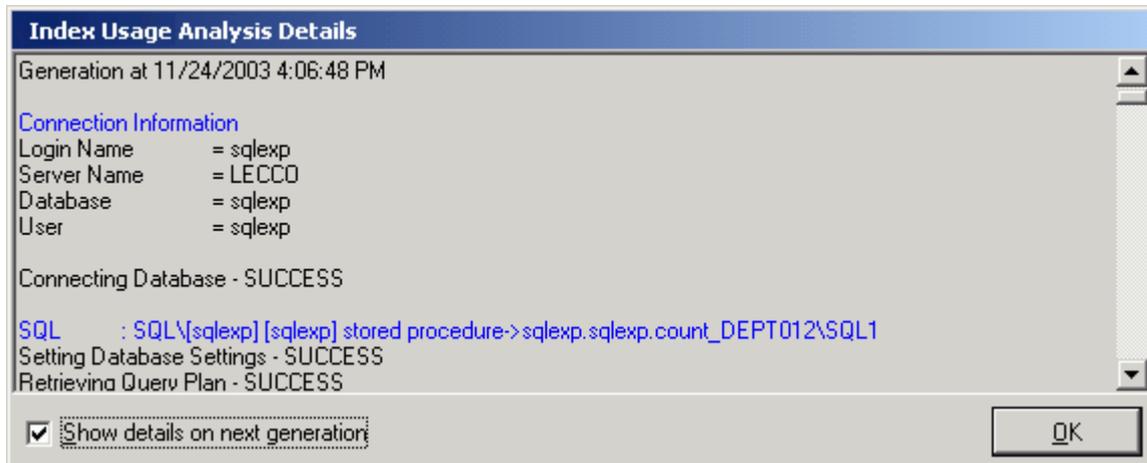
Opens the Abstract Plan Manager window to view, create, and modify abstract plan group.

Related Topic

[Create a Migration Analysis](#)

View Migration Analysis Details

View Migration Analysis Details Window



The Migration Analysis Details window displays information about the retrieve of the query plan for each SQL statement.

The Migration Analysis Details window displays after the batch run process is completed unless the **Show details on next batch run** checkbox in the Migration Analysis Details window is unchecked.

To view the Migration Analysis Details window after an analysis

1. Select **View | Show Migration Analysis Details**.
2. Review this information to see if an error occurred during the retrieval of the query plan. Errors may occur if the selected SQL statements are from one database and user and you have set another database and user in the drop-down boxes at the bottom left of the main window.

Related Topic

[Create a Migration Analysis](#)

Migration Analysis Results

Upon completion of the migration analysis process, a summary of the analysis is the Migration Analysis Details window, which tells if any of the query plans retrieval failed for any of the SQL in the Scenario. The comparison of the query plans, estimated I/O cost, abstract plans, Trace On information, and SQL Classification displays in the right pane of the Migration Analyzer window.

Once the analysis is complete, the best way to compare the behavior of the SQL in the analysis is to select the Scenario from the left tree structure.

Properties Button

The initial view is of the Properties of the Scenario and it gives a basic summary of the analysis including the configuration parameters that were specified as different between the databases.

Prognosis Button

The Prognosis section of an analysis is the best way to analyze the results to see how the performance of the SQL statements will be affected by the migration. The Prognosis has two tabs in the top pane that provide the overview of the impact of the migration on SQL performance: Chart and Summary. Specific information about the individual SQL statements displays in the middle and bottom panes.

Click any portion of the chart and the SQL statements represented are highlighted in the grid below the charts.

Chart Tab

Total Cost Chart

This chart shows the overall Estimated I/O cost differences of the SQL statements when connected to the current Adaptive Server instance and when connected to the destination Adaptive Server instance.

SQL with Cost Change Chart

This chart represents those specific statements whose query plans undergo a change in Estimated I/O cost.

Click any portion of the chart and the SQL statements represented are highlighted in the grid below the charts.

SQL with Plan Structure Change Chart

This chart represents those statements that experience a structure change in their query plan, regardless of any cost changes.

Classification Changes Chart

This chart indicates any SQL statements whose query plans undergo a change in [classification](#) (Simple, Complex, and Problematic). The statements that experience degradation in performance would be the best statements to work with to optimize before migrating to the new Adaptive Server instance, since they are the statements that could potentially cause performance problems.

SQL Classification Chart

This chart gives the aggregate view of the classifications of the SQL statements in the analysis.

Summary Tab

Presents a summary of the analysis in text format showing the statistics that are used in the charts.

Configuration Parameters Button

Displays the configuration parameters for the destination Adaptive Server instance. For each parameter it displays current value and the old parameter value before the analysis. The Configuration Parameters pane is only displayed if changes to configuration parameters were specified for the destination instance during the migration analysis.

Apply Changes

Displays the script for making the configuration parameters changes on Adaptive Server. Press **Execute**  to run the script.

Related Topic

[Migration Analyzer Window](#)

Add a Scenario

After creating a migration analysis, you can add more Scenarios to it. A Scenario can be added in two ways:

To add a Scenario

Method One

1. In the left pane, right-click the analysis and select **Add Scenario**.
2. In the Add Scenariowizard, click the [Migration](#) page and set the user connection information.
3. Click the [Destination Configuration](#) page and set the new database parameters.

Method Two

1. In the left pane, right-click and select **New Analysis**.
2. In the New Analysis wizard, select the Continuing an existing Analyzer using the Analyzer's select SQL option.
3. From the **Select Analyzer** field, click the **Analyzer** name.
4. Click the **Migration** page and set the user connection information
5. Click the **Destination Configuration** page and set the new database parameters.

Related Topic

[Migration Analyzer Window](#)

Rollback Changes to Configuration Parameters

The Migration analysis connects to the database you are migrating to and first executes the configuration parameter changes that you have selected on your database. Then it retrieves the query plan for each SQL statement that you have selected for your analyzer. When you set up the individual scenarios, you determine if you would like the configuration parameter changes rolled back or to leave them set on Adaptive Server.

If you do not have the analysis process rollback the configuration changes, you can roll them back in any one of the following ways:

- In the tree pane (left pane), right-click the scenario name and select **Go To Undo Changes**.
- When you close the Migration Analyzer window, you are prompted to rollback the changes. Click **Yes**.
- In the tree pane (left pane), click the scenario name and select the **Apply/Undo Changes** button at the top of the right pane. Click  next to the Undo Change pane.

You can make the changes permanent, in one of the following ways:

- In the tree pane (left pane), right-click the scenario name and select **Confirm Changes**. Click **Yes**.
- When you close the Migration Analyzer window, you are prompted to rollback the changes. Click **No**.

Related Topic

[Migration Analyzer Window](#)

Add SQL to a Migration Analysis

After creating a migration analysis, you can add more SQL statements to it.

To add SQL to an Analysis

1. From the left pane, select the analysis in which you want to add the SQL.
2. Right-click and select **Add SQL**.
3. From the Add SQL window under the **Select SQL to be added** pane, check the SQL statements you want added.
4. Under the Select Scenario to include SQL pane, check the Scenarios you want the SQL added to.
5. Click **OK** to add the SQL, retrieve the query plans and re-execute the Scenario.

Related Topic

[Migration Analyzer Window](#)

Modify a Migration Analysis

You can only modify the analysis name and description of a Migration analysis.

To modify an Analysis

1. Select the Analysis name you want to modify from the left pane.
2. Right-click and select **Modify** to open the Modify Analyzer window.
3. Modify the name and/or description.

Related Topic

[Migration Analyzer Window](#)

Regenerate a Migration Scenario

The Scenario can be re-executed using the exact same logon and configuration parameters. You can choose to retrieve the query plans for all SQL statements in the Scenario or select specific SQL to retrieve the query plan using the same logon and configuration parameters.

The function is useful if changes have occurred in the database environment, such as data volumes, that may change the query plans.

To regenerate a Scenario

1. Select the Scenario you want to modify from the left pane.
2. Right-click and select **Regenerate**.
3. From the Regenerate Scenario window, check the SQL statements whose current query plans you would like retrieved using the logon and parameter settings.

Related Topic

[Migration Analyzer Window](#)

Delete a Migration Analysis or Scenario

To delete an analysis or a Scenario

1. Select the item you want to delete in the left pane.
2. Right-click and select **Delete** or press Delete

Related Topic

[Migration Analyzer Window](#)

Rename a Migration Analysis or Scenario

You can rename an analysis or Scenario in the Migration Analyzer window in two ways.

To rename an Analysis or Scenario

1. Right-click the Analysis or Scenario and select **Rename**. A highlighted field is placed around the name in the tree to indicate that you can edit the name.
2. Change the name and press **Enter** or click any where outside the field.

or

1. Right-click the Analysis or Scenario and select **Modify** to open the Modify Analyzer or Modify Scenario window.
2. Modify the name.
3. Click **OK** to save the changes.

Related Topic

[Migration Analyzer Window](#)

Modify a Migration Scenario

You can only modify the Scenario name and description of the Scenario. From the Migration Analyzer window:

To modify a Scenario

1. Select the Scenario name you want to modify from the left pane.
2. Right-click and select **Modify** to open the Modify Scenario window.
3. Modify the name and/or description.

Related Topic

[Migration Analyzer Window](#)

Migration Analyzer Functions

Below is a list of available functions within the Migration Analyzer window.

Menu	Description
Analysis & Right-click Menu	New Analysis/Abort Analysis
Right-click Menu	Add Folder
Right-click Menu	Add Scenario
Right-click Menu	Add SQL
Right-click Menu	Regenerate
Right-click Menu	Modify
Right-click Menu	Rename
Right-click Menu	Delete
Right-click Menu	Go To Undo Changes
Right-click Menu	Confirm Changes
View Menu	Last Migration Analysis Details

Related Topic

[Migration Analyzer Overview](#)

[Migration Analyzer Window](#)

Abstract Plan Manager Overview

The Abstract Plan Manager provides a window for you to easily view, create, delete and modify your abstract plan groups.

In Adaptive Server version 15 and above, the abstract plan enables you to influence the optimization of a SQL statement without having to modify the SQL statement syntax. If you cannot change the source code that contains your SQL statement, you can use the abstract plan to force Adaptive Server to use a specific query plan for a SQL statement. This is particularly useful if you have third party applications where you do not have access to the source code.

Another use of the abstract plan is to protect the performance from changes to the database. When changes are made to a database, Adaptive Server may choose a different query plan for a SQL statement as a result of the changes. The abstract plans provide a means for system administrators and performance tuners to protect the overall performance of a SQL statement from these changes since the abstract plan will cause Adaptive Server to always choose the same query plan.

In the SQL Optimizer window, you can find alternate abstract plans in one of two ways while you are optimizing a SQL statement. First, you can optimize to find only the compatible alternative abstract plans. Second, you can optimize to find the semantically equivalent SQL statements with alternative query plans and then choose the ones with compatible alternative abstract plans.

Related Topics

[Abstract Plan Group](#)

[Why Save the Abstract Plan?](#)

[Abstract Plan Compatibility with Original SQL](#)

[Abstract Plan Manager Window](#)

[Open the Abstract Plan Manager](#)

[Abstract Plan Group Functions](#)

[Export a Group to a Table](#)

[Import an Abstract Plan for each User](#)

[Abstract Plan ID Functions](#)

[Use Saved Abstract Plans](#)

Abstract Plan Group

Abstract plans are saved to an abstract plan group. The group has the following elements associated with it:

- Database
- User-ID
- SQL statement
- Abstract Plan

When you save an abstract plan, you specify the database, group and user. This means that you can have a different abstract plan for different users. It enables SQL optimization to be based on the user's individual activities.

Here is an example of where you might want to have a different abstract plan for different users. In this example, the column `emp_sex` is indexed.

```
select * from employee where emp_sex=:var_sex
```

If 90% employees are male, then the SQL statement to retrieve the records where the employees are male should not be indexed for the best results.

But the SQL statement to retrieve the 10% of the employees who are female should be indexed.

For Group A users, who always select "male" employees, the Abstract Plan should do full table scan. For Group B users, who always select "female" employees, the Abstract Plan should use an index scan.

If you use the default group of *ap_stdin*, you can set Adaptive Server to use this group server-wide. You must save each abstract plan that you want to use system-wide in this group.

You can set up Adaptive Server to use different abstract plans for the same SQL statement from different applications. In this case, you create a different group for each application. For each user that you want to use a specific abstract plan, you must save that abstract plan in the group.

Related Topics

[Abstract Plan Manager Overview](#)

[Why Save the Abstract Plan?](#)

[Abstract Plan Compatibility with Original SQL](#)

[Abstract Plan Manager Window](#)

[Open the Abstract Plan Manager](#)

[Abstract Plan Group Functions](#)

[Abstract Plan Manager Overview](#)

[Export a Group to a Table](#)

[Import an Abstract Plan for each User](#)

[Abstract Plan ID Functions](#)

[Use Saved Abstract Plans](#)

Why Save the Abstract Plan?

Saving the abstract plan for a SQL statement preserves how the SQL statement is executed. Therefore, when changes occur in the database environment, the query plan used to execute the SQL statement remains constant.

One major advantage of saving the abstract plan is that you can optimize the SQL statement without altering the SQL text. This is an ideal solution for when you do not have the source code from a vendor but want to improve the performance of the database application. The saving of the abstract plan can be very useful when you are considering making changes to the configuration parameters or you are migrating to another version of Adaptive Server and you want to preserve the current performance of SQL statements that are critical to the performance of your applications.

These database environment changes may include:

- Parallel degree
- Table partitioning
- Indexing
- Database software upgrades
- Changes in data columns

You can save an abstract plan from the following modules:

To save in...

See:

SQL Optimizer	Save Abstract Plan
SQL Scanner	Save Abstract Plan
Index Impact Analyzer	Right Pane for SQL Statements
Configuration Analyzer	Right Pane for SQL Statement
Migration Analyzer	Right Pane for SQL Statement
SQL Repository	SQL Repository Window

Related Topics

- [Abstract Plan Manager Overview](#)
- [Abstract Plan Group](#)
- [Abstract Plan Compatibility with Original SQL](#)
- [Abstract Plan Manager Window](#)
- [Open the Abstract Plan Manager](#)
- [Abstract Plan Group Functions](#)
- [Export a Group to a Table](#)
- [Import an Abstract Plan for each User](#)
- [Abstract Plan ID Functions](#)
- [Use Saved Abstract Plans](#)

Abstract Plan Compatibility with Original SQL

Adaptive Server generates an abstract plan based on the SQL syntax itself. If a SQL statement is transformed to another syntax that the Adaptive Server optimizer cannot transform internally, then the abstract plans are not compatible. Therefore, even though two SQL statements may be semantically equivalent, their abstract plans are not compatible.

For example, take this SQL statement as your original SQL statement:

```
SELECT A.CITY FROM A, B, C
WHERE A.CITY =B.CITY
AND B.CITY=C.CITY
```

And take this semantically equivalent alternative SQL:

```
SELECT A.CITY FROM A, B, C
WHERE A.CITY =B.CITY
AND B.CITY=C.CITY
AND A.CITY=C.CITY
```

The transform adds the new condition, A.CITY=C.CITY, and if the Adaptive Server optimizer cannot generate A.CITY=C.CITY internally for your original SQL, then an index scan from A.CITY to C.CITY or from C.CITY to A.CITY is not possible for your original SQL. So, the new abstract plan is not compatible for the original SQL.

Also, if an OR condition is transformed to a UNION, the new abstract plan is so different from the original SQL that there is no way for the original SQL to be compatible with the new abstract plan.

Related Topics

[Abstract Plan Manager Overview](#)

[Abstract Plan Group](#)

[Why Save the Abstract Plan?](#)

[Abstract Plan Manager Window](#)

[Open the Abstract Plan Manager](#)

[Abstract Plan Group Functions](#)

[Export a Group to a Table](#)

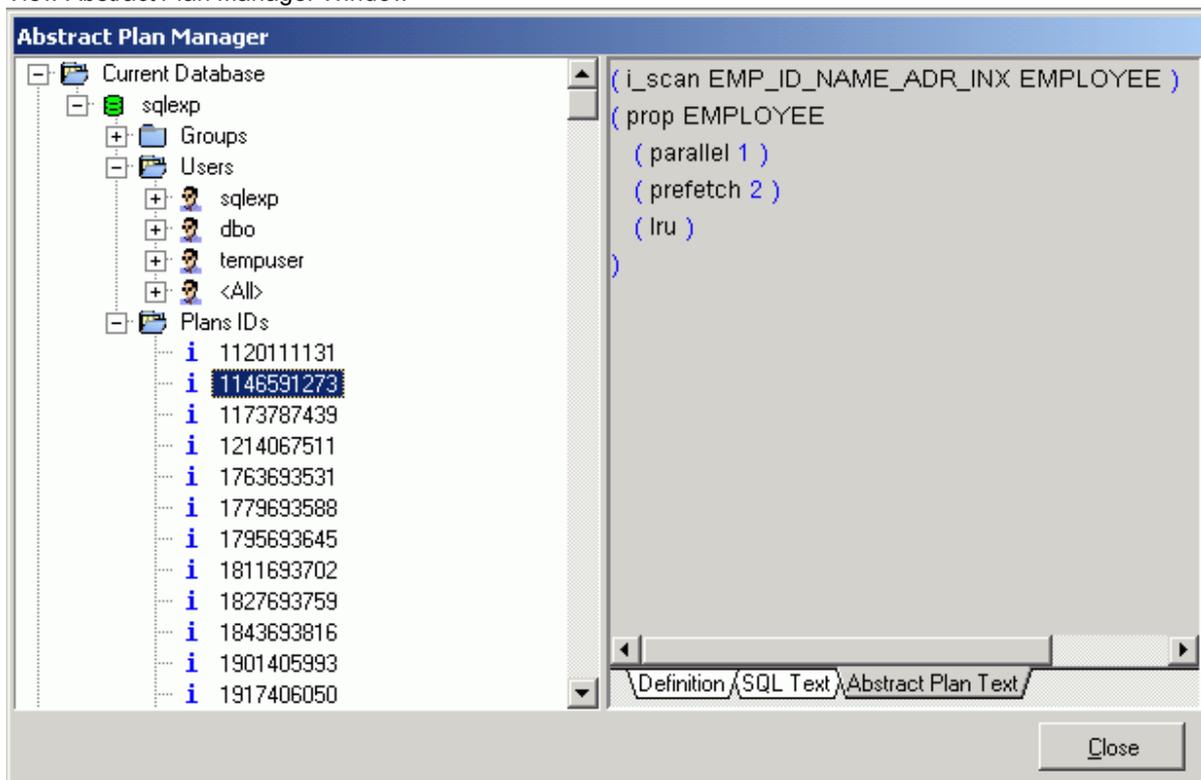
[Import an Abstract Plan for each User](#)

[Abstract Plan ID Functions](#)

[Use Saved Abstract Plans](#)

Abstract Plan Manager Window

View Abstract Plan Manager Window



Left Pane

The left pane is a tree structure for the managing of the elements of the abstract plan: Groups, Users, and Plan IDs.

Right-click any element to bring up the right-click menu with the functionality for that element.

Right Pane

The right pane displays the information that corresponds to the group, users, or plan ID that is selected in the left pane.

Related Topics

[Abstract Plan Manager Overview](#)

[Abstract Plan Group](#)

[Why Save the Abstract Plan?](#)

[Abstract Plan Compatibility with Original SQL](#)

[Open the Abstract Plan Manager](#)

[Abstract Plan Group Functions](#)

[Export a Group to a Table](#)

[Import an Abstract Plan for each User](#)

[Abstract Plan ID Functions](#)

[Use Saved Abstract Plans](#)

Open the Abstract Plan Manager

To open the Abstract Plan Manager

Click .

Related Topics

[Abstract Plan Manager Overview](#)

[Abstract Plan Group](#)

[Why Save the Abstract Plan?](#)

[Abstract Plan Compatibility with Original SQL](#)

[Abstract Plan Manager Window](#)

[Abstract Plan Group Functions](#)

[Export a Group to a Table](#)

[Import an Abstract Plan for each User](#)

[Abstract Plan ID Functions](#)

[Use Saved Abstract Plans](#)

Abstract Plan Group Functions

The following are the functions that are performed on abstract plan groups.

To use these functions

Right-click the group name in the left pane and select one of the following functions.

Item	Description
Create	Creates a new abstract plan group
Drop	Drops the select abstract plan group.
Purge	Deletes all abstract plans in the selected group
Copy All Plan	Copies all abstract plans in the selected group to another group.
Rename	Renames the abstract plan group.
Export	Exports the abstract plan group to a table.
Import	Imports an exported table to a database, user and group
Compare	Compares the differences between two groups.

Related Topics

[Abstract Plan Manager Overview](#)

[Abstract Plan Group](#)

[Why Save the Abstract Plan?](#)

[Abstract Plan Compatibility with Original SQL](#)

[Abstract Plan Manager Window](#)

[Open the Abstract Plan Manager](#)

[Abstract Plan Manager Overview](#)

[Export a Group to a Table](#)

[Import an Abstract Plan for each User](#)

[Abstract Plan ID Functions](#)

[Use Saved Abstract Plans](#)

Export a Group to a Table

In order to move the saved abstract plan from one user to another or from one database to other, you first export the group in which it is saved to a table in your database, and then you import the group to another user or database.

To export a group

1. In the left pane of the Abstract Plan Manager, right-click and select **Group | Export**.
2. In the Export Plan window under the Export from section, select the name of the database, user, and abstract plan group where you saved the abstract plan.
3. Under the Export to section, select the database and user. Enter a new table name to store the group in. You cannot use an existing table.

Related Topics

[Abstract Plan Manager Overview](#)

[Abstract Plan Group](#)

[Why Save the Abstract Plan?](#)

[Abstract Plan Compatibility with Original SQL](#)

[Abstract Plan Manager Window](#)

[Open the Abstract Plan Manager](#)

[Abstract Plan Group Functions](#)

[Import an Abstract Plan for each User](#)

[Abstract Plan ID Functions](#)

[Use Saved Abstract Plans](#)

Import an Abstract Plan for each User

Once the abstract plan group is exported to a table in your database, you can import it to as many users or databases as you would like.

To import an abstract plan for each user

1. In the left pane of the Abstract Plan Manager, right-click and select **Group | Import**.
2. In the Import Plan window under the Import from section, select the database, user and table name where you exported the group.
3. Under the Import to section, select the name of the database, user, and abstract plan group where you want to import the abstract plans.

Related Topics

[Abstract Plan Manager Overview](#)

[Abstract Plan Group](#)

[Why Save the Abstract Plan?](#)

[Abstract Plan Compatibility with Original SQL](#)

[Abstract Plan Manager Window](#)

[Open the Abstract Plan Manager](#)

[Abstract Plan Group Functions](#)

Abstract Plan ID Functions

The following are the functions that are performed on abstract plan IDs.

To use these functions

Right-click the plan ID in the left pane and select the function.

Item	Description
Drop	Drops the selected plan.
Copy	Copies selected plan to another group
Edit	Brings up the Edit Plan window so you can modify the abstract plan.
Find Abstract Plan with Text	Searches for a keyword in the SQL text and abstract plan.
Find Abstract Plan with ID	Searches for a specific plan ID

To use the Abstract Plans for specific users, you enable the use of the Abstract Plan for a specific group on a session level. Therefore, if you have saved Abstract plans in a specific group and you want to use them, you need to set Adaptive Server to use the Abstract Plans in that group. In your application you need to add the following commands:

Note: If the abstract plan is used by many users, you must have already imported the Abstract Plan for each user.

Viewing the text of the SQL statement

When you are viewing the Plan ID, you can view the Definition, the SQL Text, or the Abstract Plan Text by select the corresponding tab. When you are viewing SQL Text, you can format the text by selecting the **Format SQL** checkbox or the **Word Wrap** checkbox.

Related Topics

- [Abstract Plan Manager Overview](#)
- [Abstract Plan Group](#)
- [Why Save the Abstract Plan?](#)
- [Abstract Plan Compatibility with Original SQL](#)
- [Abstract Plan Manager Window](#)
- [Open the Abstract Plan Manager](#)
- [Abstract Plan Manager Overview](#)

[Export a Group to a Table](#)
[Import an Abstract Plan for each User](#)
[Use Saved Abstract Plans](#)

Use Saved Abstract Plans

To use the abstract plan in your applications you have to either enable the abstract plan loading in the server-wide mode or enable the Abstract Plan loading for the session-level inside your application

The abstract plans that you want all applications to use must be saved in the *ap_stdin* group. If you want different abstract plans for different applications, you must save them in a group and associate it with a user.

Using Abstract Plans for Specific Applications

To use the Abstract Plans for specific users, you enable the use of the Abstract Plan for a specific group on a session level. Therefore, if you have saved abstract plans in a specific group and you want to use them, you need to tell Adaptive Server to use the abstract plans in that group. In your application you need to add the following commands:

Note: If the abstract plan is used by many users, you must have already imported the abstract plan for each user.

Example:

```
set plan load group_name on
go
*some queries*
go
set plan load off
go
```

Using the Abstract Plans Server Wide

You can enable Adaptive Server to use the abstract plans server-wide from the default group of *ap_stdin*.

To use the same abstract plan for all applications, the system administrator must set the abstract plan load *sp_configure* parameter to 1. Adaptive Server then uses the stored abstract plans from the default group *ap_stdin*.

Note: You can override the use of the abstract plan from *ap_stdin* for a specific SQL statement with the session-level "set plan load" command in the application.

Related Topics

[Abstract Plan Manager Overview](#)
[Abstract Plan Group](#)
[Why Save the Abstract Plan?](#)
[Abstract Plan Compatibility with Original SQL](#)
[Abstract Plan Manager Window](#)
[Open the Abstract Plan Manager](#)
[Abstract Plan Group Functions](#)

[Export a Group to a Table](#)
[Import an Abstract Plan for each User](#)
[Abstract Plan ID Functions](#)

User-Defined Temp Table Overview

The technique of creating temporary tables to extract data from permanent tables is often used. Because the existence of the local temp table is session related, you need to create these temporary tables in your current session. The User-Defined Temp Table window allows you to create temp tables to use so that you may optimize and analyze SQL statements that use temporary tables. Besides temporary table that you create using the User-Defined Temp Table window, temporary table may also be created during the scanning by the SQL Scanner and during the optimization process by the SQL Optimizer.

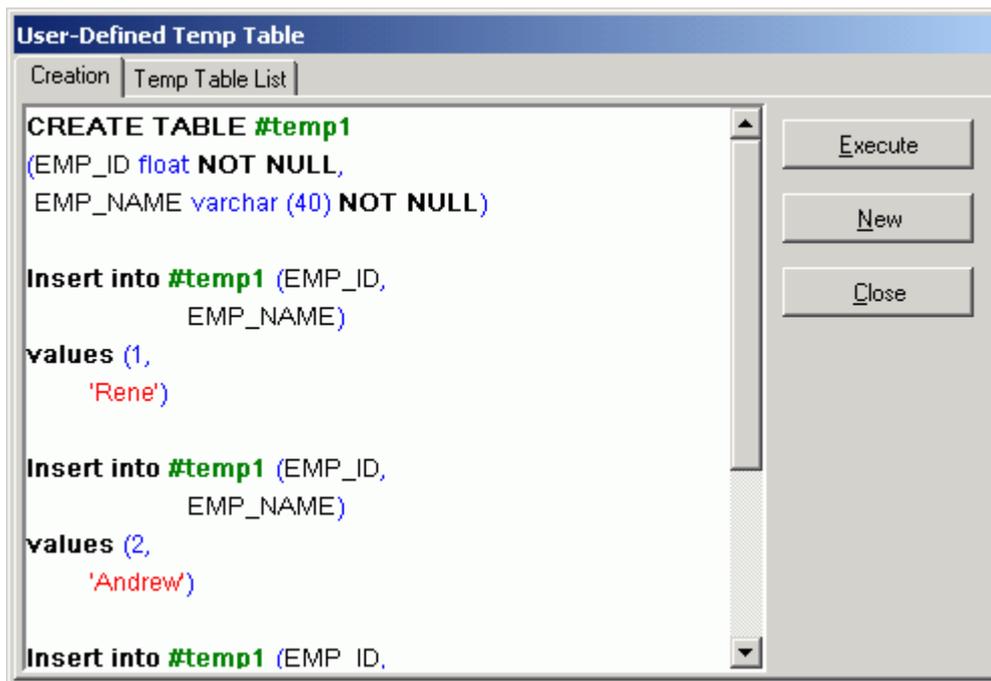
Note: Temporary Tables created in the User-Defined Temp Table module are not available in the SQL Worksheet module since a new database session is used to execute commands in the SQL Worksheet.

Related Topics

[Create Temporary Tables](#)
[Parameters in Temporary Table SQL](#)
[View SQL scripts of temporary tables](#)
[Drop Temporary Tables](#)
[Example of Temporary Tables in SQL Scanner](#)
[Copy SQL with Temporary Tables to SQL Optimizer](#)
[Create Alternative SQL Which Uses Temporary Tables](#)
[Preference Settings for Handling Temporary Tables](#)

Create Temporary Tables

[View User-Defined Temp Table Window](#)



Temporary tables only exist in your current session. In order to optimize a SQL statement that uses temporary tables, these tables need to be created in this session before you optimize. The User-Defined Temp Table window allows the creation of temporary tables to be used throughout modules that use the original sessions that is created as you logon.

Temporary tables created in the User-Defined Temp Table module are used in the SQL Optimizer and SQL Scanner and can be viewed in the Database Explorer. An icon displays in the status bar of the main window to indicate at least one temporary table was created under the current session.

Temporary tables are dropped when you

- Exit from the program
- Reconnect with the same or a different user logon
- Drop it using the User-Defined Temp Table module

To create temporary tables

1. Click .
2. Enter the SQL for creating the temporary table on the **Creation** tab. Multiple commands can be entered.
3. Click **Execute**.

After the temporary table is created you will notice an icon  on the bottom right of the main window status. This icon disappears when all user-defined temporary tables are dropped.

The User-Defined Temp Table module only supports SQL statements that create or modify temporary tables; these are:

- SELECT INTO
- CREATE TABLE
- CREATE INDEX
- DROP INDEX
- DROP TABLE
- INSERT
- UPDATE
- DELETE

Note: Temporary tables created within the User-Defined Temp Table window can be used in all modules but the SQL Worksheet. The SQL Worksheet module creates a separate session so temporary tables created in the User-Defined Temp Table cannot be used in the SQL Worksheet and vice versa.

Related Topics

[User-Defined Temp Table Overview](#)

[Parameters in Temporary Table SQL](#)

[View SQL scripts of temporary tables](#)

[Drop Temporary Tables](#)

[Example of Temporary Tables in SQL Scanner](#)

[Copy SQL with Temporary Tables to SQL Optimizer](#)

[Create Alternative SQL Which Uses Temporary Tables](#)

[Preference Settings for Handling Temporary Tables](#)

Parameters in Temporary Table SQL

In the User-Defined Temp Table window, if the commands for creating the temporary table contain parameters, you are prompted to enter the data types and values on the [Parameters window](#) after you click **Execute**.

Related Topics

[User-Defined Temp Table Overview](#)

[Create Temporary Tables](#)

[View SQL scripts of temporary tables](#)

[Drop Temporary Tables](#)

[Example of Temporary Tables in SQL Scanner](#)

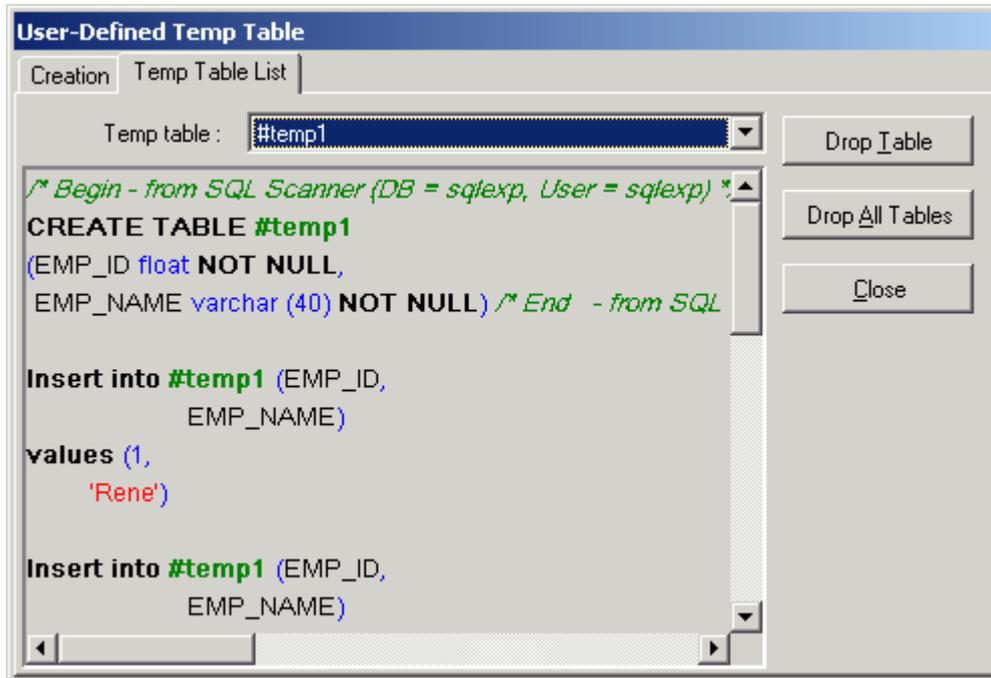
[Copy SQL with Temporary Tables to SQL Optimizer](#)

[Create Alternative SQL Which Uses Temporary Tables](#)

[Preference Settings for Handling Temporary Tables](#)

View SQL scripts of temporary tables

View User-Defined Temp Table Window



After the user-defined temporary tables are created, you can view the SQL scripts used to create and populate the tables.

To view a script

1. Click .
2. Select the Temp Table List tab.
3. Select the temporary table from the **Temp Table** drop down list.

Related Topics

[User-Defined Temp Table Overview](#)

[Create Temporary Tables](#)

[Parameters in Temporary Table SQL](#)

[Drop Temporary Tables](#)

[Example of Temporary Tables in SQL Scanner](#)

[Copy SQL with Temporary Tables to SQL Optimizer](#)

[Create Alternative SQL Which Uses Temporary Tables](#)

[Preference Settings for Handling Temporary Tables](#)

Drop Temporary Tables

After the User-Defined temporary tables are created, you can drop them.

To drop temporary tables

1. Click .
2. Select the Temp Table List tab.
3. If you want to remove all temp tables, click **Drop All Table**.

Related Topics

[User-Defined Temp Table Overview](#)

[Create Temporary Tables](#)

[Parameters in Temporary Table SQL](#)

[View SQL scripts of temporary tables](#)

[Example of Temporary Tables in SQL Scanner](#)

[Copy SQL with Temporary Tables to SQL Optimizer](#)

[Create Alternative SQL Which Uses Temporary Tables](#)

[Preference Settings for Handling Temporary Tables](#)

Example of Temporary Tables in SQL Scanner

Example with CREATE TABLE in the source code containing the following SQL statements:

```
/* SQL1 */
```

```
CREATE TABLE #temp1  
([EMP_ID] float NOT NULL,  
[EMP_NAME] varchar (40) NOT NULL)
```

```
/* SQL2 */
```

```
Insert into #temp1 (EMP_ID, EMP_NAME)  
values (1, 'Rene')
```

```
/* SQL3 */
```

```
Insert into #temp1 (EMP_ID, EMP_NAME)  
values (2, 'Andrew')
```

```
/* SQL4 */
```

```
Insert into #temp1 (EMP_ID, EMP_NAME)
```

```
values (3, 'Claudia')
```

```
/* SQL5 */
```

```
Select * from #temp1
```

In the above example, when the SQL Scanner finds the first INSERT statement, it executes the CREATE TABLE statement, so that the temporary table #temp1 is created. The query plan for the INSERT statement is obtained and the statement is classified as to how likely it is to be causing a performance problem. In the SQL Information Pane of the SQL Scanner window, the Scanner Temp Table pane displays the SQL statements used to create the temporary table. At the end of scanning process, the temporary table is dropped unless you have set the **Override User-Defined Temp Table** option.

In the Preferences, an option is provided to include the data in the temporary table. When you select the **Include data** option, the INSERT, UPDATE and DELETE statements that populate the temporary table are also executed. These statements are displayed along with the CREATE TABLE command in the Information Pane in the SQL Scanner window.

From the example above, the SQL Scanner window shows the following for SQL5 (select * from #temp1) under the **Scanner Temp Table** button. It shows the SQL statement used to create the temporary table and the data that is inserted into it during the scanning process.

```
/* with Included data option selected in Preferences */
```

```
CREATE TABLE #temp1
```

```
([EMP_ID] float NOT NULL,
```

```
[EMP_NAME] varchar (40) NOT NULL)
```

```
Insert into #temp1 (EMP_ID, EMP_NAME)
```

```
values (1,'Rene')
```

```
Insert into #temp1 (EMP_ID, EMP_NAME)
```

```
values (2,'Andrew')
```

```
Insert into #temp1 (EMP_ID, EMP_NAME)
```

```
values (3,'Claudia')
```

Example using SELECT INTO to create the temporary table.

After scanning, the SQL Scanner window shows the following two statements:

```
/* SQL1 */
```

```
select EMP_ID,
```

```
EMP_NAME
```

```
into #temp1
```

```
from EMPLOYEE
```

```
/* SQL2 */
```

```
select *
```

```
from #temp1
```

For SQL2, the Scanner Temp Table pane shows that the SELECT INTO SQL statement was used to create the temporary table.

```
select EMP_ID,
```

```
EMP_NAME
```

```
into #temp1
```

```
from EMPLOYEE
```

Related Topics

- [Options \(SQL Scanner Tab\)](#)
- [User-Defined Temp Table Overview](#)
- [Create Temporary Tables](#)
- [Parameters in Temporary Table SQL](#)
- [View SQL scripts of temporary tables](#)
- [Drop Temporary Tables](#)
- [Copy SQL with Temporary Tables to SQL Optimizer](#)
- [Create Alternative SQL Which Uses Temporary Tables](#)
- [Preference Settings for Handling Temporary Tables](#)

Copy SQL with Temporary Tables to SQL Optimizer

When you copy a SQL statement from the SQL Scanner to the SQL Optimizer module, the [Send to SQL Optimizer](#) function automates this process for you. It copies the SQL statement, opens the SQL Optimizer window and inserts the SQL statement in the SQL Editor pane. When the SQL statement you are copying also includes a temporary table, the Send to SQL Optimizer function also copies the SQL statements for creating and populating the temporary table to the User-Defined Temp Table module.

Two items need to be considered when the SQL statements are automatically copied in the User-Defined Temp Table module. Is there SQL text already in the module? And does the temporary table already exist with the same or different data?

If SQL text is already entered into the User-Defined Temp Table module, you are prompted to save the current SQL to an ASCII file before it is overlaid with the new text. Saving the text enables you to use it again.

If a table with the same name already exists and you would like to recreate it with either a new definition or different data, then you must drop the table before executing the new SQL text. You can drop temporary table from the Temp Table List tab in the User-Defined Temp Table module.

Related Topics

- [User-Defined Temp Table Overview](#)
- [Create Temporary Tables](#)
- [Parameters in Temporary Table SQL](#)
- [View SQL scripts of temporary tables](#)
- [Drop Temporary Tables](#)
- [Create Alternative SQL Which Uses Temporary Tables](#)
- [Preference Settings for Handling Temporary Tables](#)

Create Alternative SQL Which Uses Temporary Tables

One of the techniques for improving the performance of a SQL statement is to create a temporary table. When the SQL Optimizer rewrites the original SQL statement, some of the alternatives may involve this technique.

Here is an illustration of the original SQL and one of the SQL alternatives.

Original SQL

```
SELECT *
FROM A
WHERE A.KEY IN (SELECT B.KEY
FROM B)
```

SQL1 Alternative

```
SELECT DISTINCT COL1 = B.KEY INTO #TEMP1 FROM B
SELECT *
FROM A, #TEMP1
WHERE A.KEY = #TEMP1.COL1
DROP TABLE #TEMP1
```

Note: Under the Optimization tab in the *Preferences*, the section on [Temp Table Generation](#) enables you to specify whether to allow the generation of temporary tables during the optimization process.

Related Topics

[User-Defined Temp Table Overview](#)

[Create Temporary Tables](#)

[Parameters in Temporary Table SQL](#)

[View SQL scripts of temporary tables](#)

[Drop Temporary Tables](#)

[Example of Temporary Tables in SQL Scanner](#)

[Copy SQL with Temporary Tables to SQL Optimizer](#)

[Preference Settings for Handling Temporary Tables](#)

Preference Settings for Handling Temporary Tables

Under the Preferences window, the SQL Scanner and Optimization tabs contain several settings that control the use of temp tables in the SQL Scanner and SQL Optimizer modules.

SQL Scanner Tab

- [Create Scanner Temp Table](#)
- [Include data](#)
- [Override previous Scanner Temp Table](#)
- [Override User-Defined Temp Table](#)

Optimization Tab

- [Temp table generation](#)
- [Apply selected forces to temp table](#)

Related Topics

[User-Defined Temp Table Overview](#)

[Create Temporary Tables](#)

[Parameters in Temporary Table SQL](#)

[View SQL scripts of temporary tables](#)

[Drop Temporary Tables](#)

[Example of Temporary Tables in SQL Scanner](#)

[Copy SQL with Temporary Tables to SQL Optimizer](#)

[Create Alternative SQL Which Uses Temporary Tables](#)

SQL History Overview

Throughout the program, SQL statements are saved in the SQL History so that you can use them again. They are stored in a file so that they are always available to work with again.

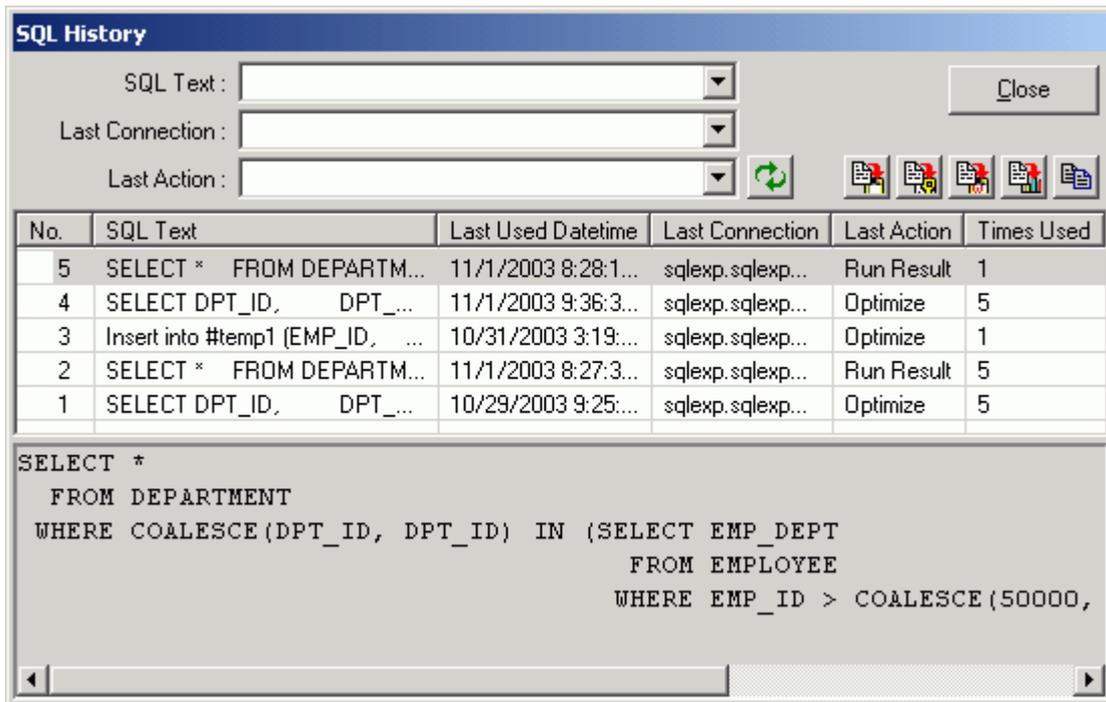
Related Topics

[SQL History Window](#)

[SQL History Functions](#)

SQL History Window

[View SQL History Window](#)



The SQL History window displays the SQL statements that were used in several modules. The functions that can save SQL are Optimize, Advise Indexes, Run Time, Run Result, Show Plan and Execute. The Preferences settings determine in which functions the SQL statements are saved.

SQL Text drop-down field

Text to search for filtering SQL statements displayed in the grid.

Last Connection drop-down field

List of each unique connection for the SQL statements saved in the history list.

Last Action drop-down field

List of each unique action type for the SQL statements saved in the history list.

Apply Filter Button

Narrows the SQL statements displayed in the grid according to the selections in the SQL Text, Last Connection, and Last Action drop-down boxes.

Button Bar

Buttons for copying the selected SQL statement to other modules: Send to SQL Optimizer, Send to Index Advisor, Save SQL to SQL Repository, Copy to SQL Worksheet, and Copy SQL and Close Window.

Grid

Item	Description
No	The SQL statements are numbered as they are saved in sequential order.
SQL Text	The text of the SQL statement. When you click a line, the full text of the SQL statement displays in the bottom pane of the window.
Last Used Datetime	The date and time the SQL was last used in one of the modules.
Last Connection	The connection information for the last time the SQL statement was used.
Last Action	The action that was used which caused the SQL statement to be saved.
Times Used	The number of times the SQL statement has been saved in the SQL History.

Related Topics

[SQL History Overview](#)

[Filter SQL Statements](#)

[Recall a SQL Statement](#)

[SQL History Functions](#)

Filter SQL Statements

The SQL statements can be filtered to help you locate the specific SQL statement that you are looking for. Three drop-down boxes provide you with the filtering options.

To filter the display of the SQL statements

1. Select the filtering criteria from one or more of the list:

List	Description
SQL Text	Enter or select text to filter the SQL statements displayed in the grid.
Last Connection	Select the connection that you want used to filter the SQL statements displayed in the grid. This drop-down list contains a list of the unique connections in the SQL History list.

Last Action

Select the action that you want used to filter the SQL statements displayed in the grid. This drop-down list contains a list of the actions for the SQL statements in the SQL History list.

2. Click  in the SQL History window.

Related Topics

[SQL History Overview](#)

[SQL History Window](#)

Recall a SQL Statement

SQL statements used in several functions are saved so that you can use them again.

To recall a SQL statement

1. Click .
2. Select the SQL statement that you want to use.
3. Click one of the following buttons:

Button	Function
	Send to SQL Optimizer
	Send to Index Advisor
	Copy to SQL Worksheet
	Save SQL to SQL Repository
	Copy SQL and Close Window

Related Topics

[SQL History Overview](#)

[SQL History Window](#)

SQL History Functions

Below is a list of available functions within the SQL History window.

Button	Function
	Apply Filter
	Send to SQL Optimizer
	Send to Index Advisor
	Copy to SQL Worksheet
	Save SQL to SQL Repository
	Copy SQL and Close Window

Related Topics

[SQL History Overview](#)

[SQL History Window](#)