SQL Optimizer for IBM® DB2® LUW 5.0

User Guide

Contents

About Quest SQL Optimizer for IBM® DB2® LUW **12**
About Product Improvement Program 12
Getting Started 13
User Logon Screen 13
Access Plan Table 13
Connect to the Database 14
Loading information from the Data Dictionary 14
User Logon Privileges 14
Temporary Table Generation 14
Index Expert Function 14
SQL Optimizer Module 14
Special Registers Settings 15
Synchronize the Data Dictionary 15
Enter Authorization Key 16
Select Database Language 16
Status Bar—Main Window 17
Toolbar 17
Options 17
About Options 17
SQL Rewrite Settings 19
SQL Rewrite Settings 19
Intelligence 19
SQL Rewrite (1) 20
SQL Rewrite (2) 22
SQL Options 25
Quota 27
General 28
Editor Settings 29
General 29
Editing 30
Lookup 30
Activity Log Settings 31
Activities to be logged 31
Information to be logged 32
Housekeeping 32
Plan Settings 32
SQL Scanner Settings 33
SQL Scanner Settings 33
Options (SQL Scanner Tab) 34
General (SQL Scanner Tab) 35
SQL Classification Settings 37

SQL Classification Settings ... 37
SQL Statement with no Access Plan ... 39
Directory Setup ... 40
Linkage Settings ... 40
General Settings ... 41
General Settings ... 41
General ... 41
Message ... 44
Result Set ... 45
Generate Indexes Settings ... 45
Generate Indexes Settings ... 45
Intelligence ... 45
Index Options ... 46
General ... 48
SQL Scanner ... 49
SQL Scanner Concepts ... 49
About SQL Scanner ... 49
Embedded and Dynamic SQL Statements ... 50
What does the SQL Scanner scan? ... 51
Scanned SQL Statement Classification ... 51
Data Directory Setting ... 52
About SQL Scanner Conversions ... 53
Trigger ... 53
Scanned SQL ... 54
After conversion ... 54
Original SQL statement ... 54
After conversion ... 55
Original SQL statement ... 55
After conversion ... 55
Original SQL statement ... 56
After conversion ... 56
Original SQL statement ... 56
After conversion ... 57
Job Manager ... 57
About Job Manager ... 57
Job Manager Window ... 57
Group Manager Window ... 59
Open the SQL Scanner ... 59
Add Scanner Jobs ... 60
Scan ... 62
Scan Source Code with Temporary Tables ... 62
Abort Scan ... 63
Find a Job ... 63
View Group Summary ... 63
View Group Properties ... 64
View Job Properties ... 64

Open Group Manager ... 65
Check Scanned SQL ... 65
Mark and Unmark Jobs ... 67
Delete Marked Jobs ... 68
Move or Copy Jobs to Another Group ... 68
Modify Job Details ... 69
Switch Schema ... 69
Place Bookmarks in the Job Manager Window ... 69
Create Benchmark Factory Import File ... 70
Job Manager Functions ... 70
Scanned SQL Viewer ... 71
About Scanned SQL Viewer ... 71
Scanned SQL Viewer Window ... 72
Understand the Access Plan ... 73
Open Scanned SQL Viewer ... 73
View a Particular Type of SQL Statement ... 73
Job Navigation ... 74
Generate Report for Scanned SQL Statements ... 75
Create Benchmark Factory Import File ... 75
Scanned SQL Viewer Functions ... 75
SQL Optimizer ... 76
SQL Optimizer Concepts ... 76
About the SQL Rewrite Function ... 76
What Function Should I Use to Retrieve the Run Time? ... 77
Unsatisfactory Performance Results ... 77
SQL Functions ... 77
SQL Optimizer Functions ... 78
SQL Optimizer Window ... 79
SQL Optimizer Window ... 79
SQL Tab ... 81
Access Plan Page ... 90
Plan Statistics Tab ... 104
Summary Tab ... 107
Compare Tab ... 109
Execution Result Tab ... 110
Use the SQL Optimizer ... 112
Enter or Edit the Original SQL Statement ... 112
Automatically Rewrite the Original SQL Statement ... 113
Rewrite Details ... 114
Add User-Defined SQL Alternatives ... 115
Add Virtual Index Alternatives ... 116
Generate a Report for Optimized SQL ... 120
Verify the Correctness of Rewritten SQL ... 120
Test for Scalability ... 121
Create Benchmark Factory Import File ... 121
Batch Run ... 121
Selecting SQL statements to test ... 126

Changing the execution order of alternatives ... 127
Always run Original SQL first ... 127
Terminate test run if the elapsed time is greater than that of ... 128
Percentage Delay ... 129
Run without Termination ... 129
Batch Run Termination Criteria ... 130
Run Time Mode ... 131
Repeat the test run by executing ... 132
Schedule setting ... 133
Start ... 133
Until ... 133
Save Optimized SQL ... 134
Optimization Information ... 134
Last Saved Access plan Information ... 134
Optimization Information ... 136
Last Saved Access Plan Information ... 137
Refresh Plan button ... 137
SQL Formatter ... 138
About SQL Formatter ... 138
SQL Formatter Window ... 138
Original SQL (Left pane) ... 139
Formatted SQL (Right pane) ... 139
Format a SQL Statement ... 139
Host Variables within SQL Formatter ... 140
SQL Formatter Functions ... 140
SQL Inspector ... 140
SQL Inspector Overview ... 140
Data Directory Setting ... 141
Open SQL Inspector ... 141
SQL Inspector Window ... 141
Inspector List pane ... 141
Statistics pane ... 142
SQL Text pane ... 142
Status Bar Information ... 142
Statistics: Executed SQL ... 142
Statistics: Executing SQL ... 143
Add Inspector ... 144
Add Inspector ... 144
General Information ... 144
Collection Period ... 144
Collection Time ... 145
SQL Filter ... 145
Monitoring Period ... 145
Target Time ... 145

Target Applications and Users ... 146
Inspect ... 146
Ad hoc Inspect ... 147
Abort Inspect ... 147
Scan Inspector ... 147
Delete Inspector ... 148
Find Inspector ... 148
View Inspector Properties ... 149
Place Bookmarks in SQL Inspector Window ... 149
Create Benchmark Factory Import File ... 149
Sum Statistics for a Stored Procedure ... 150
SQL Inspector Functions ... 150
Database Explorer ... 151
About Database Explorer ... 151
Database Explorer Window ... 151
Database Objects ... 151
Object Information ... 151
Privileges ... 152
Open Database Explorer ... 152
Sort Criteria ... 152
Modify Data ... 152
Copy Column Names to Another Window ... 153
Copy Data from Cells, Rows, or Columns ... 154
View LONG, CLOB and BLOB Columns ... 154
Scan Database Objects ... 154
Database Explorer Functions ... 155
SQL Repository ... 155
About SQL Repository ... 155
SQL Repository Window ... 155
SQL Listing (Left pane) ... 156
SQL Listing Details (Top right pane) ... 156
SQL Information Details (Bottom right pane) ... 156
Add SQL to SQL Repository ... 157
Add SQL Wizard: General Page ... 157
Add SQL Wizard: SQL Information Page ... 158
SQL Text (Left pane) ... 158
Information Pane (Right Pane) ... 158
Check SQL button ... 158
Refresh SQL Access Plan ... 158
Modify SQL ... 159
Rename SQL ... 159
Delete SQL ... 159
Create Benchmark Factory Import File ... 160
SQL Repository Functions ... 160

Index Impact Analyzer ... 160
About Index Impact Analyzer ... 160
Index Impact Analyzer Window ... 161
Index Impact Analyzer Window ... 161
Right Pane for Analyzers ... 162
Right Pane for Scenarios Analyzed Folder ... 162
Right Pane for Scenario ... 163
Right Pane for SQL Analyzed Folder ... 164
Right Pane for SQL Statements ... 164
Create an Index Impact Analyzer ... 165
New Analysis Wizard: Analyzer Page ... 165
Select a new or existing Analyzer ... 165
New Analysis Wizard: Select SQL Page ... 166
Select SQL to check for performance changes ... 166
SQL access plans to be analyzed ... 166
New Analysis Wizard: Index Page ... 166
Access plans will be grouped under a Scenario ... 166
Add Index Window ... 167
View Index Impact Analysis Details ... 168
Add SQL to an Index Impact Analysis ... 168
Add Scenario ... 169
Modify Index Impact Analyzer ... 169
Regenerate ... 169
Rename an Index Impact Analyzer ... 170
Delete an Index Impact Analysis ... 170
Index Impact Analyzer Functions ... 170
Index Usage Analyzer ... 171
About Index Usage Analyzer ... 171
Index Usage Analyzer Window ... 171
Index Usage Analyzer Window ... 171
Right Pane for Analyzers ... 173
Right Pane for Tables Analyzed ... 173
Right Pane for Analyzed Table Information ... 174
Right Pane for SQL Analyzed Folder ... 174
Right Pane for SQL Statements ... 175
Create an Index Usage Analysis ... 175
Update an Index Usage Analysis ... 175
New Analysis Wizard: Analyzer Page ... 176
New Analysis Wizard: Select SQL Page ... 176
Select SQL from ... 176
SQL Tree ... 176
View Index Usage Analysis Details ... 177
Modify an Index Usage Analysis ... 177
Rename an Index Usage Analysis ... 177
Delete an Index Unused Analysis ... 178

Index Usage Analyzer Functions ... 178
User-Defined Temp Table ... 178
About User-Defined Temp Table ... 178
Privileges for Creating User-Defined Temp Tables ... 179
Create Temporary Tables ... 179
View SQL Scripts of Temporary Tables ... 180
Drop Temporary Tables ... 180
Copy SQL with Temporary Tables to SQL Optimizer ... 181
Preference Settings for Handling Temporary Tables ... 181
SQL Scanner Tab - Options Button ... 182
Optimization Tab - Optimization Button ... 182
Editor Functions ... 182
About Editor Functions ... 182
Auto Correction ... 183
Brackets Pairing ... 184
Comment and Uncomment Text ... 185
Customize Editable Panes ... 185
Editor Panes Shortcuts ... 186
Indent and Outdent Text ... 186
Member and Argument Lookup ... 187
Syntax Highlight ... 188
Element options ... 188
Foreground color ... 189
Background color ... 189
Editor options ... 189
Reset Default button ... 189
Uppercase and Lowercase Text ... 189
SQL Functions ... 190
About SQL Functions ... 190
Retrieve Access Plan ... 190
Retrieve Run Result ... 190
Commit or Rollback ... 191
For the original SQL statement ... 191
For the SQL alternatives ... 191
New Database Session ... 191
Retrieve Run Time ... 191
191
First Record ... 191
192
All Records ... 192
192
New Database Session ... 192
Test for Scalability ... 192
Convert Parameter Markers ... 193
SQL Information and Functions ... 193

SQL Information Pane ... 193
Send SQL to the SQL Rewrite Function ... 195
Send SQL to the Generate Indexes Function ... 196
Open SQL from SQL Repository ... 196
Save SQL to SQL Repository ... 197
Supported SQL Statements ... 197
Create Benchmark Factory Import File ... 197
SQL Navigation ... 197
Find a SQL Using a Text String ... 198
Comments ... 198
Variables ... 199
Parameters Window ... 199
Filter Database Objects ... 200
Access Plan ... 201
Access Plan ... 201
Review Access Plans ... 202
Access Plan Options ... 203
Access Plan Functions ... 204
Plan Detail ... 204
Plan Help ... 205
Animate Access Plans ... 205
Copy Access Plans ... 205
DB2 Optimized Text ... 205
Search Functions ... 206
Search Functions ... 206
Find ... 206
Quick Find ... 207
Find Next ... 207
Replace ... 207
Goto Line ... 208
Regular Expression ... 208
Activity Log ... 210
About Activity Log ... 210
View Activity Log Report ... 210
User Information ... 210
Activity Information ... 211
Purge Activity Log Data ... 211
Start Recording Activities ... 212
Stop Recording Activities ... 212

Tutorials ... 213
SQL Scanner Tutorial ... 213
Open a Scanner Group ... 213
Add Scanner Jobs ... 213
Scan Jobs ... 214

View scanning results 214
SQL Optimizer Tutorial 215
Rewrite the SQL statement 215
Create index-set candidates 217
Batch test SQL alternatives 217
Review test results 217
SQL Formatter Tutorial 218
Formatting a SQL Statement 218
SQL Inspector Tutorial 218
User-Defined Temp Tables Tutorial 219
Working with Temporary Tables 219
SQL Repository Tutorial 219
Adding SQL to the SQL Repository 220
Save SQL to the SQL Repository from Other Modules Tutorial 220
Generate Virtual Indexes Tutorial 221
Create index-set candidates 221
Test index sets 221
Index Impact Analyzer Tutorial 222
Analyzer Tab 222
Select SQL Tab 222
Index Tab 223
Reviewing Index Impact Analysis Results 223
Index Usage Analyzer Tutorial 223
Analyzer Tab 224
Select SQL Tab 224
Reviewing Index Usage Analyzer Results 224
Database Explorer Tutorial 224
Browsing Database Objects 225

About us 226
Contacting Quest 226
Technical support resources 226

Copyright 2025 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 (https://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 https://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 https://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.

SQL Optimizer User Guide
Updated - 16 Oct, 2025
Version - 5.0

About Quest SQL Optimizer for IBM® DB2® LUW

Quest SQL Optimizer for IBM® DB2® LUW maximizes SQL performance by automating the manual, time-intensive and uncertain process of ensuring that SQL statements are performing as fast as possible. SQL Optimizer analyzes, rewrites, and evaluates SQL statements within multiple database objects, files. With SQL Optimizer, you can analyze and optimize all your problem SQL from multiple sources. SQL Optimizer also provides you a complete index optimization and plan change analysis solution, from index recommendations to simulated index impact analysis, through comparison of multiple SQL access plans.

SQL Optimizer provides you with the following main modules.

SQL Optimizer (including SQL Rewrite and Generate Indexes functions)

SQL Formatter

Database Explorer

SQL Scanner

SQL Inspector

SQL Repository

Index Impact Analyzer

Index Usage Analyzer

About Product Improvement Program

To prioritize enhancements in future releases, SQL Optimizer for IBM® DB2® LUW 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**

Getting Started

User Logon Screen

When you start SQL Optimizer, a User Logon dialog box is displayed to allow connection to the DB2 database.

To logon to DB2 LUW

1. In the Login Name box, enter the user name required to connect to the database.
2. In the Password box, enter the password associated with the user name.
3. In the Database Alias box, enter the name of the database to which you want to connect. Database aliases can be defined in DB2 Client Configuration Assistant.
4. Click **Connect** to connect to the database server. After the database is successfully connected, information from the data dictionary is loaded into memory.

In DB2 LUW version 7 or earlier, if the current logon user does not satisfy all of the logon privileges, an icon will appear on the bottom status bar and the Connection Information dialog box will be displayed. Full details of what database privileges are required and their workaround are displayed on the Connection Information dialog box.

To view the Connection Information dialog box

Select **Database | View Connection Information**.

Access Plan Table

When SQL Optimizer connects to the database, it determines whether the access plan tables exist. DB2 requires access plan tables to provide access plan detail. If these tables do not exist in the current schema, SQL Optimizer attempts to create the following tables according to the table definitions in the SQL script, C:\Program Files\SQLLIB\misc\EXPLAIN.DDL, provided by DB2 LUW.

ADVICE_WORKLOAD
ADVICE_INDEX
EXPLAIN_OPERATOR
EXPLAIN_OBJECT
EXPLAIN_ARGUMENT
EXPLAIN_PREDICATE
EXPLAIN_STREAM
EXPLAIN_INSTANCE
EXPLAIN_STATEMENT

Related Topic

Connect to the Database

Connect to the Database

Prior to installing SQL Optimizer, a client/server system needs to be set up. Set up the host strings in DB2 Client Configuration Assistant to connect to the databases.

SQL Optimizer directly uses the DB2 Client Configuration Assistant to connect to the DB2 LUW database.

Loading information from the Data Dictionary

Information about the tables and indexes from the data dictionary needs to be loaded into the memory of the computer for every connection to a database. The Load data dictionary after database connection option determines whether information is loaded when the connection is made or whether the specific information needed from the database is loaded when a SQL statement is parsed for functions such as scanning, optimization, and index generation.

Related Topic

Synchronize the Data Dictionary

User Logon Privileges

Every database has some kind of database privileges. These privileges are directly reflected in SQL Optimizer to limit the access and authority of each user. When logging on to SQL Optimizer, if you do not satisfy the logon privileges below, you will be presented with a Connection Information window. The Connection Information window details reason and the workaround.

Temporary Table Generation

To create or modify temporary tables in User-Defined Temp Table, Scanner Temp Table, Trigger Conversion and temporary table generation while optimizing the original SQL statement, the logon user needs the following privileges:

- Connection to DB2 LUW 7 or above.
- USE privilege on the USER TEMPORARY table space or SYSADM or DBADM authority.

Index Expert Function

The Index Expert function requires DB2 LUW version 7 or later to retrieve the indexes recommended by DB2 LUW functions. It requires DB2 LUW version 8 or later to use the Generate Indexes function with the Index Expert's unique Artificial Intelligence engine. It also requires that the statistics be run in order to be able to estimate the size of the index.

SQL Optimizer Module

On the Access Plan tab in the SQL Optimizer window, you can update the table and index statistics. In order to update statistics, you must have one of these privileges:

SYSADM

SYSCTRL
SYSMAINT
DBADM
CONTROL privilege on the table
LOAD authority

Special Registers Settings

Special Registers Settings allows you to change the current degree, optimization class, and current path settings for the current session.

To change the special register settings

1. Select **Database | Special Registers Settings**.
2. Change the settings as required.
3. Click **Apply**.

To reset back to the original database settings

Click **Reset**.

If the special registers settings had been changed you will notice icon appearing on the bottom right of the main window status. This icon will disappear if the special registers settings correspond to the database settings.

Related Topic

SQL Options

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 the program. This information can be loaded into memory of your local computer 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 selecting the **Do not load database dictionary after database connection** in the General page of the Options 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 SQL Optimizer.

To synchronize the database dictionary

Select **Database | Synchronize Data Dictionary**.

The Synchronize Data Dictionary function does not break the connection but updates the new database information in the SQL Optimizer memory.

Enter Authorization Key

SQL Optimizer requires an authorization key to use the program.

To enter another trial key or to enter the production key

1. Select **Help | Licensing**.
2. In the Licensing window, enter your key.
3. For a production key, enter your site message.
4. Click **Apply**.

Notes:

- The license key must be entered for each user logon on your computer.
- A trial key is valid for thirty days.

Tip: Access the Licensing window any time to review current license information in the License Details pane.

Select Database Language

The first time the program is launched after you have installed it, the Language window appears. This window enables you to select the language character set that was used to enter and display the SQL and data in your database.

The Interface for the program is always in English.

To select the language at the initial opening of SQL Optimizer

In the Character set field, select the language.

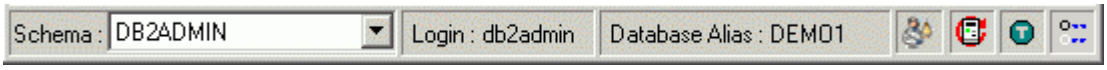
To select the language from the Options

1. Click .
2. Select the **General** tab.
3. In the **Character set** field, select the language.

Related Topic

General

Status Bar—Main Window



Item	Description
Schema	Displays the selected schema and lists available schemas. Use the Schema list to select the schema that corresponds to the SQL statements that you are working with.
Login	Read-only field, displaying the connected login for DB2 LUW.
Database Alias	Displays the DB2 LUW database alias for the current database.
Quest Assistant	Displays the Quest Assistant icon if Quest Assistant was minimized.
Connection Information	Displays the connection information icon if the user account does not have every privilege needed by every module.
Temp Table	Displays the temp table icon if a user-defined temp table has been created.
Special Registry Settings	Displays the special registry settings icon if you have made any changes to the DB2 LUW special registry settings using SQL Optimizer.

You may open the Quest Assistant, the Connection Information window, the User-Defined Temp Table window or the special Registry Settings window by clicking the corresponding icon on the Status Bar.

Toolbar

Availability of the toolbar buttons varies according to the active module. Buttons appear active when they are accessible and appear dimmed when they are not. By default all toolbar are shown.

To hide or show a toolbar

1. Select **Window | Customize Toolbars**
2. Select the toolbars you want to show.
3. Clear the toolbars you want to hide.

To reset the toolbar to the default

Select **Window | Customize Toolbars | Reset Default**.

Options

About Options

To change an Options setting

Click to specify the desired settings.

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 option settings either by saving settings from one or more tabs or by saving all the option settings.

To save specific settings so that you can use them later

1. In the Options window, click **Save**.
2. Enter a filename for the saved settings. Click **Save**.
3. From the Save Options Profile window, select the tabs for the settings you want to save. Click **OK**.

To restore the saved settings

1. In the Options window, click **Load**.
2. In the Open File window, select the filename. Click **Open**.
3. From the Save Options Profile window, select the tabs for the settings you want to load.

The Options window is divided into the following tabs and pages:

Optimization
Optimization Intelligence
Optimization (1)
Optimization (2)
SQL Options
Quota
General
Editor
Activity Log
Plan
SQL Scanner
Options
General
SQL Classification
Directory Setup
Linkage
General
General
Message
Result Set

SQL Rewrite Settings

SQL Rewrite Settings

The SQL Rewrite tab on the Options window consists of several pages that are selected from buttons on the left. These settings allow you to select the optimization level and other options that the SQL Rewrite function uses to create alternatives for your SQL.

The settings include the following:

Intelligence

SQL Rewrite (1)

SQL Rewrite (2)

SQL Options

Quota

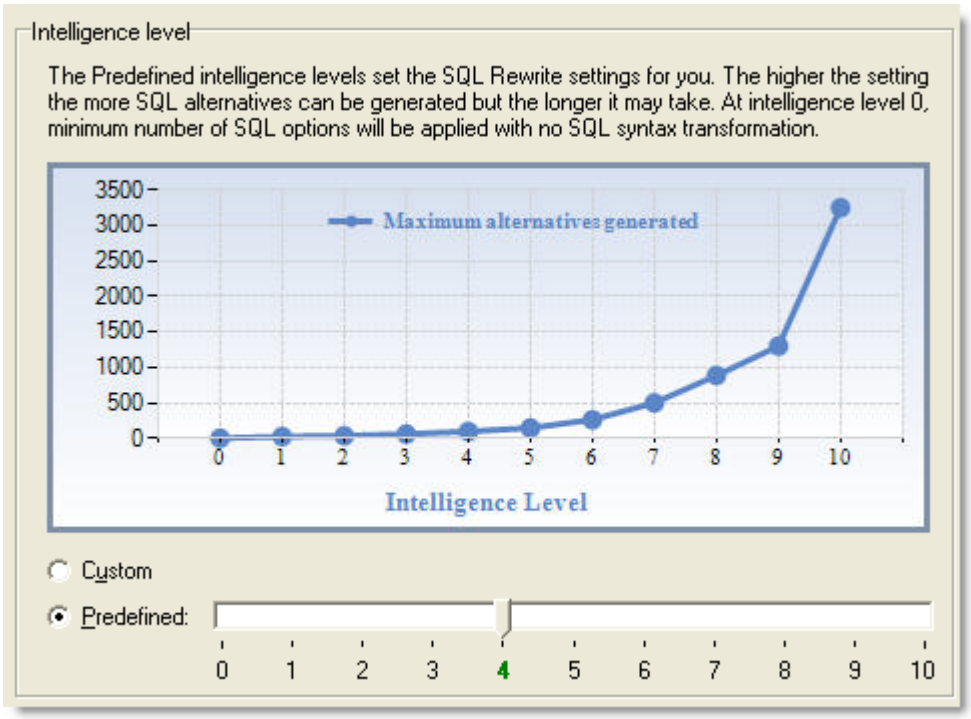
General

Related Topics

About the SQL Rewrite Function

Intelligence

View SQL Rewrite tab – Intelligence options



The Intelligence page under the SQL Rewrite tab on the Options window allows users to select the optimization level for rewriting the SQL. You can customize the optimization level or select a predefined level.

Intelligence Level

Option	Description
Custom	Enables you to customize the settings on the SQL Options, SQL Rewrite (1), SQL Rewrite (2), and Quota pages.
Predefine	Uses the predefined optimization level. The SQL Options, SQL Rewrite (1), and Quota pages are automatically updated according to the level selected. The higher the level, the more likely SQL Rewrite finds a better SQL alternative. **Note:** The settings on the SQL Rewrite(2) and the General pages are adjusted independent of the optimization-level settings.

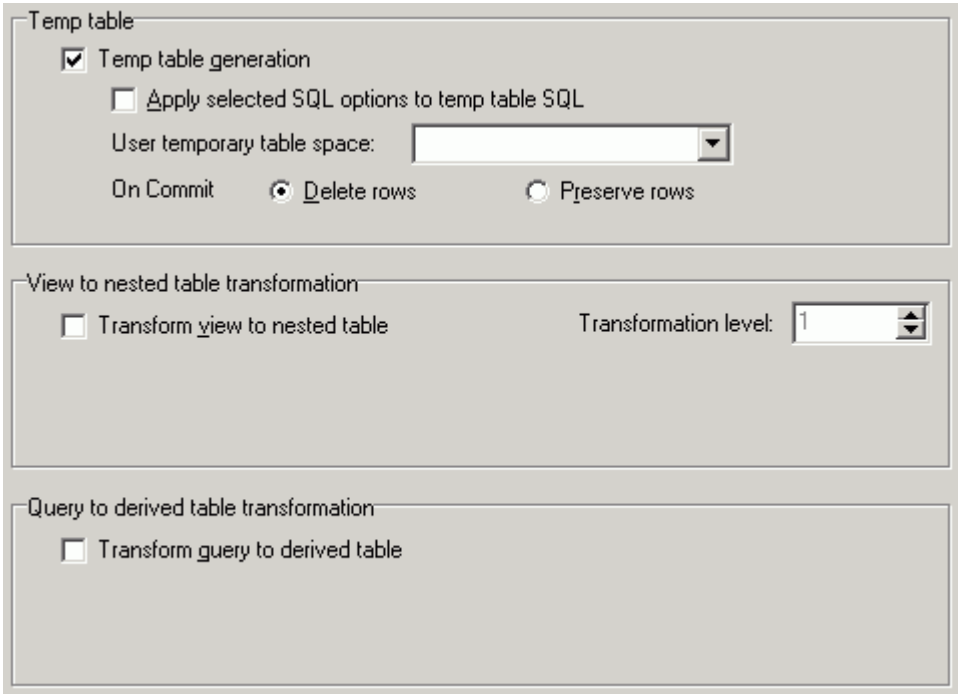
Related Topics

SQL Rewrite Settings

About the SQL Rewrite Function

SQL Rewrite (1)

View SQL Rewrite tab – SQL Rewrite (1) options



The SQL Rewrite (1) page under the SQL Rewrite tab on the Options window allows you to enable or disable certain specific SQL optimization techniques.

Temp Table

Temp Table Options	Description
Temp table generation	Specify whether to allow temp table

Temp Table Options	Description
	generation as SQL is being rewritten.
L	Specify whether to apply the selected SQL options to the generated temp table SQL.
User temporary table space	Specify the temporary table space to use when declaring the temp table. If no user temporary table space is specified then the database default user temporary table space will be used.
On commit	Specify whether to **Delete rows** or **Preserve rows** in the temp table after commit.

Note: To create or modify temporary tables in SQL Rewrite, the logon user needs the following privileges:

- Connection to DB2 LUW 7 or above
- USE privilege on the USER TEMPORARY table space or SYSADM or DBADM authority.

View to Nested Table Transformation

This transformation is only applicable if the SQL statement is using a view to access information from the database. When a SQL statement is using a view, the SQL Rewrite optimizes the view's SQL statement along with the original SQL statement. SQL Rewrite inserts the view's SQL into the original SQL statement in every place the view is referenced. Therefore the view's SQL is going to be rewritten along with the original SQL. This is very useful when you want to optimize a SQL statement that is using a poor performing view but you cannot change the view's SQL.

Original SQL

```
SELECT *
 FROM V_DEPT
WHERE DPT_MANAGER = 'SMITH'
```

Alternative SQL

```
SELECT *
 FROM (SELECT DPT_ID,
              DPT_MANAGER
         FROM DEPARTMENT)
 WHERE DPT_MANAGER = 'SMITH'
```

View to Nested Table Transformation Options	Description
Transform view to nested table	Specify whether to transform view to nested table - a subquery using as a table in the **FROM** clause.
Transformation level	Specify the recursive level to transform views inside the subquery of a nested

View to Nested Table Transformation Options	Description
	table. You can control whether the view's SQL is rewritten with the original SQL statement with the **Transform view to nested table** option. SQL Rewrite can also transform a view that is being used by another view. You control how many views will be included when the original SQL is rewritten by specifying the transformation levels that you would like to perform during the optimization.

Query to Derived Table Transformation

This transformation takes a original SQL statement with an IN or EXISTS clause and rewrites it as a derived table.

Original SQL

```
SELECT *
  FROM DEPARTMENT
 WHERE DPT_ID IN (SELECT EMP_DEPT
                  FROM EMPLOYEE)
```

Alternative SQL

```
SELECT DEPARTMENT.*
 FROM (SELECT DISTINCT EMP_DEPT AS COL1
          FROM EMPLOYEE) DERIVEDTABLE0,
       DEPARTMENT
 WHERE DPT_ID = DERIVEDTABLE0.COL1
```

Query to Derived Table Transformation Option	Description
Transform query to derived table	Specify whether to transform the query to a derived table – a subquery used as a table in a FROM clause.

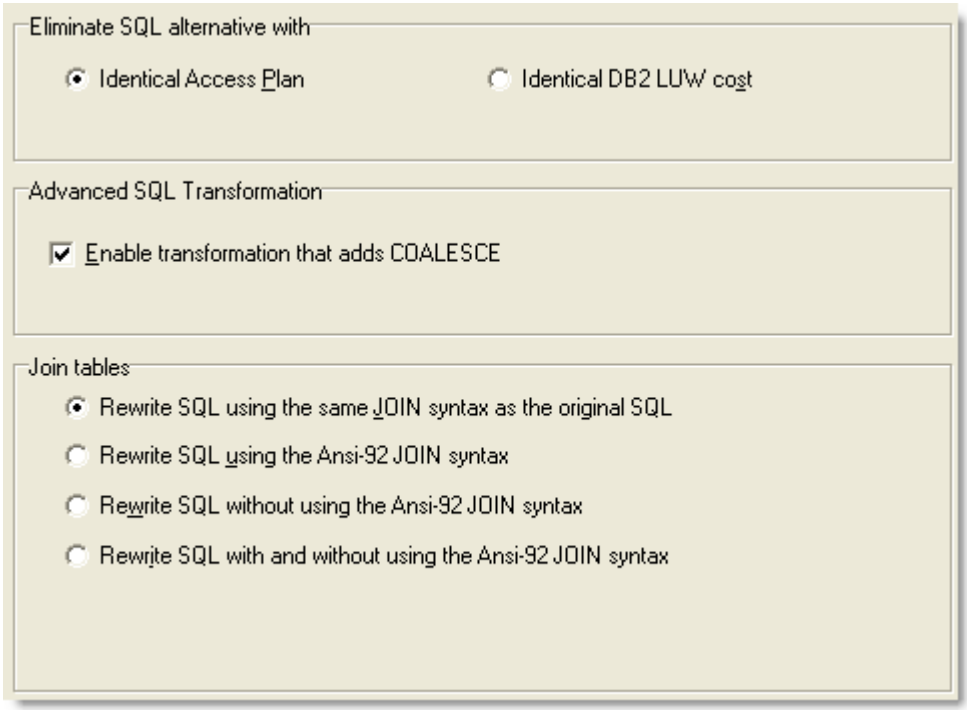
Related Topics

SQL Rewrite Settings

About the SQL Rewrite Function

SQL Rewrite (2)

View SQL Rewrite tab – SQL Rewrite (2) options



The SQL Rewrite (2) page under the SQL Rewrite tab on the Options window allows you to enable or disable certain specific optimization techniques and to define how duplicated access plans are eliminated.

Eliminate SQL alternative with

After SQL Rewrite has parsed the original SQL statement, it creates all the semantically equivalent SQL statements. It then eliminates alternative SQL statements based on one of two criteria: same access plans or same costs. Eliminating the SQL based on like access plans is more accurate, but can take longer because SQL Rewrite compares every operation in the access plans, instead of comparing only cost.

Eliminate SQL alternative with options	Description
Identical access plan (Default)	Eliminate optimized SQL statements with identical access plans.
Identical DB2 LUW cost	Eliminate optimized SQL statements that incur the same DB2 LUW cost amounts.

Advanced SQL transformation

Advanced SQL transformation option	Description
Enable transformation that adds COALESCE (Default = enabled)	Specify to apply the SQL syntax transformation rule that adds COALESCE to a column. When the data is retrieved, the COALESCE 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:

Advanced SQL transformation option	Description
	SELECT * FROM EMPLOYEE, DEPARTMENT WHERE COALESCE (DPT_ID, DPT_ID) = EMP_ DEPT

Join Tables

Join Tables options	Description
Rewrite SQL using the same JOIN syntax as the original SQL (Default)	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, the optimization process rewrites the syntax using the Ansi-92 JOIN syntax. The outer join is not included 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 DB2 SQL Options are applied to the original SQL and the transformed SQL. So you can see SQL alternatives that use the JOIN syntax from the original SQL, but these SQL alternatives are simply the original SQL with a SQL Option 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

Join Tables options	Description

Rewrite SQL without using the Ansi-92 JOIN syntax	Specify to join tables in the FROM clause without the Ansi-92 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 DB2 SQL Options 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 a SQL Option 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 use the both types of SQL syntax for joining the tables. Each type of join syntax will result in a different alternative.

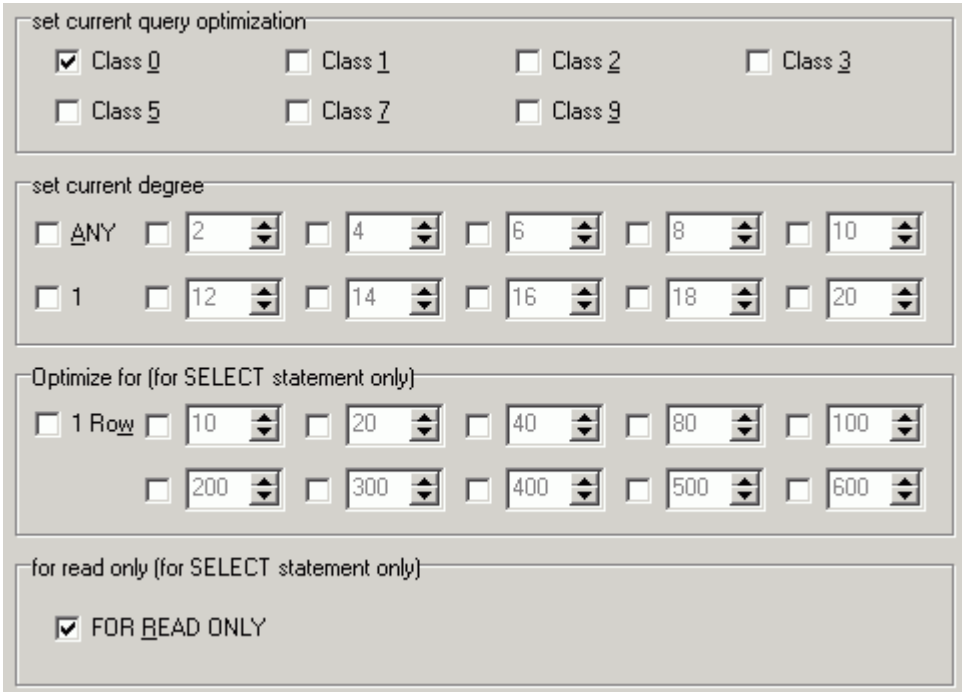
Related Topics

SQL Rewrite Settings

About the SQL Rewrite Function

SQL Options

View SQL Rewrite tab – SQL Options options



The SQL Options page under the SQL Rewrite tab on the Options window allows the user to define parameters for rewriting the SQL, as each database and optimizing objective is unique. By adding SQL options, you can force the SQL Rewrite process to choose a particular access plan. SQL options are added to the rewritten SQL statements and shown on the SQL Optimizer window.

When unsure which optimization SQL options to apply, use the defaults or choose a particular optimization level. The default settings should deal with most SQL rewriting cases.

set current query optimization

Specify whether to set the current DB2 query optimization class to 0, 1, 2, 3, 5, 7, 9 classes. The current query optimization controls the class the query optimization performs. This value affects the number of techniques the database optimizer is going to use to rewrite your SQL statement, in order to derive an adequate access plan. A higher value will allow more techniques used but may lengthen the compilation time of your SQL. Each class that you select will be applied to the rewritten SQL statements.

set current degree

Specify whether to set the current degree of intra-partition parallelism. A maximum 12 values can be selected at one time ranging from ANY, 1 to 32,767. The value ANY lets the database optimizer choose and appropriate degree depending on the resources. The fixed number values specify the number of parts the query is going to break down and execute concurrently. The higher the number, the more system resources are needed to execute the query.

Optimize for (for select statement only)

Specify whether to optimize the select statement for a particular number of rows. If this option is not selected then it is assumed that all rows will be retrieved. A maximum of 11 values can be selected at one time.

for read only (for SELECT statement only)

FOR READ ONLY

Specify whether to indicate that the result table is read-only. Specifying this clause informs the database optimizer that only the Share locks are required to execute the query. In certain situation, this piece of information helps the database optimizer derive a different access path.

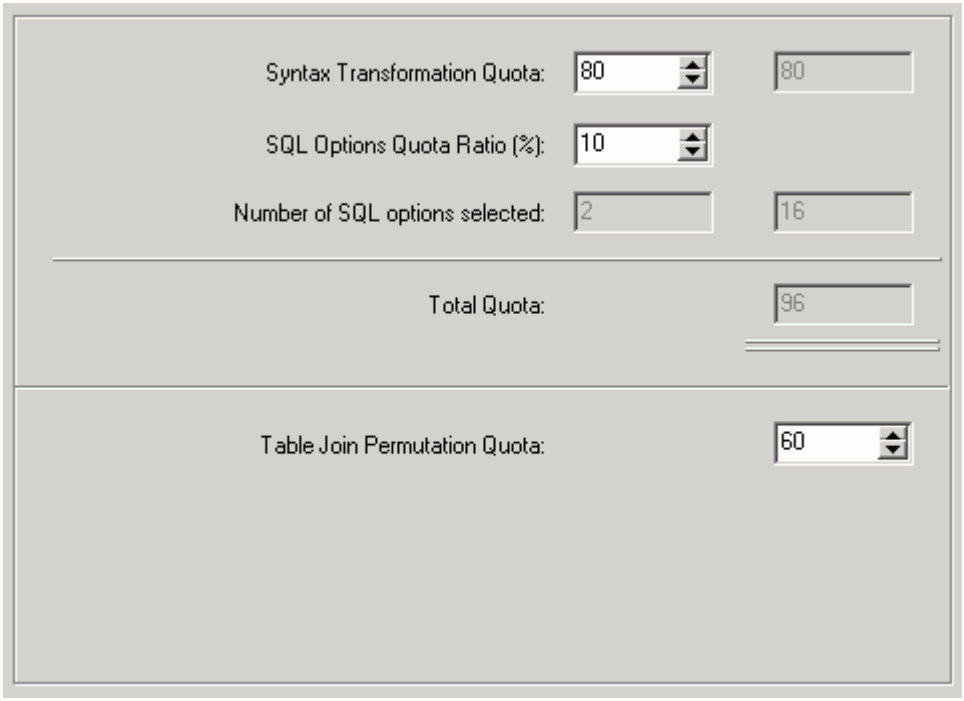
Related Topics

SQL Rewrite Settings

About the SQL Rewrite Function

Quota

View SQL Rewrite tab – Quota options



The Quota page under the SQL Rewrite tab on the Options window allows you to restrict the number of SQL transformations produced during SQL Rewrite process. You can modify quota information only when **Intelligence level** is set to *Custom* (on the Intelligence page).

Quota	Description
Syntax Transformation Quota (Default = *80*, Range = 1 to 99,999)	Specify the maximum number of SQL statements generated by applying SQL transformation rules. This quota affects how many SQL statements are created by changing the SQL syntax. The default quota of 100 is normally sufficient for most of the complicated SQL statements. However, the quota can be increased to tackle exceptionally complicated SQL statements with very high levels of table joins or multiple levels of nested sub-queries.
SQL Options	This percentage is used to calculate the maximum number of SQL statements generated by

Quota	Description
Quota Ratio (%) (Default = *10*%, Range = 1% to 100%)	applying SQL options.
Number of SQL options selected	This is a read-only field indicating the number of SQL options selected. This figure is used to calculate the maximum number of SQL statements generated by applying SQL options. *SQL Options Quota* = (*Syntax Transformation Quota* * *SQL Options Quota Ratio%*) * *Number of SQL Options selected*
Total Quota	This is a read-only field indicating the maximum number of SQL statements generated during SQL Rewrite. This figure consists of: *Syntax Transformation Quota* + *SQL Options Quota*.
Table Join Permutation Quota (Default = *60*, Range = 0 to 999,999)	Specify the maximum number of table join access path that will be considered duringSQL Rewrite.

Caution: The higher the quota, the longer it might take to rewrite a complicated SQL statement.

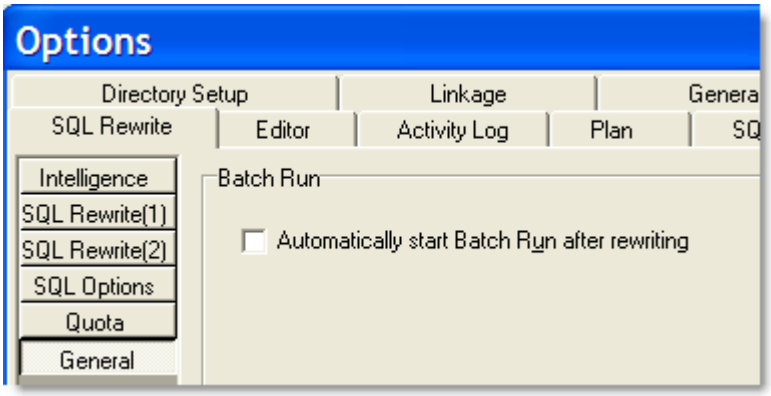
Related Topics

SQL Rewrite Settings

About the SQL Rewrite Function

General

View SQL Rewrite tab – General options



The General page under the SQL Rewritetab in the Options window provides the option to start a Batch Run immediately after the SQL Rewrite process is finished.

Batch Run

Batch Run	Description
Automatically start Batch Run after rewriting	Specify to start the Batch Run function on all the SQL alternatives as soon as the SQL rewriting is finished.

Related Topics

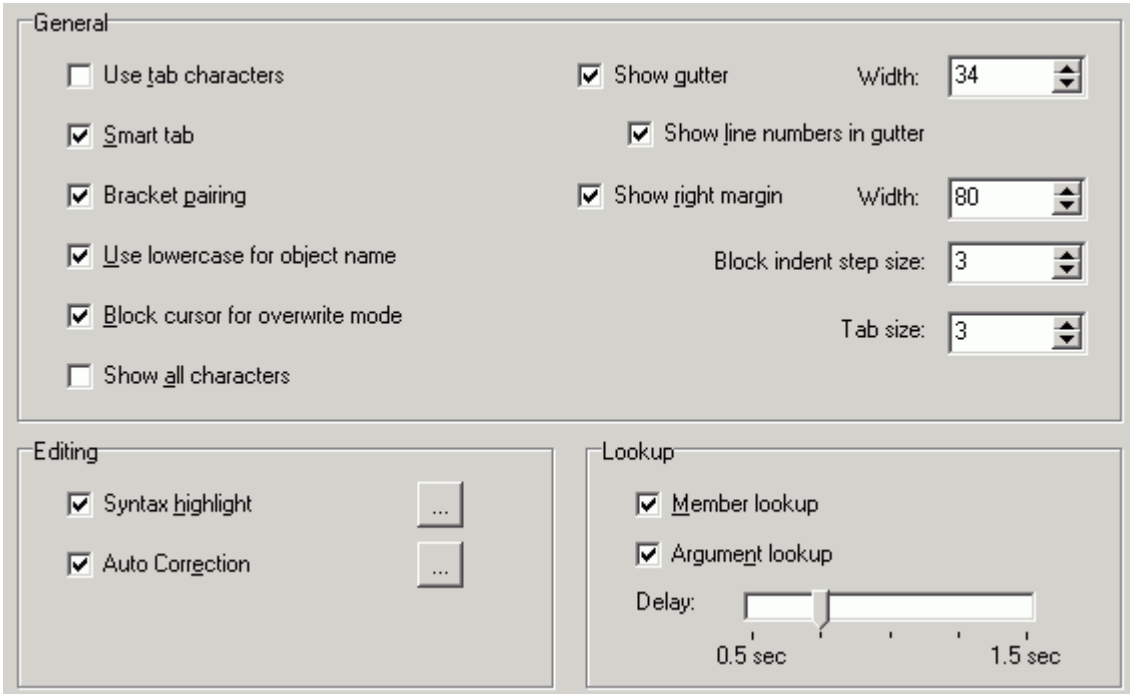
SQL Rewrite Settings

Batch Run

About the SQL Rewrite Function

Editor Settings

View Editor tab options



The Editor tab on the Options window allows users to define the editable panes' controls and layout.

General

General Settings	Descriptions
Use tab characters	Specify whether to use the tab character (ASCII 9) instead of spaces.
Smart tab (Default)	Specify whether to start a new line at the first non-whitespace character of the preceding line.
Bracket pairing (Default)	Specify whether to match the corresponding opening bracket when the end bracket is typed. This applies to the ([{ characters.
Use lowercase for object name (Default)	Specify whether to use lowercase for database object names.

General Settings	Descriptions
Block cursor for overwrite (Default)	Specify whether to change to block cursor for overwrite mode. Overwrite mode is when text entered at the cursor will overwrite existing text.
Show all characters	Specify whether to show all characters including spaces, new lines and tabs.
Show gutter (Default)	Specify whether to show a non-editable boarder on the left of the Editor pane.
Width (Default = *34*, Range = 1 to 100 pixel)	Specify the gutter width.
Show line numbers in gutter (Default)	Specify whether to show line numbers in the gutter.
Show right margin (Default)	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 step size	Specify the block indent step size in character used by the Indent/Outdent function.
Tab size	Specify the character length of a tab.

Editing

Editing Settings	Description
Syntax highlight (Default)	Specify whether to use syntax highlight. If selected, it is possible to define the format for the following syntax: reserved word, comment, identifier, quoted identifier, string, number, symbol, data type, default exception, hint, text selection, member lookup and argument lookup.
Auto Correction (Default)	Specify whether to automatically correct typing errors as characters are typed. Allows the addition, editing and deletion of the auto correction entry.

Lookup

Lookup Settings	Description
Member lookup (Default)	Specify whether to show the lookup hint for database object members. For example, displays a list column names when a table name is specified.
Argument lookup (Default)	Specify whether to show the argument

Lookup Settings	Description
	parameters hint for database functions and procedures.
Delay (Default = *0.75* sec, Range = 0.5 to 1.5 sec)	Specify the delay time before the lookup hint appears.

Related Topics

Auto Correction

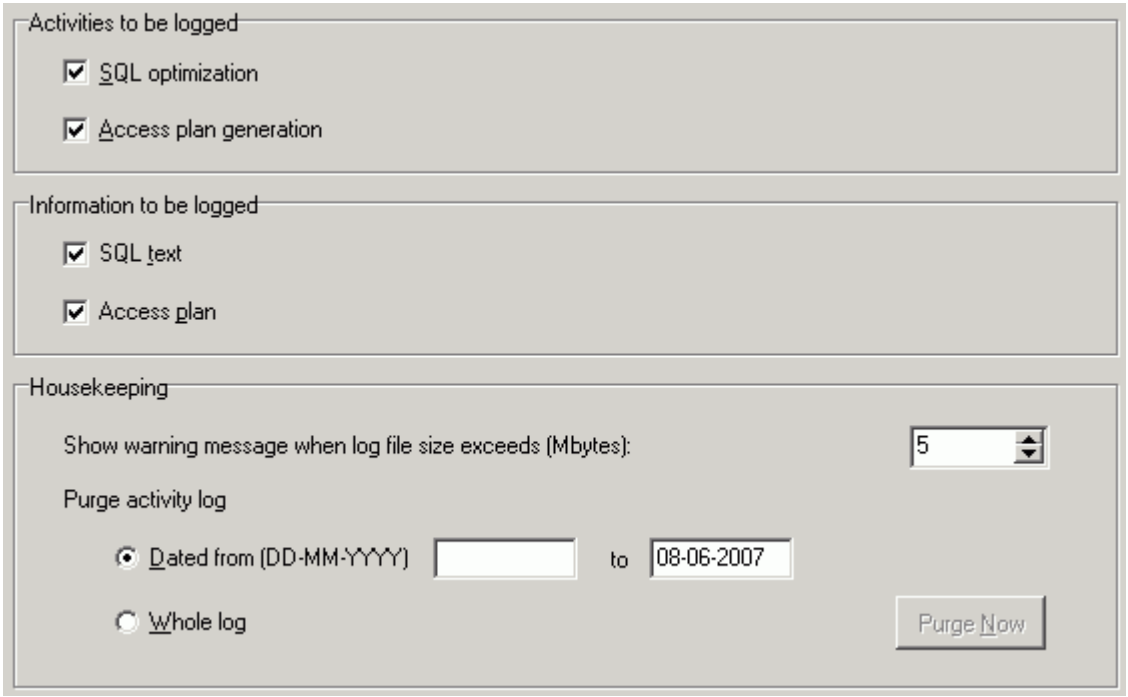
Brackets Pairing

Member and Argument Lookup

Syntax Highlight

Activity Log Settings

View Activity Log tab options



The Activity Log tab on the Options window allows users to specify whether to record the access plan generation and optimization activities during daily operations.

Activities to be logged

Activity	Description
SQL optimization	Specify to log the SQL optimization activities.
Access plan generation	Specify to log each time an access plan is retrieved.

The Activity log will record the activities from SQL optimization process and retrieval of an access plan. If **SQL optimization** or **Access plan generation** checkboxes are not selected, then no activities are recorded in the activity log. By default, no activities are logged.

Information to be logged

Item	Description
SQL text	Specify to save the text of the SQL statement.
Access plan	Specify to save the access plan for the SQL statement.

In addition to SQL text and access plan, the login user, OS user, schema, elapsed time, and SQL type are recorded automatically.

Housekeeping

Item	Description
Show warning message when log file exceeds (Default = *5MB*, Range from 1 to 500MB)	Specify the maximum size in MB of the activity log file. If the size of the activity log file exceeds the maximum value, a warning message is displayed.
Purge activity log	Use to remove information from the activity log.

To remove information

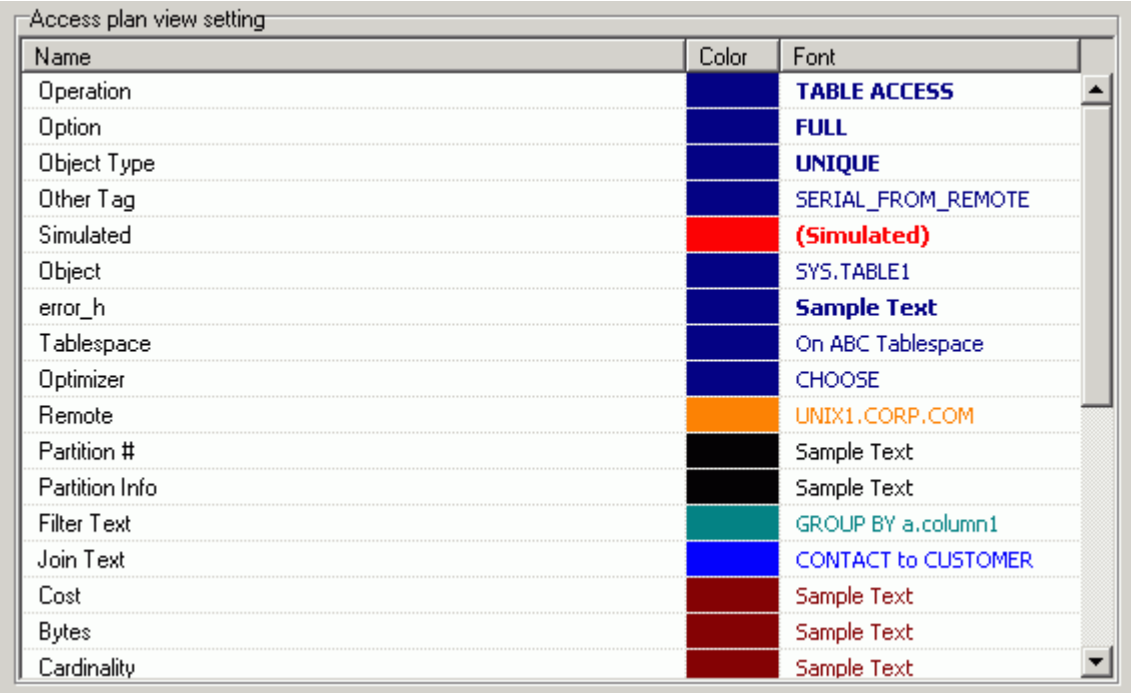
1. Select either **Whole log** to remove all information
   or
   Specify a date range to remove logs between these dates.
2. Click **Purge Now** to remove the desired logs.

Related Topic

About Activity Log

Plan Settings

Plan tab options



The Plan settings are used to select the color and fonts used in the access plan.

Access Plan View Setting	Description
Color	Specify the color of the individual items in the access 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 access plan by clicking the **Font** column in the row for the item and selecting the new settings from the dialog.

To select which elements of the access plan are displayed

Right-click and select **Plan Options**.

Related Topic

Access Plan

SQL Scanner Settings

SQL Scanner Settings

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

SQL Scanner Options 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.

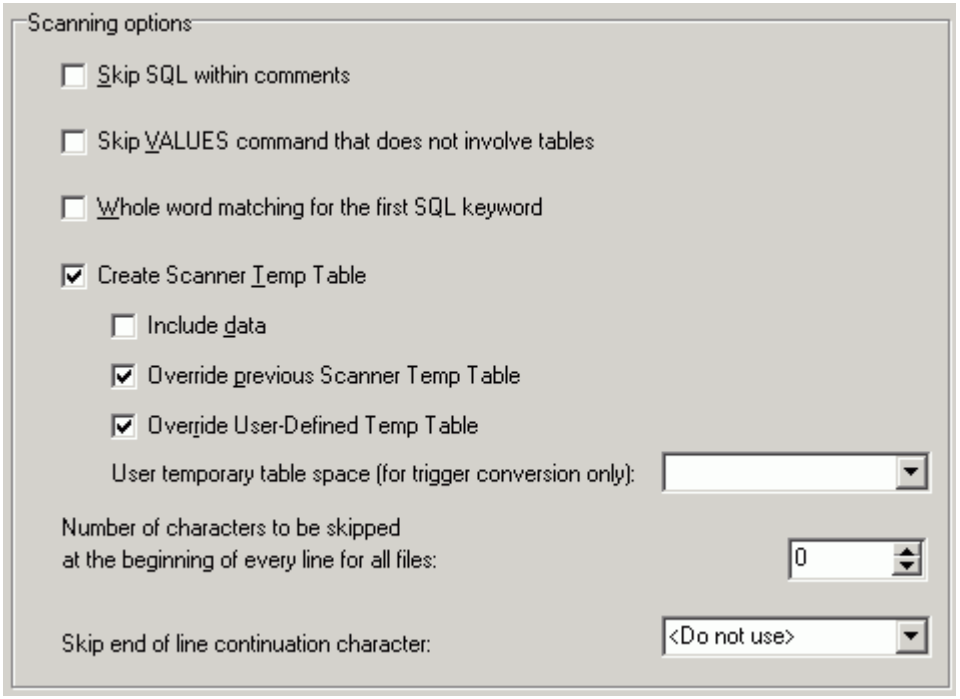
SQL Classification Settings

Related Topic

About SQL ScannerSQL Scanner

Options (SQL Scanner Tab)

View SQL Scanner tab – Options button options



The Options page on the SQL Scanner tab of the Options window allows you to specify criteria used when scanning for SQL statements.

Scanning Options	Description
Skip SQL within comments	Specify whether to 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 VALUES command that does not involve tables	Specify whether to ignore any VALUES command, such as: VALUES CURRENT SCHEMA.
Whole word matching for the first SQL keyword	Specify to search for SELECT, INSERT, UPDATE or DELETE as a whole word; the keyword must be preceded and followed by a space. The SQL Scanner therefore will not find the word INSERT in something like PROCEDUREINSERT and attempt to build a SQL statement from it.
Create Scanner Temp Table	Specify whether the scanning algorithm will automatically create temp table during scanning when a declare temp table statement is scanned. This option will not override any temp tables created in User-Defined Temp Tables window.

Scanning Options	Description
Include data	Specify whether Scanner Temp Table will include data. If this option is selected, the SQL Scanner will automatically execute any INSERT, DELETE, and UPDATE SQL statement that modifies Scanner Temp Table. **Note:** Selecting the Include data option may affect the total scanning time.
Override previous Scanner Temp Table	Specify whether to override a previous Scanner Temp Table if it scans another declare temp table statement using the same table name. This option will not override any temp tables created in User-Defined Temp Tables window.
Override User-Defined Temp Table	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.
User temporary table space (for trigger conversion only)	Specify the user temporary table space used to create the temp table for the Trigger Conversion. If no user temporary table space is specified then the database default user temporary table space will be used. **Note:** To create or modify temporary tables in the SQL Scanner the logon user needs the following privileges: • Connection to DB2 LUW 7 or above • USE privilege on the USER TEMPORARY table space or SYSADM or DBADM authority
Number of characters to be skipped at the beginning of every line for all files	When a file is scanned, the SQL Scanner skips the specified number of characters at the beginning of every line.
Skip end of line continuation character	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 box as long as it is the selected character. Once you make another selection, your own character will not be remembered.

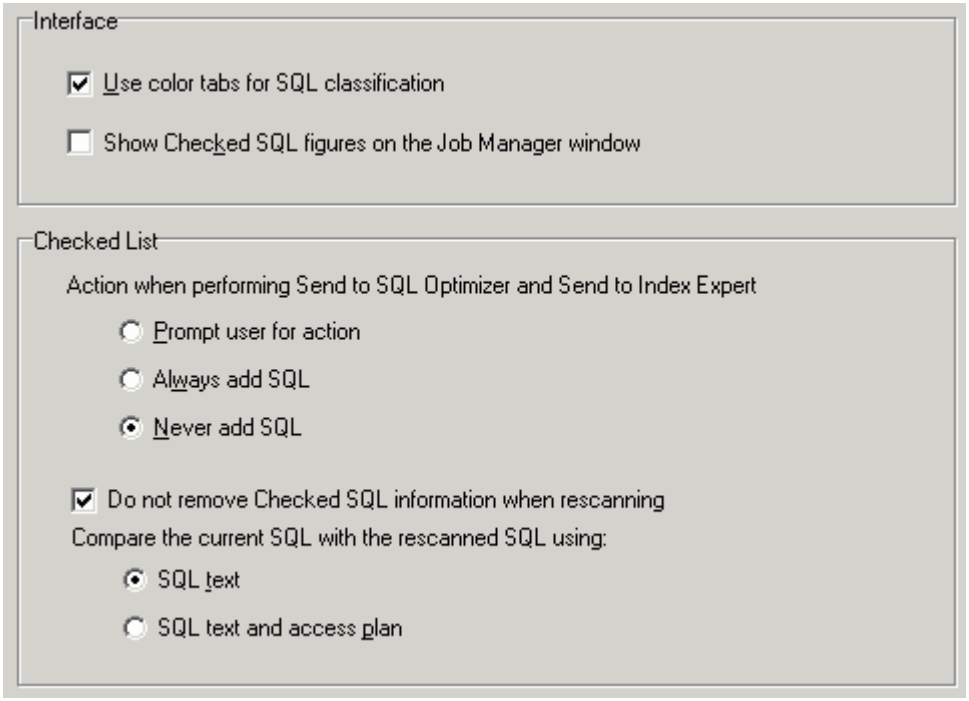
Related Topics

About SQL Scanner

SQL Scanner Settings

General (SQL Scanner Tab)

View SQL Scanner tab – General button options



The General page on the SQL Scanner tab of the Options window allows you to specify your options for the graphical interface and the Checked SQL feature in the SQL Scanner.

Interface

Interface Settings	Description
Use color tabs for SQL classification	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 gray.
Show checked SQL figures on the Job Manager window	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 Job Manager window.

Checked List

Checked List Settings	Description
Action when performing Send to SQL Optimizer or Send to Index Expert	
Prompt user for action	If selected, you are prompted to be added to the Checked List.
Always add SQL	If selected, all SQL sent to the SQL Optimizer is added to the Checked List.
Never add SQL (default)	If selected, all SQL sent to the SQL Optimizer is not automatically added to the Checked List.

Checked List Settings	Description
Do not remove Check SQL information when rescanning	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	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 access 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 access plan as a comparison for the match. The checked SQL information is only preserved for those SQL statements where the access plan and the SQL text match.

Related Topics

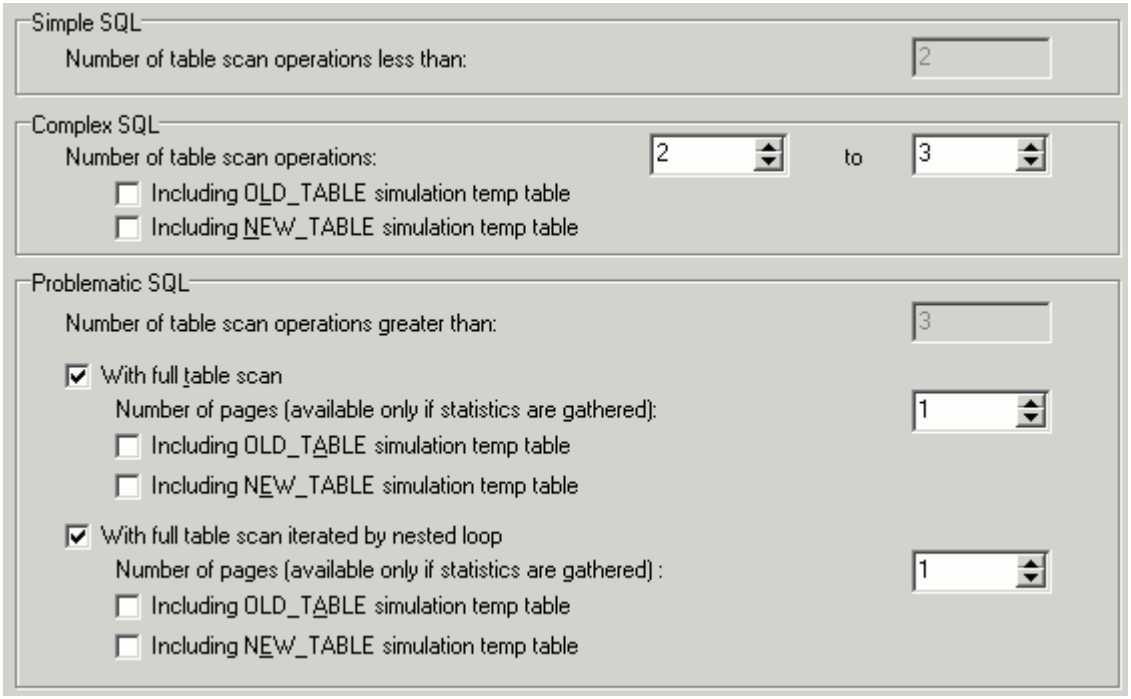
About SQL Scanner

SQL Scanner Settings

SQL Classification Settings

SQL Classification Settings

View SQL Classification tab options



The SQL Classification tab on the Options window allows users to specify the criteria to analyze a SQL statement. If the SQL statement analyzed satisfies any of the settings below, it will either be classified as Complex or Problematic SQL. Problematic SQL statement indicates potential problematic SQL statement and should be optimized, while complex SQL statements are complicated SQL statements where there is room for improvement.

The following settings are used to set the criteria for Simple, Complex and Problematic SQL statements.

Simple SQL

Simple SQL Criteria	Description
Number of table scan operations less than (Default = *2*)	Read-only field indicating the number of table scan operations references in the access plan. If the total number of table scan operations is less than this value, then this SQL statement will be classified as Simple. This value is the same as the lower limit of the complex table scan operations range.

Complex SQL

Complex SQL Criteria	Description
Number of table scan operations (Default = *2 /3*, Range 2 to 99)	Specify the range of the number of table scan operations referenced in the access plan for Complex SQL statements.
Including OLD_TABLE simulation temp table	Specify to count the OLD_TABLE simulation temp table created in Trigger Conversion.
Including NEW_TABLE simulation temp table	Specify to count the NEW_TABLE simulation temp table created in Trigger Conversion.

Problematic SQL

Problematic SQL Criteria	Description
Number of table scan operations greater than (Default = *3*)	Read-only field indicating the number of table scan operations references in the access plan. If the total number of table scan operation is greater than this value, then this SQL statement will be classified as Problematic. This value is the same as the upper limit of the complex table scan operations range.
With full table scan	Specify whether SQL statements with a single table full scan with table size greater than or equal to the defined page size (in Ketose) will be classified as Problematic SQL statements.
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 number of pages in a table.
Number of pages (available only if statistics are gathered) (Default = *1*, Range= 1 to 9999996)	Specify the number of pages in the table.
Including OLD_TABLE simulation temp table	Specify to count the OLD_TABLE simulation temp table created in Trigger Conversion.
Including NEW_TABLE simulation temp table	Specify to count the NEW_TABLE simulation temp table created in Trigger Conversion.

Related Topic

Scanned SQL Statement Classification

SQL Statement with no Access Plan

In addition to being classified as Simple, Complex, or Problematic, some SQL statements are classified as "Invalid SQL" to indicated that access plan could not be retrieved from the database. The DB2 LUW error message which indicates that the access plan was not retrieved is displayed to help you determine why the SQL statement was classified as invalid. The Invalid SQL classification may result from one of the following:

Reason for Invalid SQL Classification	Explanation
No permission to tables or views	The current user does not have privilege to the tables, views, or other database objects referenced in the SQL statement even though the syntax of the SQL statement is correct.
Dynamic SQL statements	The SQL Scanner is unable to identify SQL statements that are dynamically constructed at run time. This type of SQL is found in the source code on several command lines and the exact construct of the SQL statement is not determine until the application is executed.
Database object does not exist	A database object references in the SQL statement does not exist.

Reason for Invalid SQL Classification	Explanation
Incorrect Schema	The schema that was used when the execution plan was retrieved was not the correct schema for the SQL statement.

Related Topic

SQL Classification Settings

Directory Setup

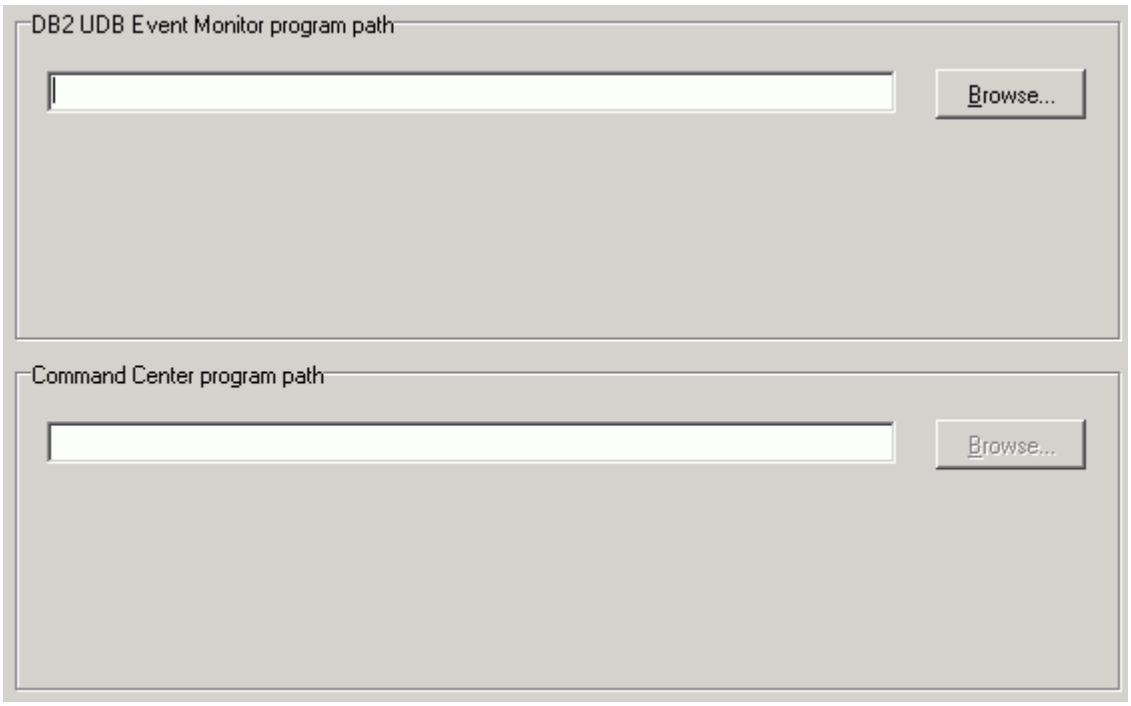
The Directory Setup tab of the Options window allows you to define the directories for saving and opening files used in the different modules.

Directory for	Description	Default Directory
Open and Save dialog	Directory for opening and saving files with the Open or Save commands.	C:\Documents and Settings*username*\Application Data\Quest Software\Quest SQL Optimizer for IBM DB2 LUW
Activity Log	Used to store the data files created while recording activities.	C:\Documents and Settings*username*\Application Data\Quest Software\Quest SQL Optimizer for IBM DB2 LUW
SQL Scanner data	Used to store the data files created while scanning. Changes to this directory cannot be made while the SQL Scanner is active.	C:\Documents and Settings*username*\Application Data\Quest Software\Quest SQL Optimizer for IBM DB2 LUW\DATA
SQL Repository data	Used to store the SQL statements saved in the SQL Repository.	C:\Documents and Settings*username*\Application Data\Quest Software\Quest SQL Optimizer for IBM DB2 LUW
Index Usage Analyzer data	Used to store the information about each Index Usage Analysis.	C:\Documents and Settings*username*\Application Data\Quest Software\Quest SQL Optimizer for IBM DB2 LUW
Index Impact Analyzer data	Used to store the information about each Index Impact Analysis.	C:\Documents and Settings*username*\Application Data\Quest Software\Quest SQL Optimizer for IBM DB2 LUW
SQL Parameter History data	Used to store the information about variables that are used in the SQL statement.	C:\Documents and Settings*username*\Application Data\Quest Software\Quest SQL Optimizer for IBM DB2 LUW\SQLPARAM

Caution: 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.

Linkage Settings

View Linkage tab options



The Linkage tab of the Options window enables you to locate the executable files that open the Command Center programs.

Item	Description
Command Center program path	Specify the directory path for batch file that is used to start DB2's Command Center program.

General Settings

General Settings

The General tab on the Options window consists of 3 buttons that allow users to specify settings used throughout the program.

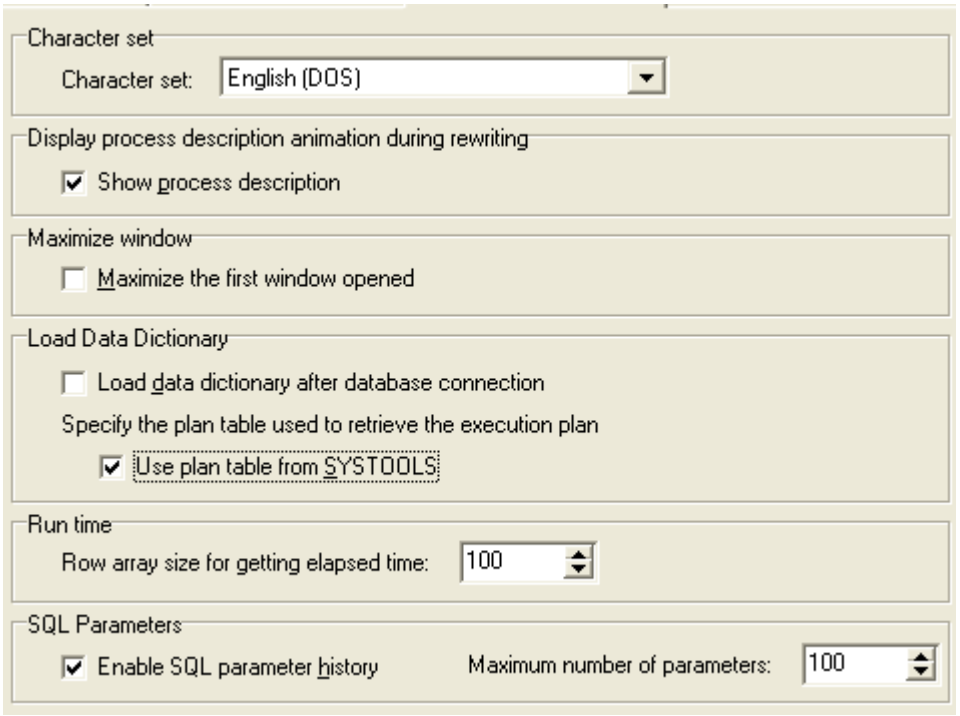
General

Message

Result Set

General

View General tab – General button options



The General page of the General tab on the Options window allows users to specify the general settings.

Character set

Character set	Description
Character set (Default = *English (DOS)*)	Specify the language set used to enter and display SQL and data. This setting does not affect the language of the program interface. It is always in English.

Display process description animation during optimization

Process Description	Description
Show process description (Default = checked)	Specify whether to show an animation describing the optimization process. This option is useful for first time users. It also indicates that the optimization process is still working when the process bar stays at the same percent for a long period of time.

Maximize window

Maximize Window	Description
Maximize the first window opened (Default = unchecked)	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 opened. Once all windows are closed, the next window that is open is maximized as it is

Maximize Window	Description
	opened. **Note:** This feature only applies to windows with the Minimize, Maximize, and Close boxes in the upper right corner of the window.

Load Data Dictionary

Load Data Dictionary	Description
Load data dictionary after database connection (Default = unchecked)	Specify that the database dictionary is automatically loaded into the memory of the computer every time you connect to a database. When this option is not selected, the specific information needed from the database is loaded when the SQL statement is parsed for functions such as scanning, optimization, and index generation. This save time when you are initially connecting to the database from each module, but it adds a slight bit of time later when the SQL statement is parsed.
Use plan table from SYSTOOLS	Select this option if you want SQL Optimizer to use the EXPLAIN tables created under SYSTOOLS. If EXPLAIN tables already exist under the current schema, SQL Optimizer asks you to specify which set of tables to use.

Run time

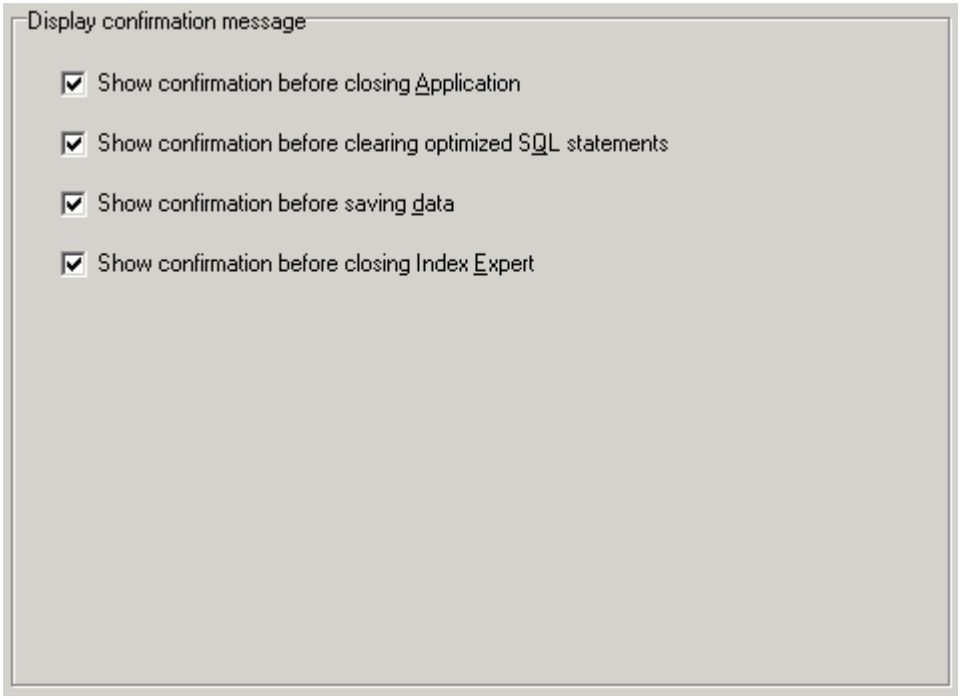
Run Time	Description
Row array size for getting elapsed time (Default = *100*, Range 0 to 99990)	Define the number of rows used to fetch rows returned by the SQL statement while retrieving the run time for all records.

SQL Parameters

SQL Parameters	Description
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. The directory where this information is save is set in the Directory Setup options.
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.

Message

View General tab – Message button options



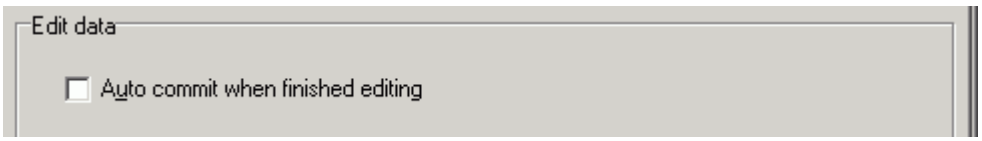
The Message page of the General tab on the Options window allows users you to decide which confirmation messages to show.

Display confirmation message

Confirmation Message	Description
Show confirmation before closing Application (Default)	Specify whether to show a confirmation message when closing SQL Optimizer.
Show confirmation before closing optimized SQL statements (Default)	Specify whether to show a confirmation message when closing the SQL Optimizer window.
Show confirmation before saving data (Default)	Specify whether to show a confirmation message when saving data. For example, from the Database Explorer (Data tab) windows.
Show confirmation before closing Index Expert (Default)	Specify whether to show a confirmation message when closing the Index Expert window.

Result Set

View General tab – Result Set button options



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

Edit data

Edit data	Description
Auto Commit when finished editing	Automatically commit the changes after each record is changed when editing the data in the Database Explorer window.

Generate Indexes Settings

Generate Indexes Settings

The Generate Indexes tab on the Options window allows users to define the optimization level and other settings used to generate index-set alternatives.

The Settings include:

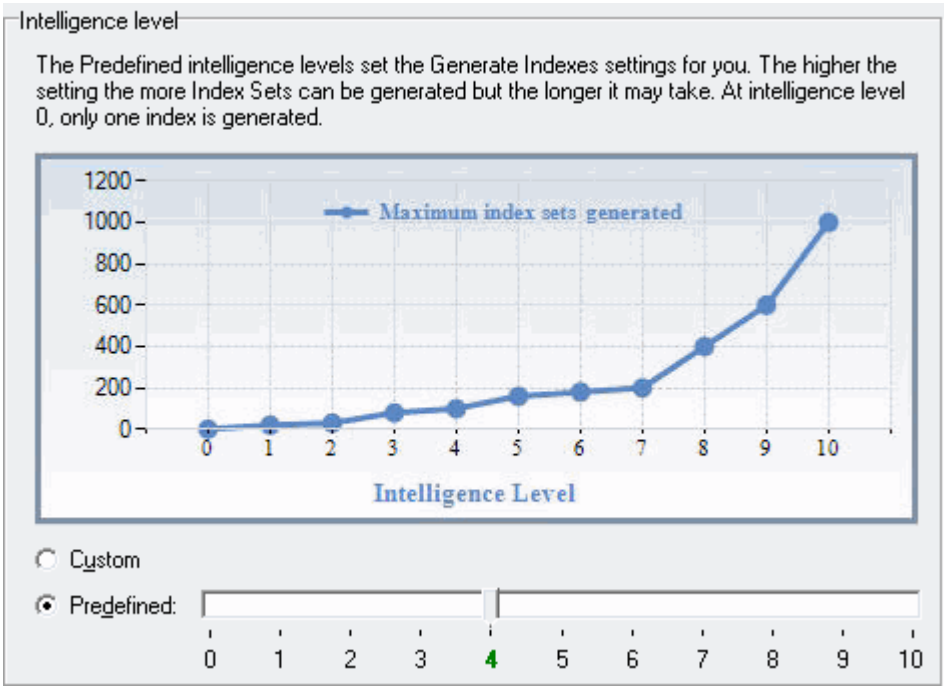
Intelligence

Index Options

General

Intelligence

View Generate Indexes tab – Intelligence button options



The Intelligence page under the Generate Indexes tab on the Options window allows you to select the optimization level for generating index-set alternatives.

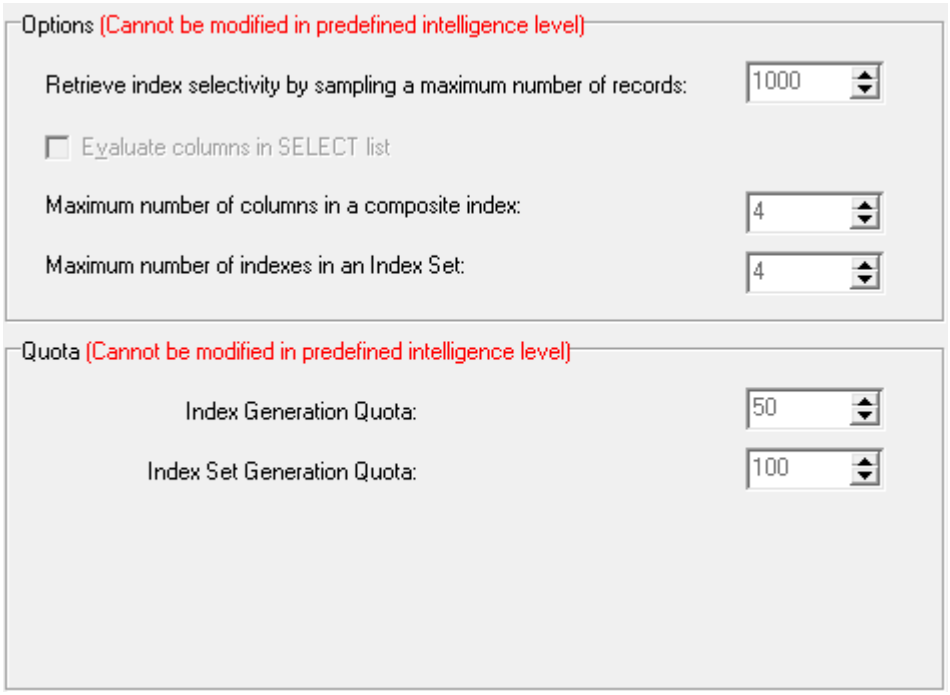
Intelligence Level

Option	Description
Custom	Use the Index Options page settings to determine the optimization level as needed to generate the index-set alternatives.
Predefine	Set a specific optimization intelligence level here. Values on the Index Options page adjust to this level. Levels range from 0 to 10. The higher the setting, the higher the number of Index sets generated. **Note:** The General page settings can be adjusted independent of which intelligence level is chosen.

Related Topics

Index Options

View Generate Indexes tab – Options button options



The Index Options page under the Generate Indexes tab on the Options window defines how to generates index-set alternatives. If you selected **Customize** on the Intelligence page, these values determine the optimization level for index generation.

Options

Index Options	Description
Retrieve index selectivity by sampling a maximum number of records (Default: *1000*, Range = 10 to 99999)	Specify the number of records to be selected for determine the selectivity of the data.
Evaluate columns in SELECT list	Specify whether to consider SELECT list columns when generating indexes.
Maximum number of columns in a composite index (Default *4*, Range 1 to 30)	Specify the maximum number of columns in a composite index.
Maximum number of indexes in an Index Set (Default *4*, Range 1 to 99)	Specify the maximum number of indexes in an index set.

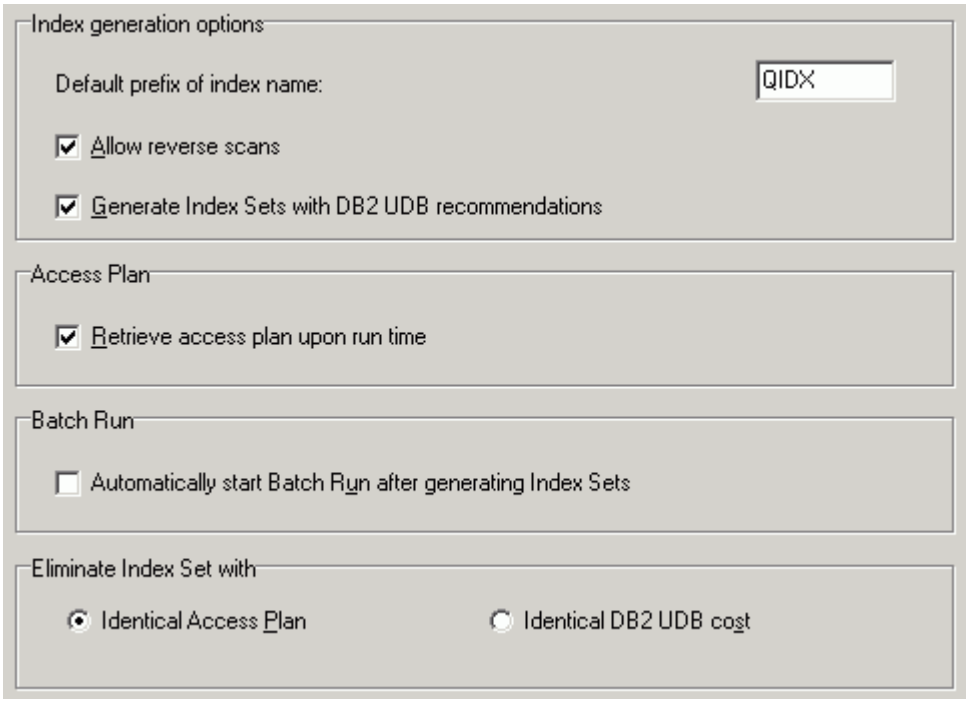
Quota

Index Generation Quotas	Description
Index Generation Quota	Specify the maximum number of indexes generated.
Index Set Generation Quota	Specify the maximum number of index sets generated.

Related Topic

General

View Index Expert tab – General button options



The General page under the Index Expert tab on the Options window allows you to control specific actions used by the Index Expert.

Index generation options

Index Generation Settings	Description
Default prefix of index name: (Default: *QIDX*)	Enter the prefix that is placed on the index name when the Index Expert automatically generates index candidates.
Allow reverse scans	Specify to create an index that can support both forward and reverse scans; that is, the index will be scanned in the order defined in INDEX CREATE statement and also in the opposite (or reverse) order.
Generate Index Sets with DB2 LUW recommendations	Specify to retrieve the indexes that are recommended by DB2 LUW for a SQL statement using DB2's SET CURRENT EXPLAIN MODE RECOMMEND INDEXES command. These indexes are combined with the indexes generated by the Index Expert's Artificial Intelligence engine into the index-set alternatives that you can evaluate.

Access Plan

Access Plan Retrieval	Description
Retrieve access plan upon run time	Specify whether to retrieve the access plan while retrieving the run time.

Batch Run

Batch Run	Description
Automatically start Batch Run after generating index sets	Specify to start the Batch Run as soon as the index sets have been generated. Batch Run creates each set of indexes, runs the SQL statement, and then drop the indexes.

Eliminate Index Set with

After Index Expert generates the initial index-set candidates, it eliminates candidates based on one of two conditions. Either an index-set access plan is identical to the original access plan or to another candidate's plan or the access plan has the same DB2 cost as the original plan or to another candidate's plan. Eliminating the candidates based on the plan comparison rather than cost comparison is more accurate. However, plan comparison can take longer because Index Expert compares each operation in the plans instead of only costs.

Eliminate Index Set With	Description
Identical Access Plan	Specify whether to eliminate the index-set candidates that produce the same access plan for the SQL statement.
Identical DB2 LUW cost	Specify whether to eliminate the index-set candidates when the access plan has same DB2 cost.

Related Topics

Index Expert Settings

Index Expert Overview

SQL Scanner

SQL Scanner Concepts

About SQL Scanner

The SQL Scanner is used to detect potentially problematic SQL statements in source code and database objects without execution.

The SQL Scanner extracts SQL statements embedded in database objects, stored in application source code and binary files. It retrieves and analyzes the execution plans for the extracted SQL statements. It then categorizes the SQL statements according to the complexity of the execution plan and determines whether it has the characteristics that typically cause performance problems. The SQL Scanner allows you to quickly review SQL statements in existing code and detect potential problems. With this approach, you can be proactive in the detection of

performance problems and identify the SQL statements that need to be optimized without physical execution of the applications or manual intervention.
Once the problematic SQL statements have been identified, you can determine the best solution by

- Sending a SQL statement to the SQL Optimizer for optimization.
- Saving a group of SQL statements to the SQL Repository for further analysis.

Each item that is scanned is referred to as a "job" and can be a database object, text file, or binary file.
The SQL Scanner has the following functions:

Job Manager

Provides a work area for adding and deleting scan Jobs. It also displays information on the Job name, location, number of SQL statements scanned, SQL classification, start date and time, and the processing time.

Scan

Scans the selected programming source code and database objects to extract SQL statements. Once identified, the scanned SQL statements are ranked based on a set of user-defined SQL classification criteria.

Scanned SQL Viewer

Displays the SQL statements that were found. The viewer shows the scanned SQL statement, corresponding access plan, and the associated information to help you identify the problem SQL. You can send these SQL statements directly to the SQL Optimizer for optimization. or the Index Expert to generate index candidates.

Related Topics
SQL Scanner Tutorial
Job Manager Overview
Job Manager Window
Job Manager Functions
Scanned SQL Viewer Overview
Scanned SQL Viewer Window
Scanned SQL Viewer Functions

Embedded and Dynamic SQL Statements

The SQL Scanner can extract a SQL statement that is an embedded SQL statement or is a dynamic SQL statement on a single command line. A dynamic SQL statement which is constructed from several command lines is not created until the user and program make decisions on what to include at run time. The SQL Scanner cannot extract this type of SQL statement.

Embedded SQL Statements

Embedded SQL statements refer to SQL statements that are placed within the source programs and are constructed at compilation time. Embedded SQL statements are complete statements found in the source code in one continuous string.

Dynamic SQL Statements on a Single Command Line

Database Objects and files may contain a SQL statement that is on one command line, is placed within quotes, and may be concatenated with other parts of the statement and/or local program variables.

For example:

```
SQL1:= "SELECT emp_id , " + variableA + " FROM employee";
```

Dynamic SQL Statements on several command lines

Dynamic SQL statements may also be constructed from parts of the SQL statement which are on several command lines. The user or program must decide at run time how the entire SQL statement will be constructed.

What does the SQL Scanner scan?

The SQL Scanner extracts SQL statements from the following items:

Item	Description
Database Objects	Database objects. The SQL Scanner only scans the database objects that contain SQL statements: materialized query tables, views, procedures, user-defined functions, triggers, packages, and plan tables.
Text/Binary Files	Application source code files and binary files.
COBOL Files	COBOL source code files.

Related Topic

Embedded and Dynamic SQL Statements

Scanned SQL Statement Classification

All SQL statements retrieved by SQL Scanner are syntactically correct. After the SQL statements have been identified, the access plan is retrieved. If the access plan is not retrieved successfully, then these SQL statements are not valid. All valid SQL statements are further classified as Simple, Complex and Problematic SQL statements. The definitions of these classified SQL statement types are dependent on user-defined parameters in the Options window.

Problematic SQL Statements

Problematic SQL statements are potentially problematic SQL statements that should be optimized. Problematic SQL should satisfy one of the following criteria:

- The number of tables referenced in the access plan of a SQL statement exceeds the upper limit of the Complex table scan operation range (option to include or exclude OLD_TABLE or NEW_TABLE simulated temp table for Triggers).
- At least one table full scan with table size larger than the threshold size. The default threshold size is 1 page which it can be changed in Options.

- At least one table full scan executed in a nested loop with table size larger than the threshold size. The default threshold size is 1 page which it can be changed in Options.

Complex SQL Statement

Complex SQL statements are complicated SQL statements where there is room for improvement. Complex SQL statements are SQL statements that involve a certain number of table references in their access plans. If the number of tables referenced in the access plan of an SQL statement falls into the Complex table scan operation range (option to include or exclude OLD_TABLE or NEW_TABLE simulated temp table for Triggers) defined in the Options window, this SQL statement will be classified as a Complex SQL statement

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 access plan is l than the lower limit of Complex table scan operation range defined in the Options window, the default value being less than 2 tables.

No Plan SQL Statements classified as "Invalid SQL"

If the execution plan is not retrieved successfully, then the SQL statement is classified as *Invalid SQL*. A SQL statement is invalid if the database object it references does not exist, the database user does not have privileges to access the database object, or the schema used is not the correct one for the SQL statement.

Data Directory Setting

SQL Scanner data directory is the location to store the data files created while executing the Scan function. The data directory path is stored in the Options window.

To change the directory path

1. Click .
2. Select the **Directory Setup** tab.
3. Click the **SQL Scanner data directory** box.
4. Click **Browse**.
5. Select a directory for storing the Scanner files.

Changes to this directory cannot be made while SQL Scanner is active.

Note: It is advisable not to change the data directory after selection, as files created during scanning will be kept in this directory.

About SQL Scanner Conversions

In order to render the SQL statement as a valid standalone SQL from other program syntax, one or more conversions may need to be applied to the SQL statement.

Conversions includes:

Trigger Conversion

Parameter Markers Conversion

Local Variable Conversion

Quoted String Conversion

COBOL Conversion

If the above conversion had been applied to the SQL statement, the Information pane shows what conversions had been applied and what changes had been made to the SQL text.

Note: If conversion had been applied, it may be necessary to reverse the changes after optimization when it is pasted back to the original source code.

About SQL Scanner Conversions

In order to render the SQL statement as a valid standalone SQL from other program syntax, one or more conversions may need to be applied to the SQL statement.

Conversions includes:

Trigger Conversion

Parameter Markers Conversion

Local Variable Conversion

Quoted String Conversion

COBOL Conversion

If the above conversion had been applied to the SQL statement, the Information pane shows what conversions had been applied and what changes had been made to the SQL text.

Note: If conversion had been applied, it may be necessary to reverse the changes after optimization when it is pasted back to the original source code.

Trigger Conversion

There are two temporary tables which identifies the set of affected rows prior to the triggering operation (OLD_ TABLE) and which identifies the set of affected rows after the triggering operation (NEW_TABLE) defined the triggers REFERENCING clause. These temporary tables cannot be referenced outside the trigger body; therefore to simulate the SQL statement execution, creation of two temp tables are used to simulate the OLD_TABLE and NEW_TABLE temporary tables.

Trigger

```
CREATE TRIGGER TRG_DEL_TEST1
AFTER DELETE on department
REFERENCING OLD_TABLE AS dept_old
FOR EACH ROW MODE DB2SQL
```

```
BEGIN ATOMIC
  INSERT INTO dept_deleted (cola, dpt_id, dpt_name)
  SELECT 1,
       dpt_id,
       dpt_name
FROM dept_old;
END
```

Scanned SQL

```
INSERT INTO dept_deleted (cola, dpt_id, dpt_name)
SELECT 1,
    dpt_id,
    dpt_name
 FROM dept_old;
```

After conversion

```
DECLARE GLOBAL TEMPORARY TABLE dept_old
LIKE department
NOT LOGGED

INSERT INTO dept_deleted (cola, dpt_id, dpt_name)
SELECT 1,
    dpt_id,
    dpt_name
 FROM dept_old;
```

Related Topics

SQL Scanner Conversions Overview

Parameter Marker Conversion

For some source code, the ? is used to define parameter markers, therefore to enable unique referencing we have changed the name so it is unique within the SQL statement.

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 Topics

SQL Scanner Conversions Overview

Local Variables Conversion

The local variable conversion finds 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 precedes the variable name with "&" and removes the concatenate character and the quotes surrounding the SQL text.

Original SQL statement

```
"SELECT " + VEMPID + " FROM EMPLOYEE WHERE EMP_ID > 100"
```

After conversion

```
SELECT &VEMPID
 FROM EMPLOYEE
WHERE EMP_ID > 100
```

Note: Each local variable in a scanned SQL statement may be replaced with a value or leave the variable in the SQL statement. The SQL Scanner uses "&" to differentiate the local variables from the host variables. The &variablename is not valid syntax for a SQL statement but the program will recognize this format and prompt you to input a value for the variable when needed to execute the SQL statement.

Related Topics

SQL Scanner Conversions Overview

Quoted String Conversion

Quoted String Conversion

The quoted string conversion finds SQL statements that are found in the application source code on one command line. The SQL Scanner removes the concatenate character and the quotes surrounding the SQL text.

Original SQL statement

```
"SELECT *" + " FROM EMPLOYEE WHERE EMP_ID > 100"
```

After conversion

```
SELECT * FROM EMPLOYEE WHERE EMP_ID > 100
```

Related Topics

SQL Scanner Conversions Overview

Local Variable Conversion

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:

- A dash or minus in a variable name
- Comments in the middle of the SQL statement
- The]] (double right square bracket) as the concatenate symbol

This conversion is only applied when the Scanner Job is added to the Job Manager window using the COBOL option under the Select Source Code page in the Add Jobs wizard.

Conversion for variable name

If a variable name contains "-" minus sign, then it will be replaced with "_".

Conversion for comment

If the 7th column of the line is an asterisk (*) then the complete line will be recognized as a line comment.

Conversion for concatenate character

If]] (two right brackets) are used to concatenate column names, they will be replace with a +.

Original SQL statement

```
SELECT * FROM EMPLOYEE
    * Get the department number
 WHERE EMP_ID > :employee-id
  AND ENAME ]] JOB = :name-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 Options window.

Related Topics

SQL Scanner Conversions Overview

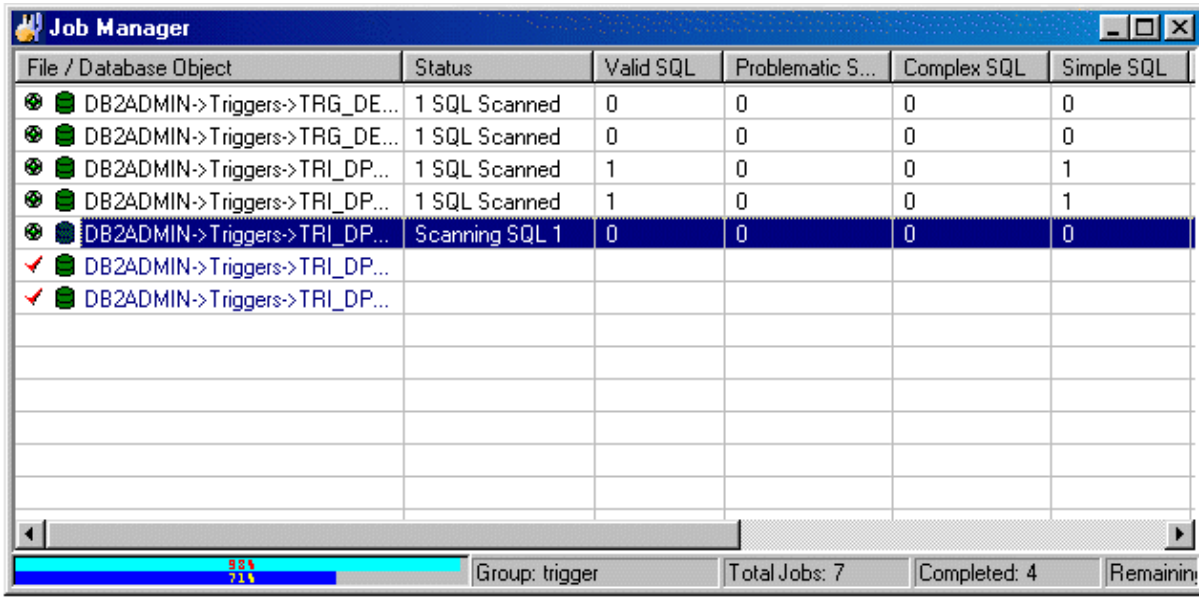
Job Manager

About Job Manager

The Job Manager window provides a work area for the SQL Scanner module. It allows the addition or deletion of items to be scanned. These items, also referred to as Jobs, can be database objects, text/binary files, or COBOL files.

Job Manager Window

View the Job Manager



The Job Manager window consists mainly of two components: grid and status bar.

Grid

The grid from the Job Manager window stores information on each single Job in the Group.

File / Database Object

Displays the job name and associated schema.

Icon	Job Type
	[Schema name]Object type->Object Name
	[Schema name]Text/Binary->Path\File name
	[Schema name]COBOL->Path\File name

Status

Displays the current Job status. This column will remain blank until the Job is scanned. It will show the total 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 SQL Scanner, and for which DB2 LUW can provide an access plan. Valid SQL statements are further classified as Simple, Complex and Problematic SQL statements.

Problematic SQL

Displays the number of Problematic SQL statements found.

If the Show Checked SQL figures on the Job Manager window option is selected within Options, 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.

Complex SQL

Displays the number of Complex SQL statements found.

If the Show Checked SQL figures on the Job Manager window option is selected within Options, 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.

Simple SQL

Displays the number of Simple SQL statements found.

If the Show Checked SQL figures on the Job Manager window option is selected within Options, 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.

File Size

Displays the size of the Job in bytes. This cell will show "(n/a)" for Plan Tables.

Started At

Displays the start date & time of when scanning began.

Processing Time

Displays the total time taken to scan the Job, time is displays in HH24:MM:SS format.

Status Bar

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 Options window. While scanning, a progress bar will be presented 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 Group.
Total Jobs	Total number of Jobs.
Completed	Total number of Jobs already scanned.
Remaining	Total number of Jobs that have not been scanned.

Group Manager Window

In the SQL Scanner, a "group" is used to store "jobs". A Scanner group is a collection of database objects, or text and binary files that you selected to group together. A Scanner job is one file or database object. Before you can start to work in the SQL Scanner, you must create a group.

The Group Manager window is used to create, open, delete and rename Groups. It also displays a list of existing Groups. A Group must be opened to display the Job Manager window. Group Manager provides the following functions:

Function	Description
Open	Opens the selected Group in Job Manager.
Create	Creates a new Group.
Modify	Modifies the 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 Options window.

Open the SQL Scanner

To open the SQL Scanner

1. Click , the first window shown is the Group Manager window. If a Group has not been created before, the Create Group window appears.
2. After creating a new Group or selecting an existing Group, click the **Open**. The Job Manager window appears after opening the working Group. If it is an empty group, the Add Jobs wizard also appears.

Add Scanner Jobs

Add Jobs to the Job Manager Window

The Add Jobs wizard enables you to select the database objects, application source, and binary files that you want to scan.

To add Jobs to the Job Manager window

Click .

Add Options

The following pages are for selecting which database objects, or files to scan.

Database Objects

Source Code

SQL Inspectors

Summary

Note: When you close the Add Jobs wizard, the newly added Jobs are automatically marked in the Job Manager window with a red checkmark in the far left of the row.

Add Database Objects

The SQL Scanner can scan all SQL statements within database objects. These include materialized query tables, views, procedures, functions, triggers, packages and plan tables. Depending on your database access privileges, available database objects can be selected by their object name, object type, or schema name.

To select database objects in the Add Jobs wizard

1. In the Database Objects page of the Add Jobs wizard, double-click each branch name to expand or collapse the associated list of sub-items.
2. Click an individual database object, object type, or schema.
3. Click or drag the item to the list on the right (Selected objects).
4. To remove an item from the right-hand box [Selected objects], click or drag that item out of the list box back to the tree diagram (Database objects).

   or

   Click the button to open the Add Database Objects window.

   a. Select the database, user, and object type.

   b. Using the % wildcard, enter the filtering criteria. The filter is case sensitive, so you must match the upper or lowercase of the database object.

   c. Select the database objects.

Add Source Code

The SQL Scanner can scan through any source code stored as a text or a binary file to identify SQL statements.

To select source code files in the Add Jobs wizard

1. In the Source Code page of the Add Job wizard, select the file type: **Text or binary files** or **COBOL programming source code**.
2. Click .
3. Select the individual files.
4. Click **Open**.
5. Select the schema that corresponds with the SQL that you are scanning from the Schema box.

Add SQL Inspectors Add Jobs (Add Jobs Wizard)

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 **Set Scheme** list to select the schema that corresponds with the SQL that you are scanning.

Related Topic

Add Scanner Jobs

Summary

The Summary page in the Add Jobs wizard in the SQL Scanner shows a list of the jobs that were selected for scanning.

Scan

The SQL Scanner only scans Jobs that are "marked." The Jobs are automatically marked when they are added by the Add Jobs wizard.

To mark a Job

Clicking the Job row. The red checkmark will be prefixed to the **File / Database Object** column. Multiple Jobs can be marked at a time. If there are no marked Jobs, the Scan button is dim and scanning cannot begin.

To unmark a Job

Click the marked row.

To scan the marked Jobs

Click .

During and after scanning, information is updated on the Job Manager window and the sign is replaced with sign. The time required to scan a Job depends on the system's processor speed, the file size and the number of valid and invalid SQL statements. See Scanned SQL Statement Classification for more information about invalid statements.

To abort the scanning process

Click .

Scan Source Code with Temporary Tables

During scanning, if the scanned SQL statement is used to create or modify a temporary table, the SQL will be automatically executed if the Create Scanner Temp Table checkbox in the Options is selected. The created Scanner Temp Tables will be dropped after the job has finished scanning.

Example

Source Code

```
declare global temporary table session.ABC
  like EMPLOYEE
  on commit delete rows
  not logged

select *
  from session.ABC
```

After scanning in the Scanned SQL Viewer, you will see the SQL statement displayed in the Scanned SQL pane as:

```
select *
 from session.ABC
```

The DDL for creating the temporary tables is displayed under the **Scanner Temp Table** tab in the SQL Information. This includes the DDL found by the SQL Scanner or the DDL used to create the User-Defined Temp Table.

```
declare global temporary table session.ABC
 like EMPLOYEE
 on commit delete rows
 not logged
```

Note: To create or modify temporary tables SQL Scanner, the logon user needs the following privileges:

- Connection to DB2 LUW 7 or above
- USE privilege on the USER TEMPORARY table space or SYSADM or DBADM authority.

Abort Scan

To abort the scanning process

Click .

The **Abort Scan** function may take a few seconds to complete, as time is needed to close down all the necessary processes.

Find a Job

To search for a specified text in all Jobs in the Job Manager window

1. Select **Job | Find Job**.
2. Enter the text string to be located.
3. Click **Find**.

All the Jobs and number of the corresponding SQL statements will be listed.

View Group Summary

The Group summary report provides a statistical list of all the Jobs and SQL statements scanned within the Group.

To view the summary information

In the Job Manager **Report | Group Summary** window, select .

View Group Properties

The Group progtfrhnbperties provides information about SQL Scanner Group.

To view the Group properties information

1. Select a Group in the Group Manager window.
2. Right-click and select **Properties**.

The Group Properties window displays the following information:

General tab

Item	Description
Name	Group name.
Description	Group description.
Last modified	Last modified date and time.
Size	Total size of the Jobs within the Group.
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.

View Job Properties

The Job properties provides information about a SQL Scanner Job.

To view the Job properties information

1. In the Job Manager window, select the Job.
2. Right-click and select **Properties**.

The Job Properties window displays the following information:

General tab

Item	Description
Name	Job name.

Item	Description
Type	Type of Job scanned (text/binary file, COBOL file, database object).
Description	Job description.
Last modified	Last modified date and time.
Size	Job size.
Data Directory	Directory where the 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.

Open Group Manager

You can use the Group Manager window to open, create, delete or modify a Group without closing the Job Manager window.

To open the Group Manager from the Job Manager window

Click .

Check Scanned SQL

After scanning is completed, you can review the scanned SQL statement in the Scanned SQL Viewer 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.

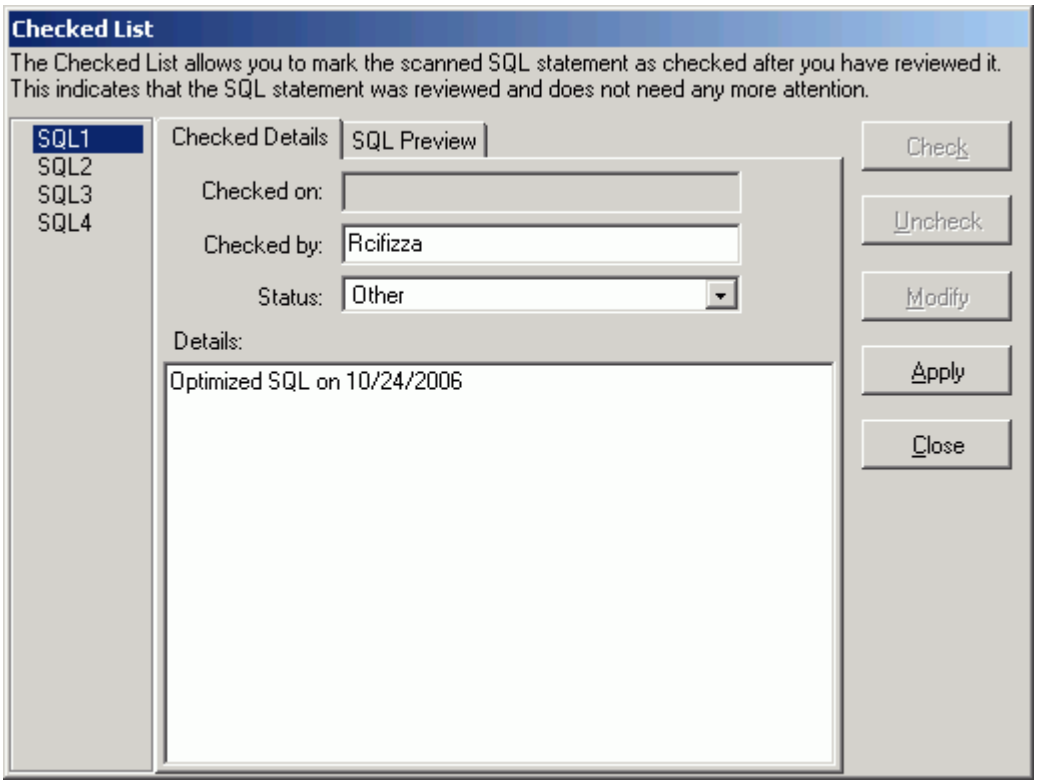


Figure: Checked List Window

You can bring up the Checked List window from both the Job Manager and Scanned SQL Viewer windows.

To open the Checked List window

- From the Job Manager window, select **Job | Checked List** or right-click and select **Checked List** .
- From the Scanned SQL Viewer window, select **SQL | Checked List** or use the right-click and select **Checked List**.

To mark a Scanned SQL statement as checked

1. All the valid SQL statements are listed on the left pane (e.g.) SQL1, SQL2 … SQLn . Select the SQL statement you want to check.
2. To ensured that the SQL statement is the one you want to check, you can view the SQL text by clicking the **SQL Preview** tab.
3. On the **Checked Details** tab, click the **Check** button. All editable fields are changed to edit mode.
4. Modify the **Checked by**, **Status**, and **Details** information.

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 computer user name is entered.

Item	Description
Status	Select the reason why you checked the SQL.
Details	Enter a short description.

5. Click **Apply**.

6. Click **Close**. A blue checkmark appears next to the SQL name at the left pane to indicate that the SQL statement is checked.

You can automatically add a scanned SQL statement to the Checked List when you use the **Send to SQL Optimizer** or **Send to Index Expert** function.

To Add the Scanned SQL Statement to Checked List Automatically

In the Options window on the **SQL Scanner** tab under General button, select **Always add SQL** in the **Checked List** section.

To unmark a checked SQL statement

1. Select the SQL statement you want to unmark from the left pane of the Checked List window.
2. Click **Uncheck**.

The blue checkmark is removed from the SQL name on the left pane.

Preserve Checked SQL Information When Rescanning

If you would like to preserve the Check SQL information when a Job is rescanned , select the **Do not remove Check SQL information when rescanning** checkbox in the **Checked List** section in the Options window on the **SQL Scanner** tab under the **General** button.

Mark and Unmark Jobs

The SQL Scanner only scans Jobs that are "marked."

To mark a Job

Clicking the Job row. The red checkmark will be prefixed to the **File / Database Object** column. Multiple Jobs can be marked at a time. If there are no marked Jobs, the Scan button is dim and scanning cannot begin.

To unmark a Job

Click the marked row.

To mark the selected Jobs

1. Using click, shift click, and control click, highlight Jobs.
2. Right-click and select **Mark Selected Jobs**.

To unmark the selected Jobs

1. Using click, shift click, and control click, highlight Jobs.
2. Right-click and select **Unmark Selected Jobs**.

To mark all Jobs in the Job Manager window

Select **Group | Mark All [Ctrl + M]**.

To unmark all Jobs in the Job Manager window

Select **Group | Unmark All [Ctrl + U]**.

The red checkmark is added to the **File / Database Object** column to indicate that the Job is marked. To show that the Job is unmarked, any appended symbol is removed to indicate that the job is unmarked.

Delete Marked Jobs

To delete Jobs from the Job Manager window

1. Mark the Jobs to be deleted.
2. Select **Group | Delete Marked Jobs**.

or

Press **Delete** .

Move or Copy Jobs to Another Group

To move Jobs in the Job Manager to another Group

1. Mark the Jobs to be moved.
2. Select **Group | Move to Other Group**.
3. Select the Group when you want to move the Jobs in the **Move to** box.

To copy Jobs in the Job Manager to another Group

1. Mark the Jobs to be copied.
2. Select **Group | Copy to Other Group.**
3. Select the Group when you want to move the Jobs in the **Copy to** box.

Modify Job Details

You can modify the Job description, change the file type for source code files, and eliminate duplicate SQL while scanning.

Modifying a Job deletes the current scanning information.

To modify the Job

1. Mark the Jobs to be modified.
2. In the Job Manager window,right-click and select **Modify**.
3. The Modify Job dialog box allows you to do the following:

Item	Description
Description	Add an comment about the Job. This description is only displayed in Modify Job window.
Job Type	When the Job is a source code file, select either **Text or binary files** or **COBOL programming source code**.
Eliminate duplicate SQL	Check **Eliminate duplicate SQL** to only extract the identical SQL statement once for each Job.

4. Click **OK**.
5. Since modifying the Job deletes the current scanning information, click to rescan the Job if desired.

Switch Schema

The Schema that is used to retrieve the access plan when the SQL statement is scanned is displayed in the File / Database Object column in the Job Manager window.

To change the schema used to scan a particular Job

1. Right-click a Job and select **Set Schema**.
2. Select the schema name from the list.

Place Bookmarks in the Job Manager Window

Bookmarks are used to mark a Job in the Job Manager 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 has a bookmark, the complete row text is displayed in fuchsia color. You can have multiple bookmarks.

To add or remove bookmark in the Job Manager window

Right-click a Job and select **Add/Remove Bookmark**.

Create Benchmark Factory Import File

All the SQL statements can be saved in a text file from the Job Manager, the Scanned SQL Viewer, the SQL Repository, and the SQL Optimizer windows. These SQL statements can then be imported into Benchmark Factory program (version 4.6 or later).

To create the text 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. Click **OK**.
3. Enter the filename and select the file location. Click **Save**.

Job Manager Functions

Below is a list of available functions within the Job Manager window.

Button or Menu	Function
Group Menu	Group Manager
Group & Right-click Menu	Add Jobs
Group 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 & Right-click Menu	Scanned SQL Viewer
Job Menu	Find Job
Job & Right-click Menu	Checked List
Report Menu	Group Summary

Button or Menu	Function
Right-click Menu	View (Large Icons/ Small Icons/ List/ Details)
Right-click Menu	Add/Remove Bookmark
Right-click Menu	Modify
Right-click Menu	Delete
Right-click Menu	Set Schema
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

Scanned SQL Viewer

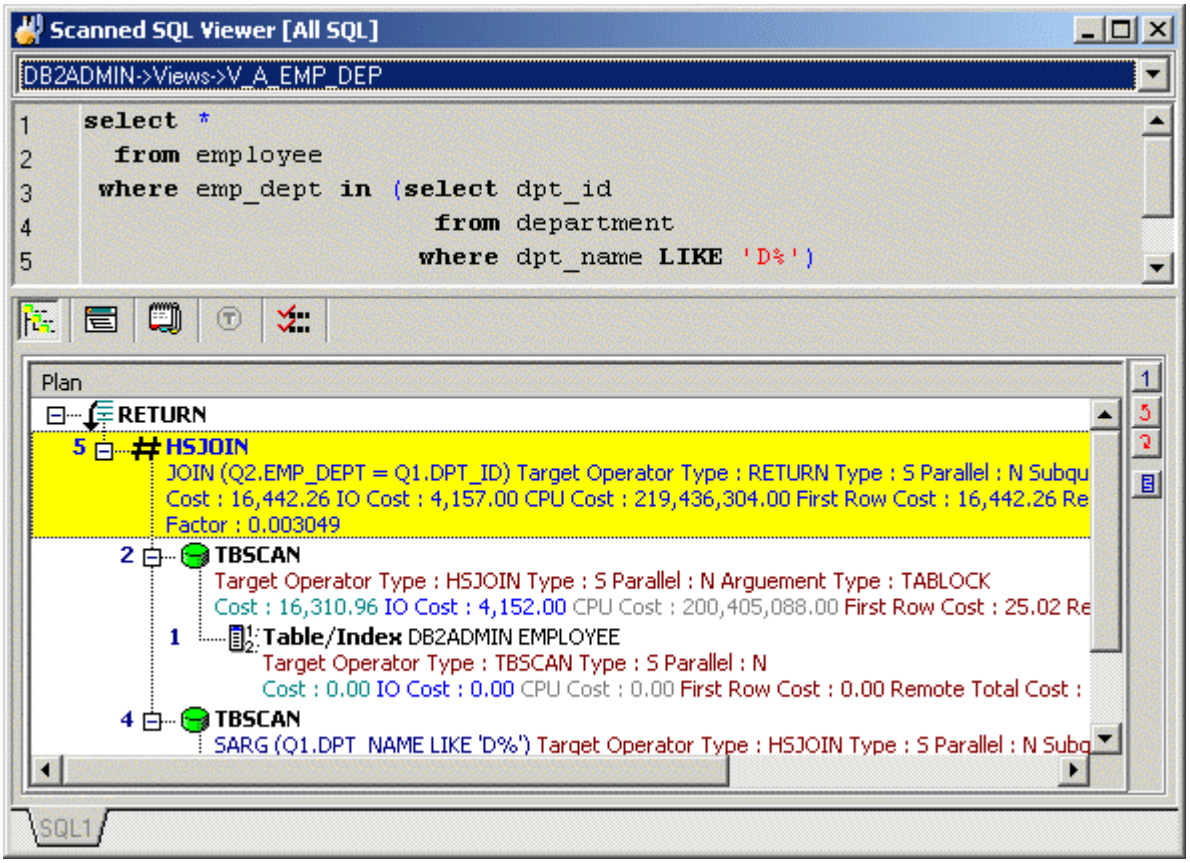
About Scanned SQL Viewer

The Scanned SQL Viewer window enables the viewing and analysis of the SQL statement scanned. It displays the formatted scanned SQL statement, access plan, temp table (if applicable), and information about the SQL statement.

- Formatted scanned SQL statement
- Access plan
- 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 access 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

Scanned SQL Viewer Window

View the Scanned SQL Viewer



The Scanned SQL Viewer window is divided into these sections:

Job list

The Job list at the top of the Scanned SQL Viewer window displays the Job description of the selected Job and contains the list of Jobs that satisfy the view criteria.

Scanned SQL (Top Pane)

The top pane of the Scanned SQL Viewer window displays the scanned SQL statement for a particular scan Job. If the SQL statement is valid, the scanned SQL statement will be laid out according to the indentation algorithm. Otherwise, the scanned SQL statement will be unformatted.

If more than one SQL statement is found in a Job, multiple tabs for selecting the SQL are displayed at the bottom right corner of the window.

SQL Information (Bottom Pane)

The SQL Information pane has buttons at the top of it to select whether to display the access plan graph, access plan tree, DB2 Optimized Text, Information (the classification of the SQL, the location in the file or database object and login information), the scanner temp table DDL, and the checked SQL details. If the button is enabled, there is information on that page. If the button icon is grayed, then no information is available for that SQL statement.

Understand the Access Plan

If the scanned SQL statement is valid, the access plan for the SQL statement will be retrieved. The access plan is a combination of steps the DB2 LUW optimizer selects to execute the SQL statement. It is a sequential set of steps required to carry out the query, complete with the access methods chosen for each table.

Examination of the access plan shows how the database executes the SQL statement and aids in the analysis of whether the SQL statement is the most efficient.

The access plan help provides online help for each operation within the access plan to help you understand each step of the plan.

To view the help for an operation

Right-click the access plan step for which you want help and select **Get help on operation_name**.

Open Scanned SQL Viewer

Once the Job has been scanned and the SQL statements found, view the SQL statements through the Scanned SQL Viewer.

To open the Scanned SQL Viewer window

- Click

  or
- Double-click the Job

View a Particular Type of SQL Statement

When the Scanned SQL Viewer window is first opened 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 Scanned SQL Viewer window displays the chosen criteria. One or more types of SQL statements can be viewed by selecting or de-selecting the menu items. The Job list contains all the Jobs using the chosen view criteria:

To display all SQL statements

Select **View | All SQL**.

To display the Simple SQL statements

Select **View | Simple SQL**.

To display the Problematic SQL statements

Select **View | Problematic SQL**.

To display the Complex SQL statements

Select **View | Complex SQL**.

To display the invalid SQL statements

Select **View | Invalid SQL**.

To display SQL statements with different bound and new access plan that satisfy the selected SQL classification type

Select **View | Different Bound and New Access Plan**.

To display SQL statements with no bound access plan that satisfy the selected SQL classification type

Select **View | No Bound Access Plan**.

To displays all Checked SQL statements

Select **View | Checked SQL**.

To displays all Unchecked SQL statements

Select **View | Unchecked SQL**.

Job Navigation

There are several ways to navigate through the Jobs in the Scanned SQL Viewer window:

- Double-click the Job to be viewed in the Job Manager window.
- Select the Job from the Job list.
- Use the navigation buttons on the toolbar or corresponding menu items.

Button	Menu Item	Description
	Job | First Job	Display the first Job.
	Job | Previous Job	Display the previous Job.
	Job | Next Job	Display the next Job.

Button	Menu Item	Description
	Job | Last Job	Display the last Job.

Generate Report for Scanned SQL Statements

To generate a report for scanned SQL statements

1. With the Scanned SQL Viewer window open select **Report | Scanned SQL** to open the Scanned SQL Report Criteria window.
2. Select the report criteria.
3. The information in the report can be saved or printed.

Note: A few minutes may be needed to generate a long report.

Create Benchmark Factory Import File

All the SQL statements can be saved in a text file from the Job Manager, the Scanned SQL Viewer, the SQL Repository, and the SQL Optimizer windows. These SQL statements can then be imported into Benchmark Factory program (version 4.6 or later).

To create the text 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. Click **OK**.
3. Enter the filename and select the file location. Click **Save**.

Scanned SQL Viewer Functions

Below is a list of available functions within the Scanned SQL Viewer window.

Button or Menu	Function
Navigate Menu	First SQL/ Previous SQL/ Next SQL/ Last SQL
Navigate Menu	First Job/ Previous Job/ Next Job/ Last Job
Navigate Menu	Go to SQL
Edit Menu	Send to SQL Rewrite

Button or Menu	Function
Edit Menu	Send to Generate Indexes
File Menu	Save SQL to SQL Repository
SQL & Right-click Menu	Checked List
SQL Menu	Find SQL
SQL Menu	Find Next SQL
View Menu	All SQL
View Menu	Simple SQL
View Menu	Complex SQL
View Menu	Problematic SQL
View Menu	Invalid SQL
View Menu	Different Bound and New Access Plan
View Menu	No Bound Access Plan
View Menu	Checked SQL
View Menu	Unchecked
Report Menu	Scanned SQL Report
Right-click Menu	Create Benchmark Factory Import File

SQL Optimizer

SQL Optimizer Concepts

About the SQL Rewrite Function

The SQL Rewrite function analyzes the input SQL statement and uses an Artificial Intelligence Engine to produce a group of semantically equivalent versions of the statement, known as SQL alternatives. You can then test run these alternatives in the SQL Optimizer window to determine the best-performing version of the SQL.

Tip: After you run SQL Rewrite, run the Generate Indexes function to add alternatives that incorporate virtual indexes. You can then test run the SQL alternatives along with index-set alternatives to measure performance. See Generate Index-Set Alternatives for more information.

The SQL Rewrite process includes these phases:

Show Plan

Retrieves the access plan and the DB2 optimized text that the DB2 LUW optimizer has chosen for the specific SQL alternative. SQL Rewrite classifies the SQL statement according to characteristics that cause performance problems.

Rewrite

Rewrites the original SQL statement to produce a list of semantically equivalent versions, or SQL alternatives.

Run Time/Batch Run

Allows the users to test run the original and the SQL alternatives to select which SQL version gives the best performance. For each alternative, Batch Run provides the run time for retrieving all records and for retrieving only the first record. You can use this information to identify which alternative is most suitable for your application.

Execute

Executes alternatives against the original SQL and displays the returned data for comparison in the SQL Optimizer window.

Related Topics

SQL Optimizer Tutorial

SQL Optimizer Window

SQL Optimizer Functions

What Function Should I Use to Retrieve the Run Time?

The SQL Optimizer provides two different measurements of performance; the run time for retrieving all records and the run time for retrieving the first *n* records. Both measurements give you an indication on the fastest running SQL statement—but with two different aims. You must understand the use of the SQL statement in the application. Generally, if the SQL statement is used for reports, then you should use the **Run for All Record** or **Batch Run** with **Run Time Mode** set to **All Records**. If the SQL statement is used for online query, then use the **Run for First Record** or **Batch Run** with **Run Time Mode** set to **First n Record**.

Note: If the aim of the SQL statement is unknown, then use **All Records** as a performance indication.

Unsatisfactory Performance Results

After optimization and execution testing, 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 Options window and optimize again to ensure all transformed SQL statements are given. Rerun the SQL statement optimization after the review.

You can also review the access plan of the optimized SQL statement to check if there should be any alterations to the database structure such as adding a new index. The Index Expert module is used to generate alternative Index Sets for a SQL statement.

SQL Functions

The following SQL Functions are available in the SQL Optimizer window to retrieve the access plan, the run time, result set and to convert parameter markers:

Show Plan

Run Result (SQL alternatives only)
Run for First Record
Record for All Records
Convert Parameter Markers
Test for Scalability

SQL Optimizer Functions

Below is a list of available functions within the SQL Optimizer window.

These functions are available only for the original SQL statement <Edit SQL> when displayed in the editable SQL Text window.

Button or Menu	Function
SQL Menu	Optimize/Abort Optimize
SQL Menu	Show Plan
SQL Menu	Convert Parameter Markers
File Menu	Open SQL from SQL Repository

These functions are available for original SQL and its alternatives.

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
Right-click Menu	Plan Help
SQL Menu	Open Optimized SQL
File Menu	Save SQL to SQL Repository
SQL Menu	Add User-Defined SQL
SQL & Right-click Menu	Test for Scalability
Right-click Menu	Create Benchmark Factory Import File

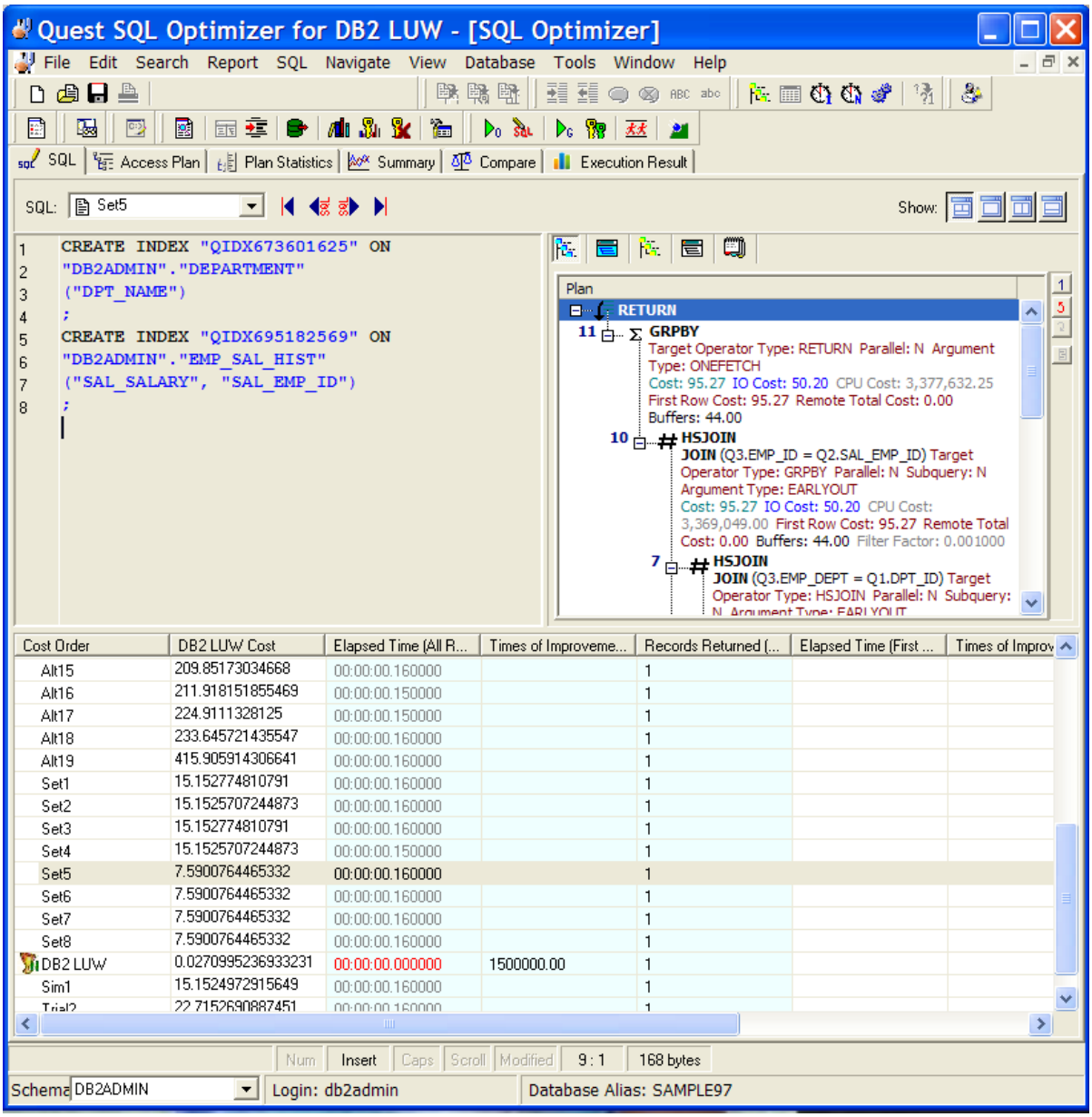
These functions are available only after the original SQL statement has been rewritten or virtual indexes have been generated.

Button or Menu	Function
SQL Menu	Batch Run/Abort Batch Run
SQL Menu	Save Optimized SQL
Search Menu	Find SQL. . . /Find Next SQL
	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
View Menu	Show Open Optimized SQL Details
View Menu	Show Refresh Plan Details

SQL Optimizer Window

SQL Optimizer Window

View the SQL Optimizer Window



SQL Optimizer helps you optimize an SQL statement. That is, it rewrites the SQL to produce tweaked versions, or alternatives, of the original SQL and suggests indexes, all of which you can test to determine whether performance improves.

The SQL Optimizer window is the point at which you start the optimization process. This window shows the original SQL statement that you have entered manually or extracted from another facility, such as Toad or SQL Scanner. From this window, run either or both of these functions to optimize the SQL:

- SQL Rewrite function to create virtual SQL alternatives
- Index Expert function to create virtual index sets that display as alternatives as well

To help you determine the best-performing version of your SQL, the SQL Optimizer window lets you do the following:

- Review the access plan, plan statistics, and optimized SQL for the original SQL and each alternative
- Execute the original SQL and the index-set and SQL alternatives to find the fastest alternative
- Compare SQL text, access plans, run-time statistics, costs, and other overhead of the original SQL to the index-set and SQL alternatives, as well as compare alternatives with each other
- Compare the execution results of the original SQL to the index-set and SQL alternatives

The SQL Optimizer window consists of the following:

SQL Tab

Access Plan Page

Plan Statistics Tab

Summary Tab

Compare Tab

Execution Result Tab

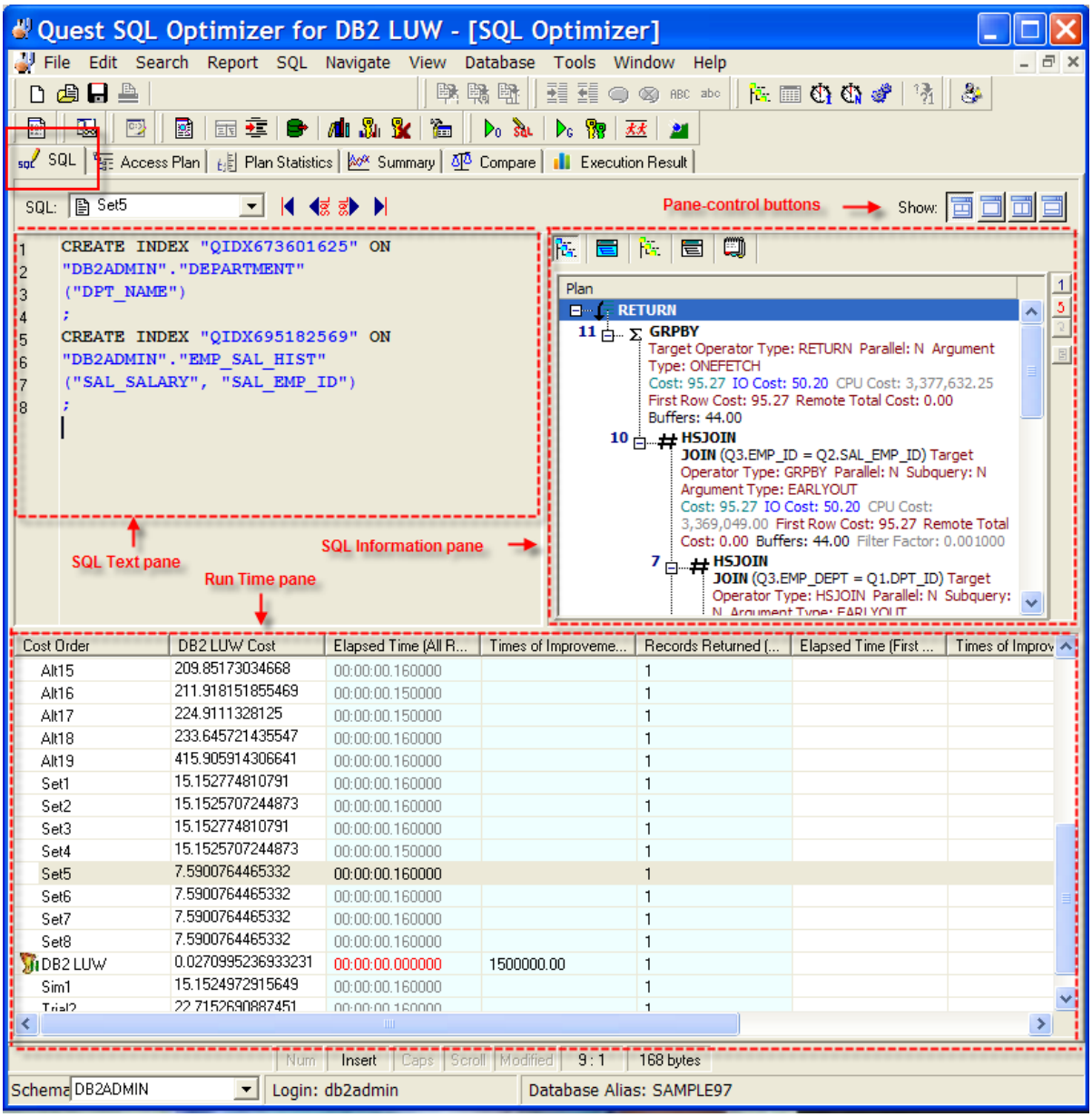
Notes:

- You can optimize only one SELECT, DELETE, UPDATE, or INSERT SQL statement in the SQL Optimizer window.
- To optimize SQL statements within database objects, SQL PL , or other application source code, use the SQL Scanner module to identify potential problematic SQL statements within the code. Then optimize each SQL statement in the SQL Rewrite function.
- In Toad, to open a SQL Optimizer session on the SQL in a view, trigger, or MQ table, click

in the Create or Alter window for the object.
- In Toad, to open a SQL Optimizer session on an SQL statement in a package, click on the SQL tab in the Database Explore or Object Explorer for the specific package.

SQL Tab

View the SQL tab



The SQL tab in the SQL Optimizer window consists of three panes: SQL Text, SQL Information and Run Time. On this tab, perform any of the following:

- Enter or edit the original SQL statement. (See Enter or Edit the Original SQL Statement.)
- View the SQL alternatives that the Optimizer generates. (See Automatically Rewrite the Original SQL Statement.)
- View the index-set alternatives that the Index Expert function generates. (See Generate Index-Set Alternatives.)
- Create your own the SQL alternatives. (See Add User-Defined SQL Alternatives.)
- View the index-set alternatives that you create. (See Add Your Own Virtual Index Sets.)

- Review the access plan for the original SQL and for each SQL or index-set alternative.
- Review the run-time statistics after all alternatives are batch-executed.

SQL Text Pane

Use the SQL Text pane to view the original SQL statement syntax or the SQL associated with a specific SQL alternative. For an index-set alternative, view the index DDL. You can also use this pane to enter or edit the text of the original SQL statement or to enter the text of an SQL alternative you are manually creating.

SQL Information Pane

Use the SQL Information pane to toggle between the access plan or the DB2 optimized text for the original SQL or an alternative. You can also view information about the original SQL.

Run Time Pane

The Run Time pane displays the SQL or index-set alternatives automatically generated or created manually on the original SQL. Use the SQL Navigation buttons to move between alternatives. When you perform a Batch Run, this list shows the run time statistics for retrieving all records and for retrieving the first *n* records for the original SQL and each alternative.

Display or Hide Panes

Use the pane-control buttons to reorganize the panes:

To display the SQL Text, SQL Information, and Run Time panes

Click .

To display only the SQL Text pane

Click .

To display the SQL Text and SQL Information panes

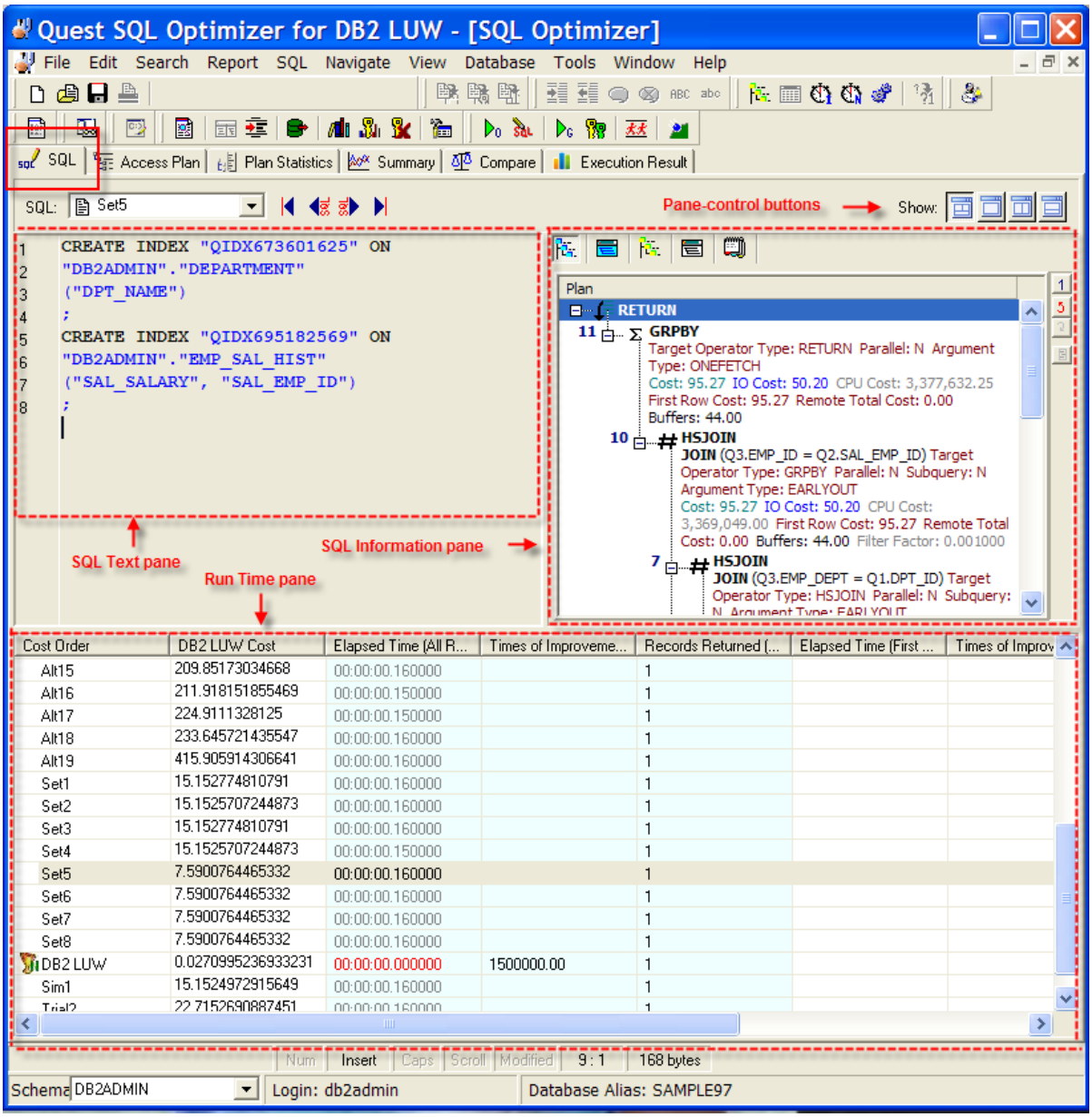
Click .

To display the SQL Text and Run Time panes

Click .

SQL Tab

View the SQL tab



The SQL tab in the SQL Optimizer window consists of three panes: SQL Text, SQL Information and Run Time. On this tab, perform any of the following:

- Enter or edit the original SQL statement. (See Enter or Edit the Original SQL Statement.)
- View the SQL alternatives that the Optimizer generates. (See Automatically Rewrite the Original SQL Statement.)
- View the index-set alternatives that the Index Expert function generates. (See Generate Index-Set Alternatives.)
- Create your own the SQL alternatives. (See Add User-Defined SQL Alternatives.)
- View the index-set alternatives that you create. (See Add Your Own Virtual Index Sets.)

- Review the access plan for the original SQL and for each SQL or index-set alternative.
- Review the run-time statistics after all alternatives are batch-executed.

SQL Text Pane

Use the SQL Text pane to view the original SQL statement syntax or the SQL associated with a specific SQL alternative. For an index-set alternative, view the index DDL. You can also use this pane to enter or edit the text of the original SQL statement or to enter the text of an SQL alternative you are manually creating.

SQL Information Pane

Use the SQL Information pane to toggle between the access plan or the DB2 optimized text for the original SQL or an alternative. You can also view information about the original SQL.

Run Time Pane

The Run Time pane displays the SQL or index-set alternatives automatically generated or created manually on the original SQL. Use the SQL Navigation buttons to move between alternatives. When you perform a Batch Run, this list shows the run time statistics for retrieving all records and for retrieving the first *n* records for the original SQL and each alternative.

Display or Hide Panes

Use the pane-control buttons to reorganize the panes:

To display the SQL Text, SQL Information, and Run Time panes

Click .

To display only the SQL Text pane

Click .

To display the SQL Text and SQL Information panes

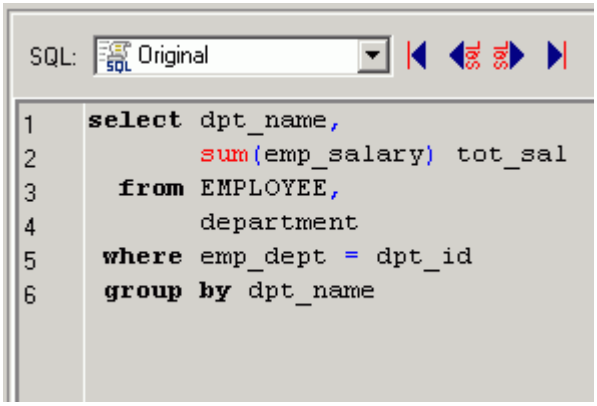
Click .

To display the SQL Text and Run Time panes

Click .

SQL Text Pane

View the SQL Text pane



Use the SQL Text pane to do any of the following:

- Enter or edit the original SQL statement syntax (when *<Edit SQL>* is selected in the Run Time pane). See Enter or Edit the Original SQL Statement for more information.
- Display the SQL statement associated with the SQL alternative selected in the Run Time pane.
- Display the DDL for the index-set alternative selected in the Run Time pane.
- Enter SQL statement text for an SQL alternative you are creating. See Add User-Defined SQL Alternatives for more information.

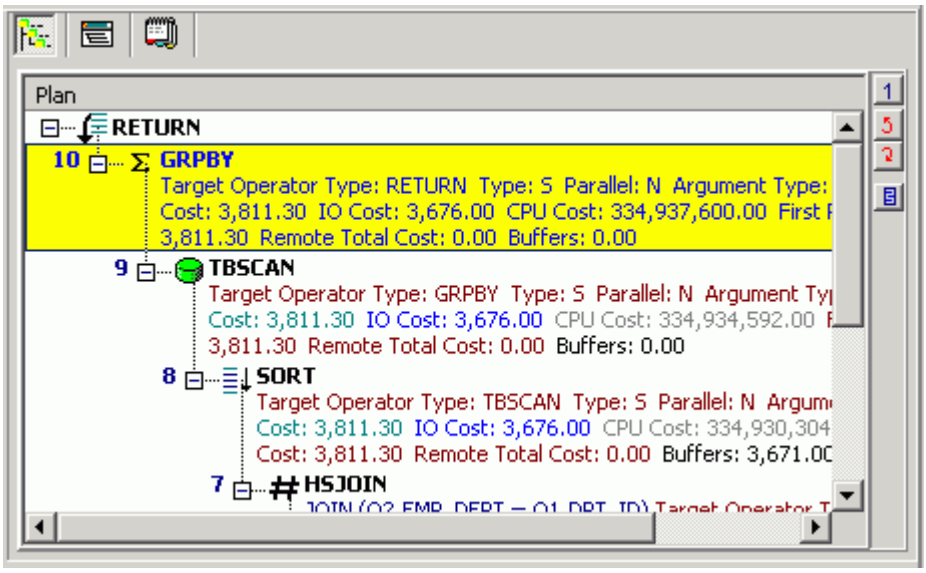
When a SQL statement is sent to this pane from other modules, the optimization process begins automatically (if you have selected .

Notes:

- All variables in the SQL text are displayed in red (default color), which indicates that a value and data type need to be defined before the SQL statement is executed. Other color coding in the SQL text is determined by the settings for syntax highlighting under the Editor tab of the Options window.
- You can enter comments in the original SQL statement text, but not in the SQL for the alternatives you manually create.

SQL Information Pane

View the SQL Information pane



The SQL Information pane shows types of information for the original SQL or the SQL associated with the currently selected alternative. Use the buttons across the top of the pane to toggle between the views.

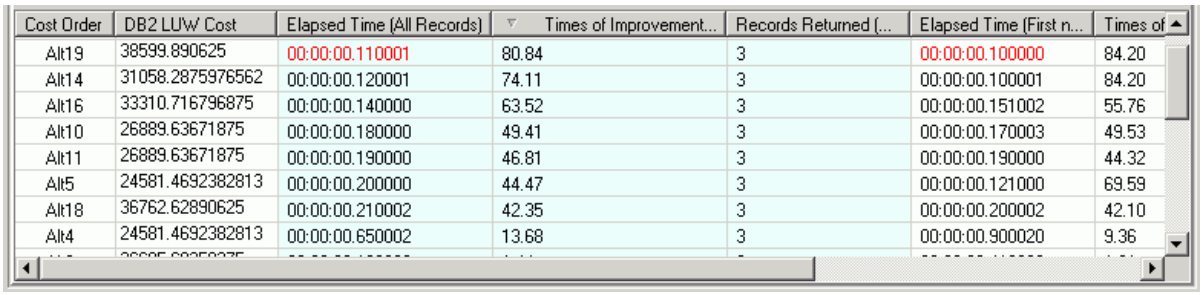
Note: Different SQL Optimizer modules use the SQL Information pane. The buttons available in the pane depend on the module you are using.

Button	Function	Description
	Virtual Access Plan Tree	Show the corresponding virtual access plan. **Notes:** • No information displays if the SQL statement in invalid. • For index-set alternatives, this is the assumed plan should the indexes actually exist.
	Virtual DB2 Optimized Text	If the SQL statement is valid, show the corresponding virtual DB2 optimized text. **Note:** No information displays if the SQL statement in invalid.
	Access Plan and Bound Access Plan	Display the access plan. **Notes:** • No information displays if the SQL statement in invalid. • For index-set alternatives, this is the actual plan used during Batch Execution (or anther execution function) when temporary indexes were physically created . • In the Scanner SQL Viewer, both the bound plan and the current plan display when the SQL statement is from a package. **Tip:** Click (if available)to display detailed information for each row of the access plan.
	DB2 Optimized Text and Bound DB2 Optimized Text	Display the SQL statement reconstructed by the DB2 LUW optimizer after it retrieves the access plan. **Note:** In the Scanner SQL Viewer, the optimized text for both the bound plan and the current plan display.

Button	Function	Description
	Information	Show any of the following, depending on the specific function using this pane: • SQL statement type classification: Problematic, Complex, Simple, or Invalid SQL. This classification is dependent on the parameters set in the Options window. • (SQL Scanner) For a SQL statement in a package, a comparison of the bound access plan and the current access plan. A database error message is displayed if SQL is classified as *Invalid*. • (SQL Optimizer) Warning or alert information about the SQL statement if the transformation is based on table constraints or indexes. **Note:** Changes to table constraints and indexes might have a direct effect on the optimized SQL statement. • (SQL Optimizer)Origin of the SQL statement. • (SQL Scanner) SQL conversion applied. • (SQL Scanner)Start position of the SQL statement in DDL and in TXT and SQL files. • (SQL Scanner) For bound access plans, package information. • (SQL Scanner) For bound access plans, new access plan information. • (SQL Scanner) Connection information. • Special register settings.
	Scanner Temp Table (SQL Scanner only)	Display 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 code.
	Checked SQL (SQL Scanner only)	Display the date and time when the SQL statement was checked, its status and description, and the name of person who checked the SQL.

Run Time Pane

View the Run Time pane



Cost Order	DB2 LUW Cost	Elapsed Time (All Records)	Times of Improvement...	Records Returned (...	Elapsed Time (First n...	Times of
Alt19	38599.890625	00:00:00.110001	80.84	3	00:00:00.100000	84.20
Alt14	31058.2875976562	00:00:00.120001	74.11	3	00:00:00.100001	84.20
Alt16	33310.716796875	00:00:00.140000	63.52	3	00:00:00.151002	55.76
Alt10	26889.63671875	00:00:00.180000	49.41	3	00:00:00.170003	49.53
Alt11	26889.63671875	00:00:00.190000	46.81	3	00:00:00.190000	44.32
Alt5	24581.4692382813	00:00:00.200000	44.47	3	00:00:00.121000	69.59
Alt18	36762.62890625	00:00:00.210002	42.35	3	00:00:00.200002	42.10
Alt4	24581.4692382813	00:00:00.650002	13.68	3	00:00:00.900020	9.36

The Run Time pane in the SQL and Statistics pages in the SQL Optimizer window lists the current SQL and index-set alternatives generated for the original SQL. The following shows for each alternative.

Notes:

- Execute Batch Run to capture a complete set of statistics.
- Initially the alternatives are listed by **DB2 LUW Cost** in ascending order.

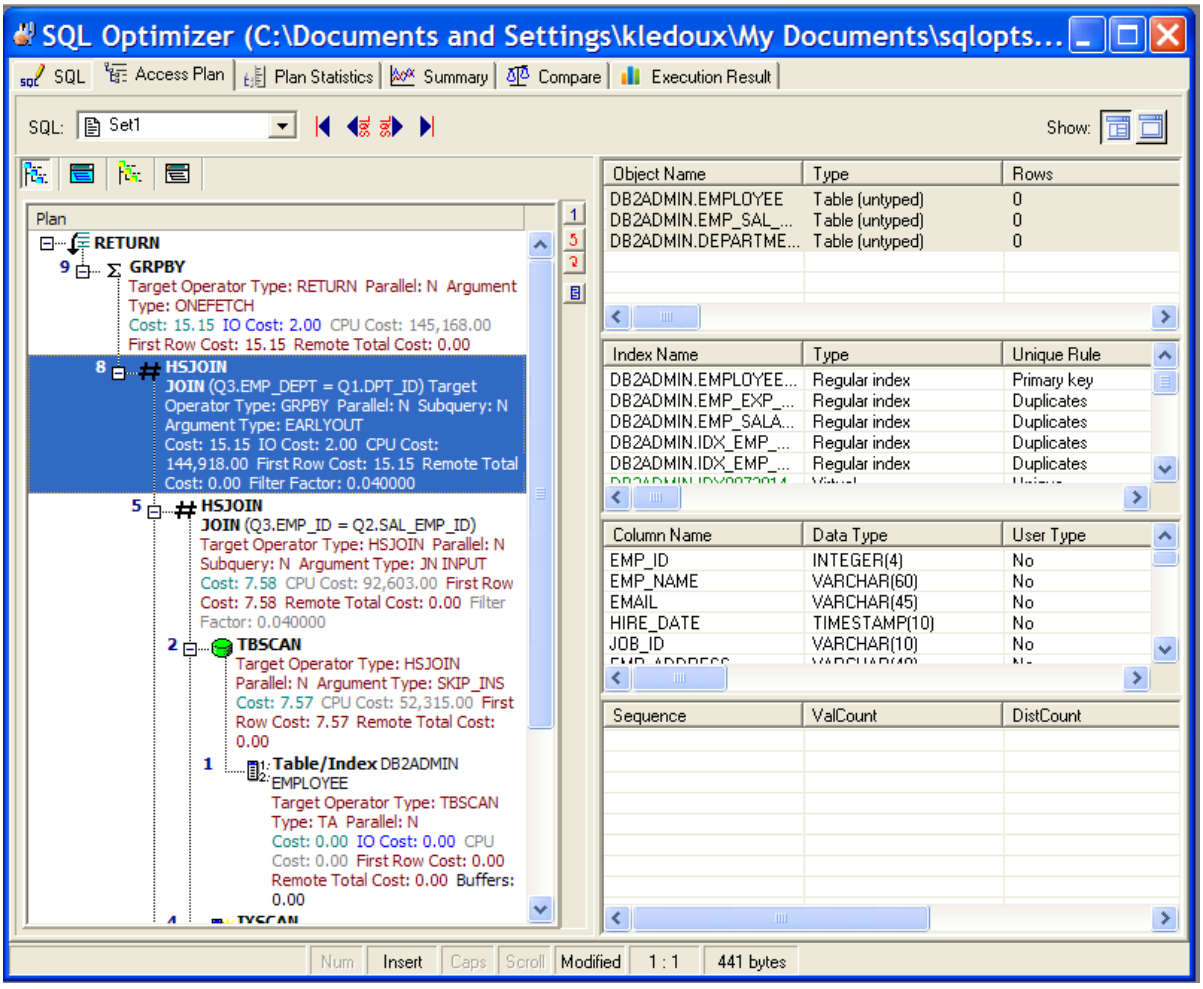
Item	Description
Cost Order	The order of alternatives is initially based on the ascending order of DB2 LUW Cost. The original SQL statement will always be the first item on the grid.
DB2 LUW Cost	The cost value provided by DB2 LUW as an estimate of performance. **Note:** The lower the DB2 LUW cost, the better the estimated performance of the SQL statement. However, the cost value should not be used as the actual indication of performance. It is best to execute the alternatives to find which performs the best in your database environment.
Elapsed Time (All Records)	The elapsed time for all records shows the actual elapsed time required to retrieve all records from the database. The calculation of the run time is based on the CPU time of the database server. Thus network traffic is excluded from the time. If the SQL statement is used to retrieve all the records from the database, such as reports or batch processes, the SQL statement with the best elapsed time for all records should be used as a criteria for selecting the alternative.
Times of Improvement (All Records)	The times of improvement shows how many times faster the alternative is for retrieving all records than the original SQL statement.
Records Returned (All Records)	The number of records indicates the total number of records influenced by the alternative. This figure should remain constant throughout the original and optimized SQL statements.
Elapsed Time (First n Records)	The elapsed time for the first record indicates the time it takes for the first record to be returned from the SQL statement. For some on-line retrieval screens, interactive applications, or processes that do not retrieve all records from the SQL statement at once, the best elapsed time for retrieving the first record should be used as a criteria for selecting the best alternative.
Times of Improvement (First n Records)	The times of improvement shows how many times faster the alternative is for retrieving the first record than the original SQL statement.
Records Returned (First n Records)	The first record retrieved, this figure should be 0 or 1.
Remarks	Information from the Batch Run is included in this column. It includes: • If the alternative was terminated by the termination criteria.

	• If the alternative was run more than once. • If a database error occurred. • If the number of records retrieved by the alternative does not match the number of records retrieved by the original SQL statement.

Access Plan Page

Access Plan Tab

View the Access Plan tab



The Access Plan tab in the SQL Optimizer window uses the following panes to display access plan details for the currently selected alternative and information about the objects that the alternative syntax accesses:

Access Plan

Object Statistics
Index Statistics
Column Statistics
Column Values Distribution Statistics

Regenerate Virtual Index Sets

You can rerun the Generate Indexes function to obtain the latest index recommendations.
Note: This process will clear all existing SQL and index-set alternative in this SQL Optimizer session.

To regenerate virtual index sets

Right-click within any of the statistics panes, and select **Recommend Virtual Indexes**.

Create Your Own Virtual Index Sets

You can add your own index sets to this SQL Optimizer session.

To create your own virtual index sets

Right-click within any of the statistics panes, and select **Create Virtual Indexes**.

Display or Hide Panes

Use the pane-control buttons to reorganize the panes.

To display the Access Plan, and the Table, Index, Column, and Column Value Distribution panes

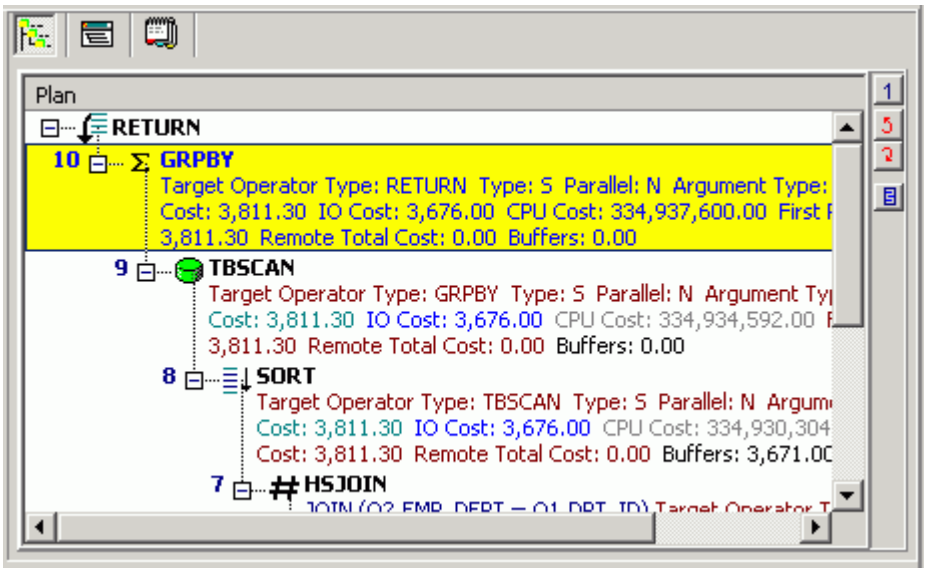
Click .

To display only the Access Plan pane

Click .

Access Plan

View the Access Plan pane



The access plan is a combination of steps the DB2 LUW database optimizer chooses to execute a SQL statement. Each node represents how the database optimizer will physically retrieve rows of data from the database or how the data is prepared. By examining the access plan, you can see exactly how the database executes your SQL statement.

Right-click Menu

The Access Plan pane contains a right-click menu that allows you to perform the following functions:

Function	Description
Print	Sends the access plan in its current view to the printer, to display on the screen (print preview), or to a file.
Copy	Copies the access plan to the clipboard.
View Plan	Changes how the access plan is displayed.
Animated Plan Steps	Highlights one-by-one the access plan steps.
Plan Options	Opens the Access Plan Options window so you can select the specific detailed information that is displayed in the access plan. You can also choose to display specific information in individual columns.
Get Help on *plan_step*	Displays the help text for the currently selected step in the access plan.
Help on Access Plan	Opens online help for the access plan.

Related Topics

Access Plan Functions

Review Access Plans

Plan Detail

Plan Help

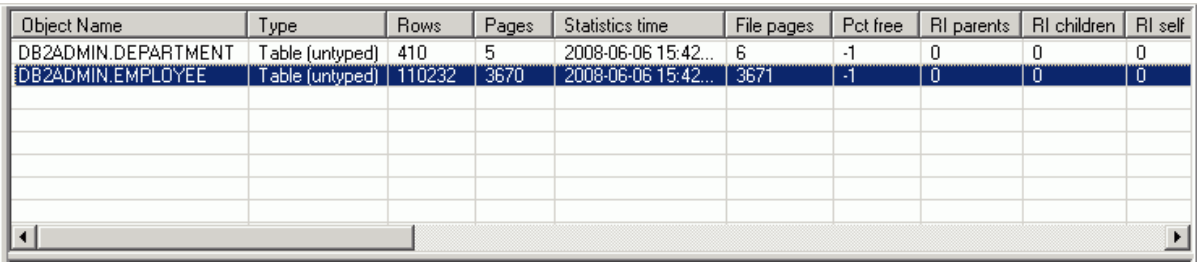
Animate Access Plans

Copy Access Plans

DB2 Optimized Text

Object Statistics Pane

View the Object Statistics pane



Object Name	Type	Rows	Pages	Statistics time	File pages	Pct free	RI parents	RI children	RI self
DB2ADMIN.DEPARTMENT	Table (untyped)	410	5	2008-06-06 15:42...	6	-1	0	0	0
DB2ADMIN.EMPLOYEE	Table (untyped)	110232	3670	2008-06-06 15:42...	3671	-1	0	0	0

The Object Statistics pane is displayed on the Access Plan tab in the SQL Optimizer window. It lists information for each table, view, or alias referenced in the access plan. If the operation in the selected access plan step accesses a specific object, this object is automatically highlighted in the Table Statistics pane. You can do the following in this pane:

- Select an object in this pane to view statistics about its indexes and columns (in the Index and Column Statistics panes, respectively).
- Right-click a specific object to update its catalog statistics.

Append mode

The method in which rows are inserted on pages in the table or view:

Value	Description
Rows appended	New rows are inserted at the end of the data.
Rows inserted	New rows are inserted into existing spaces if available.

Check count

The number of check constraints defined on the table or view.

This information is based on the CHECKCOUNT value for the table or view in SYSCAT.TABLES.

Col count

The number of columns in the table or view.

This information is based on the COLCOUNT value for the table or view in SYSCAT.TABLES.

File pages

Total number of pages allocated for this table.

This information is based on the FPAGES value for the table or view in SYSCAT.TABLES.

Index tablespace

The name of the tablespace that holds all the indexes created for the table or view.

This information is based on the INDEX_TBSPACE value for the table or view in SYSCAT.TABLES.

Key columns

The number of columns that make up the primary key for the table or view.

This information is based on the KEYCOLUMNS value for the table or view in SYSCAT.TABLES.

Key unique

The number of unique constraints (other than a primary key constraint) defined for the table or view.

This information is based on the KEYUNIQUE value for the table or view in SYSCAT.TABLES.

LOB tablespace

The name of the tablespace that holds all long data (for LONG or LOB column types) for the table or view.

This information is based on the LONG_TBSPACE value for the table or view in SYSCAT.TABLES.

Lock size

The preferred lock granularity when DML statements are executed on the table:

Value	Description
Row	Row-level lock
Lock	Table-level lock
blank	Not applicable

Object name

The qualified name of the table or view.

This information is based on the TABSCHEMA and TABNAME values for the table or view in SYSCAT.TABLES.

Overflow

Total number of overflow records in the table or view. An overflow record is an updated record that is too large to fit on the page in which it is currently stored. The record is copied to another page, and its original location is replaced with a pointer to the new page.

This information is based on the OVERFLOW value for the table or view in SYSCAT.TABLES.

Pages

Total number of pages that contain data for the table or view.

This information is based on the NPAGES value for the table or view in SYSCAT.TABLES.

Pct free

Percentage of each page in the table or view to be reserved for future inserts.

This information is based on the PCTFREE value for the table or view in SYSCAT.TABLES.

Part mode

The mode used to distribute data in the table or view within a partitioned database:

Value	Description
Hash on partition key	Hash on the partitioning key
Replicate across partitions	Replicate table or view data across partitions
blank	No partitioning. The table, view, or alias exists in a single partition nodegroup with no partitioning key defined. A blank also appears for nicknames.

RI children

The number of referential constraints in which the table or view is a parent (that is, the number of dependent tables of this table).

This information is based on the CHILDREN value for the table or view in SYSCAT.TABLES.

RI parents

The number of referential constraints in which the table or view is a dependent (that is, the number of parent tables of this table).

This information is based on the PARENTS value for the table or view in SYSCAT.TABLES.

RI self

The number of referential constraints in which the table or view is both a parent and a dependent.

This information is based on the SELFREFS value for the table or view in SYSCAT.TABLES.

Rows

Total number of data rows in the table or view.

This information is based on the CARD value for the table or view in SYSCAT.TABLES.

Statistics time

The timestamp for the last time a change was made to statistics for the table or view.

This information is based on the STATS_TIME value for the table or view in SYSCAT.TABLES.

Status

Status type for the object from which information is retrieved:

Value	Description
Normal	Normal status for the table, view, alias, or nickname.
Check	CHECK PENDING status on the table or nickname. Constraint checking is turned off, and SELECT, INSERT, UPDATE, and DELETE operations are not allowed on the table.
X	Inoperative view or nickname. This status can occur when the underlying tables of the view or nickname have been dropped.

Tablespace

The name of the primary tablespace for the table or view.

This information is based on the TBSPACE value for the table or view in SYSCAT.TABLES.

Type

The type of table or view from which information is retrieved:

Value	Description
Table	Normal table
Hierarchy	The hierarchy table associated with the implementation of a typed table hierarchy
Summary	Summary table
View	Normal view
Alias	An indirect reference to a table
Nickname	A reference to a table (data source) in a federated database
Typed table	Table created from a structured type
Type view	A view created from a structured type

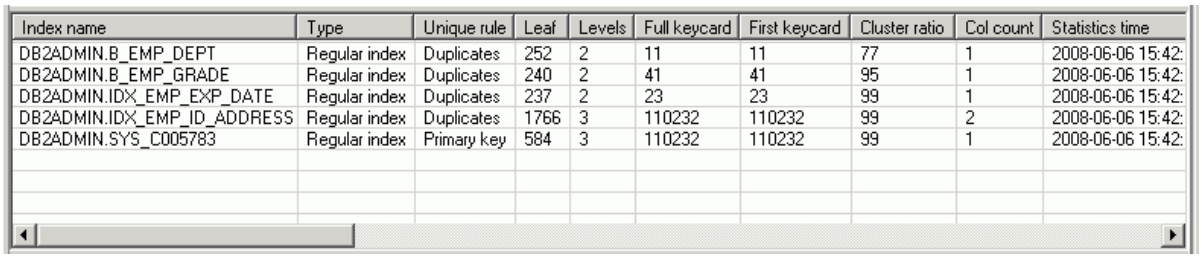
Volatile

The indicator showing whether the table or view has been declared volatile. A volatile table is a table whose contents vary greatly--from empty to large--at any point in time. To generate the access plan for a volatile table, DB2's SQL optimizer tends not to rely on existing table statistics and, therefore, often uses an index scan instead of a table scan. Values for this detail include:

Value	Description
Cardinality of table is volatile	Table is volatile
blank	Table is not volatile

Index Statistics Pane

View the Index Statistics pane



Index name	Type	Unique rule	Leaf	Levels	Full keycard	First keycard	Cluster ratio	Col count	Statistics time
DB2ADMIN.B_EMP_DEPT	Regular index	Duplicates	252	2	11	11	77	1	2008-06-06 15:42:
DB2ADMIN.B_EMP_GRADE	Regular index	Duplicates	240	2	41	41	95	1	2008-06-06 15:42:
DB2ADMIN.IDX_EMP_EXP_DATE	Regular index	Duplicates	237	2	23	23	99	1	2008-06-06 15:42:
DB2ADMIN.IDX_EMP_ID_ADDRESS	Regular index	Duplicates	1766	3	110232	110232	99	2	2008-06-06 15:42:
DB2ADMIN.SYS_C005783	Regular index	Primary key	584	3	110232	110232	99	1	2008-06-06 15:42:

The Index Statistics pane is displayed on the Access Plan tab in the SQL Optimizer window. It lists information about each index that exists for the highlighted object in the Object Statistics pane. Additionally, if the operation in the selected access plan step uses an index, this index is automatically highlighted in the Index Statistics pane. You can do the following in this pane:

- Select any index to view statistics about the columns for the index. (The appropriate column rows are highlighted in the Column Statistics pane.)
- Right-click a specific index to update its catalog statistics.

Note: If the object highlighted in the Object Statistics pane is a view, then no indexes display in the Index Statistics pane.

For each index listed in the Index Statistics pane, the following information is obtained from the SYSCAT.INDEXES catalog view:

Col count

The number of columns that make up the index key for the index, plus any INCLUDE columns, which are additional non-key columns maintained in a unique index.

This information is based on the COLCOUNT value for the index in SYSCAT.INDEXES.

Cluster factor

The ratio of the number of table rows that are ordered on data pages in index key sequence to the total number of table rows. This value is expressed as a decimal value between 0 and 1, with a higher precision than the Cluster Ratio. The optimizer uses this statistic to estimate the I/O cost of reading data pages into the bufferpool. The value –1 is used to indicate that no statistics are available.

Note: Either this statistic or the Cluster Ratio—not both—is recorded in the SYSCAT.INDEXES catalog. Generally, only one of the indexes for a table can have a high degree of clustering.

This information is based on the CLUSTERFACTOR value for the index in SYSCAT.INDEXES.

Cluster ratio

The ratio of the number of table rows that are ordered on data pages in index key sequence to the total number of table rows. This value is expressed as a percentage (a whole number between 0 and 100). The optimizer uses this statistic to estimate the I/O cost of reading data pages into the bufferpool. The value –1 is used to indicate that no statistics are available.

Note: Either this statistic or the Cluster Factor—not both—is recorded in the SYSCAT.INDEXES catalog. Generally, only one of the indexes for a table can have a high degree of clustering.

This information is based on the CLUSTERRATIO value for the index in SYSCAT.INDEXES.

Definer

The user who created the index.

This information is based on the DEFINER value for the index in SYSCAT.INDEXES.

Density

The ratio of the Seq Pages value to the total number of pages occupied by the index. This ratio is expressed as a percentage.

This information is based on the DENSITY value for the index in SYSCAT.INDEXES.

Entry type

The type of table on which the index is based:

Value	Description
Untyped Table	A regular table or view (or the alias or nickname referring to a regular table). In a regular table, the data type for each column is defined separately in the CREATE TABLE statement.
Typed Table	A table based on a user-defined structured type. In a typed table, the column definitions are derived from attributes in the structured type on which the table was created. Note that a typed table can be a supertable or a subtable within a table hierarchy. (A subtable inherits its column definitions from the supertable.)
Hierarchy Table	A hierarchy table, which contains one column for each unique column in the table hierarchy. DB2's SQL optimizer uses this table to generate access plans for queries written against the individual tables in the hierarchy.

First keycard

The number of distinct key values based on the first column in the index key.

This information is based on the FIRSTKEYCARD value for the index in SYSCAT.INDEXES.

First2 keycard

The number of distinct keys using the first two columns of the index. (The value -1 is used if statistics have never been updated or if this statistic does not apply to the index key.)

This information is based on the FIRST2KEYCARD value for the index in SYSCAT.INDEXES.

First3 keycard

The number of distinct keys using the first three columns of the index. (The value -1 is used if statistics have never been updated or if this statistic does not apply to the index key.)

This information is based on the FIRST3KEYCARD value for the index in SYSCAT.INDEXES.

First4 keycard

The number of distinct keys using the first four columns of the index. (The value -1 is used if statistics have never been updated or if this statistic does not apply to the index key.)

This information is based on the FIRST4KEYCARD value for the index in SYSCAT.INDEXES.

Full keycard

The number of distinct values for the index key. If the index key is unique, the Full Keycard value is equal to the number of rows in the table, or the table's cardinality. If the index key is not unique, the Full Keycard value is equal to the number of distinct values available for the key. For example, if the index key is made up of one column called Status, which has three possible values--Paid, Billed, or Late, the Full Keycard value is 3.

This information is based on the FULLKEYCARD value for the index in SYSCAT.INDEXES.

Index name

The qualified name of the index.

This information is based on the INDSCHEMA and INDNAME values for the index in SYSCAT.INDEXES.

Internal ID

The internal ID of the index.

This information is based on the IID value for the index in SYSCAT.INDEXES.

Leaf

The number of leaf pages in the index.

This information is based on the NLEAF value for the index in SYSCAT.INDEXES.

Levels

The number of non-leaf and leaf levels in the B+ structure of the index.

This information is based on the NLEVELS value for the index in SYSCAT.INDEXES.

Made unique

Indicator as whether the index was converted from a non-unique to a unique index to support a unique or primary key constraint. (If you drop the constraint, the index reverts to non-unique.)

Value	Description
Yes	The index was converted to unique.
No	The index remains as it was created.

Min pct used

The minimum percentage of used space allowed before the online index reorganization feature merges index pages. (This number is expressed as an integer between 0 and 99.) This statistic is not available for a nickname.

This information is based on the MINPCTUSED value for the index in SYSCAT.INDEXES.

Page fetch pairs

A list of integer pairs that provide page fetch estimates for different buffer sizes. The first integer in a given pair represents the number of pages in a hypothetical buffer. The second integer in the pair represents the number of pages fetches required to scan the table with the index using the hypothetical buffer. DB2's SQL optimizer uses this information to estimate the cost of reading data pages into the bufferpool when the Cluster Ratio statistic is not available.

This information is based on the PAGE_FETCH_PAIRS value for the index in SYSCAT.INDEXES.

Pct free

The percentage of space on each leaf page that is available for future inserts. (This value is determined when the index is created.)

This information is based on the PCTFREE value for the index in SYSCAT.INDEXES.

Reverse scan

Indicator as to whether the index supports reverse scans, enabling DB2's Index Manager to scan the index in a forward and backward direction to retrieve data:

Value	Description
Yes	The index supports reverse scan.
No	The index does not support reverse scans.

Seq page

The number of leaf pages located physically located in index key order with few or not large gaps between them.

This information is based on the SEQUENTIAL_PAGES value for the index in SYSCAT.INDEXES.

Statistics time

The timestamp for the last time a change was made to statistics for the index.

This information is based on the STATS_TIME value for the index in SYSCAT.INDEXES.

System req

Indicator as to whether the index is required by the system:

Value	Description
1	The index is required for a primary or unique key constraint or is required as the index on the OID column in a typed table.
2	The index is required for a primary or unique key constraint and is required as the index on the OID column in a typed table.
3	The index is not required.

Type

The type of index. This value is based on the INDEXTYPE value for the index in SYSCAT.INDEXES.

Value	Description
Block	A composite block index, which contains the key columns for all dimensions in a multi-dimensional clustering (MDC) table
Clustering	The table's clustering index
Dimension block	An index containing pointers to each occupied extent for a single dimension in a multi-dimensional clustering (MDC) table
Regular*	A non-clustering index on the table
Virtual (R)	A recommended virtual index on the table
Virtual (U)	A user-defined virtual index on the table

Unique col count

The number of columns that make up the unique key in the index (if the index has a unique key).

This information is based on the UNIQUE_COLCOUNT value for the index in SYSCAT.INDEXES.

Unique rule

The type of unique constraint that the Database Manager enforces using the index key during the execution of any operation--such as an INSERT or UPDATE--that changes data values in the table.

Value	Description
Duplicates	The index key has no unique constraint. Duplicate key values are allowed in the table.
Primary Index	The index key enforces a primary key constraint. No two primary key values in the table can be equal. The columns that make up the primary key cannot contain NULL values.
Unique	The index key enforces a unique constraint on the table. No two unique key values in the table can be equal. The columns that make up the unique key cannot contain NULL values.

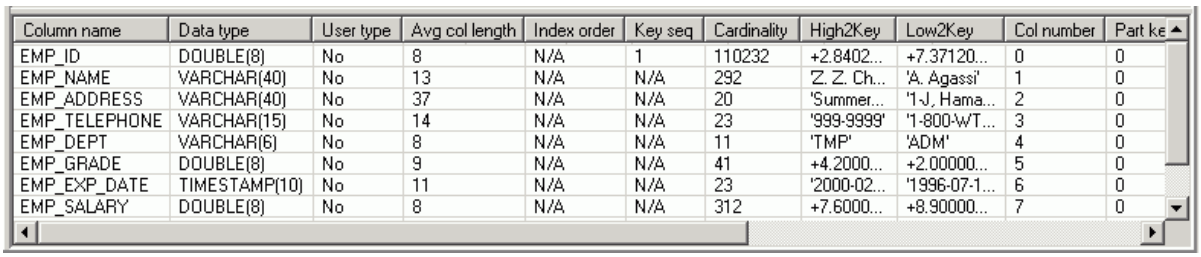
User defined

The indicator as to whether the index is user defined (using the CREATE INDEX statement) or system defined.

Value	Description
Yes	The index is user defined and has not been dropped.
No	The index is system defined or automatically created at the time when the database was installed.

Column Statistics Pane

View the Column Statistics pane



Column name	Data type	User type	Avg col length	Index order	Key seq	Cardinality	High2Key	Low2Key	Col number	Part ke
EMP_ID	DOUBLE(8)	No	8	N/A	1	110232	+2.8402...	+7.37120...	0	0
EMP_NAME	VARCHAR(40)	No	13	N/A	N/A	292	'Z. Z. Ch...	'A. Agassi'	1	0
EMP_ADDRESS	VARCHAR(40)	No	37	N/A	N/A	20	'Summer...	'1-J, Hama...	2	0
EMP_TELEPHONE	VARCHAR(15)	No	14	N/A	N/A	23	'999-9999'	'1-800-WT...	3	0
EMP_DEPT	VARCHAR(6)	No	8	N/A	N/A	11	'TMP'	'ADM'	4	0
EMP_GRADE	DOUBLE(8)	No	9	N/A	N/A	41	+4.2000...	+2.00000...	5	0
EMP_EXP_DATE	TIMESTAMP(10)	No	11	N/A	N/A	23	'2000-02...	'1996-07-1...	6	0
EMP_SALARY	DOUBLE(8)	No	8	N/A	N/A	312	+7.6000...	+8.90000...	7	0

The Column Statistics pane is displayed on the Access Plan tab in the SQL Optimizer window. It lists information about each column in the object highlighted in the Object Statistics or Index Statistics pane. This information is

retrieved from the SYSCAT.COLUMNS catalog view excepted for the Index order statistic, which is obtained from SYSCAT.INDEXCOLUSE catalog view.

Avg col length

The average length of the values in the column.

This information is based on the AVGCOLLEN value for the column in SYSCAT.COLUMNS.

Cardinality

The number of rows in a database table.

This information is based on the COLCARD value for the column in SYSCAT.COLUMNS.

Col number

The numerical position of the column in the table or view, beginning at zero.

This information is based on the COLNO value for the column in SYSCAT.COLUMNS.

Column name

The name of the column.

This information is based on the COLNAME value for the column in SYSCAT.COLUMNS.

Data type

The data type of the column along with the length and scale (if applicable).

This information is based on the TYPENAME, LENGTH and possibly the SCALE value for the column in SYSCAT.COLUMNS.

Default

The column's default value appropriate for the column's data type. This value can be the keyword NULL.

This information is based on the DEFAULT value for the column in SYSCAT.COLUMNS.

High2Key

The second highest value in this column.

This information is based on the HIGH2KEY value for the column in SYSCAT.COLUMNS.

Index order

The order of the values in this column within the index. Statistic values include:

Value	Description
Ascending	Ascending order
Descending	Descending order
Include	INCLUDE column (ordering ignored)

Key seq

The column's numerical position within the primary key for the table or view. The value N/A appears if the column is not part of the primary key.

This information is based on the KEYSEQ value for the column in SYSCAT.COLUMNS

Low2Key

The second lowest value in this column.

This information is based on the LOW2KEY value for the column in SYSCAT.COLUMNS.

Null count

The number of NULL values in the column.

This information is based on the NUMNULLS value for the column in SYSCAT.COLUMNS.

Nulls

Indicator as to whether the column can contain a NULL value.

Value	Description

Yes	The column is nullable.
No	The column is not nullable.

Part key seq

The column's numerical position within the table's partitioning key for the table or view. The value 0 or NULL is displayed if the column is not part of the partitioning key. Additionally, if the table is a hierarchy table or a subtable in a hierarchy, the value NULL is displayed.

This information is based on the PARTKEYSEQ value for the column in SYSCAT.COLUMNS.

User type

The qualified name of the data type if the type is user defined (or distinct). A user-defined type has a qualified name other than SYSIBM. If the data type is not distinct (that is, its qualified name is SYSIBM), the value No appears.

This information is based on the TYPESCHEMA value for the column in SYSCAT.COLUMNS.

Related Topic

Column Values Distribution Statistics Pane

Column Values Distribution Statistics Pane

View the Column Values Distribution Statistics pane

Sequence	ValCount	DistCount	Type	ColValue
1	882		Frequency	+3.6000000000000...
2	854		Frequency	+2.6500000000000...
3	716		Frequency	+2.8500000000000...
4	716		Frequency	+3.6500000000000...
5	716		Frequency	+3.3500000000000...
6	716		Frequency	+3.2500000000000...
7	716		Frequency	+3.2000000000000...
8	716		Frequency	+3.3000000000000...
9	716		Frequency	+3.4000000000000...
10	689		Frequency	+2.9000000000000...
1	28		Quantile	+6.6000000000000...
2	3058		Quantile	+1.5000000000000...
3	5841		Quantile	+1.8500000000000...
4	8707		Quantile	+2.0700000000000...
5	11793		Quantile	+2.2500000000000...
6	14603		Quantile	+2.4000000000000...

The Column Values Distribution Statistics pane is displayed on the Access Plan tab in the SQL Optimizer window. It lists statistics used by DB2's SQL optimizer to handle non-uniform distribution of column values. For the column highlighted in the Column Statistics pane, this pane describes the position of column values in the order of most frequently used values or in quantile order. SQL Optimizer obtains this information from the SYSCAT.COLDIST catalog view. The DB2's SQL optimizer uses this information to estimate more accurately the number of rows in a column that satisfy the specified equality or range predicates.

Sequence

Sequence

The position of the column value (listed for **ColValue**) within the order of most frequently used values or within the quantile order.

- If **Type** is *Frequency*, the sequence number *n* identifies the column value as the *n*the most frequent value for the column.
- If **Type** is *Quantile*, the sequence number *n* identifies the column value as the *n*th quantile value for the column.

ValCount

ValCount

The count for the column value listed for **ColValue**. The meaning of this count depends on the type of statistics collected for the column value: frequency or quantile.

- If **Type** is *Frequency*, the value count is the number of rows that contain this value in the column.
- If **Type** is *Quantile*, the value count is the number of rows containing values in the column that are less than or equal to **ColValue**.

In a partitioned database, **ValCount** is the estimated value of the count at the database partition multiplied by the number of database partitions. However, when **Type** is *Frequency* and the column is the single-column partitioning key of the table, **ValCount** is simply the count at the database partition.

DistCount

DistCount

The number of distinct values in the column that are less than or equal to the column value listed under **ColValue**.

Note: This count is collected only for columns that are the first key column in an index.

Type

Type

The type of distribution statistics collected for the column value listed under **ColValue**:

Value	Description
Frequency	The statistics show the values for the highest number of duplicate values in the column. The number of rows displayed is dependent upon the num_freqvalues database configuration parameter or the value that was specified when the table statistics were updated.
Quantile	Statistics about value's quantile position in the column are collected. The number of rows displayed is dependent upon the num_quantiles database configuration parameter or the value that was specified when the table statistics were updated..

ColValue

ColValue

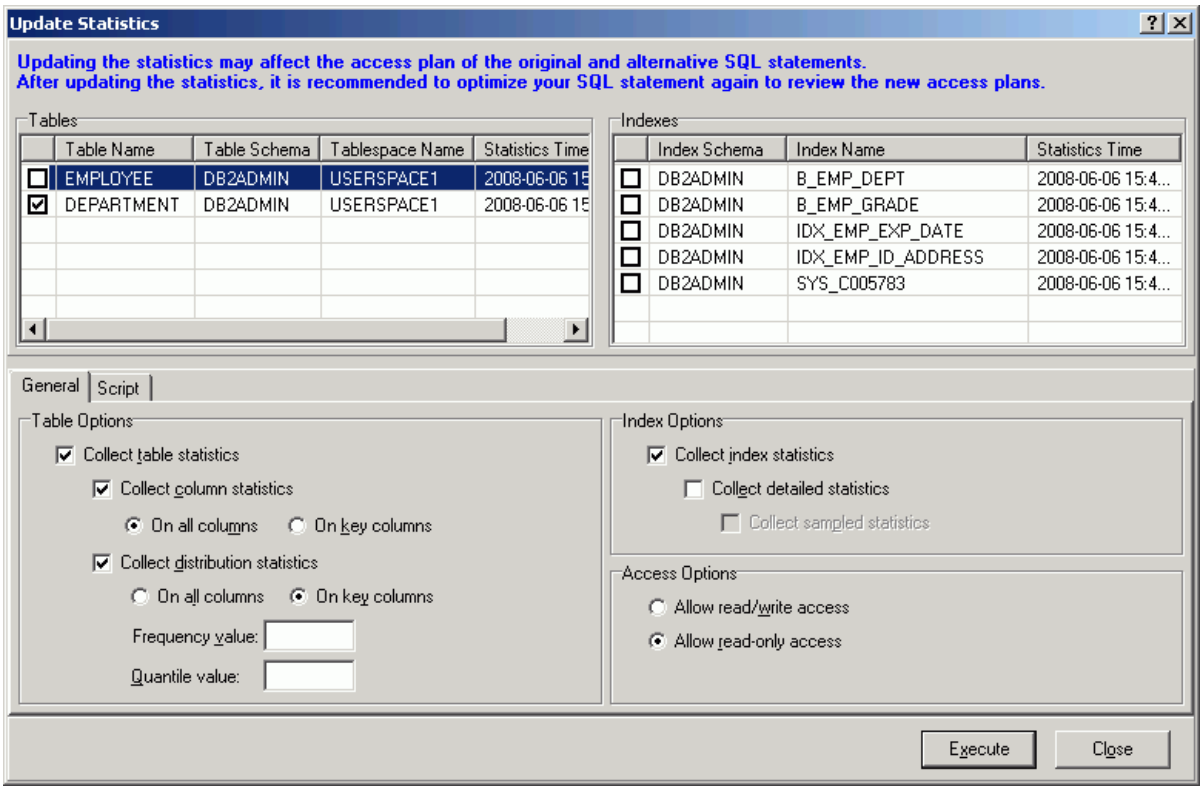
The value of data in the column displayed as a character literal or as a NULL value.

Related Topic

Column Statistics Pane

Update Statistics

View Update Statistics Window



You can update the statistics for the tables and their indexes when you are viewing the Access Plan tab.

To update the statistics for a table

1. Right-click the table in the Table Statistics pane. and select **Update Statistics**.
2. In the Update Statistics window, select one or more tables to update.
3. In the Table Options section, select the specific options for updating the table statistics.

Note: This table only covers unfamiliar information. It does not include all field descriptions.

Frequency value	Specify the maximum number of frequency values to collect for columns values distribution. If you do not specify a value, the num_freqvalues database configuration parameter is used.
Quantile value	Specify the maximum number of quantile values to collect for columns values distribution. If you do not specify a value, the

	num_quantiles database configuration parameter is used.

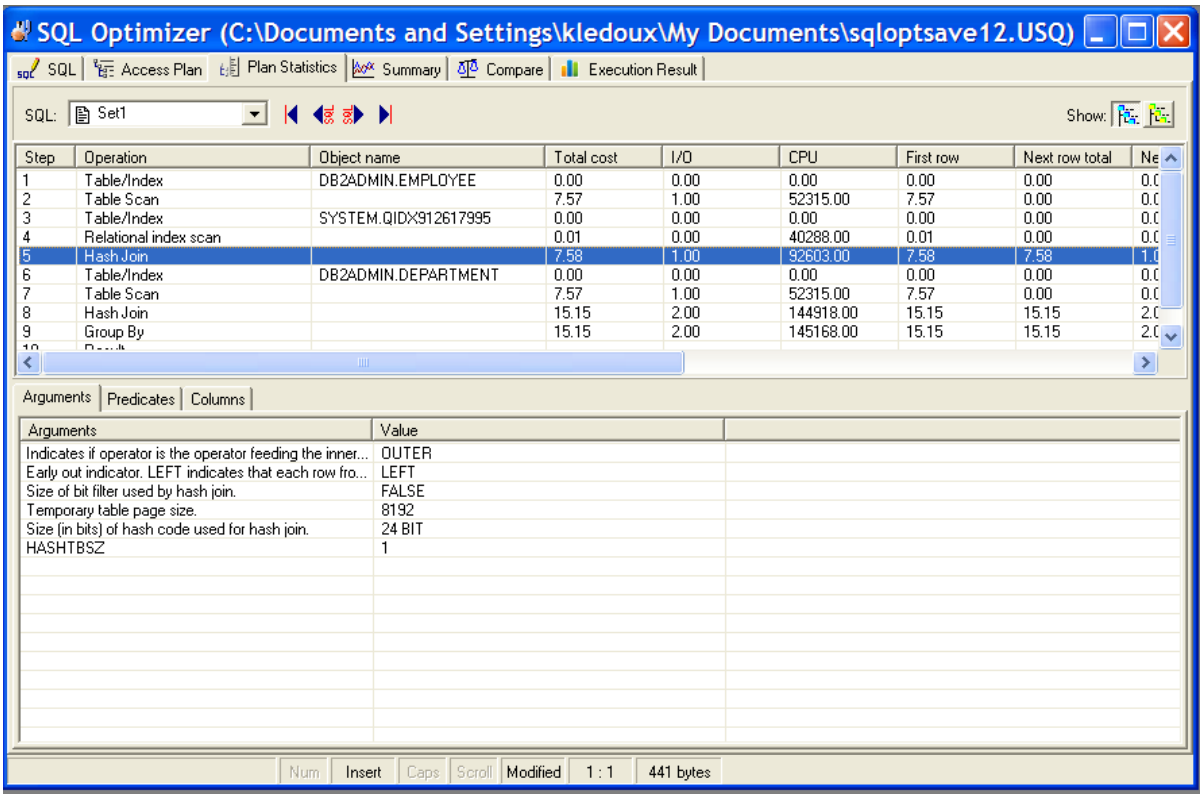
1. Check the indexes to update for each table you have checked.
2. In the Index Options section, select the specific options for updating the index statistics.
3. Click the **Script** tab and review the RUNSTATS command.
4. Click **Execute**.

Note: In order to update statistics, you must have these privileges:

SYSADM
SYSCTRL
SYSMAINT
DBADM
CONTROL privilege on the table
LOAD authority

Plan Statistics Tab

View the Plan Statistics tab



The Plan Statistics tab in the SQL Optimizer window displays the cost for each step in the access plan for the currently selected SQL or index-set alternative, along with the plan's arguments and values, its predicates, and the columns involved in the data stream.

Cost Statistics Pane

The Cost Statistics pane provides detailed descriptions of each operation used in the access plan.

Estimated Cost Statistics

The following statistics are shown for each row of the access plan:

Statistic	Description
Buffers	The estimated cumulative number of logical read requests for data pages that have gone through the buffer pool. This value is expressed in 4KB page units.
Communication	The estimated cumulative communication cost of executing the access plan up to and including the operation specified in the selected step. This value is expressed in TCP/IP frame units.
CPU	The estimated cumulative number of CPU instructions required to execute the access plan up to and including the operation specified in the selected step.
First row	The estimated cumulative cost of fetching the first row of data for the access plan up to and including the operation specified in the selected step. This statistic is expressed in timeron units.
First row comm.	The estimated cumulative communication cost of fetching the first row of data for the plan up to and including the operation specified in the selected step. This statistic is expressed in TCP/IP frame units.
I/O	The estimated total number of I/Os required for the access plan, up to and including the operation specified in the selected step. This value is expressed in units of 4KB pages.
Next row CPU	The estimated cumulative number of CPU instructions required to fetch the second row of data for the plan up to and including the operation specified in the selected step.
Next row I/O	The estimated cumulative number of I/Os required to fetch the second row of data for the plan up to and including the operation specified in the selected step. This statistic is expressed in timeron units.
Next row total	The estimated cumulative cost of fetching the second row of data up to and including the operation specified in the selected step. This statistic is expressed in timeron units.
Object name	The name of the table, view, or alias involved in the operation.
Operation	A brief description of the operation performed on the data, as specified in the selected access plan step.
Remote comm.	The estimated cumulative communication cost of executing the remote access plan up to and including this operator. This statistic is expressed in TCP/IP frame units.
Remote total	The estimated cumulative cost of performing operations on a remote database for this access plan up to and including the operation specified in the selected step. This statistic is expressed in timeron units.
Step	The number of the step within the sequence of steps performed in the access plan.
Total cost	The estimated cumulative cost of executing the access plan up to and including the operation specified in the selected step. This statistic is expressed in timeron units.

To see the Argument, Predicate, and Column information for a specific step

1. Select a specific plan step in the Cost Statistics pane.
2. Click the **Argument**, **Predicate**, or **Column** tab.

Argument

The Arguments tab lists the arguments and their values that are used in the access plan operation highlighted in the Statistic pane.

Predicate

The Predicates tab provides information about the SQL predicates that the operation highlighted in the Statistic pane is processing.

The following information shows for each predicate:

Text

The text of the predicate as interpreted by the SQL compiler.

Filter factor

The estimated fraction of rows that the predicate qualifies.

How applied

The label specifying how the predicate is used in the operation:

Value	Description
JOIN	Used to join tables
RESID	Evaluated as a residual predicate
SARG	Evaluated as a sargable predicate for index or data pages
START	Used as a start condition
STOP	Used as a stop condition

When applied

The label specifying when the subquery used in this predicate is evaluated:

Value	Description
blank	The predicate does not contain a subquery.
EAA	(Evaluated at application) The subquery is re-evaluated each time the operation applies the predicate to a row.
EAO	(Evaluated at open) The subquery is evaluated once for the operation. The subquery results are reused each time the operation applies the predicate to a row.
MUL	(Multiple) More than one type of subquery exists for this predicate.

Relop type

The label specifying the type of relational operator used in the predicate:

Value	Description
blank	Not applicable
EQ	Equals
GE	Is greater than or equal to
GT	Is greater than
IN	Is in list
LE	Is less than or equal to
LK	Is like
LT	Is less than
NE	Is not equal to
NL	Is null
NN	Is not null

Subquery

The flag indicating whether or not the predicate requires a data stream from a subquery:

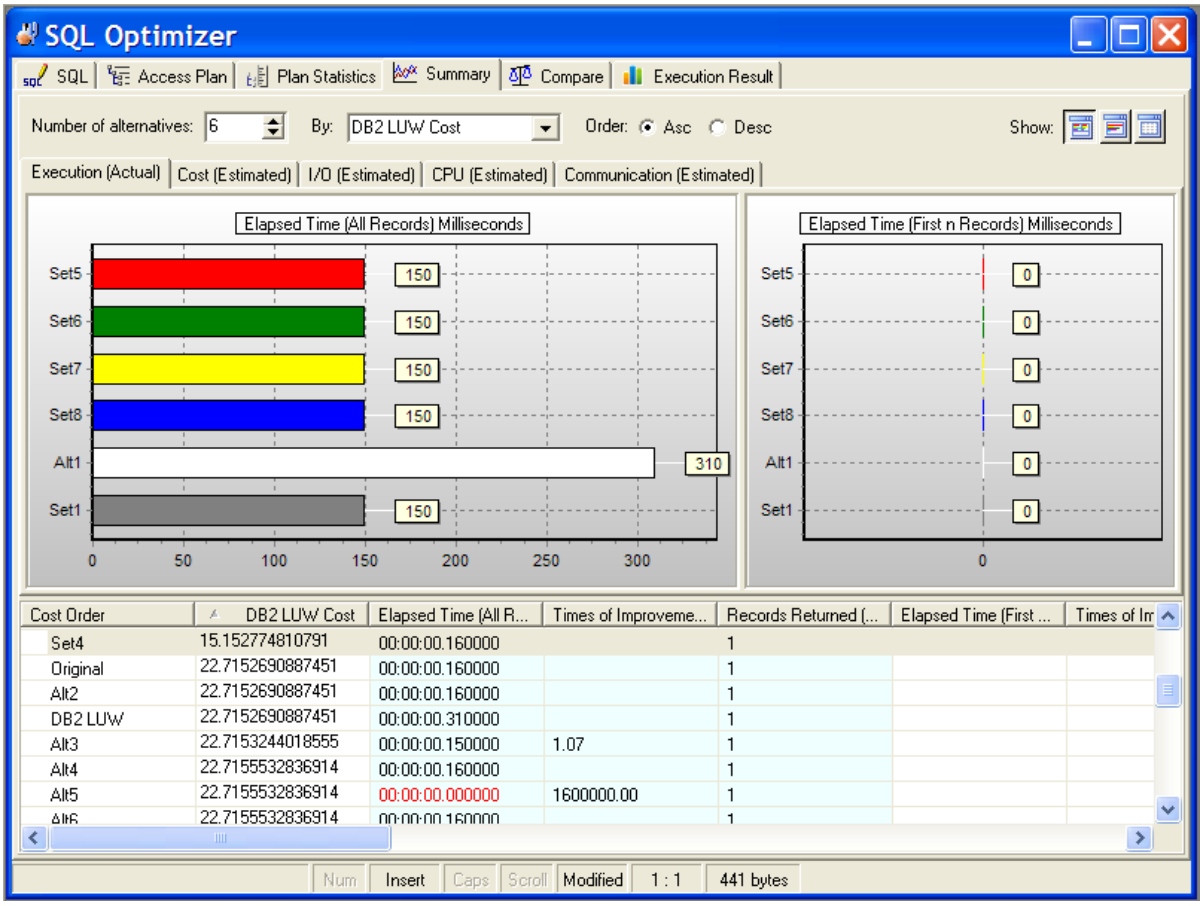
Value	Description
N	No subquery stream is required.
Y	One or more subquery streams are required.

Columns

The Columns tab lists the table columns involved in the data stream for the access plan operation highlighted in the Cost Statistics pane.

Summary Tab

View the Summary tab



The Summary tab in the SQL Optimizer window displays the Run Time information and the cost estimates from various perspectives.

Run Time Pane

The Run Time pane displays the run time statistics for retrieving all records and for retrieving the first record.

Cost Estimates

Cost statistics shown on the Summary tab are estimated cumulative costs across all operations used in the access plan.

Item	Description
Buffers	The estimated cumulative number of logical read requests for data pages that have gone through the buffer pool. This value is expressed in 4KB page units.
Communication	The estimated cumulative communication cost of executing the access plan up to and including the operation specified in the selected step. This value is expressed in TCP / IP frame units.
CPU	The estimated cumulative number of CPU instructions required to execute the access

	plan up to and including the operation specified in the selected step.
First row	The estimated cumulative cost of fetching the first row of data for the access plan up to and including the operation specified in the selected step. This statistic is expressed in timeron units.
First row comm .	The estimated cumulative communication cost of fetching the first row of data for the plan up to and including the operation specified in the selected step. This statistic is expressed in TCP / IP frame units.
I/O	The estimated total number of I/Os required for the access plan, up to and including the operation specified in the selected step. This value is expressed in units of 4KB pages.
Next row CPU	The estimated cumulative number of CPU instructions required to fetch the second row of data for the plan up to and including the operation specified in the selected step.
Next row I/O	The estimated cumulative number of I/Os required to fetch the second row of data for the plan up to and including the operation specified in the selected step. This statistic is expressed in timeron units.
Next row total	The estimated cumulative cost of fetching the second row of data up to and including the operation specified in the selected step. This statistic is expressed in timeron units.

Charts

The statistics from the grid are grouped together and displayed in charts.

To select how many records are displayed in the charts

Select the number from the **Number of Alternatives** list.

To select which statistic to use to select the records

Select the statistic from the **By** list.

To select the top or the bottom range of the statistic

Select **Asc** (Ascending) or **Desc** (Descending) from the **Order** option.

Display or Hide Panes

Use the pane-control buttons to reorganize the panes:

To display the statistics and charts

Click **Chart and Statistics** .

To display only the charts

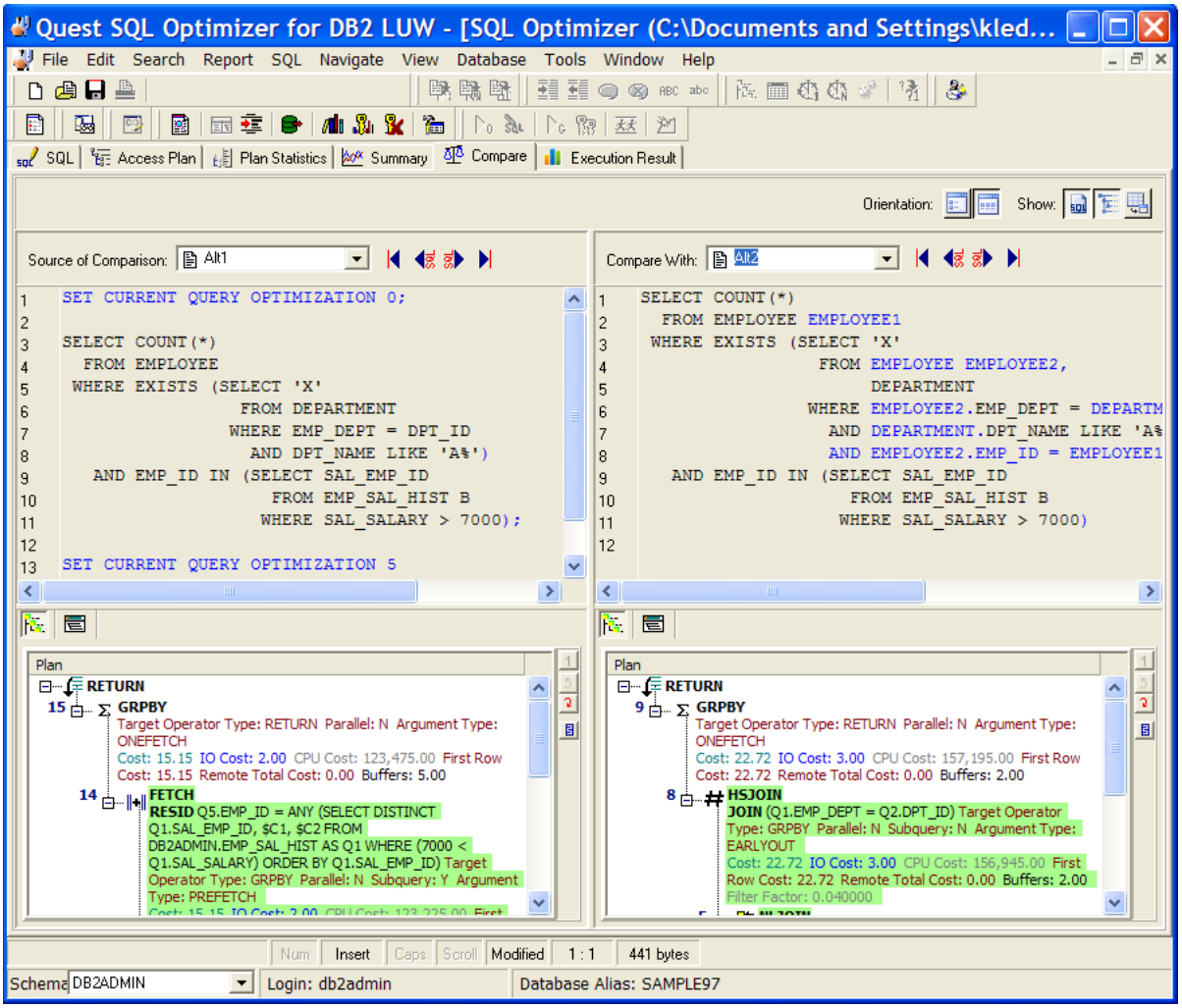
Click **Chart Only** .

To display only the statistics

Click **Statistics Only** .

Compare Tab

View the Compare Tab



The Compare tab in the SQL Optimizer window enables you easily to compare SQL text, access plans, and statistics between two alternatives or between an alternative and the original SQL.

The Compare tab consists of two major panes—Source of Comparison and Compare With. Each pane shows information for one alternative. Differences in syntax and access plans are highlighted in either pane.

Display or Hide Panes

Use the pane-control buttons to reorganize the panes:

To display panes vertically

Click .

To display panes horizontally

Click .

To display or remove SQL Text panes

Click .

To display or remove Access Plan panes

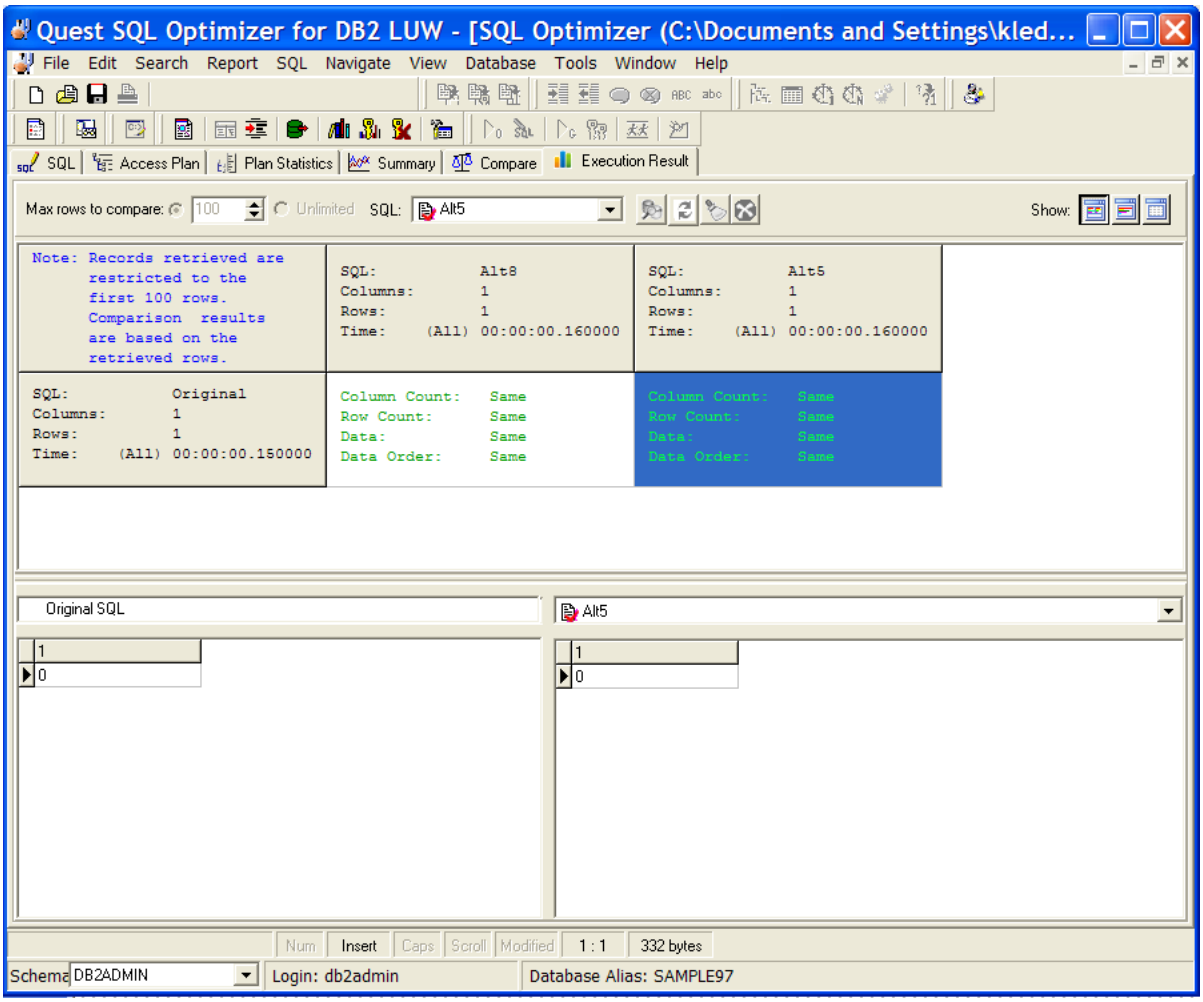
Click .

To display or remove Statistics panes

Click .

Execution Result Tab

View the Execution Result tab



The Execution Result tab in the SQL Optimizer window compares the data returned by the original SQL with the data returned by any one of the SQL alternatives.

To compare all rows of data between the original SQL and an SQL alternative

1. Select **Unlimited** for **Max rows to compare**.
2. From the **SQL** list, select the SQL alternative that you want to compare with the original SQL.
3. Click .

To compare a limited number of rows between the original SQL and an alternative

1. Enter the number of rows in **Max rows to compare**.
2. From the **SQL** list, select the SQL alternative that you want to compare with the original SQL.
3. Click .

Comparison Grid

The grid in the top pane provides quick overview of returned-data differences. The numbers of returned columns and rows show whether the SQL alternative produces the same result as the original SQL statement. The displayed time shows how long it took to execute Batch Run or the Run Time, not the time it took to retrieve the data for the Execution Result tab.

Note: If the **Data** statistic shows that the returned data differs when you retrieve limited rows, do not assume that the alternative is returning wrong data. The difference can suggest that the alternative has returned data in an order different from the order returned by the original SQL. Two SQL statement are considered to be semantically equivalent if they retrieve the same data in any order (as long as the ORDER BY clause is not used in the SQL statement).

Comparison Functions

To refresh the data for the currently selected alternative

Click .

To clear all the data for original and all alternatives

Click .

To abort Compare SQL or Refresh SQL process

Click .

Use the SQL Optimizer

Enter or Edit the Original SQL Statement

You can optimize a single SELECT, DELETE, UPDATE, and INSERT SQL statement in the SQL Optimizer window. This optimization process includes running two major functions—SQL Rewrite and Generate Indexes—on this original statement to obtain alternatives on which you can test performance.

The following procedures describe how to obtain or edit this original SQL statement.

To enter the original SQL statement

1. Click .
2. Enter the SQL statement in the SQL Text pane on the SQL tab:
   - Type or paste the statement from a another location.
   - Open an existing file.
   - Send an SQL statement directly from another window using the Send SQL to the SQL Rewrite Function or Send SQL to the Generate Indexes Function function.

     Note: When you use either of these functions, the SQL Rewrite or Generate Indexes function automatically runs on the SQL once the SQL Optimizer window opens.
   - From Toad, open a SQL Optimizer session on the SQL in the Toad Editor or the SQL used in view, trigger, or MQ table DDL.
3. Perform any of these functions:
   Retrieve Access Plan
   Retrieve Run Time
   Retrieve Run Result
   Automatically Rewrite the Original SQL Statement
   Generate Index-Set Alternatives
   Batch Run

To edit the original SQL statement

1. Select <*Edit SQL*> in the Batch Run pane.
2. Edit the SQL text in the SQL Text pane.

Notes:

- SQL Optimizer processes only a single SELECT, DELETE, UPDATE, or INSERT SQL statement.
- You can have only one SQL Optimizer window open at any one time. SQL Optimizer prompts you to save the original SQL statement if you attempt to open another file or create a new one.
- If the SQL statement uses temp tables, the temp tables must already exist before you can perform any function on the SQL statement. Create the temp tables by using the User-Defined Temp Table window.
- You can insert variables and comments within the original SQL statement.

Tip: To help construct a SQL statement, use the Editor functions: member lookup, argument lookup, auto correction, indent, outdent, comment, and uncomment functions.

Automatically Rewrite the Original SQL Statement

Use the SQL Rewrite function to generate alternatives to an SQL statement that you enter in the SQL Optimizer window or that you sent to SQL Optimizer from another tool, such the Scanned SQL Viewer, Toad Database Explorer, or SQL Formatter.

To rewrite an SQL statement

1. Provide the SQL statement.

   Note: In Toad, if you open a SQL Optimizer session on the SQL used in a view, trigger, or MQ table, the statement automatically displays in the SQL Optimizer window.

2. Click to rewrite the original SQL. The rewrite time is dependent on the complexity of the original SQL statement, the SQL Options applied, and the quota values set in the Options window. When the rewrite completes, each rewritten SQL version displays as a SQL alternative in the Run Time pane.

Warning: When you define the optimization quota values that, note that the higher the quota, the longer it may take to rewrite a complicated SQL statement.

Note: The SQL Rewrite process executes multiple transformation rules to produce a list of semantically equivalent SQL alternatives, each with a unique access plan. SQL options defined in the Options window are also applied to produce the list of SQL alternatives.

To abort the rewrite process

Click .

Allow a few seconds to terminate all processes.

Automatically Start the Batch Run

If you have selected the Automatically start Batch Run after rewriting option, the Batch Run Criteria window automatically opens before the rewrite process begins so you can select the settings for the Batch Run.

Tips:

- To generate virtual index sets that you can add to the list of SQL alternatives generated during SQL Rewrite, click . You can then include these index-set alternatives in the Batch Run process. See Generate Index-Set Alternatives for more information.
- To create your own virtual index sets that you can add to the list of SQL alternatives, click . See Add Your Own Virtual Index Sets for more information.

Rewrite Details

The Rewrite Details dialog is optional and can be displayed after the SQL Rewrite process completes. This dialog displays the number of semantically equivalent SQL statements investigated and the number of valid SQL alternatives produced. The dialog displays the following:

- Time calculations for the rewrite process.
- Number of semantically equivalent SQL statements investigated.
- Number of alternative access plans produced.
- How many rewrites were eliminated because they have identical access plans or DB2 LUW costs.

- Warning message if the **Syntax Transformation Quota**, **Total SQL Options Quota**, or **Table Join Permutation Quota** is reached.

Note: Use the Optimization (2) page on the SQL Rewrite tab in Options to define whether to eliminate alternative SQL statements when they have identical access plans or identical DB2 LUW costs. Eliminating the SQL based on identical access plan is more accurate but it can take longer to determine.

The Rewrite Details dialog is shown every time the rewrite process completes. To disable this window, clear the **Show details on next optimization** option in the dialog.

To view the Rewrite Details dialog at anytime

Select **View | Show Optimization Details** in the SQL Optimizer window.

If no alternatives are available, the **Generate Indexes** button is provided to start the Index Expert process of determining virtual index sets that might improve performance. See Generate Index-Set Alternatives for more information.

Add User-Defined SQL Alternatives

Once you have provided the original SQL statement in the SQL Optimizer window, you can create your own tweaked version of the statement and add it as a SQL alternative. You can do this either before or after you have rewritten the original SQL statement. You can then test run the user-defined SQL alternatives with the alternatives created by the SQL Rewrite and Generate Indexes functions. Or, you can simply test your alternative against the original SQL statement.

To insert your own SQL alternative

1. In the Run Time Pane on the SQL tab in the SQL Optimizer window, select a SQL alternative (or the original SQL statement) to use as a template.
2. Click .
3. In the SQL Text Pane, edit the SQL syntax to create your own version.
4. Click to retrieve the access plan for your SQL alternative. The plan is automatically parsed to determine whether its cost is identical to the cost of another alternative or the original SQL. If identical costs are found, SQL Optimizer displays a message.

Note: User-defined SQL alternatives are not checked to determine whether they are semantically equivalent to the original SQL. However, after you perform a Batch Run on the alternative, you can check the **Records Returned** and **Remarks** columns in Run Time pane to see whether the alternative's record count matches the record count for the original SQL.

To delete a user-defined SQL alternative

1. Select the alternative in the Run Time pane.
2. Select **SQL | Delete User-Defined SQL**.

Add Virtual Index Alternatives

About the Generate Indexes Function

The Generate Indexes function in your SQL Optimizer session generates virtual index sets that can potentially improve the performance of your SQL. This function is part of Index Expert, an internal component that facilitates the management and testing of generated and user-defined virtual indexes in your SQL Optimizer session.

When you run the Generate Indexes function, Index Expert analyzes the tables referenced in the original SQL and generates index candidates. It groups these candidates into virtual index sets, each of which generates a unique plan that accesses the virtual indexes in the set.

When Generate Indexes completes, the generated virtual index sets display as alternatives, along with SQL alternatives generated during SQL Rewrite, on the SQL Optimizer window. From the window, you can view the virtual DB2 access plan and cost for any index-set alternative. The Batch Run process executes the index-set alternatives along with SQL alternatives, allowing you to compare run-time results among *all* alternatives to determine the best-performing version of your SQL. (During Batch Run, Index Expert physically creates the indexes to retrieve run times and then drops them.)

Additionally, you can use the Index Impact Analyzer to perform an impact analysis of a virtual index set on other SQL statements stored in the database system. This feature determines which SQL statements are impacted by the index set and identifies the index-set alternative that yields the highest performance gain with the least impact on the database system.

Index Expert Database Requirements

The Index Expert function requires DB2 LUW 7 or later to retrieve the indexes recommended by DB2. It requires DB2 LUW 8 or later to generate its own index-set recommendations (using an Artificial Intelligence engine), along with the DB2-recommended indexes. Additionally, Index Expert requires that the statistics be run in order to be able to estimate the size of the index.

Generate Index-Set Alternatives

The Generate Indexes function creates virtual indexes that can potentially improve performance of your original SQL statement.

This function is part of Index Expert, an internal component that facilitates the management and testing of generated and user-defined virtual indexes in your SQL Optimizer session. When you run Generate Indexes, Index Expert retrieves DB2-recommended virtual indexes, as well as generates its own virtual indexes. To generate index candidates, Index Expert analyzes the SQL syntax, relationships between tables, and data selectivity processes. It then combines the index candidates into one or more index sets, each set with a unique virtual access plan.

When the Generate Indexes process is complete, the resulting virtual index sets display as alternatives in the SQL Optimizer window. These alternatives are listed along with any SQL alternatives that were generated using the SQL Rewrite function. You can then test run and compare *all* alternatives to determine the best-performing version of your SQL.

To generate index-set alternatives for an SQL statement

1. Click to open a SQL Optimizer session.
2. On the SQL tab, enter the SQL statement for which you want to analyze for index alternatives.
   Note: To move an SQL statement from anther tool—such as Scanned SQL Viewer, SQL Formatter, Database Explorer, or SQL Comparer—into SQL Optimizer, select the statement within the tool, and click .
3. Click to generate new index-set alternatives for the original SQL statement.

To terminate the index generation process

Click .

Allow a few seconds to terminate all processes.

Include Index Recommendations from DB2

Index Expert uses DB2's SET CURRENT EXPLAIN MODE RECOMMENDED INDEXES command to retrieve DB2-recommended indexes for the SQL statement. These indexes are combined into a single index-set alternative labeled as *DB2 LUW* in the Run Time pane. To include DB2 index recommendations in the Generate Indexes process, select **Generate Index Sets with DB2 LUW recommendations** in the Index Expert Options settings.

Note: Consider setting the DB2 LUW MEM_UDF heap size to 1024 or higher if the index generation process does not produce DB2-recommended indexes.

Automatically Start the Batch Run

If you have selected the Automatically start Batch Run after generating index sets option, the Batch Run Criteria window automatically opens before the Generate Indexes process begins, allowing you to customize the Batch Run process.

View Index Generation Details

Once index generation is complete, the Index Generation Details window shows counts for generated, eliminated, and accepted index-set candidates. The final virtual index sets display as alternatives in the Run Time pane. These alternatives have the label *Set*x.

Tips:

- To create your own virtual index-sets, click . See Add Your Own Virtual Index Sets for more information.
- To analyze the impact of virtual index sets on other SQL in the database, click . See Analyze the Impact of New Indexes for more information.

Index Generation Details

The Index Generation Details dialog is optional and can be displayed after index generation. This dialog displays the following:

- Number of virtual indexes investigated
- Number of virtual index-sets initially created
- Number of virtual index sets that produced unique access plans
- Number of virtual index sets eliminated due to identical access plan
- A warning message if the **Index Generation Quota** or **Index Set Generation Quota** is reached.
  Note: Consider increasing the quota values on the Quota page for SQL Rewrite in Options to increase the likelihood of finding index sets that can give you better performance for the original SQL statement.

Unless disabled, the Index Generation Details dialog displays each time an index-generation process completes. To disable this window, clear the **Show details on next index generation** option in the Index Generation Details dialog.

If no index-set alternatives are available, the **Optimize** button is enabled to let you start an automatic rewriting of the original SQL statement for optimization.

To open the Index Generation Details dialog anytime

Select **View | Show Index Generation Details**.

Add Your Own Virtual Index Sets

Use the Simulate Indexes function to create your own virtual index sets.

This function is part of Index Expert, an internal component that facilitates the management and testing of generated and user-defined virtual indexes in your SQL Optimizer session.

Creating Your Own Virtual Indexes or Index Sets

1. Click to open a SQL Optimizer session.
2. Enter the original SQL statement for which you feel performance improvement can be achieved with the addition of new indexes.
   Tip: To move an SQL statement from anther tool—such as Scanned SQL Viewer, SQL Formatter, Database Explorer, or SQL Comparer—into SQL Optimizer, select the statement within the tool, and click . This features automatically run the Generate Indexes function described in the next step.
3. Click to generate virtual index-set alternatives that Index Expert recommends. Use these alternatives as a basis for determining which additional virtual indexes you want to define.
4. Click to open the Simulate Indexes window.
   Note: If you have generated index sets (see step 3), these index sets are already listed in the Simulate Indexes window.
5. If no user-defined index exists, the Add Index window appears automatically. Otherwise, click on the right side of the **Create Index here** box.
6. Enter the index properties.

7. Select one of the three options:
    - **Create a new Index Set for this index**to create the index and the index set to which the index belongs
    - **Add to existing index set** to create the index and add it to an existing user-defined index set
    - **Not assign to any index set** to add the index without adding it to any index set
8. Click **OK** to create the index and add it to the specified index set (in the top right pane).
9. Continue to create indexes and add them to index sets as needed.

Deleting User-Defined Indexes

1. Select the index you want to delete from the list in the top left pane (**Create Index here**).
2. Click .

    Note: You can delete only those indexes not belonging to an index set.

Adding Index Sets

1. Click in the top right pane (**Create Index Set here**).
2. Enter a name for the index set.
3. Select the indexes you wish to add to the set.
4. Click **OK**.

Deleting user-defined index set

1. Select the index set from the list in the top right pane (**Create Index Set here**).
2. Click .

Adding and Removing Index from Index Set

In the bottom pane of the Simulate Indexes window, add or remove indexes from the index set by double-clicking the appropriate cell.

Adding Index Sets as Alternatives

To add the finalized virtual index sets to the list of alternatives in SQL Optimizer, click **OK**.

The index sets display as alternatives in the Run Time pane on the SQL Optimizer window. From here, you can compare the index sets with other index sets and SQL alternatives to determine the best-performing alternative.

Analyze the Impact of New Indexes

Analyzing the impact of new indexes on other SQL statements before physically creating them on your database is essential. As new indexes may improve the performance of one SQL but downgrade the performance of others.

To analyze the impact of new indexes

1. From the SQL Optimizer window, generate or create the virtual index sets for your original SQL statement:
   - Click to use the Index Expert function to generate index-set alternatives. See Generate Index-Set Alternatives for more information.
   - Click to create your own virtual index-set alternatives. See Add Your Own Virtual Index Sets for more information.
2. Before performing the **Impact Analysis** function, check that you have a good sample of SQL statements stored in the SQL Repository. Analysis is based on the SQL statements stored the repository. Therefore, the SQL repository should contain a good sample of SQL statements that maybe affected by the new indexes if they were to be created.
3. In Run Time pane, select the index- set alternative that offers the best performance for your SQL statement.
4. Click . The Index Impact Analyzer window will be opened followed by the New Analysis wizard.
5. Follow the New Analysis wizard using the following pages to select the options for creating the analysis.

   Analyzer
   Selected SQL
   Index

Generate a Report for Optimized SQL

The details of the SQL and index-set alternatives and their run-time results 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. Click **OK** to generate the report.
4. The information in the report can be **Saved** and **Printed**.

Note: A few minutes may be needed to generate a long report.

Verify the Correctness of Rewritten SQL

Verify the correctness of the rewritten SQL statement by comparing it with the original SQL statement. Use the **Run Result** function and the **Records Returned** column of the Run Time pane. This information enables you to see whether the optimized SQL statement provides the same results as the original SQL statement.

Note: The Run Result function is not available for index-set alternatives listed in the Run Time pane.

Run Result

The **Run Result** function retrieves the queried records from the connected database.

Records Returned

The **Records Returned** columns display the number of records influenced by the SQL statement. The figure should remain constant for both the original and optimized SQL statements.

Test for Scalability

You can test a single SQL or index-set alternative 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. Click to optimize the original SQL statement. The rewritten SQL versions display as alternatives in the Run Time pane..
2. Click to generate virtual index sets and add them as alternatives.
3. Click **Test for Scalability** .
4. Select the alternative you want to test.

Create Benchmark Factory Import File

All the SQL statements can be saved in a text file from the Job Manager, the Scanned SQL Viewer, the SQL Repository, and the SQL Optimizer windows. These SQL statements can then be imported into Benchmark Factory program (version 4.6 or later).

To create the text 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. Click **OK**.
3. Enter the filename and select the file location. Click **Save**.

Batch Run

Retrieve the Run Time for a Group of Alternatives

The Batch Run function is used to retrieve run times for a group of index-set and SQL alternatives. You can then evaluate run-time results to determine the best performing alternative.

To setup and start the Batch Run

1. Click .

   The Batch Run Criteria window is divided into the following tabs:

   Selected SQL/Index Set

   SQL Termination

   Batch Termination

   Run Time Mode/ Repeat Test

   Batch Run Schedule

2. Select the batch run criteria.

3. The settings can be saved for next optimization by selecting the **Save settings for next batch run** checkbox. By default, this checkbox is unselected.

The Batch Run window displays the run-time criteria and posts the run time for each alternative as it is retrieved. **Stop Current** and **Stop Batch Run** functions are available to terminate the currently running alternative and to stop the Batch Run process. Each selected SQL statement is executed one by one, retrieving the run time unless it is terminated.

Terminate Batch Run

To terminate the complete batch run process

Click .

To terminate the current SQL statement

Select **SQL | Stop Current**.

New Database Session

When you log on to SQL Optimizer, it extracts the logical structure of the database, and maintains this connection throughout the application execution. Another database session is required for when executing the SQL statements for retrieve the run time. This connection is established when the **Run for First Record**, **Run for All Records**, or **Batch Run** functions are initiated and then this new session is dropped after the execution is complete.

Input Parameter Values

SQL Optimizer identifies variables defined within the SQL statement for each SQL or index-set alternative and highlights them in red in the SQL Optimizer window. When the test run process begins, SQL Optimizer prompts you to enter a value for the variable and to select the data type from the Parameters window.

Time

The run time calculation from the Run for All Records, Run for First Record and the Batch Run is used to compare the alternative SQL and index-set alternatives in the SQL Optimizer window. It is used to help determine the best

performing SQL statement versions and the best indexes for your applications. All run times are calculated to the nearest milliseconds in the format of HH:MM:SS.MS.

Excludes network traffic time for data

The calculation of the run time is based on the CPU time of the database server. The data is not moved to the client computer.

Fluctuation of elapsed time

If you execute an alternative more than once, there will be a slight fluctuation in the run time as database optimizer will store the SQL statement in memory. The optimizer does not need to re-parse the SQL statement because the parse tree and the access plan are already in the available cache. Other factors, such as, memory cache status, data pages and index pages distribution status, also affect the performance of SQL statements. In addition, other jobs may be competing for CPU time while the SQL statement is executing. These factors all contribute to a fluctuation of the run time from one execute to the next.

Run Time Calculation for All Records

The execution process retrieves the run time for all records using the following steps.

1. Set the **Row array size for getting elapsed time** according to the Options setting on the General tab. This will have the server allocate enough buffer to accommodate the specified number of records for each Fetch call.
2. Get the current time as the start time.
3. Run the SQL.
4. Fetch the data until no more record are returned. Each fetch will retrieve the specific number of records from the **Row array size for getting elapsed time** setting to the buffer in the database server. Note that records are not actually transferred back to the client but are put into a buffer in database.
5. Get the current time as the end time.
6. Calculate the run time (end time - start time = run time).

For example: if you have a SQL that returns 250 records and the Options setting for array size is 100, then the calls that contribute to the run time would be:

1. Get the current time as the start time.
2. Execute the SQL.
3. Fetch (first 100 records)
4. Fetch (second 100 records)
5. Fetch (last 50 records)
6. Get the current time as the end time.
7. Calculate the run time (end time - start time = run time).

Note: Each of the calls to the network does involve the network time, since the call travels over the network from the client computer to database server. However, the data does not travel over the network, since it is retrieved to a database buffer.

Now with the same SQL but the array size changed to 500, the number of calls would be:

1. Get the current time as the start time.
2. Execute the SQL.
3. Fetch (all 250 records)
4. Get the current time as the end time.
5. Calculate the run time (end time - start time = run time).

You can see from this example how the **Row array size for getting elapsed time** setting in the Options affects the number of fetch calls and in turn affects the amount of network traffic because each fetch travels over the network.

Run Time Calculation for First Record

The run time for retrieving the first record is calculated using the following steps.

1. Set the row array to 1.
2. Get the current time as the start time.
3. Run the SQL.
4. Fetch the first record to the database buffer.
5. Get the current time as the end time.
6. Calculate the run time (end time - start time = run time).

Commit or Rollback

In order to obtain the correct result, SQL Optimizer executes the SQL statement against the database. After executing an UPDATE, INSERT, and DELETE statement, SQL Optimizer prompts you to commit changes that the execution made to the database:

Select **Yes** to commit the changes or **No** to roll back the changes.

How to select the Optimal SQL Statement?

SQL Optimizer provides two different measurements of performance; run time for all records and the run time the first record. Both measurements will provide an indication on the fastest running SQL statement–but with two different aims. It is best to understand the intention of the SQL statement and in what type of source code it will be embedded. Generally, if the SQL statement is used for reports, then select the SQL statement with the run time for all records. If the SQL statement is used for on-line query, then select the SQL statement with the best run time for the first record.

We recommend the use of the actual run time of the SQL or index-set alternative rather than its DB2 LUW cost to determine which alternative is the most suitable. The DB2 LUW cost is just an estimate of the alternative's performance in relation to the other alternatives. Actual performance may be much different.

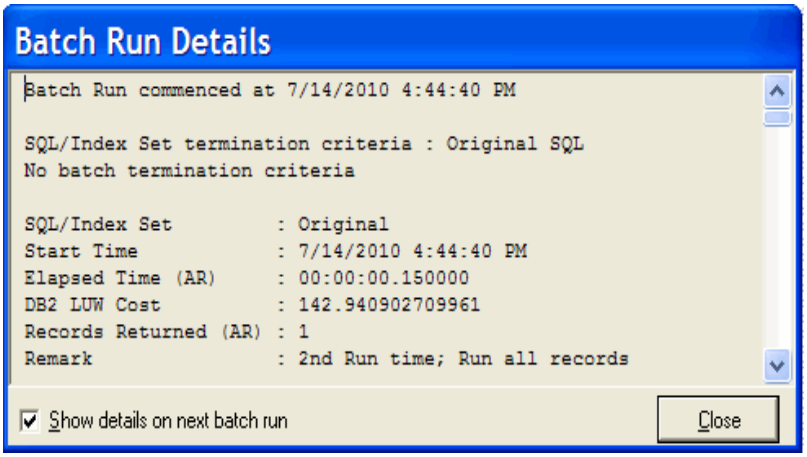
After selecting the alternative with the desired results, copy and paste it into the source program (or use the displayed index DDL to create indexes as needed). We recommend that the alternative be verified and tested before pasting it into the source program.

Caution: Do not select the SQL or index-set alternative with the lowest cost value without retrieving the run time because experience has shown that alternative with the lowest cost is not necessary the SQL with the best performance.

Note: If the aim of the SQL statement is unknown, then we recommend the use of run time for all records as a performance indication.

View Batch Run Details

View the Batch Run Details window



The Batch Run Details window displays a summary of the run time information for all SQL and index-set alternatives executed. The Batch Run Details window will appear after the batch run process is completed unless the **Show details on next batch run** checkbox in the Batch Run Details window is unchecked.

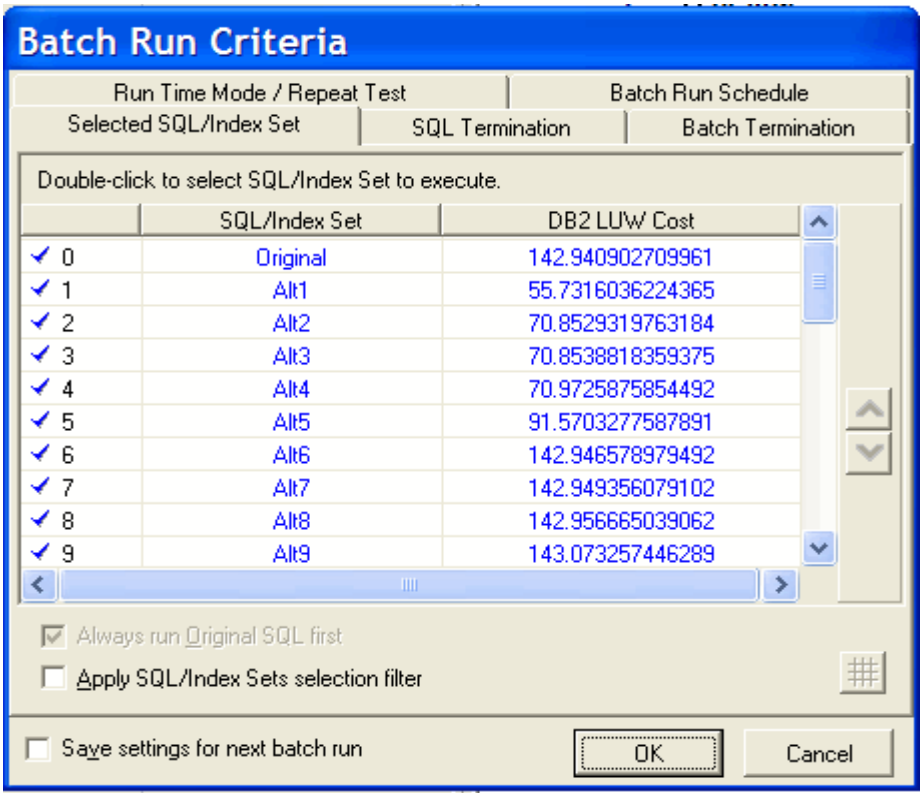
To review the Batch Run Details window

After the Batch Run is complete and the SQL Optimizer window is active, select **View | Show Batch Run Details**.

Batch Run Criteria Window

Selected SQL/Index Set

View the Batch Run Criteria Window—Selected SQL/Index Set



The Selected SQL/Index Set tab of the Batch Run Criteria window is used to select the SQL and index-set alternatives you want to execute. By default, all alternatives are initially selected.

Tip: To save all settings on the Batch Run Criteria dialog for subsequent batch runs, select **Save setting for the next batch run**.

Selecting SQL statements to test

To select or deselect an alternative

Double-click an SQL or index-set alternative.

To select or deselect all alternatives

Right-click and select **Select All** or **Unselect All**.

To select alternatives according to the cost values

1. Select the **Apply SQL/Index Sets selection filter** checkbox.
2. Click **SQL/Index Sets selection filter** .
3. Review the DB2 LUW cost retrieved from the virtual plans.
4. Check the **Exclude SQL/Index Set with DB2 LUW cost greater than** checkbox and enter a DB2 LUW cost.

Note: The original SQL statement can be de-selected only if the **Original SQL** option is not selected in the **SQL Termination** tab of the Batch Run Criteria window and the **Always run Original SQL first** option is not selected.

Changing the execution order of alternatives

The alternatives are ranked by DB2 LUW Cost by default, with the exception of the original SQL, which is placed at the top.

To change the execution order using the DB2 LUW cost

Click either the DB2 LUW or the SQL/Index Set column heading to reorder the alternatives for execution.

Note: High DB2 LUW Cost does not necessarily mean slower performance. If possible, it is recommended that all alternatives be tested.

To change the execution order for individual alternatives

Select the SQL or index-alternative; then click **Move Up** or **Move Down** .

Always run Original SQL first

The original SQL statement can always be run first even if it does not have the lowest DB2 LUW cost.

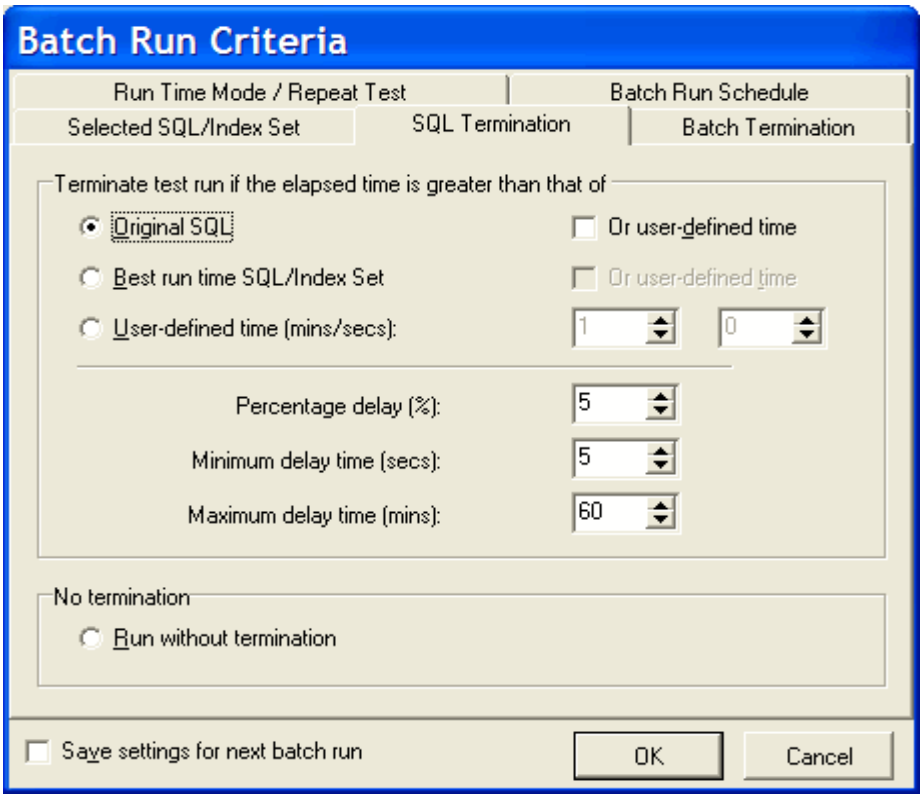
To always run the original SQL statement first

Select the **Always run Original SQL first** checkbox.

This checkbox is disabled if **Original SQL** is selected as a termination criterion on the SQL Termination tab or the Batch Termination tab since the run time from the original SQL is going to be used to terminate the alternatives or to terminate the Batch Run itself. In this case, you must run the original SQL statement first to obtain the termination time.

SQL Termination

View the Batch Run Criteria Window—SQL Termination



The SQL Termination tab of the Batch Run Criteria window is used to set the SQL termination criteria for each SQL and index-set alternative for which you are retrieving run times. If the current run time for a particular alternative exceeds the termination time, then the execution of the alternative terminates automatically. This will save time during testing.

Tip: To save all settings on the Batch Run Criteria dialog for subsequent batch runs, select **Save setting for the next batch run**.

Terminate test run if the elapsed time is greater than that of

Option	Description
Original SQL	Retrieves the run time of alternatives that run faster than the original SQL statement. It terminates all alternatives 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/Index Set tab. During the Batch Run, you cannot terminate the original SQL because its run time is needed to determine when to terminate the alternatives.
Best run time SQL/Index Set	Retrieves the run time of alternatives that run faster than the current best run time. With this option, the first alternative is run and the time from that statement is used as the termination time. When an alternative 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 alternative.

	Note: The first alternative is either the original SQL or the alternative with the lowest DB2 LUW Cost. This depends on whether the **Always run Original SQL first** option is checked on the Selected SQL/Index Set tab.
User-defined time	Retrieves the run time of alternatives that are less than the defined time. If your original SQL statement takes a long time to execute and there are many alternatives, executing all alternatives can 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 executes 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 run time SQL/Index Set** by clicking the **Or user-defined time** checkbox next to each one.

Percentage Delay

The percentage-delay calculation adds additional time to the termination time. It is used to account for the time it takes an alternative to travel from the local computer to the database server over the network.

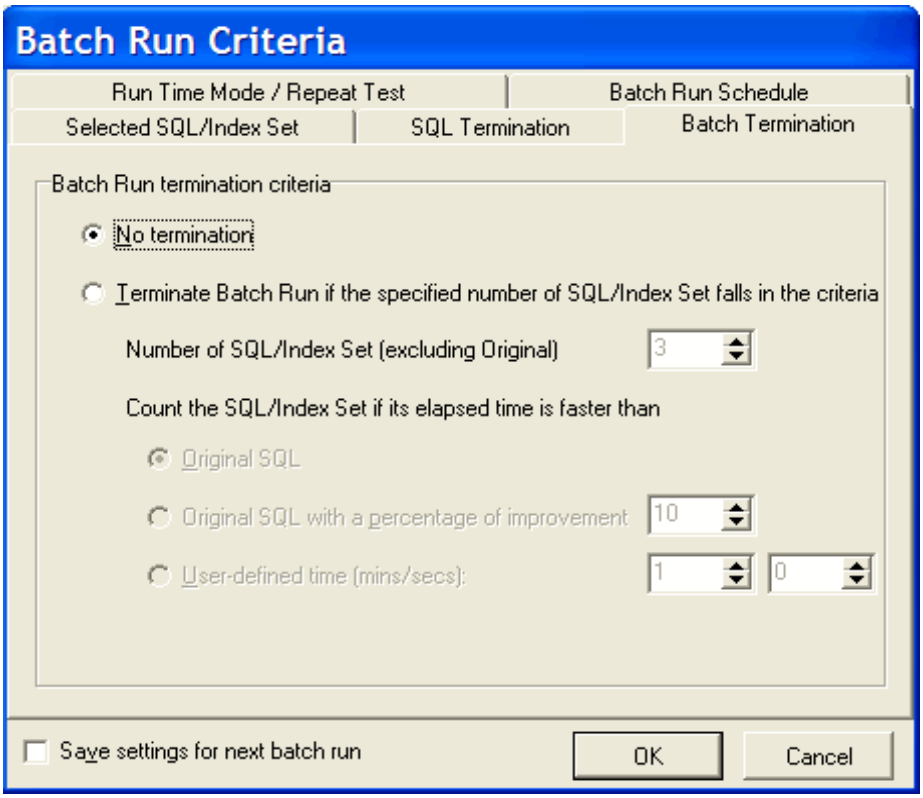
Option	Description
Percentage delay (%)	The value entered into this box is used as a percentage to calculate the additional time that is added to the termination time. For example, if the termination time is 10 minutes and the percentage delay is 5%, then all alternatives executed are terminated if the run time exceeds 10.5 minutes. (10+ (10 * 5%))
Minimum delay time (secs) (Default = *5*, Range 1 – 99)	This is 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 alternative be sent to the database server from the local computer before it starts to run. This number is only used if the percentage-delay calculation is lower than this value.
Maximum delay time (mins) (Default = 60, Range 1 – 9999)	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.

Run without Termination

Option	Description
Run without termination	Executes all the alternatives until they are completed.

Batch Termination

View the 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 SQL and index-set alternatives that give you performance improvement without having to execute every alternative.

Tip: To save all settings on the Batch Run Criteria dialog for subsequent batch runs, select **Save setting for the next batch run**.

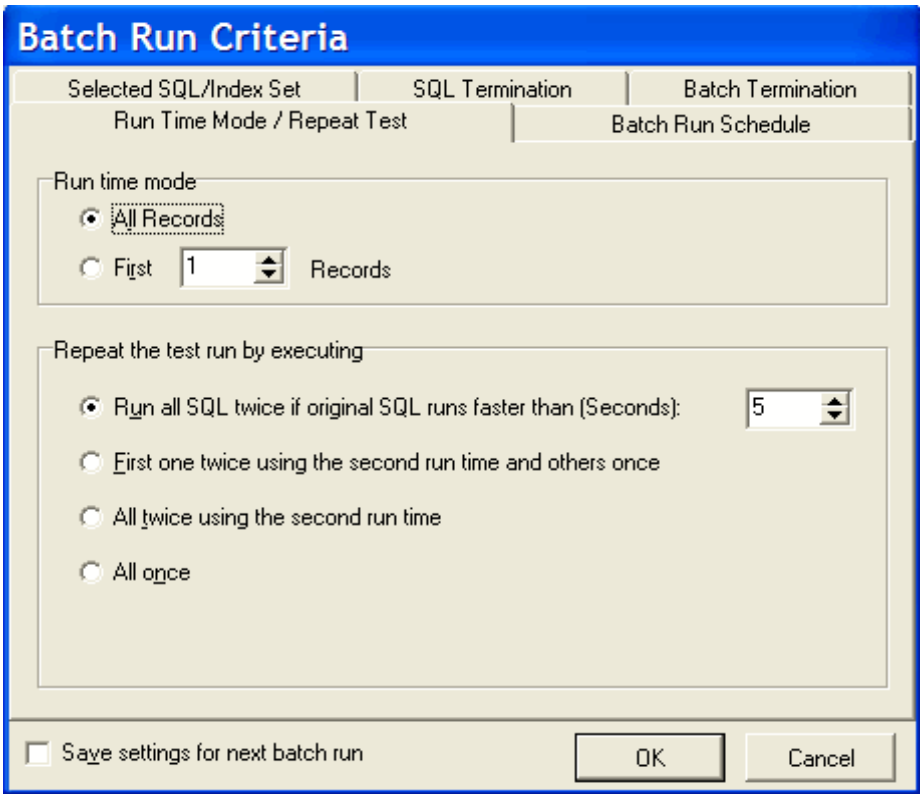
Batch Run Termination Criteria

Option	Description
No termination	Specify to run the Batch Run to completion.
Terminate Batch Run if the specified number of SQL/Index Sets falls in the criteria	Specify to terminate the Batch Run when a specified number of alternatives are found that meet the following requirements for terminating the Batch Run.
Number of SQL/Index Sets (excluding the Original)	Specify how many alternatives must be found that show performance improvement over the original SQL before the Batch Run will be terminated.
Count the SQL/Index Sets 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 alternatives that run faster than the elapsed run time from the original SQL.
Original SQL with a percentage of improvement	Count all alternatives where the elapsed run time for the alternative is the specified percentage faster than time for the original SQL statement.

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

Run Time Mode/Repeat Test

View the Batch Run Criteria Window—Run Time Mode/Repeat Test



The Run Time Mode/Repeat Test tab of the Batch Run Criteria window is divided into two sections.

Tip: To save all settings on the Batch Run Criteria dialog for subsequent batch runs, select **Save setting for the next batch run**.

Run Time Mode

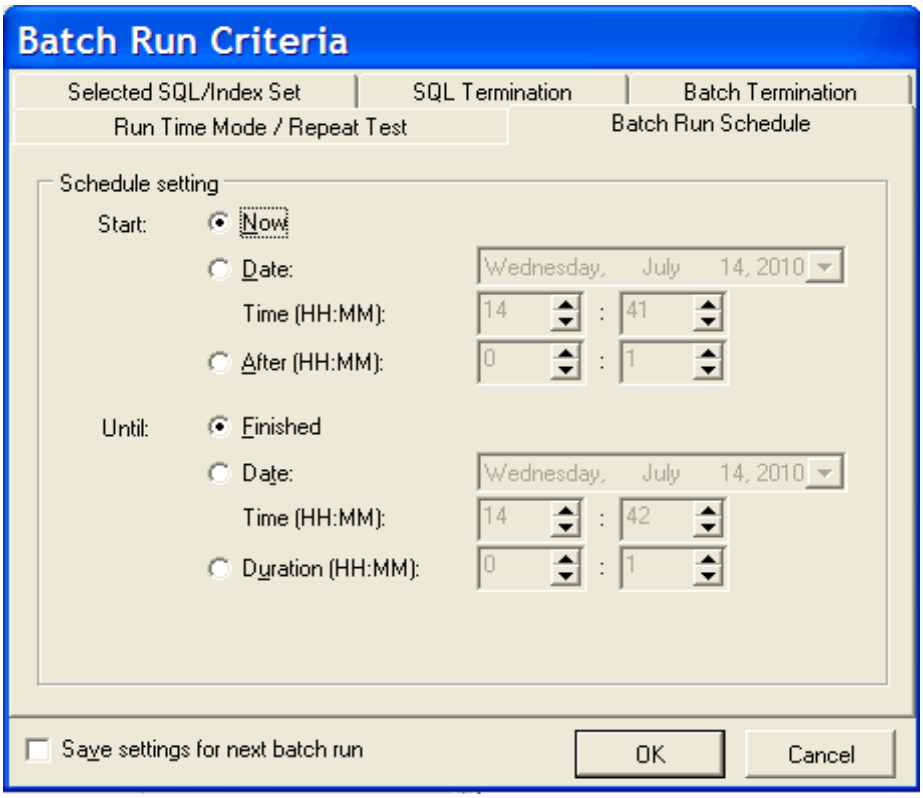
Run Time Mode	Description
All Records	Specify to retrieve the run time for processing all records.
First n Record (s)	Specify to retrieve the run time for processing *n* records where you specify the number of records retrieved.

Repeat the test run by executing

Repeat the test run by executing	Description
Run all SQL twice if original SQL runs faster than (Seconds)	This option executes the SQL and index-set alternatives twice if the run time for the original SQL is greater than the value you enter in the associated field.
First one twice using the second run time and others once	This option is suitable for SQL and index-set alternatives with normal run times. The first time you access data from table, the data is cached into memory. This process takes few moments. The next time you access that data, it is already in memory so the following alternatives will run faster. So to have a comparable test, the first SQL is run twice and the time from the second run is compared to the time from the other alternatives.
All twice using the second run time	This option is suitable for SQL and index-set alternatives that are often executed and have short running time. A significant portion of the running time of this kind of SQL is affected by the caching of index and the parsing time. To eliminate these factors, you can run all alternatives twice to use only the second run for a more comparable result.
All once	This option is suitable for long-running SQL and index-set alternatives, as it is unlikely that all the entire SQL statement is still kept in memory. There is no need to run any alternative twice since the caching of the table and the parsing time are not significant to the overall running time.

Batch Run Schedule

View the Batch Run Criteria Window (Batch Run Schedule)



The Batch Run Schedule criteria enable you to schedule when to start and stop the Batch Run.

Tip: To save all settings on the Batch Run Criteria dialog for subsequent batch runs, select **Save setting for the next batch run**.

Schedule setting

Start

Start	Description
Now	Specify to start the Batch Run immediately.
Date and Time	Specify to start the Batch Run at a specific date and time. If you click the month, day, year, you can use the up or down arrows to change the value.
After	Specify to start the Batch Run after it has been running for a specified number of hours and minutes.

Until

Until	Description
Finished	Specify to run the entire Batch Run until all alternatives are executed.

Date and Time	Specify to terminate the Batch Run on the specified date and time. If you click the month, day, year, you can use the up or down arrows to change the value.
Duration	Specify to terminate the Batch Run after it has executed a specified number of hours and minutes.

Save Optimized SQL

Save Optimized SQL

Use this procedure to save all the current SQL and index-set alternatives and their test results to a file that you can later reload into a SQL Optimizer session.

To save the current SQL and index-set alternatives

Select **SQL | Save Optimized SQL**.

Unlike a report, which captures information about the current alternatives in your SQL Optimizer session for viewing purposes, this process captures all the information needed to reload the current alternatives into a new session. See the Optimization Information for more information.

Open Optimized SQL

After you have saved the SQL and index-set alternatives to a file, you can load them and their test results back into the SQL Optimizer window.

To reload the saved alternatives

1. Select **SQL | Open Optimized SQL**.
2. Select the file you want to load and click **Open**. This loads the saved alternative into 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 Access plan Information

Displays the saved and current connection information and whether there are any changes in SQL structure and access plans.

User-Defined Temp Table Tab

If the SQL statement uses a temporary table, the User-Defined Temp Table tab appears 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 appears in this window. It displays the SQL text along with the saved and current access plan.

If there are any changes in SQL structure or access plans, you should refresh the access plans so that the reloaded image is a truth reflection of the current environment before any further testing is done.

To refresh the execution plans

Click **Refresh Plan** from the Open Optimized SQL Details dialog.

If there are changes in the access plan the corresponding SQL statements run time information is deleted. All invalid SQL statements are removed, except for the original SQL statement. You have an option to eliminate SQL statements with duplicate access plans.

After refresh, the Refresh Plan Details dialog can be displayed. This dialog displays the number of access plans refreshed, the total eliminated, and the total invalid plans.

To review the Open Optimized SQL Details dialog anytime

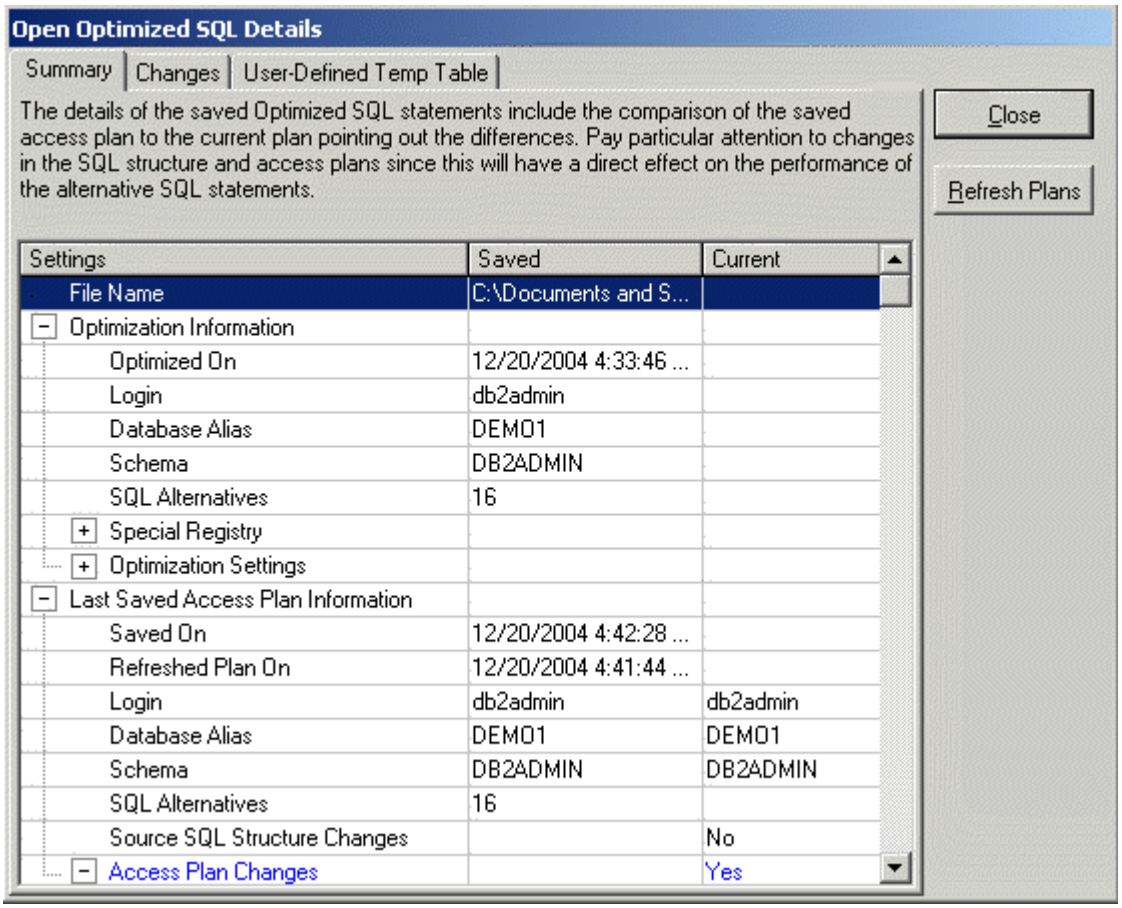
Select **View | Show Open Rewrite SQL Details**.

To review the Refresh Plans Details dialog anytime

Select **View | Refresh Plan Details**.

View Open Optimized SQL Details

View the Open Optimized SQL Details Window



The Open Optimized SQL Details window displays information about the SQL Rewrite and Generate Indexes processes and the resulting access plans. The Open Optimized SQL Details window appears after the SQL and index-set alternatives are reloaded into the SQL Optimizer window.

To review the Open Optimized SQL Details window

Select **View | Show Open Optimized SQL Details** when the SQL Optimizer window is active.

Note: The Optimized SQL Details cannot be viewed if the SQL Text 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 each SQL and index-set alternative and changes that have occurred in any of their access plans since the last SQL Rewrite or Generate Indexes was run.

Optimization Information

The optimization information includes the date and time of the last SQL Rewrite or Generate Indexes execution, how many alternatives were created, the connection information, and the optimization settings from the Options window at the time of the rewrite or index generation.

Last Saved Access Plan Information

The access plan information includes the date and time that the alternatives were saved, the number of alternatives created, connection information, and whether any of the access plans changed since the save. The changes are noted in the Current column as follows:

Current column	Explanation
Cost only	The only change to the access plan was a change in the DB2 LUW Cost.
Structure and cost	Both the structure and DB2 LUW cost of the access plan changed.
Structure only	The structure of the access plan changed but the DB2 LUW cost remained the same.
No	The current access plan is the same as the access plan that was saved with the SQL statement.

Refresh Plan button

The **Refresh Plan** button enables you to replace old access plans with the current plans. This process retrieves the current access plans, deletes any alternative that is now invalid, and removes any run time information. You can select to eliminate alternatives that now have duplicate access plans. After the access plans are refreshed, the Refresh Plan Details window is displayed.

Note: The **Refresh Plan** button is only enabled when there is a difference between the current access plan and the saved plan.

Changes Tab

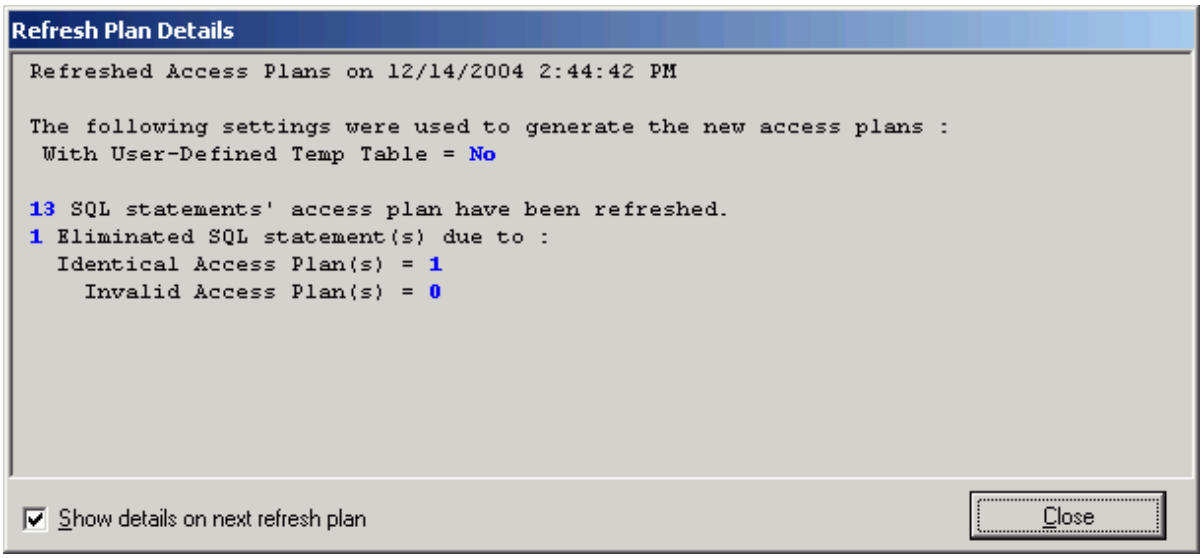
The Changes tab displays the text of the SQL statement in the top pane. The bottom pane displays the saved access plan and the current access plan side by side for comparison.

User-Defined Temp Table Tab

The User-Defined Temp Table tab is available only if the SQL statements use a temporary table. When the alternatives were saved, the DDL for the temporary table is also saved. When the alternatives are reloaded, the DDL is displayed.

Refresh Plan Details Window

View the Refresh Plan Details Window



The Refresh Plan Details window tells how many alternatives were refreshed and whether any of the alternatives were eliminated due to duplicate access plans or invalid SQL statements.

This window is only available after the saved access plans are replaced with the current access plans by clicking **Refresh Plan** on the Open Optimized SQL Details window.

SQL Formatter

About SQL Formatter

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 hence making the SQL statement easier to understand and read. The SQL Formatter not only formats the SQL statement but also checks its syntax, and highlights variables, DB2 optimizer SQL options, and comments.

Related Topics

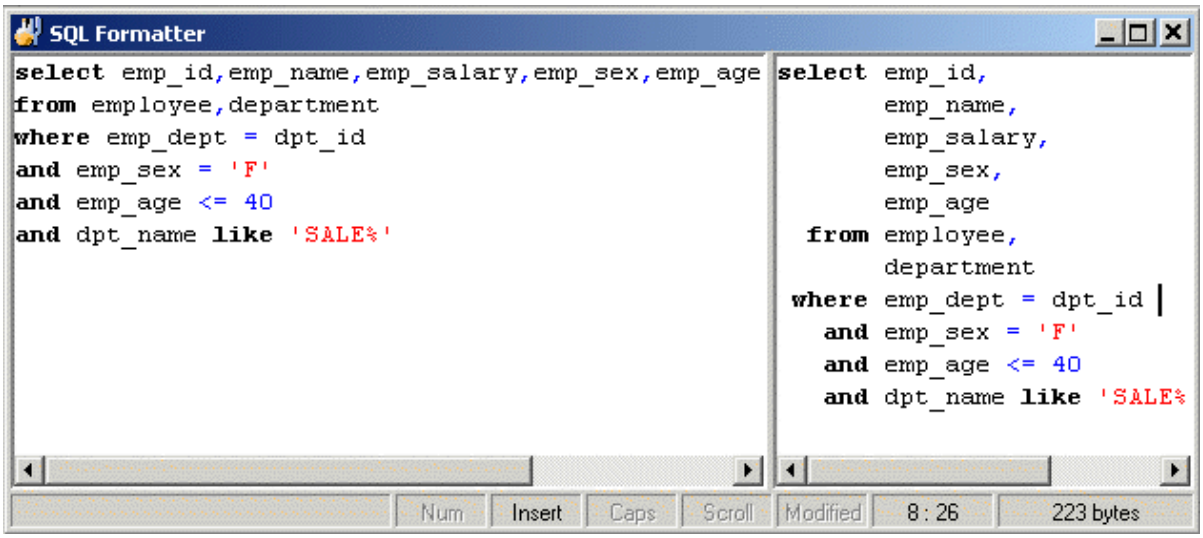
SQL Formatter Tutorial

SQL Formatter Window

SQL Formatter Functions

SQL Formatter Window

View the SQL Formatter



The SQL Formatter window is divided into two sections:

Original SQL (Left pane)

Provides an area for entering the SQL statement to be formatted.

Formatted SQL (Right pane)

Displays the formatted SQL statement. Host variables, SQL options and comments are color-coded for easy recognition.

Format a SQL Statement

To format a SQL statement

1. Click .
2. If the SQL Optimizer window is opened, the selected SQL statement will automatically be copied to the SQL Formatter window. Otherwise, enter a SQL statement onto the left pane] of the SQL Formatter window.
3. Click .

The SQL Formatter will check the syntax of the SQL statement. Only if the syntax is correct will the SQL statement be formatted. The formatted SQL statement. is displayed in the right pane of the SQL Formatter window.

To terminate the format process

- Click .

Host Variables within SQL Formatter

Declarations of variables within the SQL statement are allowed, prefixed with a ":" sign or without. If variables are recognized, they will be highlighted in red after executing the Format function.

Note: If table names are in red it indicates that the table and corresponding columns are not recognized. Check these things:

- Is there sufficient privileges under the login user to access the table
- Are you connected to the correct database and schema
- Execute the Synchronize Data Dictionary function to make sure you have the most up-to-date database information in memory.

SQL Formatter Functions

Below is a list of available functions within the SQL Formatter window.

Button or Menu	Function
SQL Menu	Format/Abort Format
Edit Menu	Send to SQL Rewrite
Edit Menu	Send to Generate Indexes
File Menu	Save SQL to SQL Repository
File Menu	Open SQL from SQL Repository

SQL Inspector

SQL Inspector Overview

The SQL Inspector identifies SQL statements with performance problems by extracting SQL statements and SQL performance statistics from the database. With the Add Inspector window, you can create an Inspector, select which approach to use, and specify your retrieval criteria and timeframe for capturing the SQL statements.

Related Topics

SQL Inspector Window

Add Inspector

Inspect

SQL Inspector Functions

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 Options window.

1. Click .
2. Select the **Directory Setup** tab.

The default setting for this directory is:

C:\Users*User***\Application Data\Quest Software\Quest SQL Optimizer for DB2 LUW\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 Topics

SQL Inspector Window

SQL Inspector Window

The SQL Inspector window consists of the Inspector List pane, 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.
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.
Last Modified	Date and time of when the Inspector was last modified.
Description	Description of the Inspector defined by the user.

Statistics pane

Displays statistics for each SQL statement retrieved from the database:

- Executed SQL
- Executing SQL

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 Topics

SQL Inspector Overview

Add Inspector

Inspect

SQL Inspector Functions

Open SQL Inspector

Statistics: Executed SQL

Each executed SQL statement has the following performance statistics:

Statistics	Description
Executions	Number of times the statement has been executed.
Logical Reads	Number of data pages read from the buffer pool (logical).
Physical Reads	Number of data pages read from the buffer pool (physical).
CPU Time	CPU time used by the statement.
Compilations	Number of different compilations for a specific statement.
Worst Preparation	The longest number of milliseconds required to prepare a specific statement.

Time	
Best Preparation Time	The shortest number of milliseconds required to prepare a specific statement.
Rows Deleted	Number of attempted row deletions.
Rows Inserted	Number of attempted row insertions.
Rows Updated	Number of attempted row updates.
Rows Read	Number of rows read to return result set.
Rows Written	Number of rows in the table changed (inserted, deleted or updated).
Stmt Sorts	Number of times a set of data was sorted.
Sort Overflows	Number of sorts that may have required temporary disk space for storage after running out of sort heap.
Sort Time	Total number of milliseconds for all executed sorts.
Total Execute Time	Total time spent to execute a particular statement in the SQL cache.
Stmt Text	Text of the statement
Partition	Database partition from which data was retrieved for the row.
Snapshot Time	Date and time of the snaphot.

Statistics: Executing SQL

Each executing SQL statement has the following performance statistics:

Statistics	Description
Executions	Number of times the statement has been executed.
Logical Reads	Number of data pages read from the buffer pool (logical).
Physical Reads	Number of data pages read from the buffer pool (physical).
CPU Time	CPU time used by the statement.
Compilations	Number of different compilations for a specific statement.
Worst Preparation Time	The longest number of milliseconds required to prepare a specific statement.
Best Preparation Time	The shortest number of milliseconds required to prepare a specific statement.
Rows Deleted	Number of attempted row deletions.
Rows Inserted	Number of attempted row insertions.
Rows Updated	Number of attempted row updates.
Rows Read	Number of rows read to return result set.
Rows Written	Number of rows in the table changed (inserted, deleted or updated).
Stmt Sorts	Number of times a set of data was sorted.
Sort Overflows	Number of sorts that may have required temporary disk space for storage after running out of sort heap.
Sort Time	Total number of milliseconds spend performing sorts when executing a section.
Total Execute Time	Total time spent to execute a particular statement in the SQL cache.
Stmt Text	Text of the statement
Partition	Database partition from which data was retrieved for the row.
Snapshot Time	Date and time of the snaphot.

Add Inspector

Add Inspector

To add an Inspector

Click .

General Information

The General Information page of the Add Inspector wizard consists of the following settings:

Item	Description
Inspecting Approach	Select from the following: • Executing SQL sent by an application or user within a set time period • Executing SQL sent by an application or user • SQL executed within a set time period • SQL executed at a set time
Inspector name	Enter a name for the Inspector.
Last Modified	The date and time the Inspector was last modified.
Description	Enter a description for the Inspector.

Related Topic

Add Inspector

Collection Period

When you collect SQL executed within a specific period, the Collection Period page of the Add Inspector wizard displays:

Item	Description
Start date	Specify the collection start date.
End date	Specify the collection end date.
Interval (minutes)	Specify the frequency of the collection in minutes.

Related Topic

Add Inspector

Collection Time

When you collect SQL executed before a specific time, the Collection Time page of the Add Inspector wizard displays:

Item		Description
Start collecting SQL only when the **Inspect** button is clicked		Start collection when you click the **Inspect** button.
Start collecting SQL at a specific time:	Date	Specify the date you want to collect SQL.
	Time (HH:MM):	Specify the time you want to collect SQL in hours and minutes.
Start collecting SQL after a wait period:	Wait time (HH:MM):	Specify the time you want to wait before you collect SQL in hours and minutes.

Related Topic

Add Inspector

SQL Filter

When you collect SQL executing before a specific time, the SQL Filter page of the Add Inspector wizard displays:

Item	Description
Select SQL Type	Select the appropriate checkbox for the SQL type you want to filter.
All SQL	Select to filter all SQL.

Related Topic

Add Inspector

Monitoring Period

When you collect SQL currently executing my monitoring an application our a user during a specific period, the Monitoring Period page of the Add Inspector wizard displays:

Item	Description
Start Date	Specify the start date of the monitoring period.
End Date	Specify the end date of the monitoring period.
Interval (minutes)	Specify how frequently the Inspector monitors SQL in milliseconds.

Related Topic

Add Inspector

Target Time

When you collect SQL currently executing by an application or a user at a specific moment, the Target Time page of the Add Inspector wizard displays:

Item		Description
Capture immediately the currently executing SQL when the **Inspect** button is clicked		Click **Inspect** button to start capture immediately.
Capture SQL executing at:	Date	Specify the date you want to capture executing SQL.
	Time (HH:MM):	Specify the time you want to capture executing SQL in hours and minutes.
Wait for some time and then capture what SQL are executing:	Wait time (HH:MM):	Specify the time you want to wait before you capture executing SQL in hours and minutes.

Related Topic

Add Inspector

Target Applications and Users

When you collect SQL currently executing, the Target Applications and Users page of the Add Inspector wizard displays:

Client Applications	Description
All Applications	Monitor all applications currently running.
Selected Application	Monitor a specific application.
Selected Agent ID	Displays Agent ID for the selected application.
Users	**Description**
All Users	All available users.
Selected User	Use the arrows to select from Available Users.

Related Topic

Add Inspector

Inspect

After adding or modifying an Inspector in the SQL Inspector window, SQL statements and statistics from the database are retrieved by executing the **Inspect** function. Only one Inspector can be started at a time.

To start the inspecting process

1. Select an Inspector job.
2. 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. Select an Inspector job
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. Select an Inspector job.
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
About SQL Scanner
SQL Inspector Overview
SQL Inspector Window

Delete 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

Find Inspector

To search for a specified text in all Inspectors

1. Select **Inspector | Find Inspectors**.
2. Enter the search text string
3. Click **Find**.

Related Topics
SQL Inspector Overview
SQL Inspector Window

View Inspector Properties

To view the Inspector properties

Right-click an Inspector and select **Properties**.

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

SQL Inspector Window

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.

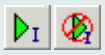
To create a file to import into Benchmark Factory

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.

Sum Statistics for a Stored Procedure

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	Delete 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
	Refresh
Right-click Menu	Create Benchmark Factory Import File

Related Topics

SQL Inspector Overview

SQL Inspector Window

Database Explorer

About Database Explorer

The Database Explorer accesses information on database objects from the database server and provides details on tables, views, aliases, nicknames, procedures, functions, triggers, packages and user-defined types. The type of information given is dependent on the object type and your privileges to access specific database objects.

Related Topics

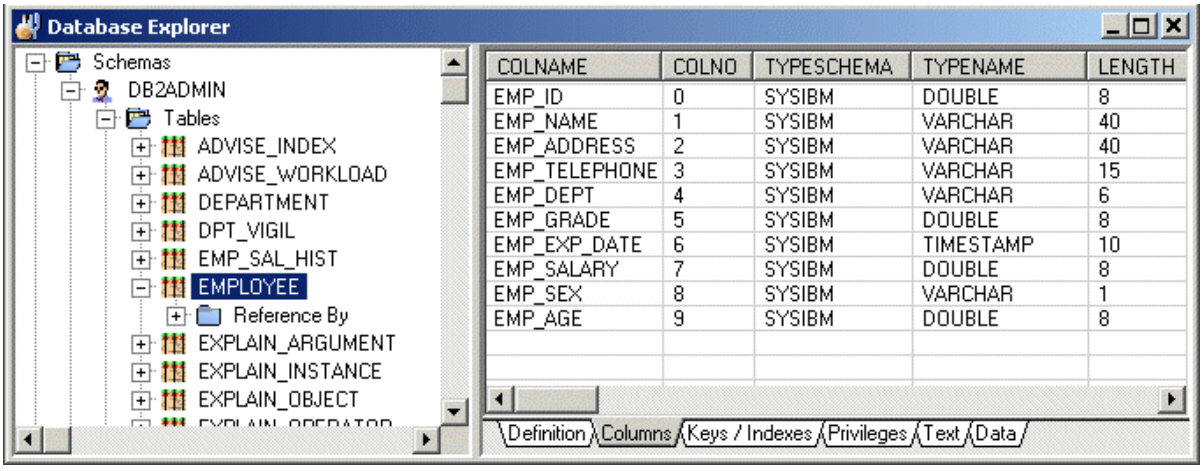
Database Explorer Tutorial

Database Explorer Window

Database Explorer Functions

Database Explorer Window

View the Database Explorer



The Database Explorer window is divided into two sections:

Database Objects

Displays a tree diagram of the database objects these include schemas, tables, views, aliases, procedures, functions, triggers, packages, nicknames and user-defined types.

Object Information

Displays relevant information of the selected database object.

Privileges

User privileges within the database server are directly reflected in the Database Explorer. Schema with full access will be able to view all database objects within the catalog. Otherwise, those granted to the schema will be viewable.

Open Database Explorer

To open the Database Explorer window

1. Click .

Sort Criteria

The current schema is presented as the first item of the Database Objects tree followed by the remaining names in alphabetical order.

Modify Data

When the data is display, a button bar appears at the bottom of the pane when you can edit the data. This button bar offers functions to view, edit, insert, and delete data.

To navigate through the data

- Use the scroll bar.
  or
- Use the navigator buttons. Depending on the amount of data, scrolling to the bottom of the page or executing the **Last Record** function may take sometime as all records needs to be retrieved from the database.

To edit a record

1. Place the cursor on the record you want to edit.
2. Click . You will be able to change the data for that record in the data grid.
3. Press the tab key or use the mouse to move to the next column. For LONG, CLOB, and BLOB data types,

double-click the column to bring up the column edit window.

4. Once you have completed the modifications, click **Commit** or **Rollback** .

To insert a new record

1. Place the cursor on the record on which you want to insert the new record before it .
2. Click .
3. Enter data for each column, press the tab key or use the mouse to move to the next column. For LONG, CLOB, and BLOB data types, double-click the column to bring up the column edit window.
4. Once you have completed entering the data, click **Commit** or **Rollback** .

To delete a record

1. Place the cursor on the record you want to delete.
2. Click .
3. When you click **OK** on the confirmation dialog the deletion will be automatically committed to your database.

To refresh the data

1. Click .
2. If an INSERT or UPDATE was not committed, you will be prompted to commit before the data is refreshed.

Note: You cannot update some views. 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.

Copy Column Names to Another Window

To facilitate the construction of an SQL statement, copy the selected column names to the SQL Optimizer or SQL Formatter windows. A new line and a comma will separate each column name when it is copied to the editor pane of the other module.

To copy column names

1. Select the table or view name displayed on the tree diagram in the left pane of the Database Explorer window.
2. In the right pane, select the Column tab.
3. Select the column names by clicking the left-hand mouse button with the Shift or Ctrl key to select multiple columns.

4. Do one of the following:
    - Right-click and select **Copy columns to SQL Optimizer**.
    - Right-click and select **Copy columns to SQL Formatter.**

Copy Data from Cells, Rows, or Columns

You can copy the data from a single cell, a row, or a column into the Windows clipboard from the Database Explorer and the Parameters windows.

To copy data

1. Select the table or view name displayed on the tree diagram in the left pane of the Database Explorer window.
2. In the right pane, select the **Data** tab.
3. Highlight a cell with data.
4. Right-click and select **Copy | Cell**, **Row**, or **Column**.

View LONG, CLOB and BLOB Columns

You can view the data in a column that is a LONG, CLOB, or BLOB data type.

To view the data

1. Select the table or view name displayed on the tree diagram in the left pane of the Database Explorer window.
2. In the right pane, select the **Data** tab.
3. Double-clicking the cell in the data grid.

Scan Database Objects

The SQL statements in the DDL for a database object 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 DDL for a database object from the Database Explorer window

1. In the left pane, right-click the database object name and select **Scan**.
2. If the Job Manager window is opened, then the selected database object is added as a job in the window and the scanning starts automatically. If the Job Manager window is not open, the Group Manager window is opened for you to select or create a Job Group first.

Database Explorer Functions

Below is a list of available functions within the Database Explorer window.

Button or Menu	Function
Edit Menu	Send to SQL Rewrite
Edit Menu	Send to Generate Indexes
Right-click Menu	Scan
Right-click Menu	Copy columns to SQL Optimizer
Right-click Menu	Copy columns to SQL Formatter
Data Tab	Modifying Data

SQL Repository

About SQL Repository

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.

The SQL Repository is stored in a file directory that can be accessed by your computer, either a directory on your local computer or a network drive. This directory is specified in the Options settings.

Related Topics

SQL Repository Tutorial

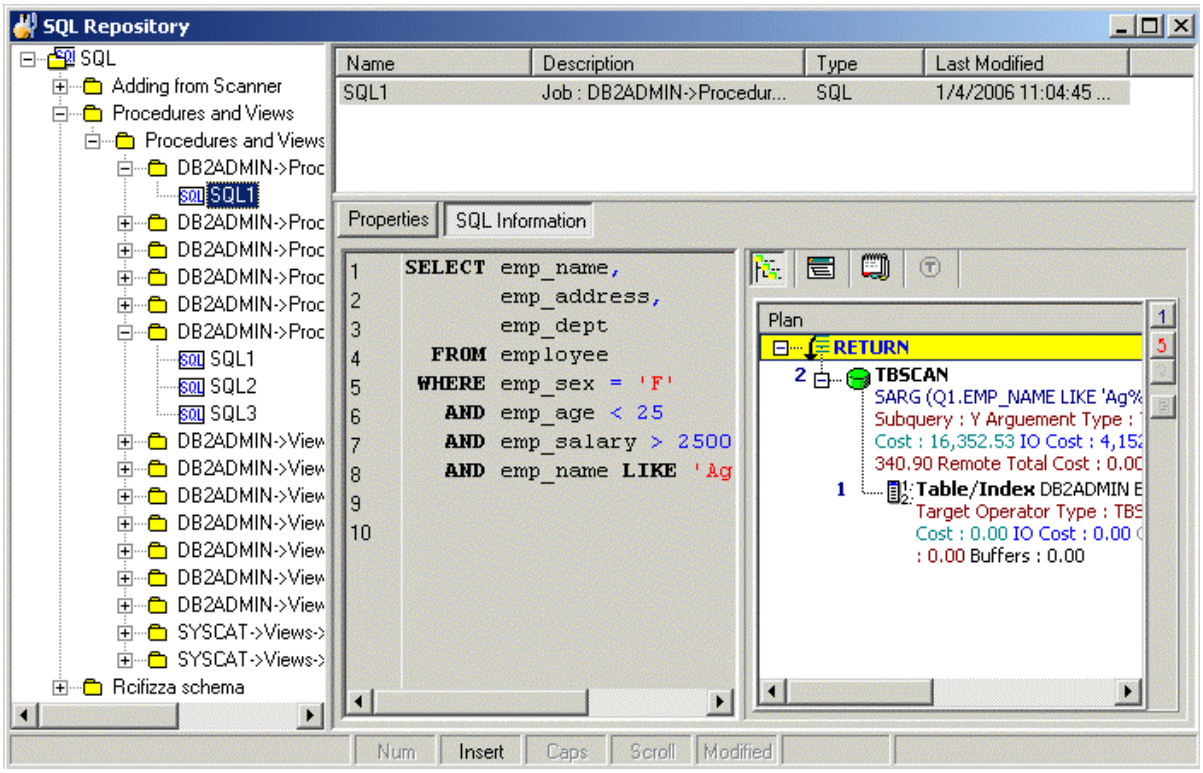
SQL Repository Window

SQL Repository Functions

Saving SQL to the SQL Repository from Other Modules Tutorial

SQL Repository Window

View the 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 is displayed 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 SQL Information pane where you can select to display the SQL text, the access plan, SQL classification, and DDL for any temporary tables used in the SQL statement.

Add SQL to SQL Repository

SQL statements are stored in the SQL Repository for use in an Index Impact Analysis, and an Index Usage Analysis.

To add SQL statements

1. Click .
2. If no SQL exists in the SQL Repository Add SQL wizard appears automatically. Otherwise, click .
3. Select **SQL | Add SQL** or **Add SQL from SQL Scanner**.
4. Use the following pages in the Add SQL wizard has pages to select the options for saving the SQL statement to the SQL Repository.
   General
   SQL Information

The SQL syntax is checked and the access 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 an access plan, SQL classification type (Simple, Complex or Problematic) and the current connection information (login name, database alias, and schema). The access plan stored with the SQL statement is important as it indicates the current performance of the SQL.

SQL statements can also be added to the SQL Repository from other modules.

Add SQL Wizard: General Page

The General tab 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.
Database alias	Displays the connected database alias.
Schema	Displays the connected schema.
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

Item	Description
	create a new folder, click .

Add SQL Wizard: SQL Information Page

The SQL Information page of the Add SQL wizard is where you can enter the SQL text and view information about the SQL statement, such as the access plans and temp table DDL.

SQL Text (Left pane)

Enter the SQL text.

Information Pane (Right Pane)

The SQL Information pane has buttons at the top of it to select whether to display the access plan graph, access plan tree, Information (the classification of the SQL), and the DDL for creating any temporary tables used by the SQL statement. If the button is enabled, there is information on that page. If the button is , then no information is available for that SQL statement.

Check SQL button

Checks the SQL syntax using the current database connection and retrieves the access plan, SQL type classification, and other information relating to the SQL statement.

Refresh SQL Access Plan

If you feel the access plan stored with the SQL statement in the SQL Repository does not represent the current performance due to database changes, you can retrieve the current access plan from the database.

To retrieve the current access plan

1. From the SQL Repository window, select the SQL.
2. Select **SQL | Refresh Plan** or right-click the SQL statement in the tree view and select Refresh Plan
3. In the Refresh Plan window the original access plan is displayed on the left pane while the newly retrieved access plan is displayed on the right pane for comparison. You can view and analyze the newly retrieved access plan and SQL classification before saving the new access plan with the SQL statement.
4. Click **Save** to replace the original access plan with the newly retrieved one.

Modify SQL

You can only modify the SQL name and description of the SQL statements stored in the SQL Repository.

To modify the 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 to open the Modify SQL window.
3. Modify the name and/or description.

Note: SQL text cannot be modified. You must delete the current SQL and recreating a new one to change the SQL text.

Rename SQL

You can rename a SQL statement stored in the SQL Repository window in two ways:

To rename a SQL statement

1. Select the SQL you want to modify from the left pane.
2. Right-click and select **Rename**. Change the name and press **Enter**.
   or
   Select **Modify** and change the name.

Delete SQL

When you delete a SQL statement in the SQL Repository, it will delete all references.

To delete a SQL statement stored in the SQL Repository

1. Select the SQL you want to delete.
2. Right-click and select Delete or press the **Delete** key to delete the selected SQL statement and all references to this SQL statement in other modules.
3. If the SQL statement is in an Index Impact Analysis or an Index Usage Analysis, you will be prompted before it is deleted.

Create Benchmark Factory Import File

All the SQL statements can be saved in a text file from the Job Manager, the Scanned SQL Viewer, the SQL Repository, and the SQL Optimizer windows. These SQL statements can then be imported into Benchmark Factory program (version 4.6 or later).

To create the text 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. Click **OK**.
3. Enter the filename and select the file location. Click **Save**.

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 & Right-click Menu	Add SQL from SQL Scanner
SQL & Right-click Menu	Refresh Plan
SQL & Right-click Menu	Add Folder
Right-click Menu	Modify
Right-click Menu	Delete
Right-click Menu	Rename
Right-click Menu	Create Benchmark Factory Import File

Index Impact Analyzer

About Index Impact Analyzer

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. The 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 that creating new indexes might have on SQL performance, the proposed indexes are created as virtual indexes on the database. Then new access plans are retrieved showing the impact the new indexes would on the access plans of individual SQL statements. The virtual indexes are dropped after the retrieval of the access 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 have changes to access plan with the proposed indexes.

Related Topics

Index Impact Analyzer Tutorial

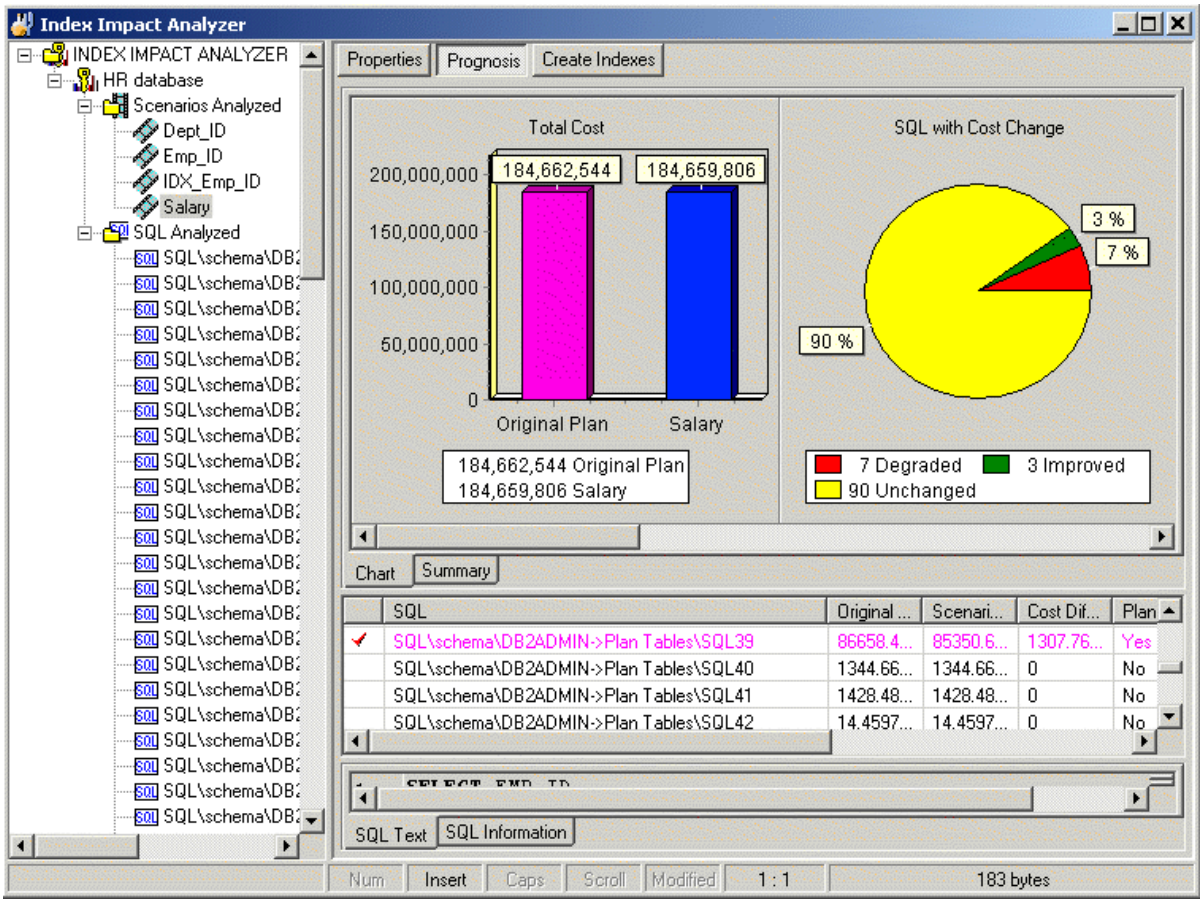
Index Impact Analyzer Window

Index Impact Analyzer Functions

Index Impact Analyzer Window

Index Impact Analyzer Window

View the 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

Right Pane for Analyzers

In the Index Impact Analyzer window, when an Analyzer is selected in the left pane.

Properties Button

The Properties displays the following information:

- General information about the analysis
- Connection information used to retrieve the access plans and the SQL statements used in the analysis
- Connection information used to retrieve the Baseline access plans
- Indexes used for each Scenario
- SQL Information for each SQL statement used in the analysis

Right Pane for Scenarios Analyzed Folder

In the Index Impact Analyzer pane, the following information is displayed in the right pane.

Summary button

Provides a summary of the DB2 LUW Cost and SQL classification for the SQL statements before and after the proposed indexes were created.

Chart button

Displays a bar chart which compares the accumulative total of the DB2 LUW Cost for the Baseline to each of the Scenarios. The Baseline shows the accumulative total of the DB2 LUW Cost of the SQL statements without the proposed indexes. The other bars show the accumulative total of the DB2 LUW Cost with the each proposed index set.

Right Pane for Scenario

In the Index Impact Analyzer window, when the Scenario folder is selected in the left pane, the following information is displayed 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 charts of the comparison information for the access plans from before the analysis and during the analysis. These charts compare:

- Total DB2 LUW cost for all SQL statements
- Number of SQL statements with changes in the DB2 LUW cost
- Number of SQL statements that use the indexes in the index set
- Number of SQL statements whose access plan is different from the Baseline access plan
- Number of SQL statements whose SQL classification changed
- Number of SQL statements in each of the SQL classification categories

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 access 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 is displayed 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 without the index set and one for the SQL Information with the index set. The SQL Information panes contain buttons that show the access plans, DB2 LUW Optimized Text, SQL Classification, and any temporary tables used for the statement with and

without the Index Set. As you review the access plans, the differences between the plan with and without the index set will be highlight in green to make it easier to find the differences.

Create Indexes button

Displays the script for creating the index or indexes.

To create the indexes

1. Review the index name and, if desired, modify it.
2. Click .

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 is displayed in the right pane.

Cost Summary button

Displays all the SQL statements and shows the DB2 LUW cost for the SQL statement in each Scenario so that you can compare the cost of each SQL statement to see if the cost increases or decreases with the index set. For each SQL statement, it displays the highest DB2 LUW cost in red and the lowest cost in green.

SQL Classification button

Displays all the SQL statements and shows the SQL Classification (Simple, Complex, and Problematic). The Baseline column shows the SQL Classification without the proposed indexes. The other columns show the SQL Classification with each proposed index set.

Right Pane for SQL Statements

In the Index Impact Analyzer window, when a SQL statement is selected in the left pane, the following information is displayed in the right pane.

Properties button

Displays general information about the analysis, information on the connection used to retrieve the access plans at the time of the analysis.

SQL Information button

Displays the SQL statement selected as well as two SQL Information panes that show the access plans, DB2 LUW Optimized Text, SQL Classification, and any temporary tables used for the statement with and without the Index Set. As you review the access plans, the differences between the plan with and without the index set will be highlight in green to make it easier to find the differences.

Create an Index Impact Analyzer

To create an Index Impact Analyzer

1. Click .
2. If no analysis exists in the Index Impact Analyzer, then the New Analysis wizard appears automatically.
   Otherwise, click .
3. Use the following pages in the New Analysis wizard to select the options for creating an analysis.
   Analyzer
   Selected SQL
   Index

New Analysis Wizard: Analyzer Page

In the Analyzer page, you can select to create an new Analyzer or to use an existing Analyzer with the SQL statements that are already in the SQL Repository for that Analyzer.

Select a new or existing Analyzer

Create a new Analyzer and select SQL to analyze

Specify to create an new Analyzer and to then select the SQL statements from the SQL Repository that you want to use in this new Analyzer.

Continue an existing Analyzer using the Analyzer's selected SQL

Specify to use an Analyzer that you have previously created with the SQL statements that are already stored with that Analyzer.

Analyzer Information

Item	Description
Name	If you are creating a new Analyzer, enter a name. If you are using an existing Analyzer, select the from the **Select Analyzer** box.
Description	If you are creating a new Analyzer, 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. In the bottom pane, select the folder where you want to save the Analyzer. To create a new folder, click .

New Analysis Wizard: Select SQL Page

In the New Analysis wizard, the Select SQL page is used to select the SQL statements whose access 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 access 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 access plans from before the analysis can be obtained two ways:

Using existing access plan saved with the SQL

This option uses the access plan that was saved with the SQL statement when it was saved in the SQL Repository or SQL Scanner.

Obtaining a new access plan under the current connection

This option retrieves the access plan with the current logon and the current database settings. This current access plan is compared to the access plan that is retrieved after the new indexes are created.

New Analysis Wizard: Index Page

In the New Analysis wizard, the Index page is used to specify the index(es) that are used for this analysis.

Access plans will be grouped under a Scenario

An individual Index Impact Analyzer consists of a group of stored SQL statements with the access plans called the " Baseline." The actual Impact Analysis is displayed in a "Scenario." A Scenario shows the comparison information for the stored SQL statements identifying the impact on the access plan that the creation of new index will have.

Item	Description
Name	Enter a name for the Scenario.
Description	Enter a description.
Add Index	Click .

Item	Description

Item	Description
Index Name	Enter a name for the index.
Schema	Select a name for the index.
Table	Select a table for the index.
Table columns	Double-click one or more columns.
Index type: Unique, Clustered, Allow Reverse Scans	Select the type of index.

Remove Index Click .

After you have created the first Scenario, you can add more Scenarios.

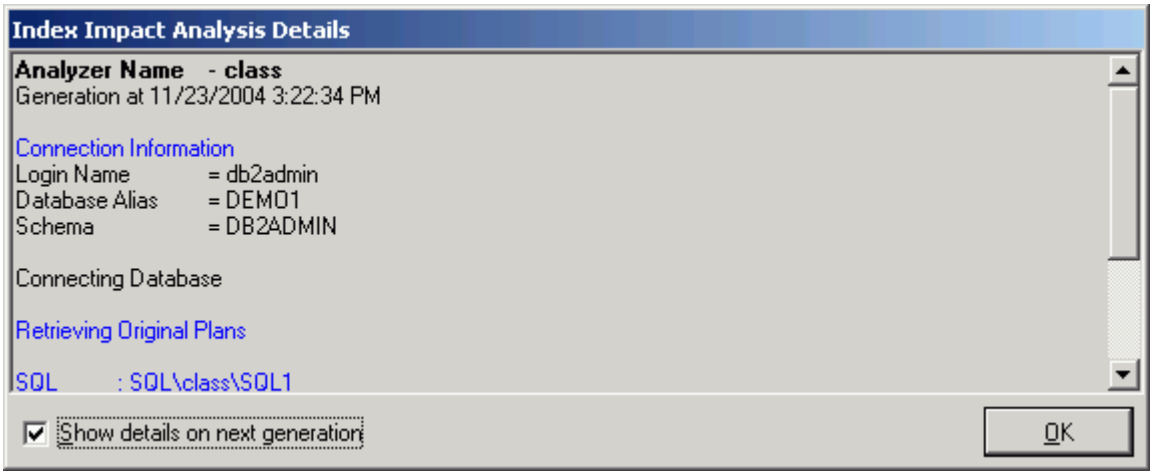
Add Index Window

The Add Index window is used to create the DDL for creating of the indexes for the Index Impact Analysis.

Item	Description
Index name	Enter a name for the index. By default, the prefix of the index name is taken from the Index Expert settings in the Options window and then a random number is generated for the rest of the index name. **Note:** It is best not to change the prefix for the index. The indexes are generated as virtual indexes. If the Impact Analysis process terminates abnormally and a virtual index is left on the database, it can be easily identified as a virtual index which was created by SQL Optimizer by the prefix.
Database	Select the database where the index is to be created from the database 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 highlighting the column and clicking .
Unique	Specify to create a unique index.
Clustered/Nonclustered	Select to create either a clustered or a nonclustered index.

View Index Impact Analysis Details

View the Index Impact Analysis Details Window



The Index Impact Analysis Details window displays information about the creation of the indexes and the retrieval of the access plans. The Index Impact Analysis Details window appears 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 view the Index Impact Analysis Details window

After an Index Impact Analysis is complete, select **View | Last Index Impact Analysis Details** when the Index Impact Analyzer window is active.

Review this information to see if an error occurred during the retrieval of the access plan. Errors may occur if the selected SQL statements are from one schema and you have set another schema in the Schema list at the bottom left of the main window.

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. This process will re-execute the Scenario.

To add SQL statements

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.

Add Scenario

After creating an Index Impact Analysis, you can add more Scenarios to it. A Scenario can be added in two ways:

To add a Scenario to an Analyzer

Method One

1. In the left pane, right-click the Analyzer and select **Add Scenario**.
2. Click the Index page and specify the information for the proposed new indexes.

Method Two

1. In the left pane, right-click an Analyzer and select **New Analysis**.
2. Select the **Continuing an existing Analyzer using the Analyzer's selected SQL** option.
3. From the Select Analyzer box, click the Analyzer name.
4. Click the Index page and specify the information for the proposed new indexes.

Modify Index Impact Analyzer

To modify the analysis name or description

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 or description.

Regenerate

The Scenario can be re-executed using the same indexes. You can choose to retrieve the access plans for all SQL statements in the Scenario or select specific SQL to retrieve the access plan again using the same indexes.

The function is useful if changes have occurred in the database environment, such as data volume changes, that may affect access plans.

To regenerate the access plan

1. Select the Scenario from the left pane.
2. Right-click and select **Regenerate**.
3. Check the SQL statements whose current access plans you would like retrieved again.

Rename an Index Impact Analyzer

You can rename an analysis in the Index Impact Analyzer window in two ways.

To rename an Analyzer

1. Select the Analysis you want to modify from the left pane.
2. Right-click and select **Rename**. Change the name and press **Enter**.

or

Select **Modify** and change the name.

Delete an Index Impact Analysis

To delete an analysis

1. In the Index Impact Analyzer window, select the analysis you want to delete in the left pane.
2. Right-click and select **Delete**.

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
	Create Indexes

Index Usage Analyzer

About Index Usage Analyzer

The Index Usage Analyzer identifies unused indexes by analyzing access plans from SQL statements in your database applications. It examines their access 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 access plans, and the number of referenced SQL for each index
- Indexes in each table that are not used in the access 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 Tutorial

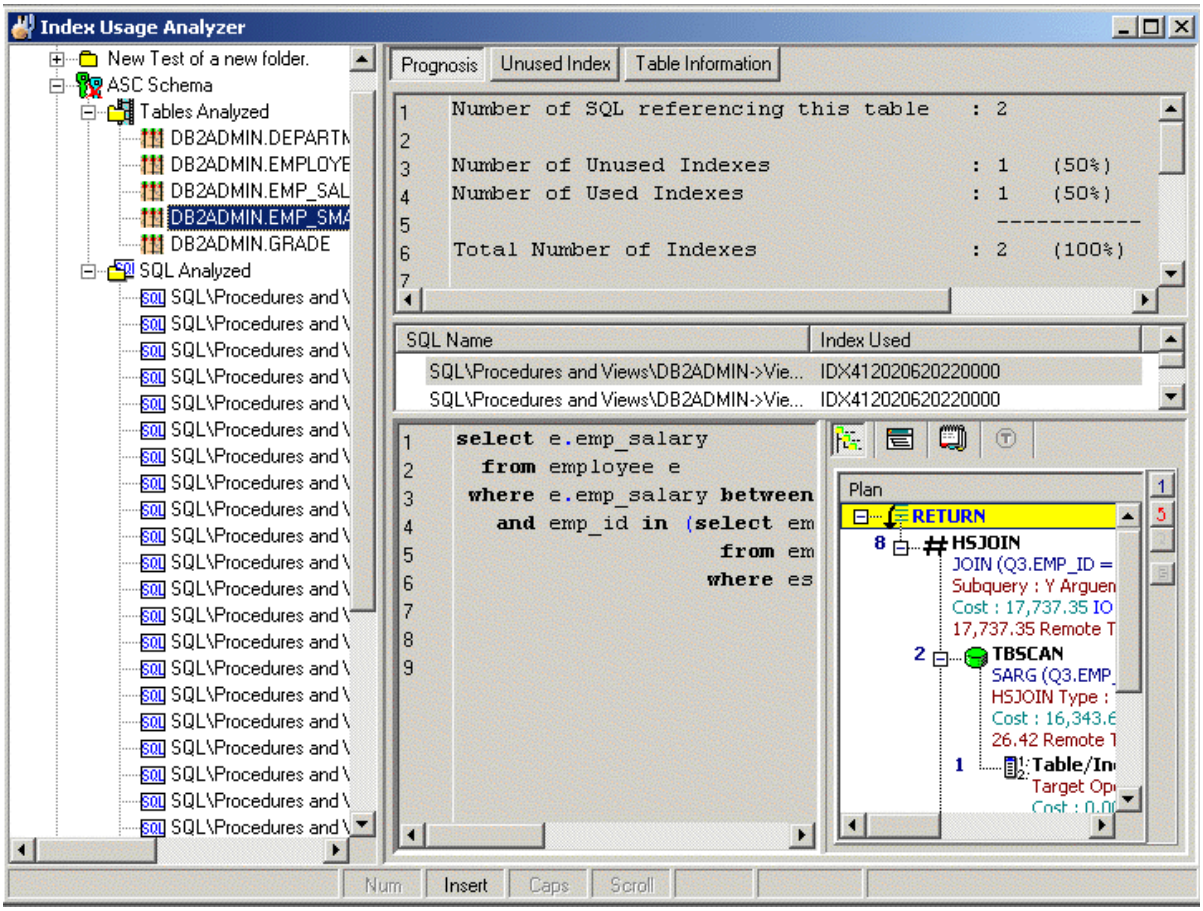
Index Usage Analyzer Window

Index Usage Analyzer Functions

Index Usage Analyzer Window

Index Usage Analyzer Window

View the 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 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

Table Analyzed

Table Information

SQL Analyzed Folder

SQL Statements

Right Pane for Analyzers

In the Index Usage Analyzer window, when an Analyzer is selected in the left pane, the following information is displayed 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 access plans for the selected SQL statements.

Analysis Information (Bottom right pane)

The detailed information about the Analysis is displayed 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 access 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.

To save a list of all the unused indexes

1. Right-click and select **Save**.
2. In the Save Format window, select **Text**, **HTML**, or **Excel Convertible (Spreadsheet)**.

To drop an index

Right-click the line with the index you want to drop and select **Drop Index**.

Right Pane for Tables Analyzed

In the Index Usage Analyzer window, when the **Tables Analyzed** is selected in the left pane, the following information is displayed 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.

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 is displayed in the right pane.

Prognosis Button

Pane	Function	Description
Top	Summary Information	Displays a summary of the indexes in the table that are used or unused.
Middle	Index Used Information	Displays the indexes used by the SQL statement.
Bottom Left	SQL Text	Displays the text of the SQL statement.
Bottom Right	SQL Information	Displays the access plan, 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.

To save a list of all the unused indexes

1. Right-click and select **Save**.
2. In the Save Format window, select **Text**, **HTML**, or **Excel Convertible (Spreadsheet)**.

To drop an index

Right-click the line with the index you want to drop and select **Drop Index**.

Table Information Button

Tab	Description
Definition	Displays when and where the table was created.
Columns	Displays the column information for the table.
Index	Displays the index information for the table.
DDL	Displays the DDL for creating the table and indexes.

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 is displayed in the right pane.

Index Used

Displays all the SQL statements and shows the tables and indexes used.

Right Pane for SQL Statements

When a SQL statement is selected in the left pane, the following information is displayed in the right pane.

Properties button

Displays general information about the analysis, information on the connection used to retrieve the access plans and the database settings at the time of the analysis.

SQL Information button

Displays the access plan and the SQL Information pane.

Index Used button

Lists the table name, indexes and the key for all indexes used in the SQL statement.

To save a list of all the used indexes

1. Right-click and select **Save**.
2. In the Save Format window, select **Text**, **HTML**, or **Excel Convertible (Spreadsheet)**.

Create an Index Usage Analysis

To create an Index Usage Analysis

1. Click .
2. If no analysis exists in the Index Usage Analyzer, then the New Analysis wizard appears automatically. Otherwise, you can open the New Analysis wizard in the following ways:
3. Click .
3. Use the following pages in the New Analysis wizard to select the options for creating an analysis.

Analyzer

Selected SQL

When the analysis is finished, the Index Usage Analysis Details window appears with information about the retrieval of the access plan for each SQL statement.

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**.
2. The Update Analysis wizard has two pages to select the options for changing an analysis before it is updated. It is not necessary to make any changes on these pages before rerunning the analysis.

On the Analyzer page, you can change the name and description of the analysis.

On the Selected SQL page, you can add or delete SQL statements to the analysis.

When the analysis is finished, the Index Usage Analysis Details window appears with information the retrieve of the access plan for each SQL statement.

New Analysis Wizard: Analyzer Page

The Analyzer tab 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 **Select save location** pane, select the folder where you want to save the analysis. To create a new folder, click .

New Analysis Wizard: Select SQL Page

The Select SQL tab is used to select the SQL statements whose access plans you want to analyze to see which Indexes are used and which are not used by these SQL statements.

Select SQL from

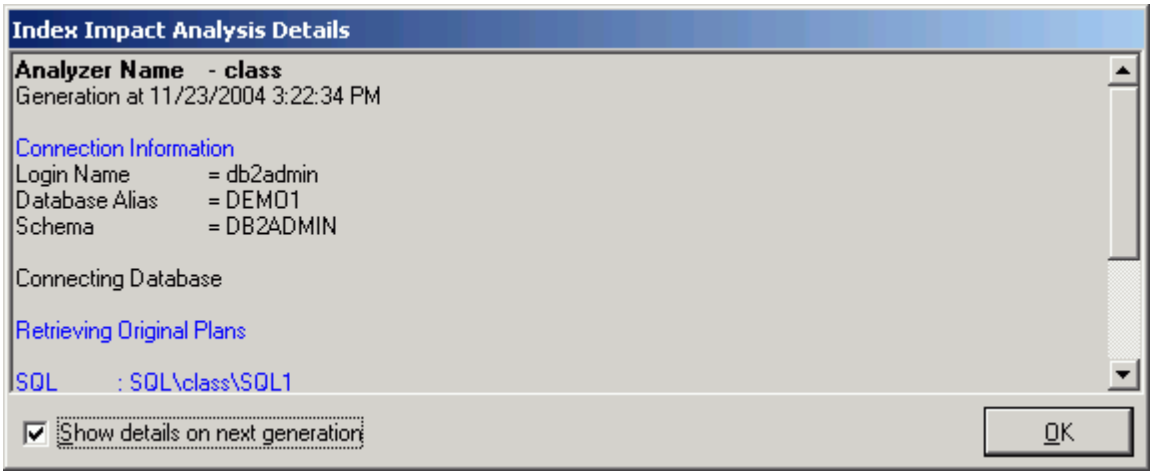
The SQL statements used in an analysis can be taken from the SQL Repository or the SQL Scanner.

SQL Tree

Select the SQL statements for the analysis.

View Index Usage Analysis Details

View the Index Usage Analysis Details Window



The Index Usage Analysis Details window displays information about the retrieve of the access plan for each SQL statement.

The Index Usage Analysis Details window appears 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 view the Index Usage Analysis Details window

After an Index Usage Analysis is complete, select **View | Last Index Usage Analysis Details** when the Index User Analyzer window is active.

Review this information to see if an error occurred during the retrieval of the access plan. Errors may occur if the selected SQL statements are from one schema and you have set another schema in the Schema list at the bottom left of the main window.

Modify an Index Usage Analysis

To modify the analysis name or description

1. From the Index Usage 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 or description.

Rename an Index Usage Analysis

You can rename an analysis in the Index Usage Analyzer window in two ways.

To rename an Analyzer

1. Select the Analysis you want to modify from the left pane.
2. Right-click and select **Rename**. Change the name and press **Enter**.

or
Right-click and select **Modify** and change the name.

Delete an Index Unused Analysis

To delete an analysis

1. In the Index Usage Analyzer window, select the analysis you want to delete in the left pane.
2. Right-click and select **Delete**.

Index Usage Analyzer Functions

Below is a list of available functions within the Index Usage Analyzer window.

Button or 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

User-Defined Temp Table

About User-Defined Temp Table

The technique of creating temp tables to extract data from permanent tables is often used, because the existence of the temp table is session related and you will need to create them within SQL Optimizer. The User-Defined Temp

Table window allows you to create temp tables to be use throughout SQL Optimizer.

Related Topics
Privileges for Creating User-Defined Temp Table
Creating Temporary Tables
Viewing SQL Scripts of Temporary Tables
Deleting Temporary Tables
Copying SQL with Temporary Tables to SQL Optimizer
Preference Setting for Handling Temporary Tables
User-Defined Temp Table Tutorial

Privileges for Creating User-Defined Temp Tables

To create or modify temporary tables in the User-Defined Temp Table module, the logon user needs the following privileges:

- Connection to DB2 LUW 7 or above
- USE privilege on the USER TEMPORARY table space or SYSADM or DBADM authority.

Related Topics
User-Defined Temp Table Overview
Creating Temporary Tables

Create Temporary Tables

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 use 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, Index Expert, and SQL Scanner. They also can be viewed in the Database Explorer.

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 temporary table on the **Creation** tab. Multiple commands can be entered.
3. Click **Execute**.

After the temp table is created you will notice an icon appearing on the bottom right of the main window status. This icon will disappear when all User-Defined Temp Tables are dropped.

User-Defined Temp Table only supports SQL statements that create or modify temporary . These are:

DECLARE GLOBAL TEMPORARY TABLE

DROP TABLE

INSERT

UPDATE

DELETE

Temp tables created within the User-Defined Temp Table window can be used throughout SQL Optimizer.

Related Topics

User-Defined Temp Table Overview

Privileges for Creating User-Defined Temp Table

User-Defined Temp Table Tutorial

View SQL Scripts of Temporary Tables

After the user-defined temporary tables are created, you can view the SQL scripts used to create and populate the tables.

To review the DDL for creating temporary tables

1. Click .
2. Select the **Temp Table List** tab.
3. Select the desire temporary table from the Temp Table list.

Related Topics

User-Defined Temp Table Overview

Drop Temporary Tables

After the User-Defined temporary tables are created, you can delete them.

To delete temporary tables

1. Click .
2. Select the **Temp Table List** tab.
3. If you want to remove all temp tables, click **Drop All Table**.
4. Otherwise, select the desire temp table from the **Temp Table** list and then click **Drop Table**.

Related Topics

User-Defined Temp Table Overview

Copy SQL with Temporary Tables to SQL Optimizer

When you copy a SQL statement from the Scanned SQL Viewer 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

Preference Settings for Handling Temporary Tables

Under the Options 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 - Options Button

Create Scanner Temp Table
Include data
Override previous Scanner Temp Table

Optimization Tab - Optimization Button

Temp table generation
Apply selected SQL Options to temp table SQL

Related Topics
User-Defined Temp Table Overview

Editor Functions

About Editor Functions

Editor functions are available on all editable windows to help format and construct SQL statements. The layout of the SQL statement is important to make it easier to read and maintain particularly when dealing with large and complicated SQL statements.

Editor functions settings can be changed within the Options window under the Editor tab.

Button or Menu	Function
	Syntax Highlight
	Auto Correction
	Pairing Brackets
	Argument Lookup
	Member Lookup
Edit Menu	
	Indent
Edit Menu	Outdent
Edit Menu	Comment
Edit Menu	Uncomment
Edit Menu	Uppercase

Button or Menu	Function
Edit Menu	Lowercase

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

To setup auto correction

1. Click .
2. Select the **Editor** tab.
3. Under the Editing section, click next to the **Auto Correction** checkbox to open the Auto Correction Settings window.
4. Modify the auto correction entries
5. Click **Apply**.
6. Click **OK**.
7. Check the **Auto Correct** checkbox in order for auto correction function to be enabled.

The 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 keys that will 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 key. By default, the action keys are "**;,:=[]\ n\t\s**" representing the following:

Action Key	Description
;	semi-colon
,	comma
:	colon
=	equal sign
[	left bracket
]	right bracket
\n	new line
\t	tab
\s	space

To add an correction entry

1. Click **Add**.
2. Enter the original text [Replace] and replacement text [With].
3. Click **Apply**.
4. Click **OK**.
5. Click **Apply**.
6. Click **Close**.

To modify an auto correction entry

1. Select the entry to be modified from the Auto Corrections box.
2. Click **Edit**.
3. Modify the text.
4. Click **OK**.
5. Click **Apply**.
6. Click **Close**.

To delete an auto correction entry

1. Select the entry to be modified from the Auto Corrections box.
2. Click **Delete**.
3. Click **Apply**.
4. Click **Close**.

Brackets Pairing

Simple mistakes are often made during SQL construction because of uneven left and right brackets. An easy way to identify the corresponding open and closed brackets for round brackets, curly brackets, or square brackets is to turn on the Bracket Pairing parameter in the Options window.

To enable or disable Bracket Pairing

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

Comment and Uncomment Text

It is possible to comment or uncomment a line of text by selecting the line or highlighting the text and use the Comment and Uncomment functions. These two functions use the single line comment delimiter "- -". Commenting out a line means that the line will be ignored during execution.

To comment a line of text

1. Select the text or place the cursor at the line to be commented.
2. Click .
3. Notice that the beginning of a text or line prefix is marked with the " - - " delimiter.

To uncomment a line of text

1. Select the text or place the cursor at the line to be uncommented.
2. Click .
3. Notice the first occurrence of the " - - " comment delimiter will be removed.

Customize Editable Panes

There are many ways to customize the behavior of editable panes; the editor options can be set in the Options window under the **Editor** tab. These options include:

Editor Option	Description
Use tab characters	Specify whether to use the tab character (ASCII 9) instead of spaces.
Smart tab	Specify whether to tab to align the next non-whitespace character of the preceding line.
Use lowercase for object name	Specify whether to use lowercase for object names.
Block cursor for overwrite	Specify whether to change to a block cursor for overwrite mode. Overwrite mode will overwrite existing text when is text entered at the cursor. You can toggle between Insert and Overwrite mode using the Insert key.
Show all characters	Specify whether to show all characters including spaces, new lines and tabs.
Right margin	Specify whether to show the right margin and define the width.
Gutter	Specify whether to show the gutter and line number, and define the gutter width.
Block indent step size	Specify the block indent character size.
Tab size	Specify the tab character length.

Editor Panes Shortcuts

The following list is the common shortcut keys provided for users to quickly accomplish tasks that is performed frequently within the Editor panes for SQL Editor and SQL Formatter window.

Action	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
Insert a new line character	Shift+Ctrl+M
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

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 step size (default = 3 characters) parameter in the Options

window.

To indent text

1. Highlight the text or place the cursor at the line needed to be indented.
2. Click .

To outdent text

1. Highlight the text or place the cursor at the line to be outdent.
2. Click .

Member and Argument Lookup

Member and argument lookup are lookup hints to help construct SQL statements.

Member lookup displays the list of database object members for schemas, tables, views and alias.; For example, if a table name or alias is typed followed by "." then the corresponding lookup hint will be the column names. **CTRL+SPACE** can be used as short-cut keys 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. **CTRL+SHIFT+SPACE** can be used as short-cut keys to bring up the argument lookup.

To enable or disable the member and argument lookup

1. Click .
2. Select the **Editor** tab.
3. Under the Lookup section, select the Member lookup and/or the Argument Lookup checkboxes .

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 for the lookup hint to appear.

Note: The member lookup selects the information about the database objects from the system view. Since the system view does not contain the information about temporary tables, member lookup does not find the columns in temporary tables.

Syntax Highlight

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

To define the syntax highlighting

1. Click .
2. Select the **Editor** tab.
3. Under the Editing section, click next to the Syntax highlight checkbox.
4. Modify the colors and attributes for any of the elements listed below. Click **OK**.
5. Check the **Syntax highlight** checkbox in order for your selections to apply the syntax highlighting to the code.

Element options

Elements

The font format and color for the following elements can be defined for the following elements in the syntax of a SQL statement:

Syntax Element	Description
Reserved Word	DB2 SQL reserved words. For example: ACCESS, ADD, ALL, ALTER, AND, ANY, AS, ASC , BETWEEN, BY
Function	DB2 build-in functions. For example: ABS, ACOS , ASCII, ASIN , ATAN , ATAN2, CEIL , SIGN
Comment	Comment delimiters (**--, /* */**)
Quoted Identifier	String enclosed in double quotes (**" "**)
String	String enclosed in single quotes (**' '**)
Number	Numbers 0 to 9 and floating point.
Symbol	Symbols. For example: !, $, %, ^, &, *, (,), -, =, +, {, }, [,], :, ;, @
Data type	DB2 data type. For example: CHAR, INTEGER, LONG, NUMBER, RAW, VARCHAR , VARCHAR2
Invalid object	Unrecognized database object under the current database and schema.

Host variable	Host variable names.
Difference highlight	The differences between two SQL for the SQL Comparer window.
SQL Options	Optimizer SQL options.
Text Selection	Selected text.
Member Lookup	Member lookup dialog box.
Argument Lookup	Argument lookup dialog box.
Identifier	Any words that do not fall in the above categories.

Foreground color

Specify the foreground color of the selected element; use the Foreground color list or click button 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 Background color list or click button to select the desired color. Select the **Use default background color** checkbox to use the Windows default background color. As the 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 of the selected element.

To change the color and attribute of a particular element, select the syntax name from the **Element option** list-box or select the text from the Preview section and change the attributes accordingly. Changes will be reflected on the Preview section, click **Apply** to implement the changes.

Uppercase and Lowercase Text

To change text to uppercase

Select the text and click

.

To change text to lowercase

Select the text and click .

SQL Functions

About SQL Functions

SQL Functions are available in evaluating index-set and SQL alternatives. These functions are:

Show Plan

Run Result (SQL alternatives only)

Run for First Record

Run for All Records

Test for Scalability

Convert Parameter Markers

Retrieve Access Plan

The access plan is a combination of steps the optimizer chooses in order to execute a SQL statement. It is the ordered set of steps required to carry out the query with the operators chosen for each table.

To view only the access plan of the original SQL statement

From the SQL Information pane, click .

Note: The Rewrite SQL, Generate Indexes, Run for All Records, Run for First Record, and Run Result functions also retrieve the corresponding access plan.

Retrieve Run Result

Note: This feature is available for only SQL alternatives, not index-set alternatives.

To retrieve the SQL statement results from the database

In the SQL Optimizer window, select the SQL alternative, and click .

If the SQL statement contains a result set, the SQL Result window displays, showing all data retrieved. Otherwise, an information dialog box showing the number of rows affected displays.

In the SQL Result window, the first rows of the result set are displayed as soon as they are returned from the database. Therefore, the time it takes for you to see the result does not equal the time measured by the run- time functions to retrieve either the first record or all records run time. Run Result only retrieves enough records to display in the window; the complete result set is not retrieved until you view the last records.

To terminate the Run Result process

Click .

Commit or Rollback

Two types of action, commit or rollback, can be made after the Run Result has been executed for a UPDATE, INSERT, or DELETE SQL statement depending on which window you are in when you execute the Run Result function.

For the original SQL statement

When you retrieve run results for the original SQL statement (displayed as <*Edit SQL*>), you are prompted to commit or rollback for UPDATE, INSERT, and DELETE SQL statements.

For the SQL alternatives

For the SQL alternatives, all rows affected by UPDATE, INSERT, and DELETE SQL statements are rolled back automatically.

New Database Session

When you log on to DB2 LUW from SQL Optimizer, it connects to the database and maintains this database connection throughout the application execution. Another database session is required for the Run Result processing. This connection is established when the Run Result function is executed and dropped after all the records have been retrieved.

Retrieve Run Time

There are two functions used to retrieve the run time for an SQL statement: **Run for All Records** and **Run for First Record**.

On the SQL tab of SQL Optimizer window, select the SQL or index-set alternative for which you would like the run time.

First Record

To retrieve the time for the first record

Click .

To terminate the run time process for the first record

lick .

All Records

To retrieve the time for the all the records

Click .

To terminate the execution to retrieve the run time for all the records

Click .

All run time information is displayed in the SQL Run Time pane of the SQL Optimizer window.

New Database Session

When you log on to SQL Optimizer, it connects to the database and maintains this database connection throughout the application execution. Another database session is required for run time processing. This connection is established when the **Run for First Record**, **Run for All Records**, or **Batch Run** function is initiated and then dropped after the performance figures are acquired.

Test for Scalability

You can test a single SQL or index-set alternative 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. Click to optimize the original SQL statement. The rewritten SQL versions display as alternatives in the Run Time pane..
2. Click to generate virtual index sets and add them as alternatives.
3. Click **Test for Scalability** .
4. Select the alternative you want to test.

Convert Parameter Markers

Your SQL statement may have several question marks "?" as parameter markers.

To enable unique referencing of parameter markers

Click from the SQL Editor pane.

This will assign a unique number to all parameter markers within the SQL statement, for example:

Original SQL

```
SELECT *
 FROM employee
WHERE emp_id = ?
  OR emp_salary < ?
```

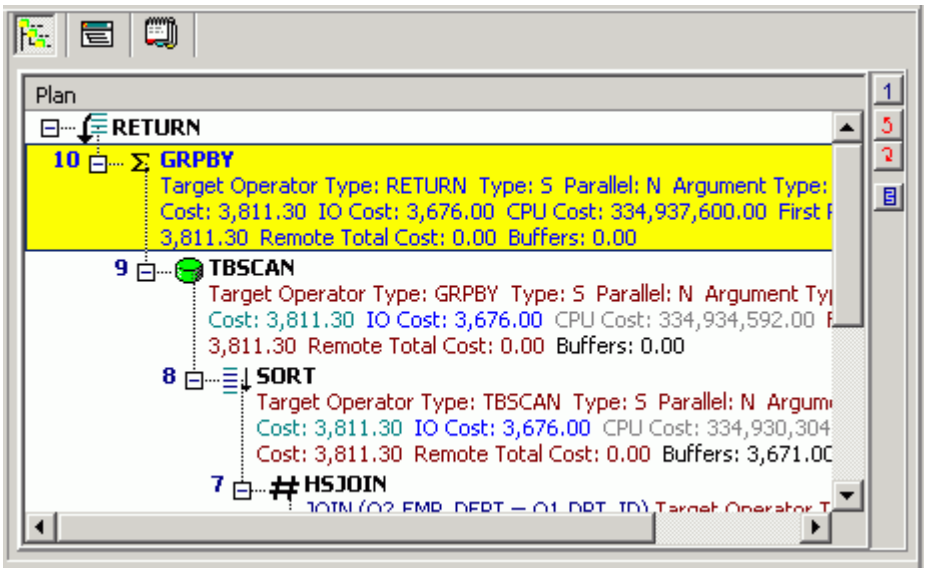
Convert parameter markers

```
SELECT *
 FROM employee
WHERE emp_id = ?1
  OR emp_salary < ?2
```

SQL Information and Functions

SQL Information Pane

View the SQL Information pane



The SQL Information pane shows types of information for the original SQL or the SQL associated with the currently selected alternative. Use the buttons across the top of the pane to toggle between the views.

Note: Different SQL Optimizer modules use the SQL Information pane. The buttons available in the pane depend on the module you are using.

Button	Function	Description
	Virtual Access Plan Tree	Show the corresponding virtual access plan. **Notes:** • No information displays if the SQL statement in invalid. • For index-set alternatives, this is the assumed plan should the indexes actually exist.
	Virtual DB2 Optimized Text	If the SQL statement is valid, show the corresponding virtual DB2 optimized text. **Note:** No information displays if the SQL statement in invalid.
	Access Plan and Bound Access Plan	Display the access plan. **Notes:** • No information displays if the SQL statement in invalid. • For index-set alternatives, this is the actual plan used during Batch Execution (or anther execution function) when temporary indexes were physically created . • In the Scanner SQL Viewer, both the bound plan and the current plan display when the SQL statement is from a package. **Tip:** Click (if available)to display detailed information for each row of the access plan.
	DB2 Optimized Text and Bound DB2 Optimized Text	Display the SQL statement reconstructed by the DB2 LUW optimizer after it retrieves the access plan. **Note:** In the Scanner SQL Viewer, the optimized text for both the bound plan and the current plan display.

Button	Function	Description
	Information	Show any of the following, depending on the specific function using this pane: • SQL statement type classification: Problematic, Complex, Simple, or Invalid SQL. This classification is dependent on the parameters set in the Options window. • (SQL Scanner) For a SQL statement in a package, a comparison of the bound access plan and the current access plan. A database error message is displayed if SQL is classified as *Invalid*. • (SQL Optimizer) Warning or alert information about the SQL statement if the transformation is based on table constraints or indexes. **Note:** Changes to table constraints and indexes might have a direct effect on the optimized SQL statement. • (SQL Optimizer)Origin of the SQL statement. • (SQL Scanner) SQL conversion applied. • (SQL Scanner)Start position of the SQL statement in DDL and in TXT and SQL files. • (SQL Scanner) For bound access plans, package information. • (SQL Scanner) For bound access plans, new access plan information. • (SQL Scanner) Connection information. • Special register settings.
	Scanner Temp Table (SQL Scanner only)	Display 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 code.
	Checked SQL (SQL Scanner only)	Display the date and time when the SQL statement was checked, its status and description, and the name of person who checked the SQL.

Send SQL to the SQL Rewrite Function

Performance improvement can be obtained by running the SQL Rewrite function on an SQL statement. This function transforms the SQL syntax into equivalent statements with different access plans.

Use the following procedure to send the selected SQL statement to the SQL Optimizer window, where the SQL Rewrite function is automatically run on the statement.

To send a SQL statement to the SQL Rewrite function from another module

1. Select the single SQL statement you want to optimize.
2. Click . The selected SQL statement is copied to the SQL Optimizer window, and the rewrite process is automatically started.

When you copy a SQL statement from the Scanned SQL Viewer, the following can occur:

- If the SQL statement uses temporary tables and the DDL for creating the table was found when scanning, you are prompted to create the temporary tables through the User-Defined Temp Table window.
- If the DB2 LUW Special Registers are different from current settings, you are prompted to set the special registers through the Special Register Settings window.
- If your current schema does not match the one previously used to retrieve the access plan of the SQL statement, a message alerts you to change the schema from the box at the left-corner of the main window.

Send SQL to the Generate Indexes Function

Use this procedure to send the SQL statement selected in the current tool to a SQL Optimizer session, where the Generate Indexes function automatically runs on the statement. The Generate Indexes function creates virtual index sets that you can test on the SQL statement to determine whether the SQL's performance improves.

See Generate Index-Set Alternatives for more information about the Generate Indexes function.

To execute Generate Indexes on an SQL statement in another module

1. Select the SQL statement you want to optimize.
2. Click .

   The selected SQL statement is copied to the SQL Optimizer window, where the Generate Indexes function automatically starts.

When copying a SQL statement from the Scanned SQL Viewer,

- If the SQL statement uses temporary tables and the DDL for creating the table was found when scanning, you are prompted to create the temporary tables through the User-Defined Temp Table window.
- If the DB2 LUW Special Registers are different from current settings, then you are prompted to set the special registers through the Special Register Settings window.
- If your current schema does not match the one previously used to retrieve the access plan of the SQL statement, a message alerts you to change the schema from the box at the left-corner of the main window.

Open SQL from SQL Repository

SQL statements saved in the SQL Repository can be copied to the editable pane in other modules.

To copy SQL from the SQL Repository

1. From an editable pane in a module other than the SQL Repository, select **File | Open SQL from SQL Repository**.
2. Navigate through the tree structure and select the SQL statement.
3. Click **Open**.

Save SQL to SQL Repository

To save SQL statements to the SQL Repository module

1. Click .
2. Select the specific SQL to save and click the **Save** button.
3. Select the location in which to save the SQL statements.

Only valid SQL statements can be added to the SQL Repository.

Supported SQL Statements

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

Create Benchmark Factory Import File

All the SQL statements can be saved in a text file from the Job Manager, the Scanned SQL Viewer, the SQL Repository, and the SQL Optimizer windows. These SQL statements can then be imported into Benchmark Factory program (version 4.6 or later).

To create the text 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. Click **OK**.
3. Enter the filename and select the file location. Click **Save**.

SQL Navigation

Several ways are provided to navigate through the SQL statements displayed in multiple tab sets.

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

Button	Menu Item	Description
	Navigate | First SQL	Display the first SQL statement.
	Navigate | Previous SQL	Display the previous SQL statement.
	Navigate | Next SQL	Display the next SQL statement.
	Navigate | Last SQL	Display the last SQL statement.

Find a SQL Using a Text String

The Find SQL function is available in the SQL Optimizer and Scanner SQL Viewer window. It searches through SQL statements until it finds the first occurrence of the text string you specify. In SQL Optimizer, Find SQL searches the SQL syntax currently displayed in the SQL Text pane. In SQL Scanner, the function searches across all SQL statements displayed on the SQL tabs.

To find specific text within the SQL statements

1. Select **SQL | Find SQL**.
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]**.

Comments

Due to the lack of uniform standards in the presentation of comments within source code, it is almost impossible to implement each and every commenting indicator. We have therefore chosen to support the three most popular delimiters:

Comment Syntax	Comment
--	single line
//	single line
starting with **/***	block
ending with ***/**	

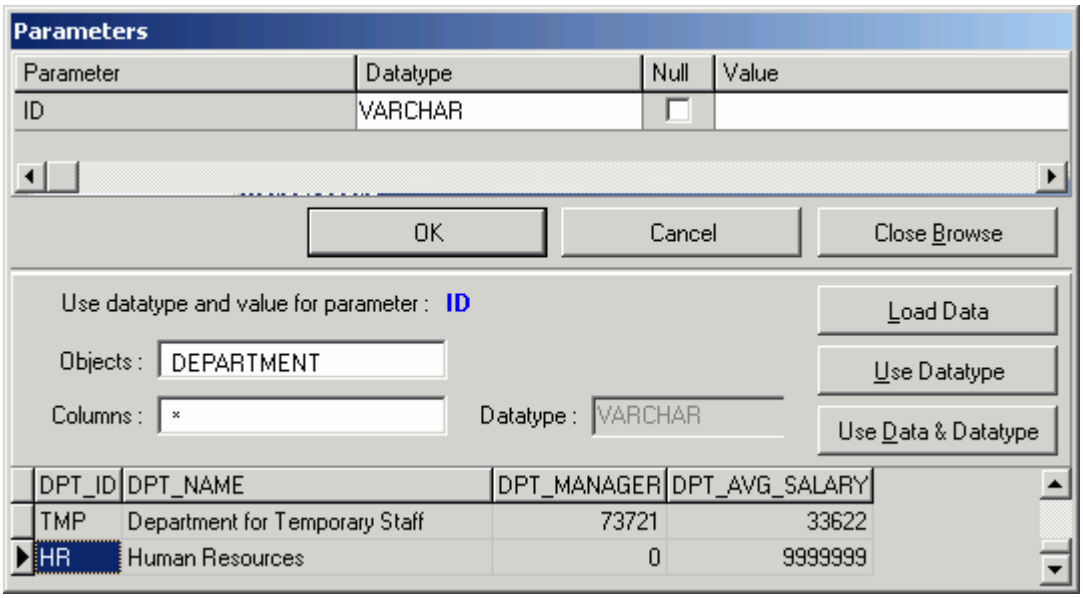
During optimization these comments will be removed and will not be shown in the list of optimized SQL statements. SQL Optimizerdoes not accept any other comment delimiters.

Variables

Variables can be embedded in the original SQL statement without pre-defining the data type and value. All variable names will be highlighted in red after the optimization process. SQL Optimizer will recognize declaration of variables either with or without a ":" (host variables) or "?" (parameter markers) sign. During optimization, run time or run result, the data type and value of the variables will require definition by using the Parameters window.

Parameters Window

View the Parameters Window



Data type and value of variables are entered using the Parameters window. If variables are embedded in the original SQL statement, the Parameters window will be displayed each time you launch the **Run Result**, **Run for First Record**, **Run for All Records**, and **Batch Run** functions. The values entered in the Parameters window have a direct result on the run time and run result retrieved.

The parameter values that you have previously entered can be saved by selecting the Enable SQL parameter history Options 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

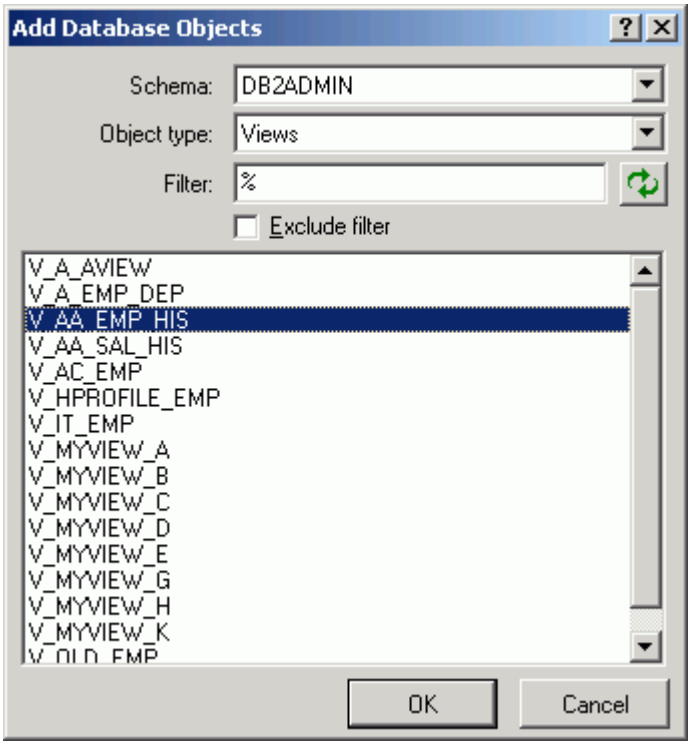
1. Click **Browse** to expand the Parameters window. The bottom pane appears displaying object and data information corresponding to the selected objects from the SQL statement. Select the object to browsed using the **Objects** and **Columns** lists. If a column is selected (asterisk * selects all columns from the selected object), the corresponding data type is shown.
2. Click **Load Data** and highlight the cell with the variable value from the data grid. Then click **Use Data & Data type** or double-click the cell to copy the compatible data type and value to the parameter selected at the top of the window.

3. Click **Use Data type** to copy a compatible data type to the parameter selected at the top of the window.
4. After the data type and value of all the parameters have been selected, click **OK**.

Note: If the Parameters window appears when you do not have variables in the SQL statement, this may be caused by incorrect spelling of column or table names, pointing to the wrong schema, or the tables or columns do not exist in the database. If the tables or columns have been created since you opened SQL Optimizer, you may need to Synchronize the Data Dictionary.

Filter Database Objects

View the Add Database Object 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 filter database objects

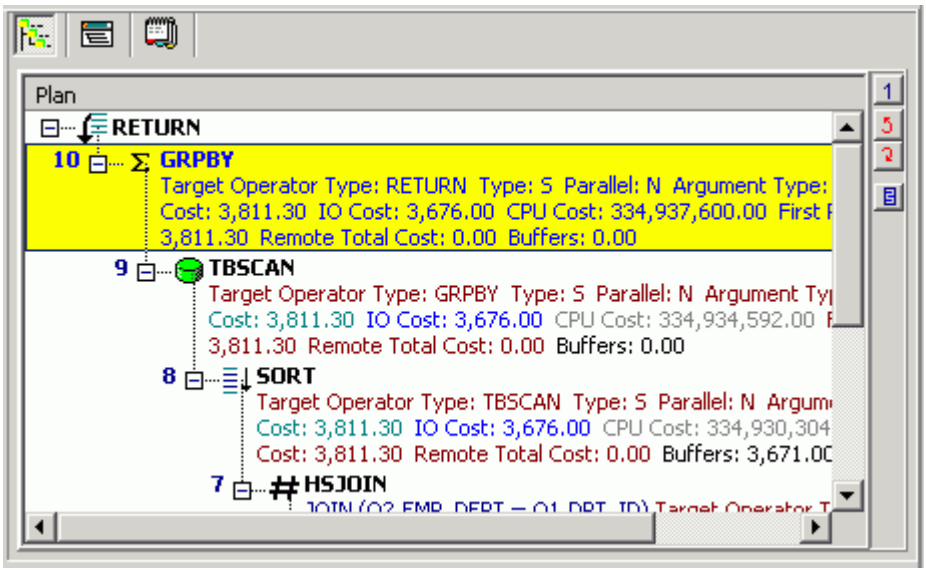
1. Click ... to open the Add Database Objects window.
2. In the **Schema** box, select the schema name where the objects reside.
3. In the **Object type** box, select the database object type.

4. In the **Filter** box, enter the filtering criteria using the % wildcard to replaces multiple characters. The filter is case sensitive, so you must match the upper or lowercase characters of the database object name.
5. If you would like to exclude the database objects that meet the filter criteria instead of including them, check the **Exclude filter** box.
6. Select the database objects from the list.

Access Plan

Access Plan

View the Access Plan pane



The access plan is a combination of steps the DB2 LUW database optimizer chooses to execute a SQL statement. Each node represents how the database optimizer will physically retrieve rows of data from the database or how the data is prepared. By examining the access plan, you can see exactly how the database executes your SQL statement.

Right-click Menu

The Access Plan pane contains a right-click menu that allows you to perform the following functions:

Function	Description
Print	Sends the access plan in its current view to the printer, to display on the screen (print preview), or to a file.
Copy	Copies the access plan to the clipboard.
View Plan	Changes how the access plan is displayed.
Animated Plan Steps	Highlights one-by-one the access plan steps.

Function	Description
Plan Options	Opens the Access Plan Options window so you can select the specific detailed information that is displayed in the access plan. You can also choose to display specific information in individual columns.
Get Help on *plan_step*	Displays the help text for the currently selected step in the access plan.
Help on Access Plan	Opens online help for the access plan.

Related Topics

Access Plan Functions

Review Access Plans

Plan Detail

Plan Help

Animate Access Plans

Copy Access Plans

DB2 Optimized Text

Review Access Plans

The access plan can be displayed in different ways to help you get a clear picture of the steps or detailed information about the steps that DB2 LUW is taking to execute a SQL statement.

For each location where the access plan is displayed, it will remember the display view option (As Tree Plan, As Plain Language Plan, As Graphic Plan, or As MS Graphic Plan) that you have selected from the right-click View Plan option. It will use that option the next time you view the access plan in the same location in the program.

To change how the access plan is displayed

1. Right-click the Access Plan pane 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

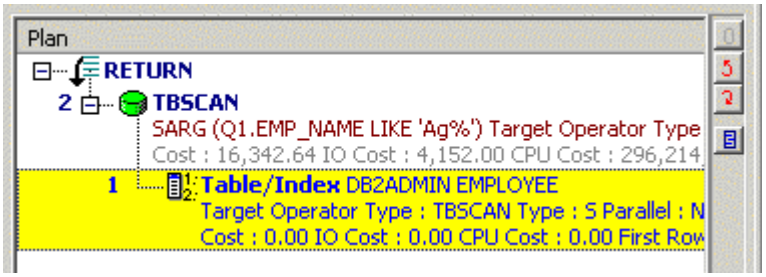


Figure: Access Plan "As Tree Plan"

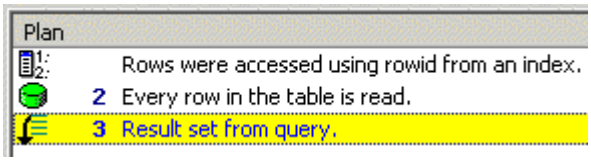


Figure: Access Plan "As Plain Language Plan"

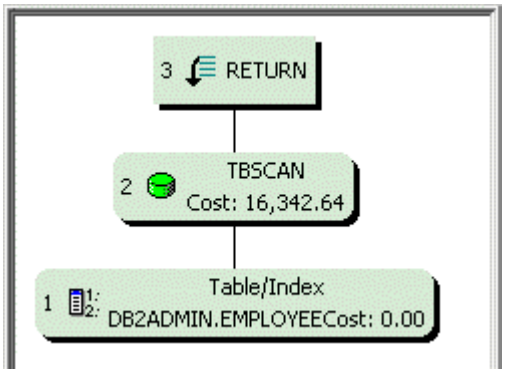


Figure: Access Plan "As Graphic Plan"

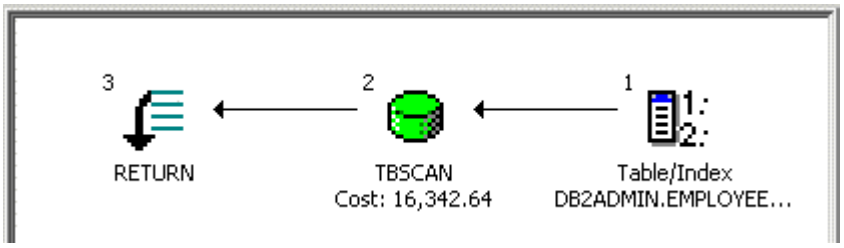


Figure: Access Plan "As MS Graphic Plan"

Access Plan Options

The display of the Access Plan options can be customized for each window in which the access plan is displayed. You can select which elements of the access plan you would like displayed. You can also have the elements displayed in separate columns.

To customize the Access Plan window

1. Right-click the access plan and select **Plan Options**.
2. In the **Visible** column, select the elements you want displayed in the access plan text.
3. In the As Column column , select the elements for which you want a column added to the display in the Access Plan.

When you are viewing the access plan in plain language, you can also display any of the elements in a column.

To include elements when viewing the access plan in plain language

1. Right-click the access plan and select **Plan Options**.
2. In the **As Column** column, select the elements for which you want a column added to
3. the display in the Access Plan.

Note: The color and font settings for the access plan are set on the Plan tab in the Options window.

Access Plan Functions

The following functions are available from the button pane at the right edge of the access plan pane:

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.

Plan Detail

The Plan Detail window provides detailed information on the selected node of the access plan and the relating object details.

To display the Plan Detail

1. Click a node of the access plan.
2. Click located on the right edge of the Access Plan pane.

Object Details

The **Object Details** tab displays statistics about the selected access plan step. You will see the specific values from the selected step in the DETAILS column. Or, if the selected access plan step is performing an action using the table or index, statistics for the table or index are displayed in two columns: EXPLAINED and CURRENT. The EXPLAINED column displays the statistics retrieved at the time the access plan was retrieved. The CURRENT column displays the current statistics that are retrieved when the Plan Detail window was opened.

Optimization Parameters

The **Optimization Parameters** tab displays the DB2 database parameters values that were set in DB2 LUW at the time the access plan was retrieved.

Table/View Detail

To see the table or view details, click the table or view name in the left pane of the Plan Detail window. Then click one of the four tabs at the bottom of the right pane to see specific information about the definition, columns, indexes, or data for the table or view. If you have the appropriate privileges, you can modify the data under the Data tab.

Plan Help

The access plan help provides online help for each operation within the access plan to help you understand each step of the plan.

To view the help for an operation

Right-click the access plan step for which you want help and select **Get help on operation_name**.

Animate Access Plans

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

To animate the plan or cancel the animation

Right-click the Access Plan pane and select **Animate Plan Steps**.

Copy Access Plans

You can copy the access plan to the clipboard.

To copy an access plan

Right-click the Access Plan pane and select **Copy**.

When the access plan is displayed as Tree Plan or Plain Language Plan, the access plan can be pasted as text or as a bitmap picture. Graphic applications, such as MS Paint, only paste the access 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.

DB2 Optimized Text

The DB2 Optimized Text shows the reconstructed SQL provided by DB2 and is retrieved at the same time the access plan is retrieved.

Search Functions

Search Functions

The Search functions search through the current text pane to find the text search string. This applies to text pane that are editable with a white background (Windows default color) and the panes that are read only with a gray background (Windows default color).

These functions are available from the Search menu:

Quick Find

Find

Find Next

Replace

Goto Line

For windows that have multiple tabs in the text pane, use the Find option on the SQL menu to search though all the tabs for the text search string.

Find

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

To search for text the currently displayed SQL statement

1. Select **Search | Find [Ctrl + F]**.
2. Enter the search string.

The search is refined in the Find Text window using the following settings:

Options

Search Option	Description
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

Scope Options	Description
Global	Searches the entire text pane.
Selected text	Searches only the highlighted text.

Direction

Direction Options	Description
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

Origin Options	Description
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.

Quick Find

The Quick Find search function searches the currently displayed SQL statement for the next occurrence of the text that you have highlighted in the SQL statement. If it finds this text, it moves the line containing the search string to the top of the text pane.

To find the currently highlighted text

With text highlighted in the active window, select **Search | Quick Find [CRTL + ALT + F]**.

Find Next

The Quick Find search function finds, in the currently displayed SQL statement, the next occurrence of the text search string that you previously searched for. The function is not available until after you have done an initial search.

To find the next occurrence of the search string

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

Replace

The Replace search function brings up the Replace Text window where you can enter both the text string and text to replace the text string.

To replace the search string text

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

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 move the cursor to the beginning a of specific line

1. Select **Search | Goto Line [Ctrl + Alt + G]**.
2. Enter the line number.

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 whitespace 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	Any place	A colon matches a class of characters described by the character following the colon. :a matches any alphabetic character :d

Type of Search	Search Characters	Location of character in the search string	Explanation
	finds SQLSTATE or SQLTABLE SQL:d Finds SQL1 or SQL3		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 brackets) Example: r[au]n finds run or ran	Any place	Characters enclosed in brackets matches any one character that appears 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 range of characters in one	[-] (hyphen within the	Any place	A hyphen within the brackets signifies a range of characters. Note: the range must be in a progressive order, i.e. "[a-x]"

Type of Search	Search Characters	Location of character in the search string	Explanation
character position	brackets) Example: SQL[1-3] finds SQL1, SQL2 and SQL3		matches any character from a through x while "[x-a]" does not match anything.

Activity Log

About Activity Log

The Activity Log is used to record activities during optimization and access plan generation. Information recorded can be used to review SQL quality and any improvement obtained from optimization.

Related Topics

View Activity Log Report

Start Recording Activities

View Activity Log Report

To print or view information stored in the Activity Log

1. Select **Report | Activity Log**.
2. Select the information to view.

The following information may be displayed according to the selected criteria.

User Information

Item	Description
PC User	The user name from the PC
Logon User	Database logon name
Database Alias	Database alias name
Schema	Schema name

Activity Information

Item	Description
Date	Date and time when the activity was executed.
Activity	The type of activity recorded: OP – SQL Optimization EP – Access Plan Generation
Original SQL Type	The classified type of the original SQL statement. For example: Simple, Complex or Problematic. The SQL type is dependent on the Options settings.
Status (All Records)	Current status of the activity relating to run time for all records.
Original SQL Elapsed Time (All Records)	Run time for retrieving all the records of the original SQL statement.
Best Alternative Elapsed Time (All Records)	Run time for the SQL alternative that has the best alternative run time for all records.
Times of Improvement (All Records)	The number of times faster the best alternative SQL is compared with the original SQL statement when comparing the run times for all records.
Status (First Record)	Current status of the activity relating to run time for the first record.
Original SQL Elapsed Time (First Record)	Run time for retrieving the first record of the original SQL statement.
Best Alternative Elapsed Time (First Record)	Run time for the first record from the SQL alternative that has the best alternative run time for the all records.
Times of Improvement (First Record)	The number of times faster the best alternative SQL is compared with the original SQL statement when comparing the run times for the first record.

Note: The Activity Log information will be retrieved from the Activity Log directory. Therefore if any changes have been made to this directory, the information may not be retrieved.

Purge Activity Log Data

To purge the Activity Log data

1. Click .
2. Select **Activity Log** tab.
3. Under the **Housekeeping** section, select **Whole Log** to remove all information

or

Specify a date range to remove logs between these dates.

4. Click **Purge Now**.

Start Recording Activities

By default, the Activity Log is not created.

To start recording the activities during optimization and access plan retrieval

1. Click .
2. Select the **Activity Log** tab.
3. In **Activity to be logged** section, select **SQL optimization** and/or **Access plan generation**.
   Note: If no option is selected, the Activity Log is not created.
4. In the **Information to be logged** section, select **SQL text** or/and **Access plan**.
   The following information will be recorded automatically for each activity logged:

Item	Description
PC User	The user name from the PC
Logon User	Database logon name
Database Alias	Database alias name
Schema	Schema name

5. Select the **Directory Setup** tab to change the directory where the Activity Log is stored. The default is the installation directory.

C:\Documents and Settings*username*\Application Data\Quest Software\Quest SQL Optimizer for IBM® DB2® LUW.

Stop Recording Activities

To stop recording activities in the Activity Log

1. Click .
2. Select the **Activity Log** tab.
3. In the Activity to be logged section, clear both the **SQL optimization** and **Access plan** generation checkboxes to disable the Activity Log.

Tutorials

SQL Scanner Tutorial

Use the SQL Scanner to analyze SQL statements embedded within database objects, text/binary files and application source code. The SQL Scanner extracts each SQL statement embedded within the scanned objects or files, retrieves their respective access plans from DB2 LUW, and performs an analysis that determines which of these SQL statements may be performance bottlenecks. You can examine the extracted SQL statements with their access plans and then, copy the SQL statements identified as problematic (top priority) or complex (second priority) into the SQL Optimizer or the Index Expert.

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

Add Scanner Jobs

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

   Note: If you are using an existing Group, click .
6. In the Add Jobs wizard, click the **Next** button until you are at the page for the item that you want to scan.

Database Objects page

a. Expand the database user branch on the left side of the window.

b. Highlight the schema, a database object type, or an individual database object, and click to move the item to the right pane. (Whether or not you can scan all of the selected database objects depends on your database privileges.)

Source Code page

a. Click the **Text or binary files or COBOL programming source code** option.

b. Click and select the files you want to scan.

c. Click **Open** to insert the files in the Add Jobs wizard.

d. Set the schema in the **Schema** list to correspond with the SQL that you are scanning.

Summary page

a. Review the jobs you selected.

b. Click **Finish**.

Scan Jobs

7. Click **Scan** .
8. Review the details that are filled in the Job Manager grid 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

9. To view the scanned SQL statements, highlight the item by clicking the row and click .
10. The name of the source file or database object appears at the top of the window in the Job list.
11. The first SQL statement found is shown in the left pane. Click the tabs, e.g. SQL1, SQL2, SQL3, etc., at the bottom left of the window to view the other SQL statements.
12. Notice the buttons on the top of the right pane. These buttons display in the right pane the access plan, the SQL classification and connection information, the DDL for temporary tables used by the SQL, and details about SQL statements that you have reviewed.
13. You can narrow the number of 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 have index option generated for the SQL. Click to copy the SQL statement to the Index Expert window and generate index options.

Related Topics

SQL Scanner Overview

Job Manager Overview

Job Manager Window

Job Manager Functions

Scanned SQL Viewer Overview

Scanned SQL Viewer Window

Scanned SQL Viewer Functions

SQL Optimizer Tutorial

SQL optimization is a four-phase process:

- The SQL Rewrite step creates virtual alternative SQL statements, retrieves their respective access plans from DB2 LUW, and indicates each statement's DB2 LUW cost. These alternatives produce the same results as your original SQL statement, but each has a different access plan.
- The Generate Indexes step creates virtual index sets and adds these as alternatives.
- The next phase test-runs your original SQL statement and any or all of the alternatives against your database to obtain run times.
- The last phase is determining the SQL or index-set alternative that performs best.

Tip: The DB2 LUW Cost is only an estimate of the resources it takes to execute a SQL statement. It is essential to run Batch Execute on the alternatives to determine which statement actually performs the best.

Rewrite the SQL statement

1. Click to open a SQL Optimizer session.
2. On the SQL tab, enter the SQL statement you want to optimize.

3. Click . This step launches the SQL Rewrite process that automatically transforms the syntax of the SQL statement.

   Notes:

   - The use of SQL Options and other optimization options such as temp table generation, ANSI JOIN syntax are optional and configurable in the Options.
   - The degree of the SQL transformation process is controlled by the Intelligence Level in the Options. The Intelligence Levels control how many options are applied to transformed SQL and how many SQL alternatives are created.
   - If your SQL statement uses a temporary table, see the section User-Defined Temp Table for the steps to create a temporary table in the User-Defined Temp Table module.

   After the rewrite, the Rewrite Details dialog shows the total number of semantically equivalent SQL statements, the number of alternative statements with access plans different from your original statement, and a warning message if the number of SQL transformations reaches any of the optimization quotas set in the Options.

4. Click **Close**.

   The SQL Optimizer window shows several tabs that provide information about the original SQL statement and its SQL alternatives. The SQL tab displays the SQL text and the access plan of the currently selected SQL alternative (or the original SQL statement). At the bottom of the tab is the Run Time pane. This pane lists the original SQL statement, the SQL alternatives, and the run-time statistics for all of these after they are executed. At this point, since you have not yet run Batch Execute, the pane shows displays only DB2 LUW cost value for the original SQL statement and each SQL alternative.

5. In the Run Time pane, select an SQL alternative.

6. Go to the Access Plan tab to view the alternative's access plan and statistics for the objects accessed by the SQL statement.

7. To see how the syntax of an alternative SQL statement differs from that of your original SQL, do the following:

   a. Go to the Compare tab.

      Your original SQL statement is displayed in one pane of the page and the alternative statement in the other pane. Blue highlighted shows differences in the SQL syntax.

   b. Click , , or to customize what is displayed on the page.

8. Go to the Plan Statistics tab to compare the cost estimates between the original SQL access plan and the plan for each alternative.

9. Go back to the SQL tab.

Create index-set candidates

10. In the SQL Optimizer window, click to generate virtual index sets. These index sets include those that the SQL Optimizer's Index Expert component recommends and those that DB2 recommends.

    Note: If you are connected toDB2 LUW 8 or later, Index Expert generates its own virtual index sets and optionally includes indexes that DB2 recommends. If you are using DB2 LUW 7, only DB2-recommended index sets are retrieved.

    The resulting virtual index-sets display as alternatives in the in the Run Time pane. Index-set alternatives recommended by Index Expert are labeled *Set*x; those recommended by DB2 are labeled *DB2 LUW*.

Batch test SQL alternatives

11. To prepare to execute the original SQL, SQL alternatives, and index-set alternatives, click .

12. In the Batch Run Criteria window, select the Selected SQL/Index Set tab.

    a. Select which alternatives to execute. The blue checkmark in the left column indicates that the alternative is selected. By default, all alternatives are selected.

    b. To unselect a statement, right-click the alternative and select the appropriate option.

13. Select the defaults on the remaining tabs in the Batch Criteria window, and click **OK**.

    Note: If you need to make edits to any criteria, see Batch Run for more information.

    The Batch Run window opens enabling you to view the results as each statement executes.

14. When all the selected SQL statements have finished executing, the Batch Run Details window appears. Click **OK**.

Review test results

17. In the Run Time pane on the SQL tab, review the columns that contain various types of execution time.

18. Once you have identified the most-efficient alternative SQL statement, you can any of the following:

    - Copy and paste it back in your application.
    - Save the alternative SQL statement in a text file either individually or with multiple SQL statements in the Optimized SQL report.
    - Save your SQL optimization results for later review. Select **SQL | Saved Optimized SQL**.

Related Topics

SQL Optimizer Window

SQL Optimizer Functions

Automatically Rewrite the Original SQL Statement

Generate Index-Set Alternatives

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.

Formatting a SQL Statement

1. Click .
2. After entering a SQL statement in the left pane of the window, click .
3. The formatted SQL statement displays in the right pane of the window. 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 DB2 LUW is displayed.

Related Topics

SQL Formatter Overview

SQL Formatter Window

SQL Formatter Functions

SQL Inspector Tutorial

To open the SQL Inspector window

Click .

To add an Inspector

Click .

After adding an Inspector in the SQL Inspector window, SQL statements and statistics from the monitoring tables are retrieved by executing the **Inspect** function. The Inspector must first be selected before the inspection can begin. Only one Inspector can be marked at a time.

To start the inspecting process

1. Select an Inspector job.
2. 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.

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 create are dropped.

Working with Temporary Tables

1. Click **User-Defined Temp Table** .
2. On the **Creation tab**, enter the statements for creating the temp table. You may include DECLARE GLOBAL TEMPORARY TABLE, INSERT, UPDATE, and DELETE statements.
3. Click **Execute**.

Related Topics

User-Defined Temp Table Overview
Privileges for Creating User-Defined Temp Table
Creating Temporary Tables
Viewing SQL Scripts of Temporary Tables
Deleting Temporary Tables
Copying SQL with Temporary Tables to SQL Optimizer
Preference Setting for Handling Temporary Tables

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.

Adding SQL to the SQL Repository

1. Click .
2. If no SQL exists in the SQL Repository, then the Add SQL window appears automatically. Otherwise, you can open the Add SQL window by clicking button.
3. In the Add SQL window enter the SQL text in the **SQL Information** tab. Click **OK**.
4. The SQL syntax is checked and the access 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 an access plan, SQL classification type (Simple, Complex or Problematic), and the current connection information (login name, database alias, and schema). The access plan stored with the SQL statement is important as it indicates the current performance of the SQL.

Related Topics

SQL Repository Overview

SQL Repository Window

SQL Repository Functions

Saving SQL to the SQL Repository from Other Modules Tutorial

Save SQL to the SQL Repository from Other Modules Tutorial

You can save SQL statements to the SQL Repository from other modules such as the SQL Scanner and SQL Optimizer.

1. Click **Save SQL to SQL Repository** .
2. Select the location in which to save the SQL statements
3. Click **OK**.

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

Related Topics

SQL Repository Tutorial

SQL Repository Overview

Save SQL to SQL Repository

Generate Virtual Indexes Tutorial

The Generate Indexes function analyzes the syntax of your original SQL statement and the database structure and then proposes new index candidates to help improve performance. The SQL statement can be executed using the index recommendations to identify which index yields the greatest performance gain.

Create index-set candidates

1. Click to open a SQL Optimizer session.
2. On the SQL tab, enter the SQL statement for which you want to analyze for index alternatives.

   Note: To copy a SQL statement from other windows such as Scanned SQL Viewer, SQL Formatter, Database Explorer, or SQL Comparer, click .
3. Click to view the current access plan for the SQL.
4. Click to generate alternative index sets.

   All index-set alternatives that Index Expert generates are listed in the in the Run Time pane. Index sets identified by the Index Expert Artificial Intelligence engine are labeled *Setx*; those recommended by DB2 are labeled *DB2 LUW*.

Tips:

- In the Run Time pane, index-set alternatives are listed along with any SQL alternatives (which you generate by clicking). In this way, you can easily compare the performance of the index-sets with each other and the SQL alternatives.
- For the currently selected index-set alternative, view the its virtual DDL in the SQL Text pane. Use the SQL Information pane to view the index's virtual access plan and the optimized SQL that uses this plan.
- Create you own virtual index sets by clicking .

Test index sets

5. To obtain actual run-time statistics when the SQL statement when it uses each index-set scenario, click , select your benchmarking options, and click **OK**.

   The Batch Run window opens, enabling you to view the run-time results as each scenario executes.

   Important Note: This benchmark process may impact the performance of SQL statements executing on your database server.
6. When Batch Run is finished, use the Run Time pane view the results for each index-set alternative.
7. To analyze the impact of the index-set alternatives on the access plans of other SQL statements, click to run Impact Analysis.

Tip: To analyze the impact of virtual index sets on other SQL in the database, click . See Analyze the Impact of New Indexes for more information.

Related Topics

Generate Index-Set Alternatives

Index Impact Analyzer Tutorial

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

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

Analyzer Tab

3. Under the **Analyzer** tab in the New Analysis window, select to check the effects of index creation by either **Creating a new Analyzer or** Continuing an existing Analyzer.
4. Enter a name and description.
5. If you would like to create a folder, 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 Access Plans will be analyzed** section, select the options for retrieving the access plans.

- **Using existing access plan saved with the SQL**: This option uses the access 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 access plan under the current connection**: This option retrieves the access plan with the current database logon. This current access plan is compared to the access plan that is retrieved after creating the new index(es).

Index Tab

9. Name the Index Scenario and enter an optional description for easy reference.
10. If the Index Impact Analyzer was called from the Index Expert, the index scenarios are automatically display and you can choose the index scenarios to include for analysis.
11. To create the indexes, click .
    - Give the index a name for easy reference.
    - In the 'Index based on' section of this window, specify in which schema to create this index and specify the table that contains the column(s) to include in the index.
    - The columns of the table appear in the 'Table columns' window and can be selected to use in the index scenario, under the 'Index columns' window, by double clicking them or using the right arrow button.
    - Specify the Index type by selecting the option to have the index **Unique** and whether to **Allow Reverse Scans**.
12. Click **OK** to perform the Index Impact Analysis.

Reviewing Index Impact Analysis Results

13. 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.
14. In the right pane for the scenarios, click the **Prognosis** button to see overall performance changes and SQL details.

Related Topics

Index Impact Analyzer Overview

Index Impact Analyzer Window

Index Impact Analyzer Functions

Index Usage Analyzer Tutorial

The Index Usage Analyzer identifies the usage of the indexes in your database application by uncovering both the indexes that are being used and the indexes that are not being unused. It analyzes the access plans from SQL statements in the SQL statements that are save in the SQL Repository. 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 window automatically opens.

Analyzer Tab

3. Give the analysis a name and description for easy reference.
4. If you would like to create a folder, click .

Select SQL Tab

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

Reviewing Index Usage 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 the **Index Summary** button to see a list of indexes used and unused.

Related Topics

Index Usage Analyzer Overview

Index Usage Analyzer Window

Index Usage Analyzer Functions

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.

Browsing 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 Topics

Database Explorer Overview

Database Explorer Window

Database Explorer Functions

About us

We are more than just a name

We are on a quest to make your information technology work harder for you. That is why we build community-driven software solutions that help you spend less time on IT administration and more time on business innovation. We help you modernize your data center, get you to the cloud quicker and provide the expertise, security and accessibility you need to grow your data-driven business. Combined with Quest's invitation to the global community to be a part of its innovation, and our firm commitment to ensuring customer satisfaction, we continue to deliver solutions that have a real impact on our customers today and leave a legacy we are proud of. We are challenging the status quo by transforming into a new software company. And as your partner, we work tirelessly to make sure your information technology is designed for you and by you. This is our mission, and we are in this together. Welcome to a new Quest. You are invited to Join the Innovation.

Our brand, our vision. Together.

Our logo reflects our story: innovation, community and support. An important part of this story begins with the letter Q. It is a perfect circle, representing our commitment to technological precision and strength. The space in the Q itself symbolizes our need to add the missing piece — you — to the community, to the new Quest.

Contacting Quest

For sales or other inquiries, visit https://www.quest.com/company/contact-us.aspx.

Technical support resources

Technical support is available to Quest customers with a valid maintenance contract and customers who have trial versions. You can access the Quest Support Portal at https://support.quest.com.

The Support Portal provides self-help tools you can use to solve problems quickly and independently, 24 hours a day, 365 days a year. The Support Portal enables you to:

- Submit and manage a Service Request
- View Knowledge Base articles
- Sign up for product notifications
- Download software and technical documentation
- View how-to-videos
- Engage in community discussions
- Chat with support engineers online
- View services to assist you with your product