DB2® Fundamentals and Data Studio

Hands-On Lab

Table of Contents

1 Introduction

DB2 for Linux, UNIX and Windows is a relational database management system developed by IBM®. Since its inception, DB2 has received constant enhancements in areas such as compression, autonomics, security, XML data handling, manageability, and performance.

With the advent of IBM Data Studio comes a major advance in the way DB2 developers and administrators alike carry out their day to day functions. Historically, depending on the tasks to be completed, it was common to switch back and forth between disparate tools such as Control Center, Health Monitor, Developer Workbench, and even the DB2 Command Line Processor (CLP).

IBM Data Studio version 4.1.0.0 changes all this, facilitating DB2 administration, design, development, and monitoring all within an integrated Eclipsed-based environment. By leveraging the power of Data Studio, users are certain to enjoy increased productivity as they find themselves able to perform a majority of their tasks within a single environment.

In this lab, you will explore the basic concepts of DB2 with Data Studio 4.1.0.0 from the perspective of both a developer and a database administrator.

2 Suggested Reading

IBM Data Studio version 3.1 Information Center: http://publib.boulder.ibm.com/infocenter/dstudio/v3r1/index.jsp

3 About this Lab

By the end of this lab, you will be able to:

- Familiarize with the layout and navigation of Data Studio
- Establish a database connection
- Browse database objects
- Leverage task assistants to carry out administrative tasks
- Manage database changes with change plans
- Create and execute SQL statements using Data Studio's SQL Editor
- Analyze data access plans using Visual Explain
- Export query results for later use

4 Environment Setup Requirements

To complete this lab you will need the following:

1. DB2 10 Bootcamp VMware® image
2. VMware Workstation 6.5 or later

5 Initial Steps

1. Start the VMware image by clicking the button in VMware Workstation.
2. At the login prompt, log in with the following credentials :
   - Username : **db2inst1**
   - Password : **password**
3. Open a terminal window by right-clicking on the Desktop and choosing the Open Terminal item.

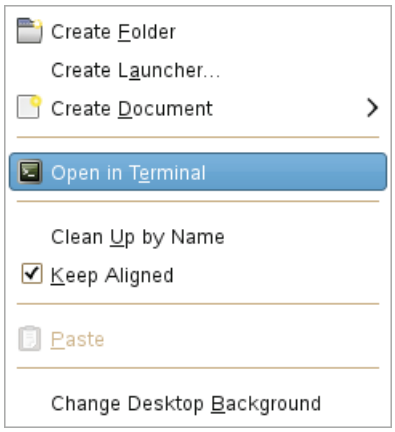


Figure 1 – Open in Terminal on Desktop

4. Ensure that the DB2 Database Manager has been started by issuing the following

```
db2start
```

A message like the following should be displayed to if the DB2 Database Manager had not been started yet:

```
db2inst1@db2awse:~/Desktop> db2start
07/11/2013 11:36:11     0   0   SQL1063N  DB2START processing was successful.
SQL1063N  DB2START processing was successful.
```

5.1 The SAMPLE Database

The SAMPLE database can be used for various purposes like testing your own applications, or trying different features of DB2 using the provided sample programs. During this lab, we will use the SAMPLE database for demonstrating various features of DB2 that makes it easy to understand the technology.

Once the sample database is created, you will notice:

- Organizational schema for non-XML data and
- Purchase order schema for XML data is created.

The data and database objects are created using a real time environment on a small scale.

1. Use the DB2 command **db2sampl** to create a sample database that we will be using in the subsequent sections. Enter the following command in the terminal window:

```
db2sampl -sql -xml -name DB2FUND
```

Note: -sql: creates tables, triggers, functions, procedures, and populates the tables with data.
-xml: creates tables with columns of data type XML, creates indexes on the XML columns, registers XML schemas, and populates these tables with data including XML document values.
-name: defines the database name.

The following message should be displayed, indicating that the DB2FUND database has been created:

```
db2inst1@db2awse:~/Desktop> db2sampl -sql -xml -name DB2FUND

  Creating database "DB2FUND"...
  Connecting to database "DB2FUND"...
  Creating tables and data in schema "DB2INST1"...
  Creating tables with XML columns and XML data in schema "DB2INST1"...

  'db2sampl' processing complete.
```

6 Launch IBM Data Studio

1. Double click ‘IBM Data Studio’ on the Desktop.

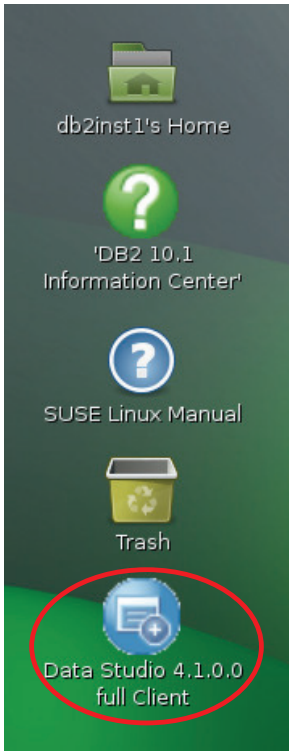


Figure 2 – IBM Data Studio on Desktop

2. If this is the first time Data Studio is launched, it may take a moment. Click OK to accept the default workspace.

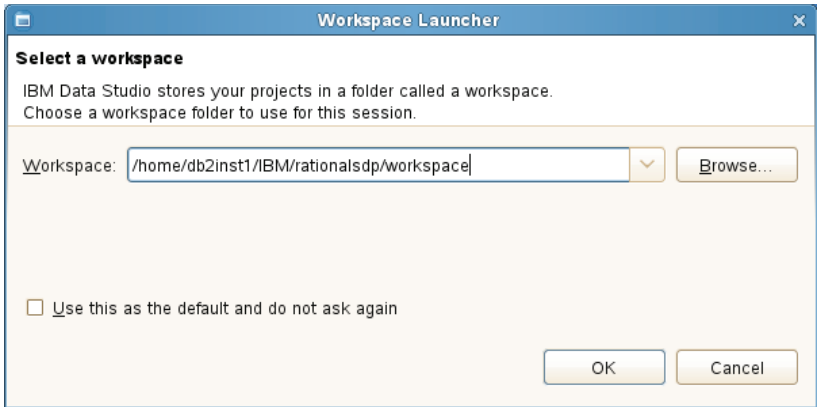


Figure 3 – Select a workspace

3. By default, the **Database Administration** perspective is opened. Your workspace should look like the following :

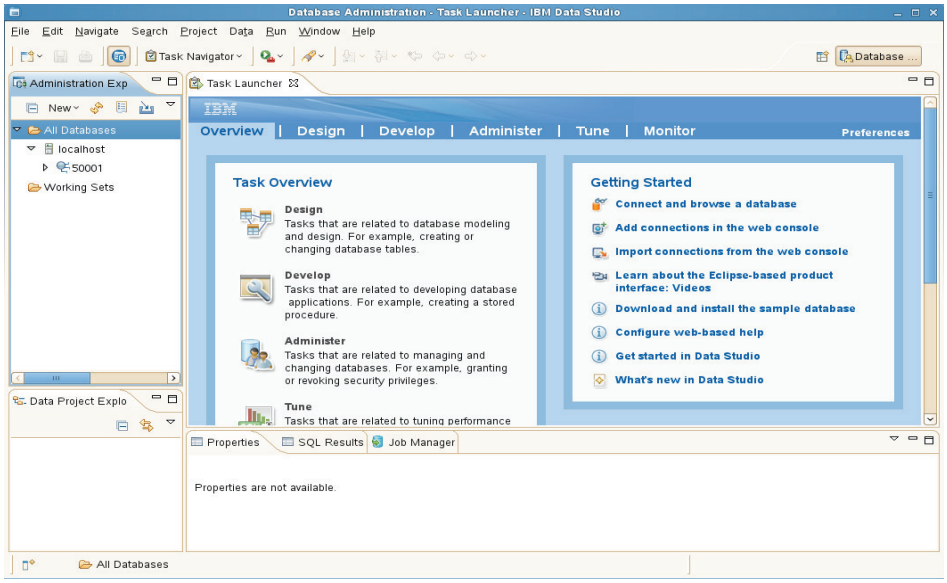


Figure 4 – IBM Data Studio workspace

7 Data Studio General Layout

Since Data Studio is built on the Eclipse framework, it has a similar structure that consists of perspectives and views. As mentioned earlier, the default perspective is **Database Administration**. In this perspective, three sections are central to Data Studio:

- **Administration Explorer**
- **Object List**
- **Properties** / **Results** view

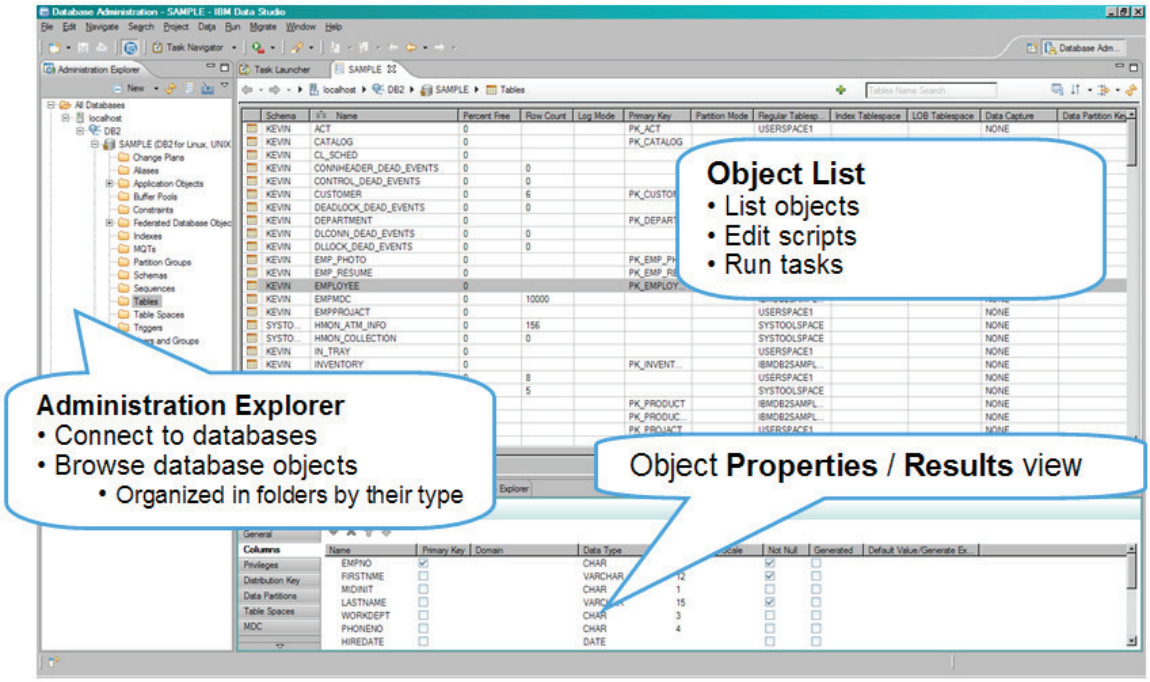


Figure 5 – Data Studio layout overview

Note: **Perspective** defines the initial set and layout of the components in your screen. Each perspective provides a set of functionality aimed at accomplishing a specific type of task or works with specific types of resources.
For instance, the "Database Administration" perspective allows you carry on administrative tasks, while the "Query Tuning" perspective provides the tools for you to optimize SQL queries.

Views are screen components that present information and allows you to navigate through the system. Eg: "Administration Explorer"and "SQL Results" views.

Note: in case if you are not in Database Administration perspective, change it using  **at the upper right corner of your Data Studio workspace.**

If you are familiar with Control Center, which has been deprecated since DB2 9.7, the following table shows the mapping of the main display areas between Control Center and Data Studio.

Control Center	Data Studio	Purpose
Object Tree	Administration Explorer view	Connect to databases and navigate through different database object types, including DB2 pureScale members and CFs
Content pane	Object List area	Lists database objects. In Data Studio, several tabs can be open for different purposes.
Content Details pane	Properties view	After selecting an object, its properties can be seen in this area. Data Studio can have multiple tabs in this area for different purposes.

8 Working with Databases in Data Studio

8.1 Connecting to a database

Before you can do anything productive, a connection must be established to a database. The **Administration Explorer** view is central in Data Studio, where users can perform many administrative tasks including connecting to a database. If DB2 databases have already been catalogued on the machine where Data Studio is installed, the database connection profiles would automatically be created by default when Data Studio is launched. Since you have created the DB2FUND database in the "Initial Steps" section, you should be able to see it in Administration Explorer.

1. In Administration Explorer, expand the folder All Databases until you see DB2FUND.

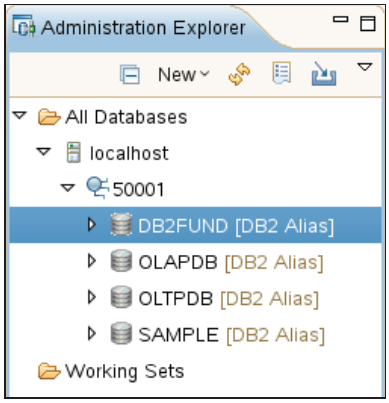


Figure 6 – Data connection profiles

2. Still in Administration Explorer, right click DB2FUND and select **Connect**.

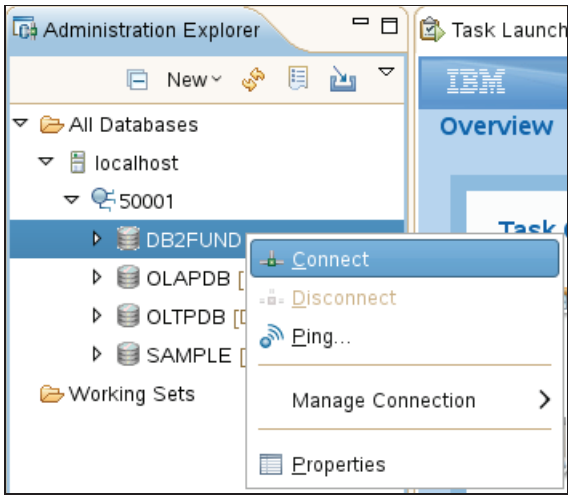


Figure 7 – Connect to a database

Note: To connect to a remote database, users can click New on the toolbar in the Administration Explorer view, select New Connection to a database, fill in required fields, and connect. The new database connection will be available in the Administration Explorer view.

3. In the pop-up dialog, fill in the connection details as follows :
    - Database : **DB2FUND**
    - Host : **localhost**
    - Port number : **50001**
    - User name : **db2inst1**
    - Password : **password**

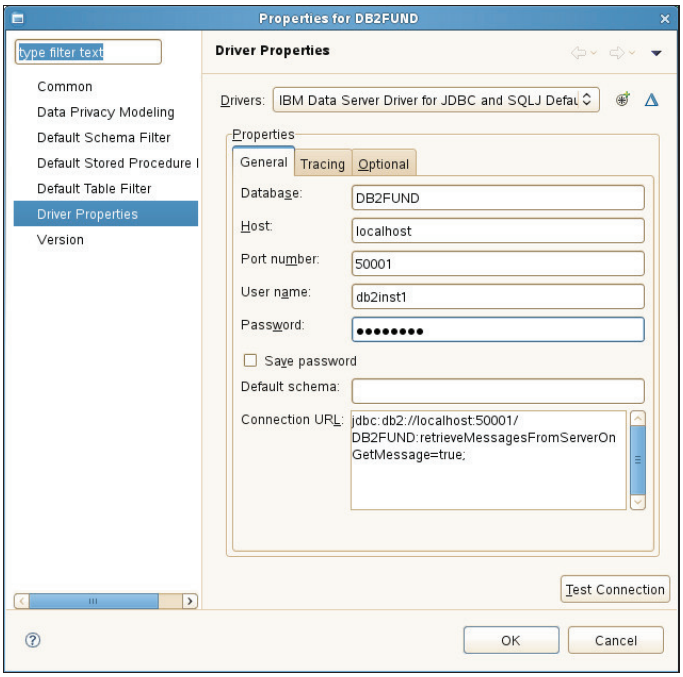


Figure 8 – Database Connection dialog

4. Click OK.
5. After a connection has been successfully established, expand the **db2inst1** instance folder in Administration Explorer, you should see this icon beside the **DB2FUND** database, which indicates a connected database. Disconnected databases are listed with . Expanding the DB2FUND database folder will list supported database object types, as seen in the following screen capture :

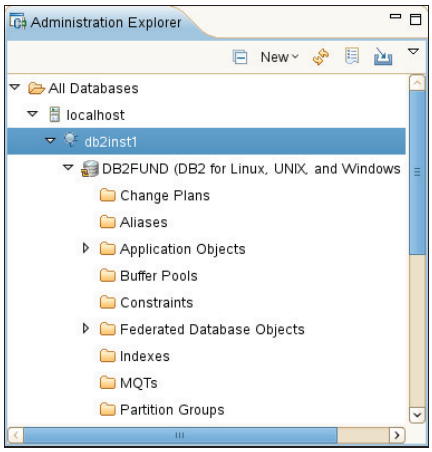


Figure 9 – Connected database with supported database object types

8.2 Browsing Database Objects

Now that you have connected to database DB2FUND, let's see how you can browse database objects in Data Studio.

In the Administration Explorer, database objects are grouped by type, and they are displayed as folders. Clicking any folder will have its content automatically displayed in the Object List area. We'll use tables as an example.

1. Expand DB2FUND in Administration Explorer, click the Tables folder. The list of content is automatically displayed in the Object List area on the right.

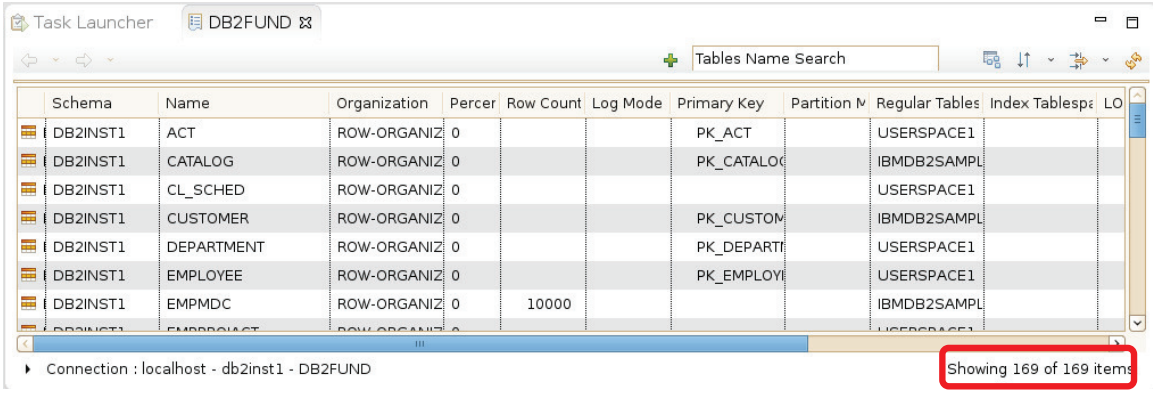
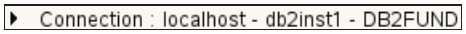


Figure 10 – List of tables in DB2FUND

Notice that the current path  and the total number of objects are shown at bottom of the view.

2. Click on the **EMPLOYEE** table in the Object List, its properties are automatically displayed in the **Properties** view at the bottom of your workspace, and they are read-only. The information displayed is dependant on the type of the database object you select. For tables, you can view properties such as columns definition, privileges, table spaces used, etc.

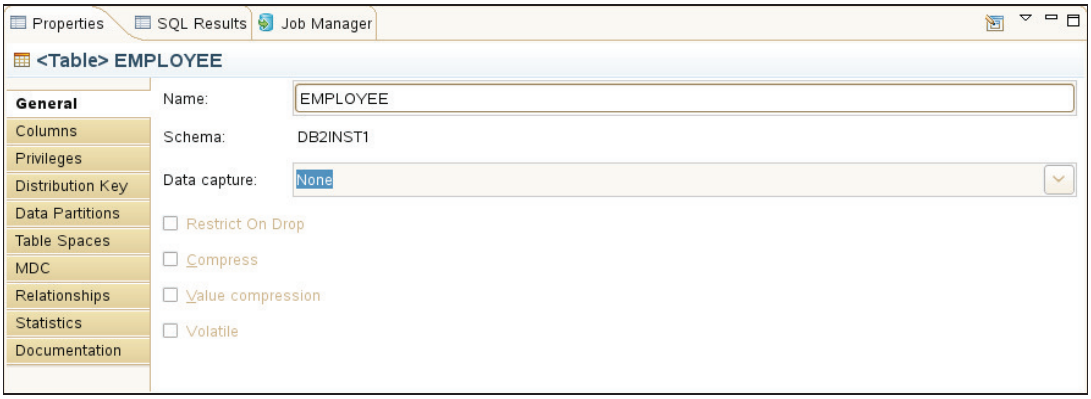


Figure 11 – Properties view

At the top of the Object List area, you can see a multi-purpose navigation bar. It consists of the following sections:

- Web-browser-like back / forward navigation.
- Breadcrumb navigation, which facilitates moving through the hierarchy of objects.
- "Create new object" button  .
- In-place search function. Simply type the object name or part of it, and the list would be dynamically updated. Wildcard characters '%' and '_' (underscore) are supported.
- Options to sort, filter, refresh objects list.

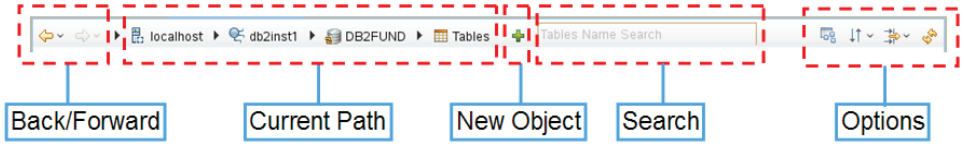


Figure 12 – Object List navigation bar

3. Clicking any of the arrows on the current path would give you a list of all objects at that hierarchical level. For example, click the arrow next to  gives you access to all objects at the DB2FUND database level.

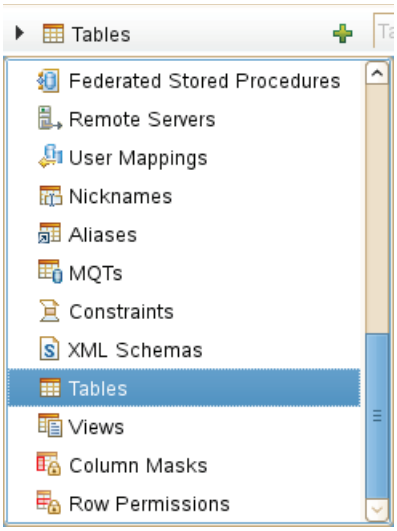


Figure 13 – List of objects in database DB2FUND using the navigation bar

The in-place search feature allows users easily find a particular object they look for.

4. Using the navigation bar, move to the "**Table Name Search**" field (since we were looking at tables in DB2FUND) and type 'E'. All tables with a name that starts with E are returned.

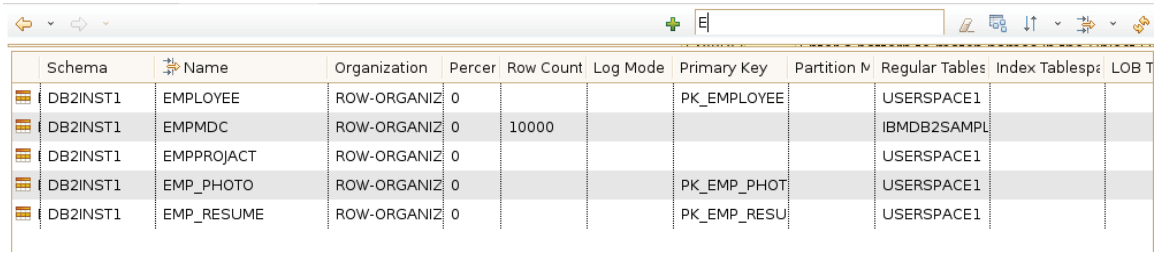
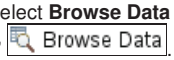
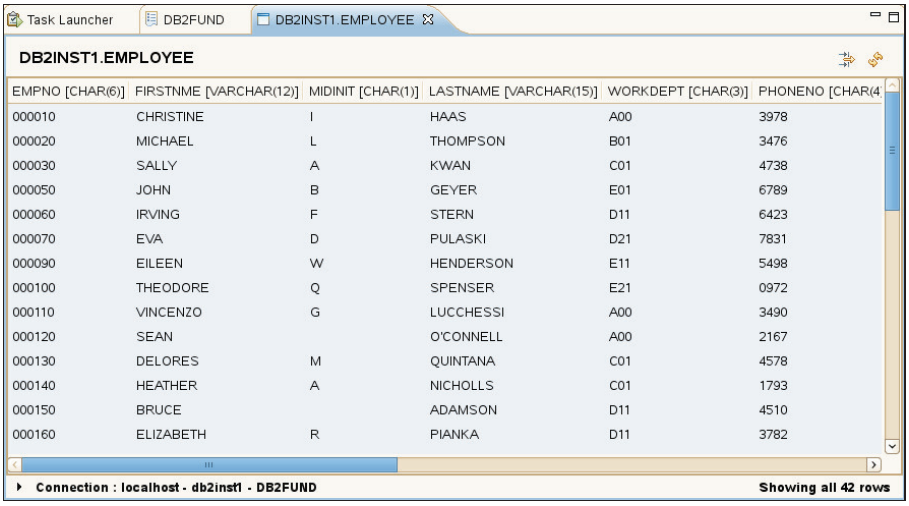


Figure 14 – Using in-place search feature to quickly find database objects

5. Remove 'E' to bring back the complete list of tables.
6. To browse the data in a table, you can right click on a table in the Object List area and select **Browse Data** from the context menu. Right click on the table EMPLOYEE and select the option "Browse Data" 

. A new tab opens in the Object List area with the records in the table displayed.



EMPNO [CHAR(6)]	FIRSTNME [VARCHAR(12)]	MIDINIT [CHAR(1)]	LASTNAME [VARCHAR(15)]	WORKDEPT [CHAR(3)]	PHONENO [CHAR(4
000010	CHRISTINE	I	HAAS	A00	3978
000020	MICHAEL	L	THOMPSON	B01	3476
000030	SALLY	A	KWAN	C01	4738
000050	JOHN	B	GEYER	E01	6789
000060	IRVING	F	STERN	D11	6423
000070	EVA	D	PULASKI	D21	7831
000090	EILEEN	W	HENDERSON	E11	5498
000100	THEODORE	Q	SPENSER	E21	0972
000110	VINCENZO	G	LUCCHESSI	A00	3490
000120	SEAN		O'CONNELL	A00	2167
000130	DELORES	M	QUINTANA	C01	4578
000140	HEATHER	A	NICHOLLS	C01	1793
000150	BRUCE		ADAMSON	D11	4510
000160	ELIZABETH	R	PIANKA	D11	3782

Figure 15 – Browsing data in EMPLOYEE table

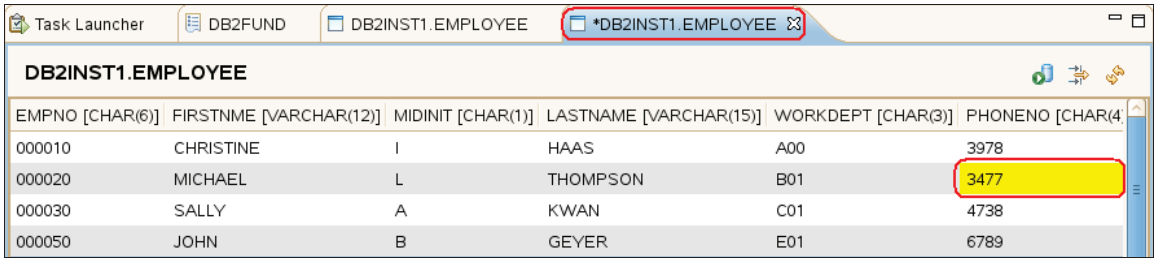
Note: When using the option "Browse Data", the data is read-only.

Note: By default in Data Studio 3.1, the maximum number of records to be displayed is set to 500. To change this behavior, on the top menu bar in Data Studio, go to Window > Preferences > Data Management > SQL Development > SQL Results View Options.

To modify individual records in a table, a simple way is the **Edit Data** option.

7. Move back to the DB2FUND tab. Right click on the EMPLOYEE table and select the option Edit Data. A new tab is opened just like browsing data, but this time the records can be modified. Double click the phone number cell for employee *Michael Thompson*, change the number from *3476* to *3477*, and then press enter.

   The modified cell is highlighted. Now the EMPLOYEE table should look like the following:



EMPNO [CHAR(6)]	FIRSTNME [VARCHAR(12)]	MIDINIT [CHAR(1)]	LASTNAME [VARCHAR(15)]	WORKDEPT [CHAR(3)]	PHONENO [CHAR(4
000010	CHRISTINE	I	HAAS	A00	3978
000020	MICHAEL	L	THOMPSON	B01	3477
000030	SALLY	A	KWAN	C01	4738
000050	JOHN	B	GEYER	E01	6789

Figure 16 – Editing data

Notice that an asterisk (*) has been added to the tab name. This is to indicate that you have uncommitted changes in that editor.

8. Click  at the upper right corner of the tab to deploy changes to the target database. Click “Yes” to confirm. Then a message should appear notifying you that the data change has been successfully committed.

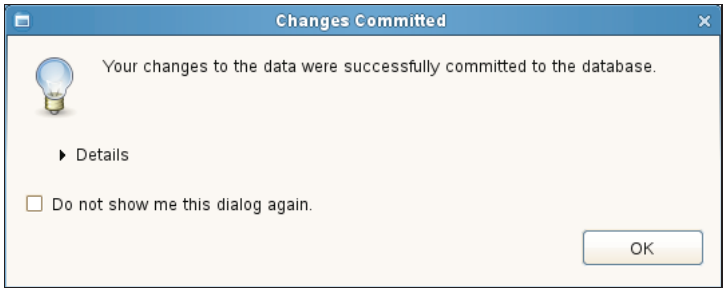


Figure 17 – Changes committed

9. Click OK to close the dialog

   The result, as well as the exact command(s) executed, can also be seen in the **SQL Results** view at the bottom of your Data Studio workspace.

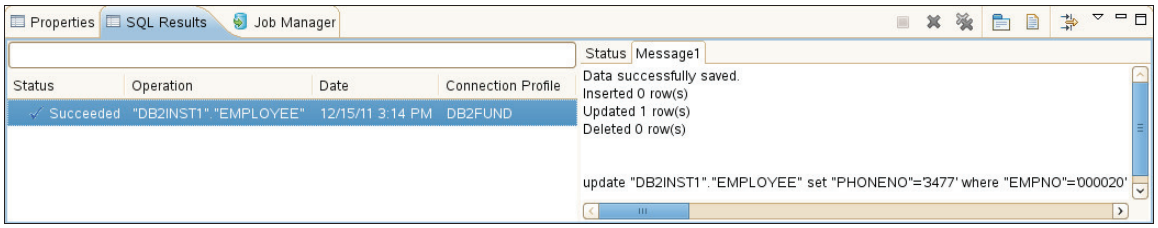


Figure 18 – SQL Results view

8.3 Administering with Task Assistants

As mentioned earlier, right clicking on an object would bring out a context-based menu of administrative tasks. If you then select a particular task, a dialog may appear in a tab in the Object List area to help you execute the task you want; such dialogs are called **Task Assistants** in Data Studio. If you are familiar with DB2 Control Center, task assistants work in a similar fashion as the wizards and advisors. Since task assistants appear in tabs, multiple task assistants can be open simultaneously. This means that you can navigate to other parts of Data Studio then come back to the task assistants that you opened earlier.

We will take the “Run Statistics” task assistant as an example to illustrate how task assistants help database administrators in managing databases. This task assistant updates statistics about the characteristics of a table and/or associated indexes, or statistical views. These characteristics include number of records, number of pages, and average record length. The optimizer uses these statistics when determining access paths to retrieve the data.

1. Go to the list of tables. Right click table SALES, go to **Manage**, select **Run Statistics**.

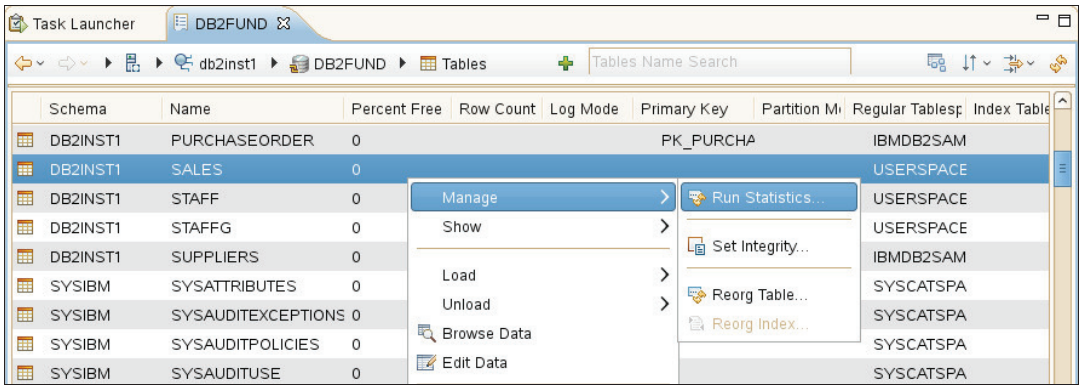


Figure 19 – Run statistics menu

The task assistant opens in a new tab.

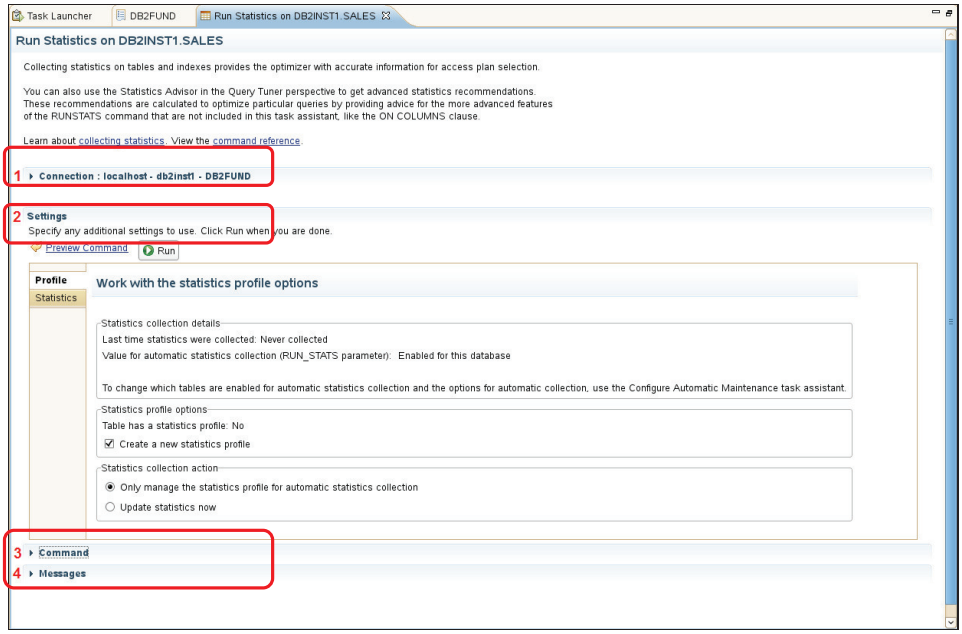


Figure 20 – Task assistant dialog for run statistics

As it is number labeled in the above screen capture, a task assistant dialog consists of 4 sections: Connection, Settings, Command, and Messages.

1. **Connection**: shows database connection details
2. **Settings**: specifies options for the database commands to be run
3. **Command**: displays the generated commands for the selected settings. Users can edit these generated commands before execution or save them for later use
4. **Messages:** shows the status of executing the commands

2. You can review the generated command to be run in the **Command** section. For this exercise, we keep the auto-generated commands, click 

to execute the commands.

   As the commands are being executed, status messages will be displayed in the **Messages** section. When the commands finished executing, the Messages section displays the overall result.

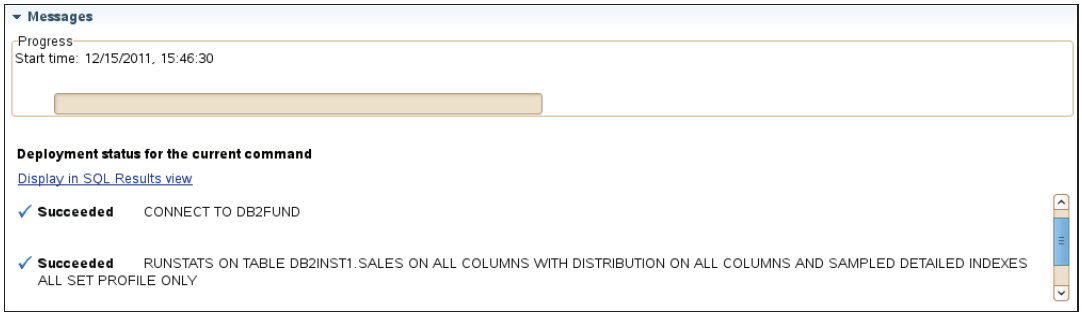


Figure 21 – Task assistant execution result in Messages section

The result can also be seen in the **SQL Results** view.

8.4 Managing Database Changes

In Data Studio, all changes made to a database are first collected in a "Change Plan" in the local workspace. A database can have multiple change plans, but only one plan can be active at a time, which means that changes are added to the current active plan. When a change plan is active, the Object List area includes an additional toolbar for the plan. A change plan remains active until the user deploys the changes to the target database, closes the plan to work on it later, or makes another plan the active plan. The list of change plans for the database can be found in the Change Plans folder in Administration Explorer.

For simplicity, we'll alter a table as an example.

1. Go back to the list of tables. Right click on table EMPLOYEE and select 

. The properties of the table should be displayed in the **Properties** view. You can now make changes directly in this view and deploy them.

2. Navigate to the Columns tab in the **Properties** view

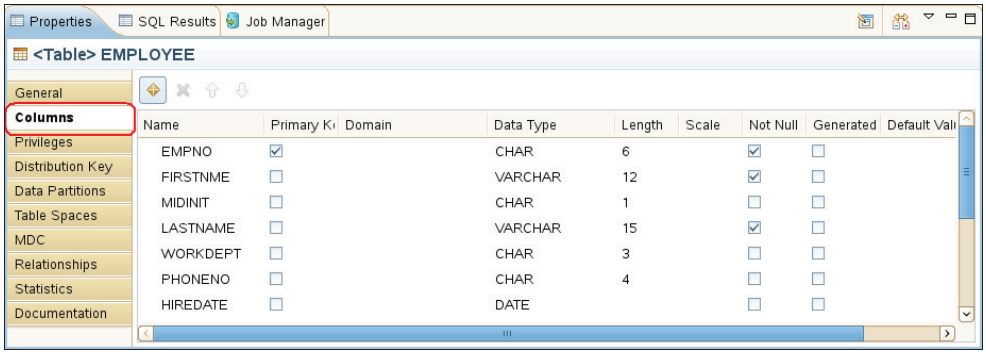


Figure 22 – Altering table definition in Properties view

3. Click  to add a new column. A new column with default name “Column1” is added at the bottom of the list. Click its **name** and change it to LOCATION.
4. Click its data type, and select VARCHAR from the drop-down menu.

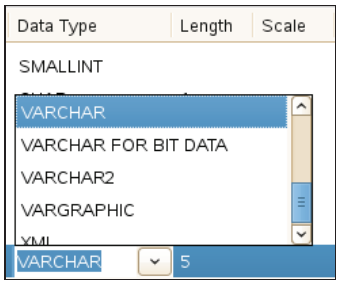


Figure 23 – Change the data type of a column

5. Change the length of the column from 5 to 20.
6. The position of a column can be changed as well. While the newly added LOCATION column is still highlighted, click  until it is placed directly under WORKDEPT.

   The final LOCATION column should look like the one below:

Name	Primary K	Domain	Data Type	Length	Scale	Not Null	Generated	Default Value/Generate Expression
WORKDEPT	☐		CHAR	3		☐	☐	
LOCATION	☐		VARCHAR	20		☐	☐	
PHONENO	☐		CHAR	4		☐	☐	

Figure 24 – New LOCATION column in table EMPLOYEE

As you are making changes to the EMPLOYEE table, a **change plan** is created in Data Studio. This can be verified by the additional toolbar for change plans in the Object List area.

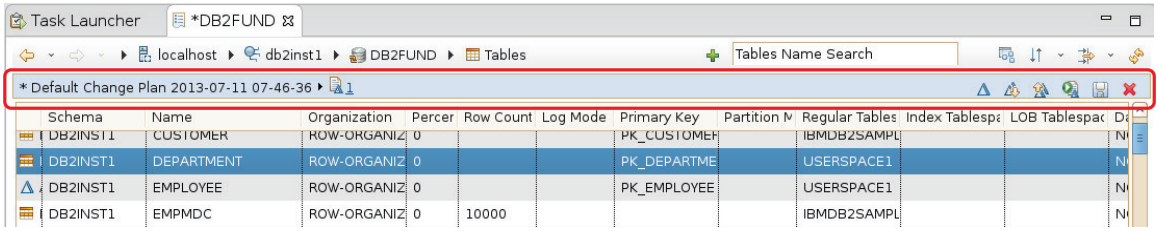


Figure 25 – Toolbar for change plans

A change plan is created with a default name of "Default Change Plan *timestamp*", where the *timestamp* has the form *yyyy-mm-dd hh-mm-ss*.

In the Object List, icons indicate whether objects for the active change plan are being created ( ), altered ( ), or dropped ( ). For example, the delta icon on the left of the EMPLOYEE table indicates that the table has been altered.

Icons on the change plan toolbar and their actions:

Icon	Description
	Shows the number of changes that are currently in the change plan. You can click the linked number to display the list of changes for the change plan.
	For the list of objects that are currently displayed in the Object List, moves the focus to the next object that is being changed in the current change plan.
	For the list of objects that are currently displayed in the Object List, moves the focus to the previous object that is being changed in the current change plan.
	Generates the DDL to implement the changes in the change plan and displays the generated DDL in the Review and Deploy dialog.
	Saves any newly defined changes to the change plan.
	Closes the change plan. If any changes have not been saved, you are prompted to save or discard the changes when the change plan is closed.

Before deploying changes, you can compare the object before and after the changes.

7. Click the altered EMPLOYEE table and go to the Properties view.
8. In the **Properties** view, click to compare both versions. A new window appears and the changes are outlined.

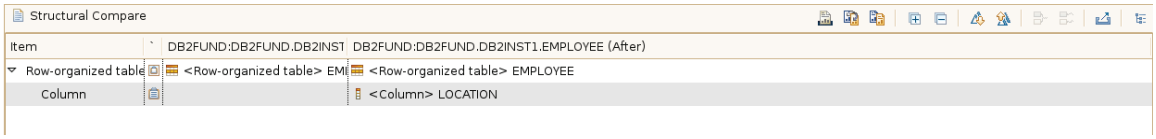


Figure 26 – Compare object before and after changes

Note: options such as generating a difference report, an impact analysis report, etc. are available on the toolbar.

9. Close the Compare window.
10. Then, in the Object List area, click the red icon on the change plan toolbar to close the plan. Finally, select **Save**. The change plan you just created has been saved in your local workspace so you can continue working on it in the future.

Note: changes you made in your change plan are kept local in your workspace. They do not affect the actual target database.

11. Go to **Administration Explorer**, expand the database folder DB2FUND and click **Change Plans**. The change plan you just saved should be listed in the Object List area.

8.5 Working with SQL

In Data Studio, the **SQL Script Editor** offers development and debugging capabilities for database routines. It also provides syntax highlighting, content assist, and several other well integrated tools to facilitate database application development.

1. In Administration Explorer, ensure that database DB2FUND is selected. On the menu bar, select "**New SQL Script**".

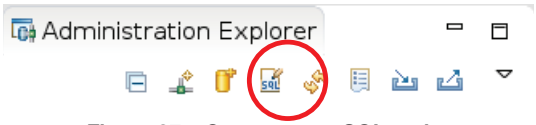


Figure 27 – Create a new SQL script

The editor is opened as a new tab in the Object List area.

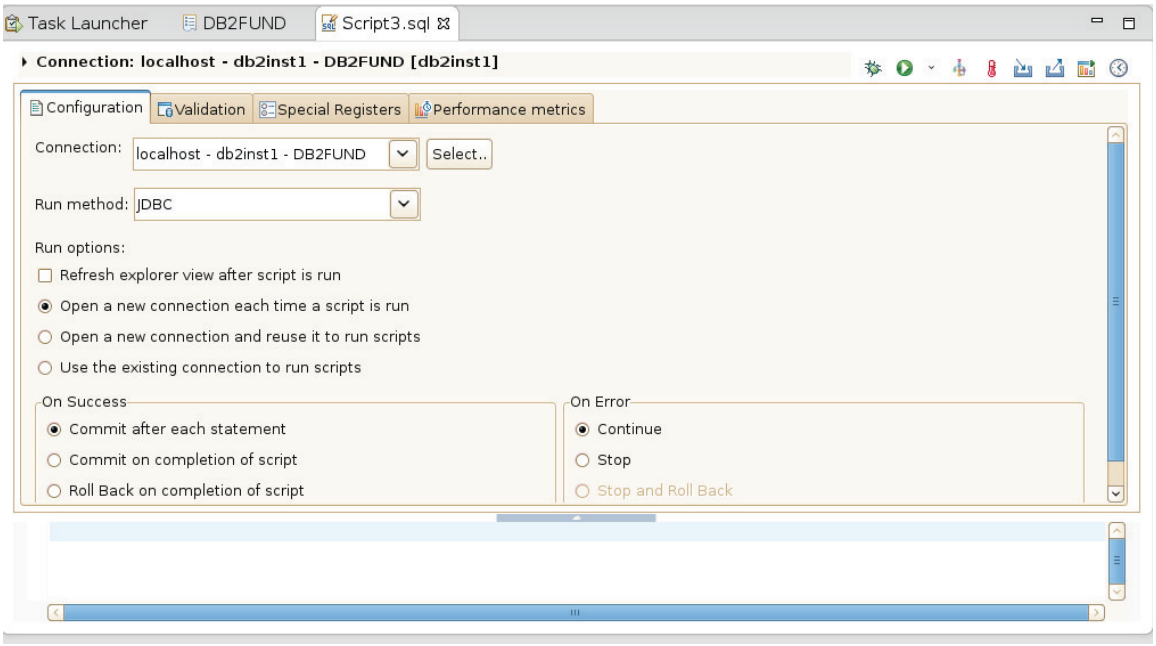


Figure 28 – SQL Editor

There are 4 tabs at the upper half of the editor to provide users with more options:

- **Configuration**: specify execution environment, commit control, error control, etc.
- **Validation**: validate syntax against a specific data server, set statement terminator, etc.
- **Special Registers**: specify values for CURRENT SCHEMA and CURRENT PATH special registers.
- **Performance Metrics**: specify an Optim Performance Manager profile for gathering performance metrics.

The integrated toolbar at the top provides convenient links to tools such as: **Visual Explain, Tuning, Import** / **Export, Job Manager,** etc.

Some useful keyboard shortcuts you can use in the editor:

- **Content Assist**: Ctrl + Space
- **Content Tip**: Shift + Ctrl + Space
- **Format SQL**: Ctrl + Shift + F
- **Toggle Comment**: Ctrl + /

2. In the editor, enter the following SQL statement:

```
select * from department;
```

Before executing the query, you can ask Data Studio to properly format your query.

3. Inside the editor, press CTRL+SHIFT+F on your keyboard, or right click within the editor and select Format SQL from the context menu. Your query should be transformed into the following format:

Figure 29 – Formatted SQL statement

4. Right click at the margin of the editor and select Show Line Numbers. This feature is especially useful if you have many lines of SQL statements.

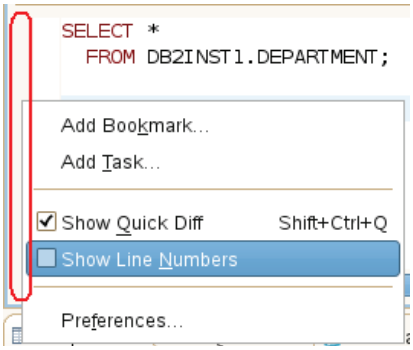


Figure 30 – Show line numbers

5. Press F5 on your keyboard or click the Run SQL icon to execute your statement.

   The query result is displayed in the SQL Results view. The **Status** tab shows the status of the query execution. The **Result** tab shows actual data records returned by your query. You can also view the history of all previously executed commands in the left panel of the “SQL Results” tab.

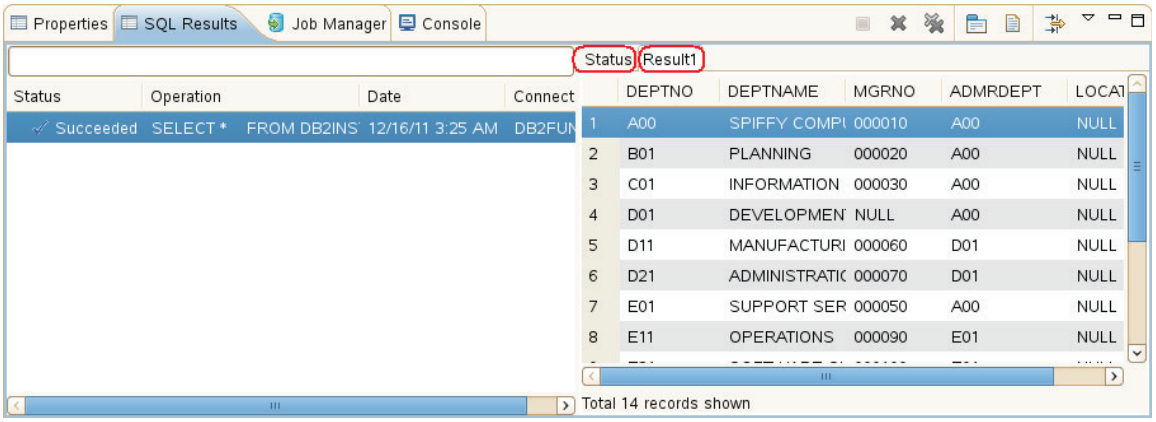


Figure 31 – Result of executed SQL statement

6. Toggle the icon at the top right of the SQL Results view to display the query result in text mode. This is especially useful if you need to copy and paste the results to an external text editor.

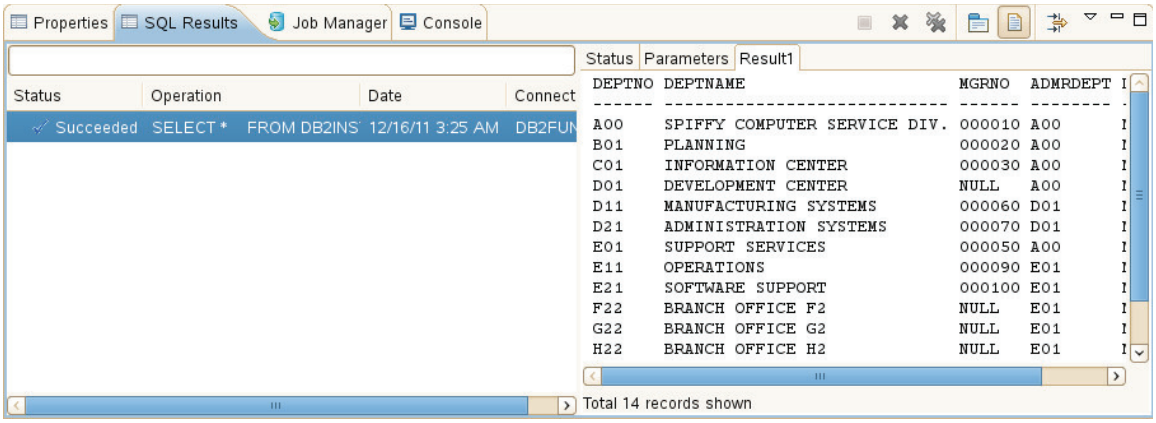


Figure 32 – Result of executed SQL statement in text mode

7. To display query results in a single tab, i.e. not to make the execution status tab the default opening tab, toggle the button.

8. Execute the query again by pressing button. This time you should see the results right away.

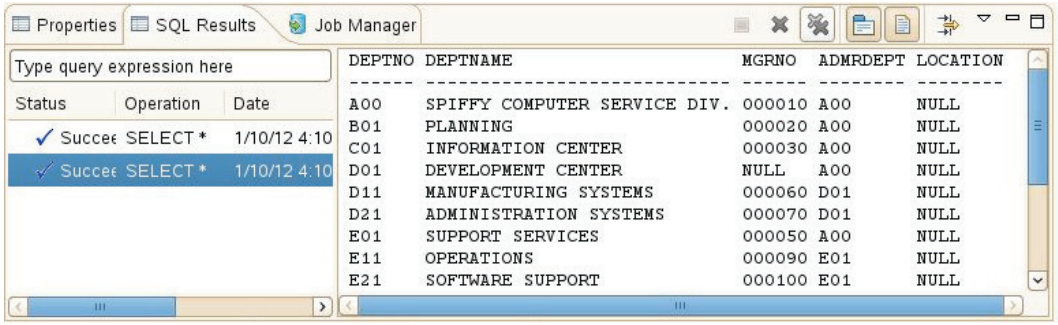


Figure 33 – Result displayed in a single tab

9. Toggle the icon again to turn off the text mode and display the results in the graphical tabular format.

8.5.1 Visual Explain

You can use **Visual Explain** to view the access plan generated by DB2's SQL compiler as a graph. You can use the information in the graph to tune your SQL queries for better performance. The nodes in the graph represent tables and indexes and each operation on them. The links between the nodes represent the flow of data.

1. We want to find out which department has contributed how many sales. Clear your previous SQL statements in the editor and enter the following SQL statement.

```
SELECT d.deptno, d.deptname, SUM(s.sales) as total_sales
      FROM sales s INNER JOIN employee e ON s.sales_person = e.lastname
           INNER JOIN department d ON d.deptno = e.workdept
      GROUP BY d.deptno, d.deptname
      ORDER BY d.deptno;
```

2. To view the access plan diagram of this query, click the **Open Visual Explain** icon on the integrated toolbar to open Visual Explain. In the pop-up window, keep the default values, click Finish.

3. The diagram is then displayed in the Access Plan Diagram view at the lower section in your workspace. Double-click the tab Access Plan Diagram to maximize the view.

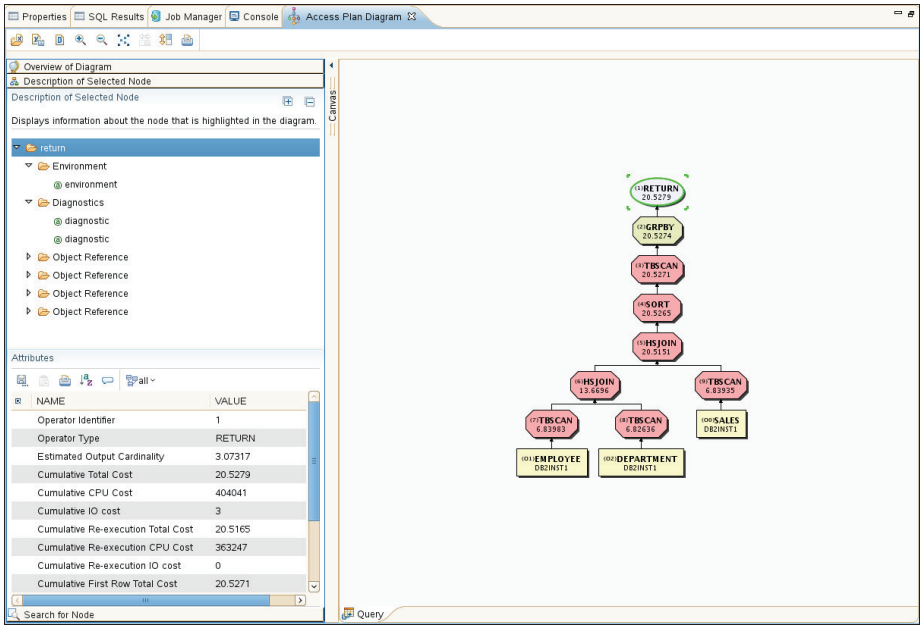


Figure 34 – Visual Explain

The Access Plan Diagram view has three sections:

- The toolbar on the top contains the following buttons:

Button	Action
	Open a saved diagram To open the Access Plan Diagram view so that you can open a saved diagram, select Window > Show View > Other. In the Show View window, select Data > Access Plan Diagram.
	Save the diagram **Note**: On Linux, Visual Explain appends the extension .xml to the file name automatically.
	View the SQL or XPATH statement
	Zoom in
	Zoom out
	Scale the diagram to fit the Canvas
	Flip the diagram horizontally

	Flip the diagram vertically
	Print the entire diagram or only a portion that you selected

- The left-side area contains three panes:
  - The **Overview of Diagram** pane helps you to navigate through a diagram.
  - The **Description of Selected Node** pane lists the data elements and the associated predicates of the selected node.
  - The **Search for Node** pane lets you search for a node in a diagram.
- The right-side area contains the **Canvas**, which displays the access plan graph. These graphs show the operations that are required by data servers to derive the results of SQL, XPATH, or XQUERY statements.

  The access plan graph consists of a tree displaying nodes. These nodes represent:
  - Tables, shown as rectangles
  - Indexes, shown as diamonds
  - Operators, shown as octagons (8 sides). TQ operators, shown as parallelograms
  - Table functions, shown as hexagons (6 sides).

  Numbers displayed on each node represent ***cost***, which is the estimated total resource usage necessary to execute the access for a statement or an element of a statement. Cost is derived from a combination of CPU cost (in number of instructions) and I/O (in numbers of seeks and page transfers).The unit of cost is the ***timeron***. A timeron does not directly equate to any actual elapsed time, but gives a rough relative estimate of the resources (cost) required by the database manager to execute two plans for the same query. The cost shown in each operator node of an access plan graph is the cumulative cost, from the start of access plan execution up to and including the execution of that particular operator. It does not reflect factors such as the workload on the system or the cost of returning rows of data to the user.

  Clicking on a node in the access plan graph would reveal detailed information about that node.

4. Minimize the Access Plan Diagram view by double-clicking again the 

tab.
5. In the SQL Editor, press F5 on your keyboard or click  to execute this statement. The query result is displayed in the SQL Results view.

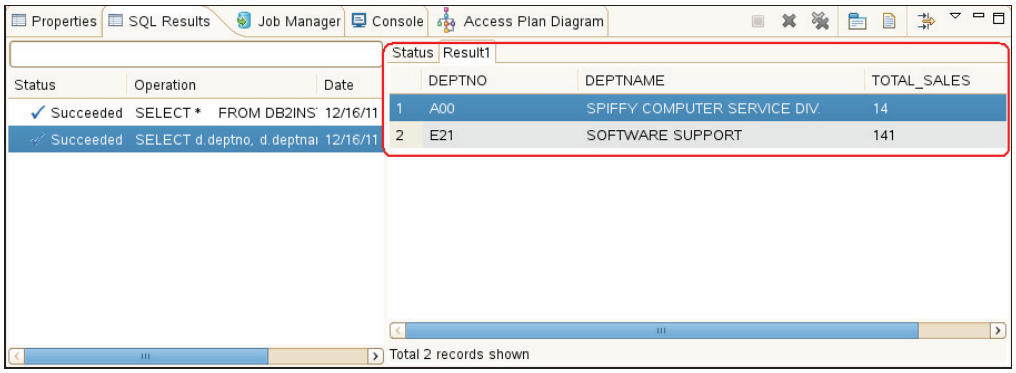


Figure 35 – Sales by department query result

8.5.2 Exporting Results

Exporting data is a way to output data to a spreadsheet or other formats for sharing, archiving, or further calculations.

1. Export this result set by right-clicking in the Result1 tab, select **Export**, then select **Current Result**

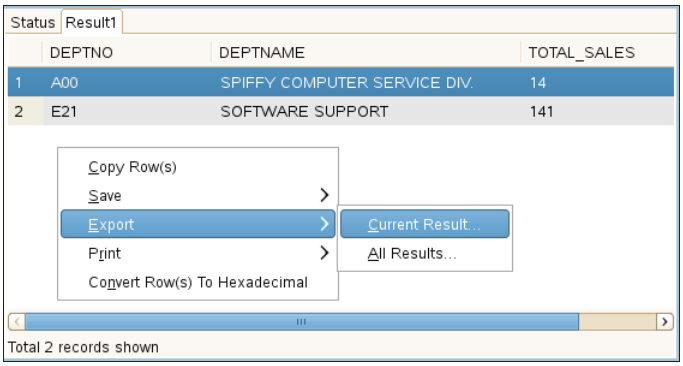


Figure 36 – Context menu to export query result

2. Enter "/home/db2inst1/query_result" in the **File Name** field, and select **CSV File** as the Format. The file extension .csv will automatically be appended to the file name when you select the CSV file format.

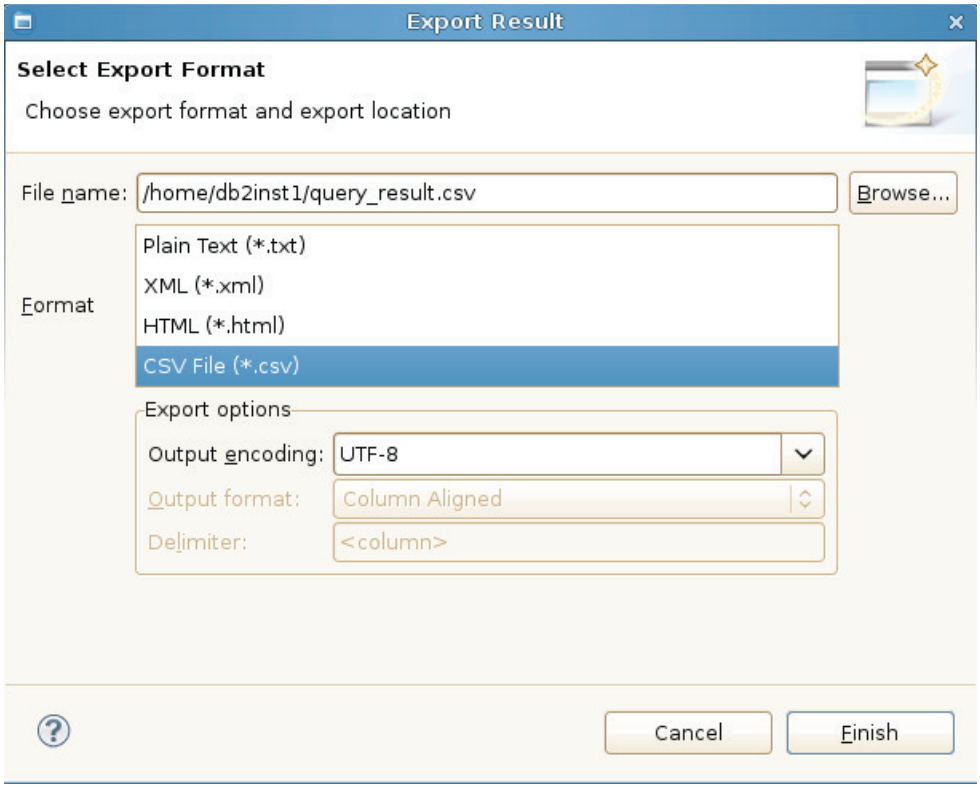


Figure 37 – Export Result dialog

3. Click **Finish**.

   File **query_result.csv** is saved **at** /**home**/**db2inst1**/ in your file system.

9 Summary

IBM DB2 for Linux, UNIX and Windows is an industry-leading database management system that has constantly delivered new features to address ever-increasing demands and requirements on data. Database administrators and application developers can reap immediate benefits from using DB2.

IBM Data Studio is a comprehensive database management tool that allows administrators and developers to effectively work with DB2 databases.

In this lab, you have learned how to leverage Data Studio to perform basic tasks on DB2 such as:

- Establish a database connection
- Browse database objects

- Leverage task assistants to carry out administrative tasks
- Manage database changes with change plans
- Create and execute queries in SQL
- Analyze data access plan using Visual Explain
- Export query result for later use

10 Cleanup

1. To clean your environment after completing the exercise, right-click on **DB2FUND** database in the **Administration Explorer** view and select **Disconnect.**
2. Next, close Data Studio and switch back to the terminal window that you have been using so far.
3. Finally, execute the commands below. Use “password” as password.

```
su -
cd /home/db2inst1/Documents/LabScripts/Fundamentals/setup
./cleanup.sh
```

4. If you would like to redo this lab in the future, please execute the following commands in a terminal window as root:

```
cd /home/db2inst1/Documents/LabScripts/Fundamentals/setup
./config.sh
```

IBM Canada
8200 Warden Avenue
Markham, ON
L6G 1C7
Canada

DB2® Storage Optimization

f *Hands-On Lab*

Table of Contents

50 Introduction

Formatted: Bullets and Numbering

Every business model looks for the best way to minimize production costs in order to maximize profits. One of the most prominent features introduced in DB2 is the ability to perform row compression. The goal of row compression is to reduce database storage needs. When using this feature, DB2 is able to shrink row sizes by building and utilizing a compression dictionary, which is used as an alphabet to compress the rows of data in the database tables. Since more data can reside on a data page, fewer data pages are allocated per table. The net effect is a reduced rate of storage consumption which may lead to fewer disks needed when allocating storage requirements. Since disk storage is one of the most expensive components on any data server, this feature can significantly reduce Total Cost of Ownership (TCO).

As part of the Storage Optimization feature, DB2 allows row compression of regular SQL data types, XML data, index compression, temporary table compression, and new in DB2 10, adaptive compression further enhances the benefits of classic row compression.

51 Suggested Reading

Formatted: Bullets and Numbering

Best Practices: Deep Compression

http://www.ibm.com/developerworks/data/bestpractices/deepcompression/

DB2 Deep Compression – Best Practices and Customer Experiences

http://www.channeldb2.com/events/db2-deep-compression-best

52 About this Lab

Formatted: Bullets and Numbering

In this lab, we will be taking a closer look at the various compression techniques available including row compression (available since DB2 9), XML XDA compression (available since DB2 9.7), index compression (available since DB2 9.7), LOB inlining (available since DB2 9.7), and the **adaptive compression feature new to DB2 10**. You will also learn the basics on how to enable different compression technologies for a table and collect various statistics

- **Examine table properties**
  - Make use of the available table function ADMIN_GET_TAB_INFO to see the physical sizes of the different data objects (relational data, XML and LOB data, and indexes).
- **Inline LOB column**
  - Get an estimate of the maximum length required to inline a particular LOB column.
- **Run sample workload against table**
  - Use Data Studio to run a workload and view the access plan of the DB2 Optimizer.
- **View compression estimates**
  - Use the available table functions to view an estimate of the possible savings when row compression is enabled (includes XML and index compression).
- **Apply Static and Adaptive compression**
  - Compress tables
  - Examine table properties post-compression
  - Run sample workload post-compression

53 DB2 Storage Optimization

Formatted: Bullets and Numbering

53.1 Environment Setup Requirements

Formatted: Bullets and Numbering

To complete this lab you will need the following:

- DB2 10 VMware® image
- VMware Workstation 6.5 or later

53.2 Preparing for the lab

Formatted: Bullets and Numbering

1. Login to the VMware virtual machine using the following information:
   - Username: **db2inst1**
   - Password: **password**
2. Right-click on the Desktop area and choose the "Open in Terminal" option:

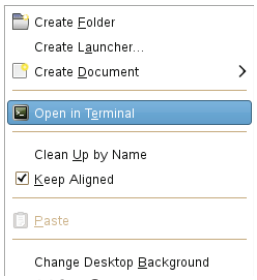


Figure 116 - Open a new terminal

3. As the user **db2inst1**, enter the following command in the terminal window to start the database instance. Make sure the instance is started successfully.

```
db2start
```

The Database Manager is now started and we can begin the exercises by inspecting the available tables and data. There are three pre-created tables for this exercise as described below:

- **CUSTOMERS** – This is a large table that contains customer data such as names, address, phone number, date of purchase, an XML document, which describes the purchase in detail, and a text document (stored in the table as a CLOB value), for each customer. The number of rows in the table is 100,000.
- **CUSTOMERS_ARCH** – This is a small table that contains similar data as in the table described above but is very small in cardinality – only 30 rows.
- **CODES** – A relatively large table with about 50,000 rows, which contains coding information for about 50,000 customers. The data included is numerical.

To see the description of these tables and understand the data types of the included columns, connect to CUSTINFO and view the tables.

4. Open up Data Studio by clicking on the **"IBM Data Studio"** icon on the desktop.

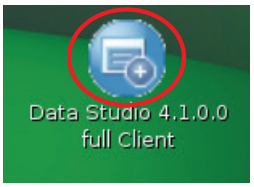


Figure 117 - Open Data Studio

5. Click OK to accept the default workspace.

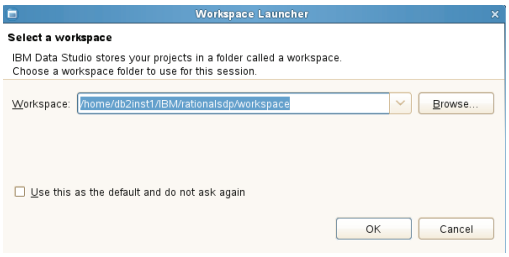


Figure 118 - Select the default workspace

6. By default, the **Database Administration** perspective is opened. Your workspace should look like the following :

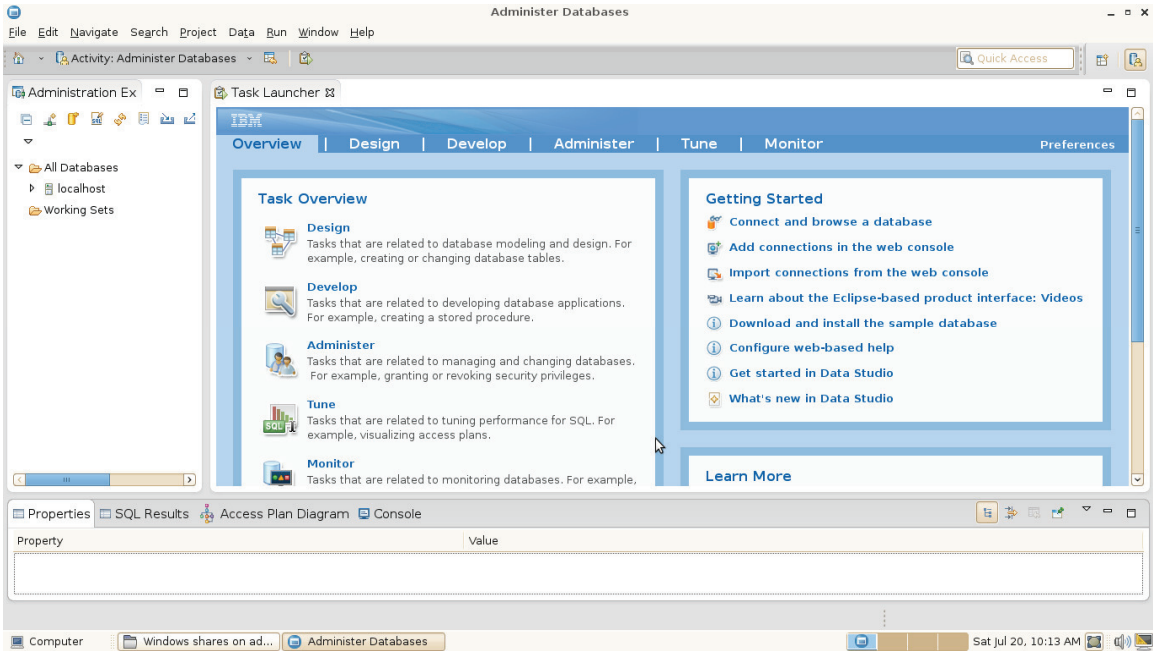


Figure 119 - Data Studio Workspace

7. In the **Administration Explorer** view, expand the folder **All Databases** and subfolders until you see **CUSTINFO**. Right click on **CUSTINFO** and select **Connect**.

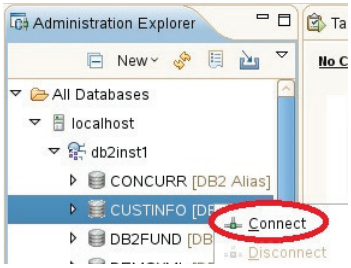


Figure 120 - Connecting to CUSTINFO database

8. In the pop-up dialog, fill in the connection details as follows, and click **Ok**.
    - Database : **CUSTINFO**
    - Host : **localhost**

- Port number : **50001**
- User name : **db2inst1**
- Password : **password**

9. After a connection has been successfully established, expand the **CUSTINFO** database folder in **Administration Explorer** and select the **Tables** folder.

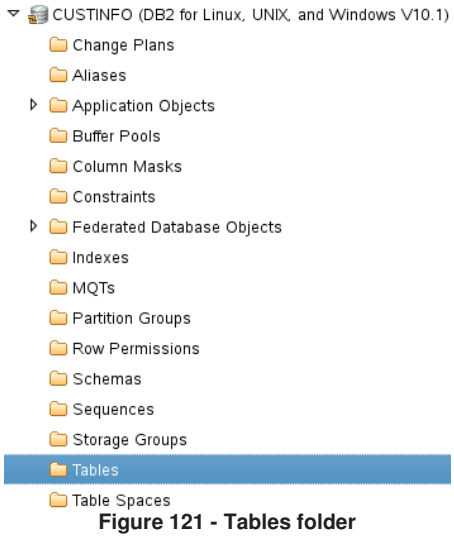


Figure 121 - Tables folder

The list of tables is automatically displayed in the Object List area on the right. Note that its properties are also displayed (table space, row count, etc.).

10. Select each one of the tables listed below and inspect their properties in the **Properties** view.
    - **CUSTOMERS**
    - **CUSTOMERS_ARCH**
    - **CODES**

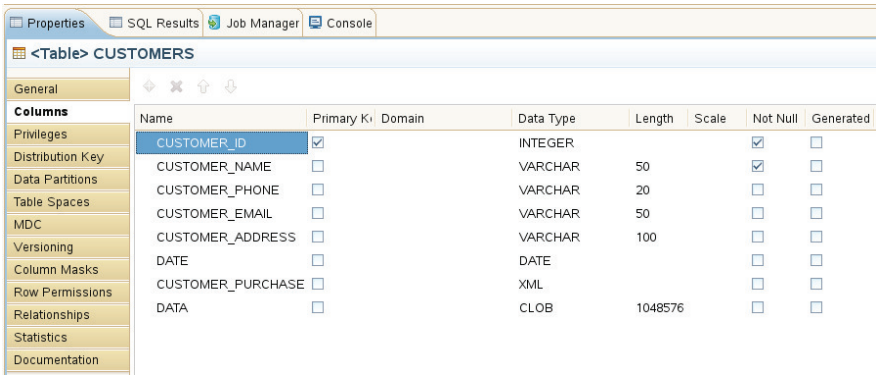


Figure 122 – Table's properties

To improve search on the large table **CUSTOMERS**; let's create one index on the **CUSTOMER_NAME** column and another index on the XML data, which would point to the `/PurchaseOrder/itemDetails/item/partid` node.

11. In the **Administration Explorer** view, right click the **Indexes** folder and select **Create Index**

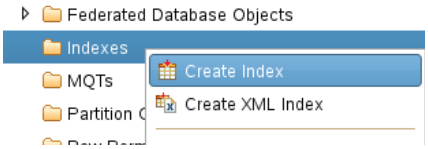


Figure 123 - Create new index

12. In the **Select a table** popup, expand the **DB2INST1** option, select **CUSTOMERS** and press **OK.**
13. In the **Properties** view, change the index name to **NAMEINDEX**, then select the **Columns** tab.

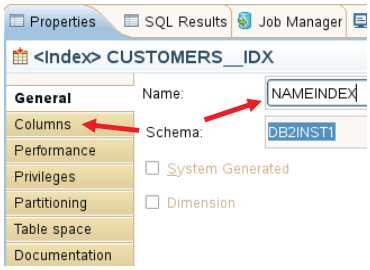


Figure 124 – NAMEINDEX properties

14. Click on the **Search (…)** option, select **CUSTOMER_NAME** and click **OK**.

    Your index should look similar to the image below:

Figure 125 – Use ascending sort

15. Now click on the **Review and Deploy (**  **)** button on the top right section of the screen, uncheck the **Save Data** option and press **Refresh DDL**.
16. Next, click on **Advanced Options**, uncheck all maintenance options and press **Finish.**

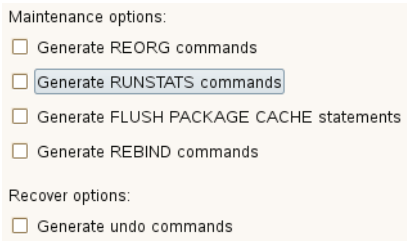


Figure 126 - Uncheck all maintenance options

Your **Review and Deploy** popup should be similar to this:

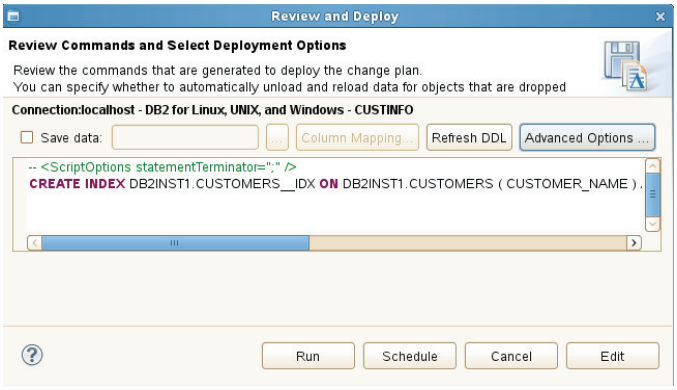


Figure 127 - Review and Deploy popup

17. Finally, press **Run** to execute the CREATE INDEX SQL statement.

Besides Data Studio, indexes can also be created using command line statements.

18. Switch to the Terminal Window and execute the following command to create the **ITEMID** xml index.

```
db2 connect to CUSTINFO

db2 "CREATE INDEX ITEMID ON CUSTOMERS(CUSTOMER_PURCHASE) GENERATE KEY USING
XMLPATTERN '/PurchaseOrder/itemDetails/item/partid' as SQL VARCHAR(6)";
```

We have now prepared our data and in the next section we will examine the properties of each table.

53.3 Examine Table Properties

Formatted: Bullets and Numbering

We will now use the various techniques available to view compression statistics for the tables as well as table properties. After we have examined the statistics, we will make a decision on how to optimize the storage for the CUSTINFO database.

The administrative table function SYSPROC.ADMIN_GET_TAB_INFO provides a method to retrieve table size and state information that is not currently available in the catalog views. It can be used to retrieve significant information about tables. Some of the information pieces include the actual physical storage that relational, XML, and large object data occupy on disk. We can use this information to calculate the overall table size. Based on this we can determine if there is a need to use compression to reduce the overall storage and possibly improve performance of queries.

8. In Data Studio, click menu **File > Open File** on the top left hand corner.
9. Navigate to **/home/db2inst1/Documents/LabScripts/Compression** directory and open the file named **get_table_properties.sql.** If prompted, press **Ok** to use the default statement terminator (semi-colon).

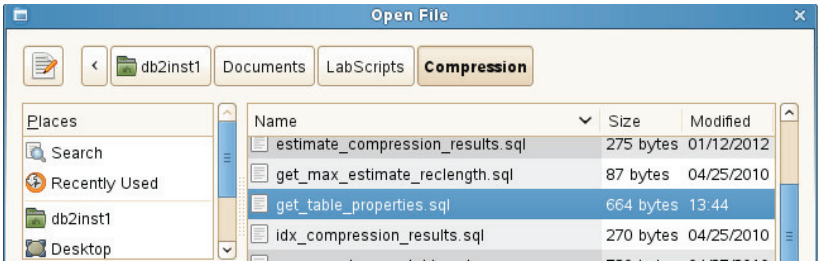


Figure 128 - Open file get_table_properties.sql

10. Before running the file, click on the **No Connection** link on the top left corner of the script as shown below.

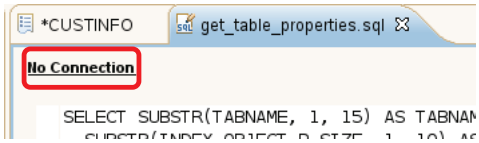
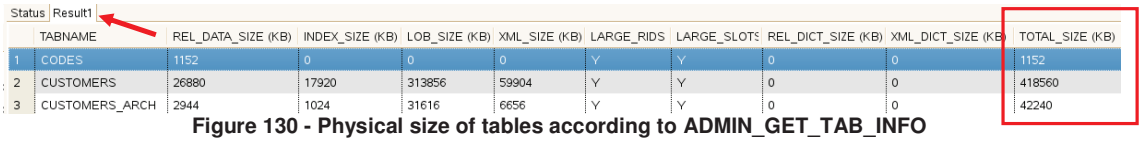


Figure 129 - No Connection link

11. Select **CUSTINFO** connection profile and click **Finish**.

12. Now execute the script by pressing the run button in Data Studio, or press **F5** while the cursor is inside the SQL Editor.

    The results will be displayed on the bottom left hand corner of the screen, in the **Result1** tab in the **SQL Results** view,



Status | Result1

	TABNAME	REL_DATA_SIZE (KB)	INDEX_SIZE (KB)	LOB_SIZE (KB)	XML_SIZE (KB)	LARGE_RIDS	LARGE_SLOTS	REL_DICT_SIZE (KB)	XML_DICT_SIZE (KB)	TOTAL_SIZE (KB)
1	CODES	1152	0	0	0	Y	Y	0	0	1152
2	CUSTOMERS	26880	17920	313856	59904	Y	Y	0	0	418560
3	CUSTOMERS_ARCH	2944	1024	31616	6656	Y	Y	0	0	42240

Figure 130 - Physical size of tables according to ADMIN_GET_TAB_INFO

Notice the **TOTAL_SIZE** column:

- The largest table is the CUSTOMERS table, which contains 100,000 rows and around 418MB in size.
- The second largest is the CUSTOMERS_ARCH table, with 10,000 rows, and it is about 42MB.
- The smallest table is our CODES table, which is only around 1MB.

There are no indexes on the CODES table. The indexes on the CUSTOMERS_ARCH table are system-required and the ones on the CUSTOMERS table are a mix of system-required and the two indexes we created in the previous section. *Notice the different sizes for the relational, XML, and LOB data.*

For your reference, the next table has the table sizes from above. You'll use these to compare the compression results later in this lab.

Note: the values below might differ, slightly, from your results depending on your current environment.

Column name	CUSTOMERS	CUSTOMERS_ARCH	CODES
REL_DATA_SIZE	26880	2944	1152
INDEX_SIZE	17920	1024	0
LOB_SIZE	313856	31616	0
XML_SIZE	59904	6656	0
REL_DICT_SIZE	0	0	0
XML_DICT_SIZE	0	0	0
TOTAL_SIZE (KB)	418560	42240	1152

Now, check if these tables are enabled for compression.

13. In the **Administration Explorer** view and click on the **Tables** folder.
14. Select the **CUSTOMERS** table, go to the **Properties** view and verify that there are no compression techniques enabled on the table.

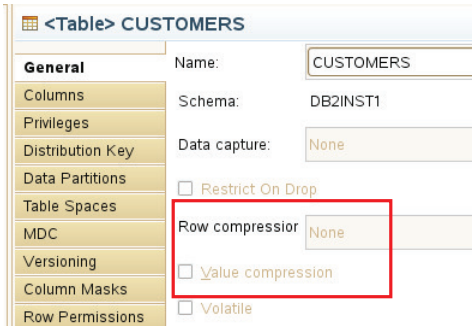


Figure 131 - Compression not enabled

15. Repeat the previous step for the **CUSTOMERS_ARCH** and **CODES** tables.

    The result shows that compression is not activated on any of the three tables.

53.4 Inline Large Object (LOB) Data

Formatted: Bullets and Numbering

LOBs can be quite large in size (from some kilobytes up to 2GB in DB2), and are generally stored separately from the base table, in a location managed by the database manager. When retrieving such objects, DB2 fetches the page with the relational data, and uses the LOB descriptor stored within the table row to locate the LOB on the system. This process results in an additional I/O operation to access the LOB itself.

To simplify the manipulation of smaller LOBs, DB2 9.7 introduced the ability to have LOB data that falls below a size threshold to be inlined within the base table rows. This reduced the overhead of retrieving LOBs separately as these LOBs could then be fetched with the base table data and cached in the buffer pool.

In addition, inlining LOBs within a base table row, qualifies them for row compression. DB2 cannot compress LOB values that are stored in the separate LOB storage area. By inlining the LOB values, we get better performance and improved compression of data within the system.

We will demonstrate how to inline LOB values using the CUSTOMERS table. The first step is to know what inline length to specify. DB2 provides a table function called `ADMIN_EST_INLINE_LENGTH`, which can estimate the inline length required to inline data stored in an XML or LOB column. We will use it to determine the approximate inline length for the CLOB column in the CUSTOMERS table.

1. In Data Studio, click menu **File > Open File** on the top left hand corner.
2. Navigate to **/home/db2inst1/Documents/LabScripts/Compression** directory and open the file named **get_max_estimate_reclength.sql.** If prompted, press **Ok** to use the default statement terminator (semi-colon).
3. Click on the **No Connection** link, select the **CUSTINFO** connection profile and press **Finish.**
4. Now execute the script by pressing the run button in Data Studio, or press **F5** while the cursor is inside the SQL Editor.

You should see an output similar to the one shown below. The returned value (5408) is the largest length that can be specified to inline the column **DATA**.

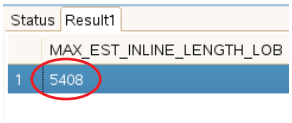


Figure 132- Estimate maximum record inline length

Lets alter the table **CUSTOMERS** to inline the **DATA** column.

5. Right click on the **CUSTOMERS** table, and select **Show > Columns**.

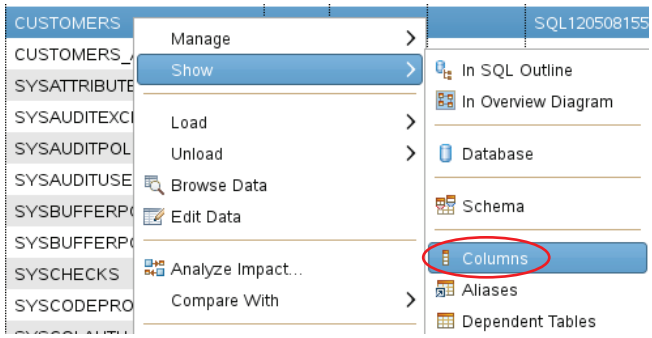


Figure 133 - Show CUSTOMERS' table columns

6. Click on the **DATA** column and select **Edit** from pane below.

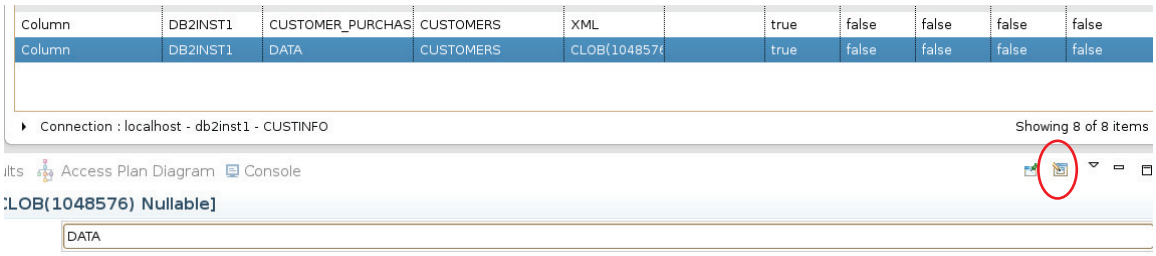


Figure 134 - Alter the DATA column definition

7. Go to the **Properties** view, select the **Type** tab and set the **Inline Length** value to **5408.**

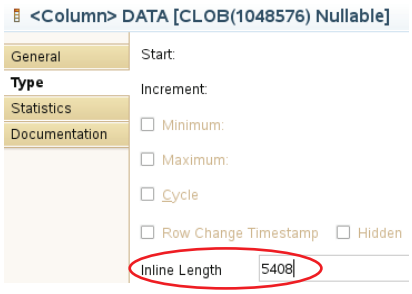


Figure 135 - Input the Inline Length value

8. Click on the **Review and Deploy (**  **)** button on the top section of the screen, uncheck the **Save Data** option, then click on **Refresh DDL.**
9. Next, click on **Advanced Options**, uncheck all maintenance options and press **Finish.**

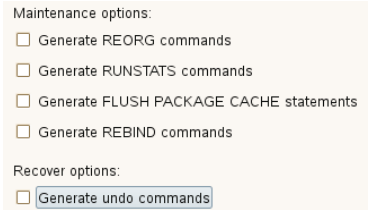


Figure 136 - Uncheck maintenance options

10. Finally, review the ALTER TABLE SQL statement, and press **Run.**

    The command should report a successful status, as shown below.

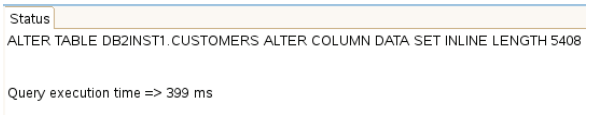


Figure 137 - Set inline length

With the above ALTER TABLE command, LOB inlining has been enabled, although the existing LOB data has not been moved as part of the command. To move LOB data, a table reorg is to be performed on the LOB data. With the inline length set to 5408, all LOBs that are smaller than 5408 bytes will be inlined within the base table rows. Later, a Runstats will update the statistics.

11. Right click on the **CUSTOMERS** table and select **Manage > Reorg Table.**

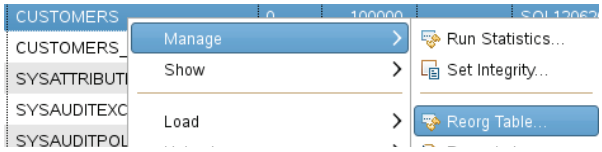


Figure 138 - Execute a reorg operation on CUSTOMERS table

12. Mark the **Reorganize long field and large object data** option.

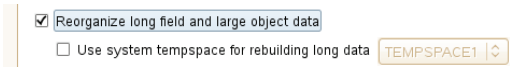


Figure 139 – Reorg on table CUSTOMERS

13. Click on the **Run (**) button on the top section of the screen to execute the reorg operation.
    The commands might take a few minutes to complete because of the size of the table.

    Next, let's update the statistics for the **CUSTOMERS** table using the Runstats command so that the DB2 optimizer can generate an optimized access plan.
14. Right click on the **CUSTOMERS** table, select **Manage > Run Statistics.**
15. Mark both **Update the existing profile** and **Update statistics now** options and go to the **Statistics** tab**.**

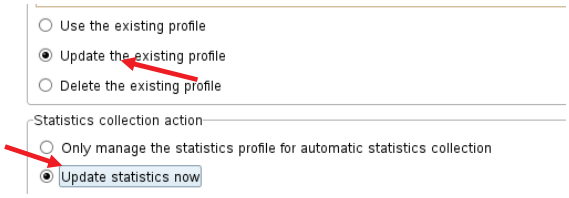


Figure 140 - Configure runstats operation

16. Mark the **Collect no statistics on all columns** and **Collect basic statistics on all indexes** options**.**

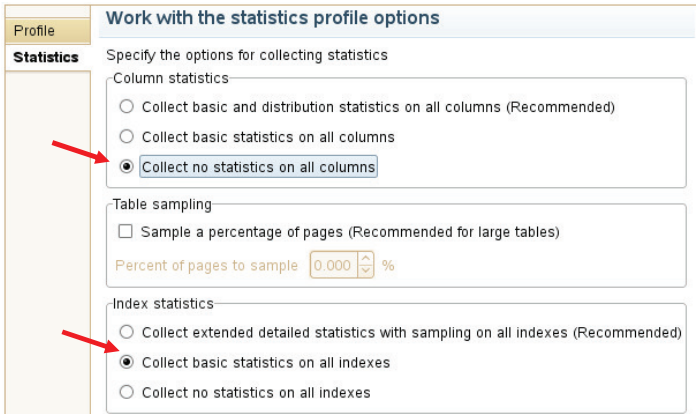


Figure 141 - RUNSTATS on table CUSTOMERS

17. Click on the **Run (** **)** button on the top section of the screen to execute the runstats command.
18. Now check the table properties again using the previously open **get_table_properties.sql** script to see how data storage has changed.

Status | Result1

	TABNAME	REL_DATA_SIZE (KB)	INDEX_SIZE (KB)	LOB_SIZE (KB)	XML_SIZE (KB)	LARGE_RIDS	LARGE_SLOTS	REL_DICT_SIZE (KB)	XML_DICT_SIZE (KB)	TOTAL_SIZE (KB)
1	CODES	1152	0	0	0	Y	Y	0	0	1152
2	CUSTOMERS	292608	15616	1024	59904	Y	Y	0	0	369152
3	CUSTOMERS_ARCH	2944	1024	31616	6656	Y	Y	0	0	42240

Figure 142 - CUSTOMERS table property after REORG and Runstats

Notice the **LOB_SIZE** column value for the **CUSTOMERS** table. It is only 1MB compared to the initial value of 313MB (section 53.3). Most of the LOB values were inlined, increasing the **REL_DATA_SIZE** in physical size. The reorganization of the LOB data has resulted in some of the system-required indexes to be reduced in size, reducing the **TOTAL_SIZE** of the data as well.

Record the column values for the three tables here and compare them to your previous recordings.

Column name	CUSTOMERS	CUSTOMERS_ARCH	CODES
REL_DATA_SIZE			
INDEX_SIZE			
LOB_SIZE			
XML_SIZE			
REL_DICT_SIZE			
XML_DICT_SIZE			
TOTAL_SIZE(KB)			

53.5 Run Sample Workload

Formatted: Bullets and Numbering

For our reference, we will run a sample workload and use the **Visual Explain utility** to check the estimated execution cost. This estimated execution cost will be used in later sections to compare the effect that various compression techniques have on query performance.

Open a new SQL Editor. In the **Administration Explorer** view, click on **New > New SQL Script**.

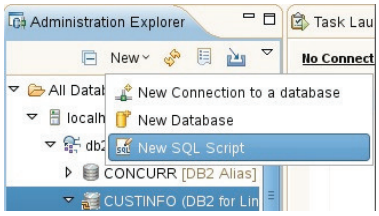


Figure 143 - Create new SQL script

19. In the SQL editor, type in the following SQL statement and press F5 (or click  on the top-right hand corner of the SQL Editor) to execute the query. If prompted, select the **CUSTINFO** connection profile and click **Finish**.

```
SELECT * FROM CUSTOMERS;
```

20. At the bottom of the Data Studio screen, in the SQL Results pane, you should see a Successful result message.

Figure 144 - SQL execution succeeded

21. In the SQL Editor, click the Open Visual Explain button  . The button is next to the ‘  button.
22. Keep the default settings in the configuration window that pops up, click **Finish** and wait until the utility completes execution.
23. Double click on the **Access Plan Diagram** tab to expand the bottom pane of Data Studio to see the results:

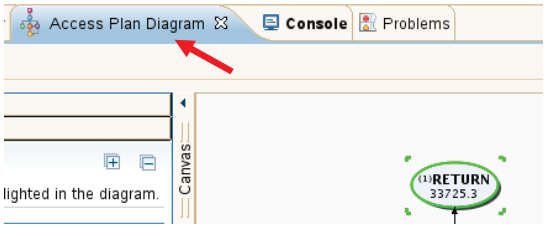


Figure 145 - Expand the Access Plan Diagram pane

24. Click on the **RETURN** operator. Note the estimated cost to execute the query and record it. It should be something close to 33725.3 timerons.

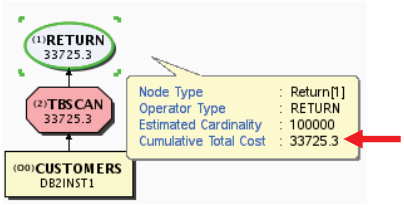


Figure 146 - Cumulative total cost before compression

25. Restore the Access Plan Diagram window to its original size by clicking the button.

53.6 Estimate Compression Savings

Formatted: Bullets and Numbering

Now we are going to estimate the compression savings if we were to enable row compression on our various tables. For that, we'll make use of the table function `SYSPROC.ADMIN_GET_TAB_COMPRESS_INFO`. This function provides a variety of estimates, some of which include an estimated percentage of how many pages could be saved, an estimated percentage of how many bytes would be saved, as well as overall estimated record length after compression. This information is reported for both relational and XML data. The estimate provides an analysis of both static and adaptive compression.

With regards to indexes, another table function is made available to users to estimate index compression savings – `SYSPROC.ADMIN_GET_INDEX_COMPRESS_INFO`.

1. In Data Studio, click menu **File > Open File** on the top left hand corner.
2. Navigate to **/home/db2inst1/Documents/LabScripts/Compression** directory and open the file named **estimate_compression_results.sql.** If prompted, press **Ok** to use the default statement terminator (semi-colon).
3. Click on the **No Connection** link on top left hand corner of the script, select **CUSTINFO** connection profile and click **Finish**.

4. Now execute the script by pressing the run button in Data Studio, or press **F5** while the cursor is inside the SQL Editor.

   Note that the commands might take a few seconds to complete.

Status | Result1

	TABNAME	TYPE	ROWCOMPMODE	PCTPAGESSAVED_CURRENT	AVGROWSIZE_CURRENT	PCTPAGESSAVED_STATIC	AVGROWSIZE_STATIC	PCTPAGESSAVED_ADAPTIVE	AVGROWSIZE_ADAPTIVE
1	CODES	DATA		0	18	1	17	1	17
2	CUSTOMERS	DATA		0	2340	77	536	81	189
3	CUSTOMERS	XML		0	579	79	119	79	119
4	CUSTOMERS_ARCH	DATA		0	263	65	91	65	91
5	CUSTOMERS_ARCH	XML		0	579	79	115	79	115

Figure 147 - Estimate compression results for both relational and XML data

The result of the compression estimate run against the DB2INST1 schema shows that most of the savings are related to the CUSTOMERS table.

- 77% relational data page savings and 79% XML data page savings using static compression
- 81% relational data page savings and 79% XML data page savings using adaptive compression.

The CUSTOMERS_ARCH table has similar estimation results as it has similar data to the CUSTOMERS table but it only has ~ 10,000 records in cardinality compared to the 100,000 records in the CUSTOMERS table. The CODES table only reports 1% relational data page savings. This was expected since the table contains random numerical data and with very few repeating patterns for row compression to take much affect.

Based on these results, we will enable compression on the **CUSTOMERS** and **CUSTOMERS_ARCH** table, in the next section. CUSTOMERS_ARCH table will not have any significant benefit due to its small size; however we will still enable compression to see how its total physical size changes compared to the previously recorded results. The CODES table is left as is.

5. Next, you will estimate Index compression savings. In Data Studio, click menu **File > Open File** on the top left hand corner.
6. Navigate to **/home/db2inst1/Documents/LabScripts/Compression** directory and open the file named **idx_compression_results.sql.** If prompted, press **Ok** to use the default statement terminator (semi-colon).
7. Click on the **No Connection** link on top left hand corner of the script, select **CUSTINFO** connection profile and click **Finish**.
8. Now execute the script by pressing the run button in Data Studio, or press **F5** while the cursor is inside the SQL Editor.

Status | Result1

	INDSCHEMA	INDNAME	TABNAME	INDEX_COMPRESSED	PCT_PAGES_SAVED	NUM_LEAF_PAGES_SAVED
1	SYSIBM	SQL120508155459	CUSTOMERS	N	0	0
2	SYSIBM	SQL120508155459	CUSTOMERS	N	0	0
3	SYSIBM	SQL120508155459	CUSTOMERS	N	0	0
4	DB2INST1	NAMEINDEX	CUSTOMERS	N	32	41
5	DB2INST1	ITEMID	CUSTOMERS	N	0	0
6	SYSIBM	SQL120619131811	CUSTOMERS	N	12	135

Figure 148 - Estimate index compression

When running the table function to estimate index compression savings, we notice very little or no savings on the **ITEMID** index. One of the system-required indexes and **NAMEINDEX** both see relatively good improvement with 12%

and 32% data page savings respectively. Since Index Compression is automatically enabled with Row Compression, any new indexes will automatically be compressed. However, existing indexes need to be altered to enable compression. We will demonstrate this on the user created index; **NAMEINDEX**.

The **ITEMID** index may not be compressed well as it contains information that is not contained within the base data, and may not be added to the relational dictionary.

53.7 Enable Classic Row Compression

Formatted: Bullets and Numbering

We will now enable classic row compression as we decided in the previous section for the **CUSTOMERS** and **CUSTOMERS_ARCH** tables and the **NAMEINDEX** index. Since these objects are pre-populated with data, we will need to perform an offline reorg after we enable compression.

1. Right click on the **CUSTOMERS** table and select **Alter**.
2. In the **Properties** view, click on the **General** tab, locate the **Row compression** attribute and select **Static** from the drop down menu.

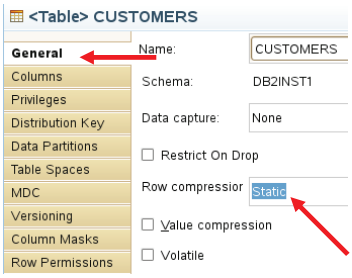


Figure 149 - Enable classic row compression on CUSTOMERS table

3. Repeat steps 1 and 2 to enable row compression on **CUSTOMERS_ARCH** table.
4. Next, in the **Administration Explorer** view, click on the **Indexes** folder and locate the **NAMEINDEX** index.
5. Right click on the **NAMEINDEX** index and select **Alter**.
6. In the **Properties** view, click on the **Performance** tab.
7. Locate the **Index compression** option and select the **Yes** from the drop down menu.

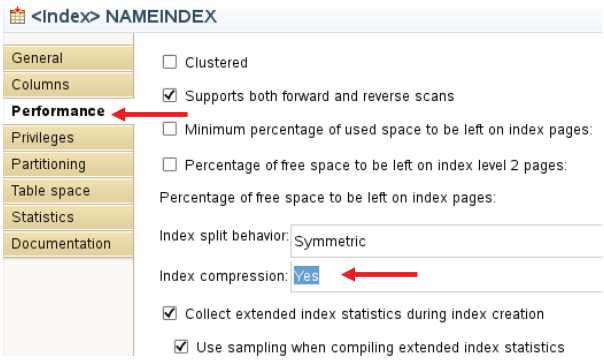


Figure 150 - Enable index compression on NAMEINDEX index.

8. Click on the **Review and Deploy (** **)** button on the top section of the screen, uncheck the **Save Data** option, click on **Refresh DDL** and select **Advanced Options**.

   Since we have a lot of data in our CUSTOMERS table, we want to perform an offline reorg on the tables to create a dictionary and compress all rows.

9. Leave only the **Generate REORG commands** option marked, then click **Finish**.

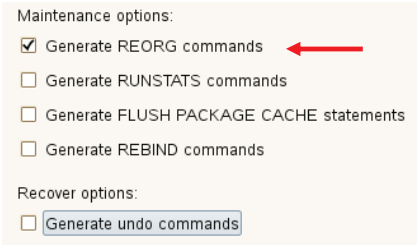


Figure 151 - Execute REORG after altering the tables' definition

Your Review and deploy popup should be similar to the following:

```
ALTER TABLE DB2INST1.CUSTOMERS_ARCH COMPRESS YES STATIC;
ALTER TABLE DB2INST1.CUSTOMERS COMPRESS YES STATIC;
ALTER INDEX DB2INST1.NAMEINDEX COMPRESS YES;
CALL SYSPROC.ADMIN_CMD( 'REORG TABLE DB2INST1.CUSTOMERS_ARCH' );
CALL SYSPROC.ADMIN_CMD( 'REORG TABLE DB2INST1.CUSTOMERS' );
```

Figure 152 - Reorg and Runstats on tables CUSTOMERS and CUSTOMERS_ARCH

10. Finally, click on **Run** to execute the SQL statements.

    All commands should report a successful status. At this point compression is enabled on both tables and the index NAMEINDEX.

Next, we will reorganize XML table data and indexes related to both **CUSTOMERS** and **CUSTOMERS_ARCH** tables

11. Right click on the **CUSTOMERS** table, select **Manage > Reorg Table.**
12. Mark the **Reorganize long field and large object data** option.

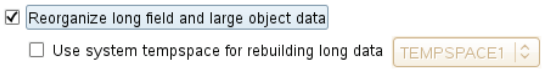


Figure 153 - Reorganize LOBs and XML

13. Click on the **Run (** **)** button on the top right section of the screen to execute the reorg operation.
    The commands might take a few minutes to complete because of the size of the table.
14. Repeat steps 11 and 12 to reorganize the XML table data on **CUSTOMERS_ARCH** table.
15. To reorganize the **ITEMID** xml index, right click on the **CUSTOMERS** table, select **Manage > Reorg Index.**
16. Use the default options and click **Run (** **).**

    Lastly, update the statistics on the **CUSTOMERS** table.
17. Right click on the **CUSTOMERS** table and select **Manage > Run Statistics**.
18. In the **Properties** view, select the **Update the existing profile** and **Update statistics now** options**.**

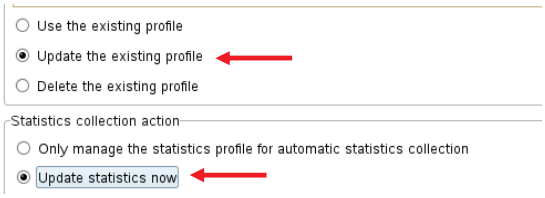


Figure 154 - Update the existing profile and statistics

19. Next, go to the **Statistics** tab and select the **Collect no statistics on all columns** and the **Collect basic statistics on all indexes** options.

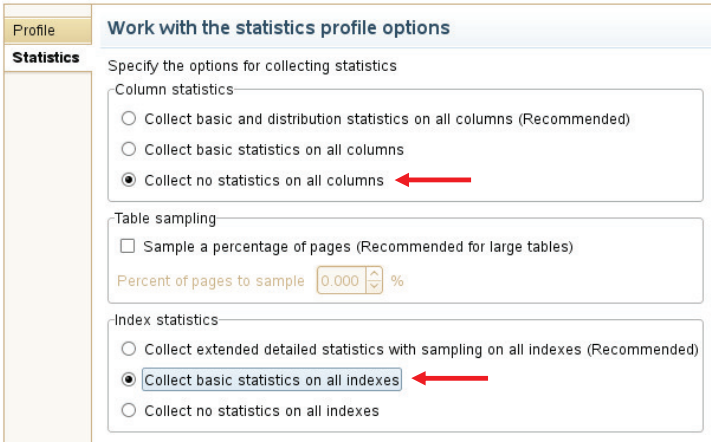


Figure 155 - Set the statistics options

20. Click on the **Run (** **)** button on the top right section of the screen to execute the runstats operation.

    The XML dictionary is now created and the XML data is compressed in the XML Data Area.

53.8 Inspecting Compression Results

Formatted: Bullets and Numbering

In this section we will use the available tools to see how our data is compressed.

Let's see what the actual savings were for data and indexes as reported by the two compression table functions.

1. Execute the previously open **estimate_compression_results.sql** script.

   Note that the command might take a few seconds to complete.'

	TABNAME	TYPE	ROWCOMPMODE	PCTPAGESSAVED_CURRENT	AVGROWSIZE_CURRENT	PCTPAGESSAVED_STATIC	AVGROWSIZE_STATIC	PCTPAGESSAVED_ADAPTIVE	AVGROWSIZE_ADAPTIVE
1	CODES	DATA		0	18	0	17	1	17
2	CUSTOMERS	DATA	S	76	548	77	527	81	188
3	CUSTOMERS	XML	S	78	127	79	118	79	118
4	CUSTOMERS_ARCH	DATA	S	63	95	66	88	66	88
5	CUSTOMERS_ARCH	XML	S	78	125	79	116	79	116

Figure 156 - Inspect compression results

You will find that executing the script after enabling compression has changed its output. You will notice that ROWCOMPMODE now has an S against the tables with compression enabled. The character 'S' represents static compression. To see the total data page savings achieved by using static compressions, refer to the column "PCTPAGESSAVED_CURRENT". It can be observed that 78% of XML data pages and 76% of relational data pages from the CUSTOMERS table were saved and 78 % and 63% data pages, respectively, from the CUSTOMERS_ARCH table.

2. Let's see how the physical sizes of the different database objects changed.

3. Execute the previously open **get_table_properties.sql** script again.

Status | Result1

	TABNAME	REL_DATA_SIZE (KB)	INDEX_SIZE (KB)	LOB_SIZE (KB)	XML_SIZE (KB)	LARGE_RIDS	LARGE_SLOTS	REL_DICT_SIZE (KB)	XML_DICT_SIZE (KB)	TOTAL_SIZE (KB)
1	CODES	1152	0	0	0	Y	Y	0	0	1152
2	CUSTOMERS	57088	15104	1024	12800	Y	Y	87	90	86193
3	CUSTOMERS_ARCH	1280	768	31616	1792	Y	Y	78	88	35622

Figure 157- Inspect table sizes

Now take a look at the physical sizes of the various database objects, including the total size of the CUSTOMERS table, the table has shrunk to nearly half the original size. The inlined LOB objects being included in the compression process further reduced the overall table size.

4. Record the column values for the three tables and compare them to your previous recordings.

Column name	CUSTOMERS	CUSTOMERS_ARCH	CODES
REL_DATA_SIZE			
INDEX_SIZE			
LOB_SIZE			
XML_SIZE			
REL_DICT_SIZE			
XML_DICT_SIZE			
TOTAL_SIZE(KB)			

Let's see how the indexes were compressed.

5. Execute the previously open **idx_compression_result.sql** script.

Status | Result1

	INDSCHEMA	INDNAME	TABNAME	INDEX_COMPRESSED	PCT_PAGES_SAVED	NUM_LEAF_PAGES_SAVED
1	SYSIBM	SQL120620164556	CUSTOMERS	N	0	0
2	SYSIBM	SQL120620164556	CUSTOMERS	N	0	0
3	SYSIBM	SQL120620164556	CUSTOMERS	N	0	0
4	DB2INST1	NAMEINDEX	CUSTOMERS	Y	42	47
5	DB2INST1	ITEMID	CUSTOMERS	N	0	0
6	SYSIBM	SQL120621095019	CUSTOMERS	N	12	135

Figure 158 - Examine index compression results

The actual index compression results are better than expected; the **NAMEINDEX** has been compressed with 42% pages saved. You can see that it has been compressed by the **INDEX_COMPRESSED** attribute having changed to **'Y'.** All other indexes remain uncompressed with estimates provided for one of the system indexes.

6. Repeat steps 17 to 20 from section 4.7 to execute the RUNSTATS command on the **CUSTOMERS** table. It will update the statistics in order to be able to leverage the performance increase in Visual Explain in the following steps.

53.9 Run Sample Workload on Compressed Data

Formatted: Bullets and Numbering

We can now run the same workload as in section 4.5 and see how the query performs after data has been compressed.

1. In the SQL editor, type in the following SQL statement and press F5 (or click on the top-right hand corner of the SQL Editor) to execute the query. If prompted, select the **CUSTINFO** connection profile and click **Finish**.

```
SELECT * FROM CUSTOMERS;
```

   At the bottom of the Data Studio screen, in the SQL Results pane, you should see a Successful result message.

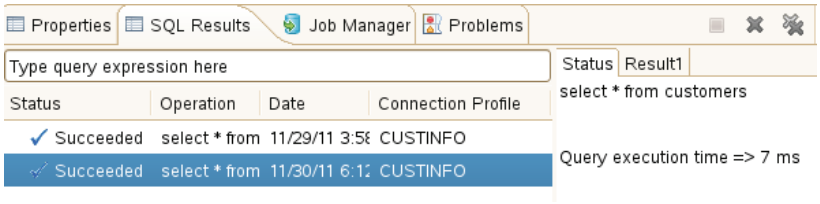


Figure 159 - Check workload result

2. Next, in the SQL Editor, click the **Open Visual Explain** button .

3. Click **Finish** in the configuration window, which pops up and wait until the utility has finished executing. Finally, expand the bottom pane of Data Studio to see the results.

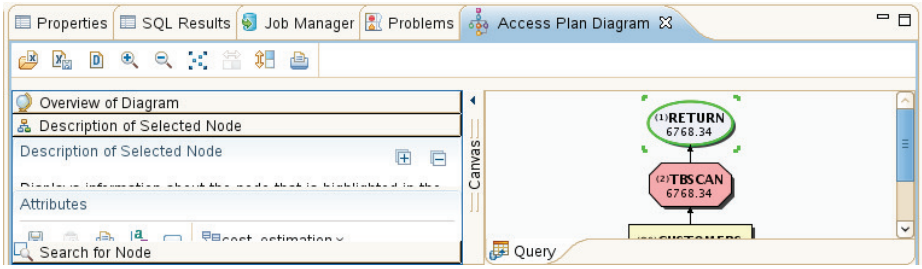


Figure 160 - Expand Visual Explain pane

Note that the time it takes DB2 to perform the workload is now close to 6768.34 timerons compared to the 33722.1 timerons as reported by Visual Explain before classic row compression was enabled on the table. Not only did we save on storage by enabling compression, but we also improved the performance of the query.

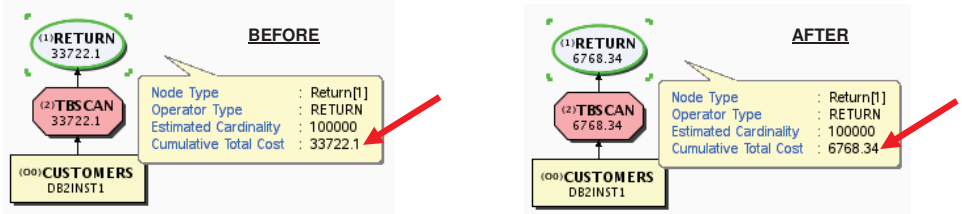


Figure 161 Comparative cumulative total cost BEFORE and AFTER classic compression

53.10 Applying Adaptive Compression

Formatted: Bullets and Numbering

DB2 10 has further improved on classic row compression by introducing a new hybrid compression technique, referred to as adaptive compression. Adaptive compression offers the most dramatic possibilities for storage savings by leveraging classic row compression and also working on a page-by-page basis to further compress data.

Up until now, you have been using static compression in order to examine performance and storage benefits. You will now apply adaptive compression. This section of the lab will have you upgrade a specific table from using static compression to adaptive compression, and analyze the results afterwards.

1. Execute the previously open **get_table_properties.sql** script to inspect the current **CUSTOMER** table properties.

Status | Result1

	TABNAME	REL_DATA_SIZE (KB)	INDEX_SIZE (KB)	LOB_SIZE (KB)	XML_SIZE (KB)	LARGE_RIDS	LARGE_SLOTS	REL_DICT_SIZE (KB)	XML_DICT_SIZE (KB)	TOTAL_SIZE (KB)
1	CODES	1152	0	0	0	Y	Y	0	0	1152
2	CUSTOMERS	57088	15104	1024	12800	Y	Y	87	90	86193
3	CUSTOMERS_ARCH	1280	768	31616	1792	Y	Y	78	88	35622

Figure 162 - Inspect CUSTOMER table properties

Take note of the CUSTOMERS table relational data size and total size.

Column name	Value
REL_DATA_SIZE (KB)	
TOTAL_SIZE (KB)	

2. Next, check the estimated savings on the current **CUSTOMERS** table.
3. Using steps 1 to 5 from section 4.3, to Navigate to **/home/db2inst1/Documents/LabScripts/Compression** directory, open and execute the file named **compare_static_adaptive.sql.**

Status | Result1

	1	OBJECT_TYPE	ROWCOMPMODE	PCTPAGESSAVED_CURRENT	AVGROWSIZE_CURRENT	PCTPAGESSAVED_STATIC	AVGROWSIZE_STATIC	PCTPAGESSAVED_ADAPTIVE
1	CUSTOMERS	DATA	S	76	548	77	527	81
2	CUSTOMERS	XML	S	78	127	79	118	79

Figure 163 - Estimated adaptive compression savings in CUSTOMERS table

The current savings showcase the savings achieved using static compression. Expand the screen to properly view the output.

You will notice that the estimated **PCTPAGESSAVED_ADAPTIVE** is higher than the current static compression values. This is due to the fact that adaptive compression uses two compression approaches. It first uses the same table-level compression dictionary used in classic row compression to compress data based on repetition within a sampling of data from the table as a whole. The second approach uses a page-level dictionary-based compression algorithm to compress data based on data repetition within each page of data.

4. Take note of the columns PCTPAGESSAVED_CURRENT, and AVGROWSIZE_CURRENT.

Column name	Value
PCTPAGESSAVED_CURRENT	
AVGROWSIZE_CURRENT	

Next, apply adaptive compression on the **CUSTOMERS** table.

5. Right click on the **CUSTOMERS** table and select **Alter.**
6. In the **Properties** view, click on the **General** tab and select the **Adaptive** row compression method from the drop down menu.

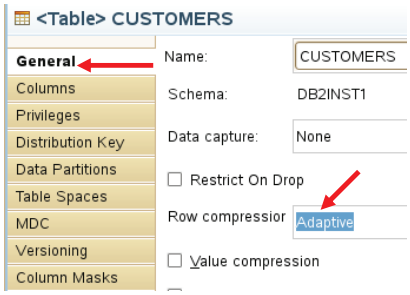


Figure 164 - Enable adaptive row compression on CUSTOMERS table

7. Now click on the **Review and Deploy (** **)** button on the top right section of the screen, uncheck the **Save Data** option and press **Refresh DDL**.
8. Next, click on **Advanced Options**, uncheck all maintenance options but **Generate REORG commands** and press **Finish.**

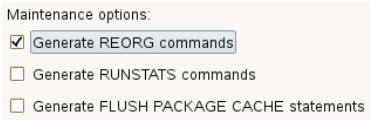


Figure 165 - Mark only the reorg command

9. Finally, press **Run** to execute the SQL script.

The last step is to run the Runstats command to keep the statistics updated for the **CUSTOMERS** table.

10. Right click on the **CUSTOMERS** table, select **Manage > Run Statistics.**
11. Mark both **Update the existing profile** and **Update statistics now** options and go to the **Statistics** tab**.**

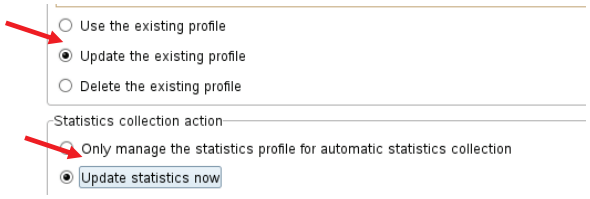


Figure 166 - Configure runstats operation

12. Select **Collect no statistics on all columns** and **Collect basic statistics on all indexes** options**.**

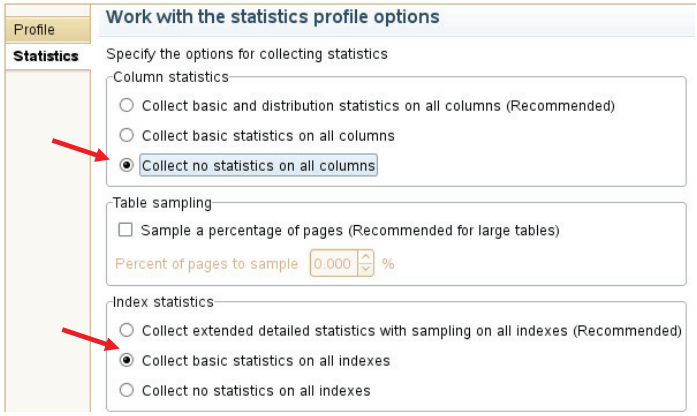


Figure 167 - RUNSTATS on table CUSTOMERS

13. Click on the **Run (** **)** button on the top section of the screen to execute the runstats operation.
14. Then, using the previously open **get_table_properties.sql** script, inspect the table properties to check the new results after applying adaptive compression.

Status | Result1

	TABNAME	REL_DATA_SIZE (KB)	INDEX_SIZE (KB)	LOB_SIZE (KB)	XML_SIZE (KB)	LARGE_RIDS	LARGE_SLOTS	REL_DICT_SIZE (KB)	XML_DICT_SIZE (KB)	TOTAL_SIZE (KB)
1	CODES	1152	0	0	0	Y	Y	0	0	1152
2	CUSTOMERS	55808	15104	1024	12800	Y	Y	87	90	84913
3	CUSTOMERS_ARCH	1280	768	31616	1792	Y	Y	78	88	35622

Figure 168 - Inspect table properties after applying adaptive compression

Notice that the relational data size has been further compressed, thereby promoting a reduction in the total size of the table.

15. Finally, using the previously open **compare_statis_adaptive.sql** script, inspect the savings after applying adaptive compression.

Status | Result1

	1	OBJECT_TYPE	ROWCOMPMODE	PCTPAGESSAVED_CURRENT	AVGROWSIZE_CURRENT	PCTPAGESSAVED_STATIC	AVGROWSIZE_STATIC	PCTPAGESSAVED_ADAPTIVE
1	CUSTOMERS	DATA	A	86	134	77	535	81
2	CUSTOMERS	XML	S	78	127	79	118	79

Figure 169 - Compare static compression vs. adaptive compression

Notice there are additional savings in the amount of pages stored, as well as a reduction of the average row size. Performing adaptive compression results in 86% page savings, more than was estimated. Performing Adaptive compression led to 10% increase in savings when compared to the earlier results from static compression.

Now, inspect the performance benefits of adaptive compression.

16. In the SQL editor, type in the following SQL statement and press F5 (or click  on the top-right hand corner of the SQL Editor) to execute the query. If prompted, select the **CUSTINFO** connection profile and click **Finish**.

```
SELECT * FROM CUSTOMERS;
```

17. Next, in the SQL Editor, click on the **Open Visual Explain** button  .

18. Click Finish in the configuration window, which pops up and wait until the utility has finished executing. Finally, expand the bottom pane of Data Studio to see the results.

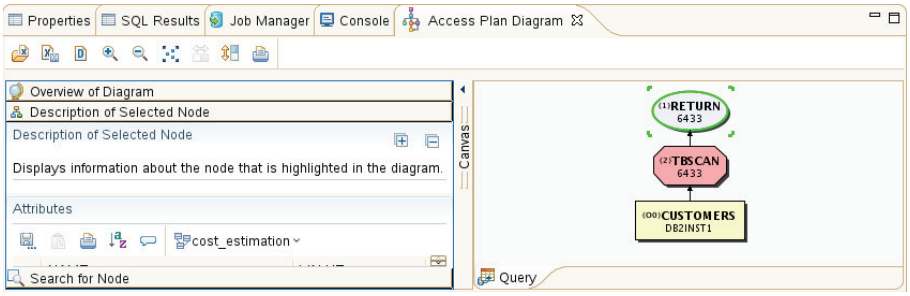


Figure 170 - Expand Visual Explain pane

Note that the time it takes DB2 to perform the workload is now close to 6433 timerons compared to the 33722.1 and 6768.34 timerons as reported by Visual Explain with no compression techniques, and classic row compression respectively. Not only did we save on storage but we also improved the speed of the query by enabling adaptive compression, thereby illustrating the performance advantages of this new technology.

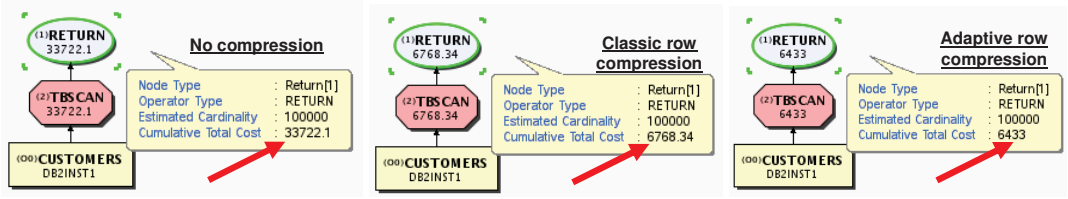


Figure 171 – Comparative cost between no compression, classic row compression and adaptive compression

54 Summary

Formatted: Bullets and Numbering

This concludes the DB2 Storage Optimization lab. In this lab, we performed several operations including demonstrating various compression techniques that DB2 employs, as well as compressing different data types including relational, xml, large object, and index data. We examined the compression and performance savings by using various tools and built-in DB2 functions that perform detailed reporting such as the Visual Explain tool and the ADMIN_GET_TAB_COMPRESS_INFO function. Finally, we also had a look at the new adaptive compression feature in DB2 10 and the additional storage and performance savings that can be achieved through its use.

With industry-leading compression technologies, DB2 not only allows greater savings in storage cost, it also improves performance as queries on a compressed table needs fewer I/O operations to access the same amount of data.

55 Cleanup

Formatted: Bullets and Numbering

1. Right-click on CUSTINFO database from within the Data Source Explorer and select Disconnect.

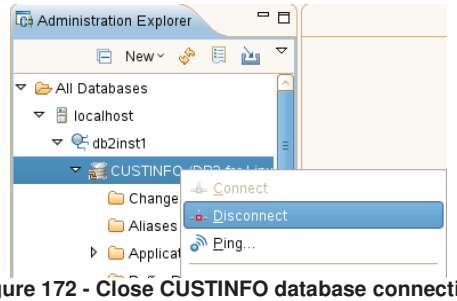


Figure 172 - Close CUSTINFO database connection

1. Close Data Studio and switch back to the terminal window that you have been using so far, and then execute the command below to clean your environment after completing the exercise. Use 'password' as password.

```
su -
cd /home/db2inst1/Documents/LabScripts/Compression/setup
./cleanup.sh
```

2. If you would like to redo this lab in the future, please execute the following commands in a terminal window as root:

```
cd /home/db2inst1/Documents/LabScripts/Compression/setup
./config.sh
```

IBM Canada
8200 Warden Avenue
Markham, ON
L6G 1C7
Canada

DB2® Security

Hands-On Lab

Table of Contents

83 Introduction

Database security is of utmost importance today. Your database may allow customers to purchase products over the Internet, or it may contain historical data used to predict business trends. In order to protect your data, it is crucial to control who has access to your database and limit the operations that the user can perform on the data.

A database security plan should define:

- Who is allowed access to the instance and/or database
- Where and how a user's password will be verified
- The authority level that a user is granted
- The commands that a user is allowed to run
- The data that a user is allowed to read and/or alter
- The database objects a user is allowed to create, alter, and/or drop

84 Suggested Reading

DB2 Security and Compliance Solutions for Linux, UNIX, and Windows (Redbook)
http://www.redbooks.ibm.com/abstracts/sg247555.html

DB2 Best Practices: IBM Data Server Security
http://www.ibm.com/developerworks/data/bestpractices/security/

85 About this Lab

In this lab, you will learn how to control access to the instance, how to control access to the database itself, and finally how to control access to the data and data objects within the database.

By the end of this lab, you will be able to:

- Grant and revoke authorities to/from users
- Grant and revoke privileges to/from users
- Create roles
- Grant and revoke roles to/from users

86 Environment Setup Requirements

To complete this lab you will require the following:

- DB2 10 VMware® image
- VMware Workstation 6.5 or later

87 Basic Setup

1. Start the VMware image by clicking the 

button in VMware Workstation.
2. At the login prompt, log in with the following credentials:

- Username: **db2inst1**
- Password: **password**

3. Open a terminal window by right-clicking on the Desktop and choosing the **Open in Terminal** item:

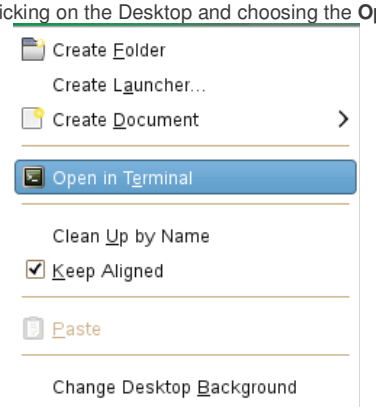


Figure 237 - Open a new Terminal window

4. Start the Database Manager by issuing the following command:

```
db2start
```

88 Authentication

When you first attempt to access an instance or database, the authentication system will try to determine if you are who you say you are. By default, DB2 works closely with the authentication mechanism of the underlying operating system to verify your user IDs and passwords. DB2 can also use third-party authentication facilities such as Kerberos or LDAP server to authenticate users.

By using an external authentication system, DB2 does not need to store a redundant set of passwords and sensitive credentials. This minimizes security vulnerabilities and hacker attacks.

88.1 Where Does Authentication Take Place?

Authentication type defines where and how authentication will take place. This is specified by the AUTHENTICATION parameter in the database manager configuration file on the server. Thus all the databases in an instance will have the same authentication type. On the client, the authentication type is specified when a remote database is cataloged.

Authentication Type	Description
SERVER	All authentications take place at the server. When you connect to a database, you will have to include your user ID and password. This information will then be verified against the credentials at the server's operating system.
SERVER_ENCRYPT	This is similar to SERVER authentication type where authentication occurs at the server, but the password is encrypted by DB2 at the client before it is sent to the server for authentication.
CLIENT	Authentication occurs at the client's operating system.

KERBEROS	Authentication occurs at the server and is handled by a Kerberos server. The KERBEROS authentication type is available if both the DB2 server and client operating systems support Kerberos. The Kerberos security protocol uses conventional cryptography to create a shared secret key which becomes the credentials used to verify the identity of the user. This eliminates the need to pass a user ID and password across the network.
KRB_SERVER_ENCRYPT	This authentication type is the same as KEBREOS, except it will use SERVER_ENCRYPT if the client does not support Kerberos security system. If none of these options are available, the client will receive a connection error and will not be able to connect.
DATA_ENCRYPT	Authentication occurs at the server and its behavior is similar to SERVER_ENCRYPT. In this type of authentication, not only is the password encrypted, but also all user data is encrypted during transmission between the client and the server.
DATA_ENCRYPT_CMP	This type of authentication is identical to DATA_ENCRYPT. However, this setting provides compatibility to those clients who do not support DATA_ENCRYPT authentication and will instead connect using SERVER_ENCRYPT so user data will not be encrypted.
GSSPLUGIN	Authentication occurs at the server using an external GSS-API plug-in. If the client's authentication type is not specified, the server will send a list of server-supported plug-ins to the client. These plug-ins are listed in the SRVCON_GSSPLUGIN_LIST database manager configuration parameter. The client then selects the first plug-in found in the client plug-in directory from the list. If the client does not support any plug-in in the list, the client is authenticated using the KERBEROS authentication method.
GSS_SERVER_ENCRYPT	Authentication occurs at the server using either the GSSPLUGIN or the SERVER_ENCRYPT authentication method. Authentication uses a GSS-API plug-in and if the client does not support any of the plug-ins found in the server-supported plug-ins list, the client is authenticated using KERBEROS. If the client does not support the Kerberos security protocol, the client is authenticated using the SERVER_ENCRYPT authentication method.

88.2 Specifying Authentication Type

In this section, you will experience how authentication is easily managed through Data Studio.

1. Double-click the IBM Data Studio icon on the Desktop.

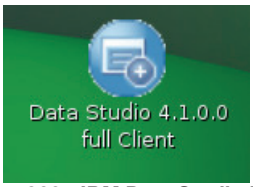


Figure 238 - IBM Data Studio icon

2. Click **OK** to accept the default workspace.

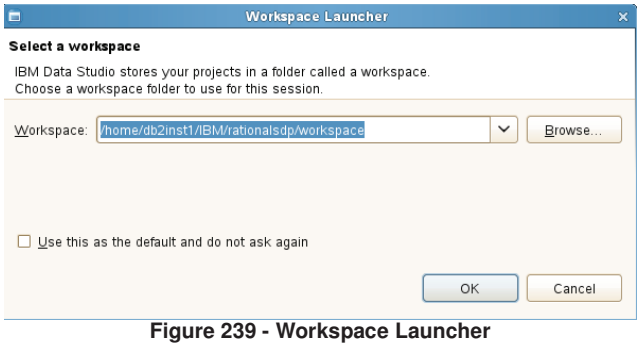


Figure 239 - Workspace Launcher

3. Locate the **Administration Explorer** view. Expand **All Databases > localhost > db2inst1**. Right-click on the **SECLAB** database and select **Connect**.

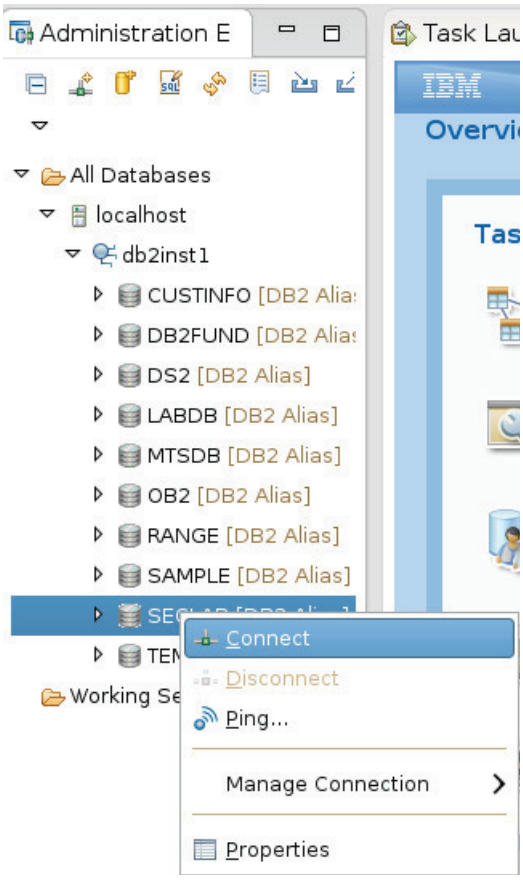


Figure 240 – Connect to SECLAB database

4. Login with the following credentials:
    - **User:** db2inst1
    - **Password:** password

5. After a connection has been established, right click on **DB2INST1** and select **Configure**.

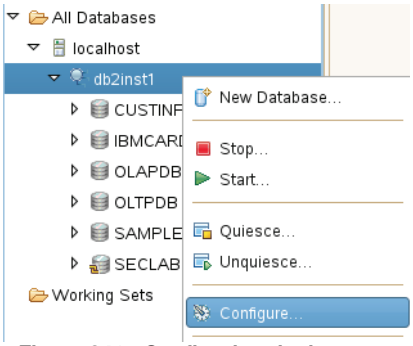


Figure 241 - Configuring the instance

6. If the connection profile popup appears, select **SECLAB** and click **OK**.

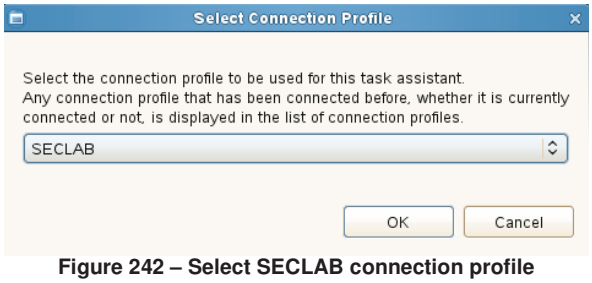


Figure 242 – Select SECLAB connection profile

7. Select the Instance Tab. Locate the **AUTHENTICATION** parameter. Double-click the correspondent **Pending Value** field and select the **SERVER_ENCRYPT** authentication. Notice you can not use the **Immediate** option, therefore you must restart the instance after changing the authentication type of the **db2inst1** instance.

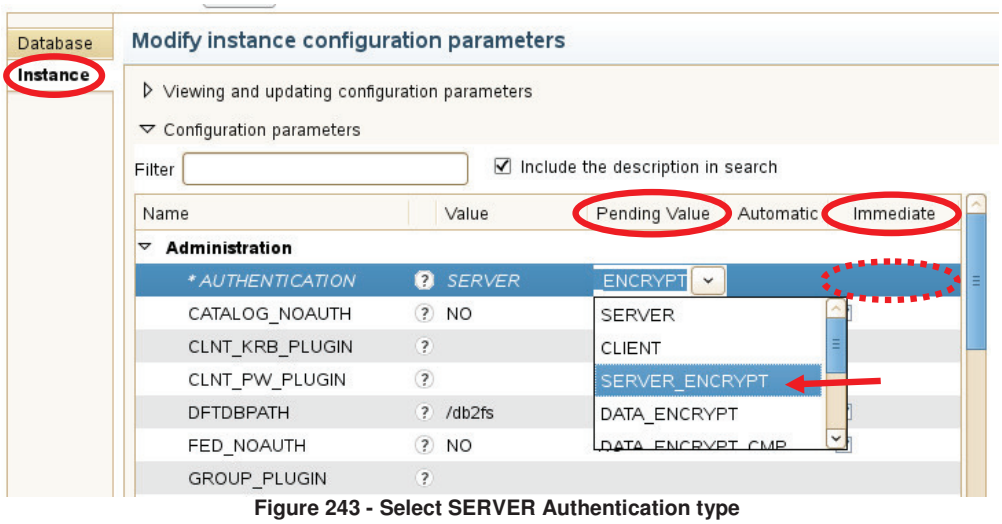


Figure 243 - Select SERVER Authentication type

8. Click on Run button to apply the changes. Check the execution status on the SQL Results tab.

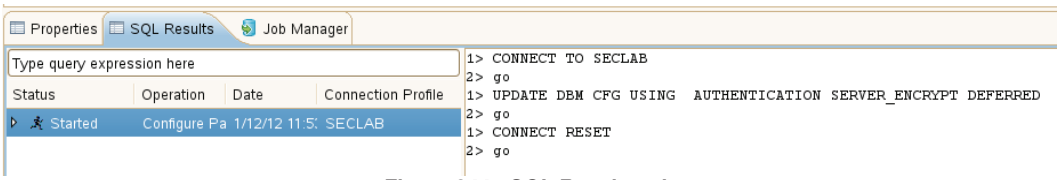


Figure 244 - SQL Results tab

9. Click to close the **Configure Parameters** window.
10. Right-click on **db2inst1** and select **Stop**. A new **Stop Instance db2inst1** tab will open.

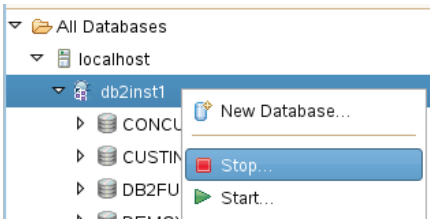


Figure 245 - Stop db2inst1 instance

11. Use the following credentials when prompted and click **Finish**.
    - **User:** db2inst1
    - **Password:** password

12. Select the first option **(force the applications off)**, then click Run to execute the db2stop command. Check the **SQL Results** tab for errors. If a warning dialog appears to inform you that databases, such as SECLAB, have been disconnected, click OK to close the dialog.

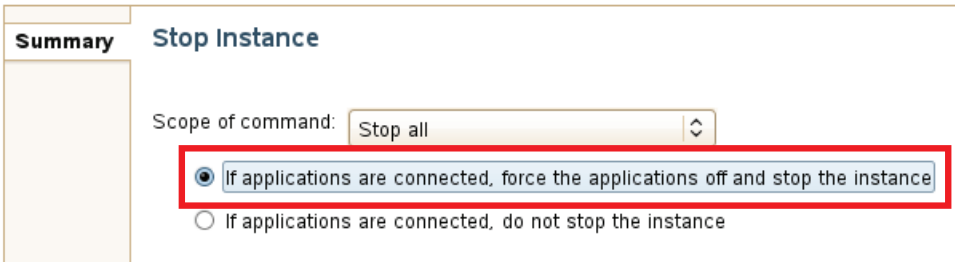


Figure 246 - Stop db2inst1 instance

13. Now start the db2inst1 instance again. Right-click on **db2inst1** and select **Start**.

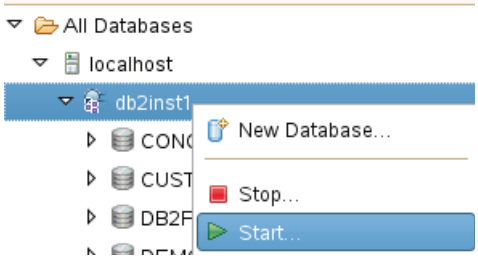


Figure 247 - Start db2inst1 instance

14. Use the same credentials to execute the command and click **Finish**.
    - **User:** db2inst1
    - **Password:** password

15. Click Run to execute the **db2start** command. Check the **SQL Results** tab to confirm the command was executed with success.

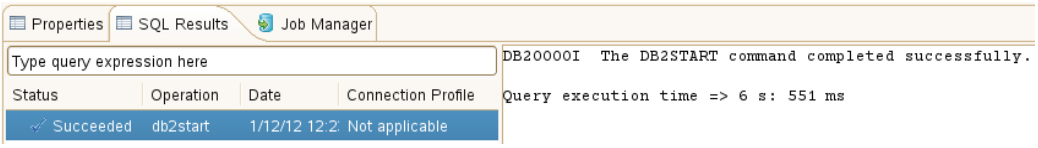


Figure 248 - db2isnt1 instance restarted

16. Now you must reconnect to the SECLAB database. Just like before, right-click the **SECLAB** database and select **Connect**. Login with the following credentials:
    - **User:** db2inst1
    - **Password:** password

17. After a connection has been established, right click on **DB2INST1** and select **Configure**.

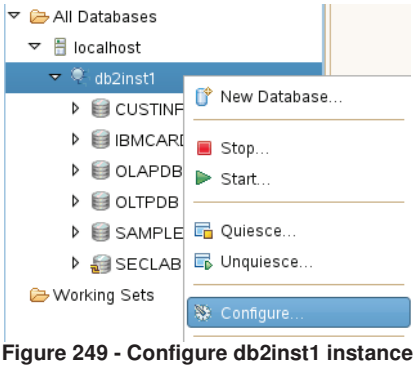


Figure 249 - Configure db2inst1 instance

18. If the connection profile popup appears, select **SECLAB** and click **OK**.

19. Select the Instance Tab. Locate the **AUTHENTICATION** parameter. Notice the authentication value has changed to **SERVER_ENCRYPT**.

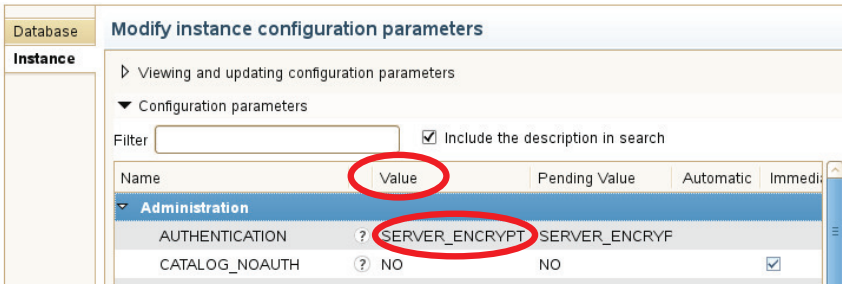


Figure 250 - SERVER_ENCRYPT authentication is now in use

89 Authorization

After a user has been authenticated, authorization serves as the second security mechanism which determines what operations a user can perform, and which data objects that user can access within a database or instance. The set of permissions granted to users can come from privileges, authorities, roles, user groups and label-based access control (LBAC) credentials.

Administrative authorities group together certain privileges so that users can perform functions at the database or instance level.

A privilege is a permission to perform an action or task. For instance, privileges define the objects that a user can create or drop and commands that a user can use to access objects like tables, views, indexes, and packages.

Roles can group together a set of privileges, so these can be assigned to users through the role instead of individually.

LBAC uses security labels to control who has read/write access to individual rows and columns of a table.

Note: LBAC capabilities are not covered in this hands-on lab.

89.1 Authorities

Authorities are needed for managing databases and instances and can be divided into two groups:

- System-level authorities
- Database-level authorities

89.1.1 System-level Authorities

System level authorities enable you to perform instance-wide functions, such as creating and upgrading databases, managing table spaces, and monitoring activity and performance on your instance. These authorities are granted through the database manager configuration and can only be assigned to user groups defined at the operating system level.

Below you will find the list of system-level authorities and their descriptions.

Database-level Authority	Description

SYSADM	for users managing the instance as a whole
SYSCTRL	for users administering a database manager instance
SYSMAINT	for users maintaining databases within an instance
SYSMON	for users monitoring the instance and its databases

By default, on Linux systems, the SYSADM group is set to the primary group of the instance owner: DB2GRP1. On Windows, members of the local Administrators group are all granted SYSADM authority.

1. Using the **Configure Parameters** tab previously open, type **SYSADM_GROUP** in the **filter** text field. As soon as you start typing, you will notice that Data Studio dynamically filters the parameters. You should see the **SYSADM_GROUP** parameter below the **filter** text box.

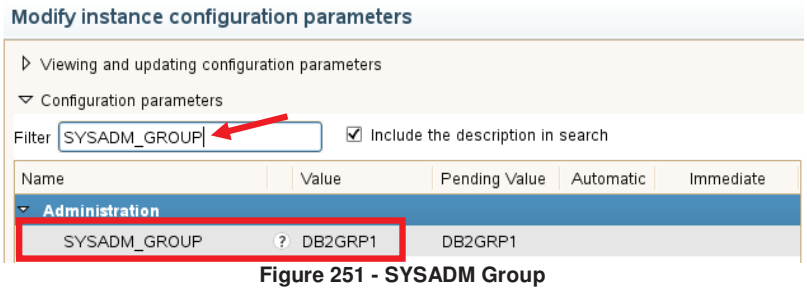


Figure 251 - SYSADM Group

2. Click on the 'X' to close the "Configure Parameters" tab.

Figure 252 - Close the "Configure Parameters" tab

89.1.2 Database-level Authorities

Database level authorities enable you to perform functions within a specific database, such as granting and revoking privileges, inserting, selecting, deleting and updating data, and managing workloads.

The database-level authorities are: **ACCESSCTRL, BINDADD, CONNECT, CREATETAB, CREATE_EXTERNAL_ROUTINE, CREATE_NOT_FENCED_ROUTINE, DATAACCESS, DBADM, EXPLAIN, IMPLICIT_SCHEMA, LOAD, QUIESCE_CONNECT, SECADM, SQLADM, WLMADM**.

89.2 Privileges

A privilege is a permission to perform an action or task. They can be obtained in three different ways:

- **Explicitly:** These are the privileges explicitly granted to a user/group/role using the GRANT command.

- **Implicitly:** When a user creates a database object, that user will implicitly receive all privileges for that object. For example, when a user creates a database, that user implicitly receives DBADM authority for that database.
- **Indirectly:** An indirect privilege is usually associated with a package. When a user is given permission to execute a package, the user may not have privileges to access the objects used by the package. Therefore, the user will be indirectly given these privileges temporary in order to execute the package.

89.3 Granting and Revoking Authorities and Privileges

So far, you have been issuing all database commands as the instance administrator (db2inst1) which has privileges to access all the utilities, data, and database objects within DB2. It is important that users be only given privileges that are necessary to complete their tasks.

In the following scenario, a new member (USERDEV) has joined your team. We will look at how to assign specific authorities and privileges to him to safeguard the security of the database.

USERDEV is an application developer and as such he requires SELECT, INSERT, UPDATE, and DELETE privileges to access the various tables in the database. USERDEV should also be able to add new packages to the database and execute them; therefore, he needs BINDADD authority.

89.3.1 Adding a new user

1. Open a terminal window by right-clicking on the Desktop and choosing the **Open in Terminal** item:

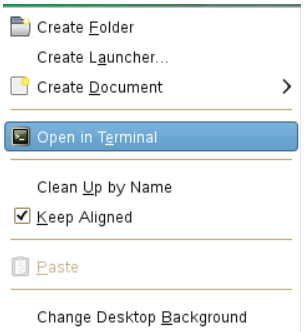


Figure 253 - Open a new terminal window

   As mentioned before, DB2 relies on the OS authentication mechanism, so we need to create the new users and groups at the operating system level.

2. In the terminal window, execute the commands below to: 1) Log in as **root** user (use '**password**' as password); 2) Add a new user called **USERDEV**; 3) Change USERDEV's password to '**password**'.

```
su -
useradd userdev
```

```
passwd userdev
exit
```

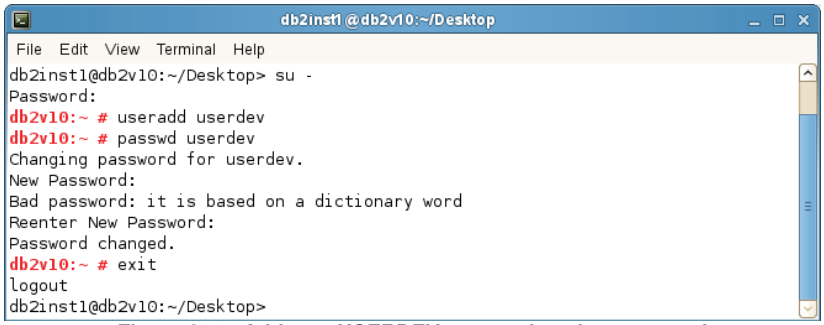


Figure 254 - Add new USERDEV user and set its password

When the new user is added, this user he has no authorities or privileges other than those defined in the PUBLIC group.

3. Try to connect to **SECLAB** database as **USERDEV**.

```
db2 CONNECT TO SECLAB USER userdev USING password
```

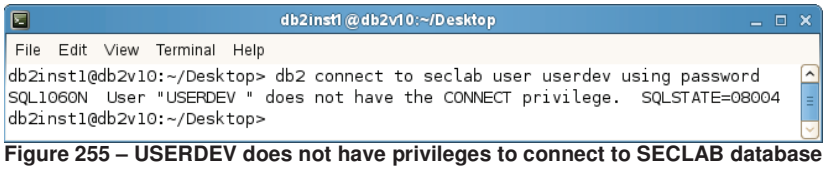


Figure 255 – USERDEV does not have privileges to connect to SECLAB database

89.3.2 Granting Database Privileges

1. Switch to Data Studio. In the Administration Explorer, expand **SECLAB > Users and Groups**. Right-click on **Users** and select Add and Manage User

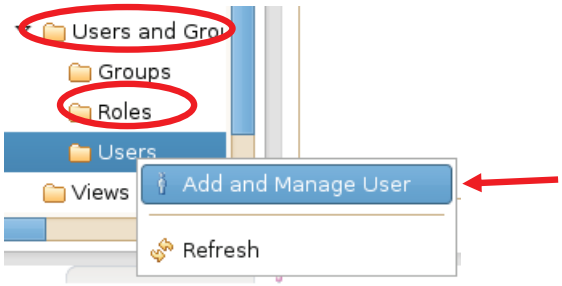


Figure 256 - Create a new database user

2. In the Properties view, input **userdev** into the Name field and select the **Privileges** option.

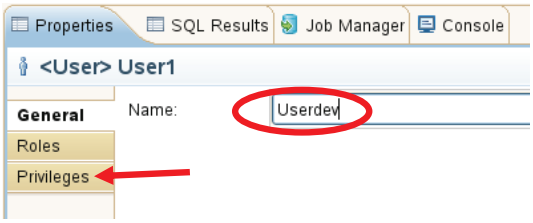


Figure 257 - Define user name

3. The **Connect** should be already selected. Check the **BINDADD** and **CREATETAB** privileges as well.

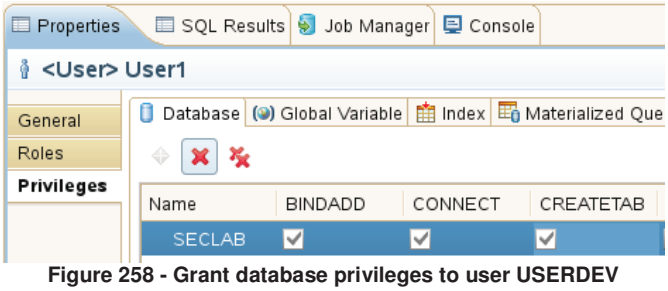


Figure 258 - Grant database privileges to user USERDEV

4. Next, locate the toolbar on the top right corner () and click on the **Review and Deploy Changes** button (). Ignore the warnings and click **OK**.

   The **Review and Deploy** screen will display the commands that were generated to apply the desired changes.

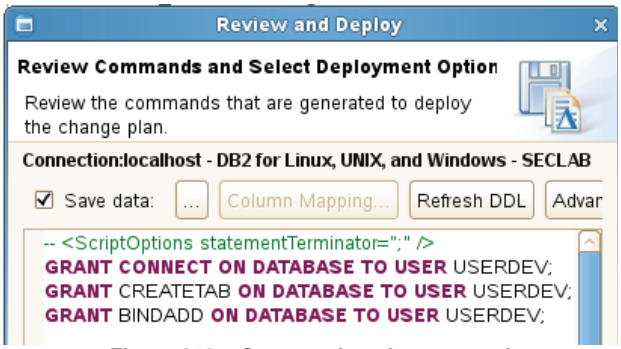


Figure 259 – Commands to be executed

5. Select the **Run** radio option and click **Finish**.

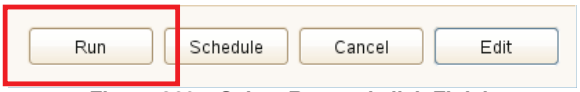


Figure 260 – Select Run and click Finish

6. Check the **SQL Results** tab to ensure the commands were executed without errors.

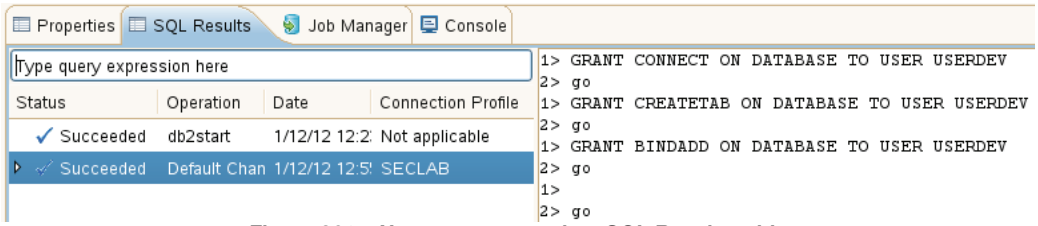


Figure 261 – No errors reported on SQL Results table

USERDEV has now privileges to connect, bind a package and create tables in the **SECLAB** database.

7. Switch to the previously open terminal window. Try again to connect to **SECLAB** database as **USERDEV**.

```
db2 CONNECT TO SECLAB USER userdev USING password
```

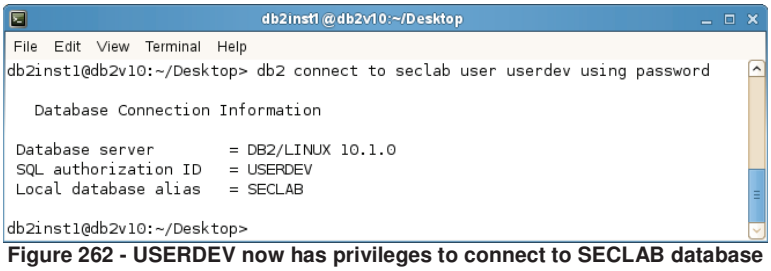


Figure 262 - USERDEV now has privileges to connect to SECLAB database

89.3.3 Object-level Authorities

USERDEV must be able to query and modify the table EMPLOYEE.

1. In the same terminal window, try to query the **EMPLOYEE** table. You will notice that USERDEV does not have privileges on **EMPLOYEE** table.

```
db2 "SELECT * FROM DB2INST1.EMPLOYEE"
```

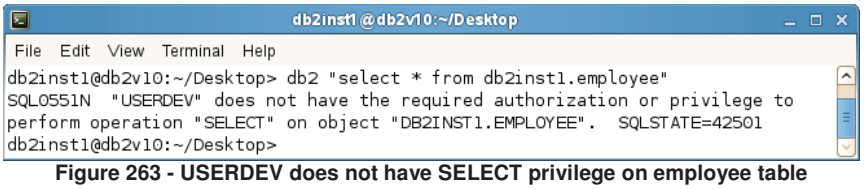


Figure 263 - USERDEV does not have SELECT privilege on employee table

2. Go back to Data Studio and locate the **SECLAB** database on the **Administration Explorer** view. Select the **Tables** folder. Data Studio will list all SECLAB tables into the main view. Locate the **Filter** text box and type **EMPLOYEE** into the text box. Notice that Data Studio's dynamic filter will display the EMPLOYEE table.

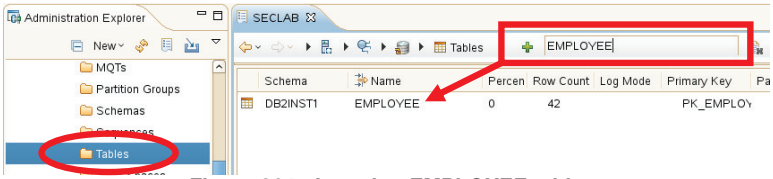


Figure 264 - Locating EMPLOYEE table

3. Right click on **EMPLOYEE** table and select the **Manage Privileges** option.

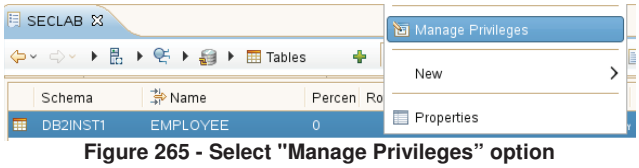


Figure 265 - Select "Manage Privileges" option

4. Locate the **USERDEV** and tick the **SELECT**, **INSERT**, **UPDATE** and **DELETE** privileges.

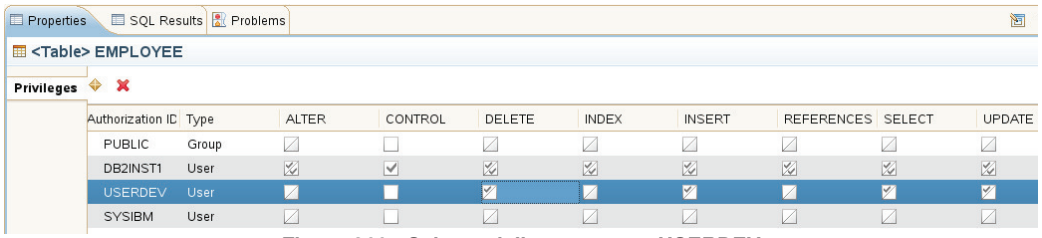


Figure 266 - Select privileges to user USERDEV

Note: a single tick means the user has that privilege. Double-tick means the user has that privilege with the ability to GRANT the privilege to other users (WITH GRANT OPTION).

5. Next, locate the toolbar on the top right corner ( ) and click on the **Review and Deploy Changes** button ( ). Again, ignore the warnings and click **OK**.

   The Review and Deploy screen will display the commands that were generated to apply the desired changes.

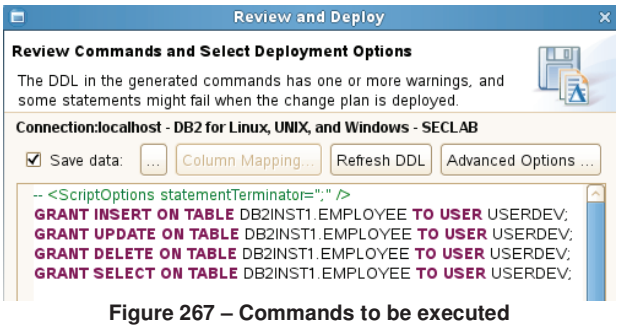


Figure 267 – Commands to be executed

6. Select the **Run** option and click the **Finish** button.

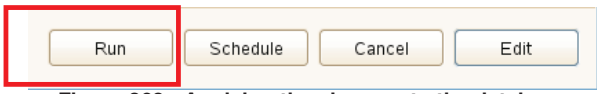


Figure 268 - Applying the changes to the database

7. Finally, check the **SQL Results** tab to ensure the commands were executed without errors.

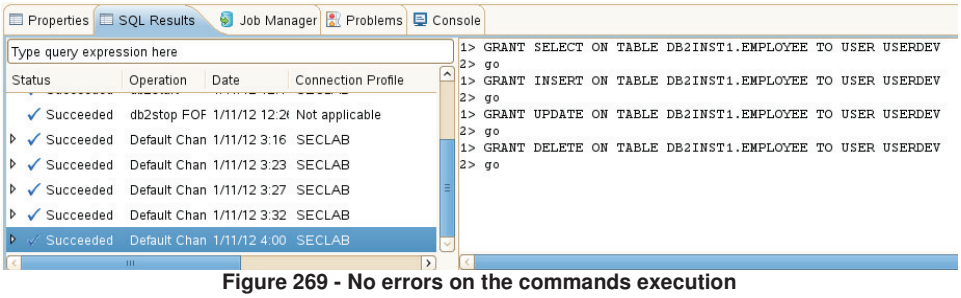


Figure 269 - No errors on the commands execution

USERDEV now has the privilege to interact with the SECLAB database.

8. Switch back to the terminal and try the same query again user **USERDEV**. You will notice that the operation will now succeed.

```
db2 "SELECT * FROM DB2INST1.EMPLOYEE"
```

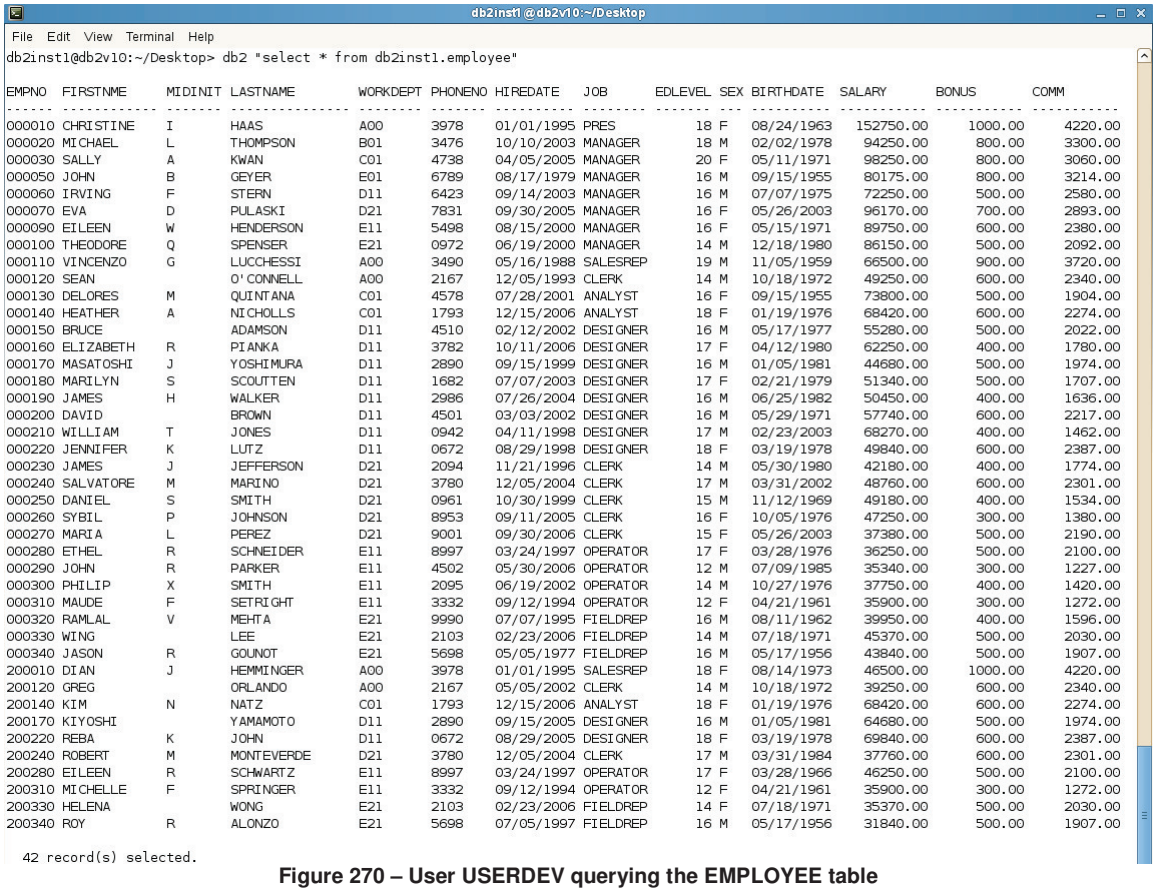


Figure 270 – User USERDEV querying the EMPLOYEE table

89.4 Using Views to Define Granular Privileges

There are three ways in which access to specific portions of data in a table can be restricted: views, Label-Based Access Control (LBAC) and the new Row and Column Access Control (RCAC). The use of LBAC and RCAC is beyond the scope of this lab, so we will instead focus on the implementation of views.

Views are virtual tables (computed dynamically and not stored explicitly) that are derived from one or more tables or views. They can be used to provide a customized subset of data to the users, allowing them to see different presentations of the same set of data or hide data to which a user should not have access. Views can perform delete, insert, update operations or be read-only.

The strategy to provide granular data access using views is as follows:

- Create a view that returns the portion of the data that the user should have access to.
- Grant the user access to the view.
- Revoke from the user access to the base table.

89.4.1 Creating a View

We would like to create a view that returns a directory of only those who are in department E11. This view will contain only first name, last name, phone number, and job role.

1. Switch back to Data Studio, right-click the **Views** folder and select **Create View** option.

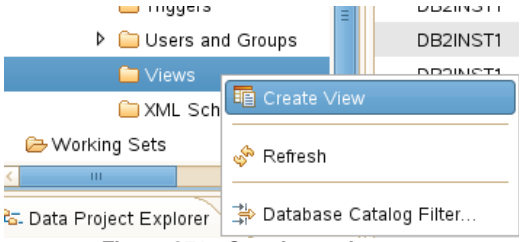


Figure 271 - Creating a view

2. Select the **DB2INST1** schema and click **OK**.

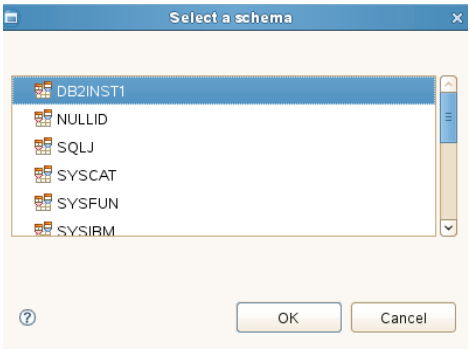


Figure 272 – Select DB2INST1 schema

The new view will be added at the bottom of the list and you will be able to define its definitions using the **Properties** view. Select the General tab, enter **E11INFO**, and then select the **SQL** tab.

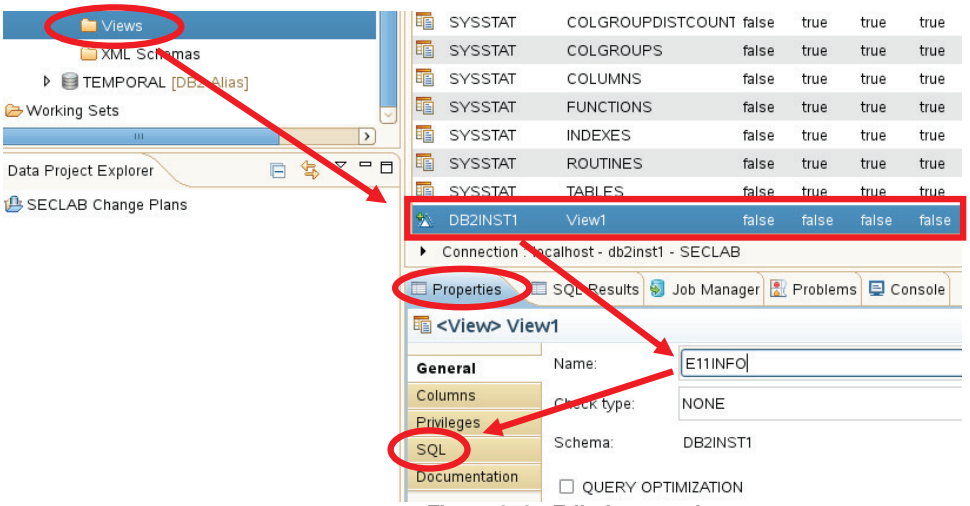


Figure 273 - Edit the new view

3. Type the following SELECT statement to the **Expression** field, and then press the **Update** button to check the syntax.

```
SELECT FIRSTNME, LASTNAME, PHONENO, JOB FROM EMPLOYEE WHERE WORKDEPT='E11'
```

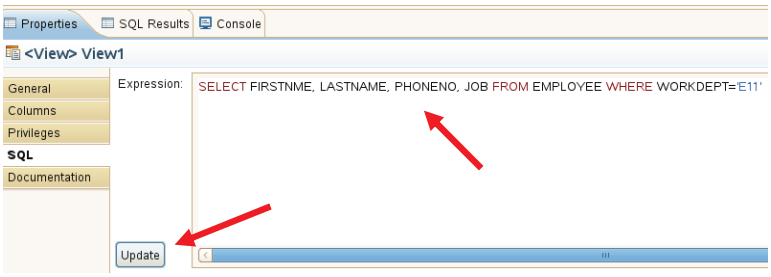


Figure 274 - Create E11INFO view

4. Next, locate the toolbar on the top right corner () and click on the **Review and Deploy Changes** button (). The **Review and Deploy** screen will display the commands generated to reflect the desired changes.

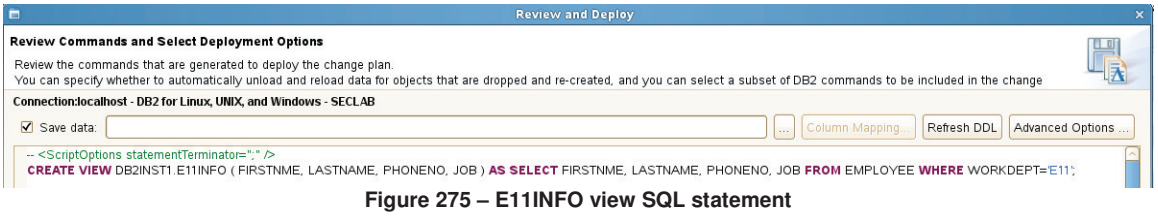


Figure 275 – E11INFO view SQL statement

5. Select the **Run** option and click the **Finish** button.

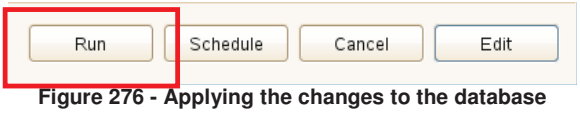


Figure 276 - Applying the changes to the database

6. Finally, check the **SQL Results** tab to ensure the commands were executed without errors.

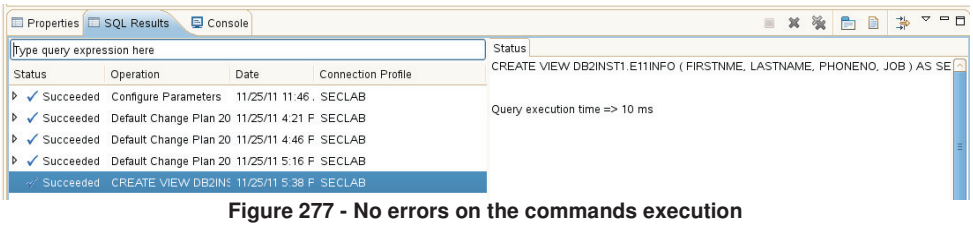


Figure 277 - No errors on the commands execution

The view **E11INFO** is now created and a user issuing a select statement against the view will see only the four columns defined.

7. In **Administration Explorer**, on the menu bar select **New SQL Script**:

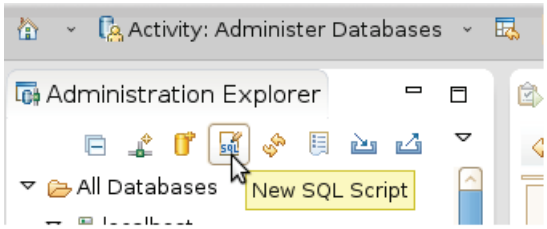


Figure 278 - Create a new SQL script

8. In the editor, enter the SQL statement below and press to execute the statement.

```
select * from e11info;
```

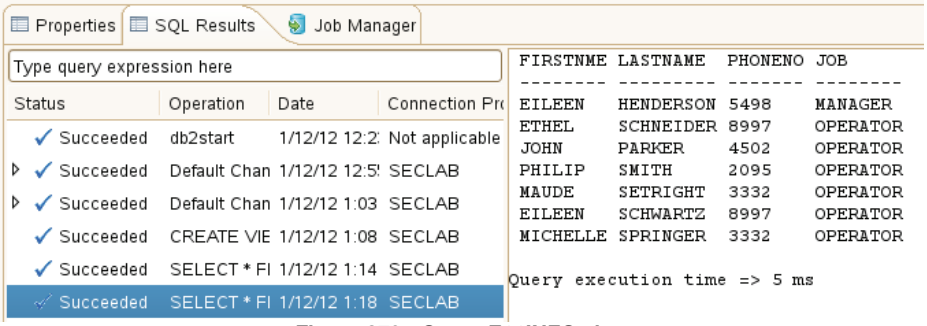


Figure 279 - Query E11INFO view

From now on, any new users that require access only to employees of department E11 can be granted access to the E11INFO view instead of the EMPLOYEE base table.

90 Roles

A role is a database object that may group together one or more privileges and can be assigned to users, groups, PUBLIC or to other roles via a GRANT statement. Roles simplify the administration and management of privileges.

90.1 Creating Roles

Consider now that your team is expanding and more application developers have joined in. Since all developers share the same access requirements, roles can be used instead of managing privileges individually to each user.

The security administrator holds the authority to create, drop, grant, revoke, and comment on a role.

1. Switch back to Data Studio and expand the **Users and Groups** folder on **Administration Explorer** View. Then, right click on **Roles** and select **Create Role**.

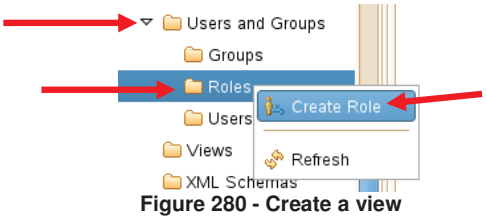


Figure 280 - Create a view

2. Name the role **Developer** and click on the **Privileges** tab.

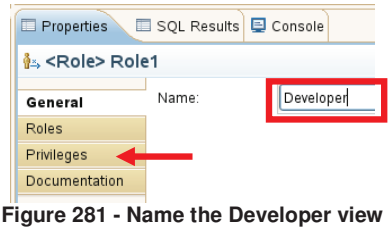


Figure 281 - Name the Developer view

3. Grant **BINDADD**, **CONNECT** and **CREATETAB** privileges on database.
4. Find the **Table** tab on the top. You might need to use the show list button ( ) if tab is not be visible.

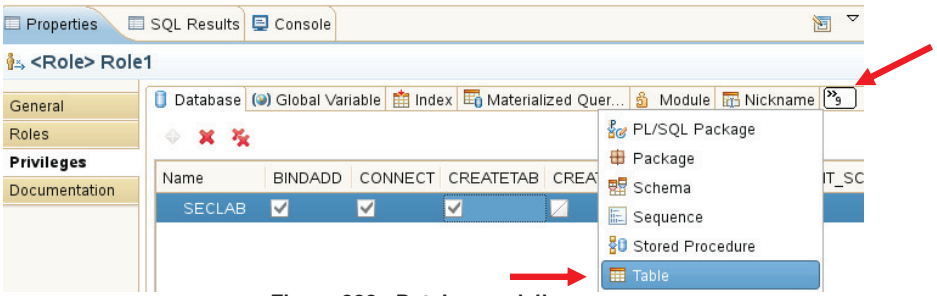


Figure 282 - Database privileges

5. Click on the **grant new privilege** button ( ), expand the **db2inst1** schema, select the **EMPLOYEE** table and click **OK**. Notice that the **EMPLOYEE** table is now visible in the **Privileges** tab.

6. Grant **SELECT**, **INSERT**, **UPDATE** and **DELETE** on **Employee** table:

Figure 283 Employee table privileges

7. Next, locate the toolbar on the top right corner ( ) and click on the **Review and Deploy Changes** button ( ). The Review and Deploy screen will display the commands that were generated to apply the desired changes.

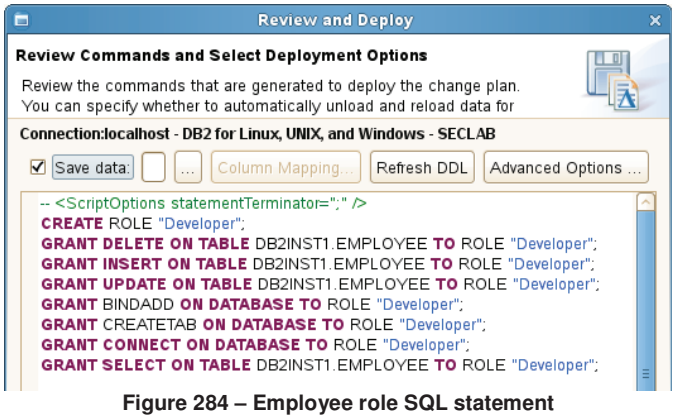


Figure 284 – Employee role SQL statement

8. Select the **Run** option and click the **Finish** button.

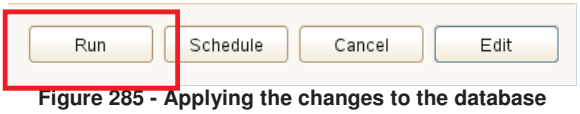


Figure 285 - Applying the changes to the database

9. Finally, check the **SQL Results** tab to ensure the commands were executed without errors.

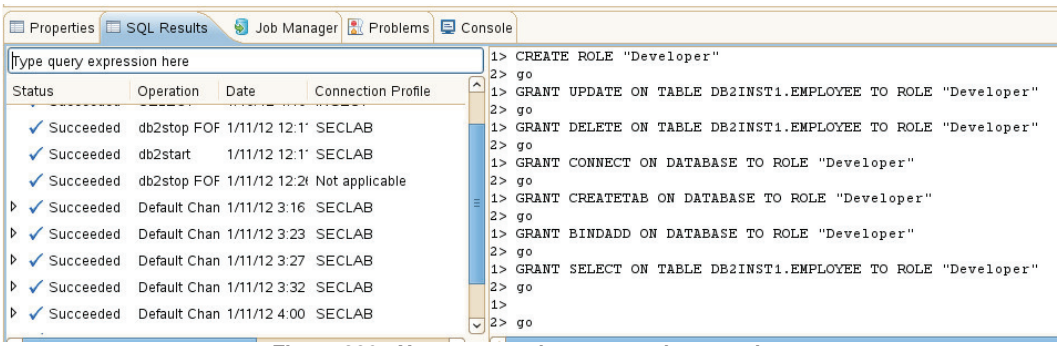


Figure 286 - No errors on the commands execution

The role Developer is now created. Now you must grant the role Developer to user **USERDEV**.

10. Click on the **Users** folder, right click on **USERDEV** and select **Alter**.

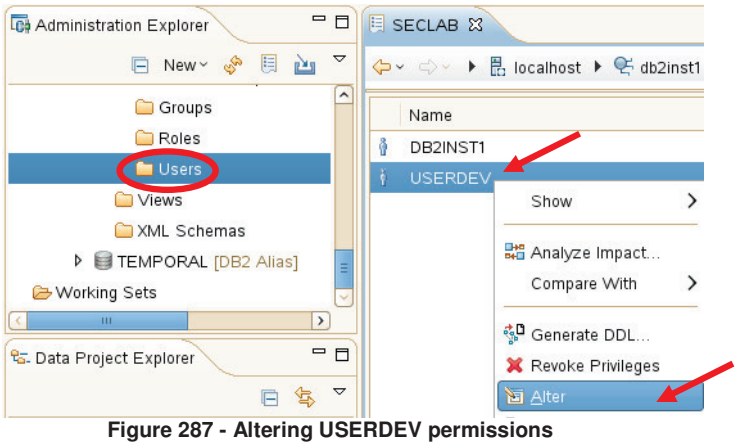


Figure 287 - Altering USERDEV permissions

Note that USERDEV is now in edit ( ) mode.

11. Click on the **Roles** tab, then click on the **grant new privilege** button ( ).

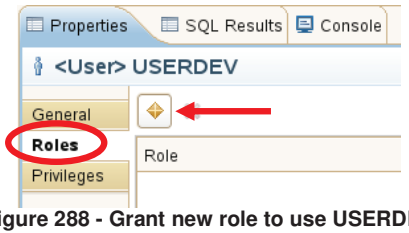


Figure 288 - Grant new role to use USERDEV

12. Select **Grant** on the role **Developer** and click **OK**.

Figure 289 - Select GRANT on role Developer

13. Next, locate the toolbar on the top right corner ( ) and click on the **Review and Deploy Changes** button ( ).

The Review and Deploy screen will display the commands that were generated to apply the desired changes.

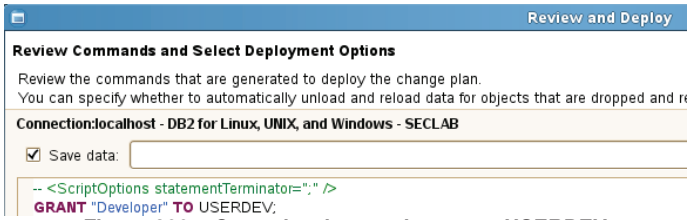


Figure 290 – Grant developer role to user USERDEV

14. Select the **Run** option and click the **Finish** button.

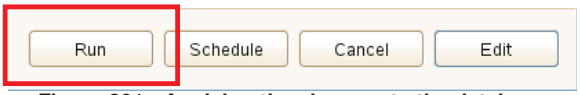


Figure 291 - Applying the changes to the database

15. Finally, check the **SQL Results** tab to ensure the commands were executed without errors.

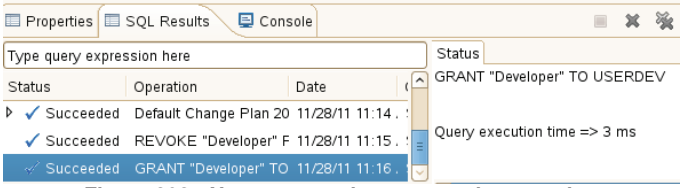


Figure 292 - No errors on the commands execution

The role **Developer** is now granted to user **USERDEV**. As more developers join the team, the **Developer** role can be assigned to them in the same manner we did for USERDEV. In the future, if developers require any changes to their security access, the security administrator can simply modify the privileges of the **Developer** role. All users associated to this role will automatically inherit the new privileges.

91 Summary

This lab presented an overview of the basic security mechanisms in DB2 for LUW.

Before connecting to a DB2 database, users must be authenticated, i.e. their user ID and password are validated. DB2 does not store user/password information in the database. Instead, it relies on an external authentication mechanism, by default the OS authentication mechanism.

After a user is authenticated, DB2 performs an authorization check, where the database manager verifies if the user is authorized to perform the requested operations on specific data objects. Users can be granted specific privileges, authorities or

roles. DB2 also allows you to create your own database roles to bundle several privileges together so you can grant them at once to a user, group or role.

92 Cleanup

5. To clean your environment after completing the exercise, right-click on **SECLAB** database**,** select **Disconnect**. Close Data Studio and execute the commands below. Use “password” as password.

```
su -
cd /home/db2inst1/Documents/LabScripts/Security/setup
./cleanup.sh
```

6. If you would like to redo this lab in the future, please execute the following commands in a terminal window as root:

```
cd /home/db2inst1/Documents/LabScripts/Security/setup
./config.sh
```

IBM Canada
8200 Warden Avenue
Markham, ON
L6G 1C7
Canada

DB2® Backup & Recovery

Hands-On Lab

Table of Contents

41 Introduction

An effective backup and recovery solution is critical for organizations to be protected against hardware/ software failures. Power interruptions, storage problems, or application failures can render a database useless, requiring a different recovery action for each of these failure scenarios. DB2 for LUW has a rich set of flexible backup and recovery features that make it easy for the user to backup and recover their data.

42 About the lab

This lab demonstrates some of the main backup and recovery features in DB2. It walks you through the recovery parameters used during a recovery operation along with demonstrating the different ways of backing up the database. At the end of this lab, you should have a good appreciation of DB2 recovery architecture and how to leverage it. Moreover, it also provides guidance on how to automate most of the tasks to make a database administrator's job simple.

43 Environment Setup Requirements

To complete this lab you will require the following:

34. DB2 10 VMware® image
35. VMware Workstation 6.5 or later

43.1 Preparation Step

Note: You many skip this section if you have already started your VM and DB2 instance in a previous lab session.

4. Start the VMware image by clicking the 

button in VMware Workstation if it is not already on.
5. At the login prompt, login with the following credentials:
    36. Username: **`db2inst1`**
    37. Password: **`password`**
6. Open a terminal window as follows by right-clicking on the **Desktop** and choosing the **Open in Terminal** item.

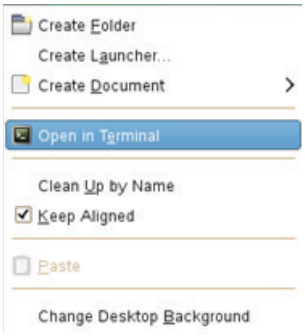


Figure 85 - Opening a Terminal

43.2 Launching Data Studio

7. Double-click on the **IBM Data Studio** icon on the **Desktop**.

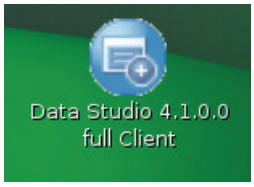


Figure 86 – IBM Data Studio Desktop Icon

8. In the **Select a workspace** dialog, accept the default and click **OK**.

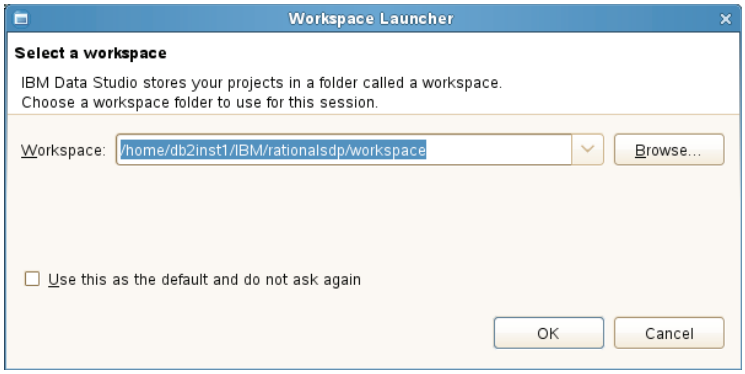


Figure 87 – Workspace Launcher

9. Data Studio will now start in the Database Administrator perspective as shown below.

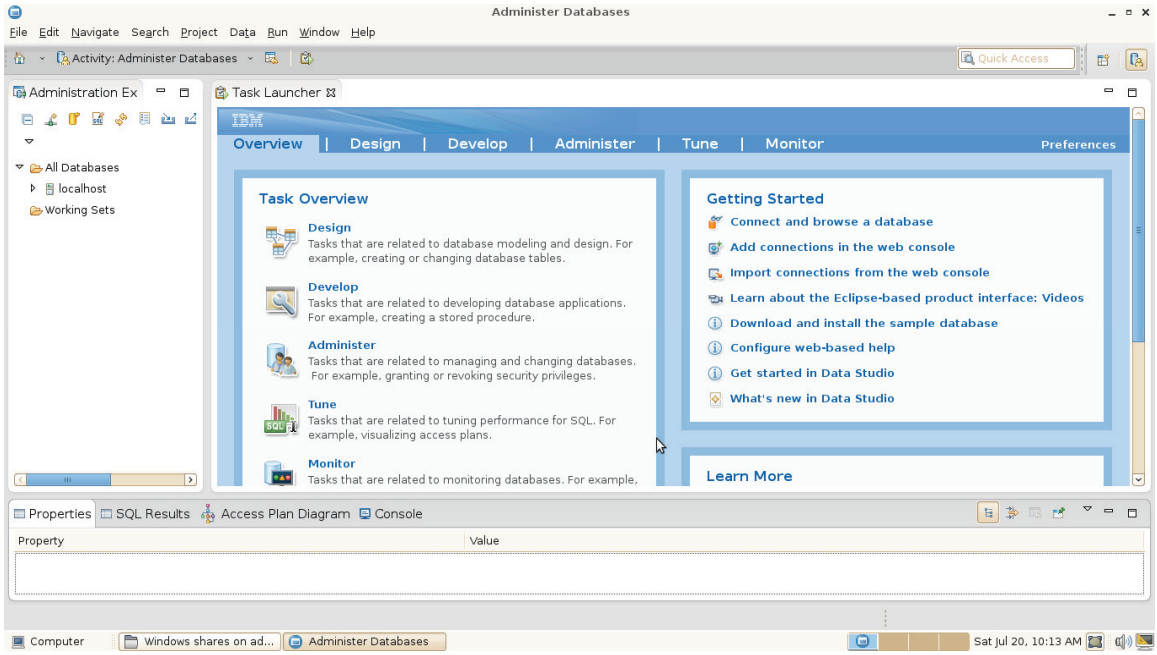


Figure 88 –Database Administration perspective

10. From Data Studio, under the **Database Administration** perspective and within the **Administration Explorer**, expand the nodes **All Databases** > **localhost** > **db2inst1**. Right click on the **DS2** database and **Connect**.

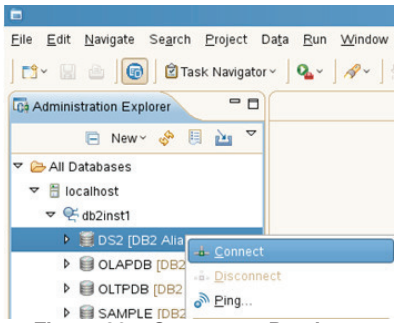


Figure 89 – Connect to Database

In the **Properties for DS2** window, specify the following credentials:

38. Username: **`db2inst1`**
39. Password: **`password`**

You can also optionally tick the **Save password** check box so you don't need to re-enter this information every time you connect to the database. Click **OK**.

43.3 Preparing the Database

In this lab, you will be working with the DS2 database. It has three user defined table spaces. The following table provides the relationship between the table space and tables they contain. All tables are created under the **DS2** schema.

TABLESPACE	TABLES or objects
CUSTTBS	CUSTOMERS, CUST_HIST
ORDERTBS	ORDERS, ORDERLINES
MISCTBS	PRODUCTS, INVENTORY, CATEGORIES, REORDER

44 DB2 Recovery Parameters

In DB2, there are several parameters and commands available to the user to control recovery behaviour. The default configurations may not meet your requirements. It is recommended that default recovery configuration parameters be updated to better suit your recovery needs. Backup images and archived transactional log files are used to perform recovery.

You have the flexibility to decide where to store these objects, for how long, etc. Therefore, one of the first things you should do before taking any backups is to update database configuration to match your needs. In this section we will look at important database recovery parameters and configure them appropriately.

44.1 Listing Database Configuration Parameters

1. From the terminal, issue the following command:

```
db2 connect to DS2
db2 get db cfg for DS2 | grep LOG
```

Record the values for database parameter in the following table:

Database Parameter	Value
LOGFILSIZ	
LOGPRIMARY	
LOGSECOND	
LOGBUFSZ	
NEWLOGPATH	
LOGARCHMETH1	

44.2 Configure Database for Recovery

44.2.1 LOGFILSIZ

It is the size of each transactional log file and it is given in 4KB pages. The default size is 1000 pages or 3.9 MB, which implies it can hold up to 3.9 MB of transactional data.

For the purposes of this exercise, make it smaller because transaction rate is low.

1. Issue the following command:

```
db2 update db cfg using logfilsiz 500
```

Note: SQL 1363W is returned on issuing the command; it indicates that the database needs to be reactivated for the parameter change to take effect.

This message will be returned for most of the commands in this section. The last step in this section is to restart the database for all the configuration changes to take effect.

44.2.2 LOGPRIMARY AND LOGSECOND

LOGPRIMARY is the number of primary log files. By default LOGPRIMARY is 3, therefore if you have 3 log files worth of uncommitted transactions, any new transactions would start utilizing the secondary log files.

LOGSECOND is the number of secondary log files. You can have this configured to accommodate spikes in transactional activity. By default LOGSECOND is 2 and it has a special setting of -1 which indicates infinite logging space, more on that later. Set LOGPRIMARY to 6 and LOGSECOND to 2.

1. Issue the following commands from terminal:

```
db2 update db cfg using LOGPRIMARY 6
db2 update db cfg using LOGSECOND 2
```

44.2.3 LOGBUFSZ

It is the memory buffer for log records. This is where all log records get written in memory before getting flushed to disk. The default of 256 (4KB) pages is small for most scenarios. Set the LOGBUFSZ to 512.

1. Issue the following command from the terminal:

```
db2 update db cfg using LOGBUFSZ 512
```

44.2.4 NEWLOGPATH

Log path is a directory location on the server where the log files get created. By default it is in the same location as the database control files.

1. Change the log path by issuing the following command:

```
db2 update db cfg using NEWLOGPATH /home/db2inst1/dblogs
```

44.2.5 LOGARCHMETH1, LOGARCHMETH2 AND FAILARCHPATH

By default, when a new database is created; it is configured for circular logging. Circular logging causes log files to be reused in a circular fashion with no history of log files preserved. For example, if you have LOGPRIMARY set to 5, after the 5th file is used up, DB2 will start reusing the first log file, considering it is available. With circular logging only crash recovery and offline backups are possible. In order to take online backups and perform roll-forward recovery, the database needs to be enabled for archival logging.

1. Enable database for archival logging and provide an archival location by issuing the following commands from the terminal window:

```
db2 update db cfg using logarchmeth1 'disk:/home/db2inst1/logarch'
```

This enables the database for archival logging and at the same time sets the archival location. We specified a disk location but it could be TSM server or another vendor API library.

LOGARCHMETH2 is a secondary log archival method. If this path is specified, log files will be archived to both this destination and the destination specified by LOGARCHMETH1. **FAILARCHPATH** is yet another layer of protection. If both archival methods fail, the database will try to archive to this location. Leave these two parameters blank for this exercise.

44.3 Restart the Database

1. Issue the following commands from Linux terminal to shutdown and restart the database:

```
db2 force applications all
db2 terminate
db2 deactivate db DS2
```

2. From Data Studio, under the **Database Administration** perspective and within the **Administration Explorer**, expand the nodes **All Databases** > **localhost** > **db2inst1**. Right click on the **DS2** database and **Connect**.

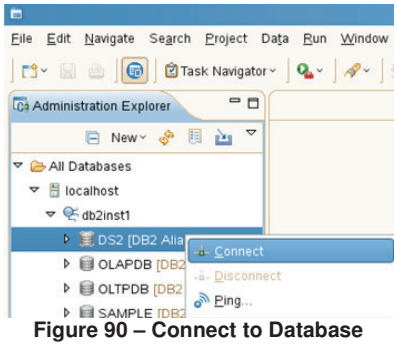


Figure 90 – Connect to Database

In the **Properties for DS2** window, specify the following credentials:

40. Username: **`db2inst1`**
41. Password: **`password`**

You can also optionally tick the **Save password** check box so you don't need to re-enter this information every time you connect to the database. Click **OK**.

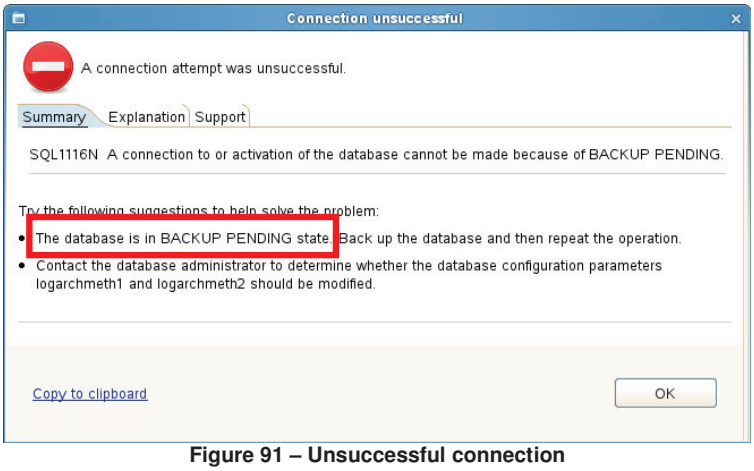


Figure 91 – Unsuccessful connection

This message is received because archival logging has just been enabled, which by design puts the database in ***BACKUP PENDING*** mode. This mode indicates that you need to do a full backup of the database before accessing it. Recall that once archival logging is enabled for the database, roll forward recoveries can be performed. However, roll forward recovery can only be performed once a backup image has been restored and database is placed in Roll Forward Pending status. For this reason, a backup operation is imposed after setting archival logging.

3. Right click on the **DS2** database, **Back Up and Restore** > **Back Up…**.

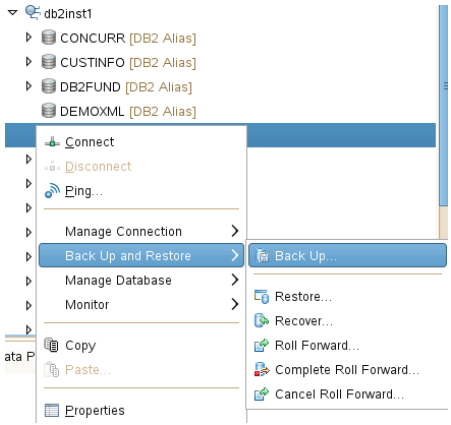


Figure 92 – Opening the Back Up task assistant

4. Within the **Back up DS2** tab, select the **Backup Image** settings, and click **Add…** to specify a backup location.

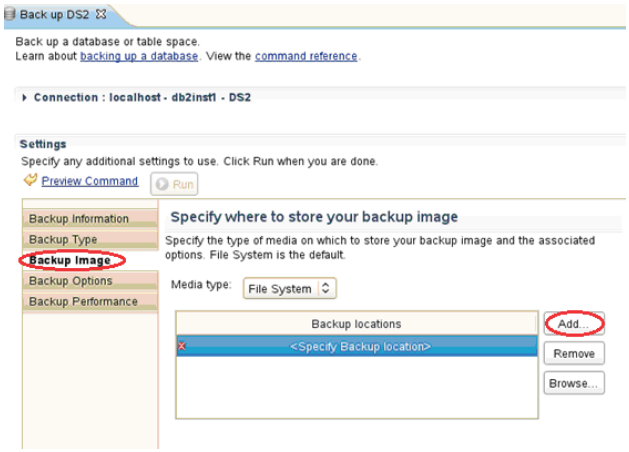


Figure 93 – Specifying where to store the backup image

5. Select the newly added backup location as shown above and click the **Browse…** button. Navigate to **/home/db2inst1/backups** and press **Ok** to select this location.

6. Click on Preview Command. This will allow you to see the command to be executed. You might have to expand the task assistant window to view the command displayed at the bottom.

```
BACKUP DATABASE DS2 TO "/home/db2inst1/backups" EXCLUDE LOGS WITHOUT
PROMPTING ;
```

7. Click Run button to execute the command. You will be requested to provide connection details for DS2.

8. The results of the backup command are provided in the SQL results pane at the bottom right of the screen.

```
Backup successful. The timestamp for this backup image is : <T1>
```

Every time there is a successful backup in DB2, a backup image timestamp is also returned. The Restore utility uses this timestamp to differentiate between multiple available backup images.

Note: Timestamp is dumped in the following format: **yyyymmddhhmmss.** Note the timestamp for the backup as T1

Time Label	Timestamp
T1	

Now the database should be connectable.

45 Backup and Restore Modes and Types

Backup and restore utilities are very flexible, allowing you to run backups at different object levels and in different modes. One of the simplest backups is the full offline database backup that was taken at the end of the previous exercise. The term “offline” here signifies that the database is offline with no applications connected and no activity in the database. The term “full” refers to the fact that entire database was backed up. We will see in next sections that it is possible to backup just a subset of the database. For example, one could backup only a single table space or a group of table spaces.

45.1 Full Online Backup

We have already taken a full offline backup in previous section. Now we'll take a full online backup of database DS2.

1. Go back to the terminal and connect to the DS2 database and verify there is a connection to the database by issuing the following commands:

```
db2 connect to DS2
db2 list applications
```

Output from command above should look something like:

```
Auth Id  Application    Appl.      Application Id                                          DB       # of
         Name           Handle                                                             Name     Agents
-------- -------------- ---------- ------------------------------------------------------- -------- -----
DB2INST1 db2bp          81         *LOCAL.db2inst1.111222224624                            DS2      1
```

2. Now try to perform a back up using Data Studio in the same way as we did in the previous section. Then select a location for the backup (as shown below) and click Run.

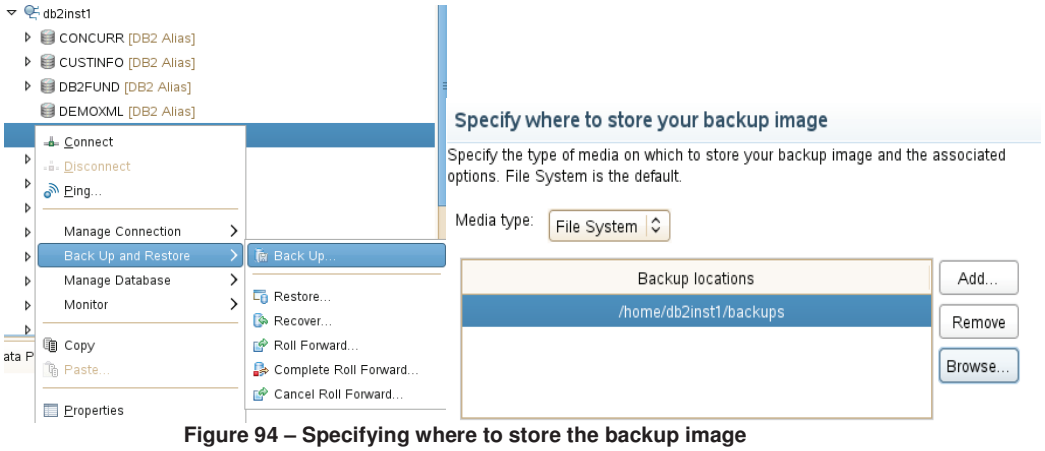


Figure 94 – Specifying where to store the backup image

The following error should be returned. The command failed as an offline backup is being performed while the database is online due to the connection in the terminal.

```
SQL1035N  The database is currently in use.  SQLSTATE=57019
```

3. To take an online backup, first connect to the DS2 database from Data Studio (Right click **DS2** > **Connect**)
4. Open the Back Up task assistant by following the directions outlined in step **2.**
5. Repeat the same procedure as before to add the backup location: within the **Back up DS2** tab.

6. Also, under the **Backup Options** settings, verify that **Online** is selected within the **Availability** section.

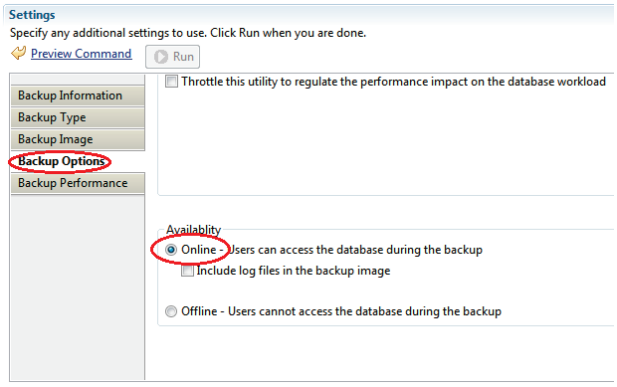


Figure 95 – Selecting online backup mode

7. Click on the Preview Command. This will allow you to see the command to be executed.

```
BACKUP DATABASE DS2 ONLINE TO "/home/db2inst1/backups" EXCLUDE LOGS WITHOUT
PROMPTING ;
```

8. Click the **Run** button to execute the command.

   Backup attempt should now be successful. Note the timestamp, it will be referred to as T2 in the next section.

Time Label	Timestamp
T2	

45.2 Restore from Online Backup

In this section, we will demonstrate the ease of restoring your database from a backup.

1. Go back to the terminal, issue the following command to close all database connections:

```
db2 force applications all
```

   Make sure you are still connected to the DS2 database through Data Studio. Open the Restore task assistant by right clicking on the **DS2** database, **Back Up and Restore** > **Restore…**.

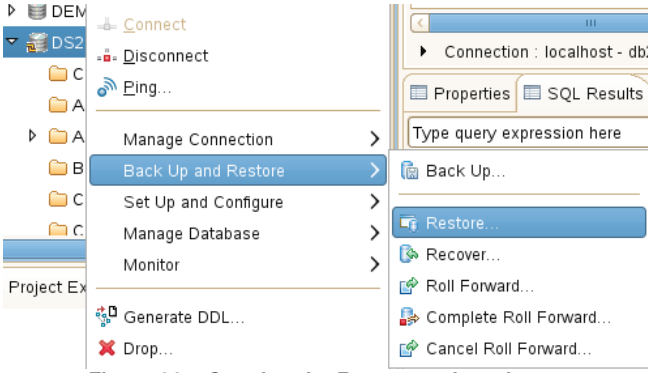


Figure 96 – Opening the Restore task assistant

2. Under the **Restore Objects** settings, navigate to the **Backup image** section, and select the backup image to restore which corresponds to timestamp T2. T2 is the timestamp for the online backup taken in previous section.

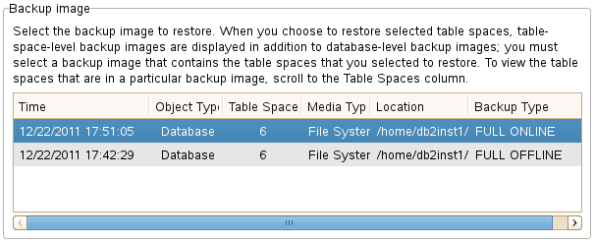


Figure 97 – Selecting Backup Image to Restore

3. Click on the Preview Command. This will allow you to see the command to be executed.

```
RESTORE DATABASE DS2 FROM "/home/db2inst1/backups" TAKEN AT <T2> WITHOUT
PROMPTING;
```

4. Click **Run** to execute the command.
5. The deployment status of the command is displayed in the ***Restore Database DS2*** tab.
    a. The command has been executed successfully but with a warning. This warning is ok; essentially it is stating that you are overwriting an existing database.
    b. The attempt to connect to the database fails. If we try connecting again (through Data Studio), we receive the following message:

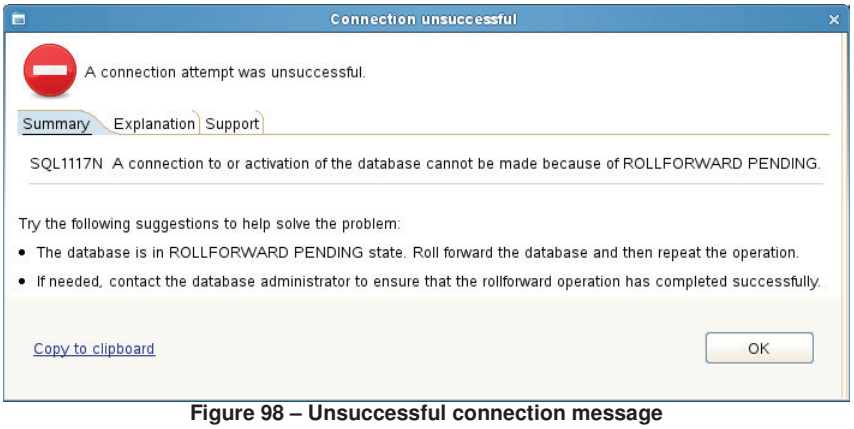


Figure 98 – Unsuccessful connection message

When an online backup is taken, any activity that may be going on during the online backup is recorded into the database transaction logs. Therefore, once you restore from an online backup there is a mandatory step needed to do a Roll Forward operation to a minimum Point in Time before the database can be made available.

6. In Data Studio, open the Roll Forward task assistant by right clicking on the **DS2** database, **Back Up and Restore** > **Roll Forward…**.

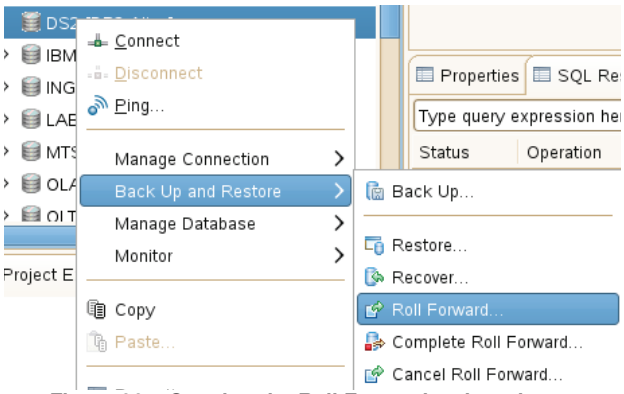


Figure 99 – Opening the Roll Forward task assistant

7. Under the **Roll-forward Final State** settings, within the **Final state options** section, select **Complete the roll-forward operation and return to the active state**.

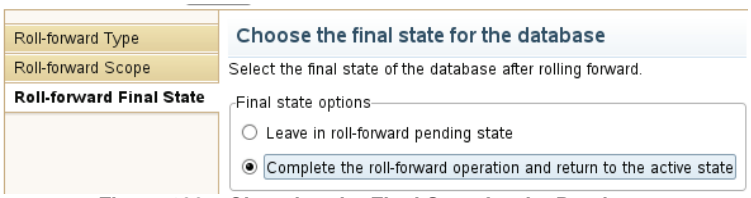


Figure 100 – Choosing the Final State for the Database

8. Click on Preview Command. The **COMPLETE** keyword at the end of ROLLFORWARD command indicates that once all available logs have been applied the database should be made available for user connections.

```
ROLLFORWARD DATABASE DS2 TO END OF LOGS AND COMPLETE;
```

9. Click the **Run** button to execute the command. You will be prompted to connect to the database, enter database password.

Output should look similar to the following:

```
                              Rollforward Status
 Input database alias                 = DS2
 Number of members have returned status = 1
 Member ID                            = 0
 Rollforward status                   = not pending
 Next log file to be read             =
 Log files processed                  = S0000000.LOG - S0000000.LOG
 Last committed transaction           = 2011-12-22-22.51.06.000000 UTC
DB20000I  The ROLLFORWARD command completed successfully.
```

Note: Rollforward status is set to **NOT PENDING,** meaning that database has been taken out of roll forward pending state and it is now connectable.

45.3 Restore from Offline backup

Restoring from an offline backup is simpler, as there is no hard requirement to perform a roll forward but you can do one if needed. Let us restore the database from an offline backup image taken at T1. If you do not recall, you can always check the Recovery history file to find the backup timestamp.

1. Go back to the terminal and issue the following command:

```
db2 list history backup all for database DS2
```

```
Op Obj Timestamp+Sequence Type Dev Earliest Log Current Log  Backup ID
 -- --- ------------------ ---- --- ------------ ------------ --------------
  B  D  20111222174229001   F    D  S0000000.LOG S0000000.LOG
 ----------------------------------------------------------------------------
  Contains 6 tablespace(s):

  00001 SYSCATSPACE
  00002 USERSPACE1
  00003 CUSTTBS
  00004 MISCTBS
  00005 ORDERTBS
  00006 SYSTOOLSPACE
 ----------------------------------------------------------------------------
    Comment: DB2 BACKUP DS2 OFFLINE
 Start Time: 20111222174229
   End Time: 20111222174231
     Status: A
 ----------------------------------------------------------------------------
  EID: 10 Location: /home/db2inst1/backups
```

Note “Timestamp” for the backup image and use it in restore operation as follows:

2. Connect to the DS2 database from Data Studio and open the **Restore Database** task assistant

3. Within the **Restore Database** tab, navigate to the **Restore Objects** settings. From the **Backup image** section, select the offline backup image which corresponds to timestamp T1.

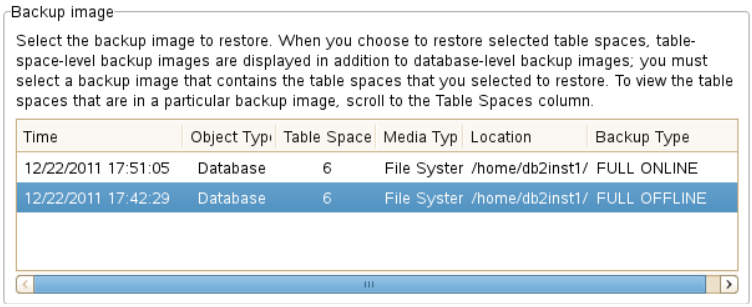


Figure 101 - Selecting Backup Image to Restore

4. Under the **Restore Options** settings, within the **Roll-forward options**, select the check box **“Do not put the database in roll forward pending state after restore”**.

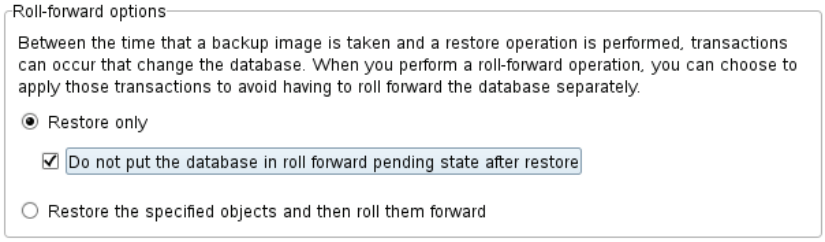


Figure 102 – Selecting Roll-forward options

5. Click on the Preview Command. This will allow you to see the command to be executed.

```
RESTORE DATABASE DS2 FROM "/home/db2inst1/backups" TAKEN AT <T1> WITHOUT
ROLLING FORWARD WITHOUT PROMPTING;
```

6. Click the **Run** button to execute the command.

Note: When restoring an offline backup image, a roll forward is not mandatory. Therefore, the clause `WITHOUT ROLLING FORWARD` is allowed in the restore command. This is useful when a roll forward is not needed and restore can finish in just one step. After restore finishes, you are able to connect without having to do a roll forward explicitly.

45.4 Database Roll-forward to a Point in Time

We briefly touched upon the Roll-forward task assistant in the last section. Roll-forward is the process of applying transaction log files after a restore has been performed. For example, last backup was taken Sunday and the database was lost on the following Tuesday. Once the backup from Sunday is restored, transactions in log files need to be applied in order to recover transactions that were done after the backup. This is achieved by rolling forward to "**END OF LOGS**".

There might be a situation where it is not desired to apply all the transactions. For example, a large set of records are deleted from the database mistakenly by the user. In such a case, in order to recover all the deleted records, rolling forward to a **POINT IN TIME** before the deletions took place would be more appropriate.

1. Create an online database backup exactly as we did in the previously using the same backup location. Refer to section 45.1 if needed.
2. Note the time stamp returned by the backup command. It will be needed for the restore operation later on. We'll refer to this timestamp as T3.

Time Label	Timestamp
T3	

3. Go back to the terminal window and check the number of rows in the ORDERS table:

```
db2 connect to DS2
db2 "select count(*) from DS2.ORDERS"
```

The expected number of rows in the ORDERS table is **12000**.

4. Run "**date**" command and note the time stamp before issuing a delete statement, it will be needed for point in time recovery. This timestamp will be referred to as T4 within this section. Issue the following command:

```
date
```

Time Label	Timestamp
T4	

5. Now run the following commands to delete some of the data in the ORDERS table:

```
db2 "delete from DS2.ORDERS where orderdate > date('11/30/2004')"
```

6. Check the count again:

```
db2 "select count(*) from DS2.ORDERS"
```

The expected number of rows in the ORDERS table is **11000**.

Imagine these rows were deleted mistakenly and they need to be recovered. This can be done by restoring from the backup image taken at T3.

7. First close the connections from the terminal:

```
db2 force applications all
db2 terminate
```

8. Make sure you are still connected to the DS2 database through Data Studio and than open the **Restore** task assistant by right clicking on the **DS2** database, **Back Up and Restore** > **Restore…**.

9. Under the **Restore Objects** settings, within the **Backup image** section, select the backup image to restore which corresponds to timestamp T3. T3 is the timestamp of the online backup taken previously in this section:

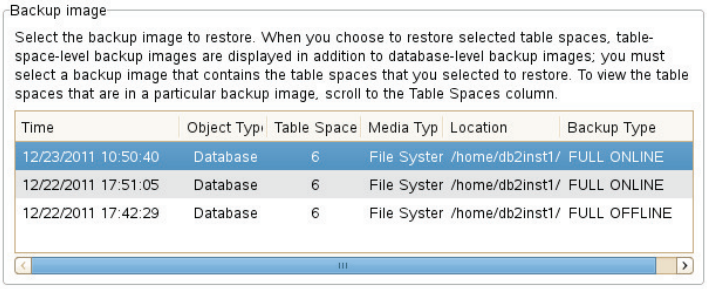


Figure 103 - Selecting Backup Image to Restore

10. Click on the Preview Command. This will allow you to see the command to be executed.

```
RESTORE DATABASE DS2 FROM "/home/db2inst1/backups" TAKEN AT <T3> WITHOUT
PROMPTING;
```

11. Click the **Run** button to execute the command.

Now that the database is restored, roll forward to a point in time before the delete statement was issued, which in this case is T4.

12. Open the Roll Forward task assistant by right clicking on the **DS2** database, **Back Up and Restore** > **Roll Forward…**.

13. Under the **Roll-forward Type** settings, within the **Type of roll-forward operation** section, select **A point in time specified in local time**.

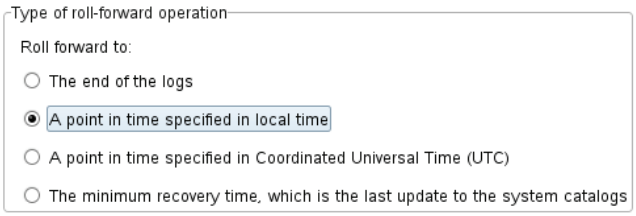


Figure 104 – Selecting Type of Roll-Forward Operation

14. Also, under **Last transaction to roll-forward** settings, within the **Backup image** section, specify the transaction time which corresponds to timestamp T4.

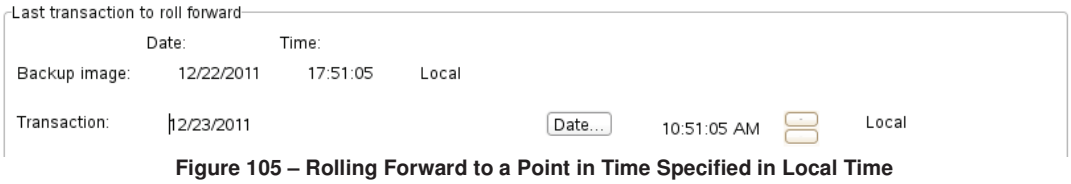


Figure 105 – Rolling Forward to a Point in Time Specified in Local Time

15. Click on Preview Command. This will allow you to see the command to be executed.

```
ROLLFORWARD DATABASE DS2 TO <T4> USING LOCAL TIME;
```

16. Click the **Run** button to execute the command.

Note: timestamp for roll forward has to be provided in this format: **yyyy-mm-dd-hh.mm.ss**.

Output from roll forward:

```
                              Rollforward Status
 Input database alias                   = DS2
 Number of members have returned status = 1
 Member ID                              = 0
 Rollforward status                     = DB  working
 Next log file to be read               = S0000005.LOG
 Log files processed                    = S0000000.LOG - S0000004.LOG
 Last committed transaction             = 2011-12-23-10.50.43.000000 Local
DB20000I  The ROLLFORWARD command completed successfully.
```

Lastly, take database out of the roll-forward pending status.

17. Open the Roll-forward task assistant by right clicking on the **DS2** database, **Back Up and Restore** > **Complete Roll Forward...**.

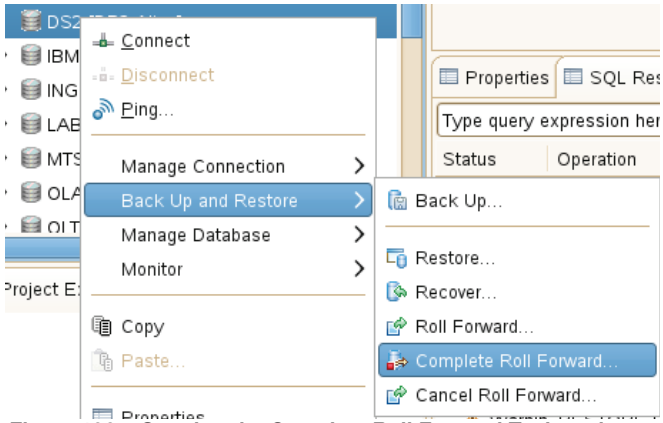


Figure 106 – Opening the Complete Roll Forward Task assistant

18. Leave the defaults and click Preview Command. This will allow you to see the command to be executed.

```
ROLLFORWARD DATABASE DS2 COMPLETE ;
```

19. Click the **Run** button to execute the command.

20. Go back to the terminal and check the number of rows in the ORDERS table again:

```
db2 connect to DS2
db2 "select count(*) from DS2.ORDERS"
```

The expected number of rows in the ORDERS table is **12000**, which means the deleted rows have been recovered. This was possible because the roll-forward was done up to a time before the delete statement was issued, thus all transactions since the backup was taken were applied except the delete statements in question

If an “end of logs” roll forward was done in this case, it would have also replayed the delete statement, thereby deleting the rows again.

21. Issue following command in any open terminals:

```
db2 terminate
```

45.5 Table Space Backup

Backups can be performed at either the database level or at the table space level. In large environments, it may not be desirable to perform a full database backup. You may have some table spaces where data is static and other table spaces where data changes regularly. With the table space backups you can backup only subsets of a database. If there is a requirement to only backup tables individually, it can be achieved by placing only 1 table per table space and then backing up the table spaces.

45.5.1 Take an Online Table Space Backup

We take the same scenario in the last section of recovering deletes, however in this case table space backups will be used to perform recovery. Take a table space backup of ORDERTBS table space. This table space contains the ORDERS table. Do the following to verify:

1. Connect to the DS2 database from Data Studio. Under the DS2 database node within Administration Explorer, click on **Tables** as shown:

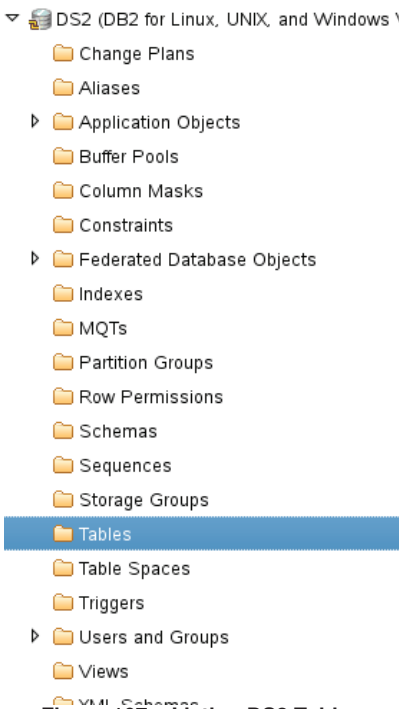


Figure 107 – Listing DS2 Tables

2. Than within the DS2 tab check the **Regular Tablespace** column corresponds to table ORDERS to verify the associated table space:

	Schema	Name	Percen	Row Count	Log Mode	Primary Key	Partition M	Regular Tablespace
	DS2	CATEGORIES	0					MISCTBS
	DS2	CUSTOMERS	0					CUSTTBS
	DS2	CUST_HIST	0					CUSTTBS
	DS2	INVENTORY	0					MISCTBS
	DS2	ORDERLINES	0					ORDERTBS
	DS2	ORDERS	0					ORDERTBS

Figure 108 – Verifying Associated Table space

3. Connect to the DS2 database from Data Studio and open the Backup task assistant (Right click **DS2** > **Back Up and Restore** > **Back Up…**).

4. Under the **Backup Type** settings, within the **Type of backup** section, select the **Back up selected table spaces** radio button.

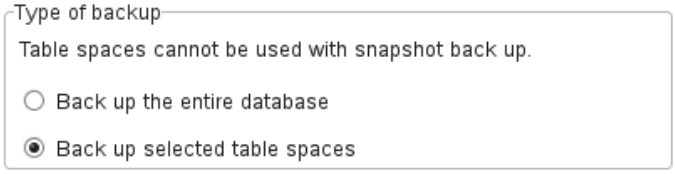


Figure 109 – Selecting the Type of Backup

5. Also under the **Backup Type** settings, within the **Table spaces** section, select the **ORDERTBS** table space.

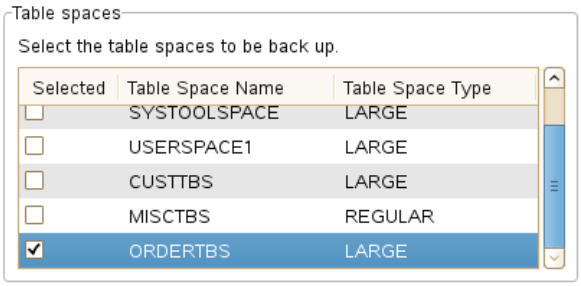


Figure 110 – Selecting Table Spaces to Backup

6. Within the **Back up DS2** tab, select the **Backup Image** settings, and click **Add...** to specify a backup location.

7. Click on Preview Command. This will allow you to see the command to be executed.

```
BACKUP DATABASE DS2 TABLESPACE ( ORDERTBS ) ONLINE TO
"/home/db2inst1/backups" EXCLUDE LOGS WITHOUT PROMPTING ;
```

8. Click the **Run** button to execute the command.

9. Note the timestamp of this backup, it will be referred to as T5.

Time Label	Timestamp
T5	

10. Go back to the terminal window and list directory for the backups:

```
ls /home/db2inst1/backups/ -ltrh
```

Output should look similar to the following:

```
total 550M
-rw------- 1 db2inst1 db2grp1 183M 2011-12-22 17:42 DS2.0.db2inst1.DBPART000.20111222174229.001
-rw------- 1 db2inst1 db2grp1 177M 2011-12-22 17:51 DS2.0.db2inst1.DBPART000.20111222175105.001
-rw------- 1 db2inst1 db2grp1 181M 2011-12-23 10:50 DS2.0.db2inst1.DBPART000.20111223105040.001
-rw------- 1 db2inst1 db2grp1 8.6M 2011-12-26 20:00 DS2.3.db2inst1.DBPART000.20111226200017.001
```

From the output two things should be noted. First, the latest backup image is smaller compared to the others since it contains only the data for ORDERTBS table space. Second, the backup type in the file name is 3 instead of 0 like the others:
`DS2.3.db2inst1.DBPART000.<timestamp>.001`

Note: Backup images have the following naming convention:.

<Database name>.<Backup type>.<instance name>.<database partition number>.<catalog partition number>.<timestamp>.<sequence number>

Backup type: **0** for full database backup; **3** for table space backup; **4** for backup generated by LOAD…COPY INTO command

number>

46 Incremental Backup and Restore

So far we have looked at full database and table space backups. As database sizes grow larger it can be quite costly to run full backups, both in terms of storage for the backup images and time required to execute the backups. This is where incremental backups come in.

In order to use the incremental backup functionality, database has to be enabled for it. This is done by turning the **TRACKMOD** database configuration parameter on. When TRACKMOD is turned on, database keeps track of table spaces that have been modified. When an incremental backup command is issued, it will skip the table spaces that have not been modified since last backup.

1. From Data Studio, right click on the (***connected***) **DS2** database, **Set Up and Configure** > **Configure…**.

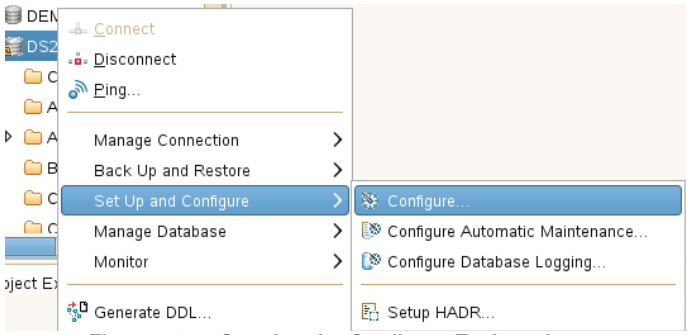


Figure 111 – Opening the Configure Task assistant

2. Under the **Configure Parameters** settings, use the Filter to search for the parameter **TRACKMOD**. Double click on the parameter and use the drop down menu to change the **Pending Value** to **ON**.

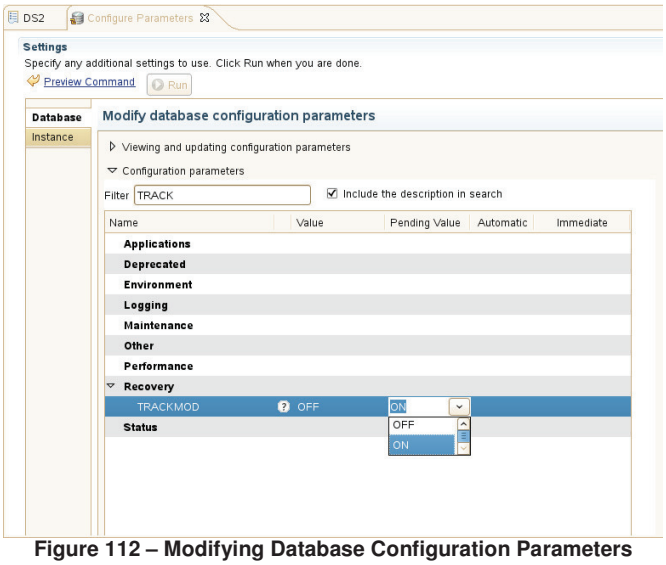


Figure 112 – Modifying Database Configuration Parameters

3. Click on the Preview Command. This will allow you to see the command to be executed.

```
CONNECT TO DS2;
UPDATE DB CFG USING TRACKMOD ON DEFERRED ;
CONNECT RESET;
```

4. Click the **Run** button to execute the command.

5. Issue the following commands from Linux terminal to shutdown and restart the database:

```
db2 force applications all
db2 terminate
db2 deactivate db DS2
```

6. Disconnect from the database and re-connect to perform a full backup. Incremental backups require a full backup to act as a reference point for incremental changes.

7. Open the backup Task Assistant:
   a. Under the **Backup Image** settings, select the backup location /**home**/**db2inst1**/**backups**
   b. Under the **Backup Options** settings, verify that **Online** is selected within the **Availability** section

8. Click on Preview Command. This will allow you to see the command to be executed. Click the **Run** button to execute the command.

```
BACKUP DATABASE DS2 ONLINE TO "/home/db2inst1/backups" EXCLUDE LOGS WITHOUT
PROMPTING ;
```

9. Note the timestamp of this backup, it will be referred to as T6.

Time Label	Timestamp
T6	…

10. Make changes to ORDERS which resides in ORDERTBS and CUSTOMERS table which resides in CUSTTBS. From terminal window run:

```
db2 connect to DS2
db2 "update DS2.ORDERS set tax=tax*1.05 where orderdate > date('11/30/2004')
"
db2 "update DS2.CUSTOMERS set income=income*0.9 where income > 90000"
```

11. With the database enabled for incremental backups, an incremental backup can be now taken to just include the changes made. Ensure you are connected to the DS2 database from Data Studio

12. Open the Back Up task assistant:
    a. Under the **Backup Image** settings, click **Add…** to specify a backup location.
    b. Under the **Backup Options** settings, verify that **Online** is selected within the **Availability** section.
    c. Also under the **Backup Options** settings, within the **Backup type** section, select the **Incremental** radio button to back up data that has changed since the last full backup:

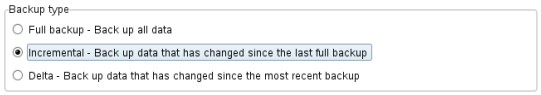


Figure 113 – Selecting incremental backup type

13. Click on the Preview Command. This will allow you to see the command to be executed. Click the **Run** button to execute the command.

```
BACKUP DATABASE DS2 ONLINE INCREMENTAL TO "/home/db2inst1/backups" EXCLUDE
LOGS WITHOUT PROMPTING ;
```

14. Note the timestamp of this backup, it will be referred to as T7.

Time Label	Timestamp
T7	

15. Check the backups directory for size of this incremental backup image. Issue this command:

```
ls /home/db2inst1/backups/ -ltrh
```

```
                                                                    …
-rw------- 1 db2inst1 db2grp1  17M 2011-12-28 10:08 DS2.0.db2inst1.DBPART000.20111228100849.001
```

16. Image backup size is much smaller than prior full backups because it contains only the changes since last full backup. Note that any table space that was not modified since the last full backup, will be skipped in incremental database backups. In this case it only backed up changes in the two table spaces ORDERTBS and CUSTTBS.

    Make further changes to table ORDERS. Issue the following command:

```
db2 connect to DS2
```

```
db2 "update DS2.ORDERS set tax=tax*1.05 where orderdate >
date('10/30/2004')"
```

Incremental backups can also be taken at table space level. Since the only change in the database was against the ORDERS table in ORDERTBS, let's back up the ORDERTBS table space.

Note: Recall that incremental backup backs up all changes since the last **full** backup. With the incremental **delta** option, only changes since the last backup (either incremental or full) will be backed up.

17. Ensure you are connected to the DS2 database from Data Studio and open the Back Up task assistant:
    a. Under **Backup Type,** within the **Type of backup** section, select **Back up selected table spaces** radio button.
    b. In the table spaces section, select the ORDERTBS table space.
    c. Under the **Backup Image** settings, specify a backup location.
    d. Under the Backup Options settings, within the Backup type section, select the **Delta** radio button to back up data that has changed since the most recent backup:

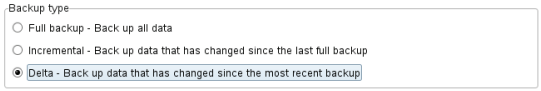


Figure 114 – Selecting incremental delta backup type

18. Click on Preview Command. This will allow you to see the command to be executed. Click the **Run** button to execute the command.

```
BACKUP DATABASE DS2 TABLESPACE ( ORDERTBS ) ONLINE INCREMENTAL DELTA TO
"/home/db2inst1/backups" EXCLUDE LOGS WITHOUT PROMPTING ;
```

Note the timestamp of this backup, it will be referred to as T8.

Time Label	Timestamp
T8	

19. Run more updates against the CUSTOMERS table, from terminal session issue:

```
db2 connect to DS2
db2 "update DS2.CUSTOMERS set income=income*0.9 where income > 80000"
```

20. Take incremental backup of CUSTTBS table space. Ensure you are connected to the DS2 database from Data Studio and open the Back Up task assistant
    a. Under the **Backup Type** settings, select the **Back up selected table spaces** radio button. Then within the **Table spaces** section, select the **CUSTTBS** table space.
    b. Select a backup location from the **Backup Image** settings
    c. Backup Options: Select **Incremental** from the **Backup type** section.

21. Click on the Preview Command. This will allow you to see the command to be executed, and click **RUN**.

```
BACKUP DATABASE DS2 TABLESPACE ( CUSTTBS ) ONLINE INCREMENTAL TO
"/home/db2inst1/backups" EXCLUDE LOGS WITHOUT PROMPTING ;
```

Note the timestamp of this backup, it will be referred to as T9.

Time Label	Timestamp

T9

Note: we only took an incremental backup and not an incremental delta backup; this would cover all the changes made since the full database backup in CUSTTBS table space.

46.1 Incremental Restore

When restoring from incremental backups, the right sequence of full, incremental and incremental delta backups has to be applied. This can become very complex very quickly in a real environment. For this reason, there is an **AUTOMATIC** option available with the restore command such that DB2 figures out the right sequence for applying backups and then applies them. There is also a **MANUAL** option available, but AUTOMATIC option is highly recommended.

1. Restore ORDERTBS table space using the incremental backups up to the last image. "**db2ckrst**" utility is available to see what backup images will be applied, prior to actually running the automatic restore.

```
db2ckrst -d DS2 -t <T8> -r tablespace ORDERTBS
```

T9 is the timestamp for the last delta backup image of ORDERTBS.

```
Suggested restore order of images using timestamp T8 for
database ds2.
=====================================================================
 restore db ds2 tablespace ( ORDERTBS ) incremental taken at T8
 restore db ds2 tablespace ( ORDERTBS ) incremental taken at T6
 restore db ds2 tablespace ( ORDERTBS ) incremental taken at T7
 restore db ds2 incremental taken at T8
=====================================================================
```

This output is showing that last incremental delta image will be read first to get the control and header information only (T8). Then the table space ORDERTBS will be restored from the full backup image (T6). Following, that first incremental image will be applied (T7). Lastly, the incremental delta image will be read again, this time applying the data in the delta image (T8)..

Note: The backup image that is being restored to is read twice; once in the beginning to obtain control information and decide how far the restore has to go and then again at the end to apply the data in last image.

2. Now, let's restore ORDERTBS to the last delta backup image. Make sure you are still connected to the DS2 database through Data Studio and than open the Restore task

    a. Under the **Restore Objects** settings, select the **Restore the select table spaces** radio button than select **ORDERTBS** from the Table spaces list.
    b. Also under the **Restore Objects** settings, within the **Backup image** section, select the backup image to restore which corresponds to timestamp T8.
    c. Under the **Restore Options** settings, select the **Allow other applications to connect to the database** radio button, as shown below.

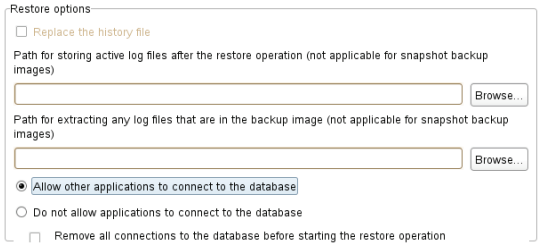


Figure 115 – Restore option to "Allow other applications to connect to the database"

3. Click on the Preview Command. This will allow you to see the command to be executed and click **RUN**.

```
RESTORE DATABASE DS2 TABLESPACE ( ORDERTBS ) ONLINE INCREMENTAL AUTOMATIC
FROM "/home/db2inst1/backups" TAKEN AT <T8> WITHOUT PROMPTING;
```

ORDERTBS will now be in roll forward pending because it has been restored. Take it out of RF pending:

4. Open the Roll Forward task assistant by right clicking on the **DS2** database, **Back Up and Restore** > **Roll Forward…**.
5. Under the **Roll-forward Scope** settings, select the **Roll the selected table spaces forward** radio button and select the **ORDERTBS** table space.
6. Click on the Preview Command. This will allow you to see the command to be executed and click **RUN**.

```
ROLLFORWARD DATABASE DS2 TO END OF LOGS TABLESPACE ( ORDERTBS ) ONLINE;
```

Output from successful roll forward command:

```
                              Rollforward Status
 Input database alias                 = DS2
 Number of members have returned status = 1
 Member ID                            = 0
 Rollforward status                   = not pending
 Next log file to be read             =
 Log files processed                  =  -
 Last committed transaction           = 2011-12-28-11.47.58.000000 Local
DB20000I  The ROLLFORWARD command completed successfully.
```

7. ORDERTBS should now be available and online. Verify using following commands:

```
db2 connect to DS2
db2 list tablespaces
```

Similarly, let's restore CUSTTBS to latest incremental backup:

1. Make sure you are still connected to the DS2 database through Data Studio and than open the Restore task assistant
   a. Within **Restore Objects** settings, select the **Restore the select table spaces** radio button than select **CUSTTBS**.
   b. Within the **Backup image** section, select the backup image to restore which corresponds to timestamp T9.
   c. Under the **Restore Options** settings, select the **Allow other applications to connect to the database** radio button.
2. Click on Preview Command. This will allow you to see the command to be executed. Click **RUN**

```
RESTORE DATABASE DS2 TABLESPACE ( CUSTTBS ) ONLINE INCREMENTAL AUTOMATIC
FROM "/home/db2inst1/backups" TAKEN AT <T9> WITHOUT PROMPTING;
```

T9 is the timestamp for the last incremental backup image of CUSTTBS.

3. CUSTTBS should now be in Roll forward pending. Open the Roll Forward task assistant:
   a. Under the **Roll-forward Scope** settings, select the **Roll the selected table spaces forward** radio button and select the **CUSTTBS** table space:

4. Click on the Preview Command. This will allow you to see the command to be executed, and **RUN**.

```
ROLLFORWARD DATABASE DS2 TO END OF LOGS AND COMPLETE TABLESPACE ( CUSTTBS )
ONLINE;
```

CUSTTBS should now be online and available.

5. Close the connection from data studio. Go to the terminal and close all connections to DS2

```
db2 force applications all
db2 terminate
```

47 Check Backup Command

The check backup command (**db2ckbkp**) can be used to test the integrity of a backup image and to determine whether or not the image can be restored. It can also be used to display the metadata stored in the backup header.

1. At its default configuration, db2ckbkp only verifies if the image is good. For instance, issue the following command with any one of the backup images taken (you find the backup image name by listing out the **~/backups** directory.):

```
cd /home/db2inst1/backups
db2ckbkp DS2.3.db2inst1.DBPART000.<TIMESTAMP>.001
```

The following should output if image verification is successful:

```
[1] Buffers processed:       ##########################################

Image Verification Complete - successful.
```

2. You can optionally use the "-h" argument to display media header information including the name and path of the image expected by the restore utility. For example:

```
db2ckbkp -h DS2.3.db2inst1.DBPART000.<TIMESTAMP>.001
```

```
=====================
MEDIA HEADER REACHED:
=====================
        Server Database Name           -- DS2
        Server Database Alias          -- DS2
        Client Database Alias          -- DS2
        Timestamp                      -- 20120116155426
        Database Partition Number      -- 0
        Instance                       -- db2inst1
        Database Configuration Type    -- 0
        Sequence Number                -- 1
        Database Member ID             -- 0
        Release ID                     -- F00
        Database Seed                  -- 929C6DE2
        DB Comment's Codepage (Volume) -- 0
        DB Comment (Volume)            --
        DB Comment's Codepage (System) -- 0
        DB Comment (System)            --
        Authentication Value           -- -1
        Backup Mode                    -- 1
        Includes Logs                  -- 0
        Compression                    -- 0
        Backup Type                    -- 3
        Backup Gran.                   -- 48
        Merged Backup Image            -- 0
        Status Flags                   -- 20
        System Cats inc                -- 0
        Catalog Partition Number       -- 0
        DB Codeset                     -- UTF-8
        DB Territory                   -- US
        LogID                          -- 1326744538
        LogPath                        -- /home/db2inst1/dblogs/NODE0000/LOGSTREAM0000/
        Backup Buffer Size             -- 4460544
        Number of Sessions             -- 1
        Platform                       -- 12
```

The above output contains a lot of important information. Just to highlight a few above, we can see if the backup image is online or offline; an incremental or an incremental delta backup; whether it is a compressed or uncompressed image, etc.

i.e.,:
Backup Mode -- 0 represents an offline backup
Backup Mode -- 1 represents an online backup
Backup Gran -- 0 represents a full backup
Backup Gran -- 16 represents an incremental backup
Backup Gran – 48 represents an incremental delta backup
Compression – 0 represents an uncompressed backup
Compression -- 1 represents a compressed backup

3. For details about other options, run db2ckbkp with no arguments or check the IBM Information Center.

```
db2ckbkp
```

48 Summary

With Database Backup and Recovery in DB2, you can increase application availability and ensure resiliency with reliable subsystem backups and point-in-time recovery. This lab outlined some of the main backup and recovery features found within DB2. By now, you should have a basic understanding of DB2 recovery architecture and how to use it with the new Data Studio tool to make your life as a database administrator simpler.

49 Cleanup

7. To clean your environment after completing the exercise, execute the command below. Use “password” as password.

```
su –
cd /home/db2inst1/Documents/LabScripts/Backup/setup
./cleanup.sh
```

8. If you would like to redo this lab in the future, please execute the following commands in a terminal window as root:

```
cd /home/db2inst1/Documents/LabScripts/ Backup /setup
./config.sh
```

IBM Canada
8200 Warden Avenue
Markham, ON
L6G 1C7
Canada

Multi-Temperature Data Management

Hands-On Lab

Table of Contents

11 Introduction

With data increasing at an exponential rate, management of the volumes of information in data warehouses is ever more critical. There is a strong tendency for a relatively small proportion of data, within data warehouses, to be frequently accessed or be *hot* data. Most of the data stored is infrequently access or cold data. This classification of *multi-temperature data* poses considerable challenges for optimization of storage management. As the data warehouse consumes increasing amounts of storage, optimizing the use of fast storage becomes increasingly important in managing storage costs.

The multi-temperature database storage feature in DB2 version 10 introduces a new way of prioritizing data and all activities touching the data. It allows users to manage multiple tiers of storage devices effectively by introducing the concept of *storage groups*.

12 About this Lab

In this lab you will explore the new concept of *Storage Groups* introduced in DB2 10. You learn how the automatic storage management capabilities of the DB2 database manager significantly reduce the management costs associated with storage.

Currently, you can only specify a single set of storage paths for an automatic storage managed database. This restricts the efficient use of drives of different speed and performance for data of varying priority. Through the usage of storage groups, the user is allowed to group together storage paths for devices with similar characteristics and can control where the data resides. This lab illustrates the ease in administration and management of storage groups and migration to start using storage groups.

12.1 Environment Setup Requirements

To complete this lab you will need the following:

1. DB2 10 VMware® image
2. VMware Workstation 6.5 or later

12.2 Preparing for the lab

1. Start the VMware image by clicking the 

button in VMware Workstation if it is not already on.
2. Login to the VMware virtual machine using the following information:
   - User: **db2inst1**
   - Password: **password**
3. Open a terminal window by right-clicking on the **Desktop** area and choose the "**Open in Terminal**" item.

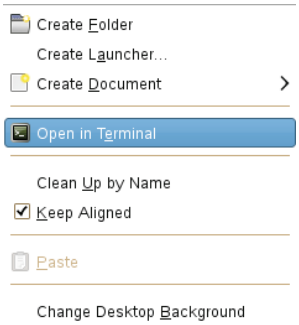


Figure 2 - Open Terminal

4. Start the DB2 server

```
db2start
```

5. The file paths to be assigned to storage groups through out the lab have already been created. Navigate to the folder that contains all the storage paths used within this lab:

```
cd /home/db2inst1/mtsdb/
```

```
fastpath1  fastpath3  medpath2  medpath4   slowpath2
fastpath2  medpath1   medpath3  slowpath1
```

Figure 3 - List of storage paths created

6. All scripts used during the lab reside within the **LabScripts** folder. Navigate to the following location:

```
cd /home/db2inst1/Documents/LabScripts/Multi_Temp_Storage/
```

13 Create and Manage Storage Groups

The distribution of data into various classes of storages can be done using storage groups introduced in DB2 10.

A storage group is a new layer of abstraction encompassing table spaces. It acts as a named set of storage paths where data can be stored and can be configured to represent different classes of storage. Each storage group can have its own unique characteristics, and you can define which table spaces reside within each storage group.

In this section, we will demonstrate how to create and alter the characteristics of a storage group. To demonstrate how to create and manage storage groups we will perform a series of operations on the database **MTSDB**, which we have created for your convenience.

4. Double click 'IBM Data Studio' on the Desktop.

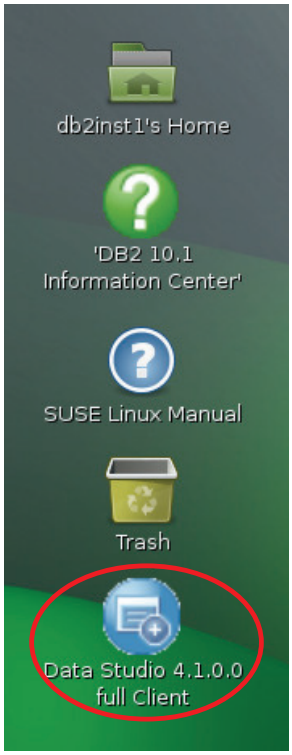


Figure 4 - IBM Data Studio's icon on Desktop

5. If this is the first time Data Studio is launched, it might take a moment. Click **OK** to accept the default workspace.

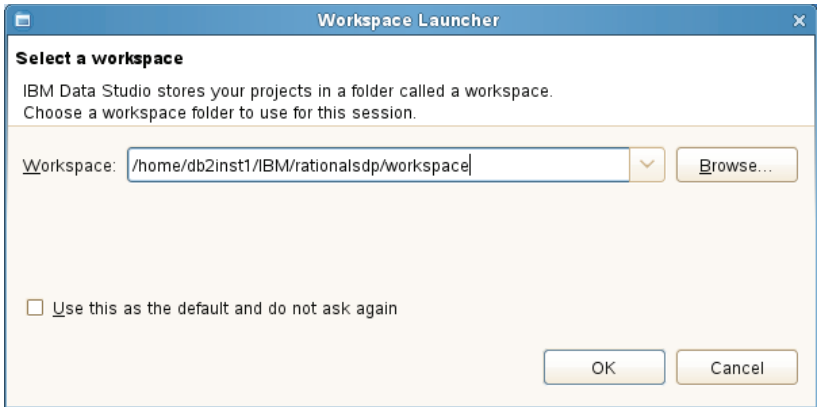


Figure 5 - Select a workspace

6. Locate the **Administration Explorer** view; expand the folder **All Databases** until you see **MTSDB**.

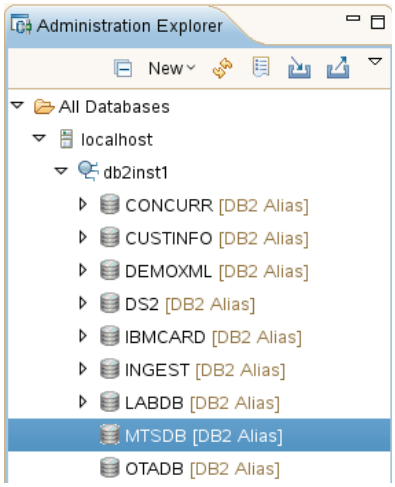


Figure 6 – MTSDB Database

7. Still in Administration Explorer, right click **MTSDB** and select **Connect**.

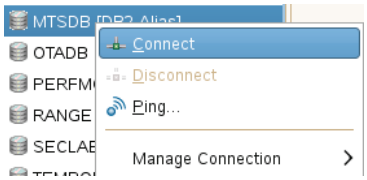


Figure 7 - Connect to MTSDB database

8. In the pop-up dialog, fill in the connection details as follows:
    - Database : **MTSDB**
    - Host : **localhost**
    - Port number : **50001**
    - User name : **db2inst1**
    - Password : **password**

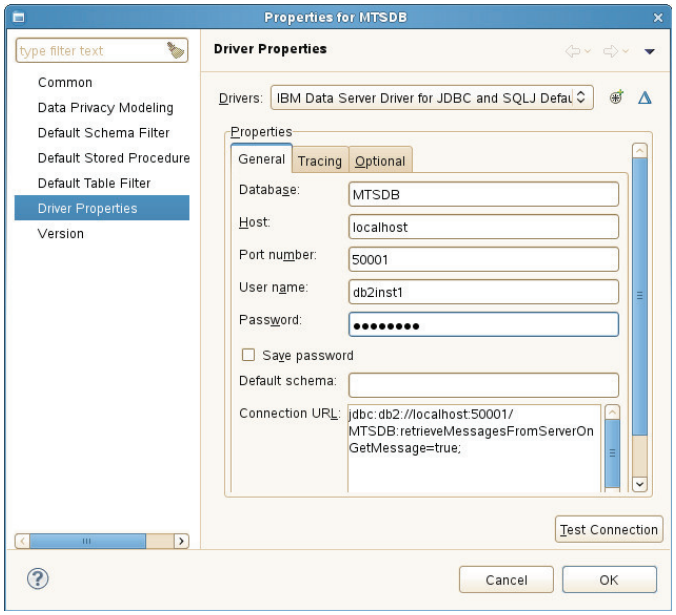


Figure 8 - Database Connection dialog

9. Click OK.

13.1 DB2's Default Storage Group – IMBSTOGROUP

When you create a database, a default storage group named **IBMSTOGROUP** is automatically created. This storage group is created within the storage paths that were specified at the time of the database creation.

There can only be one storage group designated as the default storage group.

NOTE: A database created with the *AUTOMATIC STORAGE NO* clause does not have a default storage group. The first storage group created with the CREATE STOGROUP statement becomes the designated default storage group

1. After a connection has been successfully established, expand the **db2inst1** instance folder in Administration Explorer and select the **Storage Groups** folder.

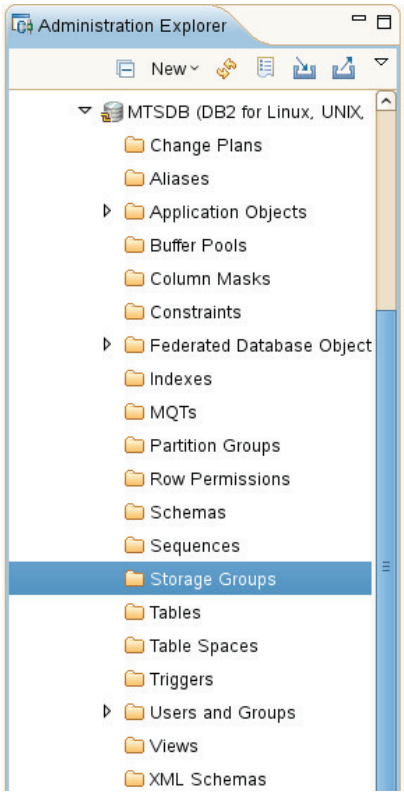


Figure 9 - Storage Groups folder

The list of storage groups is automatically displayed in the Object List area on the right. Note that its properties are also displayed (overhead, device read-rate, etc.). The value **true** in the **default** column indicates that this is the default storage group. It will be used when an automatic storage managed table space is created without explicitly specifying the storage group.

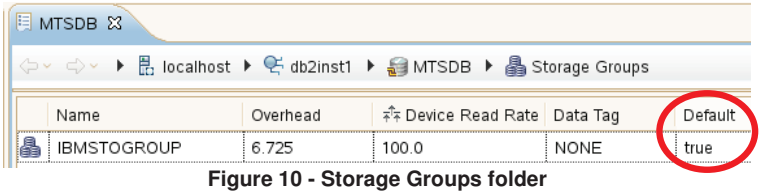


Figure 10 - Storage Groups folder

2. Next, let's check the table spaces automatically created for the MTSDB database, within IBMSTOGROUP. Click on **New SQL Script menu bar**.

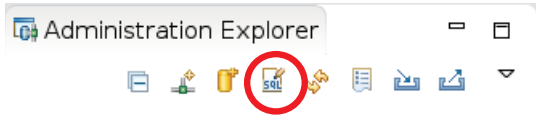


Figure 11 - New SQL Script

3. Copy the following query into the text box, then click on **Run** to execute the query. If prompted, press **Ok** to use the default statement terminator (semi-colon).

```
SELECT varchar(tbsp_name, 30) as tbsp_name, storage_group_name  FROM
TABLE(MON_GET_TABLESPACE('',-2)) AS t WHERE t.tbsp_using_auto_storage = 1 and
t.storage_group_name = 'IBMSTOGROUP'
```

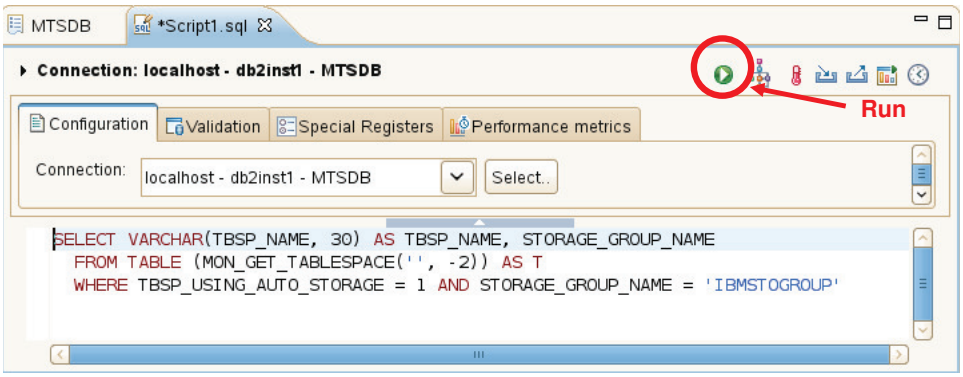


Figure 12 - SQL Query Textbox

Note that all the table spaces created within the database reside in the DB2 default storage group (IBMSTOGROUP).

Status | Result1

	TBSP_NAME	STORAGE_GROUP_NAME
1	SYSCATSPACE	IBMSTOGROUP
2	TEMPSPACE1	IBMSTOGROUP
3	USERSPACE1	IBMSTOGROUP
4	IBMDB2SAMPLEREL	IBMSTOGROUP
5	IBMDB2SAMPLEXML	IBMSTOGROUP
6	SYSTOOLSPACE	IBMSTOGROUP

Total 6 records shown

Figure 13 - Table spaces created by default within IBMSTOGROUP

13.2 Create a Storage Group

When defining storage groups, ensure that you group the storage paths according to their quality of service characteristics. The common quality of service characteristics for data follow an aging pattern where the most recent data is frequently accessed and requires the fastest access time (hot data) while older data is less frequently accessed and can tolerate higher access time (warm data or cold data). The priority of the data is based on:

- Frequency of access
- Acceptable access time
- Volatility of the data
- Application requirements

In this sub-section, we will create new storage groups and specify their media attributes which distinguish each group by its quality of service characteristics. **sg_high** caters to fast devices such as solid state storage devices while midrange performance devices such as fiber channel disks can be assigned to **sg_medium** for efficient usage of your physical storage.

1. In the **Administration Explorer** view, right-click the **Storage Groups** folder, then select **Create Storage Group**.

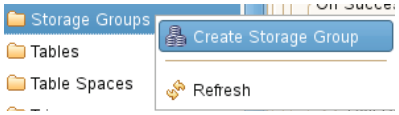


Figure 14 - Create new storage group

A new storage group item will be added to the storage group list.

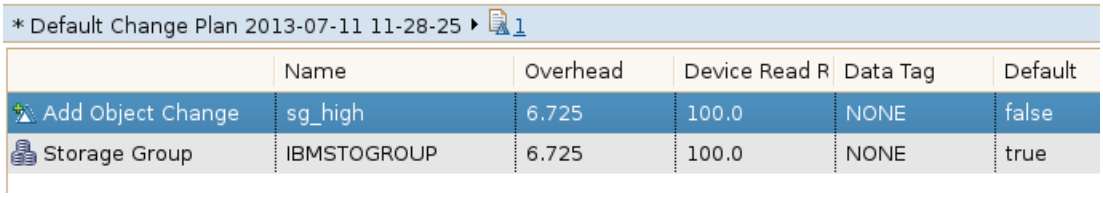


Figure 15 - New storage group

2. Click on **StorageGroup1** and locate the **Properties** view on the bottom. **Change the name** to **sg_high** and click on **Add Storage Path** button.

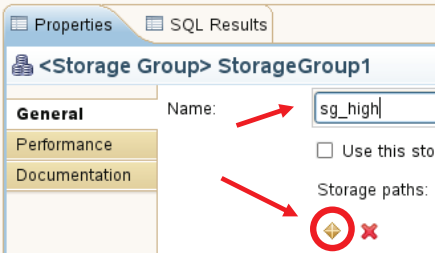


Figure 16 - Rename the storage group and add storage path

3. Navigate to /**home**/**db2inst1**/**mtsdb**/ directory, select **fastpath1** and click **OK**.

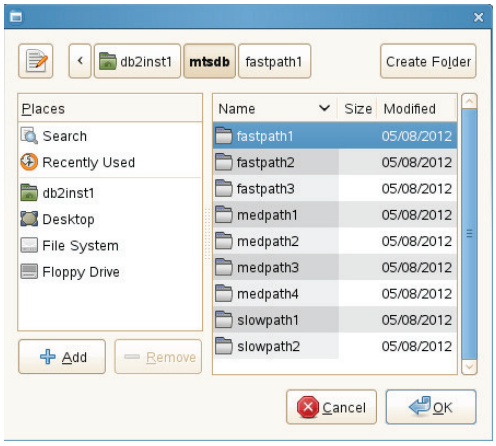


Figure 17 - Select fastpath1 directory

Note the new path is added to the storage paths list.

Figure 18 - New storage path added

4. Still in the **Properties** view, click on the **Performance** tab and input the following values for the storage group attributes:
    - **Device read rate: 500**
    - **Overhead: 0.75**
    - **Data Tag: 1 (highest priority)**

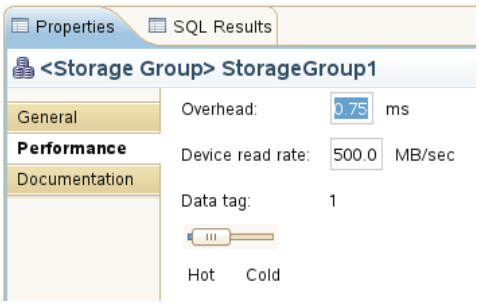


Figure 19 - Storage group attributes

5. Repeat steps 1 to 4 to create another storage group using the attributes below:
    - **Storage group name: sg_medium**
    - **Storage path: /home/db2inst1/mtsdb/medpath1**

Note that when no media attributes are defined, DB2 uses the following defaults.

Media Attribute	Default Setting
Data Tag	NONE
Device Read Rate	100 MB/sec
Overhead	6.725 ms

ⓘ Attention: as all storage paths are defined on the same machine, this lab will not show the performance aspect of data accessed on storage groups with varying performance characteristics

6. Next, click on **Review and Deploy (**  **)** button on the top right section of the screen, review the CREATE STOGROUP SQL statements and click **Run**.

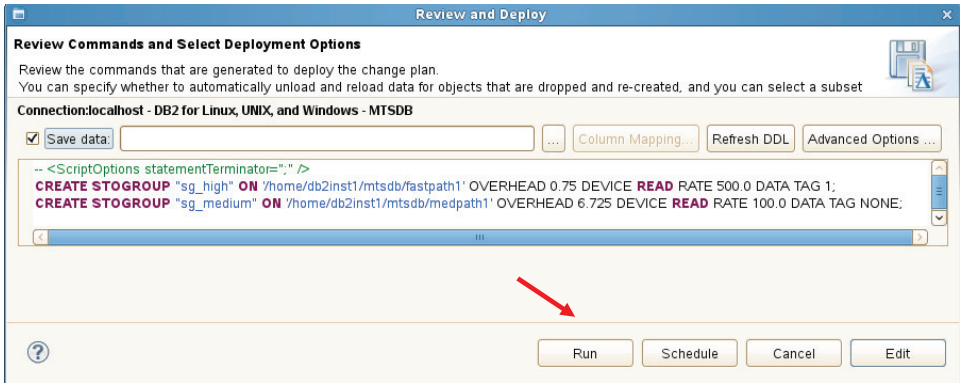


Figure 20 - Deploy the changes to the database

Note the storage groups list is automatically updated with the newly created storage groups along with the media attributes specified.

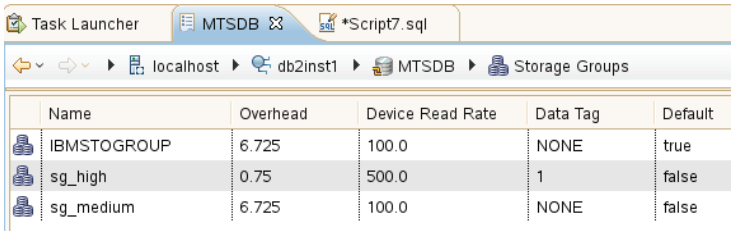


Figure 21 - sg_high and sg_medium storage groups added to the storage groups list.

13.3 Choose a Different Storage Group as Default

You can designate a default storage group of your choice, by using either the **CREATE STOGROUP** or **ALTER STOGROUP** statements. When you designate a different storage group as the default storage group, there is no impact to the existing table spaces using the old default storage group.

1. Right-click the **sg_high** storage group and select **ALTER.**

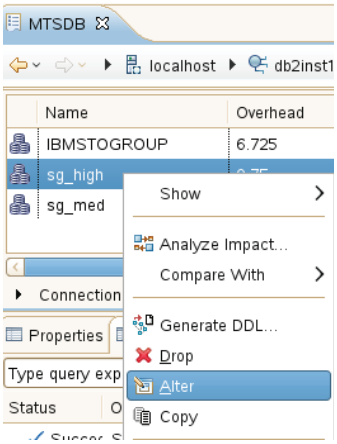


Figure 22 - Alter sg_high storage group

2. In the **Properties** view, locate the **General** tab and **check** the **"Use this storage group as the default for new table spaces"**

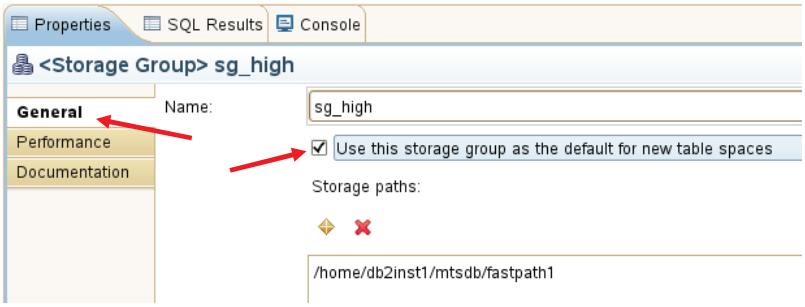


Figure 23 - Select default storage group

3. Click on **Review and Deploy** () button on the top right section of the screen, review the ALTER STOGROUP SQL statement,then click **RUN.**

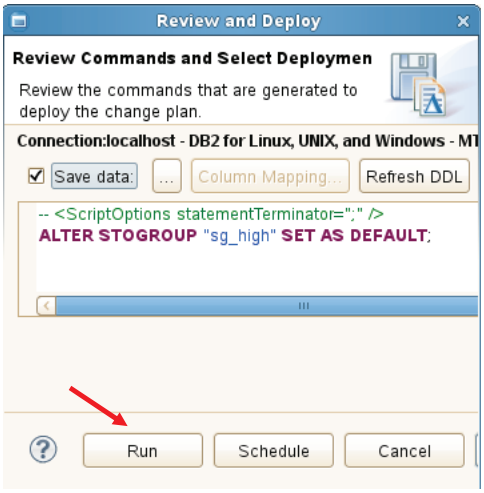


Figure 24 - Set sg_high as default storage group

Note that the storage group **sg_high** is the new default storage group in the storage groups list.

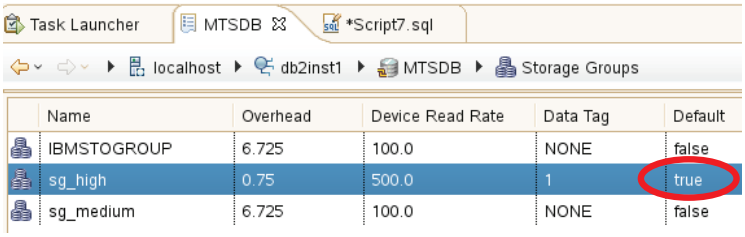


Figure 25 – sg_high storage group as default

Any new automatic storage table space created will be assigned to this storage group, unless a storage group assignment was specified, at the time of creation.

4. In the **Administration Explorer** view, locate and right-click the **Table Spaces** folder, then select **"Create Regular Table Space"**.

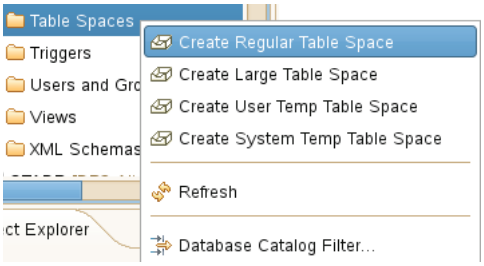


Figure 26 - Create new table space

5. Name the new table space **"Test_Default_Tbsp"**, select **Automatic Storage** management. Note that you are not specifying any storage attributes, such as the device read rate and overhead. These attributes will be inherited from the default storage group.

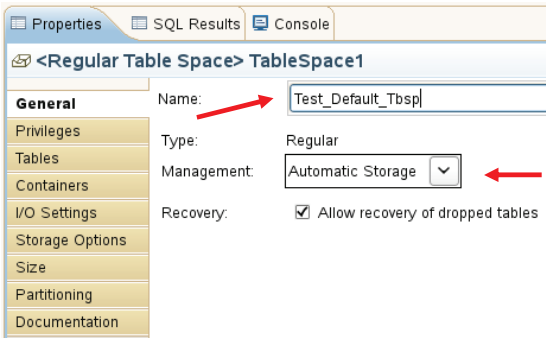


Figure 27 - Test_Default_Tbsp table space attributes

6. Click on **Review and Deploy** () button on the top right section of the screen, review the CREATE TABLESPACE SQL statement, then click **RUN.**

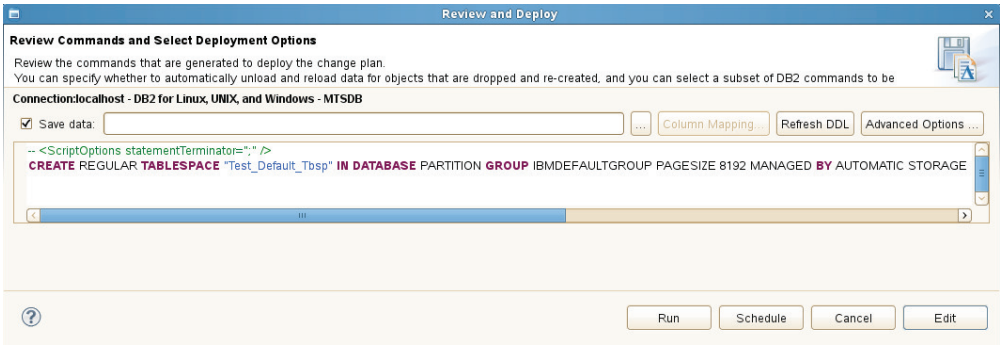


Figure 28 - Add new table space into the database

7. Now locate and select the **Test_Default_Tbsp** table space, then, in the **properties** view, select the **Storage Options** tab. Note the Test_Default_Tbsp residing within the default storage group **sg_high**.

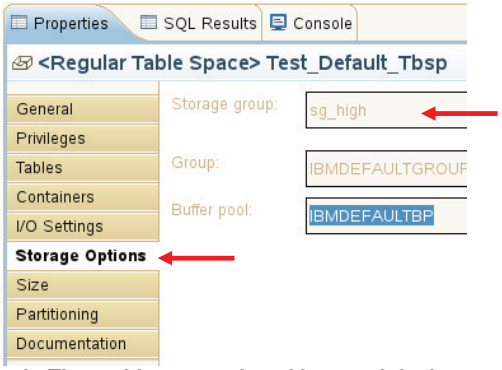


Figure 29 – Test_Default_Tbsp table space placed in new default storage group sg_high

13.4 Associate a Table space to a Storage Group

In this sub-section we will demonstrate how to associate a table space to a storage group. A new table space can be assigned to a storage group using the **CREATE TABLESPACE** statement or an existing table space can be associated using the **ALTER TABLESPACE** statement to specify or change the storage group the table space uses.

13.4.1 Create new Table space

We will create a new table space **tbspc_high** and assign it to the **sg_high** storage group. In this example, we have chosen to inherit the media attributes of the storage group. The automatic storage table space created, inherits the overhead and transfer rate attributes from the storage group it is assigned to. When a table space inherits the transfer rate, the storage group's device read rate (sg_high: 500 MB/sec) is converted to milliseconds per page read, using the page size setting of the table space.

1. Right-click the **Table Spaces** folder, select **Create Regular Table Space**. Name it **tbspc_high,** select **Automatic Storage** management, and then click the **Storage Options** tab.

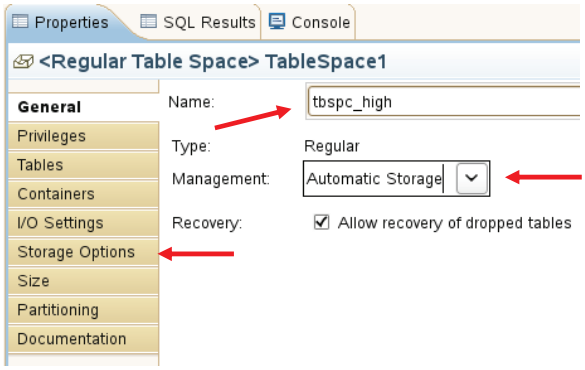


Figure 30 - tbspc_high table space definition

2. Select **sg_high** from the storage group dropdown menu.

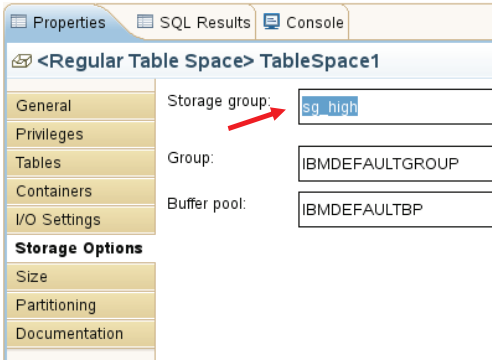


Figure 31 - Select sg_high storage group

3. Click on **Review and Deploy** () button on the top right section of the screen, review the CREATE TABLESPACE SQL statement, then click **RUN.**

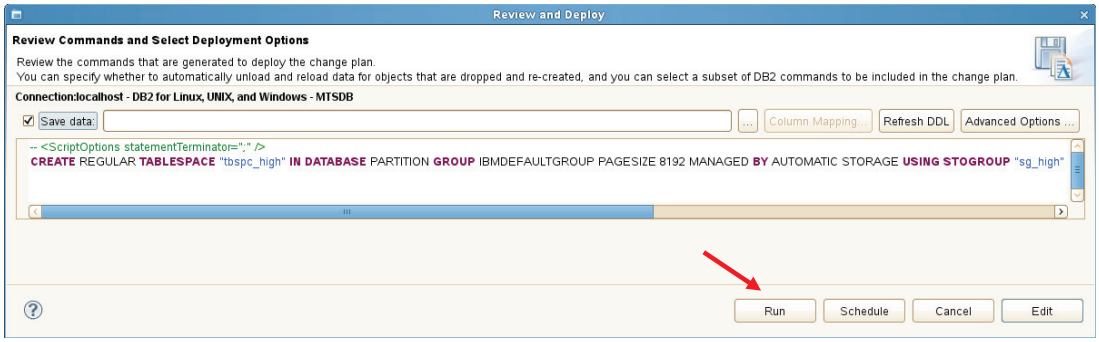


Figure 32 - Deploy new table space into the database

4. Now locate and select the **tbsp_high** table space, then, in the **properties** view, select the **Storage Options** tab. Note the Test_Default_Tbsp residing within the default storage group **sg_high**.

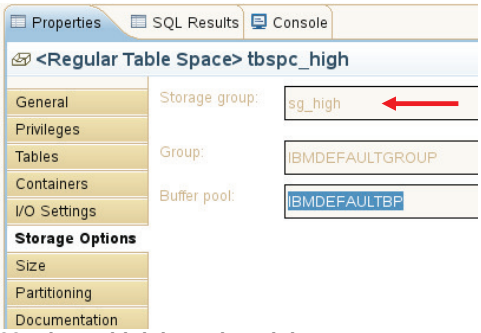


Figure 33 - tbspc_high is assigned the storage group sg_high

13.4.2 Move Table space to a new Storage Group

Pre-existing table spaces can be moved to a different storage group. In this scenario we will show you how an existing table space can be converted to start using a new storage group. We will be moving the existing table space **IBMDB2SAMPLEXML** which currently resides within the storage group **IBMSTOGROUP** to **sg_medium.**

1. Locate and select the **IBMDB2SAMPLEXML** table space, then right-click the **IBMDB2SAMPLEXML** table space and select **ALTER**.

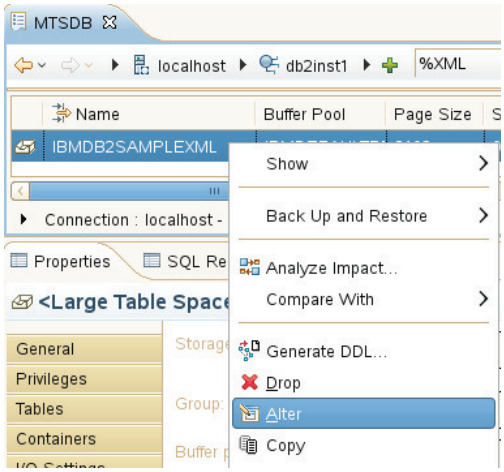


Figure 34 - Alter the IBMDB2SAMPLEXML table space

2. In the properties view, select the **Storage Options** tab. Note the table space residing within the storage group **IBMSTOGROUP**.

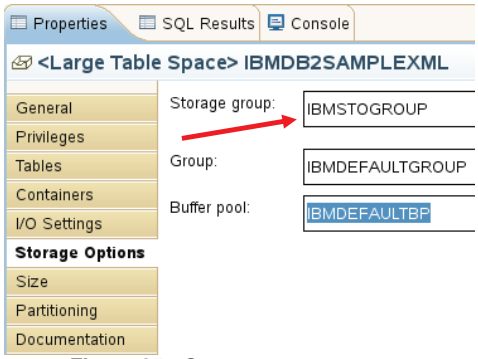


Figure 35 - Current storage group

3. Click the **Storage group** dropdown menu and select the **sg_medium**.

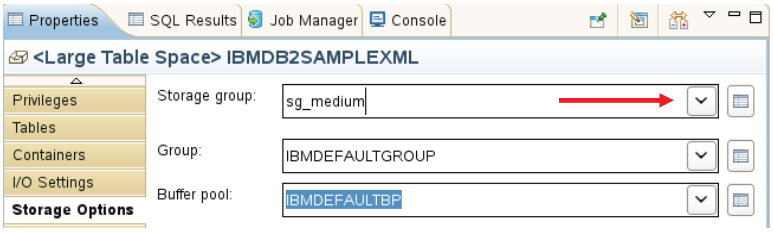


Figure 36 - Select sg_medium storage group

4. Click on **Review and Deploy** () button on the top right section of the screen, review the ALTER TABLESPACE SQL statement, then click **RUN.**

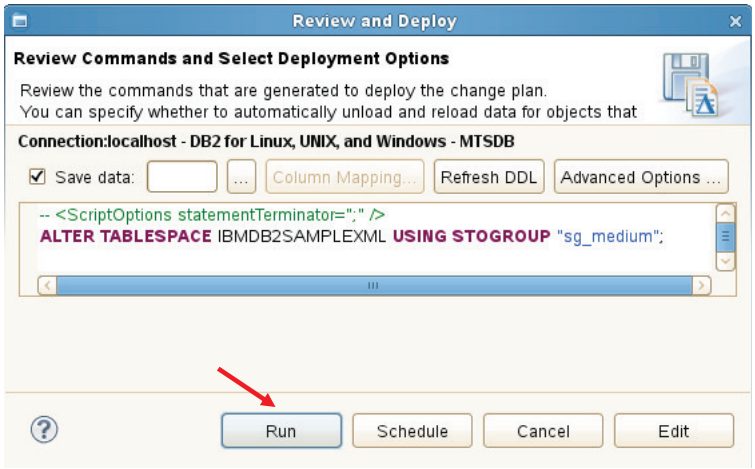


Figure 37 – Apply the changes to the table space

After the ALTER TABLESPACE statement is committed, containers are allocated on the new storage group's storage paths, the existing containers residing in the old storage group are marked as *drop pending*, and an implicit **REBALANCE** operation is initiated. This operation allocates containers on the new storage path and rebalances the data from the existing containers into the new containers. Once the move is complete, the old containers are dropped.

5. Copy the following query into the previously created SQL script, and then execute the script by pressing the run button in Data Studio, or press **F5** while the cursor is inside the SQL Editor.

```
SELECT varchar(tbsp_name, 30) as tbsp_name, storage_group_name  FROM
TABLE(MON_GET_TABLESPACE('',-2)) AS t WHERE t.tbsp_using_auto_storage = 1 and
t.storage_group_name = 'sg_medium'
```

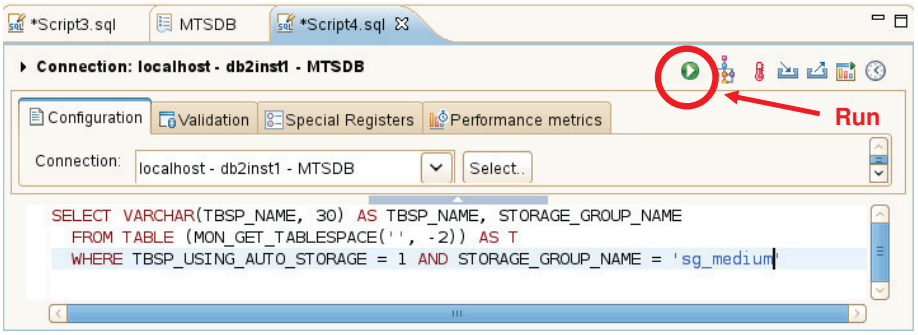


Figure 38 – SQL Query Textbox

Note the **IBMDB2SAMPLEXML** table space has successfully moved to storage group **sg_medium**.

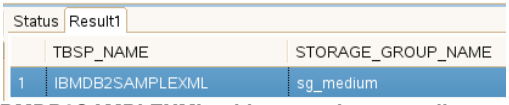


Figure 39 - IBMDB2SAMPLEXML table space in sg_medium storage group

13.5 Add/ Remove Storage Paths from a Storage Group

A storage group can be modified to add or remove storage paths as the need arises. The **ALTER STOGROUP** statement allows for the addition or removal of storage paths. We will use the table space from the previous example that now resides within the storage group **sg_medium**.

13.5.1 Add storage paths to a storage group

1. In the **Administration Explorer** view, click on **Storage Groups** folder. You will see a list of storage groups created for the **MTSDB** database.
2. Right-click the **sg_medium** storage group and select **ALTER**.
3. In the **Properties** view, select the **General** tab and click the **add Storage Path** button.

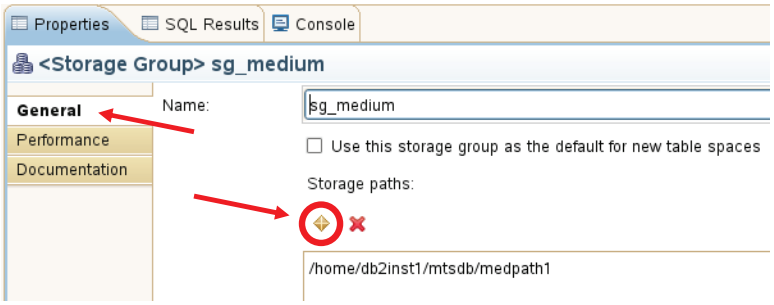


Figure 40 - Add new storage path

4. Navigate to **/home/db2inst1/mtsdb/**, select **medpath2** and click **OK**. Note the new path is added to the storage paths list.
5. Repeat step 3 and 4 to add a third storage path **/home/db2inst1/mtsdb/medpath3**.

   After adding both storage paths, your Properties view should look similar to the image below.

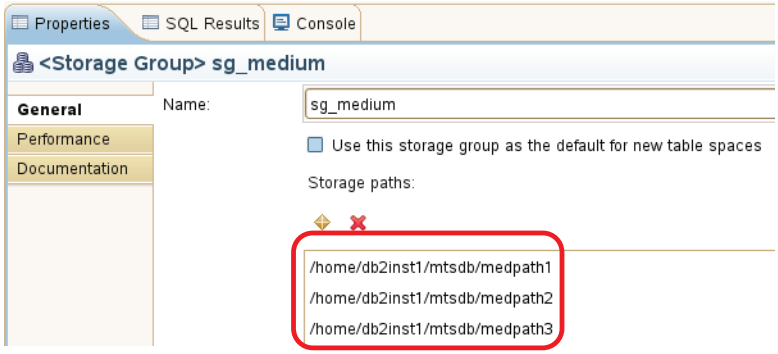


Figure 41 - sg_medium storage group with 2 new storage paths

6. Finally, click on **Review and Deploy (**  **)** button on the top right section of the screen, review the ALTER STOGROUP SQL statement and click **Run**.

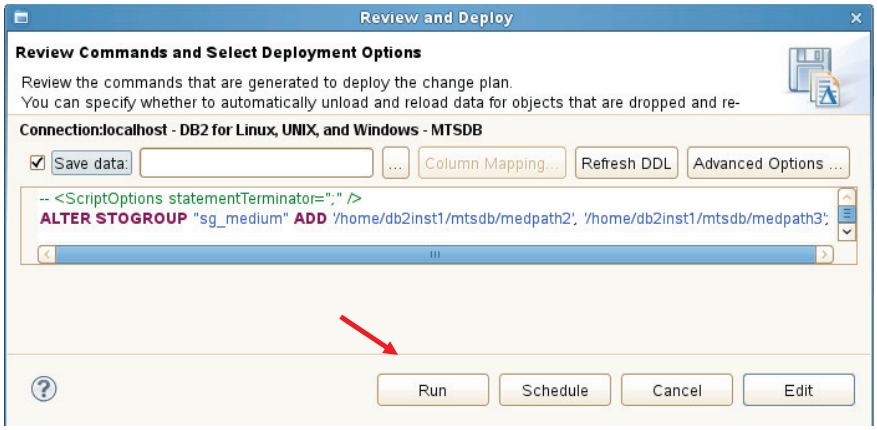


Figure 42 - Deploy the changes to the database

Attention: All paths assigned to a storage group should have similar media characteristics: underlying disks, latency, device read rate, and size. Non-uniform media characteristics may result in performance inconsistency.

After adding one or more storage paths to the storage group, you can optionally use the ALTER TABLESPACE statement to rebalance table spaces to immediately start using the new storage paths. Otherwise, the new storage paths are used only when there is no space in the containers on the existing storage paths.

7. To determine the table spaces which require an explicit REBALANCE, you can issue the command shown in section 3.4.2 step 5 to list the table spaces within the storage group **sg_medium**.

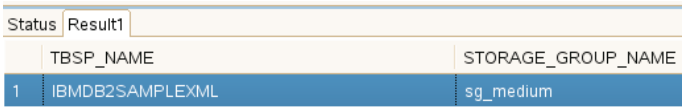


Figure 43 - Table spaces in sg_medium storage group

8. Copy the following query into the previously created SQL script, and then execute the script by pressing the run button in Data Studio, or press **F5** while the cursor is inside the SQL Editor.

```
ALTER TABLESPACE IBMDB2SAMPLEXML REBALANCE;
```

13.5.2 Remove storage path from a storage group

When intending to drop a storage path, you must rebalance all permanent table spaces that use the storage path. The rebalance is needed to move data off the path to be dropped. In this situation, the rebalance operation moves data from the storage path

that you intend to drop to the remaining storage paths and keeps the data striped consistently across those storage paths, maximizing I/O parallelism. To demonstrate this procedure, we will proceed to drop the newly added storage path.

1. To drop storage paths from a storage group, select the **Storage Groups** folder in the **Administration Explorer** view and locate the **sg_medium** storage group.
2. Right-click the **sg_medium** storage group and select **ALTER**.
3. In the **Properties** view, select the **General** tab, locate the storage path **/home/db2inst1/mtsdb/medpath3** and click the **Drop Storage Path** button.

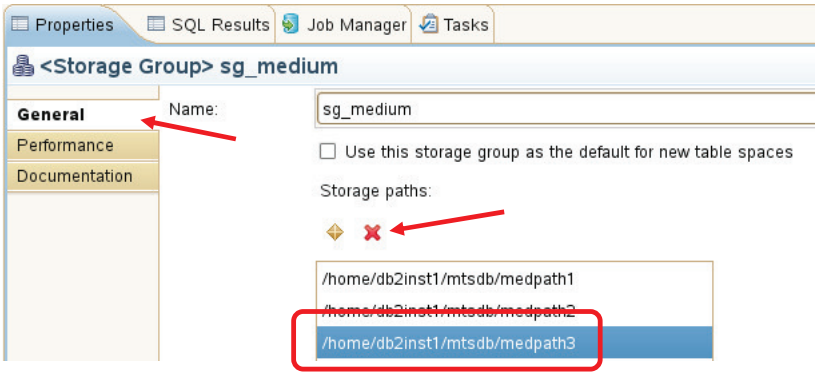


Figure 44 - Drop medpath3 storage path

4. Next, click on **Review and Deploy (**  **)** button on the top right section of the screen, review the ALTER STOGROUP SQL statement and click **Run**.

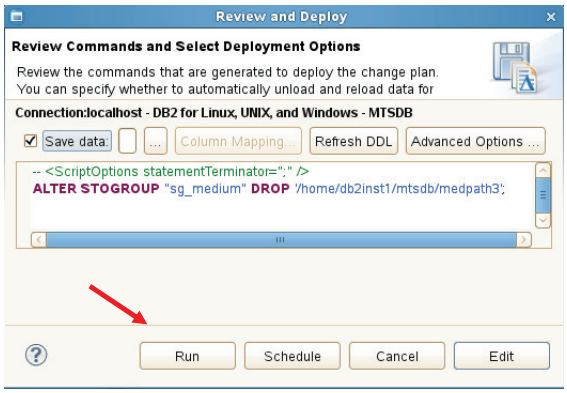


Figure 45 - Deploy the changes to the database

Note that the storage path **/home/db2inst1/mtsdb/medpath3** is now in **Drop Pending** state.

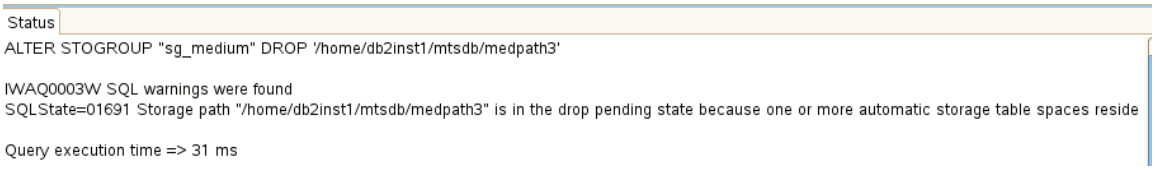


Figure 46 - Storage path medpath3 in drop Pending state

5. The following command shows you the storage path marked as **DROP_PENDING.** Copy the query into the previously created SQL script, and then execute the script by pressing the run button in Data Studi, or press **F5** while the cursor is inside the SQL editor.

```
SELECT VARCHAR(STORAGE_GROUP_NAME, 30) AS STOGROUP, VARCHAR(DB_STORAGE_PATH, 40)
AS STORAGE_PATH, DB_STORAGE_PATH_STATE AS STATE FROM
TABLE(ADMIN_GET_STORAGE_PATHS('',-1)) AS T WHERE T.STORAGE_GROUP_NAME =
'sg_medium'
```

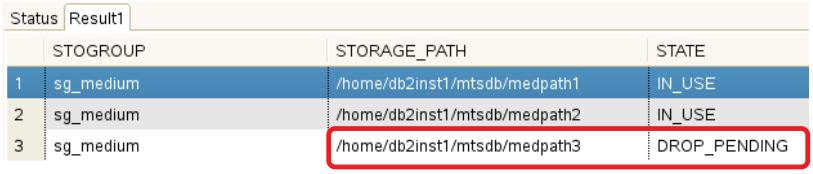


Figure 47 - Storage path medpath3 shown in DROP_PENDING State

To remove the path, rebalance the containers of the storage path being dropped.

6. Copy the query into the previously created SQL script, and then execute the script by pressing the run button in Data Studio, or press **F5** while the cursor is inside the SQL Editor.

```
ALTER TABLESPACE IBMDB2SAMPLEXML REBALANCE
```

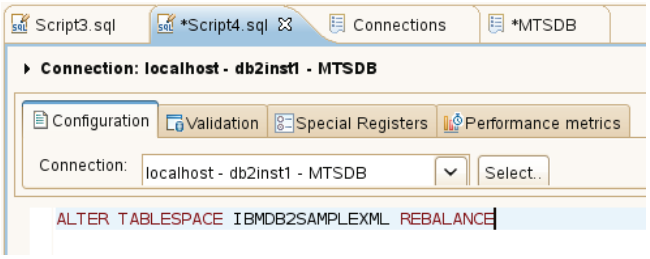


Figure 48 - Rebalancing IBMDB2SAMPLEXML table space

After the rebalance operation is complete, **/home/db2inst1/mtsdb2/medpath3** is removed from the storage group.

7. To confirm that the storage path was dropped, verify the assigned storage paths for storage group **sg_medium** by repeating step 5.

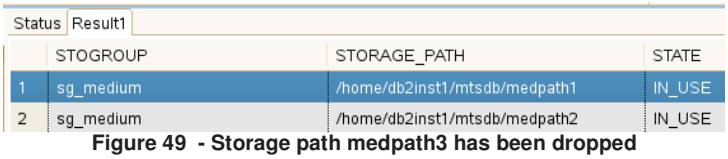


Figure 49 - Storage path medpath3 has been dropped

13.6 Drop a Storage Group

When your storage management needs require you to remove a storage group, you need to determine if that storage group is in use. Prior to the drop, all its table spaces need to be assigned to another storage group.

Attention: You cannot drop the current Default Storage Group

1. Find the table spaces that are using the storage group. In our case we are going to drop **sg_medium** and it currently holds the **IBMDB2SAMPLEXML** table space, as shown in the previous sections.

   In the case of temporary table spaces residing within the storage group, you would be required to drop the temporary table spaces and re-create them within the target storage group, where the data is being transferred to.
2. Click on **Table Spaces** folder, select **IBMDB2SAMPLEXML** table space.
3. Right-click the **IBMDB2SAMPLEXML** table space and select **ALTER** option.
4. In the **Properties** view, select the **Storage Options** tab, and then change the **Storage Group** to **IBMSTOGROUP**.
5. Finally, click on **Review and Deploy (** **)** button on the top right section of the screen, review the ALTER TABLESPACE SQL statement and click **Run**.

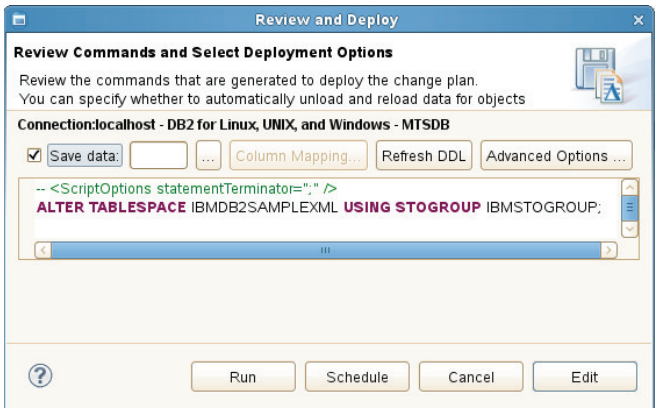


Figure 50 - Move IBMDB2SAMPLXML table space to IBMSTOGROUP storage group

In earlier operations, we have not shown a way to monitor the rebalance activity for the data movement involved in assigning a table space to a different storage group. The **MON_GET_REBALANCE_STATUS()** table function returns data for a table space only if a rebalance operation is in progress. Otherwise, no data is returned.

Since the size of the table spaces in this laboratory are small, the implicit rebalance operation is executed in factions of seconds, thus very difficult to monitor.

6. Copy the following query into the previously created SQL script, and then click on **Run** to execute the query.

```
SELECT tbsp_name, rebalancer_source_storage_group_name as sg_source,
rebalancer_target_storage_group_name as sg_target, rebalancer_status as status
from table(mon_get_rebalance_status('',-2)) as T WHERE
T.REBALANCER_SOURCE_STORAGE_GROUP_NAME= 'sg_medium'
```

As mentioned before, an empty result state indicates that the table space has finished moving to the new target storage group.

7. Next, check the table spaces being held by the **IBMSTOGROUP** storage group. Copy the following query into the previously created SQL script, and then click on **Run** to execute the query.

```
SELECT varchar(tbsp_name, 30) as tbsp_name, storage_group_name  FROM
TABLE(MON_GET_TABLESPACE('',-2)) AS t WHERE t.tbsp_using_auto_storage = 1 and
t.storage_group_name = 'IBMSTOGROUP'
```

You will see that the **IBMDB2SAMPLEXML** table space is now in the **IBMSTOGROUP** storage group.

Status | Result1

	TBSP_NAME	STORAGE_GROUP_NAME
1	SYSCATSPACE	IBMSTOGROUP
2	TEMPSPACE1	IBMSTOGROUP
3	USERSPACE1	IBMSTOGROUP
4	IBMDB2SAMPLEREL	IBMSTOGROUP
5	IBMDB2SAMPLEXML	IBMSTOGROUP
6	SYSTOOLSPACE	IBMSTOGROUP

Total 6 records shown

Figure 51 - Table spaces created by default within IBMSTOGROUP

Now you can proceed and drop the storage group **sg_medium**.

8. Click the **Storage Groups** folder, right-click the **sg_medium** storage group and select **DROP**.

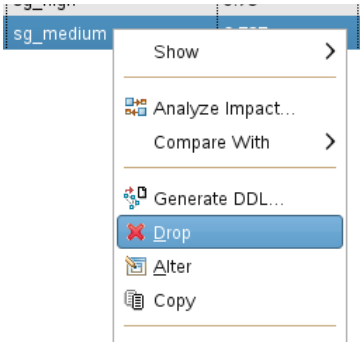


Figure 52 - Drop sg_medium storage group

9. Click on **Review and Deploy** () button on the top right section of the screen, review the DROP STOGROUP SQL statement and click **Run**.

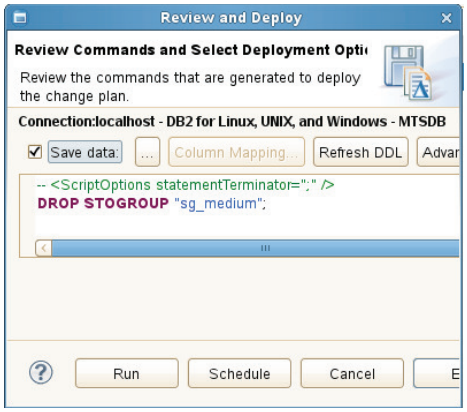


Figure 53 - Run the DROP STOGROOUP command

The storage group **sg_medium** was removed from the **MTSDB** database.

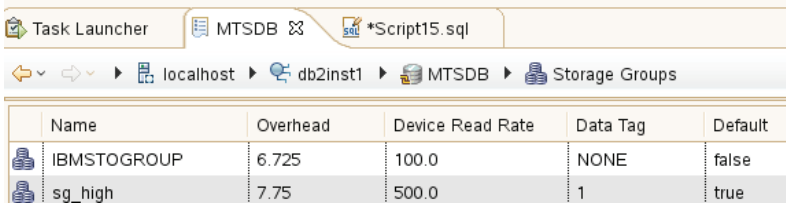


Figure 54 - sg_medium storage group removed

14 Multi-temperature Storage

Having covered the basics involved in managing storage groups and table spaces within them, we proceed to a scenario where storage of data is managed leveraging the concepts introduced in the earlier sections. We illustrate the use of storage groups with multi-temperature data.

Assume that you are the DBA for a business that does most of its processing on current-fiscal-quarter data. This business has enough solid-state drive (SSD) capacity to hold data for an entire quarter and enough Fiber Channel-based (FC) and Serial Attached SCSI (SAS) drive capacity to hold data for the remainder of the year.

You then take the following actions:

- Create two storage groups for the two types of storage within your organization
- Create table spaces for data accumulated within each quarter
- Assign the table space containing the data for the current quarter to the **sg_hot** storage group
- Assign the table space containing the data for the previous quarters to the **sg_warm** storage group

14.1 Create Storage Groups and Table Spaces for Q1 and Q2

In this section you will create all the necessary database objects required in this scenario. The following script creates the two storage groups **sg_hot** and **sg_warm**. The table spaces for Q1 and Q2 are created and populated with sales data for each quarter.

1. In Data Studio, click menu **File > Open File** on the top left hand corner.
2. Navigate to /**home**/**db2inst1**/**Documents**/**LabScripts**/**Multi_Temp_Storage** directory and open the file named **Create_Grps_Tbspcs_Q1Q2.sql.** If prompted, press **Ok** to use the default statement terminator (semi-colon).

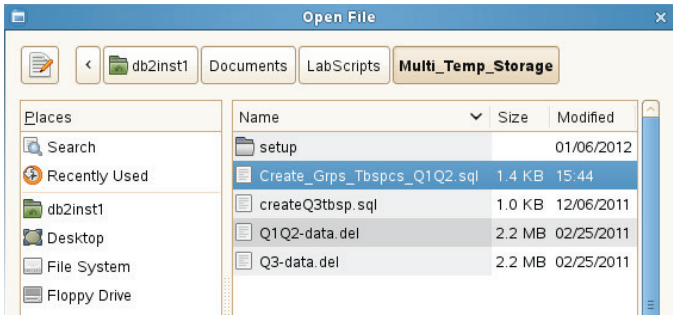


Figure 55 - Open file Create_Grps_Tbspcs_Q1Q2.sql

3. Before running the file, click on the **No Connection** link on top left corner of the script as shown below.

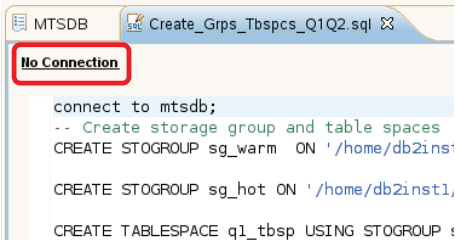


Figure 56 - No Connection link

4. Select **MTSDB** database and click **Finish**. **Do not run the script yet!**.
5. From the **Run Method** drop down menu, select **Command Line Processor**.

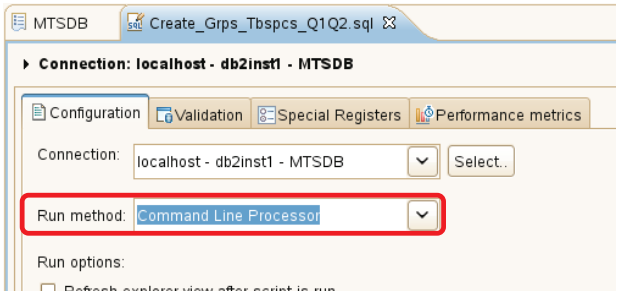


Figure 57 - Select Command Line Processor run method

6. Now execute the script by pressing the run button in Data Studio, or press **F5** while the cursor is inside the SQL Editor.

   Upon completion, you will have two new table spaces. Each table space resides in a storage group created to cater to its frequency of access requirements.

Attention: You might notice that some rows failed, this is due to the data set containing rows out of the date range required for the Q1 and Q2 quarters. These rows can be ignored.

14.2 Start of the New Quarter

As the third quarter sales data is available, data is attached to the existing partitioned table. The one set of the new quarter requires for data to be moved between storage groups. To do this, data for the previous quarter is moved to the slower storage, **sg_warm** and a new table space for Q3 is created within the faster storage group **sg_hot**.

- Move **q2_tbsp** to the slower storage group **(SG_WARM)**
- Create new table space **q3_tbsp** for data accumulated for the third and current quarter

- Assign the table space containing the data for the current quarter to the **SG_HOT** storage group
- Create a **Recent_Sales** table and populate with data for Q3
- Attach the data within **Recent_Sales** data to the **Sales** table as a partition of data for Q3

1. In the **Administration Explorer** view, click on the **Table Spaces** folder and press F5 to refresh the view, then right-click the **q2_tbsp** and select **ALTER**.
2. In the **Properties** view, select the **Storage Options** tab, then change the **Storage Group** to **SG_WARM**.

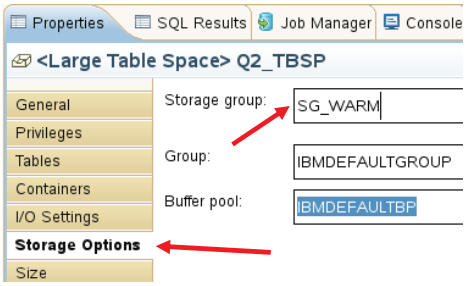


Figure 58 – Select SG_WARM storage group

3. Click on **Review and Deploy** () button on the top right section of the screen, review the ALTER TABLESPACE SQL statement and click **Run**.

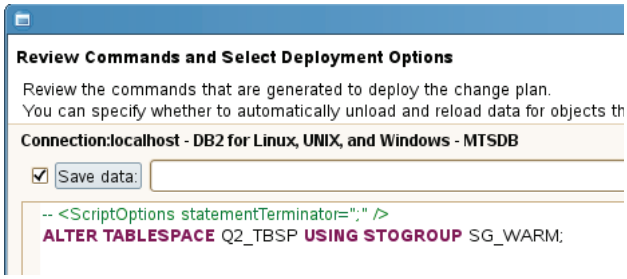


Figure 59 - Change Q2_TBSP tablespace's temperature

4. Copy the following query into the previously created SQL script, and then click on **Run** to execute the query.

```
SELECT varchar(tbsp_name, 30) as tbsp_name, storage_group_name  FROM
TABLE(MON_GET_TABLESPACE('',-2)) AS t WHERE t.tbsp_using_auto_storage = 1 and
t.storage_group_name = 'SG_WARM'
```

Note that **Q2_TBSP** table space now resides in the **SG_WARM** storage group.

Figure 4: Q2_TBSP rebalanced on SG_WARM storage group

5. In Data Studio, click menu **File > Open File** on the top left hand corner. Navigate to **/home/db2inst1/Documents/LabScripts/Multi_Temp_Storage** directory and open the file named **CreateQ3tbsp.sql.** If prompted, press Ok to use the default statement terminator (semi-colon).
6. Before running the file, click on the **No Connection** link on top left corner of the script, then select **MTSDB** database and click **Finish**.
7. From the **Run Method** drop down menu, select **Command Line Processor**, then execute the script by pressing the run button in Data Studio, or press **F5** while the cursor is inside the SQL Editor.

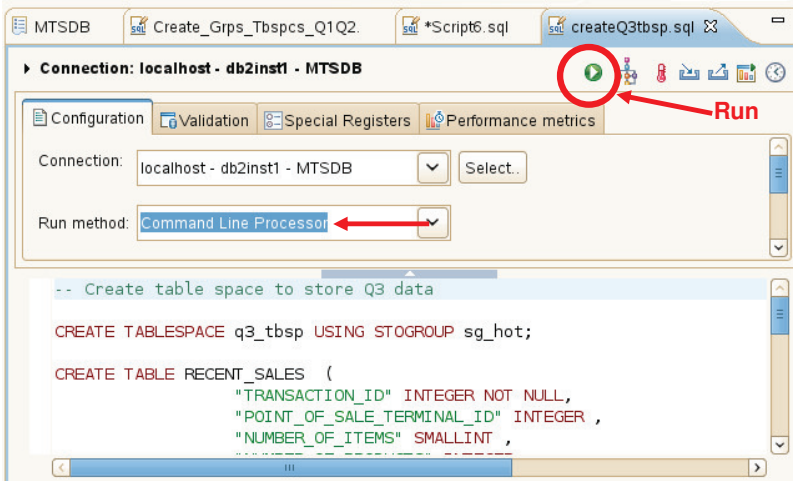


Figure 60 - Create Q3_TBSP table space

Upon completion, you will have a new table space within the storage group **SG_HOT**.

This concludes the process of moving data between storage groups as it ages within an organization. As the data residing in **q2_tbsp** grew less important, it was moved to slower storage. Faster storage being used for storage group **sg_hot** was reserved for the current quarter **q3_tbsp.**

7. Copy the following query into the previously created SQL script, and then click on **Run** to execute the query..

```
SELECT varchar(tbsp_name, 30) as tbsp_name, storage_group_name  FROM
TABLE(MON_GET_TABLESPACE('',-2)) AS t WHERE t.tbsp_using_auto_storage = 1 and
t.storage_group_name = 'SG_HOT' or t.storage_group_name = 'SG_WARM'
```

Status | Result1

	TBSP_NAME	STORAGE_GROUP_NAME
1	Q1_TBSP	SG_WARM
2	Q2_TBSP	SG_WARM
3	Q3_TBSP	SG_HOT

Figure 61 - Organization of table spaces among storage groups

15 Summary

The intent of this lab was to acquaint the user with the new concepts introduced in DB2 10. *Multi-temperature data* management allows the user to configure the database in a manner which optimizes the use of fast storage and effectively reduces storage costs. The addition of storage groups as sets of storage paths with similar qualities now allows the user to map different classes of storage according to their respective performance and reliability requirements.

To further improve the performance of data, the power of data tags and the new data-centric view for workloads can be leveraged. Different table spaces within the same storage group can be prioritized based on their data tag values.

16 Cleanup

1. To clean your environment after completing the exercise, execute the command below. Use “password” as password.

```
su -
cd /home/db2inst1/Documents/LabScripts/Multi_Temp_Storage/setup
./cleanup.sh
```

2. If you would like to redo this lab in the future, please execute the following commands in a terminal window as root:

```
cd /home/db2inst1/Documents/LabScripts/Multi_Temp_Storage/setup
./config.sh
```

IBM Canada
8200 Warden Avenue
Markham, ON
L6G 1C7
Canada

DB2® Essential Maintenance & Practical Autonomics

Hands-On Lab

Table of Contents

17 Introduction

Today's business challenges require that databases be optimized for the best possible performance. The database engine must be highly tuneable and easily configurable for the best performance regardless of the skill level of the DBA. To face these challenges, DB2 comes with a rich set of autonomic features for tuning the configuration parameters adaptively and accurately.

18 About this Lab

In this lab, you will explore some of the autonomic computing capabilities of DB2 that help reduce skill requirements while ensuring reliability and availability of your mission critical environment. After completing this lab, you will be able to use the following tools:

3. Configuration Advisor
4. REORGCHK, REORG and RUNSTATS
5. Utility Throttling
6. Automatic Maintenance
7. Design Advisor
8. Self-Tuning Memory Manager (STMM)
9. Data Movement Tools

The exercises in this lab are designed to give you a good idea of the features available with DB2 autonomics. Note that this lab will not necessarily cover the approach you would take to initially configure a data server; however, it will give you a good idea of some necessary steps required to use each of these useful autonomic features.

19 Basic Set up and Start of Lab

The lab consists of a series of tasks, each highlighting one or more concepts of the DB2 Practical Autonomics described in the introduction.

Some of the commands and queries required to perform certain parts of certain tasks are quite long. In order to avoid having to type these longer commands, we have provided you with scripts for some of these tasks. They are located in the `/home/db2inst1/Documents/LabScripts/Autonomics` directory.

19.1 Environment Setup Requirements

To complete this lab you will require the following:

10. DB2 10 VMware® image
11. VMware Workstation 6.5 or later

19.2 Preparation Step

3. Start the VMware image by clicking the 

button in VMware Workstation if it is not already on.
4. At the login prompt, login with the following credentials:
    12. Username: **`db2inst1`**
    13. Password: **`password`**

5. Open a terminal window as follows by right-clicking on the **Desktop** and choosing the **Open in Terminal** item.

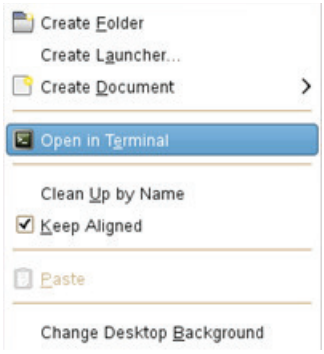


Figure 62 – Opening a Terminal

19.3 Launching Data Studio

1. Double-click on the **IBM Data Studio** icon on the **Desktop**.

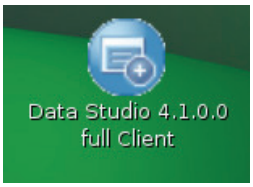


Figure 63 – IBM Data Studio Desktop Icon

2. In the **Select a workspace** dialog, accept the default path and click **OK**.

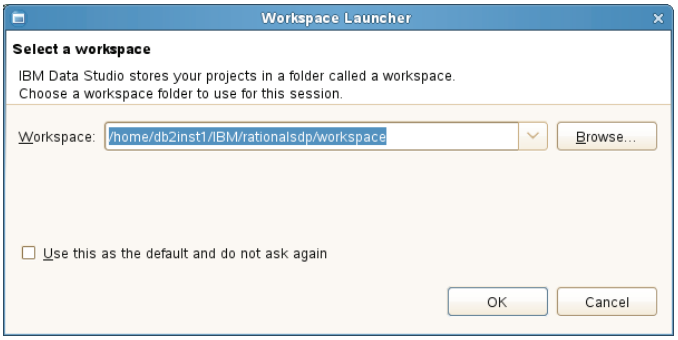


Figure 64 – Workspace Launcher

3. Data Studio should now start in the **Database Administration** perspective as shown below.

Figure 65 – IBM Data Studio, Database Perspective

If you are not in Database Administration perspective, you can easily change the perspective of Data Studio by click the **Open Perspective** button as shown below and selecting **Database Administration** from the list.

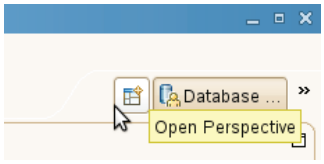


Figure 66 – Changing Perspectives

19.4 The LABDB Database

Throughout this lab we will be working with a database named LABDB that contains several table spaces and is managed by automatic storage. We will use this database to describe all of the autonomics components described above by executing various queries on this database throughout the lab. **This database has already been set up and is ready to use**.

4. Open up a new terminal window or use the terminal window from the beginning of the lab.
5. Change your current directory to the directory containing all the scripts for this lab:

```
cd /home/db2inst1/Documents/LabScripts/Autonomics
```

6. Use gedit to view the contents of the create_db_labdb.sql script by executing the following command:

```
gedit create_db_labdb.sql &
```

The following should display:

```
drop db labdb;
create db labdb on /db2fs/db2inst1/sp1 , /db2fs/db2inst1/sp2 using codeset UTF-8
territory US;
connect to labdb;
create schema vps;
create schema dupvps;
create bufferpool VPSBUFF immediate size automatic pagesize 16k;
create tablespace VPSLARGE pagesize 16k managed by database using (file
'/db2fs/db2inst1/tempspace' 300) bufferpool VPSBUFF AUTORESIZE YES;
create tablespace vpsspc1 pagesize 16k managed by automatic storage autoresize yes
bufferpool VPSBUFF;
create tablespace vpsspc2 pagesize 16k managed by automatic storage bufferpool
VPSBUFF;
create tablespace vpsspc3 pagesize 16k managed by automatic storage bufferpool
VPSBUFF;
update dbm cfg using SHEAPTHRES 0;
```

Note:

1. **Automatic Storage is enabled by default when creating a database unless it is explicitly turned off.**
2. **The database manager SHEAPTHRES parameter is set to zero. This is to enable sort memory tuning (SORTHEAP) for optimal configuration parameter settings when using the configuration advisor.**

20 Configuration Advisor

DB2 configuration parameters play an important role in performance as they affect the operating characteristics of a database or database manager. The Configuration Advisor makes recommendations on the initial settings for configuration parameters that can be easily adopted by inexperienced administrators or fine-tuned by more experienced administrators. We will see in this section how the Configuration Advisor (invoked via the **AUTOCONFIGURE** command) can generate values for various performance critical parameters.

20.1 Optimize Database Using Configuration Advisor

A good place to start with this database is to run the **AUTOCONFIGURE** command on a database which will recommend enablement of the Self Tuning Memory Manager (STMM). We will be covering STMM later in this lab.

Although when a database is created, the Configuration Advisor runs automatically, explicitly invoke the Configuration Advisor using the command below.

1. Execute the following two commands in a terminal:

```
db2 connect to labdb
db2 "AUTOCONFIGURE APPLY DB AND DBM"
```

More configuration parameters are available for the AUTOCONFIGURE command; however, by not specifying them, the defaults are accepted. The following should output:

```
        Former and Applied Values for Database Manager Configuration

Description                                  Parameter   Former Value        Applied Value
-------------------------------------------------------------------------------------------
Application support layer heap size (4KB)   (ASLHEAPSZ) = 15                15
No. of int. communication buffers(4KB)(FCM_NUM_BUFFERS) = AUTOMATIC          AUTOMATIC
Enable intra-partition parallelism     (INTRA_PARALLEL) = NO                NO
Maximum query degree of parallelism   (MAX_QUERYDEGREE) = 1                 1
Agent pool size                        (NUM_POOLAGENTS) = AUTOMATIC(100)    AUTOMATIC(100)
Initial number of agents in pool       (NUM_INITAGENTS) = 0                 0
Max requester I/O block size (bytes)          (RQRIOBLK) = 32767            32767
Sort heap threshold (4KB)                  (SHEAPTHRES) = 0                 0


        Former and Applied Values for Database Configuration

Description                                  Parameter   Former Value        Applied Value
-------------------------------------------------------------------------------------------
Default application heap (4KB)               (APPLHEAPSZ) = 256              256
Catalog cache size (4KB)                (CATALOGCACHE_SZ) = 30               300
Changed pages threshold                 (CHNGPGS_THRESH) = 80                80
Database heap (4KB)                           (DBHEAP) = 1200               4465
Degree of parallelism                       (DFT_DEGREE) = 1                1
…                                                     … = …                 …
```

Note: These results may vary since the system defaults are being used by the `AUTOCONFIGURE` statement. Also, the above command can be executed anytime if you feel your system performance is not optimized to its maximum capabilities.

2. Restart the instance for the changes to take affect by issuing the following commands in the terminal:

```
db2 FORCE APPLICATION ALL
db2stop
db2start
db2 connect to labdb
```

Note: We issued the first command to force/release all applications (users) off the database for proper restart of the instance.

You can view and verify the database and database manager configuration parameters by issuing the statements **`GET DB CFG`** and **`GET DBM CFG`** respectively.

21 REORGCHK, REORG and RUNSTATS

Before you continue with the rest of this lab, you need to load sample data into LABDB. To achieve this, a script has been written for this task.

1. In the terminal window, ensure that you are in directory /home/db2inst1/Documents/LabScripts/Autonomics, execute script PopulateDB.sh as the following

```
./PopulateDB.sh
```

2. You will be prompted for some information regarding the size of the relational data that you would like to load. For this particular section in the lab we will input 100MB as the amount of data that we would like to insert into our database.

```
Size of Relational Data (MBs):
100
```

After some time, the following should display, indicating that everything has been processed successfully:

```
- Summary -
OLTP Size: 100
XML Size: 0
User: db2inst1
Pass: password
Port: 50001

Creating data tables...[DONE]
Starting OLTP Data Creation...[DONE]
Loading data into tables...[DONE]
Performing integrity check...[DONE]
Creating indexes...[DONE]
Creating stored procedures...[DONE]
LABDB Database population completed.
Removing flatfiles...[DONE]
LABDB Database population completed.
All DB2 commands have been logged to VPS_Setup.log
```

All of the above steps should complete successfully, with the **[DONE]** tag beside them

21.1 REORGCHK

The `REORGCHK` command interrogates database metadata and statistics on the database to determine and report on if tables or indexes, or both, need to be reorganized or cleaned up.

You will run the `REORGCHK` command twice. The first time you will use the current statistics for the database then you will supply an option that will update the statistics before checking to see if a `REORG` or cleanup is recommended.

1. Run the `reorgchk` command to verify whether reorganizing the table is necessary.

```
db2 REORGCHK CURRENT STATISTICS ON TABLE vps.stock
```

2. Results show that the table does not require a reorg. But a reorg was being recommended for one of the indexes **F4**.

```
F4: CLUSTERRATIO or normalized CLUSTERFACTOR > 80

… …

CLUSTERRATIO or normalized CLUSTERFACTOR (F4) will indicate REORG is necessary
for indexes that are not in the same sequence as the base table. When multiple
indexes are defined on a table, one or more indexes may be flagged as needing
REORG.  Specify the most important index for REORG sequencing.
```

3. The `UPDATE STATISTICS` parameter calls the `RUNSTATS` routine first to update table and index statistics, and then uses the updated statistics to determine if table or index needs reorganization. Enter the following command on a single line.

```
db2 REORGCHK UPDATE STATISTICS ON TABLE vps.stock
```

Both files show similar results for both commands. This is due to the fact that a small table with relatively simple statistics has been checked for reorganization.

Note: You should note that updating the statistics at the wrong time can result in moderate to significant resource contention on your system. An alternate approach would be to enable automatic statistics collection. This will ensure that statistics are accurate, current and run with an internally capped limit (about 7%) on the amount of resources it will consume. The `RUNSTATS` command when run on its own has an option to run with throttled resource consumption (Utility Impact Priority). It can also employ sampling techniques to reduce its impact while maintaining accuracy. The update statistics command will use the default setting for updating statistics and has no throttling available, but it is a pretty fast way to update statistics for all tables or all tables in a particular schema.

21.2 Reorganizing Tables/Indexes (REORG)

The REORG command is used to reorganize (defragment) an index or a table.

INDEX
Reorganizing indexes rebuilds the index's leafs eliminating deleted RIDs, reclaiming space and possibly reducing the number of levels in structure of the index.

1. Execute the following in the terminal window to reorganize all indexes for table INVENT.

```
db2 REORG INDEXES ALL FOR TABLE vps.invent
```

A successful message should appear

```
DB20000I The REORG command completed successfully.
```

TABLE
Running a reorg for a table, reconstructs, or rebuilds the rows, compacting the table, eliminating fragmented and unused space. In addition, if your table is defined with row compression ON, you can rebuild the data compression dictionary and recompress the data based on the new dictionary. Lastly a reorganization of a table can be used to physically sort the data according to a specified clustering index to improve performance.

2. Execute the following in the terminal window to reorganize table INVENT.

```
db2 REORG TABLE vps.invent
```

A successful message similar to the one above should be displayed.

21.3 Updating Optimizer Statistics (RUNSTATS)

Database statistics should be gathered regularly as required and preferably in a throttled mode so as to not use resources excessively in the process. There are other special circumstances when you may want to explicitly run an update of a table's and indexes' statistics:

1. After a table had a lot of updates like a batch update for a new column.
2. A reload of data or after you reorganize the table.

You can base the statistics on a sample or a percentage of the data in a *table* or an *index*. DB2's `RUNSTATS` command offers two types of sampling: Bernoulli and System. Bernoulli sampling evaluates a sample set of rows from the table. System sampling evaluates a sample set of data pages. System sampling introduces less overhead than Bernoulli sampling but may not produce as accurate results when your data is highly clustered since it may sample pages where data is clustered such that statistics are skewed

1. Enter the following command in your terminal window to update STOCK table statistics on 1.5 percent of the data pages and index statistics on 2.5 percent of the index pages. Both table data pages and index pages are sampled.

```
db2 "RUNSTATS ON TABLE vps.stock AND INDEXES ALL TABLESAMPLE SYSTEM(1.5)
INDEXSAMPLE SYSTEM(2.5) "
```

A successful message should appear

```
DB20000I The RUNSTATS command completed successfully.
```

22 Set Up Automatic Maintenance for the LABDB Database

DB2 provides automatic maintenance capabilities for performing database backups, keeping statistics current and reorganizing tables and indexes as necessary. Performing maintenance activities on your databases is essential to ensure that it is optimized for performance and recoverability. Enablement of the automatic maintenance features is controlled using the automatic maintenance database configuration parameters. These are a hierarchical set of switches to allow for simplicity and flexibility in managing the enablement of these features.

Automatic maintenance is enabled by default with the database configuration parameter auto_maint, which is the parent parameter of all the automatic maintenance database configuration parameters. Most of these child parameters are turned off by default, but auto_runstats is enabled by default.

Configure Automatic Maintenance for other maintenance activities like BACKUP, REORG, STATS_PROF and PROF_UPD by following the sections below.

22.1 Connect to the LABDB Database

1. Go back to Data Studio and connect to the LABDB database by navigating to the Administration Explorer on the top left hand side of Data Studio. Navigate under All Databases > localhost > 5001 to find the LABDB database. Right-click on LABDB and click Connect.

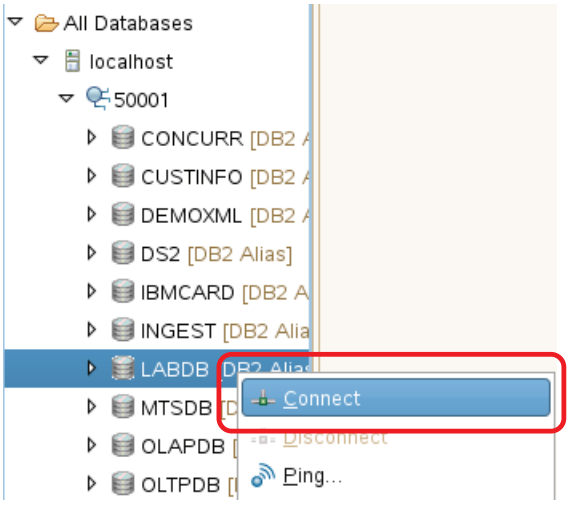


Figure 67 – Connecting to a Database

In the Properties for LABDB window, specify the following credentials:

14. Username: **`db2inst1`**
15. Password: **`password`**
16. Host: **`localhost`**
17. Portnumber: **`50001`**

2. After the connection has successfully completed, launch DB2 Automatic Maintenance by right-clicking on the LABDB artefact and select Set Up and Configure > Configure Automatic Maintenance….

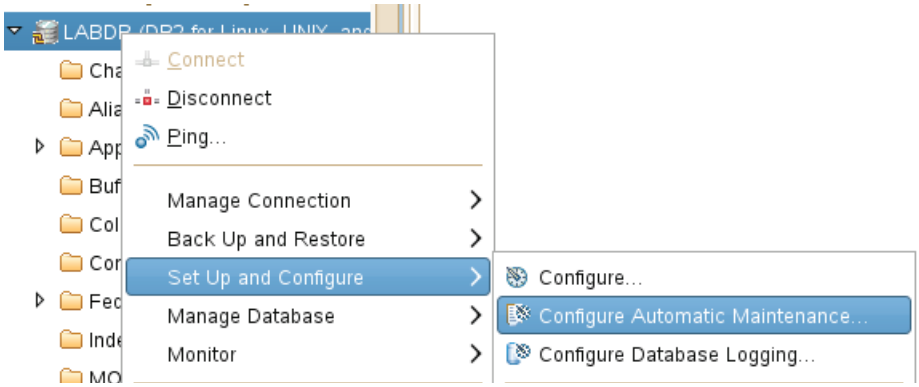


Figure 68 – Selecting to Configuring Automatic Maintenance

3. The Configure Automatic Maintenance LABDB tab will appear.

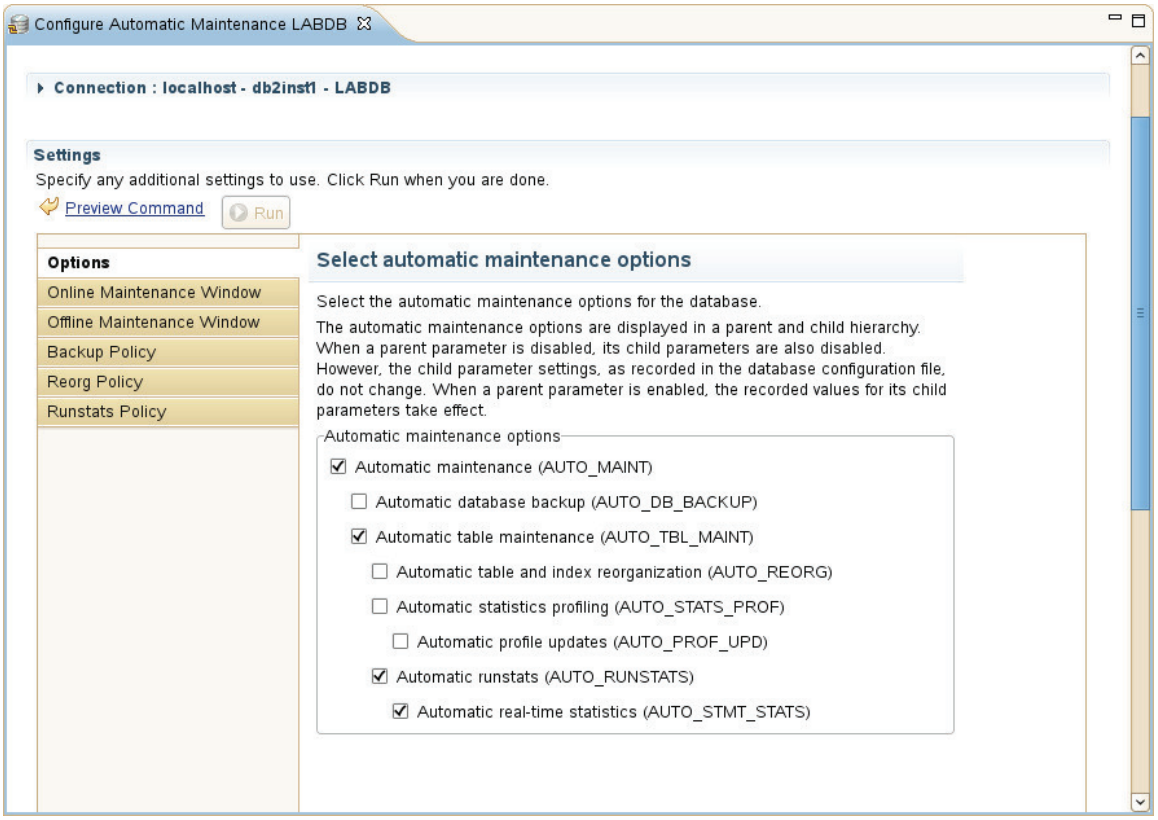


Figure 69 – Configuring Automatic Maintenance Window

22.2 Selecting Automatic Maintenance Options

Configure Automatic Maintenance by following the below steps:

1. Read the Introduction section at the top of the tab to learn about DB2 automatic maintenance. If not already selected, click on the **Options** menu item to receive a list of all maintenance options that could be activated.

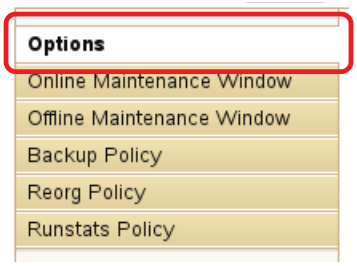


Figure 70 – Options for Automatic Maintenance

2. Select the maintenance activities that you wish to automate. In this case, select all activities by clicking the radio box beside the name (`AUTO_MAINT`, `AUTO_DB_BACKUP`, `AUTO_TBL_MAINT`, `AUTO_REORG`, `AUTO_STATS_PROF`, `AUTO_PROF_UPD`, `AUTO_RUNSTATS`, `AUTO_STMT_STATS`). As mentioned in the introduction, some of these are already selected because they are activated by default when creating a database.

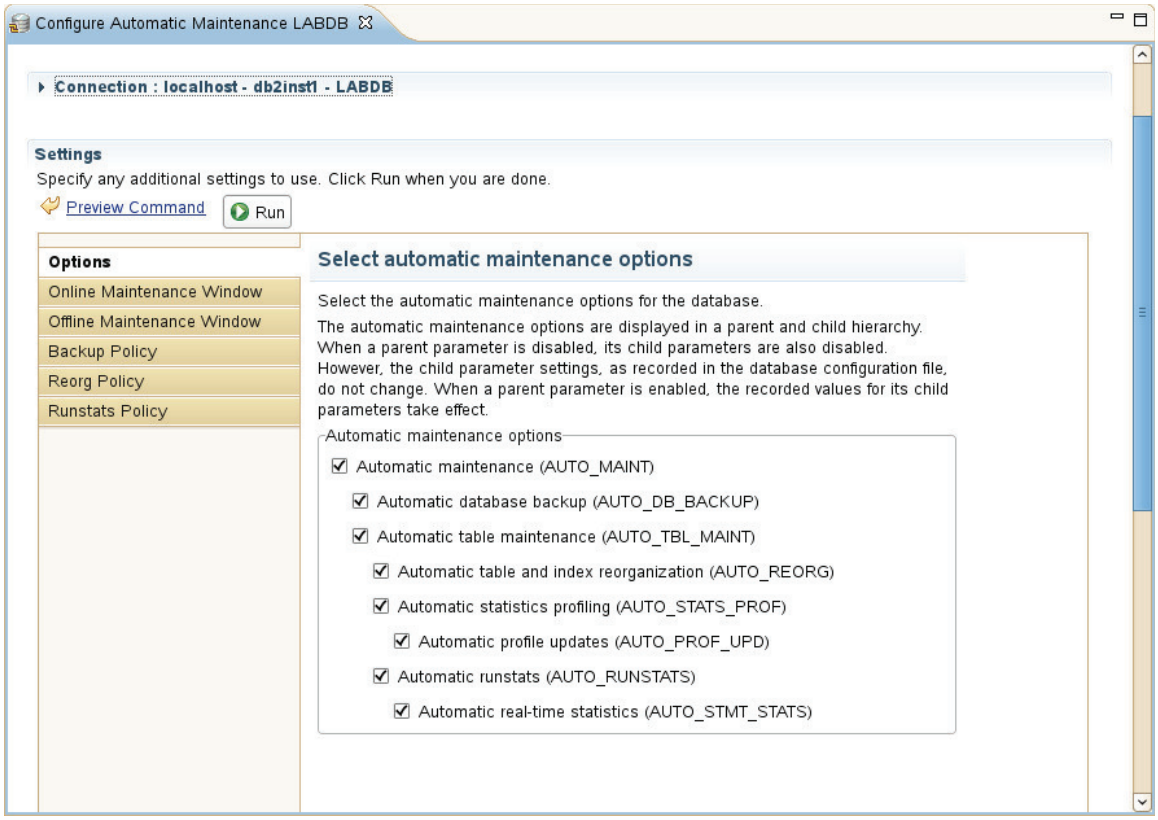


Figure 71 – Selecting Automatic Maintenance Options

Note: If AUTO_MAINT is turned off then all other options will become disabled. This is because this option is necessary for the others to be active, hence the hierarchy. A similar situation occurs for the AUTO_TBL_MAINT option and AUTO_REORG, AUTO_STATS_PROF, AUTO_PROF_UPD, AUTO_RUNSTATS, and AUTO_STMT_STATS.

22.3 Selecting the Online Maintenance Window

By defining the online maintenance window we will be able to select a time when the maintenance activities can be performed while the database is online. This is only used when DB2 determines that maintenance is necessary.

1. Select the **Online Maintenance Window** menu option to receive the following screen.

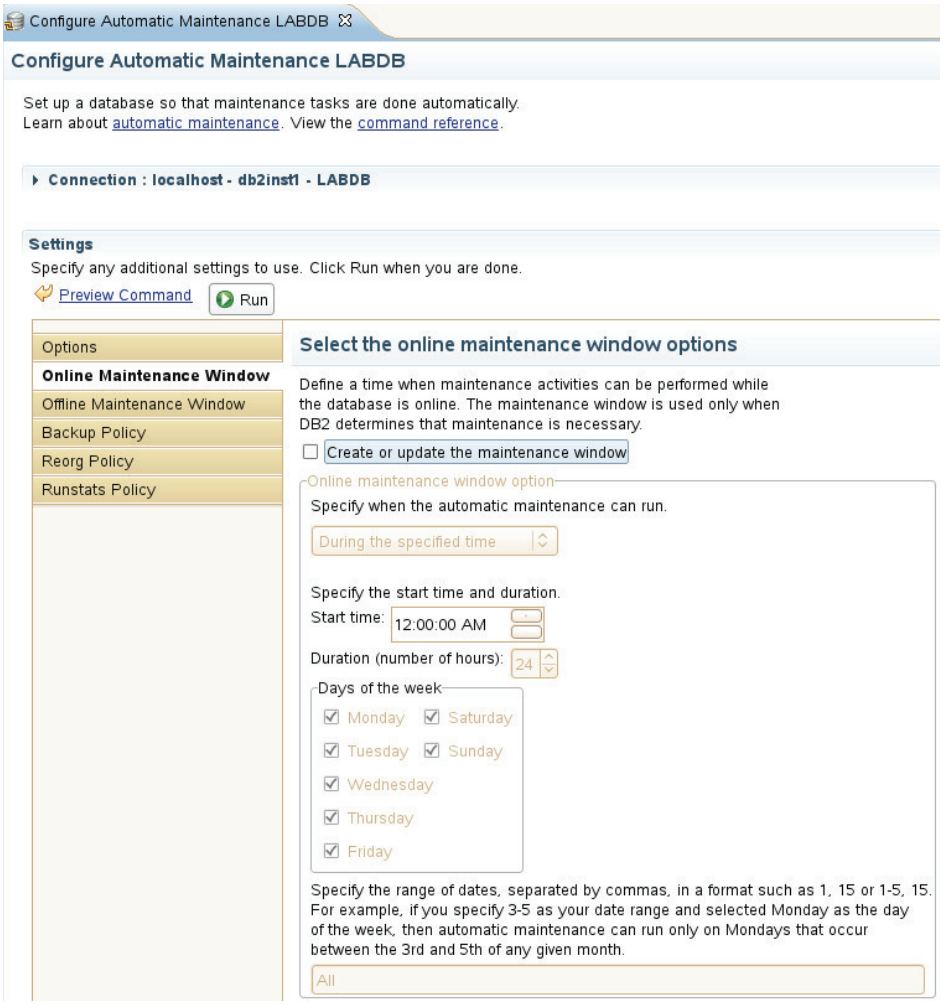


Figure 72 – Selecting the Online Maintenance Window

2. Select the **Create or update the maintenance window** checkbox to select a time slot.
3. Specify any Start Time and Duration for the online maintenance activity. You can also specify the Days of the week or Days of the Month for these activities to run. Select any days/months you would like.

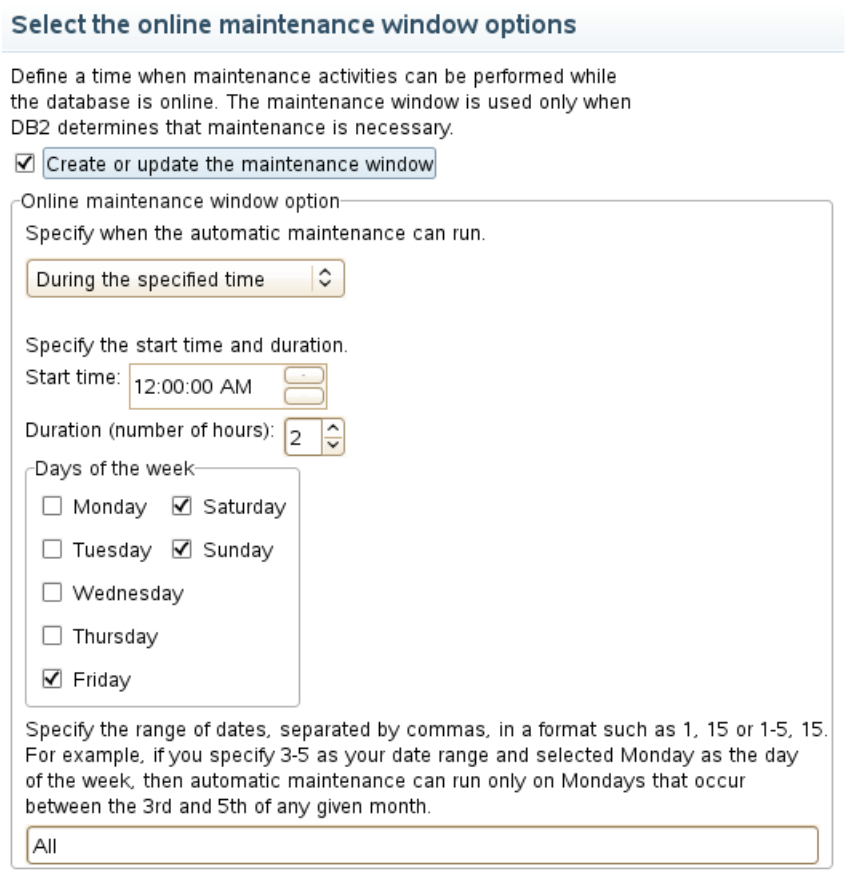


Figure 73 – Selecting the Online Maintenance Window Options

For the purpose of this document we selected:

18. **Start time**: 12:00:00 AM
19. **Duration (number of hours)**: 2
20. **Days of the week**: Friday, Saturday and Sunday
21. **Days of the month**: All

Review the maintenance windows you have selected.

22.4 Selecting the Offline Maintenance Window

It is also possible to define an offline maintenance window to perform maintenance on the database when it goes offline.

1. Select the **Offline Maintenance Window** menu option to receive the following screen.

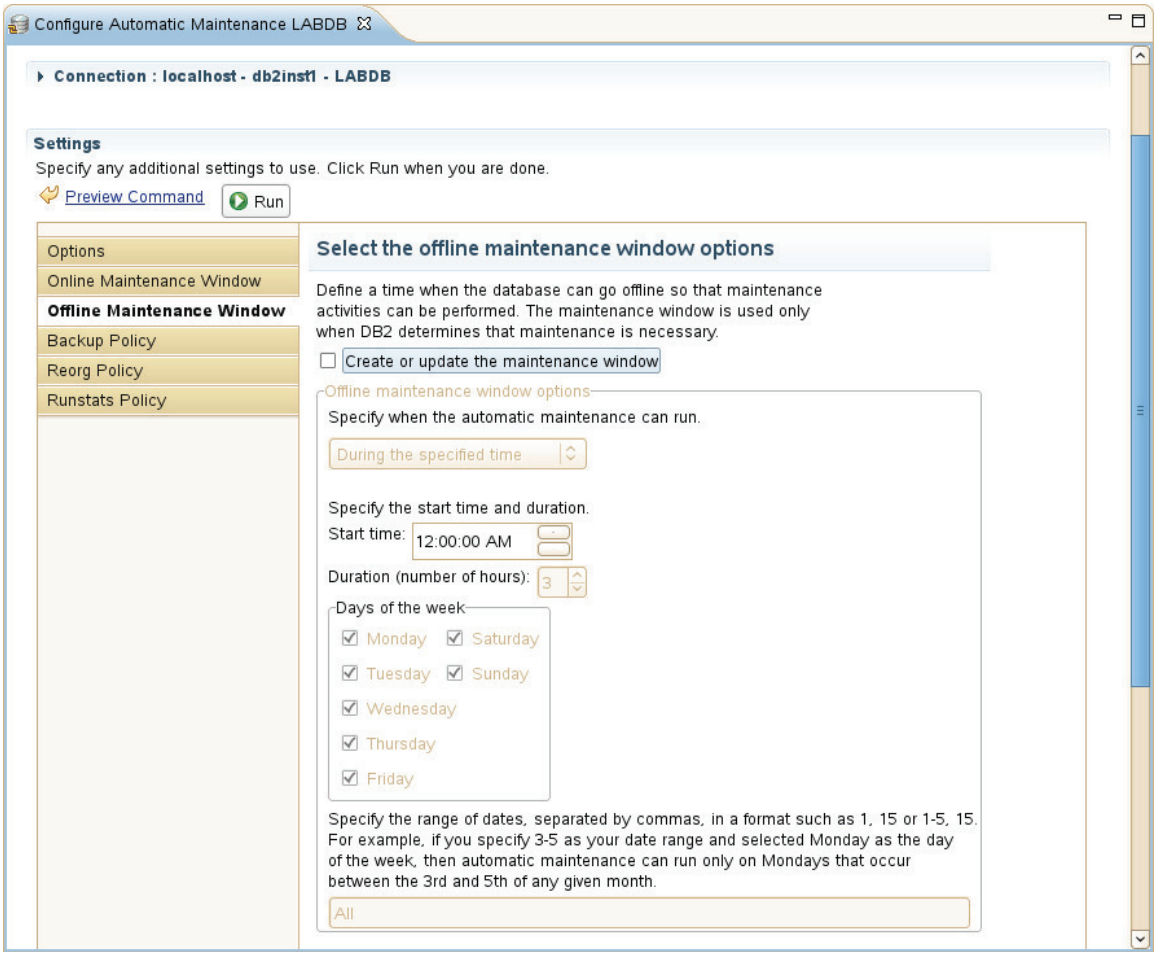


Figure 74 – Selecting the Offline Maintenance Window

2. Select the **Create or update the maintenance window** checkbox to select a time slot.
3. Specify any Start Time and Duration for the online maintenance activity. You can also specify the Days of the week or Days of the Month for these activities to run. Select any days/months you would like.

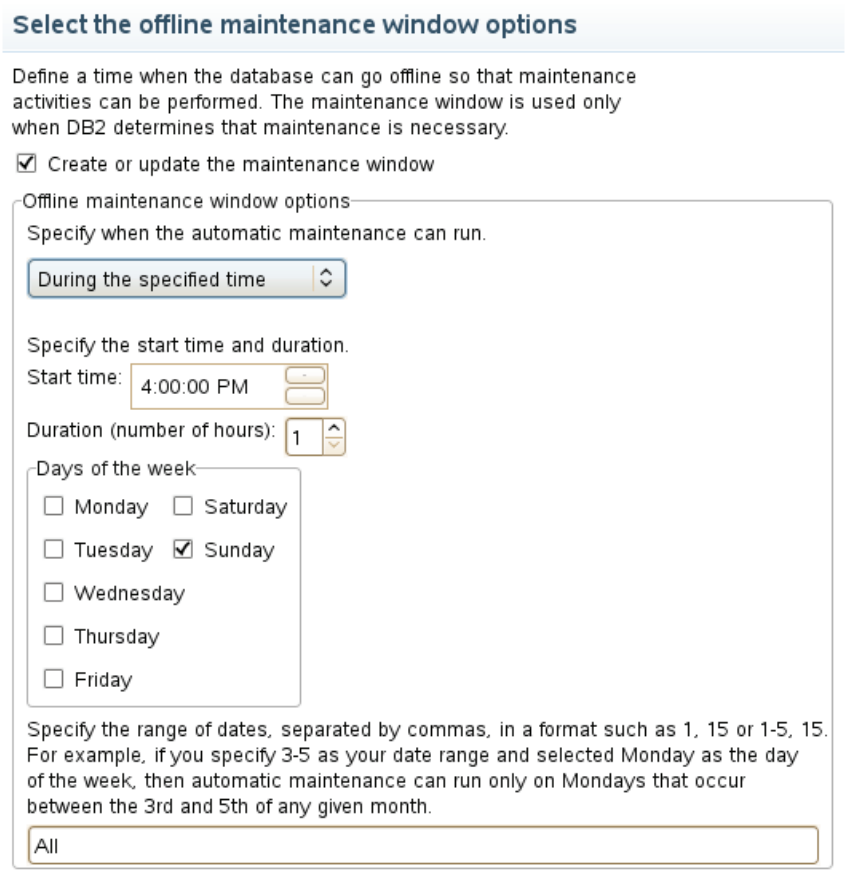


Figure 75 – Selecting the Online Maintenance Window Options

For the purpose of this document we selected:

22. **Start time:** 4:00:00 AM
23. **Duration (number of hours):** 1
24. **Days of the week:** Sunday
25. **Days of the month:** All

Review the maintenance windows you have selected.

22.5 Perform Maintenance Selection against LABDB

All of the selections defined in the previous sections have not yet been committed. Make sure that you are satisfied with all of your selections and perform the following steps:

1. Click on the Preview Command link located underneath the **Settings** header. This will allow you to see the code that will be executed against the database.

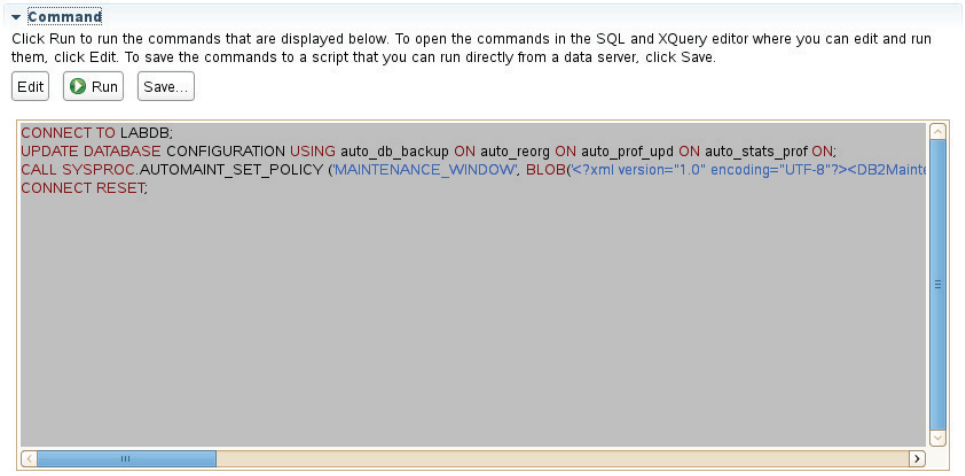


Figure 76 – Preview Command

2. Click Run to perform these actions against the database.

   All actions should complete successfully.

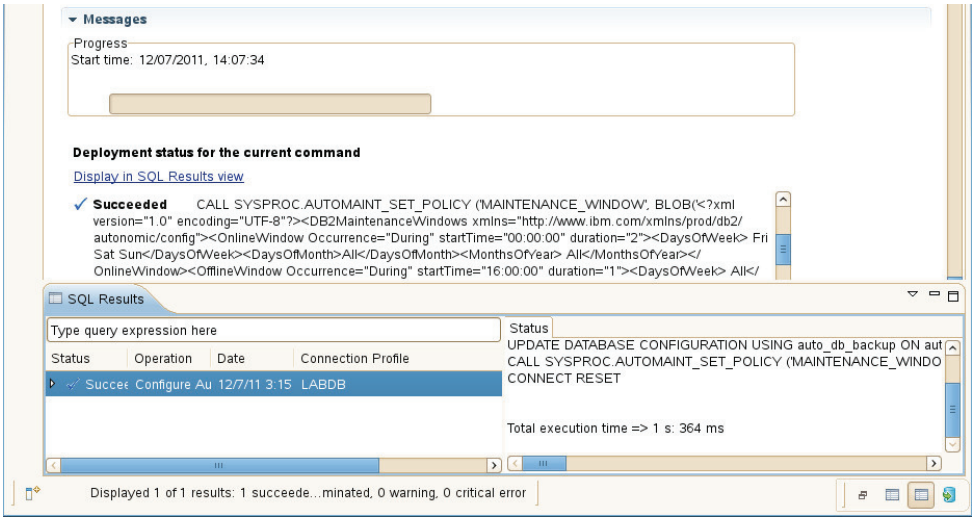


Figure 77 – Run Command

We have now succeeded in setting up regular maintenance activities such as backups, table reorganizations, and statistics collection for the database. These maintenance activities will run only if needed. As an example, a backup of a database will not initialize unless there has been changes in data or configuration on the database.

22.6 Enabling Utility Throttling

Utility throttling regulates the performance impact of maintenance utilities so that they can run concurrently during production periods. The throttling system ensures that the throttled utilities are run as frequently as possible without violating the impact policy. You can throttle statistics collection, backup operations, rebalancing operations, and asynchronous index cleanups. Let's execute a command to limit the impact of these utilities, and view the current configuration for the LABDB database.

1. Execute the following two commands in a terminal window:

```
db2 update dbm cfg using util_impact_lim 10
db2 get dbm cfg
```

Specifying a *util_impact_lim* (impact policy) value of 10 signifies that a throttled backup invocation will not impact the workload by more than 10 percent. The following output should display (note that this is not the entire output) the new value:

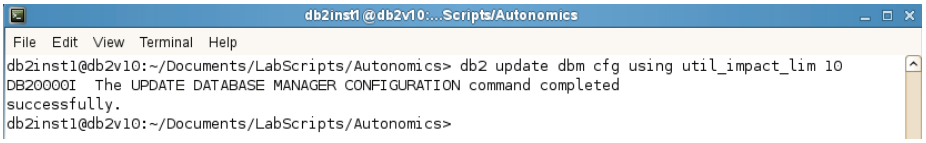


Figure 78 - UTIL_IMPACT_LIM set to 10

23 Design Advisor

The task of selecting which indexes, MQTs, clustering dimensions, or database partitions to create for a complex workload can be quite daunting. The Design Advisor identifies all of the objects needed to improve the performance of the workload.

23.1 Create the Explain Tables

Create Explain tables to capture access plans when the Explain facility is activated. The Explain tables must be created before Explain can be invoked.

1. Issue the following two commands:

```
cd /home/db2inst1/sqllib/misc
db2 -tvf EXPLAIN.DDL
```

All commands should execute successfully.

Note: DB2 must be started and a connection to the LABDB database must exist for this procedure to execute successfully.

23.2 Design Advisor Recommendations

The Design advisor can be used to get recommendations to tune individual SQL statements or a workload of statements. The Design Advisor analyzes a specified workload and considers factors such as the type of workload statements, the frequency with which a particular statement occurs, and characteristics of your database to generate recommendations that minimize the total cost to run the workload.

1. Change back to the directory containing all of the lab scripts:

```
cd /home/db2inst1/Documents/LabScripts/Autonomics
```

2. View and execute the initDB2advis.sql script to create duplicate tables of existing ones that will be used for this section.

```
db2 -tvf initDB2advis.sql
```

Creation of all tables and indexes should execute successfully. We will now use Visual Explain to see the Query plan and table scans required for a particular query. We will then use Design Advisor to see if we can improve the amount of time it takes to execute this query by creating indexes.

3. In Data Studio, click on database **LABDB**, from within the Administration Explorer. Then, click **New SQL Script** menu item.

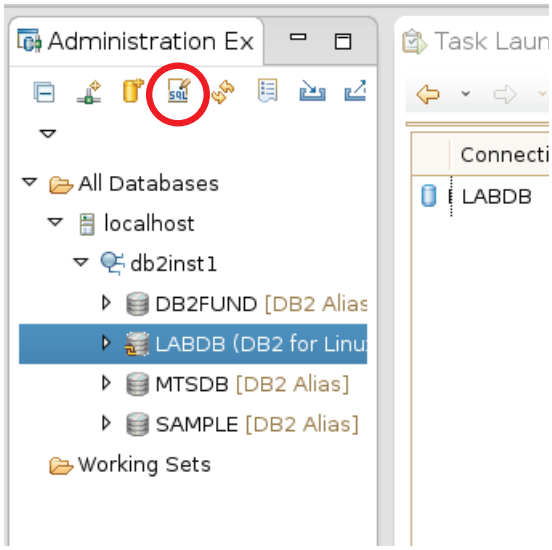


Figure 79 – Opening New SQL Script

4. Copy the following query into the text box:

```
SELECT c_first, d_name FROM dupvps.dupdealcust, dupvps.dupcustomer,
dupvps.dupdealer WHERE f_c_id=c_id AND f_d_id=d_id AND d_id=548
```

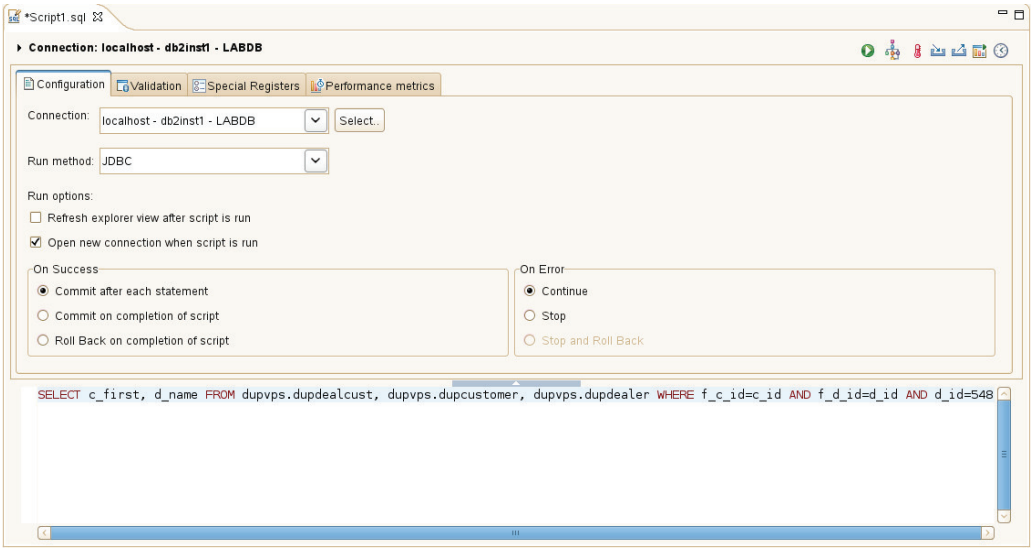


Figure 80 – SQL Query Textbox

5. After inputting the above statement, click on the **Visual Explain icon** from within this menu.

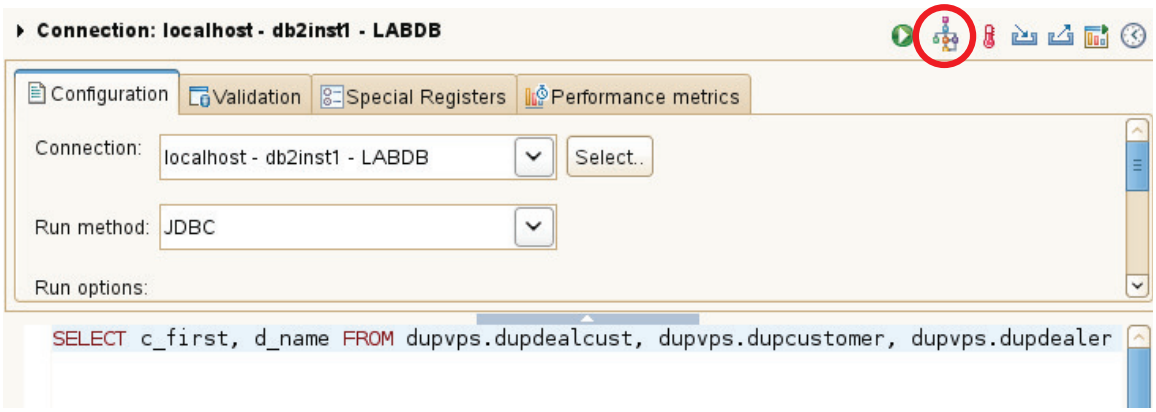


Figure 81 – Selecting Visual Explain

The Visual Explain screen lets you view the access plan for explained SQL or XQuery statements as a graph. You can use the information available from the graph to tune your queries or create objects within your database for better performance.

Accept the defaults in the pop-up by clicking **Finish**. A diagram like the following should appear in the Access Plan Diagram view at the lower half of your Data Studio workspace. **Click** node RETURN in the graph to reveal the cumulative total cost of the access plan.

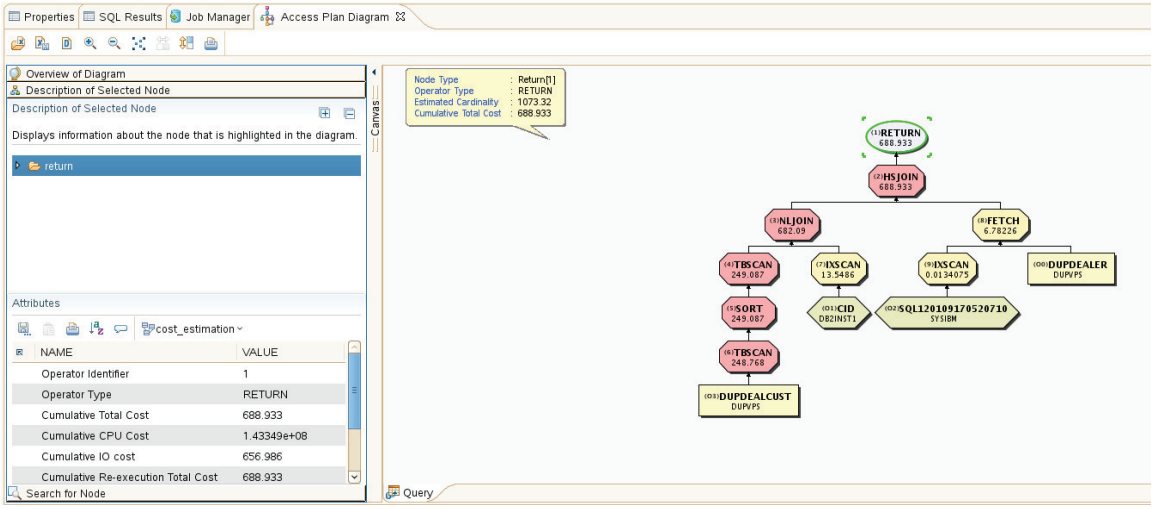


Figure 82 – Access Plan

Note: Cost is derived from a combination of CPU cost (in number of instructions) and I/O (in numbers of seeks and page transfers). The unit of cost is *timeron*. A timeron does not directly equate to any actual elapsed time, but gives a rough relative estimate of the resources (cost) required by the database manager to execute two plans for the same query.

We can see from the graph that a table scan on DUPDEALCUST is done. We can now invoke the Design Advisor to see if there are any suggestions that can help us improve this query plan.

6. Invoke the Design advisor by executing the following command in the terminal window:

```
db2advis -d labdb -s "SELECT c_first, d_name FROM dupvps.dupdealcust,
dupvps.dupcustomer, dupvps.dupdealer WHERE f_c_id=c_id AND f_d_id=d_id AND
d_id=548"
```

Design Advisor is used to suggest the necessary things to help us improve performance. An output similar to the following should display:

```
…
Optimization finished.
  3  indexes in current solution
 [641.0000] timerons  (without recommendations)
 [428.0000] timerons  (with current solution)
 [33.23%] improvement


--
--
-- LIST OF RECOMMENDED INDEXES
-- ===========================
-- index[1],    0.884MB
   CREATE INDEX "DB2INST1"."IDX1112072206560" ON "DUPVPS  "."DUPDEALCUST"
   ("F_D_ID" ASC, "F_C_ID" ASC) ALLOW REVERSE SCANS COLLECT SAMPLED DETAILED STATISTICS;
   COMMIT WORK ;
-- index[2],    0.013MB
   CREATE INDEX "DB2INST1"."IDX1112072206490" ON "DUPVPS  "."DUPDEALER"
   ("D_ID" ASC, "D_NAME" DESC) ALLOW REVERSE SCANS COLLECT SAMPLED DETAILED STATISTICS;
   COMMIT WORK ;
…
```

Note: Sizes of the recommended indexes are provided for convenience.

We will now create the recommended indexes and confirm the improvement in the query plan by executing the crt_index_recmd.sql script.

7. Execute the crt_index_recmd.sql script in DB2 by issuing the following command:

```
db2 -tvf crt_index_recmd.sql
```

8. View the new execution plan for the query by going through **steps 3-5**.

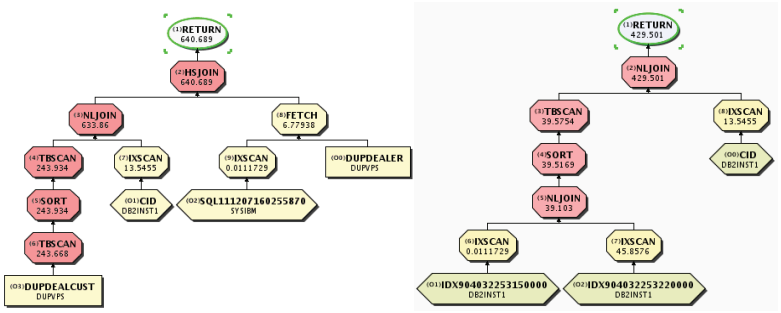


Figure 83 - Access Plan comparison: before and after

You can the see the query plan is better since DB2 has avoided a table scan and the total cost, timerons, has significantly decreased.

23.3 Executing the Query and Seeing Buffer Pool Memory Changes

Now that we have optimized the database for that particular query, let's execute this query and use memory tracker to see what happens with the buffer pools and all other memory components as this query is executed.

Let's start by decreasing the current size of the buffer pool to something very small (1.3 MB).

1. Execute the command below to change the buffer pool size and immediately take snapshot of the memory.

```
db2 alter bufferpool vpsbuff immediate size 50 automatic; db2mtrk -d >
bfr_query.snap;
```

> **Note: db2mtrk is a memory tracker command used to provide complete report of memory status, for instances, databases, agents, and applications.**

2. Use gedit to view the bfr_query.snap file:

```
Memory for database: LABDB

   utilh        pckcacheh    other        catcacheh    bph (2)      bph (1)
   64.0K        512.0K       128.0K       832.0K       1.3M         4.6M

   bph (S32K)   bph (S16K)   bph (S8K)    bph (S4K)    shsorth      lockh
   832.0K       576.0K       448.0K       384.0K       192.0K       56.9M

   dbh          apph (711)   apph (465)   apph (462)   apph (461)   apph (460)
   58.7M        128.0K       192.0K       64.0K        64.0K        64.0K

   apph (459)   apph (458)   apph (457)   apph (456)   appshrh
   128.0K       64.0K        64.0K        64.0K        832.0K
```

> **Note: This is system dependent and therefore your values may be slightly different.**

BPH (2) should be the VPSBUFF buffer pool associated with the LABDB database. We can verify this by executing `SELECT * FROM SYSCAT.BUFFERPOOLS` which provides the ID of each buffer pool.

During creation we used the `SIZE AUTOMATIC` clause for the buffer pool meaning that it is allowed to expand if needed. Let's see how this buffer pool will expand when we run the select query from the pervious section.

3. Execute the query by issuing the following command from the terminal:

```
db2 "SELECT c_first, d_name FROM dupvps.dupdealcust, dupvps.dupcustomer,
dupvps.dupdealer WHERE f_c_id=c_id AND f_d_id=d_id AND d_id=548"
```

This query should return all of the records with these specifications.

4. After a few minutes, we can check how the BP memory allocation has changed during this query:

```
db2mtrk -d > after_query.snap
```

5. Use gedit to view the after_query.snap file. Something similar to the following output should display:

```
Memory for database: LABDB

   utilh       pckcacheh   other       catcacheh   bph (2)     bph (1)
   64.0K       768.0K      128.0K      832.0K      16.2M       4.6M

   bph (S32K)  bph (S16K)  bph (S8K)   bph (S4K)   shsorth     lockh
   832.0K      576.0K      448.0K      384.0K      192.0K      56.9M

   dbh         apph (758)  apph (465)  apph (462)  apph (461)  apph (460)
   58.7M       128.0K      192.0K      64.0K       64.0K       64.0K

   apph (459)  apph (458)  apph (457)  apph (456)  appshrh
   128.0K      64.0K       64.0K       128.0K      896.0K
```

Therefore, it is evident that this buffer pool has increased in size to accommodate the query that was being performed.

24 Self-Tuning Memory Manager (STMM)

Self tuning memory simplifies the task of memory configuration by automatically setting values for several memory configuration parameters. When enabled, the memory tuner dynamically distributes available memory resources between several memory consumers including sort, package cache and lock list areas and buffer pools.

STMM can be enabled for all of the major memory consumers within DB2 or it can be individually enabled for each of the following memory consumers:

26. Buffer pools (controlled by the `ALTER BUFFERPOOL` and `CREATE BUFFERPOOL` statements)
27. Package cache (controlled by the *pckcachesz* configuration parameter)
28. Locking memory (controlled by the *locklist* and *maxlocks* configuration parameters)
29. Sort memory (controlled by the *sheapthres_shr* and the *sortheap* configuration parameters)
30. Database shared memory (controlled by the *database_memory* configuration parameter)

In this section of the lab, we will enable STMM for the LABDB database and explore a few of the related database configuration parameters.

24.1 Connect to the LABDB Database and Examine the Database Configuration Parameters

1. Launch a terminal window or use one that is already open.
2. Connect to the LABDB database by issuing the following commands:

   ```
   db2 FORCE APPLICATION ALL
   db2 connect to labdb
   ```

   The connection to the database should now be established.
3. Look at the database configuration parameters for the LABDB database. Execute the following command:

   ```
   db2 get db cfg for labdb show detail | more
   ```

```
       Database Configuration for Database labdb

 Description                                  Parameter   Current Value              Delayed Value
 ---------------------------------------------------------------------------------------------------
----------
 Database configuration release level                  = 0x0f00
 Database release level                                = 0x0f00

 Database territory                                    = US
 Database code page                                    = 1208
 Database code set                                     = UTF-8
 Database country/region code                          = 1
 Database collating sequence                           = IDENTITY                 IDENTITY
 Alternate collating sequence              (ALT_COLLATE) =
 Number compatibility                                  = OFF
 Varchar2 compatibility                                = OFF
 Date compatibility                                    = OFF
 Database page size                                    = 4096                     4096

 Statement concentrator                      (STMT_CONC) = OFF                      OFF

 Discovery support for this database       (DISCOVER_DB) = ENABLE                   ENABLE

 …                                                     … … …                      …

--More--
```

Note:

3. **Description: Description of the database configuration parameter.**
4. **Parameter: Name of database parameter.**
5. **Current Value: Current value of the database parameter that is active.**
6. **Delayed Value: Delayed value of the database parameter which will be applied the next time the instance is started.**

4. Press the **space bar** to scroll down until you see the section regarding STMM similar to the below screen.

```
Self tuning memory                    (SELF_TUNING_MEM) = ON (Active)                ON
Size of database shared memory (4KB)  (DATABASE_MEMORY) = AUTOMATIC(77792)           AUTOMATIC(78440)
Database memory threshold               (DB_MEM_THRESH) = 10                         10
Max storage for lock list (4KB)               (LOCKLIST) = AUTOMATIC(12064)          AUTOMATIC(12064)
Percent. of lock lists per application        (MAXLOCKS) = AUTOMATIC(98)             AUTOMATIC(98)
Package cache size (4KB)                    (PCKCACHESZ) = AUTOMATIC(398)            AUTOMATIC(398)
Sort heap thres for shared sorts (4KB) (SHEAPTHRES_SHR) = AUTOMATIC(260)             AUTOMATIC(260)
Sort list heap (4KB)                          (SORTHEAP) = AUTOMATIC(52)             AUTOMATIC(52)

Database heap (4KB)                             (DBHEAP) = AUTOMATIC(4465)           AUTOMATIC(4465)
Catalog cache size (4KB)                (CATALOGCACHE_SZ) = 300                      300
Log buffer size (4KB)                         (LOGBUFSZ) = 2149                      2149
Utilities heap size (4KB)                (UTIL_HEAP_SZ) = 13137                      13137
Buffer pool size (pages)                      (BUFFPAGE) = 1000                      1000
SQL statement heap (4KB)                      (STMTHEAP) = AUTOMATIC(2048)           AUTOMATIC(2048)
Default application heap (4KB)             (APPLHEAPSZ) = AUTOMATIC(256)             AUTOMATIC(256)
Application Memory Size (4KB)               (APPL_MEMORY) = AUTOMATIC(10016)         AUTOMATIC(10000)
Statistics heap size (4KB)                 (STAT_HEAP_SZ) = AUTOMATIC(4384)          AUTOMATIC(4384)
```

Note: STMM is already enabled for the LABDB database. This is because the following are enabled by default during database creation:

7. **The value of the SELF_TUNING_MEM database parameter is ON (Active) which means STMM is enabled.**
8. **Notice that the value of the DATABASE_MEMORY database parameter is AUTOMATIC. STMM can dynamically ask for and give back memory to the operating system.**
9. **Notice that the memory consumers that we have enabled for STMM all shows database parameter of value AUTOMATIC.**

5. Enter “q” to exit this view or hit the space bar until you have scrolled through the output on screen.

   To enable self tuning memory for buffer pools, you have to set the buffer pool size to `AUTOMATIC`. You can do this using the `ALTER BUFFER POOL` statement for existing buffer pools or the `CREATE BUFFER POOL` statement for new buffer pools. In this lab, we’ve already seen the automatic buffer pool increase in size when we used the `db2mtrk` command to check the memory status. This was seen in the previous section.

Note:
Changes resulting from self-tuning operations are recorded in memory tuning log files that are located in the stmmlog subdirectory. These log files contain summaries of the resource demands from each memory consumer during specific tuning intervals, which are determined by timestamps in the log entries.

In this exercise, the STMM logs are stored at /home/db2inst1/sqllib/db2dump/stmmlog/

25 Data Movement Tools (optional exercise)

25.1 EXPORT Utility

The export utility extracts data using an SQL select or an XQuery statement, and places that information into a file. You can use the output file to move data for a future import or load operation or to make the data accessible for analysis.

The following items are mandatory for a basic export operation:

31. The path and name of the operating system file in which you want to store the exported data
32. The format of the data in the output file. Export supports IXF and DEL data formats for the output files.
33. A specification of the data that is to be exported

1. Change directory to the “output” directory in your terminal window.

```
cd /home/db2inst1/Documents/LabScripts/Autonomics/output
```

2. Issue the following command to export data from table CAR.

```
db2 "EXPORT TO car.txt OF DEL MESSAGES msgs.txt SELECT * FROM vps.car"
```

Note: DEL (Delimited ASCII format) is a format of data in the output file car.txt. Warning and error messages occurred during an export operation will be stored in msgs.txt.

A message similar to the following should be displayed after successful execution of the command:

```
~/Documents/LabScripts/Autonomics/output> db2 "EXPORT TO car.txt OF DEL MESSAGES
msgs.txt SELECT * FROM vps.car"

Number of rows exported: 132008
```

3. You can look into car.txt for all exported records by using gedit or msgs.txt for any warning message. Close the gedit editor after you have finished viewing the files.

25.2 IMPORT Utility

The IMPORT utility populates a table, typed table, or view with data using an SQL INSERT statement. If the table or view receiving the imported data already contains data, the input data can either replace or be appended to the existing data. The IMPORT utility inserts data from an external file with a supported file format into a table, hierarchy, view or nickname. Like export, import is a relatively simple data movement utility. It can be activated by issuing CLP commands, by calling the ADMIN_CMD stored procedure, or by calling its API, db2Import, through a user application.

1. Change directory to the “input” directory in your terminal window.

```
cd /home/db2inst1/Documents/LabScripts/Autonomics/input
```

2. Issue the following command to import data into table ORDER.

```
db2 "IMPORT FROM order.txt OF DEL ALLOW WRITE ACCESS COMMITCOUNT 10 INSERT
INTO vps.order"
```

Note:

ALLOW WRITE ACCESS: runs import in the online mode. An intent exclusive (IX) lock on the target table is acquired when the first row is inserted. This allows concurrent readers and writers to access table dat.

COMMITCOUNT: performs a COMMIT after every n records are imported. When a number n is specified, import performs a COMMIT after every n records are imported.

A message similar to the following should be displayed after successful execution of the command:

```
… …

SQL3149N "51" rows were processed from the input file. "51" rows were successfully
inserted into the table. "0" rows were rejected.

Number of rows read           = 51
Number of rows skipped        = 0
Number of rows inserted       = 51
Number of rows updated        = 0
Number of rows rejected       = 0
Number of rows committed      = 51
```

25.3 LOAD Utility

The LOAD utility is best suited to situations where performance is your primary concern. It is faster then the IMPORT utility because it writes formatted pages directly into the database rather than using SQL INSERTS. The load utility also allows you the option to not log the transactions. Load operations can fully exploit resources such as memory and multiple processors.

The LOAD, like the EXPORT and IMPORT utilities, is cross platform compatible. This utility can be used as an alternative to the import utility. Use the LOAD utility if you want to move large quantities of data into empty tables or tables that already contain data.

To demonstrate how the LOAD utility is used, you'll use the utility to import data into the database.

1. In the terminal, start the LOAD utility by executing the command below. Note that the LOAD command should be entered on a single line.

```
db2 connect to labdb
cd /home/db2inst1/Documents/LabScripts/Autonomics/input
db2 load from order2.txt of del savecount 500 messages ../output/load.msg
insert into vps.order cpu_parallelism 2;
```

```
Number of rows read         = 14996
Number of rows skipped      = 0
Number of rows loaded       = 14996
Number of rows rejected     = 0
Number of rows deleted      = 0
Number of rows committed    = 14996
```

The table was placed in the Set Integrity Pending state at the beginning of a load operation. After the load operation is complete, the SET INTEGRITY statement must be used to take the table out of Set Integrity Pending state.

2. In terminal, execute the command below to bring the table out of set integrity pending state.

```
db2 SET INTEGRITY FOR VPS.ORDER IMMEDIATE CHECKED;
```

25.4 INGEST Utility

The ingest utility (sometimes referred to as continuous data ingest, or CDI) is a high-speed client-side DB2 utility that streams data from files and pipes into DB2 target tables. Because the ingest utility can move large amounts of real-time data without locking the target table, you do not need to choose between the data currency and availability.

The ingest utility ingests pre-processed data directly or from files output by ETL tools or other means. It can run continually and thus it can process a continuous data stream through pipes. The data is ingested at speeds that are high enough to populate even large databases in partitioned database environments.

An INGEST command updates the target table with low latency in a single step. The ingest utility uses row locking, so it has minimal interference with other user activities on the same table.

26 Summary

You can see by now that DB2's autonomic features help to make database administration as easy and low-cost as possible. By automating tasks such as memory allocation (Self Tuning Memory), storage management (Automatic Maintenance and Automatic storage features) and business policy maintenance, DB2 is able to perform many management tasks itself, freeing up DBAs to focus on new projects.

27 Cleanup

3. To clean your environment after completing the exercise, right-click on **LABDB** database in the **Administration Explorer** view and select **Disconnect.**
4. Next, close Data Studio and switch back to the terminal window that you have been using so far.
5. Finally, execute the commands below. Use "password" as password.

```
db2 force application all
su -
cd /home/db2inst1/Documents/LabScripts/Autonomics/setup
./cleanup.sh
```

6. If you would like to redo this lab in the future, please execute the following commands in a terminal window as root:

```
cd /home/db2inst1/Documents/LabScripts/Autonomics/setup
./config.sh
```

IBM Canada
8200 Warden Avenue
Markham, ON
L6G 1C7
Canada

DB2® 10.5 with BLU Acceleration

Data-Intensive Analytics for DB2

Hands-On Lab

Table of Contents

28 Introduction

IBM® DB2 Version 10.5 for Linux, UNIX, and Windows offers accelerated analytic processing by introducing a new processing paradigm and data format within the DB2 database product. Advantages include significant reductions in time-to-value and increased consumability, which can be achieved through minimal DBA design requirements and reduced query tuning and debugging efforts. Industry-leading compression, large performance gains for analytic queries, and large reductions in performance variation round out the benefits of deploying this technology.

BLU Acceleration is a new storage engine along with integrated runtime (directly into the core DB2 engine) to support the storage and analysis of column organized tables. The BLU Acceleration processing is parallel to the regular, row-based table processing found in the DB2 engine. This is not a bolt-on technology nor is it a separate analytic engine that sits outside of DB2. Much like when IBM added XML data as a first class object within the database along with all the storage and processing enhancements that came with XML, now IBM has added column organized tables directly into the storage and processing engine of DB2.

29 Suggested Reading

IBM DB2 Version 10.5 for Linux, UNIX and Windows Information Center:
http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/index.jsp?topic=%2Fcom.ibm.db2.luw.welcome.doc%2Fdoc%2Fwelcome.html

30 About this Lab

In this lab, you will explore DB2 BLU Acceleration features.
. After completing this lab, you will be able to do the following

1. Create column-organized tables
2. Load operation in column oraganized tables
3. Compression
4. Query Execution using column-organized tables

The exercises in this lab are designed to give you a good idea of the features available with DB2 BLU.
Note that this lab will not necessarily cover the approach you would take to initially configure a data server; however, it will give you a good idea of some necessary steps required to use BLU features

31 Basic Set up and Start of Lab

The lab consists of a series of tasks, each highlighting one or more concepts of the DB2 BLU Acceleration features described in the introduction.
Some of the commands and queries required to perform certain parts of certain tasks are quite long. In order to avoid having to type these longer commands, we have provided you with scripts for some of these tasks. They are located in the /home/db2inst1/Documents/LabScripts/BLU directory

32 Environment Setup Requirements

To complete this lab you will need the following:

2. DB2 10.5 Bootcamp VMware® image
3. VMware Workstation 6.5 or later

33 Initial Steps

5. Start the VMware image by clicking the button in VMware Workstation.
6. At the login prompt, log in with the following credentials :
   - Username : **db2inst1**
   - Password : **password**
7. Open a terminal window by right-clicking on the Desktop and choosing the Open Terminal item.

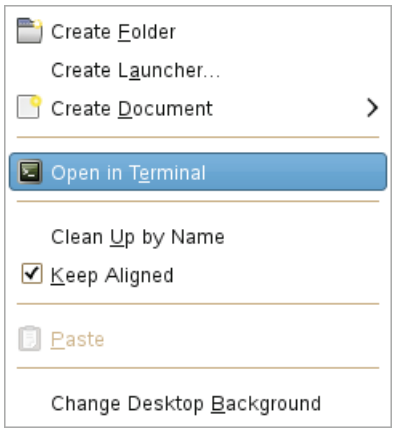


Figure 84 – Open in Terminal on Desktop

8. Ensure that the DB2 Database Manager has been started by issuing the following

```
db2start
```

A message like the following should be displayed to if the DB2 Database Manager had not been started yet:

```
db2inst1@db2awse:~> db2start
07/12/2013 09:10:39     0   0   SQL1063N  DB2START processing was successful.
SQL1063N  DB2START processing was successful.
```

33.1 Create BLU Database

During this lab, we will use the TEST database for demonstrating various features of DB2 BLU Acceleration that makes it easy to understand the technology.

Before creating the database the following mandatory steps needs to be executed:

1. Set the DB2_WORKLOAD registry variable to ANALYTICS

```
db2inst1@db2awse:~> db2set DB2_WORKLOAD=ANALYTICS

db2inst1@db2awse:~> db2set -all
[i] DB2_WORKLOAD=ANALYTICS
[i] DB2RSHCMD=/usr/bin/ssh
[i] DB2COMM=TCPIP
[i] DB2AUTOSTART=YES
[g] DB2SYSTEM=db2awse
[g] DB2INSTDEF=db2inst1
db2inst1@db2awse:~>
```

2. Stop and Start the instance db2inst1

```
db2stop force
```

```
db2inst1@db2awse:~> db2stop FORCE
07/12/2013 09:06:14     0   0   SQL1064N  DB2STOP processing was successful.
SQL1064N  DB2STOP processing was successful.
```

```
db2start
```

```
db2inst1@db2awse:~> db2start
07/12/2013 09:10:39     0   0   SQL1063N  DB2START processing was successful.
SQL1063N  DB2START processing was successful.
```

3. Create the TEST database

```
db2 drop database test
db2 create database test
```

```
db2inst1@db2awse:~> db2 create database test
DB20000I  The CREATE DATABASE command completed successfully.
```

4. Connect to the TEST database

   db2 connect to TEST

```
db2inst1@db2awse:~> db2 connect to TEST

 Database Connection Information
 Database server        = DB2/LINUXX8664 10.5.0
 SQL authorization ID   = DB2INST1
 Local database alias   = TEST
```

5. Check out the database configuration file

   db2 get db cfg | grep –i dft

```
db2inst1@db2awse:~> db2 get db cfg | grep -i dft

 Default query optimization class        (DFT_QUERYOPT) = 5
 Degree of parallelism                   (DFT_DEGREE) = ANY
 Continue upon arithmetic exceptions   (DFT_SQLMATHWARN) = NO
 Default refresh age                  (DFT_REFRESH_AGE) = 0
 Default maintained table types for opt (DFT_MTTB_TYPES) = SYSTEM
 Default prefetch size (pages)        (DFT_PREFETCH_SZ) = AUTOMATIC
 Default tablespace extentsize (pages)   (DFT_EXTENT_SZ) = 4
 Default number of loadrec sessions    (DFT_LOADREC_SES) = 1
 Default data capture on new Schemas   (DFT_SCHEMAS_DCC) = NO
 Default table organization              (DFT_TABLE_ORG) = COLUMN
db2inst1@db2awse:~>
```

34 Compare column-organized and row-organized tables

10. Check if there are any tables already created and drop them.

    db2 drop table testcol

    db2 drop table testrow

11. Create a pair of column-organized and row-organized tables. Navigate to the path /home/db2inst1/Documents/LabScripts/BLU

    cd /home/db2inst1/Documents/LabScripts/BLU

12. Execute the tables.sql file present in the BLU folder

a. db2 –td@ -f tables.sql

```
db2inst1@db2awse:~/Documents/LabScripts/BLU> db2 -td@ -f tables.sql
DB20000I  The SQL command completed successfully.
DB20000I  The SQL command completed successfully.
```

13. List tables created

db2 list tables

```
db2inst1@db2awse:~/Documents/LabScripts/BLU> db2 list tables

 Table/View                      Schema          Type  Creation time
 ------------------------------- --------------- ----- --------------------------
TESTCOL                         DB2INST1        T     2013-07-12-12.16.51.586276
TESTROW                         DB2INST1        T     2013-07-12-12.16.52.583106

2 record(s) selected.
```

14. Check the organization and compression flags in syscat.tables
select tabname, tableorg from syscat.tables where tabname like 'TEST%';

```
db2inst1@db2awse:~/Documents/LabScripts/BLU> db2 "select tabname, tableorg from syscat.tables where tabname
  like 'TEST%'"

 TABNAME     TABLEORG
 ------------------- -----------------
 TESTCOL               C
 TESTROW               R

 2 record(s) selected.
```

select tabname, tableorg, compression from syscat.tables where tabname like 'TEST%';

```
db2inst1@db2awse:~/Documents/LabScripts/BLU> db2 "select tabname, tableorg, compression  from syscat.tables
 where tabname like 'TEST%'"

 TABNAME     TABLEORG COMPRESSION
 -----------          ---------------- ------------------------
 TESTCOL     C
 TESTROW     R                  N

 2 record(s) selected.
```

15. List all the column organized tables
    select tabschema, tabname, tableorg from syscat.tables where tableorg = 'C';

```
db2inst1@db2awse:~/Documents/LabScripts/BLU> db2 "select tabschema, tabname, tableorg from syscat.tables where
tableorg = 'C'"

TABSCHEMA       TABNAME                          TABLEORG
-----------     ---------------                  -------------------
DB2INST1        TESTCOL                              C
SYSIBM          SYN130712121652069148_TESTCOL        C

  2 record(s) selected.
```

16. Describe the synopsis table.
    Note : Here the synopsis tables will be different in various machines. Check the name in your machine and query accordingly. During creation of this Lab below was the name generated.

```
db2inst1@db2awse:~/Documents/LabScripts/BLU> db2 describe table SYSIBM.SYN130712121652069148_TESTCOL

                                Data type                       Column
Column name                     schema    Data type name        Length     Scale Nulls
------------------------------- --------- ------------------    ---------- ----- ------
IDMIN                           SYSIBM    INTEGER                    4     0     No
IDMAX                           SYSIBM    INTEGER                    4     0     No
DATEMIN                         SYSIBM    DATE                       4     0     Yes
DATEMAX                         SYSIBM    DATE                       4     0     Yes
AMOUNTMIN                       SYSIBM    INTEGER                    4     0     Yes
AMOUNTMAX                       SYSIBM    INTEGER                    4     0     Yes
TSNMIN                          SYSIBM    BIGINT                     8     0     No
TSNMAX                          SYSIBM    BIGINT                     8     0     No

8 record(s) selected.
```

17. Populate the testcol table using import utility and the del file present in the Labscripts BLU folder

    import from 20000c.del of del replace into testcol

```
db2inst1@db2awse:~/Documents/LabScripts/BLU> db2 "import from 20000c.del of del replace into testcol"
SQL3109N  The utility is beginning to load data from file "20000c.del".

SQL3110N  The utility has completed processing.  "20000" rows were read from
the input file.

SQL3221W  ...Begin COMMIT WORK. Input Record Count = "20000".

SQL3222W  ...COMMIT of any database changes was successful.

SQL3149N  "20000" rows were processed from the input file.  "20000" rows were
successfully inserted into the table.  "0" rows were rejected.
Number of rows read         = 20000
Number of rows skipped      = 0
Number of rows inserted     = 20000
Number of rows updated      = 0
Number of rows rejected     = 0
Number of rows committed    = 20000
```

18. Examine the content of the synopsis table:
    select * from SYSIBM.SYN130330165216275152_TESTCOL order by TSNMIN;

19. Find out the indexes created for the column-organized table
    select INDSCHEMA,INDNAME, COLNAMES, INDEXTYPE from syscat.indexes where tabname='TESTCOL';

35 Large Tables & LOAD

Starting with DB2® Version 10.5 row size support is extended allowing you to create a table where its row length can exceed the maximum record length for the page size of the table space. In previous releases, the maximum number of bytes allowed in a table row was dependant on the page size of the table space. Any attempt to create a table whose row length exceeded the maximum record length for the page size would result in an error (SQLSTATE 54010). For example, in previous releases the following table could not be created in a 4K page size table space because of its large row size.

By extending row size support to allow for the creation of tables containing large rows that exceed the maximum record length for the page size of the table space. Existing tables can be altered to take advantage of extended row size support.

1. Create a large table
   - Drop the table if any is existing
     - db2 drop table daily_sales_col
   - Execute the below script to create the large table
     - db2 –td@ -f large_table.sql
2. Populate the large table with data using Load utility
   db2 load from sales_1M.del of del replace into daily_sales_col

```
SQL3109N  The utility is beginning to load data from file
"/home/db2inst1/Documents/LabScripts/BLU/sales_1M.del".

SQL3500W  The utility is beginning the "ANALYZE" phase at time "07/12/2013
14:55:26.908173".

SQL3519W  Begin Load Consistency Point. Input record count = "0".

SQL3520W  Load Consistency Point was successful.

SQL3515W  The utility has finished the "ANALYZE" phase at time "07/12/2013
14:55:46.033886".

SQL3500W  The utility is beginning the "LOAD" phase at time "07/12/2013
14:55:46.034633".

SQL3110N  The utility has completed processing.  "1000000" rows were read from
the input file.

SQL3519W  Begin Load Consistency Point. Input record count = "1000000".

SQL3520W  Load Consistency Point was successful.

SQL3515W  The utility has finished the "LOAD" phase at time "07/12/2013
14:56:04.962567".

SQL3500W  The utility is beginning the "BUILD" phase at time "07/12/2013
14:56:04.969036".

SQL3213I  The indexing mode is "REBUILD".

SQL3515W  The utility has finished the "BUILD" phase at time "07/12/2013
14:56:05.536066".


Number of rows read         = 1000000
Number of rows skipped      = 0
Number of rows loaded       = 1000000
Number of rows rejected     = 0
Number of rows deleted      = 0
Number of rows committed    = 1000000
```

3. Examine the synopsis table:

```
select tabname from syscat.tables where tableorg='C'
```

36 Compression

Lets examine the compression feature for large column-organized tables. Execute the below query to know the PCTENCODED values for the columns which gives percentage of values that are encoded as a result of compression for a column in a column-organized table

```
select substr(tabname, 1, 25), substr(colname, 1, 25),  PCTENCODED
from syscat.columns
where tabname like 'DAIL%'
```

36.1 Comparison of compression with row-organized tables

6. Create a row-organized table daily_sales_row
   - db2 drop table daily_sales_row
   - db2 –td@ -f large_row.sql
7. Load the table with records followed by runstats
   - db2 load from sales_1M.del of del replace into daily_sales_row
   - db2 runstats on table daily_sales_row
8. Execute the below select queries to get the comparison between column – organized and row-organized tables
   - SELECT tabname, pctpagessaved FROM syscat.tables WHERE tabname LIKE 'DAILY%'

```
db2inst1@db2awse:~/Documents/LabScripts/BLU> db2 "SELECT tabname, pctpagessaved  FROM
syscat.tables WHERE  tabname LIKE 'DAILY%'"

TABNAME              PCTPAGESSAVED
--------------------  --------------------------
DAILY_SALES_COL          78
DAILY_SALES_ROW       55

  2 record(s) selected.
```

   - Select tabname, npages, mpages, fpages, tableorg from syscat.tables where tabname like 'DAIL%';
   - SELECT substr(a.tabname,1,20) as tabname, COL_OBJECT_P_SIZE, DATA_OBJECT_P_SIZE,
     INDEX_OBJECT_P_SIZE FROM syscat.tables a, SYSIBMADM.ADMINTABINFO b
     WHERE a.tabname = b.tabname and a.tabname like 'DAILY%' ORDER BY a.tabname WITH UR;

- SELECT substr(a.tabname,1,20) as tabname, COL_OBJECT_P_SIZE, DATA_OBJECT_P_SIZE, INDEX_OBJECT_P_SIZE, (COL_OBJECT_P_SIZE + DATA_OBJECT_P_SIZE + INDEX_OBJECT_P_SIZE) as TOTAL_SIZE, dec(1.0/(1.0-(PCTPAGESSAVED*1.0)/100.0),31,2) as COMPRESSION_RATIO, PCTPAGESSAVED FROM syscat.tables a, SYSIBMADM.ADMINTABINFO b WHERE a.tabname = b.tabname and a.tabname like 'DAILY%' ORDER BY a.tabname WITH UR
- SELECT substr(a.tabname,1,20) as tabname, dec(1.0/(1.0-(PCTPAGESSAVED*1.0)/100.0),31,2) as COMPRESSION_RATIO, PCTPAGESSAVED FROM syscat.tables a, SYSIBMADM.ADMINTABINFO b WHERE a.tabname = b.tabname and a.tabname like 'DAILY%' ORDER BY a.tabname WITH UR

37 Query Execution

New explain information is captured to support column-organized table functionality. You can use this information to determine how your application performs when using this new functionality.DB2 Version 10.5 provides a new CTQ plan operator that represents the transition between column-organized data processing and row-organized data processing. Let us generate explain graphs for few queries on column organized tables.

1. We need explain tables so execute the below
    a. cd /home/db2inst1/sqllib/misc
    b. db2 –tvf EXPLAIN.DDL
2. Create a new row table for executing queries
    a. db2 –td@ -f store_row.sql
3. Load the above table with records
    a. db2 load from store.del of del replace into store
    b. db2 runstats on table store
4. Execute the below queries and observe how long does it take
    a. select * from store
    b. SELECT s.state, sum(ds.quantity_sold)
       FROM daily_sales_col ds,store s
       WHERE ds.storekey = s.storekey
       GROUP BY s.state;
5. Lets see how to capture the explain information for row-organized tables
    a. Enable the explain facility by setting the CURRENT EXPLAIN MODE special register as follows:
        i. db2 set CURRENT EXPLAIN MODE YES
    b. Issue your query against row-organized table
        i. SELECT s.state, sum(ds.quantity_sold)
           FROM daily_sales_col ds,
           store s
           WHERE ds.storekey = s.storekey
           GROUP BY s.state

Note :- Check the permissions for the folders if db2exfmt fails execute the below
Chmod –R /home/db2inst1 777

    c. Capture the explain plan in an output file output_row.exfmt
        i. db2exfmt -d TEST -1 -o output_row.exfmt

6. Convert the store row organized table to column organized table using db2convert utility
    a. Check whether the INTRA_PARALLEL is set to YES in database manager file

i. db2 get dbm cfg | grep –I INTRA_PARALLEL
ii. db2 update dbm cfg using INTRA_PARALLEL YES

b. Convert the row_organized table to column-organized table using db2convert utility
   i. db2convert -d test -z db2inst1 -t store

```
Final Summary:

Table                          RowsNum       InitSize (MB)   FinalSize (MB)  CompRate (%)    State
------------------------------------------ --------------- --------------- --------------- ---------------
"DB2INST1"."STORE"                    10            0.25            2.88        -1050.00       Completed


Pre-Conversion Size (MB): 0.25
Post-Conversion Size (MB): 2.88
Compression Rate (Percent): -1050.00

SQL2446I  The db2convert command completed successfully.  All row-organized tables that satisfy the
specified matching criteria have been converted to column-organized tables.
```

7. Capture the explain information for column-organized tables by given steps below:
   a. Enable the explain facility by setting the CURRENT EXPLAIN MODE special register as follows:
      i. db2 set CURRENT EXPLAIN MODE YES
   b. Issue your query against column-organized table which has been converted to column in the previous step
      i. SELECT s.state, sum(ds.quantity_sold)
         FROM daily_sales_col ds,
         Store s
         WHERE ds.storekey = s.storekey
         GROUP BY s.state
   c. Capture the explain plan in an output file output_row.exfmt
      i. db2exfmt -d TEST -1 -o output_column.exfmt
8. Compare both the explain plans between row-oranganized tables and column-organized tables present in the output_row.exfmt and output_column.exfmt files.
   a. Row – Organized Table where Columns join Rows (suboptimal plan) whereas in a Column-Organized tables Columns join columns generates better plans. Examine the cost of both the Access plans

38 Other 10.5 features

Expression based indexes

With expression-based indexes, you can create an index that includes expressions. The performance of queries that involves expressions is improved if the database manager chooses an index that is created on the same expressions. Expression-based indexes are best suited when you want an efficient evaluation of queries that involve a column expression. Simple index keys consist of a concatenation of one or more specified table columns. Compared to simple indexes, the index key values of expression-based indexes are not the same as values in the table columns. The values are transformed by the expressions that you specify.

You can create the index with the CREATE INDEX statement. If an index is created with the UNIQUE option, the uniqueness is enforced against the values that are stored in the index. The uniqueness is not enforced against the original column values.

- Create a table

    - CREATE TABLE NAMES (FIRST CHAR(20), LAST CHAR(20)) ORGANIZE BY ROW
- Insert records into the table
    - INSERT INTO NAMES VALUES ('Tony', 'Blair')
- Create a function based index on the FIRST column
    - CREATE INDEX name_upper on NAMES (UPPER(FIRST))
    - RUNSTATS ON TABLE NAMES AND INDEXES ALL
- Examine the explain plans for the following two queries
    - SELECT * FROM NAMES WHERE UPPER(LAST) = 'BLAIR'
    - SELECT * FROM NAMES WHERE UPPER(FIRST) = 'TONY'
- Try some other scalar functions (like functions on date or timestamp columns or string functions on character columns) to see what is possible with function based indexes.

39 Summary

- **BLU Acceleration provides three key benefits:**
    - Fast
        - Unprecedented performance for analytical workloads, often 8x to 25x faster.
        - Examples of workloads > 100x
        - Examples of individual queries > 1000x
    - Small
        - Stronger compression and less space required for auxiliary data structures.
        - 10x savings is versus uncompressed row-tables is common.
    - Simple
        - Much less tuning needed, more predictable and reliable performance
        - Tuning, statistics collection, space reclaim, workload management all tuned and automated right out of the box
        - Adapts automatically to your server's memory and CPUs

.

In this lab, you have learned the new DB2 BLU and how to

1. Create column-organized tables and large tables
2. Load operation in column oraganized tables
3. Compression in column-organized Vs row-organized tables
4. Query Execution using column-organized tables
5. Expression based indexes

40 Cleanup

5. To clean the environment after completing the lab execute the commands below. Use "password" as password.

```
su -
cd /home/db2inst1/Documents/LabScripts/BLU/setup
./cleanup.sh
```

IBM Canada
8200 Warden Avenue
Markham, ON
L6G 1C7
Canada