Laboratorio PoliCloud - Politecnico di Milano

IBM Data Studio and IBM DB2 LUW database

Management Guide

Version	Date	Edited by
1.0	04/2015	Mario Marchente

SUMMARY

1. – Introduction

This guide aims to introduce a user into the relational IBM DB2 for Linux/Unix/Windows (LUW) database environment and in particular into some essential administrative issues that are important to start working, in the short time, with DB2 LUW. To interface this relational framework the IBM Data Studio client is also discussed. It is a graphical tool to totally manage local or distributed DB2 LUW database servers and their objects and to develop specific applications. Complete information about the installed distribution (latest release: version 10.5 with BLU Acceleration technology capabilities) at the PoliCloud interdepartmental laboratory are obtainable from the IBM Knowledge Center link:
https://www-01.ibm.com/support/knowledgecenter/SSEPGG_10.5.0/com.ibm.db2.luw.kc.doc/welcome.html

2. – IBM DB2 for Linux/Unix/Windows (LUW)

DB2 for Linux/Unix/Windows (LUW) (herein indicated as DB2) is a database management server product developed by IBM. The version 10.5 “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”. Moreover this version fully integrates the IBM BLU Acceleration collection technologies created by the IBM Research and Development Labs for analytical database workloads. The BLU Acceleration capabilities consist of: in-memory processing of columnar data, the ability to “act on the data while still compressed”, also called “actionable compression” (which uses approximate Huffman encoding to compress and pack data tightly), cpu acceleration (which exploits Single Instruction Multiple Data (SIMD) technology and provides parallel vector processing), and data skipping (data not useful to the current active workload are ignored). BLU Acceleration does not need operation like indexes definition and query aggregation or tuning and utilizes the same storage and memory constructs, Structured Query Language (SQL) interfaces and administration tools as traditional DB2 databases.

2.1 – Database Objects

A database is a collection of database objects. DB2 provides different types of database objects to store and represent different information (Figure 2.1). It uses the standardized SQL language to work with them and the data they contain. Most of the available SQL statements can be categorized according to the function they have been designed to perform. The Database Definition Language (DDL) category is made up by the SQL statements designated to create, modify and remove database objects. These statements are: CREATE, DESCRIBE, ALTER, DROP. The Database Manipulation Language (DML) category includes the SQL statements assigned to data manipulation operations. SELECT, UPDATE, INSERT, and DELETE are the related statements. The DB2 database objects are described next:

- *Partition group* is a logical database object representing a collection of database partitions. In a single-partition environment, partition groups are not relevant; however, in multi-partition environments, a partition group facilitates the work of a database administrator, that can perform database operations on or within several partitions at a time. Partition groups are able to contain one or more table spaces. In Figure 2.1, partition group pg1 contains table space tbls1.

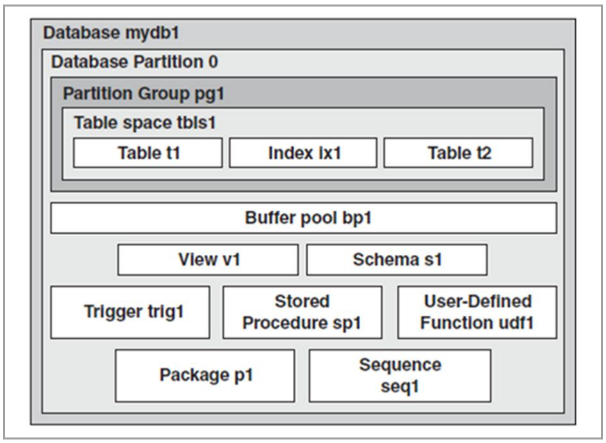


Figure 2.1 – DB2 LUW Database Objects

- *Table space* is a logical database object that associates tables and indexes: 1) to the physical devices, where these objects are stored, and 2) to the physical memory where their data are cached to be accessed. Tables and indexes must be created inside a table space as illustrated in Figure 2.1, where tables t1 and t2 and index ix1 are all created inside table space tbls1.
- *Table* consists of rows and columns, like spreadsheets. Data can be inserted, deleted, and updated within a table. Figure 2.1 has two tables, t1 and t2.
- *Indexes* are an ordered set of keys each pointing to a row in a table. They improve application performance when searching for specific rows. Indexes can also be used to guarantee the uniqueness of rows. In Figure 2.1, index ix1 is associated to table t1.
- *Buffer pool* is a physical memory area that caches the most recently used database information. Without buffer pools, every single piece of data has to be retrieved from disk, which is a very slow device. Buffer pools are associated to tables and indexes through a table space. In Figure 2.1, table space tbls1 is associated to buffer pool bp1, therefore tables t1 and t2 and index ix1 use buffer pool bp1.
- *View* is an alternate way of representing data that exists in one or more tables. A view can include some or all of the columns from one or more tables. It can also be based on other views. In Figure 2.1, view v1 is based on table t1.
- *Schema* is an object that provides a logical grouping of other database objects. A schema can be owned by an individual, who can control access to the objects within it. Schemas can be implicitly or explicitly specified when accessing an object. Every object in the database is created with a two-part name separated by a dot:

  schema_name.object_name

  The first part of this two-part name is the schema name.. The default schema name is the Id of the user that is granted to connect to and work on a database.
- *Trigger* is an object that contains application logic that is triggered by specific actions, like an update to a table. For example, in Figure 2.1, a trigger can be created so that after table t1 is updated, table t2 is also updated with some other information.
- *Stored procedure* is an object used to move application logic into a database. By inserting part of the application logic in the database, the amount of network traffic between the application and the database is considerably reduced, improving performance.
- *User-defined functions* (UDFs) allow database users to extend the SQL language by creating built-in functions that can be used anywhere. Similar to stored procedures, application logic can be moved to the database by using UDFs.

- *Package* is an object containing the compiled version of SQL queries, as well as the access path that the DB2 optimizer, the intelligent part of DB2, has chosen to retrieve data for those queries.
- *Sequence* is an object that allows the generation of unique numbers in sequence. These numbers can be used across the database as a unique identifier for tables or for applications.

2.2 – Data Types

A data type indicates what type of data can be saved in a column or variable and how large it can be. The smallest unit of data that can be manipulated in SQL is called a value. Values are interpreted according to the data type of their source. Sources are: Constants, Columns, Functions, Expressions, Special registers, Variables (such as host variables, SQL variables, global variables, parameter markers, module variable, and parameters of routines), Boolean values. All data types include the null value (exception: columns defined as NOT NULL cannot contain null values), a special element that designates the absence of a (non-null) value. DB2 data types are either:

- Built-in data types (Figure 2.2).
- User-defined types (UDTs).

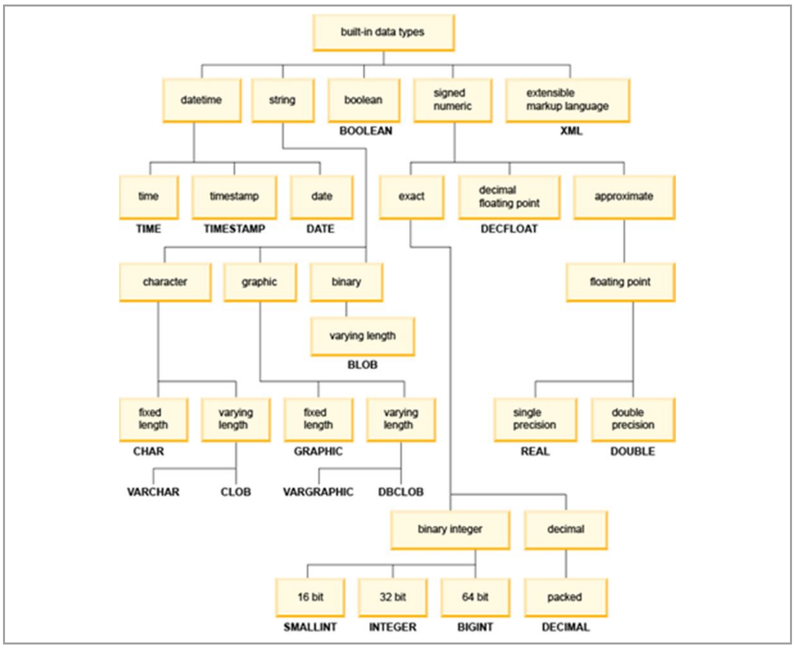


Figure 2.2 – DB2 LUW Internal Data Types

2.3 – Users, groups, roles and PUBLIC

2.3.1 - Users and groups

DB2 requires the underlying operating system for authenticating users (and finding group membership). When running on Windows, DB2 uses (by default) either the local Security Account Manager (SAM) or the Active-Directory to validate a user Id and password and to discover the Windows/Domain groups which he/she belongs to. When running on AIX/Linux/Unix, DB2 utilizes (by default) the local user account system. It is possible to configure DB2 to work with LDAP or with other methods of authentication like Kerberos. All DB2 database users are operating-system users, but not necessarily vice versa. In addition under Linux worlds there are some user Id restrictions as: 1) the length (no longer than eight characters), 2) the primary group to which it belongs to (other than guests, admins, users, and local), 3) the format (lowercase letters (a-z), numbers (0-9), and the underscore character (_)), 4) not to be part of DB2 or SQL reserved words.

To install and to manage DB2 a minimum of three users and groups are necessary (default names are provided by the setup utility):

- *Instance owner*
  The DB2 instance is created in the instance owner home directory. This user account controls all DB2 processes and owns all file systems and devices used by the databases contained within the instance (Figure 2.3.). The default user is DB2inst1 and the default group is DB2iadm1. Further instances will use a default progressive number identification (DB2inst2, DB2inst3, and so on).

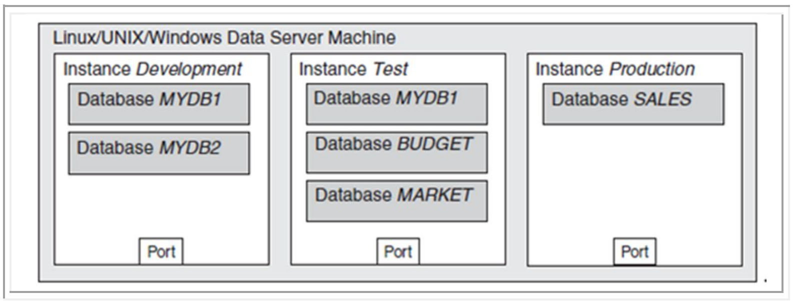


Figure 2.3 – DB2 LUW Instances

- *Fenced user*
  The fenced user runs user defined functions (UDFs) and stored procedures outside of the address space used by the DB2 database. The default user is DB2fenc1 and the default group is DB2fadm1. If this level of security is not needed, for example in a test environment, the instance owner can substitute the fenced user.
- *DB2 administration server user*
  It is the DB2 administration user that runs the DB2 Administration Server (DAS) on a system. DAS is a unique DB2 administration control point only used to execute administration tasks on local and remote DB2 servers. The default user is dasusr1 and the default group is dasadm1.

2.3.2 - PUBLIC

PUBLIC is the name of a group not defined in the operating system or in the external security facility. It is a special group to which everyone belongs. PUBLIC by default receives a few database authorities and/or database object privileges, depending on the type of operations performed.

2.3.3 - Roles

There are often locations where multiple users must share the same set of privileges. In this case it is better to manage this set as a whole, rather than deal with each privilege. A DB2 database role is an object that aggregates together one or more privileges or database authorities. It can be assigned to users, groups, PUBLIC or other roles. For example, we can define a data manager role that can insert, update and delete data on specific tables.
In the association of a role with a user, the user inherits all the privileges held by the role, in addition to the privileges already held by the user. Roles can be updated, without updating the privileges for every user. Roles are managed inside the database and DB2 can determine when an authorization changes and act accordingly.

2.4 - Security Model

The DB2 security model (Figure 2.4) consists of four major components: authentication, administrative authorization, database object security (also known as database object privileges), and row and column level security via Label-Based Access Control (LBAC).

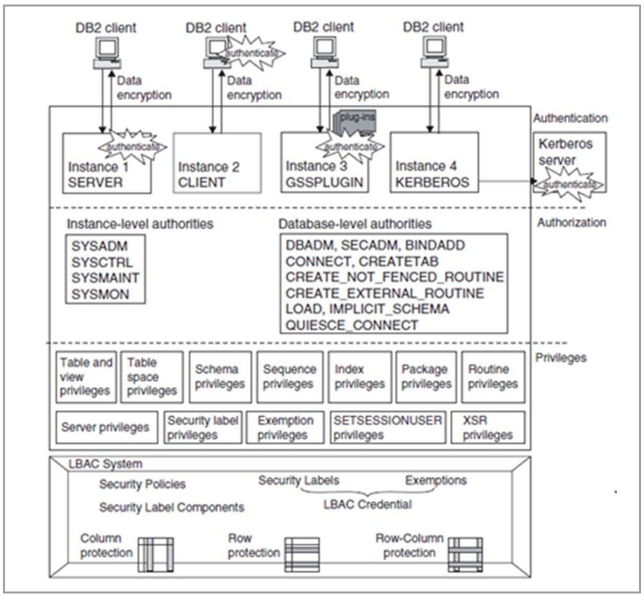


Figure 2.4 – DB2 LUW Security Model

2.4.1 - Authentication

It is the process of validating the supplied user Id and password with a security facility. This authentication occurs when a user tries to connect to a database or attach to an instance. The security facility is external to DB2, because user Ids and passwords are not stored inside a DB2 server. Authentication can occur in the following situations:

- when a DB2 server or client is using the operating system authentication service.
- when a customized loadable library via Generic Security Service (GSS) or a Kerberos security service are exploited.

The authentication process also determines which operating system groups the user belongs to, while not using GSS or Kerberos. Group membership lookup is essential, because it lets users inherit certain authorities or privileges through the groups to which they are associated.

2.4.2 – Authorization

Authorization is the process whereby the DB2 database manager obtains information about which database operations an authenticated user is authorized to accomplish, such as performing database manager maintenance operations and/or managing databases and database objects. Figure 2.5 shows the authorities supported in DB2. Data Control Language (DCL) is another SQL statement category that is used to control authorities and privileges. GRANT and REVOKE are the basic SQL statements for this category.
DB2 authorities control the following aspects of a database security plan:

- The authority level granted to a user.
- 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.

Authorities are made up of groups of privileges and higher-level database manager (instance-level) maintenance and utility operations. SYSADM, SYSCTRL, SYSMAINT, and SYSMON are instance-level authorities, meaning that their scope includes instance-level commands as well as commands against all the databases within the instance. These authorities can only be assigned to a group using the DBM CFG command on the Db2 server. The DBADM and SECADM authorities are assigned to a user or group for a particular database. This can be done explicitly using the GRANT command.

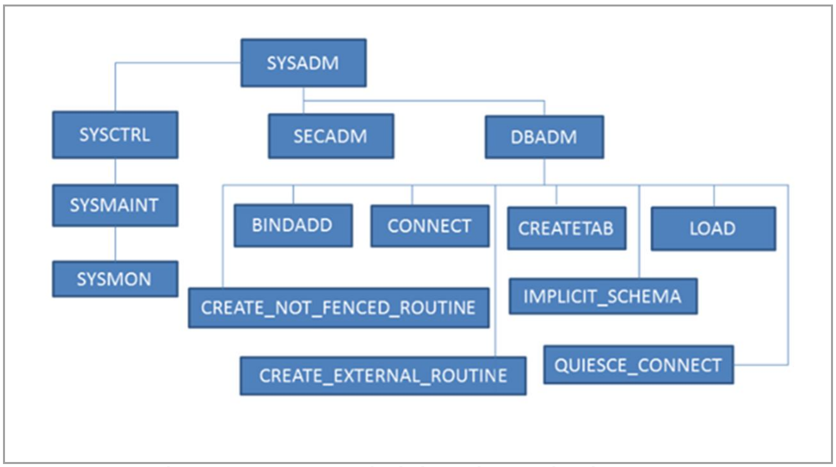


Figure 2.5 - DB2 administrative authority levels

The different sources of permissions available to an authorization Id are:

- Primary permissions: those granted to the authorization Id directly.
- Secondary permissions: those granted to the groups and roles of which the authorization Id is a member.
- Public permissions: those granted to PUBLIC.
- Context-sensitive permissions: those granted to a trusted context role.

Authorization can be given to users in the following categories:

- *System-level authorization*
  The system administrator (SYSADM), system control (SYSCTRL), system maintenance (SYSMAINT), and system monitor (SYSMON) authorities provide different degrees of control over instance-level functions. Authorities provide a way both to group privileges and to control maintenance and utility operations for instances, databases, and database objects.
- *Database-level authorization*
  Security administrator (SECADM), database administrator (DBADM), access control (ACCESSCTRL), data access (DATAACCESS), SQL administrator (SQLADM), workload management administrator (WLMADM), and explain (EXPLAIN) authorities provide control within the database. Other database authorities include LOAD (ability to load data into a table), and CONNECT (ability to connect to a database).
- *Object-level authorization*
  Object level authorization involves checking privileges when an operation is performed on an object. For example, to select from a table a user must have SELECT privilege on that table (as a minimum).
- *Content-based authorization*
  Views provide a way to control which columns or rows of a table specific users can read. Label-based access control (LBAC) determines which users have read and write access to individual rows and individual columns.

DB2 and IBM Data Studio provide numeorus features in conjunction with the DB2 audit facility for monitoring access, in order to define and manage the level of security required by a database installation.

2.4.3 - Users, groups, roles, PUBLIC privileges

Privileges can be granted/revoked to users and groups, to DB2 roles and to PUBLIC using DCL GRANT/REVOKE statements. In particular when a database is initially created, the following permissions are granted to PUBLIC:

- CREATETAB
- BINDADD
- CONNECT
- IMPLSCHEMA
- BIND on all packages created in the NULLID schema
- EXECUTE on all packages created in the NULLID schema
- CREATEIN on schema SQLJ
- CREATEIN on schema NULLID
- USE on table space USERSPACE1
- SELECT access to the SYSIBM catalog tables
- SELECT access to the SYSCAT catalog views
- SELECT access to the SYSSTAT catalog views
- UPDATE access to the SYSSTAT catalog views
- EXECUTE with GRANT on all procedures in schema SQLJ
- EXECUTE with GRANT on all functions and procedures in schema SYSPROC

If these permissions are not wanted, the option RESTRICTIVE on the CREATE DATABASE command must be specified.
According to the Role-Based Access Control (RBAC) model, the access control permissions to make use of enterprise objects are associated with roles. The relationship among users and the appropriate roles of which are members, can represent, at a level of abstraction, their position inside an enterprise structure. Once the privileges are assigned or updated at the role level, they are acquired by each user granted that role. After creating a role, the security administrator (who holds SECADM authority) can grant or revoke a role to or from a user, a group, or another role.

2.4.4 - Database Object Privileges

Controlling access to database objects is as important as authenticating users and managing administrative authorities. Privileges give users the right to access each individual database object in a specific way. Privileges can be granted explicitly and implicitly. Considering the supported privileges for each database object and their specific privileges we have:

- *Schema Privileges*
  There are three schema privileges:
  - CREATEIN allows users to create objects within the schema.
  - ALTERIN allows users to alter objects within the schema.
  - DROPIN allows users to drop objects within the schema.
- *Table Space Privileges*
  Tables are logically stored in table spaces, and table spaces are associated to physical storage devices. The USE privilege is necessary to be able to define tables in a table space. When a table space is created, its USE privilege is granted to PUBLIC by default. To restrict the table space usage, the USE privilege must be revoked from PUBLIC and individually granted to selected users or groups.
- *Table and View Privileges*
  There are many privileges that can be set for tables and views:
  - CONTROL provides users with all privileges for a table or view, as well as the ability to grant those privileges (except CONTROL) to others.
  - ALTER allows users to alter a table or view.
  - DELETE allows users to delete records from a table or view.
  - INDEX allows users to create an index on a table. This privilege does not apply to views.
  - INSERT allows users to insert an entry into a table or view.
  - REFERENCES allows users to create and drop a foreign key, specifying the table as the parent in a relationship.
  - SELECT allows users to retrieve data from a table or view.

  - UPDATE allows users to update entries in a table or view. This privilege can also limit users to update specific columns only.
  - ALL PRIVILEGES grants all the above privileges except CONTROL on a table or view.

- *Index Privileges*

  Privileges for managing indexes is only one: an index can only be dropped, after it is created. To change an index key, for example, is necessary to drop the index, before recreating it. The CONTROL privilege allows the grantee to drop the index.
- *Package Privileges*

  A package is a database object that contains the data access plan of how static SQL statements will be executed. A package needs to be bound to a database, before its associated program can execute it. The following are the privileges to manage packages:
  - BIND allows users to rebind an existing package.
  - EXECUTE allows users to execute a package.
  - CONTROL provides users the ability to rebind, drop, or execute a package, as well as the ability to grant the above privileges to other users and/or groups.

- *Routine Privileges*

  To be able to use a routine, a user must be granted with its only associated privilege: EXECUTE. It applies to all types of routines: functions, methods, and stored procedures.
- *Sequence Privileges*

  A sequence object generates unique, sequential numeric values. By default, the group PUBLIC can use any sequence object unless they are controlled by the USAGE privilege. The usage of certain sequence objects can be restricted by revoking USAGE from PUBLIC.
- *XSR Object Privileges*

  The eXtensible Markup Language (XML) schema repository (XSR) is a repository for all XML artifacts used to validate and process XML instance documents stored in XML table columns. These artifacts might contain a reference that points to an associated XML schema, Document Type Definition (DTD) of an XML document or other external entity. XSR helps the DB2 database system to manage dependencies on these XML artifacts without requiring changes, for example, to their location reference. XSR supports the creation of a copy of the information contained in an XML schema, DTD or external entity as an XSR object. A user must have the USAGE privilege on the XSR object to use it for XML validation or annotated XML schema decomposition.
- *Security Label Privileges*

  Security labels are database objects used in a LBAC system to define a certain set of security rules. Data is considered protected when a security label is applied to it. To fetch protected data, a user must obtain the proper access (ALL, READ, WRITE) for the associated security label.

2.4.5 - Implicit Privileges

Privileges usually are explicitly set with the GRANT statement. However users may also obtain privileges implicitly or indirectly by performing certain operations. For example, a user who is granted DBADM authority, receives also implicitly the BINDADD, CONNECT, CREATETAB, CREATE_EXTERNAL_ROUTINE,CREATE_NOT_FENCED_ROUTINE, IMPLICIT_SCHEMA, QUIESCE_CONNECT, LOAD privileges. When a privilege is revoked, its implicit privileges are not automatically revoked: they must be explicitly invalidated.
Database administrator must consider these kind of privileges and decide whether they are valid for the desired security policies.

3. - IBM Data Studio

IBM Data Studio (herein indicated as Data Studio) provides an integrated, modular environment for IBM DB2 LUW database development and administration. It also offers collaborative database development tools for IBM

DB2 for z/OS, IBM DB2 for i, and IBM Informix. Data Studio is the basic support for key tasks across the data lifecycle management, including administration, application design and development, monitor and view database conditions and query tuning.

- ✓ To administer the DB2 environment Data Studio supplies database management and maintenance support, authorization management, scripting, basic health and availability monitoring, job scheduling to automate changes to the database.

- ✓ To develop applications Data Studio offers a comprehensive suite of tools consisting of:
  - wizards and editors to create, test, debug, and deploy routines, such as stored procedures and user-defined functions (UDFs), written in many languages as SQL, PL/SQL, Java, XML Query Language (XQuery) and to develop XML applications.
  - JDBC, SQLJ, and pureQuery applications. pureQuery's functionality eliminates traditional JDBC programming by integrating the query language in a Java project, where database relationships are transformed into usable and optimized for DB2 Java objects.
  - environments to design and format queries by SQL, XQuery expression, XML descriptions, to view access plans, to get useful statistics with the purpose of analyzing and improving query performance.

- ✓ To manage data and database objects Data Studio let users:
  - connect to DB2 data sources, to get, manage and visualize data objects and their properties and to monitor and display database health conditions.
  - use data diagrams to visualize and print the relationships among data objects.
  - use editors and wizards to create, alter, or drop data objects.
  - modify privileges for data objects and authorization IDs.
  - copy tables, view and edit table data.

- ✓ More features are available for the administration of DB2 databases such as, among others:
  - tools to back up and recover databases and reverse engineering databases into physical models.
  - environments to create, validate, schedule, and run command scripts.
  - capabilities to manage table data including collecting statistics, reorganizing, importing, and exporting using different file formats.

3.1 - The packaging

Actually the last Data Studio suite (release 4.1.1)(*) is composed of two products:

- *Data Studio Client*

  It is an Eclipse-based integrated development environment for database administration, Java application development and query tuning that can be installed with other IBM software products sharing a common context. The client must be downloaded and set up by the IBM Installation Manager, an installation management tool that installs and maintains Installation Manager-based software packages. Normally the set up provides even the DB2 Analytics Accelerator Studio plug-in to administer IBM DB2 Analytics Accelerator for z/OS.
- *Data Studio Web Console*

  It is a web-based tool with health and availability monitoring features and job creation and management tools. An administrator may plan to install one web console for each client, or to share it among a group of clients.

Multiple instances of the Data Studio components can be deployed to represent and support a complex organization, where users play different roles with different access privileges. Figure 3.1 shows a scenario consisting of a database designer, multiple database administrators and application developers who all have different access rights to the test and production databases and to the Data Studio web console.

(*) – This release requires a fix pack. At the moment the previous release 4.1.0.1 results more stable.

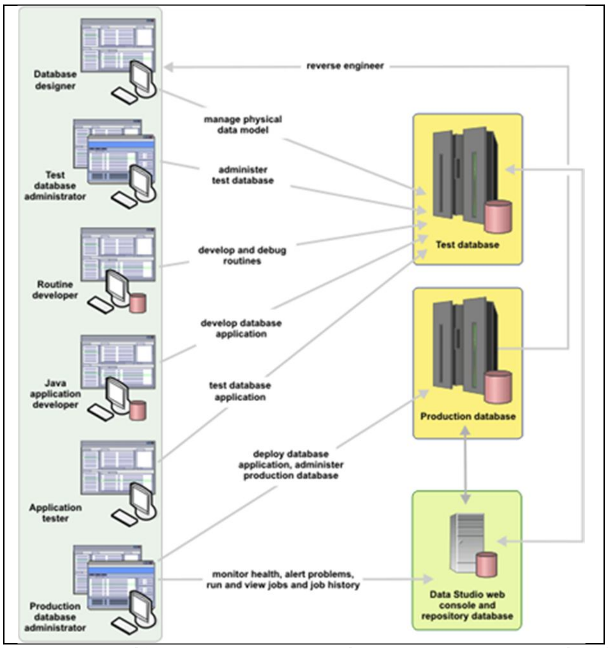

Database designer
Test database administrator
Routine developer
Java application developer
Application tester
Production database administrator
reverse engineer
manage physical data model
administer test database
develop and debug routines
develop database application
test database application
deploy database application, administer production database
monitor health, alert problems, run and view jobs and job history
Test database
Production database
Data Studio web console and repository database

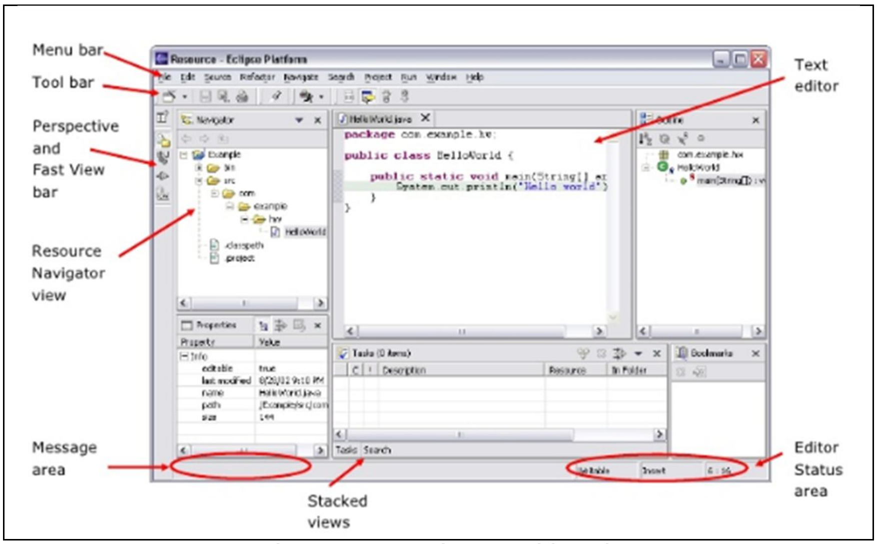
Figure 3.1 – Different DB2 roles in a complex organization

3.2 - Getting started with the Data Studio Client interface

The Data Studio Client workbench is the desktop development environment implemented by the Eclipse platform (Figure 3.2). It makes available a consistent tool set for the creation, management, and navigation of the user workspace resources. Each workbench window contains one or more perspectives. Normally the default Data Studio startup screen displays the Task Launcher (Figure 3.3) window, which highlights an Overview section with information about the product features and the following category of tasks:

- Accelerate to work with IBM DB2 Analytics Accelerator for the zero downtime Operating System (z/OS). It is a query accelerator capable of reducing response times for DB2 for z/OS queries by an order of magnitude.
- Design, Administer, Tune and Monitor database.
- Develop procedures.

Each category is described in more detail by its own tab.

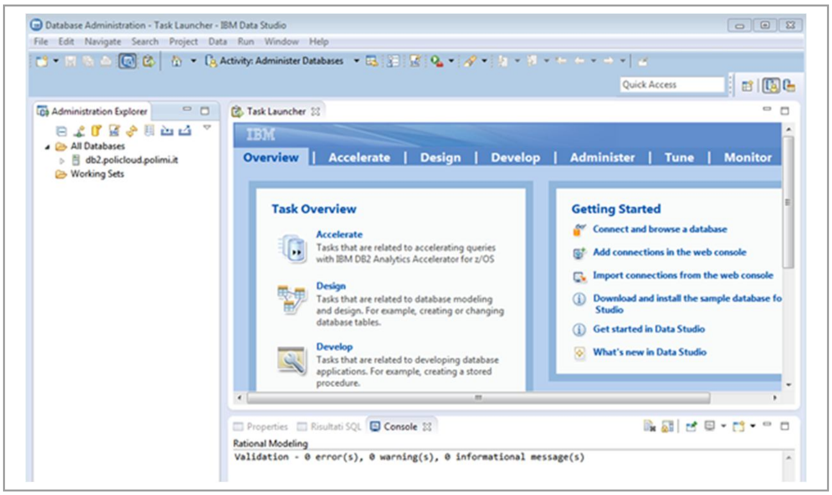


```
package com.example.hw;

public class HelloWorld {

    public static void main(String[] ar
        System.out.println("Hello world")
    }
}
```

Figure 3.2 – Eclipse workbench


Database Administration - Task Launcher - IBM Data Studio
File Edit Navigate Search Project Data Run Window Help
Activity: Administer Databases
Quick Access
Administration Explorer
All Databases
db2.policloud.polimi.it
Working Sets
Task Launcher
IBM
Overview | Accelerate | Design | Develop | Administer | Tune | Monitor
Task Overview
Getting Started
Connect and browse a database
Add connections in the web console
Import connections from the web console
Download and install the sample database fo Studio
Get started in Data Studio
What's new in Data Studio
Properties
Risultati SQL
Console

Accelerate
Tasks that are related to accelerating queries with IBM DB2 Analytics Accelerator for z/OS

Design
Tasks that are related to database modeling and design. For example, creating or changing database tables.

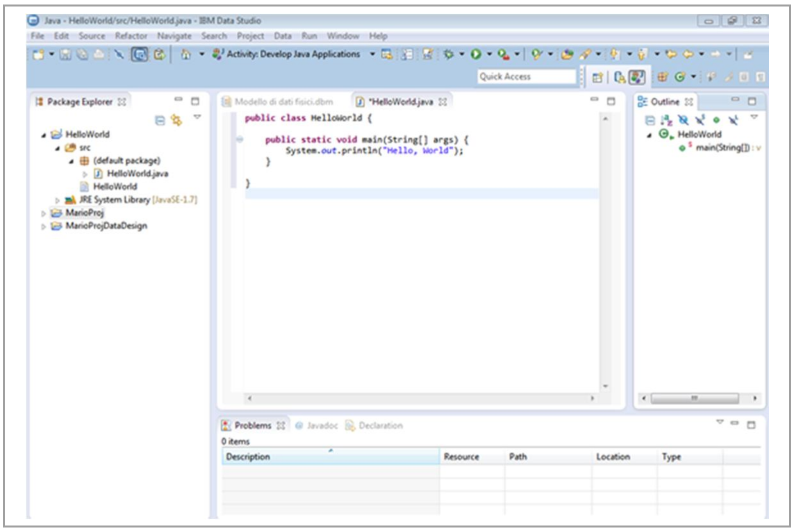
Develop
Tasks that are related to developing database applications. For example, creating a stored procedure.

Rational Modeling

```
Validation - 0 error(s), 0 warning(s), 0 informational message(s)
```

Figure 3.3 – IBM Data Studio workbench and Task Launcher

3.2.1 – Perspectives, Views and Editors



```
public class HelloWorld {

    public static void main(String[] args) {
        System.out.println("Hello, World");
    }

}
```

Figure 3.4 – IBM Data Studio perspectives, views and editors

The layout represented in Figure 3.4 composed of editors, views and tool bars specific for a Java project is named the Java perspective. Perspectives provide a set of functionality to accomplish a specific type of task or to work with particular types of resources. Window grouped perspectives contain views, share the same set of editors (sometime with slight modifications suited for particular perspective's tasks), control what appears in certain menus and tool bars based on a determinate task or role.

Perspective setup can be customized and new perspectives can be created. More than one perspective can be opened at a time. The shortcut bar has buttons, up on the right of the workbench, that allow to change opened perspectives by activating the identifying perspective tag.

An Editor is another visual component useful to edit or to browse a resource. The Java editor contains features for editing Java code (HelloWord.java file is opened within an editor in Figure 3.4).

A View is a visual component normally used to navigate a list, to display an information hierarchy or properties for the active editor. Package Explorer (Figure 3.4.) is the view showing some resources located within a workspace. Views can be chosen and opened through the tab *Window->Show View* (Figure 3.5). Views and Editors can be closed by clicking the symbol ✖ on the identifying tab.

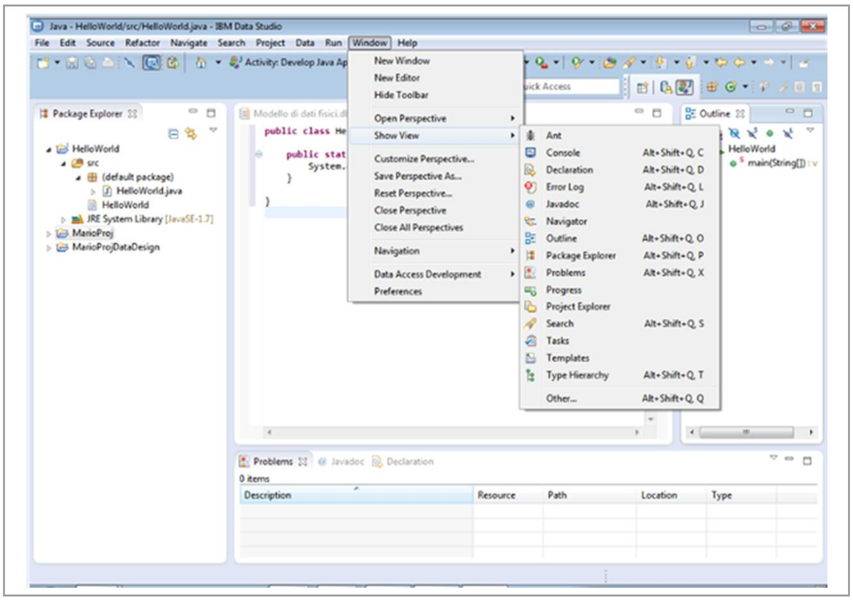


Figure 3.5 – Selecting a View

3.2.2 – The Database Administration Perspective

The Database Administration perspective (Figure 3.6) displays a set of views used by the database administrator to manage databases and the related administration tasks. The principal views are:

- the Administration Explorer: displays an overview of user configured and managed databases. Expanding the All Databases folder, the Administration Explorer shows the machines hosting DB2 servers, both local and remote, and the respective connection profiles to gain access to. Under each machine, the databases are organized into instances. Under each instance are indicated the specific connection objects to each database.
- the Object list editor: displays the results of user operation on database objects lists such as sorting and filtering tables, views, and indexes.
- the Properties view: displays the attributes of the current selection. For example, as a consequence of selecting a database object in the Object list editor, attributes of that object are grouped inside this view.
- the Data Project Explorer: shows the projects created, both automatically by the system, to store databases changes, and by the user that needs to organize his own scripts, stored procedures and packages.

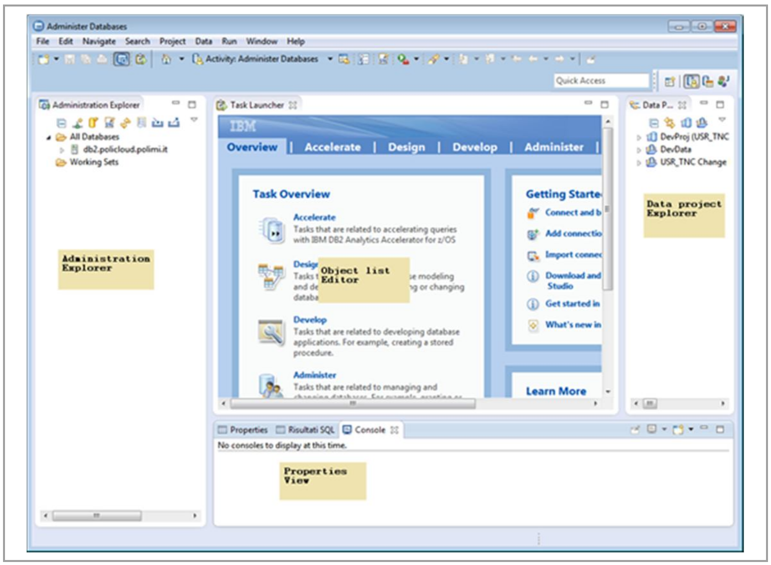


Figure 3.6 - Database Administration perspective

3.3 – Interfacing DB2 databases

A user can create a new DB2 database or connect to an existing one, both locally and remote, through the Administration Explorer window. By Clicking the down arrow (Figure 3.7) or the item self-explanatory icon, a menu with the operative options appears.

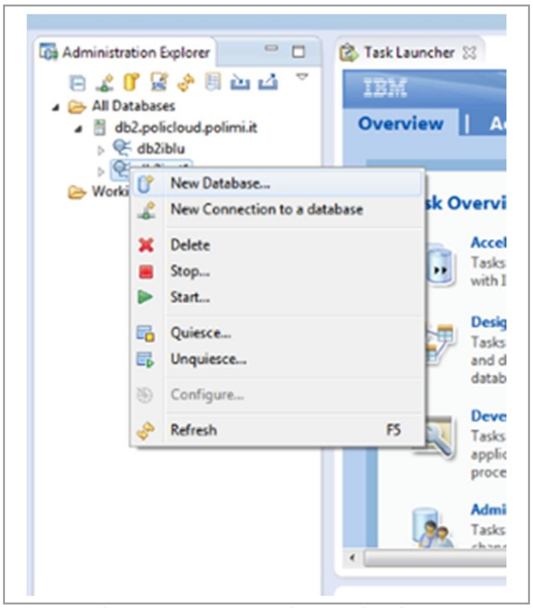


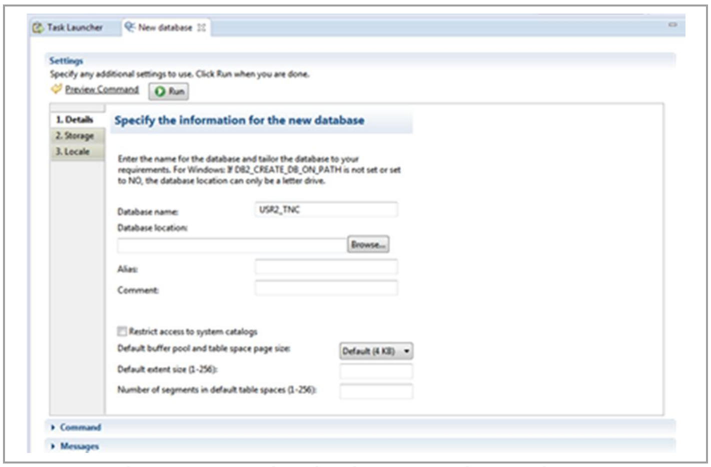
Figure 3.7 – Creating a database

3.3.1 – Creating

When the *New database* choice is selected, a window is displayed to collect the details of the instance that will contain the database. The detail elements are:

- Host: the IP address or the host name of the system, where the DB2 server is installed. On a local machine the name is localhost or 121.0.0.1.
- Port: the port number used to communicate with the DB2 instance listening on the DB2 server. Default port is 50000.
- Instance name: the name of the instance managing the database on the DB2 server. Default DB2inst1.
- User name: The user Id that is related to the database creation.
- Password: the password of the specified user.

Once the fields are set, the Data Studio client tests the connection to the server. If the connection is available the database creation assistant opens a New database view (Figure 3.8) to specify the database name and other values, as the buffer pools size and the optimizing parameters related to the table spaces. Scrolling down the window, the Preview Command tag is displayed: it allows to see the SQL command that will be executed in the database server by clicking the Run button. In this case the command is : “CREATE DATABASE USR2_TNC AUTOMATIC STORAGE YES ;”. If the command is successfully committed, the new database reference icon appears under the managing instance in the Administration Explorer view.



Settings

Specify any additional settings to use. Click Run when you are done.

Specify the information for the new database

Enter the name for the database and tailor the database to your requirements. For Windows: If DB2_CREATE_DB_ON_PATH is not set or set to NO, the database location can only be a letter drive.

Database name: USR2_TNC

Database location:

Alias:

Comment:

□ Restrict access to system catalogs

Default buffer pool and table space page size: Default (4 KB)

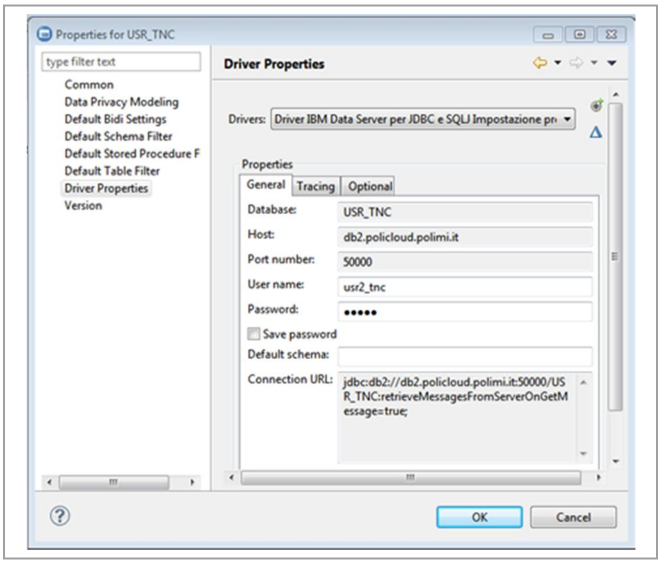
Default extent size (1-256):

Number of segments in default table spaces (1-256):

Figure 3.8 – The database creation assistant

3.3.2 – *Connecting to*

When a new database is created, a connection profile is also consequently defined. Connection profiles details, for a particular database, are stored into a file, so that other members of a team can share and re-use the same connections. Connection profiles also allow to save the password and standardize the JDBC driver for each connection. As shown in Figure 3.9. New Connection profiles to existing databases can be produced. By right-clicking in the Administration Explorer view the desired database name and choosing the *Connect* selection, the Driver property window of the specific connection profile appears. Some parameters are completed by default. user Id and password must be introduced. The Save password option memorizes the password to connect immediately to the storage system on the next access. The correctness of the introduced connection profile parameters can be verified activating the Test Connection button. Connection profile Driver properties can be modified by right clicking the database connection and selecting the Properties choice.



Drivers: Driver IBM Data Server per JDBC e SQLJ Impostazione pr

Database: USR_TNC

Host: db2.policloud.polimi.it

Port number: 50000

User name: usr2_tnc

Password: •••••

□ Save password

Default schema:

Connection URL: jdbc:db2://db2.policloud.polimi.it:50000/USR_TNC:retrieveMessagesFromServerOnGetMessage=true;

Figure 3.9. – Connection profile Driver properties

3.3.3 – *Instances*

A DB2 Instance is similar to a copy of the DB2 manager system: it controls its own databases and sets specific parameters for the environment. Data Studio lets a user manage instances, but they must be created or dropped working directly on the DB2 server with the command line processor on Windows, or from a Linux/UNIX shell.

A particular user, the instance owner, is associated to the instance. Its name is identical to the instance identifier: there is no way to separate them. It acquires SYSADM authority over the instance and all databases inside that instance. SYSADM authority is the highest level of authority in DB2 and lets this user do anything within his

databases (see chapter 1.). In general, it is a good practice to have a very limited number of database administrators that have access to the SYSADM (or instance owner) user Id. Multiple instances are useful: a) for tuning; b) to prevent or diminishing the impact of database failure respect to the other environments; c) to differentiate permissions; d) to separate fix packs; e) to distinct test and production worlds.
A DB2 instance can be started and stopped. As a consequence its memory areas are freed or instantiated and listeners for network protocols (ex. TCP-IP) previously enabled for the instance are deactivated or activated. In the Administration Explorer, an instance associated with one or more database is visualized under the machine node hosting the databases. By right clicking on the instance name, specific options to manage the instance are displayed.

3.3.4 – Browsing

By expanding a database in the Administration Explorer, an enumeration of DB2 object folders is shown. As a result of selecting an object folder, the Object list editor evidences the included components. This method can be used to descend the tree of the different objects explicitly selected.
The Object list editor can limit the list of objects retrieved from the database using a search box to filter objects by name. Wild characters ("% " character to match any number of characters in the name, " _" character to match any single character) help to further decrease the research produced items.
The Object list editor also provides a way to move among associated objects related to a specific one. By right-clicking one of the displayed objects, the Show sub-menu appears (Figure 3.10) presenting options for navigating to its linked objects.

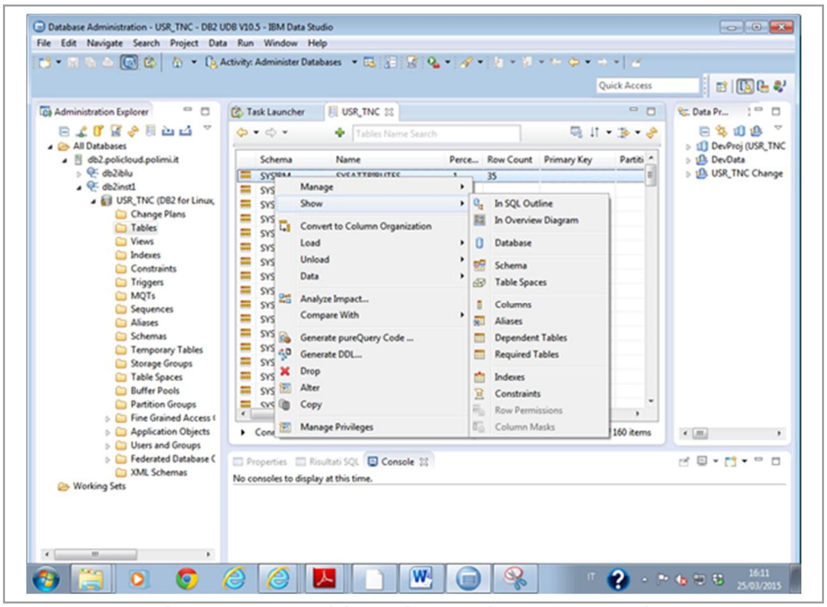


Figure 3.10 – Object Show sub menu selection

3.4 – Generating DB2 database objects

Data Studio allows to generate many of the DB2 database objects displayed in the Administration Explorer. When a user right clicks the object identifying icon, a linked sub menu is shown with specific creation commands. Some objects such as views, indexes and tables, are grouped inside a schema object, that, if not already defined, takes by default the identifier of the user name that created one of the related objects. Moreover the client supports change plans creation to collect database changes as a whole and deploy them in a single step. During the object creation phase a *planned changes* blue bar across the top of the object list editor appears, indicating that changes to the database have not yet been deployed. In particular a Δ symbol occurs next to the new object revealing that a modification is ongoing. On the bar there are some blue delta buttons to move to other objects subjected to changes. By selecting the *Review and deploy changes* button a windows opens where the generated DDL to be committed to the database is shown and some deploy parameters can be established. Then, after having accepted the settings, a file opens displaying the DDL, that, if it is needed, can be edited for further refinements. When the Run button is activated, the modifications are deployed (*). The progress of the deployment commands is shown in the SQL Results window in the Properties view. A user can click the symbol if he/she decides to discard the possible database changes.

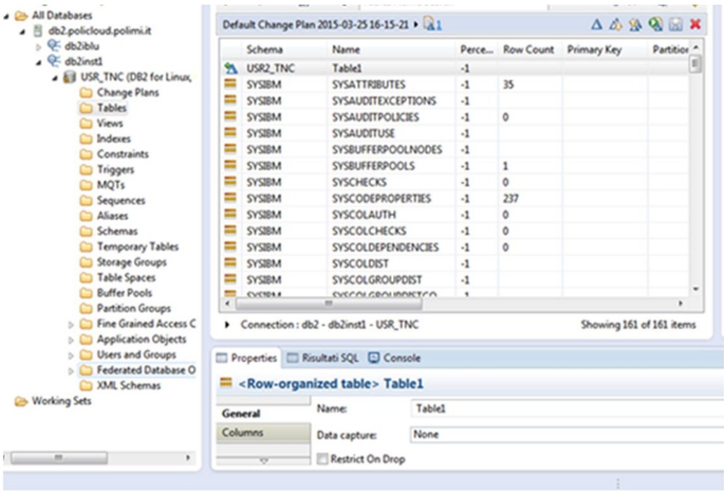
3.4.1 Creating Schema Object

A DB2 schema allows to organize database objects together and prevent name conflicts, when two objects may share the same name. While some of the schemas are already created by the DB2 installation to store system catalog tables, a user can create his/her schema to group together the generated objects. As previously mentioned, by right clicking the Schema folder in the Administration Explorer a sub menu with the create selection is shown. Usually, specific parameters for an object can be inserted in the General tab input fields of the Properties view. In this case the only accepted value for a schema is the name. The related deployment command is: “CREATE SCHEMA schema-name;”.

3.4.2 Creating Table Object

A DB2 table belongs to a schema. When the Tables folder in the Administration Explorer is right clicked to create a table, a window opens to select the schema in which the table must be created. Then the Properties view displays the new table parameters (Figure 3.11) to be set. By selecting the Column tab a user can add columns. If the symbol is clicked, an empty column appears to allow a user define the column parameters (Figure 3.12 represents the columns of a table named Author2). These items are:

- *Name*: name of the column.
- *Primary Key*: this box must be activated if the column must be the primary key for the table.
- *Data Type*: data type of the column. By clicking this field a pull down menu opens showing all the data types supported.
- *Length*: the length of the column. For some of the data types it is fixed and cannot be set.
- *Scale*: specifies the scale for the column type, wherever applicable. On the contrary a user won’t be able to edit this field.
- *Not Null*: this box must be activated, if the column value cannot be null. This option is automatically selected for primary key columns, because primary keys are not allowed to be null.
- *Generated*: this box is selected when a user requests that the DB2 system automatically generates the value of the column based on a provided default value or expression.



Default Change Plan 2015-03-25 16-15-21 ▸ 1

Schema	Name	Perce...	Row Count	Primary Key	Partitio
USR2_TNC	Table1	-1			
SYSIBM	SYSATTRIBUTES	-1	35		
SYSIBM	SYSAUDITEXCEPTIONS	-1			
SYSIBM	SYSAUDITPOLICIES	-1	0		
SYSIBM	SYSAUDITUSE	-1			
SYSIBM	SYSBUFFERPOOLNODES	-1			
SYSIBM	SYSBUFFERPOOLS	-1	1		
SYSIBM	SYSCHECKS	-1	0		
SYSIBM	SYSCODEPROPERTIES	-1	237		
SYSIBM	SYSCOLAUTH	-1	0		
SYSIBM	SYSCOLCHECKS	-1	0		
SYSIBM	SYSCOLDEPENDENCIES	-1	0		
SYSIBM	SYSCOLDIST	-1			
SYSIBM	SYSCOLGROUPDIST	-1			

Connection : db2 - db2inst1 - USR_TNC

Showing 161 of 161 items

<Row-organized table> Table1

Name: Table1

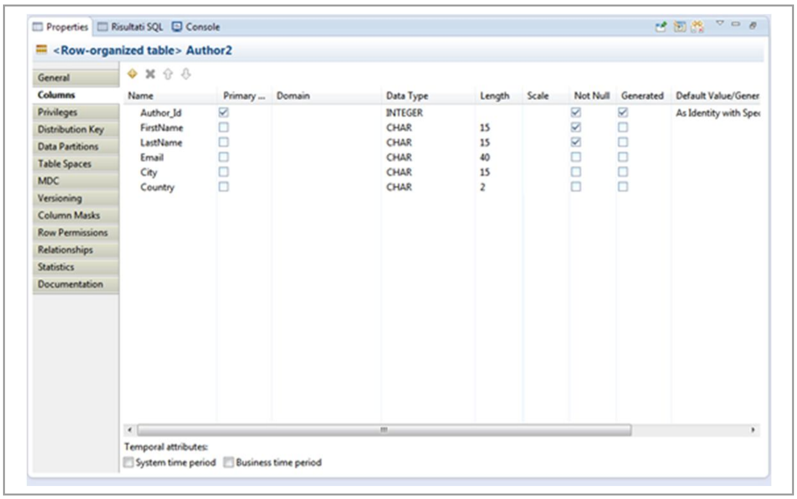
Data capture: None

☐ Restrict On Drop

Figure 3.11 – Table Property view

(*) The command sequence represented by the and buttons is common to the most of the administrative and development procedures available in Data Studio.

- *Default Value/Generated Expression*: when the Generated box is selected, the user must define a default value or an expression that the DB2 system will evaluate to generate the column value, whenever a value is not specified in the INSERT statement.



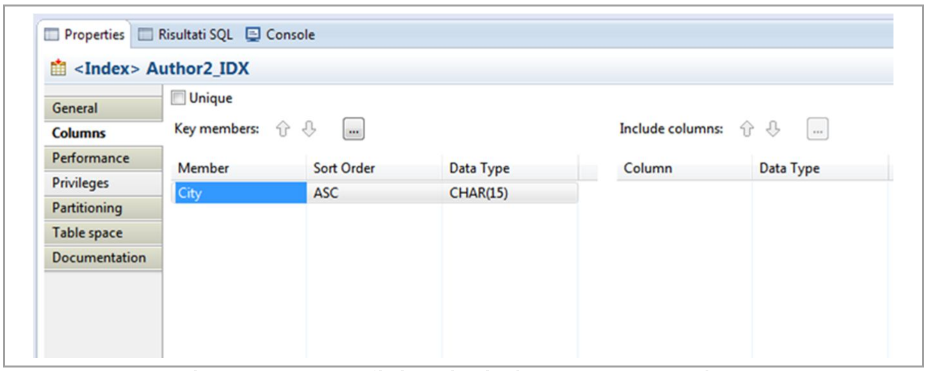
Name	Primary ...	Domain	Data Type	Length	Scale	Not Null	Generated	Default Value/Gener
Author_Id	☑		INTEGER			☑	☑	As Identity with Spec
FirstName	□		CHAR	15		☑	□	
LastName	□		CHAR	15		☑	□	
Email	□		CHAR	40		□	□	
City	□		CHAR	15		□	□	
Country	□		CHAR	2		□	□	

Temporal attributes:
□ System time period □ Business time period

Figure 3.12 – Adding columns to a table in the Property view

3.4.3 Creating Indexes

A DB2 index is linked to a table and the related schema. To create an index a user right-clicks the Indexes folder in the Administration Explorer and selects the associated choice. Once specified the schema and the table name in a pop up window, the Properties view for the new index is displayed. In the General tab an index name can be indicated. In the Column tab, to select the table columns that will be used for the index, the ellipsis button (…) must be activated. Figure 3.13 represents the components columns of index Author2_IDX for the Author2 table.



□ Unique

Key members:

Member	Sort Order	Data Type
City	ASC	CHAR(15)

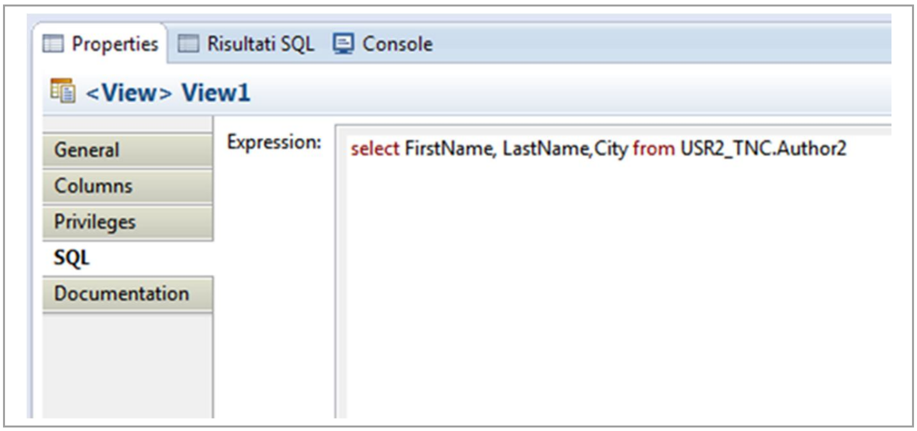
Include columns:

Column	Data Type

Figure 3.13 – Defining the index column members

3.4.4 Creating Views

A View is a representation of data stored in one or more tables. A View is organized internally to DB2 by SQL statements and doesn’t contain the data, working differently from materialized query tables (MQTs). Data stored in tables are retrieved and represented without exposing the details of how the data is actually structured. A view can SELECT an entire table, JOIN together multiple tables, or exclude specific columns to restrict access to the data. When the Views folder in the Administration Explorer is right clicked to create a view, a window opens to let user select the schema in which the view must be created. Then the Properties view appears and the attributes of the new view can be set. In the General tab, the name of the view must be specified. In the SQL tab, the columns for this view are defined, using an SQL query, edited in the Expression field. Eventually the Update button must be clicked to change the view definition.
Figure 3.14 represents the query statements to retrieve specific columns from Author2 table. The View is run on the database with the following SQL command: ’CREATE VIEW USR2_TNC.”VAuthor2” (FIRSTNAME, LASTNAME, CITY) AS select FirstName, LastName, City from USR2_TNC.”Author2”.



Expression: `select FirstName, LastName,City from USR2_TNC.Author2`

Figure 3.14– Defining the query for the View

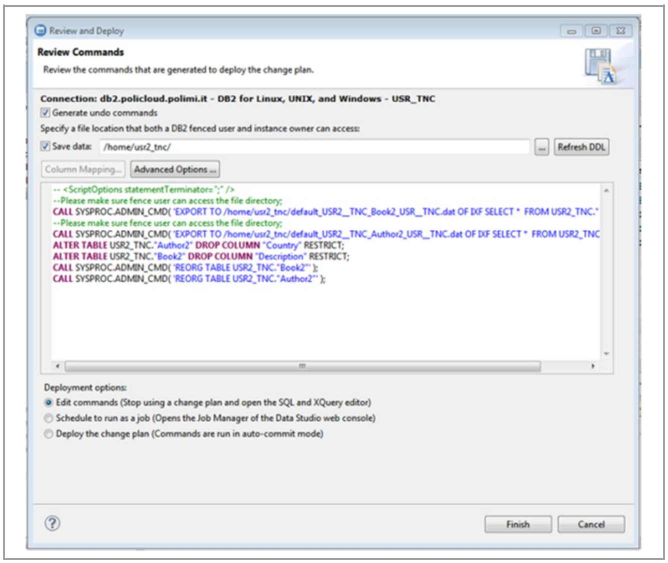
3.4.6 - Deploying multiple changes with a change plan

In the previous sections, every specific change to the database has been verified by clicking the Review and Deploy Changes button and committed to the database by activating the Run ▶ button afterwards. However these variations are recorded in a change plan, which groups multiple related changes together to deploy them in a single step. These plans are stored in the Change Plans folder of the Administration Explorer and can be modified, dropped and run again. In both cases (reviewing and deploying in a single or multiple steps) Data Studio opens the Review and Deploy window, with a preview of the commands that will be used to deploy the changes to the database. Figure 3.15 represents the database changes to be committed regarding two tables (Author2 and Book2), where two columns have been deleted.

The Save data check box lets a user define the path location on the DB2 server, where backup files, needed to preserve the contents of the database tables, will be saved. When this check box is enabled, Data Studio will create backup files for any dropped tables and changes that require to drop and to recreate a table. The Column Mapping button controls how the client arranges the columns of an existing table, respect to the columns of a varied table.

The Advanced Options button displays a window to specify which supporting commands Data Studio should generate together with the changing objects commands. Commands can be edited in the SQL and XQuery editor by activating the Edit selection. Deployments options can be selected: for example changes could be scheduled for a future time, by activating the Schedule alternative.

After having successfully deployed the modifications to the database, the planned changes bar, as previously indicated, no longer opens in the Object list editor because local objects match the objects in the database.



Review the commands that are generated to deploy the change plan.

Connection: db2.policloud.polimi.it - DB2 for Linux, UNIX, and Windows - USR_TNC

☑ Generate undo commands

Specify a file location that both a DB2 fenced user and instance owner can access:

☑ Save data: /home/usr2_tnc/

```
-- <ScriptOptions statementTerminator=";" />
--Please make sure fence user can access the file directory;
CALL SYSPROC.ADMIN_CMD( 'EXPORT TO /home/usr2_tnc/default_USR2__TNC_Book2_USR__TNC.dat OF IXF SELECT *  FROM USR2_TNC."
--Please make sure fence user can access the file directory;
CALL SYSPROC.ADMIN_CMD( 'EXPORT TO /home/usr2_tnc/default_USR2__TNC_Author2_USR__TNC.dat OF IXF SELECT *  FROM USR2_TNC
ALTER TABLE USR2_TNC."Author2" DROP COLUMN "Country" RESTRICT;
ALTER TABLE USR2_TNC."Book2" DROP COLUMN "Description" RESTRICT;
CALL SYSPROC.ADMIN_CMD( 'REORG TABLE USR2_TNC."Book2"' );
CALL SYSPROC.ADMIN_CMD( 'REORG TABLE USR2_TNC."Author2"' );
```

Deployment options:

● Edit commands (Stop using a change plan and open the SQL and XQuery editor)

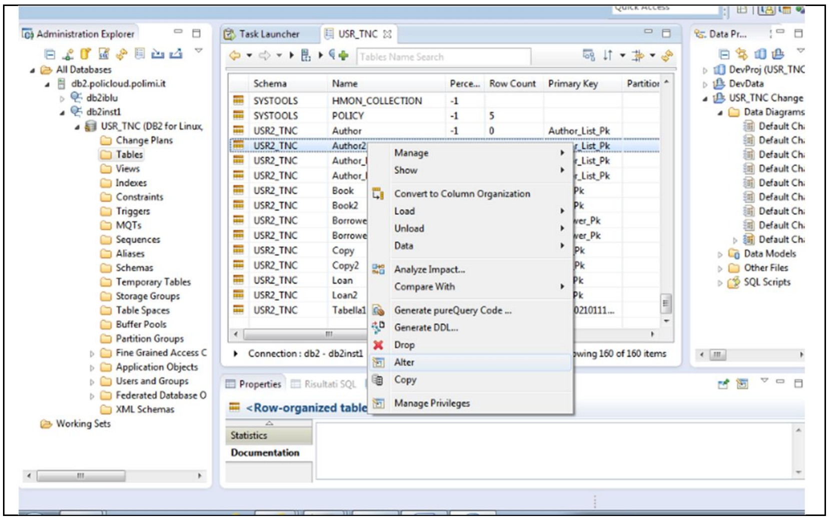
○ Schedule to run as a job (Opens the Job Manager of the Data Studio web console)

○ Deploy the change plan (Commands are run in auto-commit mode)

Figure 3.15 – Review and Deploy window

3.5 – Managing DB2 database tables

In a relational database, tables are centric as data container and their definition and relationships extremely important for data query operations. As introduced in paragraph 3.3.4 by right clicking a table in the Object list editor (Figure 3.16) a sub menu is visualized where are exposed client tools that assist in table maintenance.



Schema	Name	Perce...	Row Count	Primary Key	Partitio
SYSTOOLS	HMON_COLLECTION	-1			
SYSTOOLS	POLICY	-1	5		
USR2_TNC	Author	-1	0	Author_List_Pk	
USR2_TNC	Author2			r_List_Pk	
USR2_TNC	Author_l			r_List_Pk	
USR2_TNC	Author_l			r_List_Pk	
USR2_TNC	Book			Pk	
USR2_TNC	Book2			Pk	
USR2_TNC	Borrowe			wer_Pk	
USR2_TNC	Borrowe			wer_Pk	
USR2_TNC	Copy			Pk	
USR2_TNC	Copy2			Pk	
USR2_TNC	Loan			Pk	
USR2_TNC	Loan2			Pk	
USR2_TNC	Tabella1			0210111...	

Connection : db2 - db2inst1

owing 160 of 160 items

Figure 3.16 – Tools for database object maintenance

This sub menu contains the following items:

- *Manage*:
  - to collect accurate information on tables and indexes, useful for query tuning and optimized access plan selection.
  - to set integrity in case of referential constraint violation.
  - to reorganize tables and indexes.

- *Show*: to show database objects related to the table.
- *Convert to Column Organization*: to exploit the DB2 columnar capability. This table format suits star schema data marts (*) very well, providing significant improvements to storage, query performance, and ease of use, through simplified design and tuning.
- *Load*: to import data into or to bring back the data from the file system to the tables. Importing (as exporting) is also useful when a user wants to import a large volume of data into a table that includes large objects. Three file formats are supported: delimited, integrated exchange, and worksheet. They are respectively indicated with the following acronyms and descriptions:
  - DEL: to exchange the data between different database managers and file managers. The data is stored in a delimited text file, where rows, columns, and character strings are demarcated by a character, usually "," or ";".
  - IXF: is a rich format to store data in a binary format together with structural information about the table (expressed in the DDL language). Thereby the table itself is recreated, when the file is imported.
  - WSF: to exchange the data with products like Lotus® 1-2-3® and Symphony™.

  It possible to export large object (LOB) or XML data into one or more separate files.
  Importing data may be done with:

  1. the Import Utility: by which a user defines, among many options, one of the following import modes:
     - INSERT: the imported data will be appended to the existing data in the table.

(*) Star schema, in data warehousing and business intelligence, is the simplest form of a dimensional model, in which data is organized into facts and dimensions.

  - INSERT_UPDATE: a row is updated, if that particular row already exists. Otherwise a new row is inserted.
  - REPLACE: existing data is overwritten.

2. the Load Utility: by which large quantities of data are faster loaded than the Import Utility. Indeed it writes formatted pages directly to the database, not running the individual insert as the Import functionality does. Import modes are only INSERT and REPLACE. The load operation maintains: unique constrains, range constrains for partition tables, generated columns, LBAC security rules. Regarding the other constrains, the table is placed in the SET INTEGRITY PENDING mode at beginning of the load. After the load has been completed, the SET INTEGRITY command must be executed, to take the table out of the pending state.
3. The SQL Utility: by which a full SELECT is automatically created

- *Unload*: to export the data from a table to the file system, in one of the three file format previously specified for the Load item. Exporting data may be made with:
  1. The Export utility: by which, a user can manage, among many options, the SQL statement (in the Source tab of its setting window) automatically generated, that selects the data to export in the form of a full SELECT command**.** This command can be edited to select any specific column.
  2. The InfoSphere Optim High Performance Unload: by which a high speed solution for unloading, extracting, and repartitioning data in DB2 is offered to users. It improves data availability, mitigates risk, and accelerates the delivery of database migrations.
  3. The SQL Utility: by which a full SELECT extracts all table data.

- *Data* : to display data with a Browser utility, to edit data with the Edit utility, to query data with the Select utility.
- *Analyze Impact*: to be used whenever a change to an object is considered. It shows the entire group of objects that are dependent on a specific one.
- *Comparing With*: to compare multiple resources and to present the comparison results in a special Compare editor, where the differences between two files, for example, are displayed. The toolbar of the Compare editor includes many buttons to manage the Compare editor Viewer and to accomplish merging operations between files.
- *Generate pureQuery Code*: to create a Java bean following pureQuery methodology. An interface is generated containing one abstract annotated method for each SQL statement created.
- *Generate DDL*: to generate a DDL script that can be run on a target database. This is the simplest way to duplicate a database table. Many database objects provides this option. The Generate DDL wizard lets users select several options to be inserted in the generated DDL, including drop statements, fully qualified and delimited names, dependent object, and so forth. After choosing the preferences, the generated DDL is displayed to be run against a database server or simply saved into a local project for later use.
- *Drop*: to eliminate a database table.
- *Alter*: to modify existing objects. If the object to be altered is a table, its attributes are displayed in the Properties view, where many properties of the table can be changed, added or dropped and table's statistics and relationships be visualized.
- *Copy*: to copy a table that will be pasted into another project or folder.
- *Manage Privileges*: to manage authorizations on the specific table.

3.6 – Viewing DB2 objects relationships

The number and type of relationships among the database objects are essential to understand the structure of an existing database and the impact of planned changes to it. Data Studio provides visualizing tools to show these interconnections (dependencies and associations), helping the database administrator and developers to minimize the consequences on the database framework due to a database reorganization.

Data Studio can detect database objects dependencies by means of the Analyze Impact component. Impact analysis identifies the potential consequences of a change, such as adding a new column to a table that is used in a job. To detect dependencies through the impact analysis functionality, a table must be selected in the Object list editor and right clicked to choose the Analyze Impact option. The Impact Analysis window opens: here the Dependent objects option must be flagged to locate the objects that depend on the table. After clicking the Ok button, the Model Report view opens and lists the dependent objects, and the impact analysis diagram editor also displays a visual representation of the entire dependency structure. For example, using the Analyze Impact functionality on a table space, would display all of the tables, indexes, containers and other database objects that could be dependent on that table space and even the objects upon which this object depends. The system ensures that imposed changes do not invalidate the dependent objects, by offering to drop or recreate those objects. However, when a dependent object is composed of SQL statements, such as a view, it is necessary to verify that planned changes will not invalidate the SQL expressions of the dependent object. The Impact Analysis window offers also other options such as contained and recursive objects, that allow to know more about deeper object relationships.

Data studio provides also the functionality to create and show the entity–relationship (ER) diagrams of the database objects: it is an handy tool to help to understand the associations among different tables. Using ER diagrams during development can be crucial to understand the database design and increase user productivity. This kind of diagram is based on the ER model. It represents a process as a set of components (entities) linked with each other by relationships that express both their dependencies and their requirements. Entities may be characterized through various properties (attributes). To generate an overview ER diagram a table must be selected in the Object list editor and right clicked to choose the *Show->In Overview Diagram* option. The Overview Diagram Selection window opens: here the tables to be included in the overview diagram can be selected. After clicking the Ok button the overview diagram is displayed (Figure 3.17).

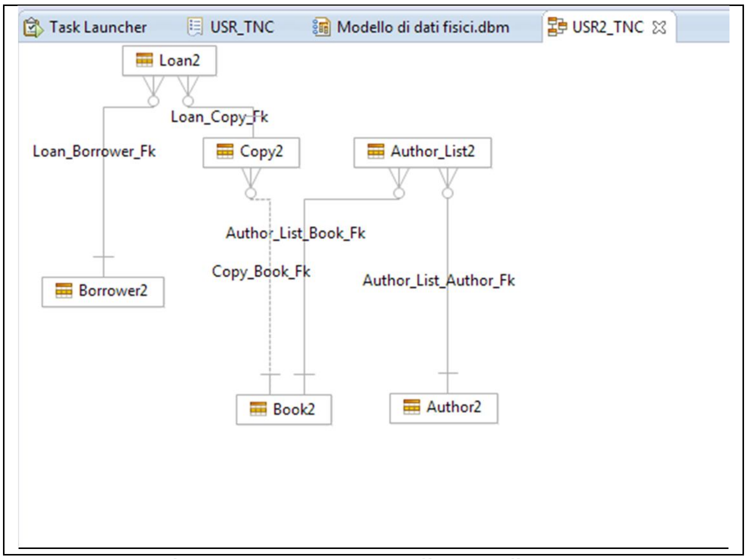


Figure 3.17 – Data Studio ER diagram

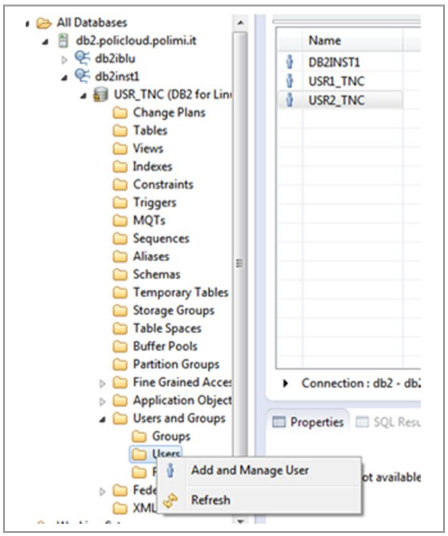
3.7 - Administering DB2 database security

Data Studio is able to interface the DB2 security model, described in chapter 1, helping the database administrator to secure data, by granting only the specific privileges that a user needs.

3.7.1 - Creating user, group and roles

Working on database authorities makes sense for existing users, groups and roles. It is important to remember that users and groups are database identifiers that should match existing figures defined outside DB2. Figure 3.18 shows how it is possible to add and modify users and groups and to create roles by expanding the Users and

Groups folder in the Administration Explorer and by right clicking the specific folders inside (Groups, Users and Roles).



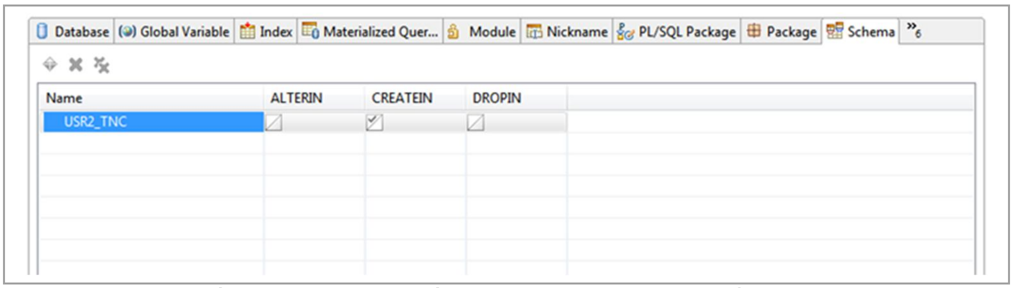
Name
DB2INST1
USR1_TNC
USR2_TNC

Figure 3.18 – Managing DB2 User, Groups and Roles

Specific authorities for every database object can be assigned/denied to the database members through the “GRANT/REVOKE privileges” commands. In DB2 a user added to a group or to a role inherits all the privileges held by the group/role, in addition to privileges already held by the user itself. Figure 3.19 displays the Properties View for a user and in particular the Privileges section of a Schema object.

3.7.2 - Assigning privileges

When a database member is created or modified, the bar at the top of the Privileges section presents all the database objects. In this way it is possible to grant privileges on multiple objects. To allow a privilege, the corresponding box must be checked, so that a checkmark appears. Some objects allow user/group/role granting privileges to others members: in this case the box must be clicked again, until two checkmarks are displayed. To revoke privileges, the checkmarks must be cleared.



Name	ALTERIN	CREATEIN	DROPIN
USR2_TNC	□	☑	□

Figure 3.19 – Managing DB2 User, Groups and Roles

In general, privileges for most of the DB2 objects can be administered by right clicking the specific object icons to select the Manage Privileges choice. Afterwards, the Privileges tab for the managed object, wherever applicable, is displayed in the Property view.

3.8 – Managing DB2 database services

Data Studio is equipped with services that: a) control, optimize and keep a DB2 database environment up, thereby running it efficiently and b) help preventing and recovering from failures. These operations are listed in the administrative database menu, usable after a database connection, and displayable by right clicking the specific database icon in the Administrative Explorer window.

3.8.1 - DB2 database logging

Database logging is an important part for database availability, because database logs make it possible to recover from a failure and to synchronize primary and secondary databases. All databases have logs associated with them. These logs record any SQL update command committed to the database. If a database needs to be restored to a point beyond the last full, offline backup, logs are required to roll the data forward to the point of failure.
DB2 logging can be categorized in two types, each providing a different level of recovery capability:

- Circular logging: it is the default behavior when a new database is created. The changes are logged only for those transactions that are not committed and only the data from the latest backup can be recovered. The database must be offline (inaccessible to users) when a full backup is taken.
- Archive logging: to log all the changes including the committed ones. With this kind of logging, differently from the Circular method, a roll forward recovery is able to consider both archived logs and active logs to restore a database either to the end of the logs, or to a specific point in time. Archive logging allows online backup operations.

Logging types, the number and the size of the logging files and their location can be determined selecting *Set Up and Configure -> Configure Database Logging* in the in the administrative database menu. Once the options are set, by clicking the Run button, the required logging is configured.

3.8.2 - Backing up and recovering databases

The most important part of a maintenance plan is backing up databases regularly, so that they can be recovered in case of data loss events as deletion or corruption due to crash or database failure. Back up files are used to restore the original data earlier archived. Backing up also allows to move complete databases from one location to another in the same or a different compatible system. Because not all system combinations are supported, compatibility, must be verified at the IBM Knowledge Center link reported in the Introduction chapter.
To back up a database, in the administrative database menu the link *Back Up and Restore* functionalities must be highlighted (Figure 3.20) to set the *Back Up* operation. A new editor opens to specify additional settings, through the following different tabs (Fig. 3.21):

- Backup Image tab: to choose the location and the media type, where to store the backup image file.
- Backup Type tab: to specify whether a user wants to back up the entire database or backup selective table spaces.
- Backup Options tab: to create three different types of back up:
    1. full backup: to produce a complete backup of the entire database.
    2. incremental backup: to save any data that is new or has changed since the last full backup
    3. delta backup: to save any data that is new or has changed since the last backup of any type: full, incremental or delta.

The choice Availability should be flagged if the database must be online during the backup process. This feature is possible only when archive logging is being used. Once the options are set, by clicking the *Run* button the database backup is created.
To restore a DB2 database the *Restore* option in the administrative database menu, under *Back Up and Restore* functionalities (Figure 3.20), must be selected. A new editor opens to specify additional settings through the following different tabs:

- Restore Type tab: to define if an existing database must be restored or a new one must be created.
- Restore Objects tab: to specify the restoring modality: a) only the history file; b) the entire database or specific table spaces. A history file contains the information regarding the backups taken in the past: it supports the recover command finding the appropriate backups to be used. Restoring and recovering processes map respectively to: a) the RESTORE SQL command and b) the RECOVER SQL command, that is made by the SQL sequence RESTORE and ROLLFORWARD commands.
- Restore Containers tab: to set a new container for the table spaces in the database, when it must be restored on a different system, where the same container path does not exist.

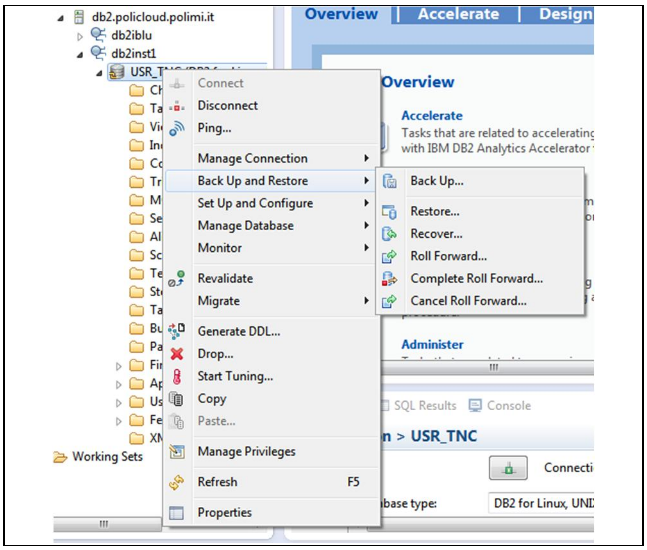
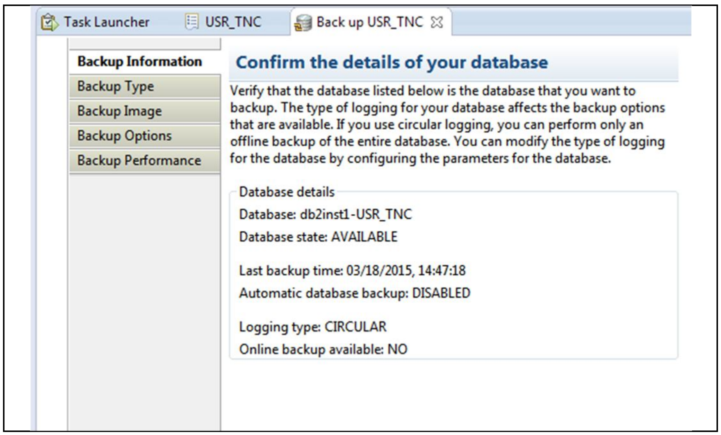


Figure 3.20 – Backup and restore with Data Studio



Confirm the details of your database

Verify that the database listed below is the database that you want to backup. The type of logging for your database affects the backup options that are available. If you use circular logging, you can perform only an offline backup of the entire database. You can modify the type of logging for the database by configuring the parameters for the database.

Database details

Database: db2inst1-USR_TNC

Database state: AVAILABLE

Last backup time: 03/18/2015, 14:47:18

Automatic database backup: DISABLED

Logging type: CIRCULAR

Online backup available: NO

Figure 3.21 – Window to set Backup settings

- Restore Options tab: to select if: a) replacing the history file, or b) restoring only the backup image, or c) restoring also the transactions, accomplished between the time the backup image was taken and a restore operation has been performed. This last operation is called roll forward. In general, the option to remove all connections to the database, before starting the restore operation, must be flagged, so that the restore operation does not fail.

Once the options are set, by clicking the Run button the database restore is created.
Another important service related to the recovering process is the roll forward operation, that applies transactions, which are recorded in the database logs, to a restored database backup image or table space backup image. Working with the same Data Studio interface previously indicated, the *Roll Forward* selection opens a new windows with the following tab:

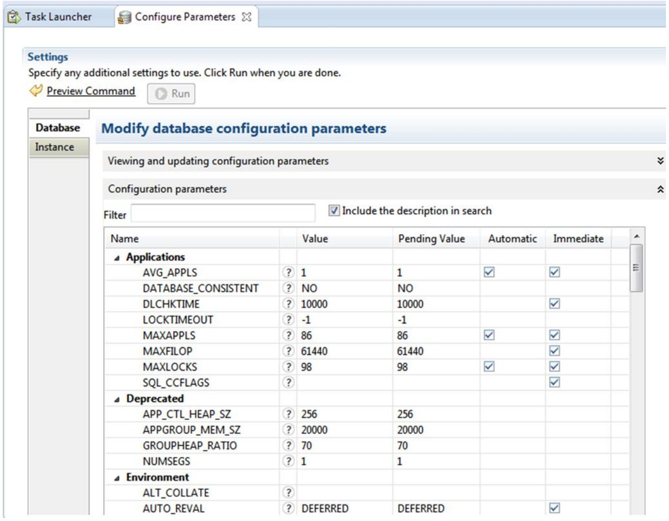
- Roll-forward Scope tab: to choose if the roll forward operation should be applied to the complete database or just to a particular table space. If roll forward has been already used to restore an entire database, the option to select table spaces will not been available at a later stage.
- Roll-forward Final State tab: to decide if to complete the roll forward procedure or to leave the database in roll forward pending state, that can be solved in a further moment.

Once the options are set, by clicking the Run button the roll forwarding operation is activated.

3.8.3 - More maintenance operations

Data Studio provides additional options to perform more database maintenance tasks:

- Recovering a database: as stated before, Recovery is a combination of a restoring followed by a roll forward operations. The DB2 system will restore an appropriate backup image depending on a specified time and the history file. This operation is available selecting *Back Up and Restore->Recover* in the database menu.
- Completing a roll forward recovery: if a user earlier decided not to complete roll forwarding in one step, this option completes a roll forward recovery by stopping the roll forward of log records, rolling back incomplete transactions, and turning off the roll forward pending state. As a consequence, users regain access to the rolled forwarded database or table spaces. This operation is available selecting *Back Up and Restore -> Complete Roll Forward* in the database menu.
- Cancelling a roll forward recovery: to cancel a roll forward operation that is in progress, but cannot be completed. This operation is available by selecting *Back Up and Restore->Cancel Roll Forward* in the database menu.
- Configuring automatic maintenance: the database manager provides automatic keep up capabilities for performing database backups, keeping statistics current, and reorganizing tables and indexes as necessary. They can be configured selecting the option *Set Up and Configure->Configure Automatic Maintenance* in the database menu.
- Configuring database or instance parameters: many parameters at the database level, as well as at the database manager level can be configured for specific behavior and to tune both (Figure 3.22). This can be achieved selecting *Set Up and Configure ->Configure* in the database menu.



Settings

Specify any additional settings to use. Click Run when you are done.

Modify database configuration parameters

Viewing and updating configuration parameters

Configuration parameters

Filter ☑ Include the description in search

Name		Value	Pending Value	Automatic	Immediate
Applications					
AVG_APPLS	?	1	1	☑	☑
DATABASE_CONSISTENT	?	NO	NO		
DLCHKTIME	?	10000	10000		☑
LOCKTIMEOUT	?	-1	-1		
MAXAPPLS	?	86	86	☑	☑
MAXFILOP	?	61440	61440		☑
MAXLOCKS	?	98	98	☑	☑
SQL_CCFLAGS	?				☑
Deprecated					
APP_CTL_HEAP_SZ	?	256	256		
APPGROUP_MEM_SZ	?	20000	20000		
GROUPHEAP_RATIO	?	70	70		
NUMSEGS	?	1	1		
Environment					
ALT_COLLATE	?				
AUTO_REVAL	?	DEFERRED	DEFERRED		☑

Figure 3.22 – Configuring database or instance parameters

3.8.4 - Moving data

In general, moving data (Export/Unload, Import/Load) is possible for some database objects (Table, View, Alias, MQT) and for some profiles as the Connection Profile. In particular the Table object is suited of the most completed moving functionalities, among the other objects. See paragraph 3.5 for a detailed description of the available utilities.

3.8.5 - Managing DB2 performance

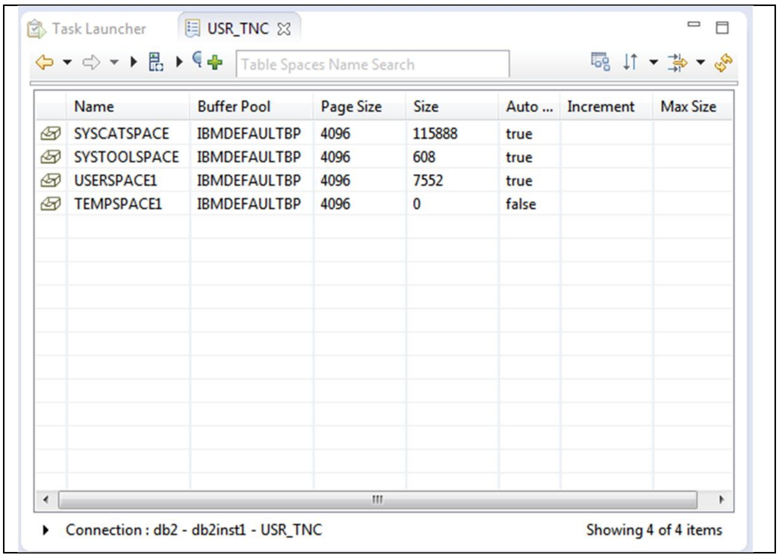
Data are stored in a DB2 data server both in the file system and in raw devices. The storage area characteristics are defined using the DB2 table space object, that contains tables, indexes, aliases. The dimension of memory areas, where data fetched from tables are stored when query runs, is another important issue. In this case DB2 provides buffer pool objects to achieve a better governance of the problem. Because design and tuning of table spaces and buffer pools can have a profound impact on how the DB2 server performs, Data Studio allows database administrator to manage the characteristics of the table storage and have more control over table memory usage.

3.8.5.1 - DB2 table space object

A table space object (see chapter 1.) is stored into database partition groups. It is helpful to organize other database objects into logical storage group, linked to physical system area, where data are placed. It consists of containers, which could be: an operating system file, a directory, or a raw device. Nowadays raw devices, although supported, are not widely used in a typical database, due to advances in disk and file system performance. Table spaces influence positively some database management aspects like: a) recoverability: a complete database backup and restore is made more convenient using a single command; b) automatic storage management: database manager creates and extends containers depending on the needs; c) memory utilization: a single buffer pool can manage multiple table spaces, so that temporary table spaces, assigned to a buffer pool, increase the performance of database activities.

When a new database is generated, the database manager creates some default table spaces for it, to store user and temporary data (Figure 3.23). They are classified as:

- Catalog: this table space holds system catalog tables for the database, is named as SYSCATSPACE and cannot be dropped.
- User: this table space contains user-defined tables. In a database one default user table space, named as USERSPACE1 exists. It is assigned to a table by the database manager when it is created and a specific user-defined table space has not been already defined
- Temporary: this table space accommodates temporary table data and includes both system temporary table space and user temporary table space. The former holds temporary data required by the database manager, while performing operation such as sorts or joins. The last is created at the time of the database generation and holds temporary data from tables. A database must have at least one system temporary table space: its name is TEMPSPACE1.
- SYSTOOLSPACE: this table space is automatically created for storing configuration and working data useful for automated maintenance.



Name	Buffer Pool	Page Size	Size	Auto ...	Increment	Max Size
SYSCATSPACE	IBMDEFAULTBP	4096	115888	true		
SYSTOOLSPACE	IBMDEFAULTBP	4096	608	true		
USERSPACE1	IBMDEFAULTBP	4096	7552	true		
TEMPSPACE1	IBMDEFAULTBP	4096	0	false		

Connection : db2 - db2inst1 - USR_TNC

Showing 4 of 4 items

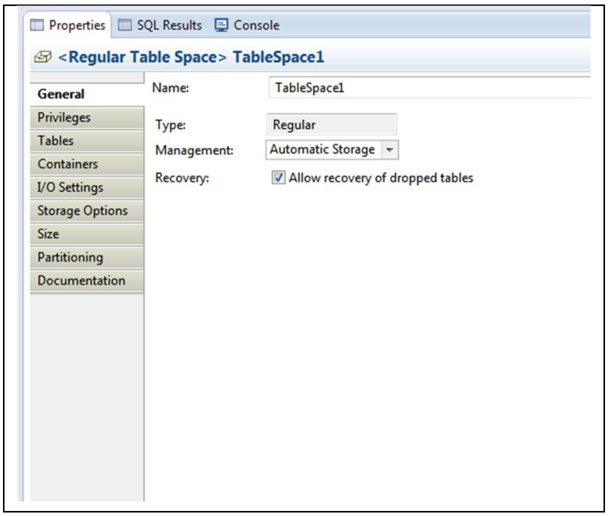
Figure 3.23 – DB2 default Tablespace

Table spaces can be set up in the following three different ways according to their spaces management:

- System Managed Space (SMS): the operating system allocates and manages on demand the space where the object is stored. This model consists of files representing database objects included into a container, associated with an absolute or relative directory name. SMS type is used for catalog and temporary table spaces.
- Database Managed Space (DMS): the database server controls the storage space, which is pre-allocated according to a container definition specified during the table space creation. DMS type containers use both files and raw devices: they are utilized for system table space and temporary table space.
- Automatic Storage Tablespace: alternative to the previously described types. In this model the database server creates and extends containers depending on data really present into the database, without a

container definition. A user must only determine the path where the containers should be created and their maximum size.

Another table space classification is related to the type of data it stores: regular, large, or temporary.
To create a table space in the Administration Explorer, the Table Spaces folder must be right-clicked and the *Create Regular Table Space* link selected. A new table space appears in the Object list editor. Its parameters can be defined in the Properties view (Figure 3.24). In the General tab, besides the table space name, the type of space management can be set. In the Containers tab a container can be created and its type assigned. Containers where the space is managed by the system are only of type Directory. Containers where the space is controlled by the database manager can also be of type File. Containers, where the space is managed with the Automatic Storage model, must not be determined. In this case the Initial size, Increase size, and the Maximum size parameters must be set under the Size tab, The Initial size is allocated at the time of creation. Whenever more storage memory is required the system will increase the container space by the increase size amount until the maximum size limit is reached.



<Regular Table Space> TableSpace1

Name: TableSpace1

Type: Regular

Management: Automatic Storage

Recovery: ☑ Allow recovery of dropped tables

Figure 3.24 –Table Space Properties view

Once defined the table space, the changes to be deployed are displayed in a new window. Selecting the Run button they are committed to the database.
The new table space can be associated with the database objects by right-clicking a specific object icon and selecting the Alter link. In this way the DB2 data server should exploit this table space for the physical storage of the related object.
A table space object can be deleted, using the DROP command. It is important to remember that, when a table space is dropped, all associated tables and data are also cancelled.
If impacted objects are related to the table space, Data Studio asks user if they must be dropped at the same time. This operation could fail answering No. In this case the impacted objects should be first altered to disassociate them from the table space, to be consequently able to delete them.

3.8.5.2 - DB2 buffer pool object

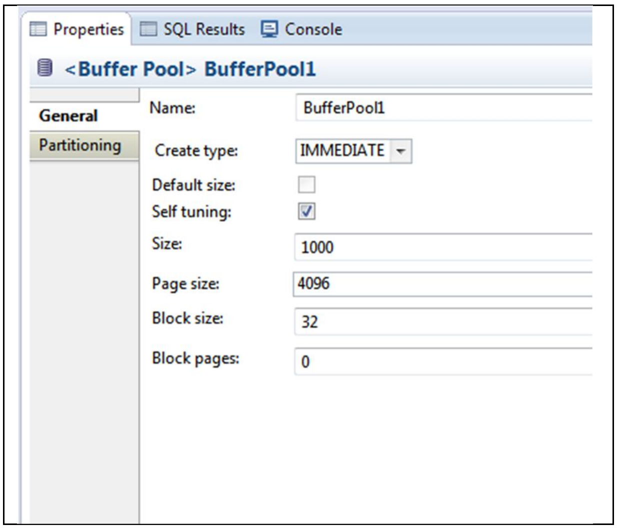
The buffer pool is a portion of a main memory space, which is allocated by the database manager to cache table and index data from disk. All databases have their own buffer pools. A default buffer pool is generated when a new database is created and is named IBMDEFAULTBP. It is used for all objects, unless they are explicitly assigned to another buffer pool. According to the user requirements, it is possible to create a number of buffer pools, in which the database manager places the table row data as a page. This page remains in the buffer pool until the database is shutdown or until it gets full, so that the old data is wiped out and the space is overwritten with new data. The updated pages in the buffer pool, not yet written onto the disk, are called “Dirty” pages. After page updating into the database is completed, the buffer pool is ready to take another data.
This area is also the repository where the changes made by an application to a database object are performed, before being really stored into the database. Great database performance improvement is also achieved gathering in a buffer pool, for faster access, collected data from table spaces to be used by query operations, instead of

fetching them directly from the disk. To fetch data from a table space, a buffer pool with the same page size must exist.

To create a buffer pool in the Administration Explorer, the Buffer Pools folder must be right-clicked and the *Create Buffer Pool* link selected. A new buffer pool appears in the Object list editor. Its parameters can be defined in the Properties view (Figure 3.25). In the General tab the buffer pool name, total size, page size, and other basic attributes can be set. A user can also decide to create this buffer pool immediately, by selecting in the Create type field the choice IMMEDIATE. Otherwise, by selecting the DEFERRED option, the script will be run the next time that the database starts. Buffer pool deployment steps are similar to those of the table space deployment procedure.

To be utilized a buffer pool must be assigned to an altered table space. In this way DB2 data server knows that this buffer pool fetches the data from this specific table space.

Buffer pools can be altered and dropped in Data Studio following the same procedures described for other objects.



Name: BufferPool1
Create type: IMMEDIATE
Default size: □
Self tuning: ☑
Size: 1000
Page size: 4096
Block size: 32
Block pages: 0

Figure 3.25 –Buffer Pool Properties view

3.8.6 – DB2 statistics utility

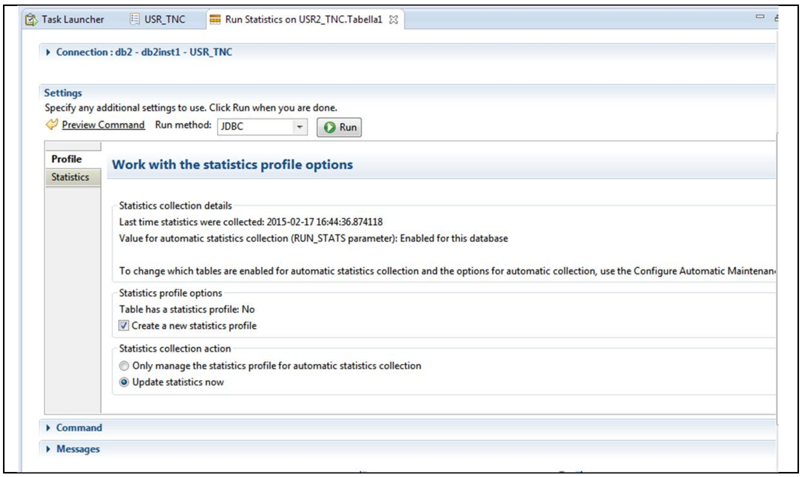
DB2 uses the statistics information in the catalog table to derive the best execution plan. Keeping regularly updated statistics about the characteristics of a table and/or associated indexes or views is important for optimal query performance. These characteristics include the number of records, the number of pages and the average record length. Normally a database administrator should call the SQL RUNSTATS utility to collect new statistics when: 1) a table has had many updates or after a table reorganization and 2) rows returned by a view are substantially affected by underlying table changes. The DB2 optimizer utility uses these statistics to compile query and to create an access plan, describing the sequence of steps, based on the best access path, to run the query and fetch the returned data. The DB2 Optimizer is a cost-based optimizer: it calculates the cost of multiple access paths (in a hardware depending time unit) and then chooses the cheapest one. The calculated cost relies on factors such as:

- configuration parameters on a system which controls how resources are used.
- database design – table spaces, indexes, MQTs.
- statistical information about the data.
- DFT_QUERYOPT database configuration parameter or CURRENT QUERY OPTIMIZATION special register.

The DB2 Optimizer utilizes a DML SQL statement to carry out its task. Changing data transactions (update, insert, and delete) grows or shrinks the database size and often modifies its distribution. Consequently, if the optimizer creates a less optimal access plan based on outdated statistics, that no longer reflect reality, operation performance may be negatively affected.

To collect statistics a table must be highlighted from the Object list editor. By right-clicking the table and selecting the *Manage->Run Statistics* option, a new editor view opens, where a user can specify how statistics should be run on the database (Figure 3.26).

The RUNSTATS utility allows to register, for further use, a statistics profile. It specifies the type of statistics that should be collected for a particular table, such as table statistics, index statistics, or distribution statistics. In the Profile tab, the Update statistics now option lets to update the statistics immediately. The Statistics tab helps to specify the type of statistics that a user wants to gather (on all columns, data distribution statistics, basic statistics on indexes). Finally the Run button must be activated to execute the utility.


Task Launcher
USR_TNC
Run Statistics on USR2_TNC.Tabella1
Connection : db2 - db2inst1 - USR_TNC
Preview Command
Run
Profile
Statistics
Command
Messages

Settings

Specify any additional settings to use. Click Run when you are done.

Run method: JDBC

Work with the statistics profile options

Statistics collection details

Last time statistics were collected: 2015-02-17 16:44:36.874118

Value for automatic statistics collection (RUN_STATS parameter): Enabled for this database

To change which tables are enabled for automatic statistics collection and the options for automatic collection, use the Configure Automatic Maintenan

Statistics profile options

Table has a statistics profile: No

☑ Create a new statistics profile

Statistics collection action

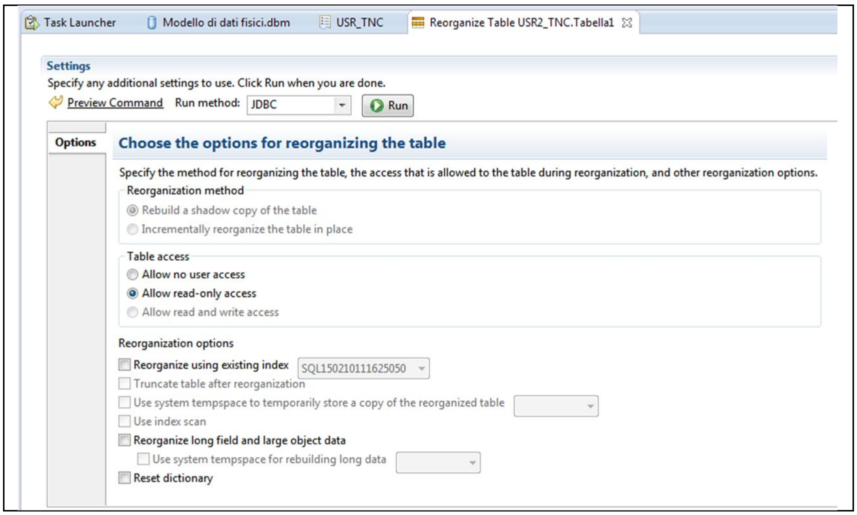
○ Only manage the statistics profile for automatic statistics collection

● Update statistics now

Figure 3.26 – Run Statistics Properties view

3.8.7 – DB2 data reorganization

Data rows inserted into a new table are usually stored in a sequential manner. Over time, however, after frequent changes to table data or when a large amount of data is deleted, logically sequential data might reside on sparse data pages. Normally, the space filled by rows is not freed and may be reused by other added rows. Data fragmentation forces the database manager to perform additional read operations to access data. A good practice to maintain performance on a database is the regular execution of the tables and indexes reorganization task. This operation defragments data, improves the I/O cost and, in some instances, reduces memory usage.
To reorganize data, by right clicking a table in the Object list editor and by selecting the *Manage->Reorg Table* combination, an editor opens to let a user configure the possible reorganization options through the Option tab (Figure 3.27). Finally the Run button must be activated to execute the task.


Task Launcher
Modello di dati fisici.dbm
USR_TNC
Reorganize Table USR2_TNC.Tabella1
Preview Command
Run
Options

Settings

Specify any additional settings to use. Click Run when you are done.

Run method: JDBC

Choose the options for reorganizing the table

Specify the method for reorganizing the table, the access that is allowed to the table during reorganization, and other reorganization options.

Reorganization method

● Rebuild a shadow copy of the table

○ Incrementally reorganize the table in place

Table access

○ Allow no user access

● Allow read-only access

○ Allow read and write access

Reorganization options

□ Reorganize using existing index SQL150210111625050

□ Truncate table after reorganization

□ Use system tempspace to temporarily store a copy of the reorganized table

□ Use index scan

□ Reorganize long field and large object data

□ Use system tempspace for rebuilding long data

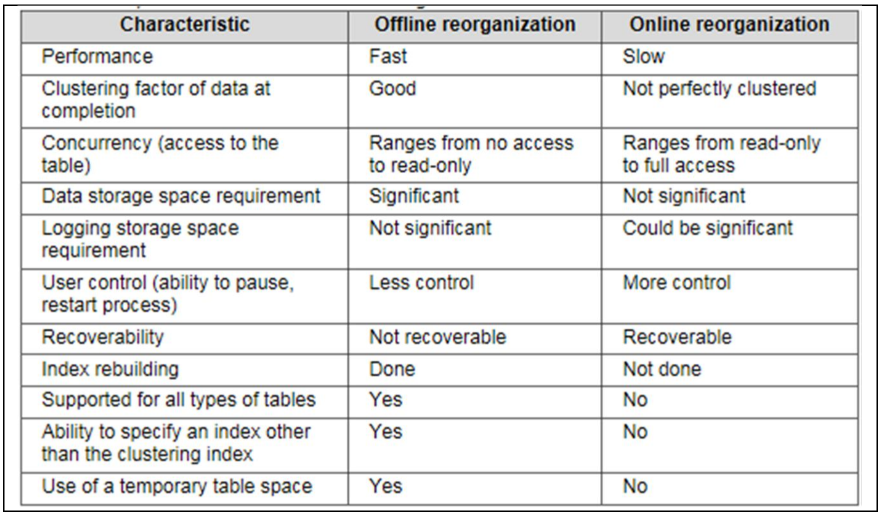
□ Reset dictionary

Figure 3.27 - Options to reorganize a table

There are two approaches to table reorganization and each one has its advantages and drawbacks (Figure 3.28):

- the offline classic reorganization (called Rebuild a shadow copy of the table in the Options tab). The benefits of this method are: a) the fastest table reorganization operations; b) perfectly clustered tables and automatically rebuilt indexes upon completion; c) the use of a temporary table space for building a shadow copy. This procedure decreases the space requirements for the table space that contains the target table or index. Negativities, instead, are principally due to: a) limited table access: read access only during the sort and build phase of a reorganization operation; b) less control over the reorganization process: an offline operation cannot be paused and restarted.

- the online inplace reorganization (called Incrementally reorganize the table in place in the Options tab). It positively outlines: a) full table access, except during the truncation phase of a reorganization operation; b) more control over the process (accept pause, resume, stop command), running asynchronously in the background; c) a recoverable process in the event of a failure and a reduced requirement for working storage, because a table is processed incrementally. However the method manifests these problems: a) imperfect data or index clustering, depending on the type of transactions that access the table during this operation; b) poorer performance than an offline reorganization operation; c) potentially high logging requirements, depending on the number of rows being moved and the number and size of indexes defined on the table.



Characteristic	Offline reorganization	Online reorganization
Performance	Fast	Slow
Clustering factor of data at completion	Good	Not perfectly clustered
Concurrency (access to the table)	Ranges from no access to read-only	Ranges from read-only to full access
Data storage space requirement	Significant	Not significant
Logging storage space requirement	Not significant	Could be significant
User control (ability to pause, restart process)	Less control	More control
Recoverability	Not recoverable	Recoverable
Index rebuilding	Done	Not done
Supported for all types of tables	Yes	No
Ability to specify an index other than the clustering index	Yes	No
Use of a temporary table space	Yes	No

3.28 - Comparing Db2 reorganization methods