Workbook

Data Warehouse Methods
(SE – 403)

Name _____

Roll No _____

Batch _____

Year _____

Department _____

Department of Software Engineering
NED University of Engineering & Technology

Workbook

Data Warehouse Methods
(SE – 403)

Prepared by

Dr. Raheela Asif
Associate Professor

Approved by

Chairman
Department of Software Engineering

Department of Software Engineering
NED University of Engineering & Technology

Table of Contents

Lab # 1

Object:
Introduction to INFORMATICA.

Theory:
Introduction
INFORMATICA is a Software development company, which offers data integration products. It offers products for ETL, Data Masking, Data Quality, Data Replica, Data Virtualization, Master Data Management, etc.

Informatica PowerCenter ETL/Data Integration tool is the most widely used tool and in the common term when we say Informatica, it refers to the Informatica PowerCenter tool for ETL.

Informatica PowerCenter is used for Data integration. It offers the capability to connect & fetch data from different heterogeneous source and processing of data.

Informatica is used to extracting required data form operation all systems and transforms the same data on its server and load it to the data warehouse.

Informatica is also introduced as a data integration tool. This tool is based on the ETL architecture. It provides data integration software and services for different industries, businesses, government organizations, as well as telecommunication, health care, insurance, and financial services.

It has a unique property to connect, process, and fetch the data from a different type of mixed sources. For example, an SQL Server Database and Oracle Database both can be connected and can integrate the data into a third system.

Uses of Informatica
- An organization migrating from existing legacy system like mainframe to a new database system. So the migration of its existing data into a system can be performed.

- Enterprises setting up their Data Warehouse would require an ETL tool to move data from the Production system to Warehouse.

- Integration of data from various heterogeneous systems like multiple databases and file-based systems can be done using Informatica.

- Informatica can be used as a data cleansing tool and to perform some operations on the data at the backend in a data system.

- It provides a broad set of features such as integration of the data from multiple unstructured, semi-structured or structured systems, operations at row level on data, and scheduling operation of the data operation.

- It also supports the features of metadata, so it keeps preserved the information of the process and data operations.

Exercise:

1. Summarize in your own words why we need Informatica.

Lab # 2

Object:
INFORMATICA Architecture.

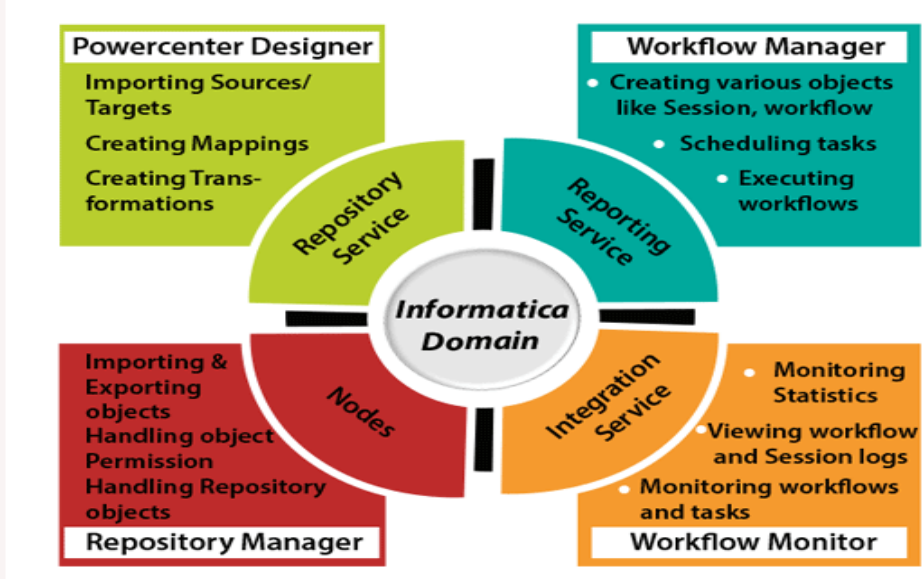
Theory:
Informatica architecture is service-oriented architecture (SOA). A service-oriented architecture is defined as a group of services that communicate with each other. It means a simple data transfer during this communication, or it can be two or more services that coordinate the same activity.

The Informatica development depends upon the component-based development techniques. This technique uses the predefined components and functional units with their functionalities to get the result.

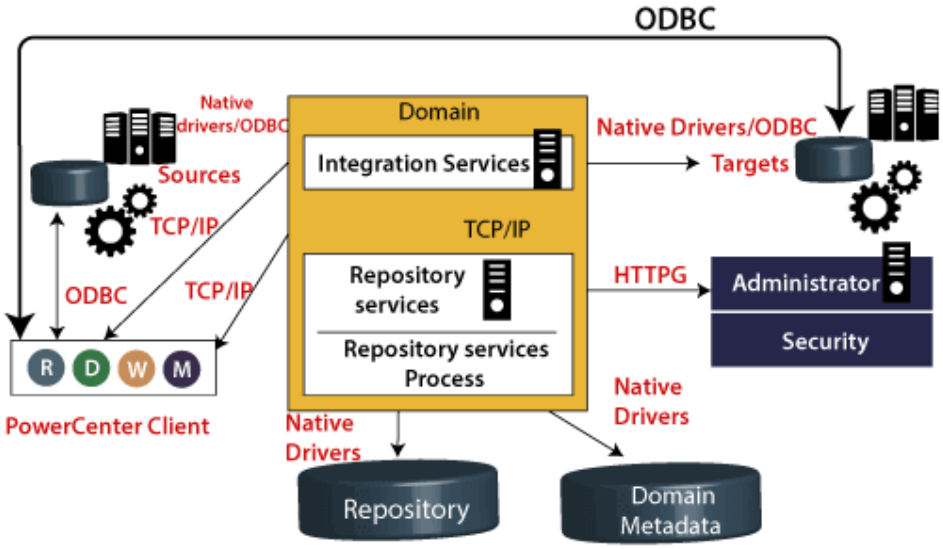
PowerCenter is based on the component-based development methodologies. To build a dataflow from the source to target, it used different components, and this process is called transformation.

Informatica ETL tool has the below services and components, such as:

- **Repository Service:** It is responsible for maintaining Informatica metadata and provides access to the same to other services.
- **Integration Service:** This service helps in the movement of data from sources to the targets.
- **Reporting Service:** This service generates the reports.
- **Nodes:** This is a computing platform to execute the above services.
- **Informatica Designer:** It creates the mappings between source and target.
- **Workflow Manager:** It is used to create workflows or other tasks and their execution.
- **Workflow Monitor:** It is used to monitor the execution of workflows.
- **Repository Manager:** It is used to manage the objects in the repository.



Informatica Domain



- The Informatica domain is the fundamental administrative unit.
- The Informatica domain consists of nodes and services. These nodes and services are categorized into folders or sub-folders based on administration requirements and design architecture.
- The Console web page of the Informatica administrator creates a domain that looks like a folder. Inside this folder, we can create a node with the services.
- In the Informatica domain, a node is a logical representation of the machine. All the services and processes run inside the domain in the Informatica. Multiple nodes can be present in a single domain. A gateway node receives the request from the clients and guides them to their respective services.
- The domain provides two types of services, such as:
  - **Service Manager:** It manages domain operations such as logging, authentication, and authorization. It runs the application services on the nodes and leads users and groups.
  - **Application Services:** It represents the server-specific services such as repository services, reporting services, and integration services. The application service can run on different nodes based on configuration.

Node

Node is a sensible study of a machine in a domain, and a domain has different hubs. To run the application administrations, we can design the hubs, such as mix administration.

PowerCenter Repository

The PowerCenter repository is a relational database such as SQL Server, Oracle, and Sybase. And these databases are maintained by the repository services. The database tables store the metadata. Informatica client tools are the three types, such as:

- Informatica designer
- Informatica workflow manager
- Informatica workflow monitor

Informatica provides repository services, and it is used to manage the repository. The repository services exclusively handle one request for one repository. But we can execute it on multiple nodes for better performance.

We can maintain the different versions of the same objects because of its version control mechanism. And also ignore multiple users that modifying the same object at the same time.

The objects created in the repository are having this three-state, such as:

- **Valid:** Valid objects have the correct syntax, according to the Informatica. And used for the execution of the workflow.
- **Invalid:** Invalid objects do not follow the standard or rules. These objects checked the syntax, and properties are valid or not during the saving of the object in Informatica.
- **Impacted:** The child objects of the affected object are invalid.

PowerCenter Repository Service

PowerCenter repository service is a different multi-strung process. It allows the customer to change the metadata in the store. It accepts demands from the considerate benefit for metadata to run work processes.

And the repository service maintains the associations from PowerCenter customers to the PowerCenter vault. It inserts the metadata inside the archive and keeps it refreshed. It is able to keep up consistency inside the archive metadata.

Domain configuration

In the Informatica ETL tool, the domain is the necessary fundamental administrative control. It is an apparent entity that provides other different services such as repository service, integration service, and various nodes. The Informatica admin console is used for the domain configuration. And the console is launched with the help of the web browsers.

PowerCenter Client and Server Connectivity

PowerCenter client tools are installed on the client-side machines. These tools are the development tools such as workflow manager, PowerCenter designer, repository manager, and workflow monitor.

Informatica repository contains all the created mapping and objects in these client tools, which resides on the Informatica server. That's why client tools must have network connectivity with the server.

Also, PowerCenter client connects to the sources and targets to import the metadata and structure definitions. Thus, it also maintains the connectivity to the source or target systems.

- PowerCenter client uses the TCP/IP protocols for connectivity with the integration service and repository service.
- And PowerCenter client uses the ODBC drivers for the connectivity with the source or targets.

Repository Service

The repository service is a multithreading process. It maintains the connection between the PowerCenter clients and the PowerCenter repository.

The repository service can fetch, insert, and update the metadata inside the repository. And it also maintains the consistency inside the repository metadata.

Integration Service

The integration service is used as an execution engine in the Informatica. It helps in executing the tasks which are created in the Informatica. Integration service works in the following manner, such as:

- A user performs a workflow.
- The Informatica instructs the integration service to execute the workflow.
- Then the integration service reads workflow details from the repository.
- The integration service starts the execution of the tasks inside the workflow.
- After the execution, the task status is updated, for example, Succeeded, Failed, or Aborted.
- Then it grants the session log and workflow log.
- This service loads the data into the target systems.
- Integration service combines data from different sources.

Exercise:

1.Explain in your own words what is Informatica Domain.

2.What are Valid, Invalid and Impacted Data Objects

Lab # 3

Object:
Mapping in INFORMATICA.

Theory:
Mapping is a collection of source and target objects which is tied up together through a set of transformations. These transformations are formed with a set of rules that define how the data is loaded into the targets and flow of the data.

Mapping in Informatica includes the following set of objects, such as:

Source definition: The source definition defines the structure and characteristics of the source, such as basic data types, type of the data source, and more.

Transformation: It defines how the source data is changed, and various functions can be applied during this process.

Target Definition: The target definition defines where the data will be loaded finally.

Links: Link is used to connecting the source definition with target tables and different transformations. And it shows the flow of data between the source and target.

Why Mapping is required
In Informatica, Mapping is an object which can define the process of modification of the source data before it reaches the target object.

For example: Suppose an employee name as "Edward Cullen" in the source system and the target system. Now we need to have an employee name in the "Edward Cullen" format. At the mapping level, these kinds of operations are designed.

Mapping can define the data transformation details and source or target object characteristics because it is a primary object in the Informatica.

Mappings define the data transformation for each row at the individual column levels. And we can hold multiple sources and targets in a single mapping.

Components of Mapping
Some essential elements used in mapping, such as:
- Source tables
- Mapping parameters and variables
- Target objects
- Mapping transformations

A mapping contains **sources, targets, mapping parameters, variables, multiple changes, mapplets,** and **user-defined functions**.

Mapping Source: Mapping sources are those objects whose allow fetching the source data. It can be a flat file, database table, COBOL file source, or XML source.

Mapping target: The mapping target is the destination objects where the final data is loaded. A mapping target can be a relational table of a database, XML file, or a flat-file. Sources and targets must be present in any mapping with their different data types.

Mapping Parameters and Variables: It is an optional user-defined data types. The Mapping parameters and variables are used to create temporary variable objects. It helps us to define and store temporary values during the processing of mapping data. This user-defined data type is designed for mapping.

Mapplets: The mapplets are objects which consist of a set of source, transformation, or targets. With the help of Mapplets, we can reuse the existing functionality of a set of changes.

What is Stage Mapping?

In a stage mapping, we create the replica of the source table.

For Example, if we have a "Student" table, and we want to create an identical table, "Student_stage" in ETL schema.

A local stage table provides some advantages, such as production downtime, so it will not affect the ETL system because we have our own "Student_stage" table instead of referring to the "Student" table. There can be some other operations and processes which can affect the performance. The ETL processes will only access it when we have a replica staging table. Then it provides advantages with better performance.

In Stage Mappings,

- Source and Target tables are having identical structures.
- In the staging table, data is a replica of source table data.
- In the source table, data is a subset of source data.

For example, if the source table contains student details of rollno 1, 2, 3, and 10. The stage table is a kind of table which has student records of rollno 1 & 3 only.

In the data warehouse, we need to create stage tables to make the process of data transformation efficiently, to minimize the dependency of ETL or Data Warehouse from the real-time operating system. It could happen when we are fetching the relevant data only.

Mapping Parameters and Variables

Informatica has a way of defining parameters and variables as other programming languages. But Informatica is not a code-based language just like other programming languages.

In Informatica, we need to follow the predefined syntax and navigation to create parameters and variables.

Some fundamental differences between the mapping parameters and mapping variables, such as:

Mapping Parameters	Mapping Variables
Mapping parameters are those data types whose value assigned at once and remains constant throughout the execution of the mapping	Mapping variables are the objects referenced throughout the execution of the mapping, and their values can be reassigned
For example: If we have created a mapping parameter rollno=2, then the value 2 will be constant throughout the whole execution of mapping. The referenced parameter will always return value 2 for that instance of mapping run. And the value of a parameter can be redefined for a new mapping instance	**For example**, A mapping variable of total_marks is used in mapping, and its value will be updated on the basis of marks.

Exercise:

1. Explain in your own words what is Stage Mapping. Also give an example to elaborate your answer.

2. Create a Stage Mapping in Informatica and attach screen shot.

Lab # 4

Object:
INFORMATICA as ETL Tool.

Theory:
What is ETL?
ETL is a process that extracts the data from different source systems, then transforms the data (like applying calculations, concatenations, etc.) and finally loads the data into the Data Warehouse system. Full form of ETL is Extract, Transform and Load.

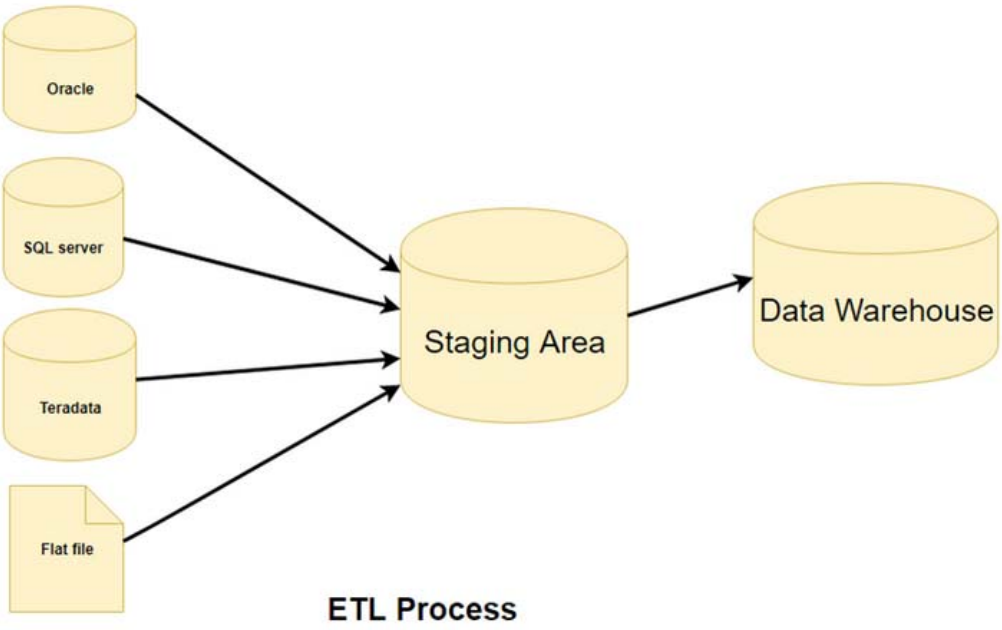
It's tempting to think a creating a Data warehouse is simply extracting data from multiple sources and loading into database of a Data warehouse. This is far from the truth and requires a complex ETL process. The ETL process requires active inputs from various stakeholders including developers, analysts, testers, top executives and is technically challenging.

In order to maintain its value as a tool for decision-makers, Data warehouse system needs to change with business changes. ETL is a recurring activity (daily, weekly, monthly) of a Data warehouse system and needs to be agile, automated, and well documented.

Why do you need ETL?
There are many reasons for adopting ETL in the organization:
- It helps companies to analyze their business data for taking critical business decisions.
- Transactional databases cannot answer complex business questions that can be answered by ETL.
- A Data Warehouse provides a common data repository
- ETL provides a method of moving the data from various sources into a data warehouse.
- As data sources change, the Data Warehouse will automatically update.
- Well-designed and documented ETL system is almost essential to the success of a Data Warehouse project.
- Allow verification of data transformation, aggregation and calculations rules.
- ETL process allows sample data comparison between the source and the target system.
- ETL process can perform complex transformations and requires the extra area to store the data.
- ETL helps to Migrate data into a Data Warehouse. Convert to the various formats and types to adhere to one consistent system.
- ETL is a predefined process for accessing and manipulating source data into the target database.
- ETL offers deep historical context for the business.
- It helps to improve productivity because it codifies and reuses without a need for technical skills.

ETL Process in Data Warehouses

ETL is a 3-step process



ETL Process

Step1: Extraction

In this step, data is extracted from the source system into the staging area. Transformations if any are done in staging area so that performance of source system in not degraded. Also, if corrupted data is copied directly from the source into Data warehouse database, rollback will be a challenge. Staging area gives an opportunity to validate extracted data before it moves into the Data warehouse.

Data warehouse needs to integrate systems that have different

DBMS, Hardware, Operating Systems and Communication Protocols. Sources could include legacy applications like Mainframes, customized applications, Point of contact devices like ATM, Call switches, text files, spreadsheets, ERP, data from vendors, partners amongst others.

Hence one needs a logical data map before data is extracted and loaded physically. This data map describes the relationship between sources and target data.

Three Data Extraction methods

- Full Extraction
- Partial Extraction- without update notification.
- Partial Extraction- with update notification

Irrespective of the method used, extraction should not affect performance and response time of the source systems. These source systems are live production databases. Any slow down or locking could effect company's bottom line.

Some validations are done during Extraction

- Reconcile records with the source data
- Make sure that no spam/unwanted data loaded

- Data type check
- Remove all types of duplicate/fragmented data
- Check whether all the keys are in place or not

Step2: Transformation

Data extracted from source server is raw and not usable in its original form. Therefore it needs to be cleansed, mapped and transformed. In fact, this is the key step where ETL process adds value and changes data such that insightful BI reports can be generated.

In this step, you apply a set of functions on extracted data. Data that does not require any transformation is called as **direct move** or **pass through data**.

In transformation step, you can perform customized operations on data. For instance, if the user wants sum-of-sales revenue which is not in the database. Or if the first name and the last name in a table is in different columns. It is possible to concatenate them before loading.

Data Integrity Problems:

- Different spelling of the same person like Jon, John, etc.
- There are multiple ways to denote company name like Google, Google Inc.
- Use of different names like Cleaveland, Cleveland.
- There may be a case that different account numbers are generated by various applications for the same customer.
- In some data required files remains blank
- Invalid product collected at POS as manual entry can lead to mistakes.

Validations are done during this stage

- Filtering – Select only certain columns to load
- Using rules and lookup tables for Data standardization
- Character Set Conversion and encoding handling
- Conversion of Units of Measurements like Date Time Conversion, currency conversions, numerical conversions, etc.
- Data threshold validation check. For example, age cannot be more than two digits.
- Data flow validation from the staging area to the intermediate tables.
- Required fields should not be left blank.
- Cleaning (for example, mapping NULL to 0 or Gender Male to "M" and Female to "F" etc.)
- Split a column into multiples and merging multiple columns into a single column.
- Transposing rows and columns,
- Use lookups to merge data
- Using any complex data validation (e.g., if the first two columns in a row are empty then it automatically reject the row from processing)

Step3: Loading

Loading data into the target data warehouse database is the last step of the ETL process. In a typical Data warehouse, huge volume of data needs to be loaded in a relatively short period (nights). Hence, load process should be optimized for performance.

In case of load failure, recover mechanisms should be configured to restart from the point of failure without data integrity loss. Data Warehouse admins need to monitor, resume, cancel loads as per prevailing server performance.

Types of Loading

- **Initial Load** — populating all the Data Warehouse tables
- **Incremental Load** — applying ongoing changes as when needed periodically.
- **Full Refresh** —erasing the contents of one or more tables and reloading with fresh data.

Load verification

- Ensure that the key field data is neither missing nor null.
- Test modeling views based on the target tables.
- Check that combined values and calculated measures.
- Data checks in dimension table as well as history table.
- Check the BI reports on the loaded fact and dimension table.

Best practices ETL process

Never try to cleanse all the data

Every organization would like to have all the data clean, but most of them are not ready to pay to wait or not ready to wait. To clean it all would simply take too long, so it is better not to try to cleanse all the data.

Never cleanse anything

Always plan to clean something because the biggest reason for building the Data Warehouse is to offer cleaner and more reliable data.

Determine the cost of cleansing the data

Before cleansing all the dirty data, it is important for you to determine the cleansing cost for every dirty data element.

To speed up query processing, have auxiliary views and indexes

To reduce storage costs, store summarized data into disk tapes. Also, the trade-off between the volume of data to be stored and its detailed usage is required. Trade-off at the level of granularity of data to decrease the storage costs.
Informatica ETL

Features of Informatica ETL Tool

Informatica ETL is used to data extraction, and it is based on the data warehouse concept, where the data is extracted from multiples different databases.Some essential features of the ETL tool, such as:

Parallel Processing

ETL is implemented by using a concept of Parallel Processing. Parallel Processing is executed on multiple processes that running simultaneously. ETL is working on three types of parallelism, such as:

- By splitting a single file into smaller data files.
- The pipeline allows running several components simultaneously on the same data.

- A component is the executables processes involved for running simultaneously on different data to do the same job.

Data Reuse, Data Re-Run, and Data Recovery

Each data row is provided with a row_id, and a piece of the process is supplied with a run_id so that one can track the data by these ids. To complete certain phases of the process as we create checkpoints. These checkpoints tell the need to re-run the query for task completion.

Visual ETL

The PowerCenter and Metadata Messenger are advanced ETL tools. These tools help to make faster, automated, and impactful structured data according to the business requirements.

We can create a database and metadata modules with a drag and drop mechanism as a solution. It can automatically configure, connect, extract, transfer, and loads the data into the target system.

Characteristics of ETL Tool

Some attributes of the ETL tool are as follows:

- It should increase data connectivity and scalability.
- It should be capable of connecting multiple relational databases.
- It should support CSV extension data files then the end-users can import these files easily or without any coding.
- It should have a user-friendly GUI so that the end-users easily integrate the data with the visual mapper.
- It should allow the end-user to customize the data modules according to the business requirements.

Exercise:

1. Summarize in your own words what is ETL and why it is required.

Lab # 5

Object:
INFORMATICA Transformations.

Theory:
Informatica Transformations are repository objects which can create, read, modifies, or passes data to the defined target structures such as tables, files, or any other targets.
In Informatica, the purpose of transformation is to modify the source data according to the requirement of the target system. It also ensures the quality of the data being loaded into the target.

A Transformation is used to represent a set of rules, which define the data flow and how the data is loaded into the targets.

Informatica provides multiple transformations to perform specific functionalities. In transformations, to passing the data, we need to connect the ports to it, and through the output ports, it returns the output.

Classification of Transformation
Transformation is classified into two categories-the first one based on connectivity and second based on the change in several rows. First, we will look at the transformation based on connectivity.

Transformation based on connectivity
- Connected Transformations
- Unconnected Transformations

In Informatica, one transformation is connected to other transformations during mappings are called **connected transformations**.

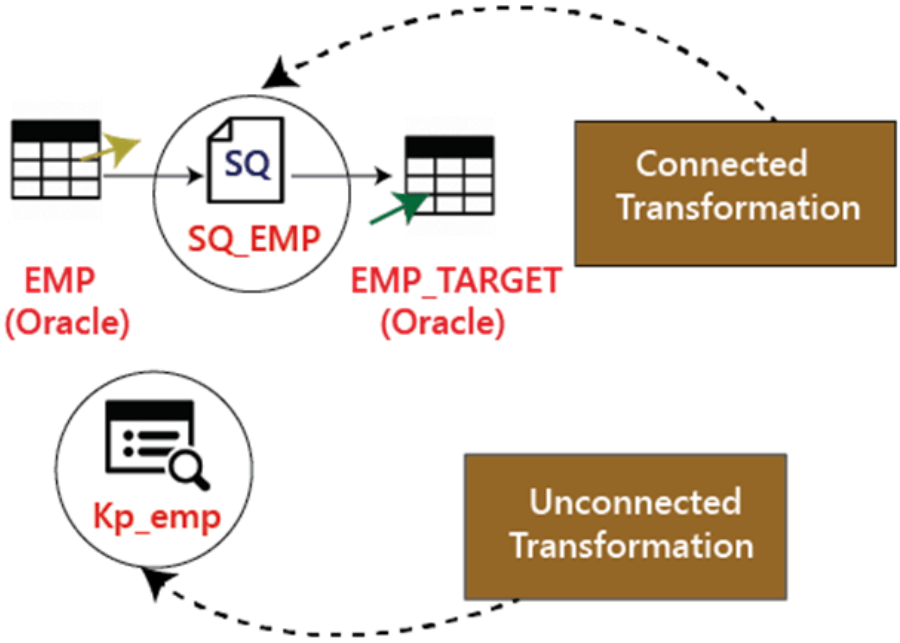
For example, Source qualifier transformation of Source table Stud is connected to filter transformation to filter students of a class.

Those transformations whose not link to any other transformations are called **unconnected transformations**.

Their functionality is used by calling them inside other transformations. And these transformations are not part of the pipeline.

The connected transformations are preferred when the transformation is called for every input row or expected to return a value.

The unconnected transformations are useful if their functionality is required periodically only or based upon certain conditions. For example, calculate the tax details if tax value is not available.



Transformations based on the change in several rows

- Active Transformations
- Passive Transformations

Active Transformations are those who modify the data rows, and the number of input rows passed to them. For example, if a transformation receives 10 numbers of rows as input, and it returns 15 numbers of rows as an output, then it is an active transformation. In the active transformation, the data is modified in the row.

Passive Transformations do not change the number of input rows. In passive transformations, the number of input and output rows remains the same, and data is modified at row level only. In the passive transformation, we cannot create new rows, and no existing rows dropped.

List of Transformations in Informatica

- Source Qualifier Transformation
- Aggregator Transformation
- Router Transformation
- Joiner transformation
- Rank Transformation
- Sequence Generator Transformation
- Transaction Control Transformation
- Lookup and Re-usable transformation
- Normalizer Transformation
- Performance Tuning for Transformation
- External Transformation
- Expression Transformation

Transformation	Type	Description
Aggregator	Active Connected	Performs aggregate calculations.
Expression	Passive Connected	Calculates a value.
Java	Active Connected or Passive Connected	Executes user logic coded in Java. The bytecode for the user logic is stored in the repository
Joiner	Active Connected	Joins data from different databases or flat file systems.
Lookup	Active Connected or Passive Connected or Active Unconnected or Passive Unconnected	Lookup and return data from a flat file, relational table, view, or synonym.
Normalizer	Active Connected	Used in the pipeline to normalize data from relational or flat file sources.
Rank	Active Connected	Limits records to a top or bottom range.
Router	Active Connected	Routes data into multiple transformations based on group conditions.
SQL	Active Connected or Passive Connected	Executes SQL queries against a database.
Union	Active Connected	Merges data from different databases or flat file systems.
XML Generator	Active Connected	Reads data from one or more input ports and outputs XML through a single output port.
XML Parser	Active Connected	Reads XML from one input port and outputs data to one or more output ports.
XML Source Qualifier	Active Connected	Represents the rows that the Integration Service reads from an XML source when it runs a session.

Exercise:

1. Use one active and one connected transformation in Informatica and attach screen shot.

Lab # 6

Object:
Filter Transformation. Source Qualifier Transformation.

Theory:
Filter Transformation
Filter Transformation is an active transformation because it changes the number of records. We can filter the records according to the requirements by using the filter condition.

Example
For loading the student records having rollno equal to 20 only, we can put filter transformation in the mapping with the filter condition rollno=20. So only those records which have rollno =20 will be passed by filter transformation, rest other records will be dropped.

Step1: Create a mapping having source "Stu" and target "Stu_target".
Step2: Then in the mapping
1. Select Transformation menu
2. Select the create option.

Step3: In the create transformation window
1. Select Filter Transformation from the list.
2. Enter Transformation name fltr_rollno_20
3. Select create option

Step4: The filter transformation will be created, click on the done button in the creative transformation window.
Step5: in the mapping,
1. Drag and drop all the source qualifier columns from the filter transformation.
2. And Link the columns of filter transformation to the target table.

Step6: Double click on the filter transformation to open its properties, and
1. Select the properties menu.
2. Click on the filter condition editor.

Step7: Then,
1. Enter filter condition rollno=20.
2. Click on the OK button.

Step8: Again in the edit transformation window,
1. We will see the filter condition in the properties tab.
2. Click on the OK button.

Now save the created mapping and execute this after creating session and workflow. In the target table, only the rollno = 20 will be loaded from the record.

Source Qualifier Transformation
The source qualifier transformation is active and connected. It is used to represent the rows that the integrations service reads when it runs a session. We need to join the source qualifier transformation with the relational or flat file definition in a mapping.

The source qualifier transformation converts the source data types in the Informatica native data types. That's why there is no need to alter the data types of the ports.

The source qualifier transformation does the following tasks, such as:

Joins: We can join two or more tables from the same source database. By default, the sources are merged based on the primary key and foreign key relationships. This can be changed by specifying the join condition in the "user-defined join" property.

Filter rows: We can filter the rows from the source database. In the default query, the integration service adds a WHERE clause.

Sorting input: We can sort the source data by specifying the number for sorted ports. In the default SQL query, the Integration Service adds an ORDER BY clause.

Distinct rows: We can get separate rows from the source by choosing the "Select Distinct" property. In the default SQL query, the Integration Service adds a SELECT DISTINCT statement.

Custom SQL Query: We can write your SQL query to do calculations.

Source Qualifier Transformation Properties

The source qualifier transformation has the following properties, such as:

Option	Description
SQL Query	It defines a custom query that replaces the default query of the Integration Service, which is used to read data from sources. A custom query overrides entries for a custom join or a source filter.
User-Defined Join	It specifies the condition which is used to join data from multiple sources represented in the same Source Qualifier transformation.
Source Filter	It specifies the filter condition of the Integration Service that applies while querying the rows.
Number of Sorted Ports	It indicates the number of columns, and it is used during the sorting rows queried from relational sources. If we go with this option, the Integration Service adds an ORDER BY to the default query when it reads source rows. The ORDER BY includes the number of ports specified, starting from the top of the transformation. When selected, the database sort order must match the session sort order.
Tracing Level	It sets the amount of detail included in the session log when we run a session containing this transformation.
Select Distinct	Specifies if you want to select unique rows. The Integration Service includes a SELECT DISTINCT statement if you choose this option.
Pre-SQL	Pre-session SQL commands are used to run against the source database before the Integration Service reads the source.
Post-SQL	Post-session SQL commands are used to run against the source database after the Integration Service writes to the target.
Output is Deterministic	Relational source or transformation output that does not change between session runs when the input data is consistent between runs.

	When you configure this property, the Integration Service does not stage source data for recovery if transformations in the pipeline always produce repeatable data.
Output is Repeatable	Relational source or transformation output that is in the same order between session runs when the request of the input data is consistent. When the output is deterministic, and output is repeatable, the Integration Service does not stage source data for recovery.

Examples

In this example, we want to modify the source qualifier of the mapping "m_emp_emp_target", so instead of returning all the columns, it will return only selected columns.

Step1: Open mapping "m_stud_stud_target" in mapping designer.
Step2: Double click on the Source Qualifier transformation "SQ_STUD". It will open the edit transformation property window for it. Then

1. Click on the properties tab.
2. Click on the SQLQuery Modify option, and this will open an SQL editor window.

Step3: In the SQL editor window

1. Enter the following query
SELECT STUDNO, SNAME, CLASS, SEC FROM STUD
2. Click on the OK button.

Step4: In the "edit transformations" window,

1. Select the Ports tab from the menu.
2. Under the ports tab, you will see all the ports. Keep only the ports STUDNO, SNAME, CLASS, SEC and delete other ports

Step5: After the deletion of ports, click OK Button.
Now, again click on the properties tab in the Edit Transformations window, and we will see only selected data. After clicking the "OK" button it will open SQL Editor Window, and

1. It will confirm the data you have chosen are correct and ready for loading into the target table.
2. Click on the OK button.

Save the mapping (using ctrl+s) and execute the workflow. After execution, only the selected columns will be loaded into the target.

Exercise:

1. Use Filter and Source Qualifier transformation on some exemplary dataset in Informatica and attach screen shot.

Lab # 7

Object:
Aggregator Transformation. Router Transformation

Theory:
Aggregator Transformation
Aggregator transformation is an active transformation. And it is used to perform calculations on the data such as sums, averages, counts, etc.

The integration service stores the group of data and row data in the aggregate cache. The Aggregator Transformation is more beneficial in comparison to the SQL. We can use conditional clauses to filter rows.

Properties of Aggregator Transformation
Some features of aggregator transformation, such as:
- Aggregate Expression
- Group by port
- Sorted Input
- Aggregate cache
- Unsorted Input

Aggregate Expression
Aggregate functions are used to drive the aggregate expression, which can be developed either in variable ports or output ports only.

Sorted input
Group by ports are sorted using a sorted transformation and receive the sorted data as an input to improve the performance of data aggregation. It keeps the sorted transformation before the aggregator transformation to perform sorting on fro up by ports.

Aggregate Cache
An integration service creates an aggregate cache.

Unsorted inputs
The aggregate cache contains group by ports, non-group by input ports, and output port, which provides aggregate expression.

Aggregate Expressions
This transformation offers more functionality to the comparison of SQL's group by statements. Because one can apply conditional logic to groups within the aggregator transformation. Many different aggregate functions can be used to individual output ports within the transformation. Below is the list of these aggregate functions, such as:
- AVG
- COUNT
- FIRST
- LAST
- MAX
- MEDIAN
- MIN
- PERCENTILE
- STDDEV
- SUM
- VARIANCE

Creating an Aggregator Transformation

Follows the following steps, such as:

Step1: Go to the Mapping Designer, click on transformation in the toolbar and create.
Step2: Select the Aggregator transformation, enter the name, and click create.
Step3: Then click on the Done button.
It will create an aggregator transformation without ports. To create ports, we can either drag the ports to the aggregator transformation or create in the ports tab of the aggregator.

Configuring the Aggregator Transformation

We can configure the following components in aggregator transformation in the Informatica.

Aggregate Cache: The integration service stores the group values in the index cache and row data in the data cache.

Aggregate Expression: We can enter expressions in the output port or variable port.

Group by Port: This option tells the integration service on how to create groups. We can configure input, output, or variable ports for the group.

Sorted Input: This option is used to improve session performance. This option will apply only when the input to the aggregator transformation in sorted on the group by ports.

Informatica Nested Aggregate Functions

We can nest one aggregate function within another aggregate function. We can either use single-level aggregate functions or multiple nested functions in an aggregate transformation.

But we cannot use both single-level and nested aggregate functions in an aggregator transformation Informatica. The mapping designer marks the mapping as invalid if an aggregator transformation contains both single-level and nested aggregate functions. If we want to create both single-level and nested aggregate functions, create separate aggregate transformations.

Incremental Aggregation in Informatica

We can enable the session option and Incremental Aggregation After creating a session that includes an Aggregator transformation. When the integration service performs incremental aggregation, it passes source data through the mapping and uses historical cache data to perform aggregation calculations incrementally.

Router Transformation

Router transformation is an active and connected transformation, and it is similar to the filter transformation, which is used to filter the source data.

In a Router transformation, Data Integration is used as a filter condition to evaluate each row of incoming data. It checks the conditions of each user-defined group before processing the default group.

If a row connects more than one group filter condition, Data Integration passes the row multiple times. We can drop rows that do not meet any of the conditions to a default output group.

If we need to check the same input data based on multiple conditions, then we use a Router transformation in a mapping instead of creating multiple Filter transformations.

The following table compares the Router transformation to the Filter transformation:

Options	**Router**	**Filter**	
Conditions	Test for multiple conditions in a single Router transformation	Check for one condition per Filter transformation	
Handle	rows that do not meet the condition	Route rows to the default output group or drop rows that do not meet the condition	Drop rows that do not meet the condition
Incoming data	The process once with a single Router transformation	Process in each Filter transformation	

For example, when filtering the data form rollno =20, we can also get those records where rollno is not equal to 20. So, router transformation gives multiple output groups, and each output group can have its filter condition.

Also, there is a default group, and this default group has record sets that don't satisfy any of the group conditions.

For example, if we have created two groups for the filter conditions rollno=20 & rollno=30 respectively, then those records which are not having rollno 20 and 30 will be passed into this default group.

The data which is rejected by the filter groups will be collected by this default group, and sometimes there can be a requirement to store these rejected data. In this way, the default output group can be useful.

To allow multiple filter conditions, the router transformation provides a group option.

- There is a default input group that takes input data.
- There is also a default output group that contains all those data which are not passed by any filter condition.
- For every filter condition, an output group is created in router transformation. We can connect different targets to these different groups.

Creating Router Transformation

Follows the following steps to create the router transformation, such as:

Step1: Create a mapping having source "STUD" and target "STUD_TARGET."
Step2: Then in the mapping

1. Select Transformation menu.
2. Click on the **Create**

Step3: In the create transformation window

1. Select router transformation
2. Enter a name for the transformation "rtr_rollno_20"
3. And click on the **Create**

Step4: The router transformation will be created in the mapping, select done option in the window.
Step5: Drag and drop all the columns from Source qualifier to router transformation.
Step6: Double click on the router transformation, then in the transformation property of it

1. Select the group tab.
2. Enter the group name rollno_30.
3. Click on the group filter condition.

Step7: In the expression editor, enter the filter condition rollno=30 and select the OK button.
Step8: Click on the **OK** button in the group window.
Step9: Connect the ports from the group rollno_30 of router transformation to target table ports.

Now, when we execute this mapping, the filtered records will get loaded into the target table.

Exercise:

1. Use Aggregator and Router transformation on some exemplary dataset in Informatica and attach screen shot.

Lab # 8

Object:
Joiner Transformation. Rank Transformation

Theory:
Joiner Transformation
Joiner transformation is an active and connected transformation. It provides the option of creating joins in the Informatica. By using the joiner transformation, the created joins are similar to the joins in databases.

The joiner transformation is used to join two heterogeneous sources. The joiner transformation joins sources on the basis of a condition that matches one or more pairs of columns between the two sources.

The two input pipelines include a master and a detail pipeline. We need to join the output of the joiner transformation with another source to join more than two sources. And to join n number of sources in mapping, we need n-1 joiner transformations.

In joiner transformation, there are two sources which we are using for joins, such as:

- Master Source
- Detail Source

In the properties of joiner transformation, we can select which data source can be a Master source and which source can be a detail source.

During execution, the master source is cached into the memory for joining purpose. So it is necessary to select the source with less number of records as the master source.

Configuring Joiner Transformation
In Informatica, we configure the following properties of joiner transformation, such as:

- **Case-Sensitive String Comparison:** The integration service uses this option when we are performing joins on string columns. By default, the case sensitive string comparison option is checked.
- **Cache Directory:** Directory used to cache the master or detail rows. The default directory path is $PMCacheDir. We can override this value as well.
- **Join Type:** The type of join to be performed as Master Outer Join, Detail Outer Join, Normal Join, or Full Outer Join.
- **Tracing Level:** It is used to track the Level of tracing in the session log file.
- **Joiner Data Cache Size:** It tells the size of the data cache. And Auto is the default value of the data cache size.
- **Joiner Index Cache Size:** It tells the size of the index cache. And Auto is the default value of the index cache size.
- **Sorted Input:** This option is used when the input data is in sorted order. And it gives better performance.
- **Master Sort Order:** It gives the sort order of the master source data. If the master source data is sorted in ascending order, then we choose Ascending. We have to

enable the Sorted Input option if we choose Ascending. And Auto is the default value for this option.

- **Transformation Scope:** We can select the transformation scope as All Input or Row.

Types of Joins

In Informatica, the following joins can be created using joiner transformation, such as:

Master outer join

In Master outer join, all records from the Detail source are returned by the join, and only matching rows from the master source are returned.

Detail outer join

In detail outer join, only matching rows are returned from the detail source, and all rows from the master source are returned.

Full outer join.

In full outer join, all records from both the sources are returned. Master outer and Detail outer joins are equivalent to left outer joins in SQL.

Normal join

In normal join, only matching rows are returned from both the sources.

Example

In the following example, we will join emp and dept tables using joiner transformation in the following steps:

Step1: Create a new target table EMP_DEPTNAME in the database using the below script and import the table in Informatica targets.
Step2: Create a new mapping and import source tables "EMP" and "DEPT" and target table, which we created in the previous step.
Step3: From the transformation menu, select create option and,

1. Select joiner transformation
2. Enter transformation name "jnr_emp_dept"
3. Select create option

Step4: Drag and drop all the columns from both the source qualifiers to the joiner transformation.
Step5: Double click on the joiner transformation, then in the edit transformation window:

1. Select the condition tab.
2. Click on the add new condition icon.
3. Select deptno in master and detail columns list.

Step6: Then, in the same window:

1. Select the properties tab.
2. Select normal Join as join type.
3. Click on the OK button.

For performance optimization, we assign the master source to the source table pipeline, which is having less number of records. To perform this task:
Step7: Double click on the joiner transformation to open the edit properties window, and then

1. Select the ports tab.
2. Select any column of a particular source that you want to make a master.
3. Click on the OK button.

Step8: Link the relevant columns from the joiner transformation to the target table.

Now save the mapping and execute it after creating a session and workflow for it. The join will be created using Informatica joiner, and relevant details will be fetched from both the tables.

Sorted Input

When both the Master and detail source are sorted on the ports specified in the join condition, then use the sorted input option in the joiner properties tab.

We can improve the performance by using the sorted input option as the integration service performs the join by minimizing the number of disk IOs. It gives excellent performance when we are working with large data sets.

Some steps to configuring the sorted input option, such as:

- Sort the master and detail source either by using the source qualifier transformation or sorter transformation.
- Sort both the source on the ports to be used in join conditions either in ascending or descending order.
- Specify the Sorted Input option in the joiner transformation properties tab.

Blocking Transformation

The joiner Transformation is called as the blocking transformation. The integration service blocks and unblocks the source data depending on whether the joiner transformation is configured for sorted input or not.

Unsorted Joiner Transformation

In the case of unsorted joiner transformation, the integration service first reads all the master rows before it reads the detail rows.

The integration service blocks the detail source while it caches all the master rows. Once it reads all the master rows, then it unblocks the detail source and understands the details rows.

Sorted Joiner Transformation

The blocking logic may or may not possible in case of sorted joiner transformation. The integration service uses blocking logic if it can do so without blocking all sources in the target load order group. Otherwise, it does not use blocking logic.

Limitations of Joiner Transformation

- We cannot use joiner transformation when the input pipeline contains an update strategy transformation.
- We cannot connect a sequence generator transformation directly to the joiner transformation.

Rank Transformation

Rank is an active and connected transformation that performs the filtering of data based on the group and ranks. The rank transformation also provides the feature to do ranking based on groups.

The rank transformation has an output port, and it is used to assign a rank to the rows.

In Informatica, it is used to select a bottom or top range of data. While string value ports can be ranked, the Informatica Rank Transformation is used to rank numeric port values. One might think MAX and MIN functions can accomplish this same task.

However, the rank transformation allows groups of records to be listed instead of a single value or record. The rank transformation is created with the following types of ports.

- Input port (I)
- Output port (O)
- Variable port (V)
- Rank Port (R)

Rank Port

The port which is participated in a rank calculation is known as Rank port.

Variable Port

A port that allows us to develop expression to store the data temporarily for rank calculation is known as a variable port.

The variable port will enable us to write expressions that are required for rank calculation.

Ports in a Rank Transformation

Ports	Number Required	Description
I	1 Minimum	Port to receive data from another transformation.
O	1 Minimum	Port we want to pass to other transformations.
V	not needed	It is used to store values or calculations for use in an expression.
R	Only 1	The Rank port is an input or output port. We have linked the Rank port to another transformation. For example: Total Salary

Configuring the Rank Transformation

Let's see how to configure the following properties of Rank transformation:

- **Cache Directory:** The directory is a space where the integration service creates the index and data cache files.
- **Top/Bottom:** It specifies whether we want to select the top or bottom rank of data.
- **Number of Ranks:** It specifies the number of rows that we want to rank.
- **Case-Sensitive String Comparison:** It is used to sort the strings by using the case sensitive.
- **Tracing Level:** The amount of logging to be tracked in the session log file.

- **Rank Data Cache Size:** The data cache size default value is 2,000,000 bytes. We can set a numeric value or Auto for the data cache size. In the case of Auto, the Integration Service determines the cache size at runtime.
- **Rank Index Cache Size:** The index cache size default value is 1,000,000 bytes. We can set a numeric value or Auto for the index cache size. In the case of Auto, the Integration Service determines the cache size at runtime.

What is Rank Index?

The Developer tool creates a rank index port for each Rank transformation. The Data Integration Service uses the Rank Index port to store the ranking position for each row in a group.

After the Rank transformation identifies all rows that belong to a top or bottom rank, it then assigns rank index values. If two rank values match, they receive the same value in the rank index, and the transformation skips the next value.

The rank index is an output port only. We can pass the rank index to another transformation in the mapping or directly to a target.

Defining Groups

The Rank transformation gives us group information like the aggregator transformation.

For example: If we want to select the 20 most expensive items by manufacturer, we would first define a group for each manufacturer.

Example

Suppose we want to load top 5 salaried employees for each department; we will implement this using rank transformation in the following steps, such as:

Step1: Create a mapping having source EMP and target EMP_TARGET
Step2: Then in the mapping,

1. Select the transformation menu.
2. And click on the Create option.

Step3: In the create transformation window,

1. Select rank transformation.
2. Enter transformation name "rnk_salary".
3. And click on the Create button.

Step4: The rank transformation will be created in the mapping, select the done button in the window.
Step5: Connect all the ports from source qualifier to the rank transformation.
Step6: Double click on the rank transformation, and it will open the "edit transformation window". In this window,

1. Select the properties menu.
2. Select the "Top" option from the Top/Bottom property.
3. Enter 5 in the number of ranks.

Step7: In the "edit transformation" window again,

1. Select the ports tab.
2. Select group by option for the Department number column.
3. Select Rank in the Salary Column.
4. Click on the OK button.

Step8: Connect the ports from rank transformation to the target table.

Now, save the mapping and execute it after creating session and workflow. The source qualifier will fetch all the records, but the rank transformation will pass only records having three high salaries for each department.

Exercise:

1. Use Joiner Transformation. Rank Transformation on some exemplary dataset in Informatica and attach screen shot.

Lab # 9

Object:
Sequence Generator Transformation

Theory:
Sequence Generator Transformation
Sequence generator is a passive and connected transformation, and it generates numeric sequence values such as 1, 2, 3, and so on. It does not affect the number of input rows.

The Sequence Generator transformation is used to create unique primary key values and replace missing primary keys.

For example, if we want to assign sequence values to the source records, then we need to use a sequence generator.

The sequence generator transformation consists of two output ports. We cannot edit or delete these ports, such as:

- CURRVAL
- NEXTVAL

NEXTVAL
The NEXTVAL port is used to generate sequence numbers by connecting it to a Transformation or target. The generated sequence numbers are based on the Current Value and Increment By properties.

If the sequence generator is not configuring to Cycle, then the NEXTVAL port makes the sequence numbers up to the set End Value.

We can connect the NEXTVAL port to multiple transformations to generate unique values for each row.

The sequence generator transformation creates a block of numbers at the same time. If the block of numbers is used, then it generates the next block of sequence numbers.

For example, we might connect NEXTVAL to two target tables in mapping to create unique primary key values.

The integration service generates a block of numbers 1 to 10 for the first target. When the first block of numbers has been loaded, only then another block of numbers 11 to 20 will be generated for the second target

CURRVAL
The CURRVAL port is NEXTVAL plus the Increment By value.

We only connect the CURRVAL port when the NEXTVAL port is already linked to a downstream transformation.

If we combine the CURRVAL port without connecting the NEXTVAL port, the Integration Service passes a constant value for each row.

When we combine the CURRVAL port in a Sequence Generator Transformation, then the Integration Service processes one row in each block.

We can optimize performance by connecting only the NEXTVAL port in a Mapping.

Example: Suppose STUD will be a source table.

Create a target STUD_SEQ_GEN_EXAMPLE in the shared folder. Structure the same as STUD. Add two more ports NEXT_VALUE and CURR_VALUE to the target table.

We can create a Sequence Generator transformation to use in a single mapping, or a reusable Sequence Generator transformation to use in multiple mappings.

A reusable Sequence Generator transformation maintains the integrity of the sequence in each mapping that uses an instance of the Sequence Generator transformation.

Properties of Sequence Generator Transformation

Below are the following properties to configure a sequence data object and a new sequence:

Property	Description
Start Value	The start value of the generated sequence is the Integration Service when using the Cycle option. If we select Cycle, the Integration Service cycles back to this value when it reaches the end value. The default value is 0. Maximum value is 9,223,372,036,854,775,806.
End Value	The maximum value that the Integration Service generates. If the Integration Service reaches this value during the session, and the sequence is not configured to cycle, the session fails. Maximum value is 9,223,372,036,854,775,807.
Increment Value	Difference between two consecutive values from the NEXTVAL port. The default value is 1. And it must be a positive integer. Maximum value is 2,147,483,647.
Cycle	If enabled, the Integration Service cycles through the sequence range and start over with the start value. If disabled, the Integration Service stops the sequence at the configured end value. The Integration Service fails the session with overflow errors if it reaches the end value and still has rows to process.
Reset	If enabled, the Integration service resets the sequence data object to the start value when the mapping completely run. If disabled, the Integration Service increments the current value after the mapping run ends, and uses that value in the next mapping run. This property is disabled for reusable Sequence Generator transformations and for non-reusable Sequence Generator transformations that use a reusable sequence data object.

Tracing Level	Level of detail about the transformation that the Integration Service writes into the mapping log. We can choose terse, normal, verbose initialization or verbose data. Normal sets as a default level.
Maintain Row Order	Maintain the row order of the input data to the transformation. Select this option if the Integration Service should not perform any optimization that can change the row order.

Example

In the below example, we will generate sequence numbers and store in the target in the following steps, such as:

Step1: Create a target table.
Step2: Import that created table in Informatica as the target table.
Step3: Create a new mapping and import STUD source and STUD_SEQUENCE target table.
Step4: Create a new transformation in the mapping,

1. Select sequence transformation as the type.
2. Enter transformation name such as seq_stud.
3. Click on the **Create**

Step5: Sequence generator transformation will be created, then click on the **Done** button.
Step6: Link the NEXTVAL column of sequence generator to the SNO column in the target table.
Step7: Link the other columns from source qualifier transformation to the target table.
Step8: Double click on the sequence generator to open the property window, and then

1. Select the properties tab.
2. Enter the properties with Start value =1, leave other properties as default.
3. Click on the **OK**

Now save the mapping and execute it after creating the session and workflow.

The SNO column in the target would contain the sequence numbers generated by the sequence generator transformation.

Exercise:

1. Why Sequence Generator Transformation is used. What is CURRVAL, NEXTVAL, Increment value and cycle.

Lab # 10

Object:
Transaction Control Transformation

Theory:
Transaction Control Transformation
A Transaction Control transformation is an active and connected transformation. It allows us to commit and rollback transactions based on a set of rows that pass through a Transaction Control transformation.

Commit and rollback operations are of significant importance as it guarantees the availability of data.

A transaction is the set of rows bound by commit or rollback rows. We can define a transaction based on the varying number of input rows. We can also identify transactions based on a group of rows ordered on a common key, such as employee ID or order entry date.

When processing a high volume of data, there can be a situation to commit the data to the target. If a commit is performed too quickly, then it will be an overhead to the system.

If a commit is performed too late, then in the case of failure, there are chances of losing the data. So the Transaction control transformation provides flexibility.

In PowerCenter, the transaction control transformation is defined in the following levels, such as:

- **Within a mapping:** Within a mapping, we use the Transaction Control transformation to determine a transaction. We define transactions using an expression in a Transaction Control transformation. We can choose to commit, rollback, or continue on the basis of the return value of the expression without any transaction change.
- **Within a session:** We configure a session for the user-defined commit. If the Integration Service fails to transform or write any row to the target, then We can choose to commit or rollback a transaction.

When we run the session, then the Integration Service evaluates the expression for each row that enters the transformation. When it evaluates a committed row, then it commits all rows in the transaction to the target or targets. When the Integration Service evaluates a rollback row, then it rolls back all rows in the transaction from the target or targets.

If the mapping has a flat-file as the target, then the integration service can generate an output file for a new transaction each time. We can dynamically name the target flat files.

TCL COMMIT & ROLLBACK Commands

There are five in-built variables available in the transaction control transformation to handle the operation.

- **TC_CONTINUE_TRANSACTION**
  The Integration Service does not perform any transaction change for the row. This is the default value of the expression.
- **TC_COMMIT_BEFORE**
  The Integration Service commits the transaction, begins a new transaction, and writes the current row to the target. The current row is in the new transaction. In tc_commit_before, when this flag is found set, then a commit is performed before the processing of the current row.
- **TC_COMMIT_AFTER**
  The Integration Service writes the current row to the target, commits the transaction, and begins a new transaction. The current row is in the committed transaction. In tc_commit_after, the current row is processed then a commit is performed.
- **TC_ROLLBACK_BEFORE**
  The Integration Service rolls back the current transaction, begins a new transaction, and writes the current row to the target. The current row is in the new transaction. In tc_rollback_before, rollback is performed first, and then data is processed to write.
- **TC_ROLLBACK_AFTER**
  The Integration Service writes the current row to the target, rollback the transaction, and begins a new transaction. The current row is in the rolled-back transaction. In tc_rollback_after data is processed, then the rollback is performed.

How to Create Transaction Control Transformation

Follows the following steps to create transaction control transformation, such as:

Step1: Go to the mapping designer.
Step2: Click on transformation in the toolbar, and click on the **Create** button.
Step3: Select the transaction control transformation.
Step4: Then, enter the name and click on the **Create** button.
Step5: Now click on the **Done** button.
Step6: We can drag the ports into the transaction control transformation, or we can create the ports manually in the ports tab.
Step7: Go to the properties tab.
Step8: And enter the transaction control expression in the Transaction Control Condition.

Configuring Transaction Control Transformation

Here are the following components which can be configuring in the transaction control transformation, such as:

- **Transformation Tab**: It can rename the transformation and add a description.
- **Ports Tab**: It can create input or output ports.
- **Properties Tab**: It can define the transaction control expression and tracing level.
- **Metadata Extensions Tab**: It can add metadata information.

Transaction Control Expression

We can enter the transaction control expression in the Transaction Control Condition option in the properties tab.

The transaction control expression uses the IIF function to check each row against the condition.

Syntax

Syntax for the Transaction Control transformation expression, such as:

IIF (condition, value1, value2)

For example:

IIF (dept_id=11, TC_COMMIT_BEFORE,TC_ROLLBACK_BEFORE)

Example

In the following example, we will commit data to the target when dept no =10, and this condition is found true.

Step1: Create a mapping with EMP as a source and EMP_TARGET as the target.
Step2: Create a new transformation using the transformation menu, then

1. Select a transaction control as the new transformation.
2. Enter transformation name tc_commit_dept10.
3. And click on the create button.

Step3: The transaction control transformation will be created, then click on the done button.
Step4: Drag and drop all the columns from source qualifier to the transaction control transformation then link all the columns from transaction control transformation to the target table.
Step5: Double click on the transaction control transformation and then in the edit property window:

1. Select the property tab.
2. Click on the transaction control editor icon.

Step6: In the expression editor enter the following expression:

1. "iif(deptno=10,tc_commit_before,tc_continue_transaction)".
2. And click on the OK button.
3. It means if deptno 10 is found, then commit transaction in target, else continue the current processing.

Step7: Click on the OK button in the previous window.

Now save the mapping and execute it after creating sessions and workflows. When the department number 10 is found in the data, then this mapping will commit the data to the target

Exercise:

1. Write in your own words (in one sentence) about TCL Commit and Rollback Commands in Transaction Control Transformation.

2. What will be the syntax of the expression for Transaction Control Transformation of we want to commit data to the target when dept no =30, and this condition is found true.

Lab # 11

Object:
Lookup transformation

Theory:
Lookup Transformation
Lookup transformation is used to look up a source, source qualifier, or target to get the relevant data.

It is a kind of join operation in which one of the joining tables is the source data, and the other joining table is the lookup table.

The Lookup transformation is used to retrieve data based on a specified lookup condition. For example, we can use a Lookup transformation to retrieve values from a database table for codes used in source data.

When a mapping task includes a Lookup transformation, then the task queries the lookup source based on the lookup fields and a lookup condition. The Lookup transformation returns the result of the lookup to the target or another transformation.

We can configure the Lookup transformation to return a single row or multiple rows. This is the passive transformation which allows performing the lookup on the flat files, relational table, views, and synonyms.

When we configure the Lookup transformation to return multiple rows, the Lookup transformation is an active transformation. The lookup transformation supports horizontal merging, such as **equijoin** and **nonequijoin**.

When the mapping contains the work up transformation, the integration service queries the lock up data and compares it with lookup input port values.

The lookup transformation is created with the following type of ports, such as:

- Input port (I)
- Output port (O)
- Look up Ports (L)
- Return Port (R)

Perform the following tasks using a Lookup transformation, such as:

- **Get a related value**: Retrieve a value from the lookup table on the basis of a value in the source. For example, the source has a student rollno. Retrieve the student name from the lookup table.
- **Get multiple values**: Retrieve the multiple rows from a lookup table. For example, return all students in a class.
- **Perform a calculation**: Retrieve any value from a lookup table and use it in a calculation. For example, retrieve the marks, calculate the percentage, and return the percentage to a target.

- **Update slowly changing dimension tables**: Determine the rows that exist in the target.

Configure the Lookup Transformation

Configure the Lookup transformation to perform the different types of lookups, such as:

Relational or flat-file lookup: Perform a lookup on a flat file or a relational table. When we create a Lookup transformation by using a relational table as the lookup source, we can connect to the lookup source using ODBC and import the table definition as the structure for the Lookup transformation.
When we create a Lookup transformation by using a flat-file as a lookup source, the Designer invokes the Flat-file Wizard.

Pipeline lookup: Perform a lookup on the application sources such as JMS or MSMQ. Drag the source into the mapping and associate the Lookup transformation with the source qualifier. When the Integration Service retrieves source data for the lookup cache then configure the partitions to improve performance.

Connected or unconnected lookup: A connected Lookup transformation receives source data, performs a lookup, and returns data to the pipeline. Or the unconnected Lookup transformation is not connected to a target.
A transformation in the pipeline calls the Lookup transformation with a :LKP expression. The unconnected Lookup transformation returns one column to the calling transformation.

Cached or uncached lookup: Cache the lookup source to improve performance. We can use static or dynamic cache for caching the lookup source.
By default, the lookup cache remains static and does not change during the session. With a dynamic cache, the Integration Service inserts or updates rows in the cache. When we cache the target table as the lookup source, we can look up values in the cache to determine if the values exist in the target. The Lookup transformation marks rows to insert or update the target.

Exercise:

1. Use Lookup Transformation on some exemplary dataset in Informatica and attach screen shot.

Lab # 12

Object:
Normalizer Transformation

Theory:
Normalizer Transformation
The Normalizer is an active transformation. It is used to convert a single row into multiple rows. When the Normalizer transformation receives a row that contains multiple-occurring data, it returns a row for each instance of the multiple-occurring data.

If in a single row, there is repeating data in multiple columns, then it can be split into multiple rows. Sometimes we have data in multiple occurring columns.

For example, a relational source includes four fields with flat sales data. We can configure a Normalizer transformation to generate a separate output row for each flat.

When the Normalizer returns multiple rows from an incoming row, it returns duplicate data for single-occurring incoming columns.

The Normalizer transformation receives a row that contains multiple-occurring columns and returns a row for each instance of the multiple-occurring data. The transformation processes multiple-occurring columns or multiple-occurring groups of columns in each source row.

Here are the following properties of Normalizer transformation in the Properties panel, such as:

Normalized Fields Tab: Define the multiple-occurring fields and specify additional fields that you want to use in the mapping.

Field Mapping Tab: Connect the incoming fields to the normalized fields.
We need the appropriate license to use the Normalizer transformation.

The Normalizer transformation parses multiple-occurring columns from COBOL sources, relational tables, or other sources. It can process multiple record types from a COBOL source that contains a REDEFINES clause.

Normalizer Transformation Types
Here are the two types of Normalizer transformation, such as:

VSAM Normalizer Transformation: A non-reusable transformation that is a Source Qualifier transformation for a COBOL source. The Mapping Designer creates VSAM Normalizer columns from a COBOL source in a mapping.
The column attributes are read-only. The VSAM Normalizer receives a multiple-occurring source column through one input port.

Pipeline Normalizer Transformation: A transformation that processes multiple-occurring data from relational tables or flat files.

We create the columns manually and edit them in the Transformation Developer or Mapping Designer. The pipeline Normalizer transformation represents multiple-occurring columns with one input port for each source column occurrence.

Example

We create the following table that represents the student marks records of different classes, such as:

Step1: Create the source table "stud_source" and target table "stud_target".

Student Name	Class 7	Class 8	Class 9	Class 10
Joy	60	65	75	80
Edward	65	70	80	90

Step2: Create a mapping having source **stud_source** and target table **stud_target**.
Step3: From the transformation menu create a new transformation

1. Select normalizer as transformation.
2. Enter the name **nrm_stud**.
3. And click on the **Create**

Step4: The transformation will be created, then click on the **Done** button.
Step5: Double click on the normalizer transformation, then

1. Select the normalizer tab.
2. Click on icon to create two columns.
3. Enter column names.
4. Set number of occurrences to 4 for marks and 0 for student name.
5. Click on the **OK**

Columns will be generated in the transformation. We will see 4 number of marks column as we set the number of occurrences to 4.
Step 6: Then in the mapping

1. Link the four-column of source qualifier of the four class to the normalizer columns, respectively.
2. Link the student name column to the normalizer column.
3. Link student_name & marks columns from normalizer to the target table.

Save the mapping and execute it after creating session and workflow. The class score column is repeating in four columns. For each class score of the student, a separate row will be created by using the Normalizer transformation.

The output of the above mapping will look like the following:

Student Name	Class	Score
Joy	7	60
Joy	8	65
Joy	9	75
Joy	10	80
Edward	7	65
Edward	8	70

Edward	9	80
Edward	10	90

The source data had repeating columns, namely class7, class 8, class 9, and class 10. We have rearranged the data to fit into a single column of class, and for one source record, four records are created in the target by using Normalizer.

In this way, we can normalize data and create multiple records for a single source of data.

Exercise:

1. Use Normalizer Transformation on some exemplary dataset in Informatica and attach screen shot.

Lab # 13

Object:
Workflows in Informatica

Theory:
Workflow is a group of instructions/commands to the integrations service in Informatica. The integration service is an entity which reads workflow information from the repository, fetches data from sources and after performing transformation loads it into the target.

Workflow - It defines how to run tasks like **session task, command task, email task**, etc.

Create a Workflow
- You first need to create tasks
- And then add those tasks to the workflow.

A Workflow is like an empty container, which has the capacity to store an object you want to execute. You add tasks to the workflow that you want to execute. In this tutorial, we are going to do following things in workflow.

Workflow execution can be done in two ways

- **Sequence**: Tasks execute in the order in which they are defined
- **Event based:** Tasks gets executed based on the event conditions.

How to open Workflow Manager
Step1 – In the Informatica Designer, Click on the Workflow manager icon
Step2 – This will open a window of Workflow Manager. Then, in the workflow Manager. Enter user name and password then select "Connect Button".
Step 3- In the workflow manager.

1. Right click on the folder
2. In the pop up menu, select open option

This will open up the workspace of Workflow manager.

How to Create Connections for Workflow Manager
To execute any task in workflow manager, you need to create **connections**. By using these connections, Integration Service connects to different objects.

For Example, in your mapping if you have source table in oracle database, then you will need oracle connection so that integration service can connect to the oracle database to fetch the source data.
Following type of connections can be created in workflow manager.
- Relational Connection
- Ftp Connection
- Queue
- Application

The choice of connection you will create, will depend on the type of source and target systems you want to connect. More often, you would be using **relational connections.**

To Create a Relational Connection

Step1 – In Workflow Manager

1. Click on the Connection menu
2. Select Relational Option

Step2 – In the pop up window

1. Select Oracle in type
2. Click on the new button

Step3 – In the new window of connection object definition

1. Enter Connection Name (New Name-guru99)
2. Enter username
3. Enter password
4. Enter connection string
5. Leave other settings as default and Select OK button

Step4 – You will return on the previous window. Click on the close button.

Now you are set with the relational connection in workflow manager.

Components of Workflow Manager

There are three component tools of workflow manager that helps in creating various objects in workflow manager. These tools are

- Task Developer
- Worklet Designer
- Workflow Designer

Task Developer – Task developer is a tool with the help of which you can create reusable objects. Reusable object in workflow manager are objects which can be reused in multiple workflows. For Example, if you have created a command task in task developer, then you can reuse this task in any number of workflows.

The role of Workflow designer is to execute the tasks those are added in it. You can add any no of tasks in a workflow.

You can create three types of reusable tasks in task developer.

- Command task
- Session task
- Email task

Command Task – A command task is used to execute different windows/unix commands during the execution of the workflow. You can create command task to execute various command based tasks. With help of this task you can execute commands **to create files/folders, to delete files/folders, to do ftp of files** etc.

Session Task - A session task in Informatica is required to run a mapping.

- Without a session task, you cannot execute or run a mapping
- A session task can execute only a single mapping. So, there is a one to one relationship between a mapping and a session

- A session task is an object with the help of which informatica gets to know how and where to execute a mapping and at which time
- Sessions cannot be executed independently, a session must be added to a workflow
- In session object cache properties can be configured and also advanced performance optimization configuration.

Email Task - With the help of email task you can send email to defined recipients when the Integration Service runs a workflow. For example, if you want to monitor how long a session takes to complete, you can configure the session to send an email containing the details of session start and end time. Or, if you want the Integration Service to notify you when a workflow completes/fails, you can configure the email task for the same.

How to Create Command Task

Step1- To create a command task we are going to use Task Developer. In Workflow Manager, open the task developer by clicking on tab "task developer" from the menu.
Step2 – Once task developer is opened up, follow these steps

1. Select Tasks menu
2. Select Create option

Step3 – In the create task window

1. Select command as type of task to create
2. Enter task name
3. Select create button

This will create command task folder. Now you have to configure the task to add command in it, that we will see in next step.
Step4 – To configure the task, double click on the command task icon and it will open an "edit task window". On the new edit task window

1. Select the commands menu
2. Click on the add new command icon
3. Enter command name
4. Click on the command icon to add command text
5. This will open a command editor box.

Step5 – On the command editor box, enter the command "mkdir C:\guru99" (this is the windows command to create a folder named "guru99") and select OK.
After this step you will return to the edit tasks window and you will be able to see the command you added in to the command text box.
Step6 – Click OK on the edit task window,

The command task will be created in the task developer under "Guru99" repository.

How to Create Workflow to Execute Command Task

To execute command taks you have to switch on to workflow designer. A workflow designer is a parent or container object in which you can add multiple tasks and when workflow is executed, all the added tasks will execute. To create a workflow

Step1 – Open the workflow designer by clicking on workflow designer menu
Step2 – In workflow designer

1. Select workflows menu
2. Select create option

Step3 – In create workflow window

1. Enter workflow name

2. Select OK Button (leave other options as default). This will create the workflow.

Naming Convention - Workflow names are prefixed with using '**wkf_**', if you have a session named '**s_m_employee_detail**' then workflow for the same can be named as '**wkf_s_m_employee_detail**'.

When you create a workflow, it does not consist of any tasks. So, to execute any task in a workflow you have to add task in it.

Step4 - To add command task that we have created in Task developer to the workflow designer

1. In the navigator tree, expand the tasks folder
2. Drag and drop the command task to workflow designer

Step5 - Select the "link task option" from the toolbox from the top menu. (The link task option links various tasks in a workflow to the start task, so that the order of execution of tasks can be defined)..

Step6 – Once you select the link task icon, it will allow you to drag the link between start task and command task. Now select the start task and drag a link to the command task.

Now you are ready with the workflow having a command task to be executed.

How to Execute Workflow

Step1 – To execute the workflow

1. Select workflows option from the menu
2. Select start workflow option

This will open workflow monitor window and executes the workflow

Once the workflow is executed, it will execute the command task to create a folder in the defined directory.

Session Task

A session task in Informatica is required to run a mapping.

Without a session task, you cannot execute or run a mapping and a session task can execute only a single mapping. So, there is a one to one relationship between a mapping and a session. A session task is an object with the help of which Informatica gets to know how and where to execute a mapping and at which time. Sessions cannot be executed independently, a session must be added to a workflow. In session object cache properties can be configured and also advanced performance optimization configuration.

How to Create a Session Task.

Here we will create a session task for the mapping "m_emp_emp_target" which was created previously.

Step1 – Open Workflow manager and open task developer

Step2 – Now once the task developer opens, in the workflow manager go to main menu

1. Click on task menu
2. Select create option
3. This will open a new window "Create Task"

Step3 – In the create task window

1. Select session task as type of task.
2. Enter name of task.

3. Click create button

Step4 – A window for selecting the mapping will appear. Select the mapping which you want to associate with this session, for this example select "m_emp_emp_target" mapping and click OK Button.

Step5 – After that, click on "Done" button.Session object will appear in the task developer

Step6 – In this step you will create a workflow for the session task. Click on the workflow designer icon.

Step7 – In the workflow designer tool

1. Click on workflow menu
2. Select create option

Step8 – In the create workflow window

1. Enter workflow name
2. Select OK. (leave other properties as default, no need to change any properties)

In workflow manager a start task will appear, it's a starting point of execution of workflow.

Step9 – In workflow manager

1. Expand the session's folder under navigation tree.
2. Drag and drop the session you created in the workflow manager workspace.

Step10 - Click on the link task option in the tool box.

Step11 - Link the start task and session task using the link.

Step12 – Double click on the session object in wokflow manager. It will open a task window to modify the task properties.

Step13 – In the edit task window

1. Select mapping tab
2. Select connection property
3. Assign the connection to source and target, the connection which we created in early steps.
4. Select OK Button

Now your configuration of workflow is complete, and you can execute the workflow.

How to Add Multiple Tasks to a Start Task

The start task is a starting point for the execution of workflow. There are two ways of linking multiple tasks to a start task.

- Parallel
- Serial

In parallel linking the tasks are linked directly to the start task and all tasks start executing in parallel at same time.

How to Add Tasks in Parallel

Step1 – In the workflow manager, open the workflow "wkf_run_command"

Step2 – In the workflow, add session task "s_m_emp_emp_target". (by selecting session and then drag and drop)

Step3 – Select the link task option from the toolbox

Step4 - link the session task to the start task (by clicking on start taks, holding the click and connecting to session task).

Step5 – Start the workflow and monitor in the workflow monitor.

How to Add Tasks in Serial Mode

But before we add tasks in serial mode, we have to delete the task that we added to demonstrate parallel execution of task. For that

Step1 – Open the workflow "w.kf_run_command"

1. Select the link to the session task.
2. Select edit option in the menu
3. Select delete option

Step2 – Confirmation dialogue box will appear in a window, select yes option. The link between the start task and session task will be removed.

Step3 – Now again go to top menu and select the link task option from the toolbox

Step4 – link the session task to the command task

Step5 - To make the visual appearance of workflow more clear

1. Right click on wokspace of workflow
2. Select arrange menu
3. Select Horizontal option

If you start the workflow the command task will execute first and after its execution, session task will start.

Workflow Variable

Workflow variables allows different tasks in a workflow to exchange information with each other and also allows tasks to access certain properties of other tasks in a workflow. For example, to get the current date you can use the inbuilt variable "sysdate".

Most common scenario is when you have multiple tasks in a workflow and in one task you access the variable of another task. For example, if you have two tasks in a workflow and the requirement is to execute the second task only when first task is executed successfully. You can implement such scenario using predefined variable in the workflow.

Implementing the scenario

We had a workflow "wkf_run_command" having tasks added in serial mode. Now we will add a condition to the link between session task and command task, so that, only after the success of command task the session task will be executed.

Step1 - Open the workflow "wkf_run_command"

Step2 - Double click on the link between session and command task An Expression window will appear

Step3 – Double click the status variable under "cmd_create_folder" menu. A variable "$cmd_create_folder.status" will appear in the editor window on right side.

Step4 - Now we will set the variable "$cmd_create_folder.status" condition to succeeded status . which means when the previous tasks is executed and the execution was success, then only execute the next session task.

1.Change the variable to "$cmd_create_folder.status=SUCCEEDED" value.
2.Click OK Button

When you execute this workflow, the command task executes first and only when it succeeds then only the session task will get executed.

Workflow Parameter

Workflow parameters are those values which remain constant throughout the run. once their value is assigned it remains same. Parameters can be used in workflow properties and their values can be defined in parameter files. For example, instead of using hard coded connection value you can use a parameter/variable in the connection name and value can be defined in the parameter file.

Parameter files are the files in which we define the values of mapping/workflow variables or parameters. There files have the extension of ".par". As a general standard a parameter file is created for a workflow.

Advantages of Parameter File

- Helps in migration of code from one environment to other
- Allows easy debugging and testing
- Values can be modified with ease without change in code

Structure of Parameter File

The structure of parameter file

- [folder_name.WF:Workflow_name]
- $Parameter_name=Parameter_value

Folder_name is the name of repository folder, workflow name is the name of workflow for which you are creating the parameter file.

We will be creating a parameter file for the database connection "guru99" which we assigned in our early sessions for sources and targets.

How to Create Parameter File

Step1 – Create a new empty file (notepad file)
Step2 – In the file enter text as shown in figure
Step3 – Save the file under a folder guru99 at the location "C:\guru99" as "wkf_run_command.par"
In the file we have created a parameter "$DBConnection_SRC", we will assign the same to a connection in our workflow.

Step4- Open the workflow "wkf_run_command"

1. Select workflows menu
2. Select edit option

Step5 – This will open up edit workflow window, in this window

1. Go to properties tab menu
2. Enter the parameter file name as "c:\guru99\wkf_run_command.par"
3. Select OK Button

Now we are done with defining the parameter file content and point it to a workflow.
Next step is to use the parameter in session.
Step6 - In workflow double click on the session "s_m_emp_emp_target", then

1. Select mappings tab menu
2. Select connection property in the left panel
3. Click on the target connection, which is hardcoded now as "guru99"

Step7 - A connection browser window will appear, in that window

1. Select the option to use connection variable
2. Enter connection variable name as "$DBConnection_SRC"
3. Select Ok Button

Step8 – In the edit task window connection variable will appear for the target, Select OK button in the edit task window.

Now we are done with creating parameter for a connection and assigning its value to parameter file.

When we execute the workflow, the workflow picks the parameter file looks for the value of its parameters/variables in the parameter file and takes those values.

Exercise:

1. Implement Workflow in Informatica and attach screen shot.

Lab # 14

Object:
Dimension Modelling

Theory:
Dimensional Modeling (DM) is a data structure technique optimized for data storage in a Data warehouse. The purpose of dimensional modeling is to optimize the database for faster retrieval of data. The concept of Dimensional Modelling was developed by Ralph Kimball and consists of "fact" and "dimension" tables.

A dimensional model in data warehouse is designed to read, summarize, analyze numeric information like values, balances, counts, weights, etc. in a data warehouse. In contrast, relation models are optimized for addition, updating and deletion of data in a real-time Online Transaction System.

These dimensional and relational models have their unique way of data storage that has specific advantages.

For instance, in the relational mode, normalization and ER models reduce redundancy in data. On the contrary, dimensional model in data warehouse arranges data in such a way that it is easier to retrieve information and generate reports.

Hence, Dimensional models are used in data warehouse systems and not a good fit for relational systems.

Elements of Dimensional Data Model
Fact
Facts are the measurements/metrics or facts from your business process. For a Sales business process, a measurement would be quarterly sales number

Dimension
Dimension provides the context surrounding a business process event. In simple terms, they give who, what, where of a fact. In the Sales business process, for the fact quarterly sales number, dimensions would be

- Who – Customer Names
- Where – Location
- What – Product Name

In other words, a dimension is a window to view information in the facts.

Attributes
The Attributes are the various characteristics of the dimension in dimensional data modeling.
In the Location dimension, the attributes can be

- State
- Country
- Zipcode etc.

Attributes are used to search, filter, or classify facts. Dimension Tables contain Attributes

Fact Table

A fact table is a primary table in dimension modelling.
A Fact Table contains

- Measurements/facts
- Foreign key to dimension table

Dimension Table

- A dimension table contains dimensions of a fact.
- They are joined to fact table via a foreign key.
- Dimension tables are de-normalized tables.
- The Dimension Attributes are the various columns in a dimension table
- Dimensions offers descriptive characteristics of the facts with the help of their attributes
- No set limit set for given for number of dimensions
- The dimension can also contain one or more hierarchical relationships

Types of Dimensions in Data Warehouse

Following are the types of Dimensions in Data Warehouse**:**

- Conformed Dimension
- Outrigger Dimension
- Shrunken Dimension
- Role-playing Dimension
- Dimension to Dimension Table
- Junk Dimension
- Degenerate Dimension
- Swappable Dimension
- Step Dimension

Steps of Dimensional Modelling

The accuracy in creating your Dimensional modeling determines the success of your data warehouse implementation. Here are the steps to create Dimension Model

- Identify Business Process
- Identify Grain (level of detail)
- Identify Dimensions
- Identify Facts
- Build Star

The model should describe the Why, How much, When/Where/Who and What of our business process

Step – 1 Identify the Business Process

Identifying the actual business process a data warehouse should cover. This could be Marketing, Sales, HR, etc. as per the data analysis needs of the organization. The selection of the Business process also depends on the quality of data available for that process. It is the most important step of the Data Modelling process, and a failure here would have cascading and irreparable defects.

Step – 2 Identify the Grain

The Grain describes the level of detail for the business problem/solution. It is the process of identifying the lowest level of information for any table in your data warehouse. If a table contains sales data for every day, then it should be daily granularity. If a table contains total sales data for each month, then it has monthly granularity.

During this stage, you answer questions like

1. Do we need to store all the available products or just a few types of products? This decision is based on the business processes selected for Datawarehouse
2. Do we store the product sale information on a monthly, weekly, daily or hourly basis? This decision depends on the nature of reports requested by executives
3. How do the above two choices affect the database size?

Example of Grain:

The CEO at an MNC wants to find the sales for specific products in different locations on a daily basis.
So, the grain is "product sale information by location by the day."

Step – 3 Identify the Dimensions

Dimensions are nouns like date, store, inventory, etc. These dimensions are where all the data should be stored. For example, the date dimension may contain data like a year, month and weekday.

Example of Dimensions:

The CEO at an MNC wants to find the sales for specific products in different locations on a daily basis.
Dimensions: Product, Location and Time

Attributes: For Product: Product key (Foreign Key), Name, Type, Specifications

Hierarchies: For Location: Country, State, City, Street Address, Name

Step – 4 Identify the Fact

This step is co-associated with the business users of the system because this is where they get access to data stored in the data warehouse. Most of the fact table rows are numerical values like price or cost per unit, etc.

Example of Facts:

The CEO at an MNC wants to find the sales for specific products in different locations on a daily basis.
The fact here is Sum of Sales by product by location by time.

Step – 5 Build Schema

In this step, you implement the Dimension Model. A schema is nothing but the database structure (arrangement of tables). There are two popular schemas

Star Schema

The star schema architecture is easy to design. It is called a star schema because diagram resembles a star, with points radiating from a center. The center of the star consists of the fact table, and the points of the star is dimension tables.

The fact tables in a star schema which is third normal form whereas dimensional tables are de-normalized.

Snowflake Schema

The snowflake schema is an extension of the star schema. In a snowflake schema, each dimension are normalized and connected to more dimension tables.

Rules for Dimensional Modelling

Following are the rules and principles of Dimensional Modeling:

- Load atomic data into dimensional structures.
- Build dimensional models around business processes.
- Need to ensure that every fact table has an associated date dimension table.
- Ensure that all facts in a single fact table are at the same grain or level of detail.
- It's essential to store report labels and filter domain values in dimension tables
- Need to ensure that dimension tables use a surrogate key
- Continuously balance requirements and realities to deliver business solution to support their decision-making

Benefits of Dimensional Modeling

- Standardization of dimensions allows easy reporting across areas of the business.
- Dimension tables store the history of the dimensional information.
- It allows to introduce entirely new dimension without major disruptions to the fact table.
- Dimensional also to store data in such a fashion that it is easier to retrieve the information from the data once the data is stored in the database.
- Compared to the normalized model dimensional table are easier to understand.
- Information is grouped into clear and simple business categories.
- The dimensional model is very understandable by the business. This model is based on business terms, so that the business knows what each fact, dimension, or attribute means.
- Dimensional models are deformalized and optimized for fast data querying. Many relational database platforms recognize this model and optimize query execution plans to aid in performance.
- Dimensional modelling in data warehouse creates a schema which is optimized for high performance. It means fewer joins and helps with minimized data redundancy.
- The dimensional model also helps to boost query performance. It is more denormalized therefore it is optimized for querying.
- Dimensional models can comfortably accommodate change. Dimension tables can have more columns added to them without affecting existing business intelligence applications using these tables.

Exercise:

1. Case Study for Dimension Modelling. (Already discussed in Theory Class)