**Joiner transformation** is an active and connected transformation that provides you the option to create joins in Informatica. The joins created using joiner transformation are similar to the joins in databases. The advantage of joiner transformation is that joins can be created for heterogeneous systems (different databases).

In joiner transformation, there are two sources which we are going to use it for joins. These two sources are called

* Master Source
* Detail Source

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

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

The following joins can be created using joiner transformation

1. 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.

1. 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.

1. 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.

1. Normal join

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

**Source Qualifier Transformation in Informatica**

The source qualifier transformation is an active,connected transformation used to represent the rows that the integrations service reads when it runs a session. You need to connect the source qualifier transformation to the relational or flat file definition in a mapping. The source qualifier transformation converts the source data types to the Informatica native data types. So, you should not alter the data types of the ports in the source qualifier transformation.

The source qualifier transformation is used to do the following tasks:

* **Joins:** You can join two or more tables from the same source database. By default the sources are joined based on the primary key-foreign key relationships. This can be changed by explicitly specifying the join condition in the "user-defined join" property.
* **Filter rows:** You can filter the rows from the source database. The integration service adds a WHERE clause to the default query.
* **Sorting input:** You can sort the source data by specifying the number for sorted ports. The Integration Service adds an ORDER BY clause to the default SQL query
* **Distinct rows:** You can get distinct rows from the source by choosing the "Select Distinct" property. The Integration Service adds a SELECT DISTINCT statement to the default SQL query.
* **Custom SQL Query:** You can write your own SQL query to do calculations.

## Sorter Transformation

Sorter transformation in [Informatica](http://tekslate.com/tutorials/informatica/) is used to sort the data in an ascending or descending order based on single or multiple keys. This is the type an [active transformation](http://en.wikipedia.org/wiki/Active_and_passive_transformation) which sorts the data either in ascending order or descending order. The port which is participated in sor operation is defined as "key". Sorter transformation is used for "eliminating duplicates" hence the sorted transformation is called as active transformation the sorter functions as "order by clause" in [SQL](http://tekslate.com/sql-server-integration-in-ssis/).

**Router Transformation in Informatica**

Router transformation is an active and connected transformation. It is similar to the filter transformation used to test a condition and filter the data. In a filter transformation, you can specify only one condition and drops the rows that do not satisfy the condition. Where as in a router transformation, you can specify more than one condition and provides the ability for route the data that meet the test condition. Use router transformation if you need to test the same input data on multiple conditions.

**Aggregator Transformation in Informatica**

Aggregator transformation is an active transformation used to perform calculations such as sums, averages, counts on groups of data. The integration service stores the data group and row data in aggregate cache. The Aggregator Transformation provides more advantages than the SQL, you can use conditional clauses to filter rows.

**Union Transformation in Informatica**

Union transformation is an active and connected transformation. It is multi input group transformation used to merge the data from multiple pipelines into a single pipeline. Basically it merges data from multiples sources just like the UNION ALL set operator in SQL. The union transformation does not remove any duplicate rows.

**What are the types of SCD?**

Very simply, there are 6 types of Slowly Changing Dimension that are commonly used, they are as follows:

* Type 0 – Fixed Dimension
  + No changes allowed, dimension never changes
* Type 1 – No History
  + Update record directly, there is no record of historical values, only current state
* Type 2 – Row Versioning
  + Track changes as version records with current flag & active dates and other metadata
* Type 3 – Previous Value column
  + Track change to a specific attribute, add a column to show the previous value, which is updated as further changes occur
* Type 4 – History Table
  + Show current value in dimension table but track all changes in separate table
* Type 6 – Hybrid SCD
  + Utilise techniques from SCD Types 1, 2 and 3 to track change

The basic steps involved in creating a **SCD Type 2** Effective Date mapping are

* Identifying the new records and inserting into the dimension table with Begin\_Date as the Current date (SYSDATE) and End\_Date as NULL.
* Identifying the changed record and inserting into the dimension table with Begin\_Date as the Current date (SYSDATE) and End\_Date as NULL.
* Identify the changed record and update the existing record in dimension table with End\_Date as Curren date.

**Connected Lookup** :

A connected Lookup transformation is part of the mapping data flow. With connected lookups, you can have multiple return values. That is, you can pass multiple values from the same row in the lookup table out of the Lookup transformation.  
Common uses for connected lookups include:  
**1.** Finding a name based on a number ex. Finding a Dname based on deptno  
**2.** Finding a value based on a range of dates  
**3.** Finding a value based on multiple conditions

**Unconnected Lookup** :

An unconnected Lookup transformation exists separate from the data flow in the mapping. You write an expression using the :LKP reference qualifier to call the lookup within another transformation.  
Some common uses for unconnected lookups include:  
**1.** Testing the results of a lookup in an expression  
**2.** Filtering records based on the lookup results  
**3.** Marking records for update based on the result of a lookup (for example, updating slowly changing dimension tables)  
**4.** Calling the same lookup multiple times in one mapping

1.query to write how many consonants and vowels are there in a word

|  |
| --- |
| DECLARE      -- Here variable V is varchar datatype      -- and  flag variable is number datatype      -- variable c is char datatype .      v              VARCHAR2(400) := 'Ramesh is a Geek';      noofvowels     NUMBER := 0;      noofconsonants NUMBER := 0;      C              CHAR;  BEGIN      FOR i IN 1..Length(v) LOOP          c := Substr(v, i, 1);            -- Check if the current character is vowel          IF c IN ( 'A', 'E', 'I', 'O', 'U' )              OR c IN ( 'a', 'e', 'i', 'o', 'u' ) THEN            noofvowels := noofvowels + 1;            -- Else current character is a consonant except space          ELSE            IF c NOT IN ( ' ' ) THEN              noofconsonants := noofconsonants + 1;            END IF;          END IF;      END LOOP;        dbms\_output.Put\_line('No. of Vowels: '                           || noofvowels);        dbms\_output.Put\_line('No. of Consonants: '                           || noofconsonants); |

2.1

1

111

3. file with different courtiers of data

4. Grep the file

**$grep -c "unix" geekfile.txt**

**Options Description**

**-c** : This prints only a count of the lines that match a pattern

**-h :** Display the matched lines, but do not display the filenames.

**-i :** Ignores, case for matching

**-l :** Displays list of a filenames only.

**-n :** Display the matched lines and their line numbers.

**-v :** This prints out all the lines that do not matches the pattern

**-e exp :** Specifies expression with this option. Can use multiple times.

**-f file :** Takes patterns from file, one per line.

**-E :** Treats pattern as an extended regular expression (ERE)

**-w :** Match whole word

**-o :** Print only the matched parts of a matching line,

with each such part on a separate output line.

5. Replace the word in file:  stream editor

**sed -i 's/old-text/new-text/g' input.txt**

1. What are the transformations used

Star and snowflake schemas are similar at heart: a central fact table surrounded by dimension tables. The difference is in the dimensions themselves. In a star schema each logical dimension is denormalized into one table, while in a snowflake, at least some of the dimensions are normalized.

Load Balancing in Informatica is a mechanism which distributes the workloads across the nodes in the gird. Informatica has moved on to the SOA (Service Oriented Architecture) from version 8.6 along with GRID options to distribute the data loads in order to provide the high performance.

One of the key properties which we have to talk about is the Dispatch Mode:

This mode can be set in domain properties in Informatica Administration console. Dispatch mode determines how the load balancer dispatches tasks like Sessions, cmd tasks, email tasks, event wait tasks, worklets and workflows across the GRID. This has three different types of options to choose from.

**Adaptive :**

Dispatch Mode considers the node with most available CPU and allocate the workflows/process to it. The load balancer takes into account the available CPU power when running in Adaptive mode. After the partition group is formed in the DTM, the load balancer sends a message to the Integration Service, to distribute the different partition groups (worker DTM) across the nodes of the grid.

**Round Robin :**

In Round Robin, the Dispatch mode allocates the tasks in round robin fashion.

**Metric Based :**

It checks all resource provision thresholds on each available node and excludes a node if dispatching a task causes the thresholds to be exceeded. The Load Balancer continues to evaluate nodes until it finds a node that can accept the task. This mode prevents overloading nodes when tasks have uneven computing requirements.

For the Load Balancing to work properly:

1) Always primary node has to be started up first and then backup.  
2) Repository service always runs on only one node so it has to be started on primary node first in case of service failures it will be failed over to backup node.  
3) Integration service runs on all the available nodes because it has to Dispatch the tasks on each of the nodes based on the Load balancing set at Domain.

Sq—lookup

Insert or update – lookup surrogate key

Based on certain values we decide to update insert

Dynamic file generation

Tc \_ transaction control

Target file name generated

**Transaction Control** is an active and connected**transformation** which allows us to commit or rollback**transactions** during the execution of the mapping. Commit and rollback operations are of significant importance as it guarantees the availability of data.