**POINT:**

1. ETL is used to populate the DWH
2. The tablename and mapping name will be present in technical documents (HLD, LLD, Technical specification etc)
3. The duration of running the mapping depends upon the data and the logic.
4. TOAD: Tool for Oracle Application Developer. It is a data access tool.In TOAD, F9 (to execute the query), F4 (table structure), Shift+delete(to delete the tables).
5. Transformation is nothing but a function.
6. Transformation will come after the qualifier and before target.
7. Negative cases use \* instead of count(\*).
8. Informatica does not support TRIM function, that the reason we use LTRIM and RTRIM for TRIM.

**ETL TESTING LIFE CYCLE:**

1. Requirement Understanding:

BRD, Technical Specification (HLD and LLD), STM(Source and Target mapping doc)

1. Test Plan Creation (test Case Creation)
2. Test Data Creation – Real Time Data, provided by Client.
3. Test execution and reporting – Scheduling jobs

**ETL TESTING**

One of the Key elements contributing to the success of a Data Warehouse Solution is the ability of the Test team to Plan, Design and Execute a set of effective test that will help to identify multiple issues related to data inconsistency, Data quality failure in the Extract, Transform and Load (ETL) Testing.

* The primary focus of testing should be on the ETL process. This includes validating the data, loading of all required rows, correct execution of all transformations and successful completion of the cleaning operations.
* DWH projects largely concentrate on data and moving data from one stream to another stream to allow reporting.
* The sources of the data can be DB or flat file processed from other application.

**OVERVIEW OF ETL OPERATIONS**

1. **Extraction:** Extract is where data is pulled from source systems. (DB, flat files, XML files, Excel file etc)
2. **Transformation:-**Data transforms into new formats according to business rules. Apply business logic in this stage.

**Note:** Transformations can be the most complex part of data warehousing.

1. **Loading:** Load is where data is located into the data warehouse.

**TEST STRETEGY FOR ETL TESTING**

There will be some standard tests for DWH that should be carried as part of testing for every DWH project. Strategies for testing ETL applications are identified as below:

1. ***DATA COMPLETENESS*** –Ensures that all expected data is loaded.
2. ***DATA TRANSFORMATION*** – Ensures that all data is transformed correctly according to business rules.
3. ***DATA QUALITY*** - Ensure that the ETL application correctly rejects, substitutes default values and reports invalid data.
4. ***INITIAL LOAD/FULL LOAD TESTING***
5. ***INCREMENTAL LOAD TESTING***
6. ***PRESENTATION LAYER (BI/REPORTING***) - Testing BI Reports in Data Warehouse testing.
7. ***INTEGRATION TESTING*** - Ensure that the ETL process function well with other Upstream and Downstream processes.
8. ***REGRESSION TESTING*** - This ensures the new data updates have not broken any existing functionalities.
9. ***UAT TESTING*** - Ensure the solutions meets user current expectations and anticipates their future expectations.

**ABOUT INFORMATICA**

START > PROGRAMS > INFORMATICA POWER CENTER 8.6.0

Current version in the market : 9.0

***Informatics supports three clients/interfaces:***

1. Power Center Designer
2. Power Center Workflow Manager
3. Power Center Workflow Monitor

***Power Center Designer:***

* In designer window, we (developer) will design/develop mappings (programs).
* Mapping: a mapping represents data flow from source to target.
* In designer window, we cannot execute the mapping, its only for desiging.

***Power Center Workflow Manager***

* In this we will run/execute the workflow (mapping), which has been created in designer window.
* In this window, we cannot design the mapping, we will only be able to run the mapping.

***Power Center Workflow Monitor***

* We can monitor/check the status of the workflow (mapping)
* We can also execute the workflow but the workflow should have executed at least once from the workflow Manager.

|  |
| --- |
| **ORACLE** |

|  |
| --- |
| **ODBC**  (It establish the connection) |

|  |
| --- |
| **INFORMATICA** |

**How to run Workflow**

1. Select the required workflow from workflow window.
2. Select the session or workflow and then right click on it.
3. It will display a popup window with multiple window with different options, from there you click on START TASK option.

**How to see the status of the Workflow?**

1. Go to the workflow MONITOR, select the respective session.
2. MONITOR the status column for respective session.

**Informatica session status in workflow window:**

1. RUNNING (EXECUTING/IN-PROGRESS)
2. SUCCEEDED (COMPLETED/PASSED)
3. FAILED
4. STOP
5. ABORT

**How to get the log file?**

Form workflow MONITOR WINDOW select required session than right click on “GET SESSION LOG” option.

**How to see the status of the running session (how many records Pass/Fail)?**

From workflow MONITOR WINDOW, select the required session, click on “get run properties” option.

It will display multiple properties from those properties select the “SOURCE/TARGET STATICES”

**Can we run the workflow in the monitor window?**

Right click > Rerun > Restart task

**STM (SOURCE TARGET MAPPING DOCUMENT)**

It is the heart of DWH. This document is for testing and development team. This is basically developed by developer in consultation with Architect/Moduler, the documents is then review by Architect/moduler and ready for use.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **STM-SOURCE & TARGET MAPPING DOCUMENT** | | | | | | |
| **S.TABLE** | **S.COLM** | **S.DATATYPE** | **BUSSINESS LOGIC** | **T.TABLE** | **T.COLM** | **T.DATATYPE** |
| EMP | EMPNO | NUMBER(5) | INTCAP() | EMP\_DIM | ID | NUMBER(10) |
| EMP | ENAME | VARCHAR2(10) |  | EMP\_DIM | EMPNAME | VARCHAR2(15) |
| EMP | SAL | NUMBER(5) |  | EMP\_DIM | BASIC | NUMBER(25) |
| EMP | JOB | VARCHAR2(10) |  | EMP\_DIM | DEPT | VARCHAR2(30) |

Note:

1. If the business logic is null/blank it means straight pull
2. If the Target data type is empty it mean it can have same data type and size as of source.

**How to cross verify the mapping are correct?**

* Through STM doc

**What are the pre-requisite for testing:**

1. Source and target table exists.
2. Source table should have data.
3. Target table should be empty
4. Mapping should be there.

**DATA COMPLETENESS TEST (DCT)**

It is designed to verify that all the expected data loads into the data warehouse. This includes running detailed tests to verify that all records, all fields and the full contents of each field are loaded.

***STRATEGIES TO CONSIDER INCLUDES:***

1. Record counts must be compared between the source data and target data.

S: SELECT COUNT (\*) FROM CUST;--1000

T: SELECT COUNT (\*) FROM CUST\_DIM; ---1000

1. Comparing record counts between source data, data loaded to the warehouse and rejected records.

S: SELECT COUNT (\*) FROM CUST;--1000

T: SELECT COUNT (\*) FROM CUST\_DIM; ---800

T: SELECT COUNT(\*) FROM CUST\_REJECT; ----2000

1. Comparing unique values of key fields between source data and data loaded to the warehouse.

T: SELECT CID, COUNT (\*) FROM CUST\_DIM GROUP BY CID HAVING COUNT (\*)>1;

1. Column mapping from source to target
2. Populating the full contents of each field to validate that no truncation occurs at any step in the process.

SELECT CID, NAME, CITY FROM OLTP\_SOURCE.CUST

MINUS

SELECT ID, NAME, CITY FROM CUST\_DIM;

**TEST CASE IN DWH**

**Write the test case to “verify the source and target record count”.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Step#** | **Description** | **SQL** | **Expected** |
| 1 | Get Source Record Count | select count(\*) from emp; | The query should be executed successfully and should display source record count. Note: note down the record count |
| 2 | Get Source Target Count | select count(\*) from dept; | The query should be executed successfully and should display source record count. Note: note down the record count |
| 3 | Compare Source and Target Count |  | Both the count should be same |

**Write the test case to “verify the duplicate records in the target DB ‘EMP\_DIM’”**

|  |  |  |  |
| --- | --- | --- | --- |
| **Step#** | **Description** | **SQL** | **Expected** |
| 1 | check the duplicate records in the 'emp\_dim' target table | select emp, count(\*) from emp\_dimgourp by empno having count(\*)>1; | The query should be executed successfully and should return zero rows. |

**Write the test case to “verify source (emp) and target (emp\_dim) table column mapping”.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Step#** | **Description** | **SQL** | **Expected** |
| 1 | Compare sourceand target column mapping | select empno, ename, sal from oltp.emp Minus select empno, ename, sal from emp\_dim; | The query should be executed successfully and should return zero rows. |

**DATA TRANSFORM TEST (DTT)**

It is a process of converting the data, cleansing the data into a required business format. Validating that data is transformed correctly based on business rules. Business rules can be the most complex part of testing in ETL application with significant transformation logic.

Between SOURCE and TARGET: - Test should make sure that the data type of each column of each table is as per the column mapping document. If no specific details are mentioned in STM document about the tables schema etc,then test should make sure on the below :

* The data type of the source column and destination column are same.
* The destination column length is equal to or greater that the source column length.
* Validation should be done that all data specified get extracted.
* Test should include the check to see that the transformation and cleansing process are working correctly. The following types of data transformation activities take place in staging.

1. Data Cleansing
2. Data Merging
3. Data Scrubbing
4. Data Aggregation

**Data Cleansing**

It is a process of changing inconsistence, inaccurate data and removing unwanted data.

The ultimate goal of data cleaning is to improve the organization confidence in their data. List types of data error that needs to be addressed such as:

* Making first character as Capital letter.
* Decoding data
* Rounding the decimal data
* Missing data—Removing records which contains NULL’s
* Data that contains unwanted junk such as an apostrophic or a comma or extra spaces.
* Telephone numbers in the wrong format.

**Data Merging**

It is process of integrating the data from multiple operational sources into a single output pipeline.

**Data Scrubbing**

It is process of deriving new attributes to meet their warehouse requirement.

Example: - Fact Table Column “SOLD\_QTY” ,”SALES\_AMOUNT”

**Data Aggregation**

It is process of calculating the summarized from details data.The following aggregation functions can be used to calculate the aggregates.

SUM (), COUNT (), AVG (), MAX (), MIN ()

* Validate correct processing of ETL generated fields such as surrogate keys.
* Testing slowly changing dimensions (TYPE1,TYPE2,TYPE3 Dimension)
* Setup data scenarios that test referential integrity between tables (Dimension & Fact).
* **Following are the list of transformations available in INFORMATICA.**

1. Aggregator transformation
2. Expression transformation
3. Filter transformation
4. Joiner transformation
5. Router transformation
6. Sequence generator transformation
7. Sorter transformation
8. Look up transformation
9. Update strategic transformation

**What is the difference between aggregator and expression transformation?**

Aggregator transformation is applied on multiple rows whereas expression transformation is applied on single rows.

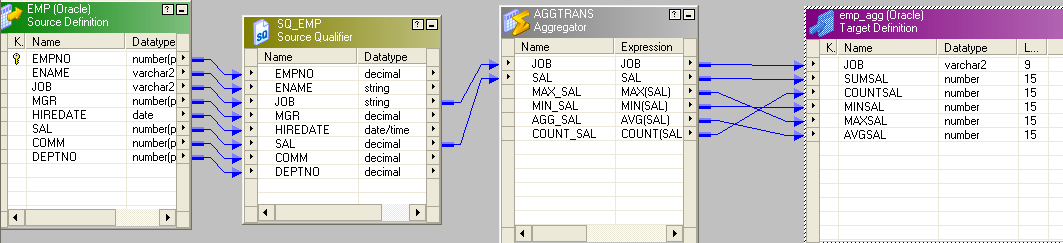
**Aggregator transformation**

This transformation is useful to perform calculations such as average, sum etc (mainly to perform calculations on multiple rows or groups.

Source table: EMP

Target Table: EMP\_AGG

Mapping name: m\_emp\_agg\_trans



|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **T: EMP\_AGG** | | | | | | | | |  | **CORRECT** |
| **JOB** | | **SUMSAL** | | **COUNTSAL** | **MINSAL** | **MAXSAL** | **AVGSAL** | |  | **SUMSAL** |
| ANALYST | | 3000 | | 2 | 3000 | 3000 | 3000 | |  | 6000 |
| CLERK | | 1300 | | 4 | 800 | 1300 | 1037.5 | |  | 4150 |
| MANAGER | | 2450 | | 3 | 2450 | 2975 | 2758.333 | |  | 8275 |
| PRESIDENT | | 5000 | | 1 | 5000 | 5000 | 5000 | |  | 5000 |
| SALESMAN | | 1500 | | 4 | 1250 | 1600 | 1400 | |  | 5600 |
|  | |  | |  |  |  |  | |  |  |
| **S:EMP** | | | | | | | | | | |
| **EMPNO** | **ENAME** | | **JOB** | | **MGR** | **HIREDATE** | | **SAL** | **COMM** | **DEPTNO** |
| 7369 | SMITH | | CLERK | | 7902 | 17-Dec-80 | | 800 |  | 20 |
| 7499 | ALLEN | | SALESMAN | | 7698 | 20-Feb-81 | | 1600 | 300 | 30 |
| 7521 | WARD | | SALESMAN | | 7698 | 22-Feb-81 | | 1250 | 500 | 30 |
| 7566 | JONES | | MANAGER | | 7839 | 2-Apr-81 | | 2975 |  | 20 |
| 7654 | MARTIN | | SALESMAN | | 7698 | 28-Sep-81 | | 1250 | 1400 | 30 |
| 7698 | BLAKE | | MANAGER | | 7839 | 1-May-81 | | 2850 |  | 30 |
| 7782 | CLARK | | MANAGER | | 7839 | 9-Jun-81 | | 2450 |  | 10 |
| 7788 | SCOTT | | ANALYST | | 7566 | 19-Apr-87 | | 3000 |  | 20 |
| 7839 | KING | | PRESIENT | |  | 17-Nov-81 | | 5000 |  | 10 |
| 7844 | TURNER | | SALESMAN | | 7698 | 8-Sep-81 | | 1500 | 0 | 30 |
| 7876 | ADAMS | | CLERK | | 7788 | 23-May-87 | | 1100 |  | 20 |
| 7900 | JAMES | | CLERK | | 7698 | 3-Dec-81 | | 950 |  | 30 |
| 7902 | FORD | | ANALYST | | 7566 | 3-Dec-81 | | 3000 |  | 20 |
| 7934 | MILLER | | CLERK | | 7782 | 28-Jan-82 | | 1300 |  | 10 |

**Requirement/Example:**

Load the count of employee, sum of sal, max sal, min sal and avgsal for job wise?

1. Check the source table has data and target table is empty.
2. Run the mapping and then validate the data with queries

|  |
| --- |
| **Source and target count should be same.**  S: select count(distinct(job)) from oltp\_source.emp --- 5  T: select count(\*) from emp\_agg ---5 |
| **Ensure the target table does not have any duplicate records**  T: select job, count(\*) from emp\_agg group by job having count(\*)>1 --- 0 |
| **Column mapping:**  select distinct(job),sum(sal), count(sal), min(sal), max(sal), avg(sal) from oltp\_source.emp group by job  minus  select job, sumsal, countsal, minsal, maxsal, avgsal from emp\_agg  note:  Actually the query should not return the value/row but it is returning as the sum(sal) column values are wrong. In order to return zero values we need to update the values |

**Create a table in the source but cannot see the values in the target table?**

* The Commit command would have not done.

**Requirement/Example:**

Load the count of employee, sum of sal, max sal, minsal for department wise?

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S:emp\_cust** | | | |  | **T: emp\_cust\_t** | | | | |
| **eno** | **ename** | **esal** | **dno** |  | **dno** | **total\_emp** | **total\_cal** | **highest\_sal** | **lowest\_sal** |
| 1 | santhosh | 3000 | 10 |  | 10 | 4 | 64000 | 50000 | 3000 |
| 2 | manas | 2000 | 20 |  | 20 | 2 | 32000 | 30000 | 2000 |
| 3 | rudra | 50000 | 10 |  |  |  |  |  |  |
| 4 | priya | 30000 | 20 |  |  |  |  |  |  |
| 5 | vamsi | 5000 | 10 |  |  |  |  |  |  |
| 6 | chethan | 6000 | 10 |  |  |  |  |  |  |

|  |
| --- |
| **Source and target count should be same.**  select count(distinct(dno)) from oltp\_source.emp\_cust ---2  select count(dno) from emp\_cust\_t ---2 |
| **Ensure the target table does not have any duplicate records**  select dno, count(\*) from emp\_cust\_t group by dno having count(\*) > 1 --- 0 |
| **Column mapping:**  select dno, count(eno), sum(esal),max(esal), min(esal) from oltp\_source.emp\_cust group by dno  minus  select dno, total\_emp, total\_cal, highest\_sal, lowest\_sal from emp\_cust\_t |

**Requirement/Example:**

Calculate the total of daily sales

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S:Transaction** | | |  | **T:total\_sales** | |
| **orderno** | **price** | **tdate** |  | **tdate** | **total\_sale** |
| 2 | 1000 | 1/1/2014 |  | 1/1/2014 | 6500 |
| 3 | 2500 | 1/1/2014 |  | 2/1/2014 | 13500 |
| 4 | 3000 | 1/1/2014 |  |  |  |
| 5 | 2500 | 2/1/2014 |  |  |  |
| 6 | 3000 | 2/1/2014 |  |  |  |
| 7 | 8000 | 2/1/2014 |  |  |  |

|  |
| --- |
| **Source and target count should be same.**  select count(distinct(tdate)) from oltp\_source.transaction ---2  select count(tdate) from total\_sales ---2 |
| **Ensure the target table does not have any duplicate records**  select tdate, count(\*) from total\_sales group by tdate having count(\*) >1 ---0 |
| **Column mapping:**  select tdate, sum(price) from oltp\_source.transaction group by tdate  minus  select tdate, total\_sale from total\_sales |

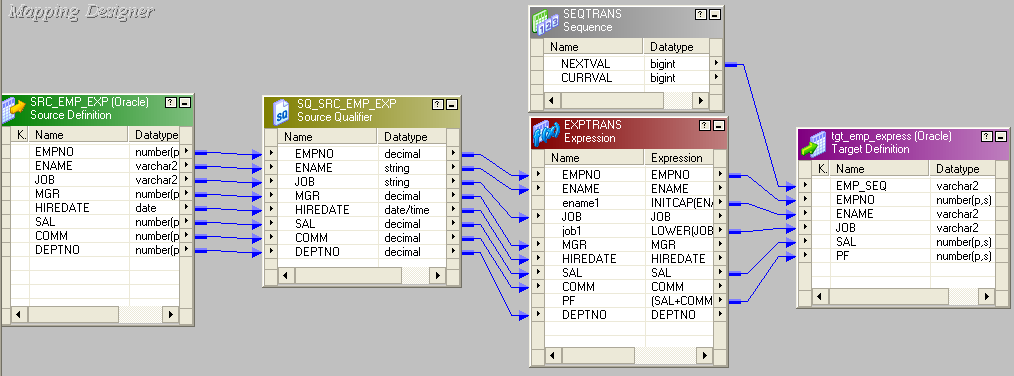
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S:EMP1** | | | | | | | |
| **EMPNO** | **ENAME** | **JOB** | **MGR** | **HIREDATE** | **SAL** | **COMM** | **DEPTNO** |
| 7369 | SMITH | CLERK | 7902 | 17-Dec-80 | 800 |  | 20 |
| 7499 | ALLEN | SALESMAN | 7698 | 20-Feb-81 | 1600 | 300 | 30 |
| 7521 | WARD | SALESMAN | 7698 | 22-Feb-81 | 1250 | 500 | 30 |
| 7566 | JONES | MANAGER | 7839 | 2-Apr-81 | 2975 |  | 20 |
| 7654 | MARTIN | SALESMAN | 7698 | 28-Sep-81 | 1250 | 1400 | 30 |
| 7698 | BLAKE | MANAGER | 7839 | 1-May-81 | 2850 |  | 30 |
| 7782 | CLARK | MANAGER | 7839 | 9-Jun-81 | 2450 |  | 10 |
| 7788 | SCOTT | ANALYST | 7566 | 19-Apr-87 | 3000 |  | 20 |
| 7839 | KING | PRESIENT |  | 17-Nov-81 | 5000 |  | 10 |
| 7844 | TURNER | SALESMAN | 7698 | 8-Sep-81 | 1500 | 0 | 30 |
| 7876 | ADAMS | CLERK | 7788 | 23-May-87 | 1100 |  | 20 |
| 7900 | JAMES | CLERK | 7698 | 3-Dec-81 | 950 |  | 30 |
| 7902 | FORD | ANALYST | 7566 | 3-Dec-81 | 3000 |  | 20 |
| 7934 | MILLER | CLERK | 7782 | 28-Jan-82 | 1300 |  | 10 |
| 8000 | PARWINDER | ANALYST | 8001 | 28-Jan-82 | 4500 |  | 50 |

|  |  |  |
| --- | --- | --- |
| **S: DEPT** | | |
| **DEPTNO** | **DNAME** | **LOC** |
| 10 | Accounting | New York |
| 20 | Research | Dallas |
| 30 | Sales | Chicago |
| 40 | Operations | Boston |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **T: EMP\_AGG\_TARGET** | | | | | |
| **DEPTNO** | **DNAME** | **TOTAL\_EMP** | **TOTAL\_SAL** | **MAXSAL** | **MINSAL** |
| 10 | ACCOUNTING | 3 | 8750 | 5000 | 1300 |
| 20 | RESEARCH | 5 | 10875 | 3000 | 800 |
| 30 | SALES | 6 | 11600 | 2850 | 950 |

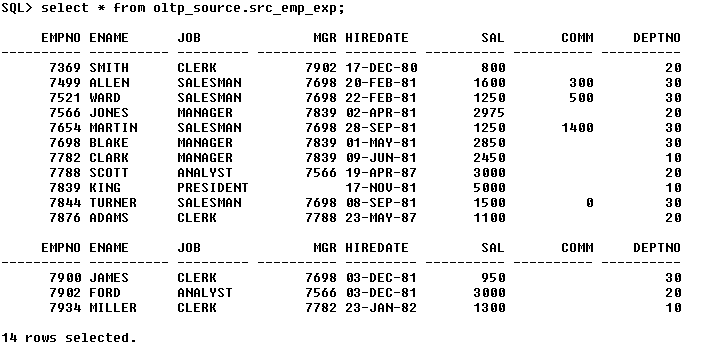
|  |
| --- |
| **Source and target count should be same.**  select count(distinct(deptno)) from oltp\_source.emp1 --- 4 (incorrect)  select d.dname, count(distinct(e.deptno))from oltp\_source.emp1 e, oltp\_source.dept d where e.deptno=d.deptno group by dname ---will display the dname with distinct count  select count(distinct(e.deptno)) from oltp\_source.emp1 e, oltp\_source.dept d where e.deptno=d.deptno ---3  select count(deptno) from emp\_agg\_target; ---3 |
| **Ensure the target table does not have any duplicate records**  select deptno, count(\*) from emp\_agg\_target group by deptno having count(\*) > 1 --- 0 |
| **Column mapping:**  select e.deptno, d.dname, count(empno), sum(sal+nvl(comm,0)), max(sal), min(sal) from oltp\_source.emp1 e, oltp\_source.dept d where e.deptno=d.deptno group by e.deptno, d.dname  minus  select deptno, dname, total\_emp, total\_sal, max\_sal, min\_sal from emp\_agg\_target |

**EXPRESSION TRANSFORMATION**

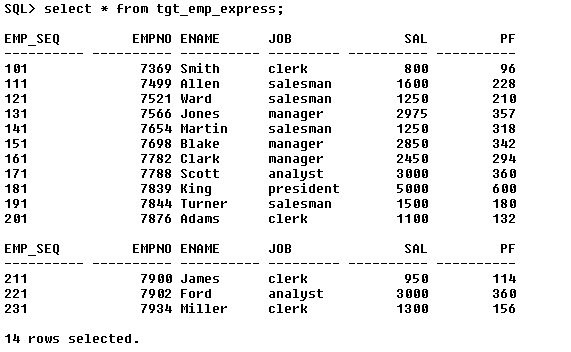
****

Mapping name: m\_emp\_express\_trans

**S:SRC\_EMP\_EXP**



**T: tgt\_emp\_express**



|  |
| --- |
| **Source and target count should be same.**  select count(\*) from oltp\_source.src\_emp\_exp ---14  select count(\*) from tgt\_emp\_express ---14 |
| **Column mapping:**  select empno, ename from oltp\_source.src\_emp\_exp  minus  select empno, ename from tgt\_emp\_express ----14  This will display 14 records as the ename at the source are in caps and in target the first letter is caps and rest small so use this:  select empno, trim(initcap(ename)) from oltp\_source.src\_emp\_exp  minus  select empno, ename from tgt\_emp\_express ---0 |
| **Ensure the target table does not have any duplicate records**  select empno, count(\*) from tgt\_emp\_express group by empno having count(\*)>1 ---0 |

**Requirement:** To calculate the discount on each product.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S:Product** | | |  | **T:Product\_Discount** | | | |
| **pid** | **pdesc** | **pprice** |  | **pid** | **pdesc** | **pprice** | **discount** |
| 1001 | TV | 50000 |  | 1001 | TV | 50000 | 5000 |
| 1002 | Laptop | 40000 |  | 1002 | Laptop | 40000 | 4000 |
| 1003 | Camera | 20000 |  | 1003 | Camera | 20000 | 2000 |

Business logic: Discount = price\*10/100

|  |
| --- |
| **Source and target count should be same.**  select count(\*) from oltp\_source.product ---3  select count(\*) from product\_discount ---3 |
| **Ensure the target table does not have duplicate records**  select pid, count(\*) from product\_discount group by pid having count(\*) > 1 --- 0 |
| **Column mapping**  select pid, pdesc, pprice, pprice\*10/100 from oltp\_source.product  minus  select pid, pdesc, pprice, discount from product\_discount --- 0 |

**Requirement:** to concatenate first and last name

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S: Concatenate** | | | |  | **T: Concatenate\_t** | | |
| **id** | **fname** | **lname** | **city** |  | **id** | **name** | **city** |
| 1 | parwinder | singh | bng |  | 1 | parwinder singh | bng |
| 2 | shefali | dash | bng |  | 2 | shefali dash | bng |
| 3 | ronnie | singh | bng |  | 3 | ronnie singh | bng |

|  |
| --- |
| **Source and target count should be same.**  select count(\*) from oltp\_source.concatenation ---3  select count(\*) from concatenation\_t ---3 |
| **Column mapping and concatenation of the string**  select id, fname||' '||lname , city from OLTP\_SOURCE.CONCATENATION  MINUS  select id, name, city from concatenation\_t ---0 |
| **Ensure the target table does not have duplicate records**  select id, count(\*) from concatenation\_t group by id having count(\*) > 1 ---0 |
| **Column mapping**  select name, count(\*) from concatenation\_t group by name having count(\*)>1 ---0 |

**Requirement:** update the target employee PF field by calculating the source salary field

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S: emp** | | |  | **T: emp\_t** | | | |
| **eno** | **ename** | **sal** |  | **eno** | **ename** | **sal** | **pf** |
| 1 | A | 5000 |  | 1 | A | 5000 | 600 |
| 2 | B | 6000 |  | 2 | B | 6000 | 720 |
| 3 | C | 2000 |  | 3 | C | 2000 | 240 |

Business logic: PF = SAL\*12/100

|  |
| --- |
| **Source and target count should be same.**  select count(\*) from oltp\_source.emp---3  select count(\*) from emp\_t ---3 |
| **Ensure the target table does not have duplicate records**  select eno, count(\*) from emp\_t group by eno having count(\*) > 1 --- 0 |
| **Column mapping**  select eno, ename, sal, sal\*12/100 from oltp\_source.emp  minus  select eno,ename, sal, pf from emp\_t --- 0 |

**Requirement:** if source field is NULL then update the corresponding target field as -1.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S: Customer** | | | |  | **T: Cusrtomer\_dim** | | | |
| **id** | **fname** | **mobile** | **email** |  | **id** | **fname** | **mobile** | **email** |
| 1 | AA | 12345 | [AA@gmail.com](mailto:AA@gmail.com) |  | 1 | AA | 12345 | [AA@gmail.com](mailto:AA@gmail.com) |
| 2 | BB | 56897 | [BB@gmail.com](mailto:BB@gmail.com) |  | 2 | BB | 56897 | [BB@gmail.com](mailto:BB@gmail.com) |
| 3 | CC |  | [CC@gmail.com](mailto:CC@gmail.com) |  | 3 | CC |  | [CC@gmail.com](mailto:CC@gmail.com) |
| 4 | DD | 26585 | [DD@gmail.com](mailto:DD@gmail.com) |  | 4 | DD | 26585 | [DD@gmail.com](mailto:DD@gmail.com) |

|  |
| --- |
| **Source and target count should be same.**  select count(\*) from oltp\_source.customer ---5  select count(\*) from customer\_dim ---5 |
| **Source NULL value count and target -1 count in Mobile column should be same**  select count(\*) from oltp\_source.customer where mobile is NULL ---2  select count(\*) from customer\_dim where mobile=-1 ---2 |
| **Mobile column should not have Null value in target**  select count(\*) from customer\_dim where mobile is NULL ---0 |
| **Ensure the target table does not have duplicate records**  select id, count(\*) from customer\_dim group by id having count(\*) > 1 ---0 |
| **Column mapping**  select id, name, nvl(mobile,-1),email from oltp\_source.customer  minus  select id, name, mobile, email from customer\_dim ---0 |

**Requirement:** decode the source field data and load into target (BNG, BANGALORE, CHN, CHENNAI)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| S:Cust | | |  | S:Cust\_dim | | |
| ID | NAME | CITY |  | ID | NAME | CITY |
| 1 | RAM | HYD |  | 1 | RAM | HYDERABAD |
| 2 | SIVA | BNG |  | 2 | SIVA | BANGALORE |
| 3 | HARI | HYD |  | 3 | HARI | HYDERABAD |

|  |
| --- |
| **Source and target count should be same.**  select count(\*) from oltp\_source.cust ---3  select count(\*) from cust\_dim ---3 |
| **Get the city wise data**  select city, count(\*) from oltp\_source.cust group by city ---2  select city, count(\*) from cust\_dim group by city ---2 |
| **City not blank**  select \* from cust\_dim where city is NULL ---0 |
| **Target should not have BNG, CHN etc same as source value**  select \* from cust\_dim where city in (select distinct(city) from oltp\_source.cust) ---0 |
| **Ensure the target table does not have duplicate records**  select id, count(\*) from cust\_dim group by id having count(\*) >1 ---0 |
| **Column mapping**  select id, name, decode(city, 'HYD', 'HYDERABAD', 'BNG', 'BANGALORE') city from oltp\_source.Cust  minus  select id, name, city from cust\_dim ---0 |

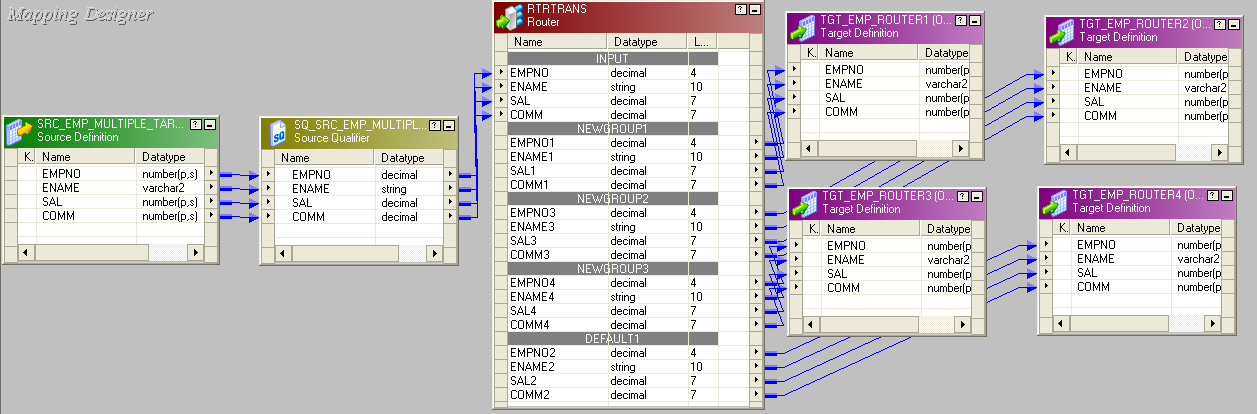
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S: EMP** | | | | | | | |  |
| **EMPNO** | **ENAME** | **JOB** | **MGR** | **HIREDATE** | **SAL** | **COMM** | **DEPTNO** |  |
| 7369 | SMITH | CLERK | 7902 | 12/17/1980 | 800 |  | 20 |  |
| 7499 | ALLEN | SALESMAN | 7698 | 2/20/1981 | 1600 | 300 | 30 |  |
| 7521 | WARD | SALESMAN | 7698 | 2/22/1981 | 1250 | 500 | 30 |  |
| 7566 | JONES | MANAGER | 7839 | 4/2/1981 | 2975 |  | 20 |  |
| 7654 | MARTIN | SALESMAN | 7698 | 9/28/1981 | 1250 | 1400 | 30 |  |
| 7698 | BLAKE | MANAGER | 7839 | 5/1/1981 | 2850 |  | 30 |  |
| 7782 | CLARK | MANAGER | 7839 | 6/9/1981 | 2450 |  | 10 |  |
| 7788 | SCOTT | ANALYST | 7566 | 4/19/1987 | 3000 |  | 20 |  |
| 7839 | KING | PRESIDENT |  | 11/17/1981 | 5000 |  | 10 |  |
| 7844 | TURNER | SALESMAN | 7698 | 9/8/1981 | 1500 | 0 | 30 |  |
| 7876 | ADAMS | CLERK | 7788 | 5/23/1987 | 1100 |  | 20 |  |
| 7900 | JAMES | CLERK | 7698 | 12/3/1981 | 950 |  | 30 |  |
| 7902 | FORD | ANALYST | 7566 | 12/3/1981 | 3000 |  | 20 |  |
| 7934 | MILLER | CLERK | 7782 | 1/23/1982 | 1300 |  | 10 |  |
|  |  |  |  |  |  |  |  |  |
| **T: EMP\_EXPR\_T** | | | | | | | | |
| **EMPNO** | **ENAME** | **JOB** | **MGR** | **HIREDATE** | **SAL** | **COMM** | **DEPTNO** | **PF** |
| 7369 | SMITH | CLERK | 7902 | 12/17/1980 | 800 |  | 20 | 96 |
| 7499 | ALLEN | SALESMAN | 7698 | 2/20/1981 | 1600 | 300 | 30 | 192 |
| 7521 | WARD | SALESMAN | 7698 | 2/22/1981 | 1250 | 500 | 30 | 150 |
| 7566 | JONES | MGR | 7839 | 4/2/1981 | 2975 |  | 20 | 357 |
| 7654 | MARTIN | SALESMAN | 7698 | 9/28/1981 | 1250 | 1400 | 30 | 150 |
| 7698 | BLAKE | MGR | 7839 | 5/1/1981 | 2850 |  | 30 | 342 |
| 7782 | CLARK | MGR | 7839 | 6/9/1981 | 2450 |  | 10 | 294 |
| 7788 | SCOTT | ANALYST | 7566 | 4/19/1987 | 3000 |  | 20 | 360 |
| 7839 | KING | PRESIDENT |  | 11/17/1981 | 5000 |  | 10 | 600 |
| 7844 | TURNER | SALESMAN | 7698 | 9/8/1981 | 1500 | 0 | 30 | 180 |
| 7876 | ADAMS | CLERK | 7788 | 5/23/1987 | 1100 |  | 20 | 132 |
| 7900 | JAMES | CLERK | 7698 | 12/3/1981 | 950 |  | 30 | 114 |
| 7902 | FORD | ANALYST | 7566 | 12/3/1981 | 3000 |  | 20 | 360 |
| 7934 | MILLER | CLERK | 7782 | 1/23/1982 | 1300 |  | 10 | 156 |

|  |
| --- |
| select count(\*) from oltp\_source.emp ---14 select count(\*) from emp\_expr\_t ---14 |
| select empno, ename from oltp\_source.emp minus select empno, ename from emp\_expr\_t ---0 |
| select count(\*) from oltp\_source.emp where comm is null or comm=0---11 select count(\*) from emp\_expr\_t where comm is null or comm=0---11 |
| select \* from emp\_expr\_t where comm is null --- 10 |
| select empno,nvl(comm,0) comm from oltp\_source.emp minus select empno, nvl(comm,0) from emp\_expr\_t ---0 |
| select count(\*) from oltp\_source.emp where job='MANAGER' ---3 select count(\*) from emp\_expr\_t where job='MGR' ---3 select \* from emp\_expr\_t where job='MANAGER' ---0 |
| select empno, decode(job, 'MANAGER', 'MGR',job) JOB from oltp\_source.emp minus select empno, job from emp\_expr\_t ---0 |
| select empno, sal\*12/100 from oltp\_source.emp minus select empno, pf from emp\_expr\_t ---0 |
| select \* from emp\_expr\_t where pf is null ---0 |
| select empno, count(\*) from emp\_expr\_t group by empno having count(\*) >1 ---0 |
| select empno, ename,decode (job, 'MANAGER', 'MGR', JOB) JOB,mgr, hiredate, sal,comm, deptno, sal\*12/100 from oltp\_source.emp minus select empno, ename,job, mgr, hiredate, sal,comm, deptno, pf from emp\_expr\_t --- 0 |

**ROUTER TRANSFORMATION**

This is similar to filter transformation. The only difference is:

- filter transformation drop the data that do not meet the condition whereas router transformation has an option to capture the data that do not meet the condition. It is useful to test multiple condition.



|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S: SRC\_EMP\_MULTIPLE\_TARGET** | | | |  |  | **T:TGT\_EMP\_ROUTER1** | | | |
| **EMPNO** | **ENAME** | **SAL** | **COMM** |  |  | **EMPNO** | **ENAME** | **SAL** | **COMM** |
| 7369 | SMITH | 800 |  |  |  | 7369 | SMITH | 800 |  |
| 7499 | ALLEN | 1600 | 300 |  |  | 7900 | JAMES | 950 |  |
| 7521 | WARD | 1250 | 500 |  |  |  |  |  |  |
| 7566 | JONES | 2975 |  |  |  | **T:TGT\_EMP\_ROUTER2** | | | |
| 7654 | MARTIN | 1250 | 1400 |  |  | **EMPNO** | **ENAME** | **SAL** | **COMM** |
| 7698 | BLAKE | 2850 |  |  |  | 7499 | ALLEN | 1600 | 300 |
| 7782 | CLARK | 2450 |  |  |  | 7521 | WARD | 1250 | 500 |
| 7788 | SCOTT | 3000 |  |  |  | 7654 | MARTIN | 1250 | 1400 |
| 7839 | KING | 5000 |  |  |  | 7844 | TURNER | 1500 | 0 |
| 7844 | TURNER | 1500 | 0 |  |  | 7876 | ADAMS | 1100 |  |
| 7876 | ADAMS | 1100 |  |  |  | 7934 | MILLER | 1300 |  |
| 7900 | JAMES | 950 |  |  |  |  |  |  |  |
| 7902 | FORD | 3000 |  |  |  |  |  |  |  |
| 7934 | MILLER | 1300 |  |  |  | **T:TGT\_EMP\_ROUTER3** | | | |
|  |  |  |  |  |  | **EMPNO** | **ENAME** | **SAL** | **COMM** |
| **BUSINESS RULES** | | | | |  | 7566 | JONES | 2975 |  |
| TGT\_EMP\_ROUTER1 | | SAL>=500 AND SAL<1000 | | |  | 7698 | BLAKE | 2850 |  |
| TGT\_EMP\_ROUTER2 | | SAL>=1000 AND SAL<=2000 | | |  | 7782 | CLARK | 2450 |  |
| TGT\_EMP\_ROUTER3 | | SAL>=2000 AND SAL<=3000 | | |  | 7788 | SCOTT | 3000 |  |
| TGT\_EMP\_ROUTER4 | | default | | |  | 7902 | FORD | 3000 |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  | **T:TGT\_EMP\_ROUTER4** | | | |
|  |  |  |  |  |  | **EMPNO** | **ENAME** | **SAL** | **COMM** |
|  |  |  |  |  |  | 7839 | KING | 5000 |  |

|  |
| --- |
| **Source and target count should be same.** select count(\*) from oltp\_source.src\_emp\_multiple\_targets where sal>=500 and sal<1000---2 select count(\*) from tgt\_emp\_router1 ---2   select count(\*) from oltp\_source.src\_emp\_multiple\_targets where sal>=1000 and sal<=2000 ---6 select count(\*) from tgt\_emp\_router2 ---6  select count(\*) from oltp\_source.src\_emp\_multiple\_targets where sal>=2000 and sal<=3000 ---5 select count(\*) from tgt\_emp\_router3 ---5  select count(\*) from oltp\_source.src\_emp\_multiple\_targets where sal>3000 ---1 select count(\*) from tgt\_emp\_router4 ---1 |
| **Checking the sal is not null in any of the tables:** select \* from oltp\_source.src\_emp\_multiple\_targets where sal is null ---0 select \* from tgt\_emp\_router1 where sal is null union all select \* from tgt\_emp\_router2 where sal is null union all  select \* from tgt\_emp\_router3 where sal is null union all select \* from tgt\_emp\_router4 where sal is null ---0 OR select \* from OLTP\_SOURCE.SRC\_EMP\_MULTIPLE\_TARGETS where sal is NULL ---0 select a.sal, b.sal, c.sal from tgt\_emp\_router1 a, tgt\_emp\_router2 b, tgt\_emp\_router3 c where a.sal is NULL AND b.sal is NULL AND c.sal is NULL --- 0 |
| **Comm is not null** select count(\*) from oltp\_source.src\_emp\_multiple\_targets where sal>=500 and sal<1000 AND comm is NULL---2 select count(\*) from tgt\_emp\_router1 where comm is NULL---2  select count(\*) from oltp\_source.src\_emp\_multiple\_targets where sal>=1000 and sal<=2000 AND comm is NULL---2 select count(\*) from tgt\_emp\_router2 where comm is NULL---2  select count(\*) from oltp\_source.src\_emp\_multiple\_targets where sal>=2000 and sal<=3000 AND comm is NULL---5 select count(\*) from tgt\_emp\_router3 where comm is null---5  select count(\*) from oltp\_source.src\_emp\_multiple\_targets where sal>3000 AND comm is NULL---1 select count(\*) from tgt\_emp\_router4 where comm is null ---1 |
| **Duplicates** select empno, count(\*) from tgt\_emp\_router1 group by empno having count(\*) > 1 ---0  select empno, count(\*) from tgt\_emp\_router2 group by empno having count(\*) > 1 ---0 select empno, count(\*) from tgt\_emp\_router3 group by empno having count(\*) > 1 ---0 select empno, count(\*) from tgt\_emp\_router4 group by empno having count(\*) > 1 ---0 |
| **Column Mapping** select empno, ename, sal, comm from OLTP\_SOURCE.SRC\_EMP\_MULTIPLE\_TARGETS where sal>=500 and sal<1000 minus select empno, ename, sal, comm from tgt\_emp\_router1  select empno, ename, sal, comm from OLTP\_SOURCE.SRC\_EMP\_MULTIPLE\_TARGETS where sal>=1000 and sal<2000 minus select empno, ename, sal, comm from tgt\_emp\_router2  select empno, ename, sal, comm from OLTP\_SOURCE.SRC\_EMP\_MULTIPLE\_TARGETS where sal>=2000 and sal<3000 minus select empno, ename, sal, comm from tgt\_emp\_router3  select empno, ename, sal, comm from OLTP\_SOURCE.SRC\_EMP\_MULTIPLE\_TARGETS where sal> 3000 minus select empno, ename, sal, comm from tgt\_emp\_router4 |

**Example:**

- load data into target and reject tables from source as per the business logic

- If salary is zero/null then reject the record else load into target table

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S: Emp** | | | |  | | **T1: Emp\_dim** | | | |
| **eno** | **name** | **sal** | **dno** |  | | **eno** | **ename** | **sal** | **dno** |
| 1 | sooraj | 25000 | 10 |  | | 1 | sooraj | 25000 | 10 |
| 2 | santhosh | 15000 | 10 |  | | 2 | santhosh | 15000 | 10 |
| 3 | chetha |  | 20 |  | | 4 | deepti | 10000 | 20 |
| 4 | deepti | 10000 | 20 |  | | 5 | radhika | 5000 | 10 |
| 5 | radhika | 5000 | 10 |  | | 7 | nirosha | 6000 | 20 |
| 6 | ovi |  | 10 |  | |  |  |  |  |
| 7 | nirosha | 6000 | 20 |  | |  |  |  |  |
|  |  |  |  |  | |  |  |  |  |
| **T2: Emp\_Reject** | | | | | | | |  |  |
| **eno** | **name** | **dno** | **reason** | | **etl\_prc\_dt** | **up\_dt** | **status** |  |  |
| 3 | chethan | 20 | sal is null | | 18-7-14 |  | open |  |  |
| 6 | ovi | 10 | sal is null | | 18-7-14 |  | open |  |  |

There are two types of attributes:

1. Technical attributes: reason. Use for technical purpose
2. Business attributes: eno, enmae. Use for business purpose

The requirement is, if the salary is null then load the records to the T2 and the status should be open.

|  |
| --- |
| S: select count(\*) from oltp\_source.emp; -----7  T: select count(\*) from emp\_t; --------5  T: select count(\*) from emp\_reject; -----2 |
| S: select count(\*) from oltp\_source.emp where sal is null; ----2  T: select count(\*) from emp\_reject; -----2 |
| T: select \* from emp\_t where sal is NULL; |
| T: select \* from emp\_reject where eno in (select eno from emp\_t); |
| T: sekect \* from emp\_t where eno in (select eno from emp\_reject); |
| T: select \* emp\_reject where etl\_prc\_dt<> ’18-7-14’; |
| T: select \* from emp\_reject where reason is NULL; |
| T: select \* from emp\_reject where etl\_prc\_dt is NULL; |
| T: select \* from emp\_reject where status<> ‘open’; |
| T: select \* from emp\_reject where up\_dt is not NULL; |
| T: select eno, count(\*) from emp\_t group by eno having count(\*) > 1; |
| T: select eno, count(\*) from emp\_reject group by eno having count(\*) > 1; |

**Example:** In the above example image the salary for Chethan is updated and the transformation is run. How the table does looks like:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S: Emp** | | | |  | | **T1: Emp\_dim** | | | | |
| **eno** | **name** | **sal** | **dno** |  | | **eno** | **ename** | | **sal** | **dno** |
| 1 | sooraj | 25000 | 10 |  | | 1 | sooraj | | 25000 | 10 |
| 2 | santhosh | 15000 | 10 |  | | 2 | santhosh | | 15000 | 10 |
| 3 | chethan | **7000** | 20 |  | | 4 | deepti | | 10000 | 20 |
| 4 | deepti | 10000 | 20 |  | | 5 | radhika | | 5000 | 10 |
| 5 | radhika | 5000 | 10 |  | | 7 | nirosha | | 6000 | 20 |
| 6 | ovi |  | 10 |  | | **3** | **chethan** | | **7000** | **20** |
| 7 | nirosha | 6000 | 20 |  | |  |  | |  |  |
|  |  |  |  |  | |  |  | |  |  |
| **T2: Emp\_Reject** | | | | | | | | |  |  |
| **eno** | **name** | **dno** | **reason** | | **etl\_prc\_dt** | **up\_dt** | | **status** |  |  |
| **3** | **chethan** | **20** | **sal is null** | | **18-7-14** | **20-7-14** | | **closed** |  |  |
| 6 | ovi | 10 | sal is null | | 18-7-14 |  | | open |  |  |

**Audit table**

|  |  |
| --- | --- |
| Mapping name/table name | Last update |
| M\_mapping | 18-7-14 |
|  |  |

**Exmaple:**

If product price is null then load the data into target and also load into report table. Null update as -1 in target table.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S: Product** | | |  | | **T: Product\_dim** | | |
| **pid** | **desc** | **price** |  | | **pid** | **desc** | **price** |
| 1 | TV | 6000 |  | | 1 | TV | 6000 |
| 2 | LAPTOP |  |  | | 2 | LAPTOP | -1 |
| 3 | MOBILE | 4000 |  | | 3 | MOBILE | 4000 |
| 4 | TAB |  |  | | 4 | TAB | -1 |
| 5 | CHAIR | 6000 |  | | 5 | CHAIR | 6000 |
| 6 | TABLE | 2000 |  | | 6 | TABLE | 2000 |
| 7 | BED |  |  | | 7 | BED | -1 |
|  |  |  |  | |  |  |  |
| **T2: Product\_Report** | | | | | | |  |
| **pid** | **desc** | **reason** | | **etl\_prc\_dt** | **up\_dt** | **status** |  |
| 2 | LAPTOP | sal is null | | 18-7-14 |  | open |  |
| 4 | TAB | sal is null | | 18-7-14 |  | open |  |
| 7 | BED | sal is null | | 18-7-14 |  | open |  |

**What is the approach of transformation in router transformation?**

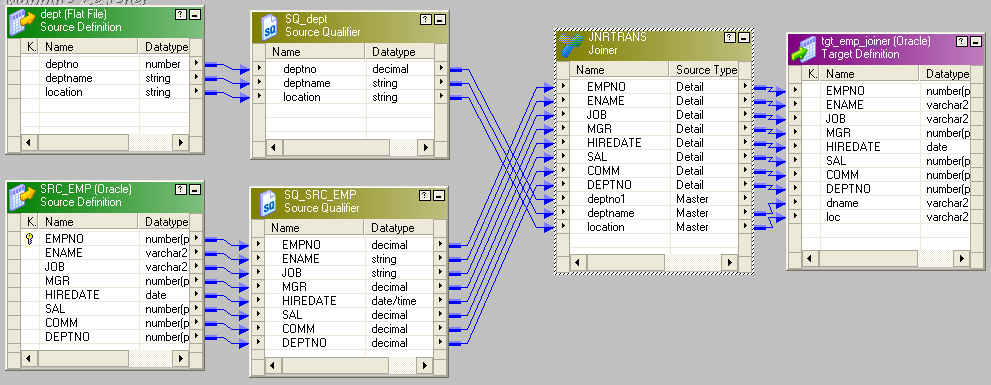
* Based on the criteria, it will check and load the records.

|  |
| --- |
| T: select count(\*) from emp\_router\_t; ----4  T: select count(\*) from emp\_reject\_target; ---10  One single statement:  Select (select count(\*) from emp\_router\_t) + ( select count(\*) from emp\_reject\_target) target\_count from dual;  The purpose of using the dual is so that it returns the single value. |
| Select count(\*) from oltp\_source.emp where comm is NULL; -----10  Select count(\*) fromemp\_reject\_target;---10 |
| Select \* from emp\_router\_t where comm is NULL; |
| Select \* from emp\_reject\_target where empno in (select empno from emp\_router\_t);  Select \* from emp\_router\_t where empno in (select empno from emp\_reject\_target); |
| MINUS |
| MINUS ULTA |
| DUPLICATE |

**JOINER TRANSFORMATION**

* Belongs to flat file testing
* It is used to join two sources coming from two different locations or from same location.Example:

1. To join a flat file and relational source (table) and then load into target.
2. To join two flat files and then load into target



In the above mapping the EMP information is getting loaded from the mapping and the DEPT information needs to be fetched from flat file.

Note: the structure of the table be created before uploading the file.

**How to upload flat file:**

1. Create the structure of the table.

Create table joiner (id number (3), name varchar2 (10), locvarchar2 (10));

1. TOAD > DATABASE > IMPORT > IMPORT TABLE DATA > SELECT THE TABLE “JOINER” > NEXT > browse the file > NEXT > select the delimiter “tab” > NEXT” > select the first row as “2” (so that the column name should not be imported) > Preview > NEXT > Execute

**How to insert the data in excel without separator?**

1. Select ‘NAME|LOC’ from dual;
2. Copy paste the value to the excel
3. In excel, Data > text to column
4. Select delimiter radio button > NEXT > Other : | > NEXT > in the data preview, Shift+click on the grid > FINISH

**TOAD:**

**1. BOOKMARK 🡪SERVICES🡪C:\ROOT FOLDER**

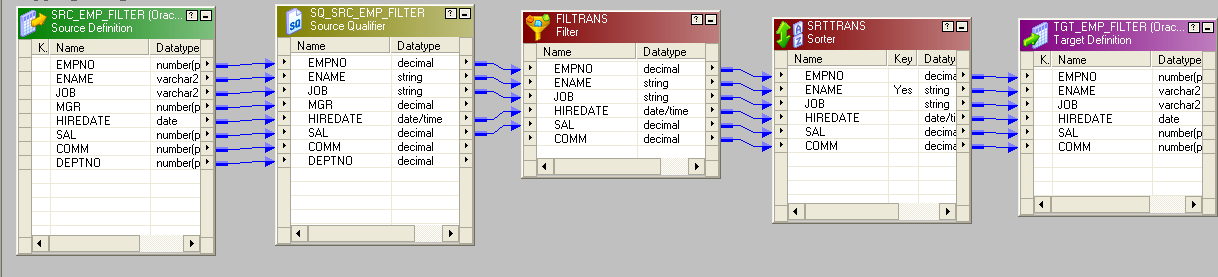
**2.** UTILITES🡪COMPARE FILES **(RED🡪ISSUE;BLACK🡪NO ISSUE; BLUE 🡪 ANY SPACES)**

**NOTE: For COLUMN NAME IF RED ALSO NO ISSUES**

**FILTER TRANSFORMATION**

FILTER Transformation can be used to filter rows in a mapping that meet the condition.

Example: Load only selected data from Source to Target.



**Example:** To know all the employees who are working in department 10.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| EMP\_S | | | |  | EMP\_T | | | |
| **EMPNO** | **NAME** | **SAL** | **DNO** |  | **EMPNO** | **NAME** | **SAL** | **DNO** |
| 1 | SHIVA | 5000 | 10 |  | 1 | SHIVA | 5000 | 10 |
| 2 | SANTOSH | 6000 | 20 |  | 2 | HARI | 4000 | 10 |
| 3 | HARI | 4000 | 10 |  | 3 | RAMESH | 7000 | 10 |
| 4 | KUMAR | 7000 | 20 |  |  |  |  |  |
| 5 | LOKESH | 7000 | 20 |  |  |  |  |  |
| 6 | RAMESH | 7000 | 10 |  |  |  |  |  |

|  |
| --- |
| **Match the record count from source and target tables**  **S:** SELECT COUNT (\*) FROM OLTP\_SOURCE.EMP\_S WHERE DNO=10;  **T:** SELECT COUNT (\*) FROM EMP\_T; |
| **Check the duplicate records at target table**  **T:** SELECT DNO, COUNT (\*) FROM EMP\_T GROUP BY DNO HAVING COUNT (\*)>1; |
| **3. Column Mapping**  SELECT DNO, EMPNO, NAME, SAL FROM OLTP\_SOURCE.EMP\_S WHERE DNO=10  MINUS  SELECT EMPNNO, NAME, SAL, DNO FROM EMP\_T; |

**Example:** Load the products into DB which falls in the rate category btn 500 and 1000

|  |  |  |
| --- | --- | --- |
| PROD\_S | | |
| **PID** | **DESC** | **PRICE** |
| 1 | TV | 1000 |
| 2 | LAPTOP | 400 |
| 3 | CAMERA | 600 |
| 4 | PENDRIVE | 100 |
| 5 | WATCH | 700 |
| 6 | IPAD | 500 |
| 7 | TAB | 1200 |

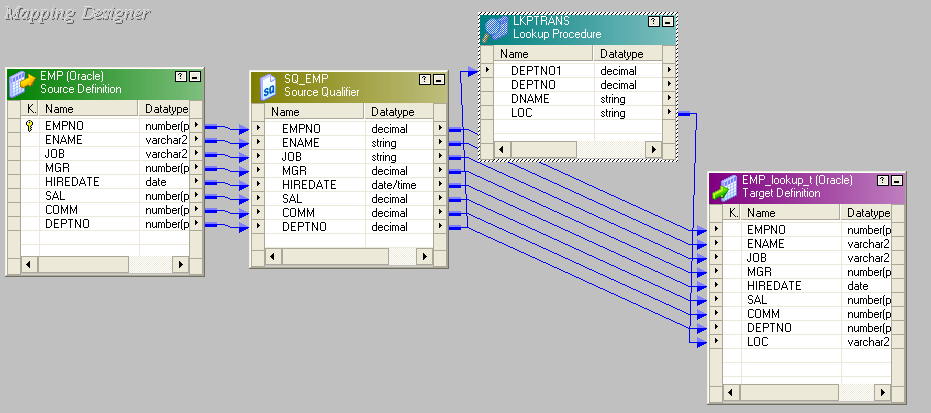
|  |  |  |
| --- | --- | --- |
| PROD\_T | | |
| **PID** | **DESC** | **PRICE** |
| 1 | TV | 1000 |
| 3 | CAMERA | 600 |
| 5 | WATCH | 700 |
| 6 | IPAD | 500 |

|  |
| --- |
| select count(\*) from oltp\_source.PROD\_S where price between 500 and 1000;  select count(\*) from PROD\_T |
| select \* from PROD\_T where price not between 500 and 1000; |
| Select pid, count(\*) from PROD\_T group by pid having count(\*) > 1; |
| Select pid,, desc, price from oltp\_source.PROD\_S where price between 500 and 1000  MINUS  Select pid, desc, price from PROD\_T |

**LOOKUP TRANSFORMATION**

It is used to lookup data in database table. Lookup definition can be imported either from source or target tables. Lookup can be either on source/target.

Example: load the required data into target from the lookup tables using source or target data.



**Example:** Load the city field into target by using source ZIP Code.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | S.CUST |  |  |  | T.CITY\_LOOKUP(***LOOKUP TRANSFORAMTION***) |  |  |  | T.CUST\_DIM |  |  |  |
|  | **CID** | **NAME** | **ZIPCODE** |  | **ZIPCODE** | **CITY** |  | **CID** | **NAME** | **ZIPCODE** | **CITY** |  |
|  | 1 | RAM | 080 |  | 080 | BNG |  | 1 | RAM | 080 | BNG |  |
|  | 2 | SIVA | 044 |  | 040 | HYD |  | 2 | SIVA | 044 | CHN |  |
|  | 3 | HARI | 040 |  | 044 | CHN |  |  |  |  |  |  |
|  | 4 | KUMAR | 080 |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |

T.CITY\_LOOKUP: the data is entered by administrator. Lookup table can be either at source or table. Lookup table are updated very rarely.

If no lookup the records would be rejected.

In lookup table what type of queries do you write?

* Joins

|  |
| --- |
| S: SELECT COUNT (\*) FROM OLTP.CUST C, CITY\_LOOKUP L WHERE C.ZIPCODE=L.ZIPCODE; ---4  T:SELECT COUNT (\*) FROM CUST\_DIM ---4 |
| Select id, count(\*) from CUST\_DIM group by id having count(\*) > 1; ---0 |
| SELECT \* FROM CUST\_DIM WHERE CITY IS NULL; |
| SELECT \* FROM CUST\_DIM WHERE ZIPCODE IN (SELECT ZIPCODE FROM CITY\_LOOKUP) |
| Select c.id, c.name, c.zipcode, ;.city from oltp\_source.Cust c, city\_lookup l where c.zipcode=l.zipcode  MINUS  Select id, name, zipcode, city from cust\_dim |

**Example:** load the dept name into target table using source dept no.

**SLOWLY CHANGING DIMENSIONS (SCD)**

SCD are also known as CDC (Change Data Capture)

**MOD-**It is a function used to generate a checksum values and it is faster.

**TYPE-1:** THE NEW RECORD REPLACES THE ORIGINAL RECORD. NO TRACE OF THE OLD RECORD EXISTS. IT’S LIKE OVERRIDING THE VALUES.

|  |  |  |  |
| --- | --- | --- | --- |
| EMP\_S | | |  |
| **EMPNO** | **NAME** | **LOC** |  |
| 1 | Suraj | BNG |  |
| 2 | Netaji | ~~BNG~~ | HYD |
|  |  |  |  |
| 3 | Deepti | BNG |  |
| 4 | Padmaja | BNG |  |

|  |
| --- |
| **ETL** |
|

|  |  |  |  |
| --- | --- | --- | --- |
| EMP\_T | | |  |
| **EMPNO** | **NAME** | **LOC** |  |
| 1 | Suraj | BNG |  |
| ~~2~~ | ~~Netaji~~ | ~~BNG~~ | REMOVED FROM TARGET TABLE |
| 2 | Netaji | HYD | ABOVE RECORD IS OVERWRITEEN |

|  |  |  |  |
| --- | --- | --- | --- |
| EMP\_CHK\_SUM | |  |  |
| **EMP** | **CHK\_SUMVALUE** |  |  |
| 1 | 3333 | BNG | (MOD(EMP.LOC)) |
| ~~2~~ | ~~2222~~ | ~~BNG~~ | It will overerite the system generated number |
| 2 | 4444 | HYD |  |

**How to ensure the values/data is loaded to the target table:**

Check the value in the EMP\_CHK\_SUM table, example, if the value 3333 is not getting displayed mean the data is not loaded to the target table.

**How does MOD function works**

Suppose Netaji moved from BNG to CHN.

Mod (eno, loc) 🡺 this function resides in the EMP\_CHK\_SUM table as a business logic.

The mod will check 2|2222, since it is there no override, now 2|4444 it will override the function

**Approach/scenarios of testing:**

1. CHK\_SUM and EMP\_T table should be empty.
2. Source record count = Target record count
3. Target count and CHK\_SUM count should match
4. CHK\_SUM value should not be NULL
5. CHK\_SUM value should not be duplicate
6. Eno also not NULL and duplicate
7. Capture the first location and then after the value. Check sum value is updated
8. Before going for ETL mechanism take backup of source table and create the backup of CHK\_SUM and compare with the new target table.

**Advantages:**

1. This is the easiest way to handle the SCD problem, since there is no need to keep record of old information

**Disadvatnages:**

1. All history is lost.

**When to used Type 1 SCD:**

When it is not necessary for DWH to keep track of historical changes.

**TYPE-2:** A NEW RECORD IS ADDED INTO THE CUSTOMER DIMENSION TABLE .THEREFORE, THE CUSTOMER IS TREATED ESSENTIALLY AS TWO PEOPLE. APPENDING NEW RECORDS.

|  |  |  |  |
| --- | --- | --- | --- |
| S.CUST | | | |
| **CID** | **NAME** | **CITY** | **UPD-Date** |
| 1001 | Rudra | BNG | 01-01-2014 |
| 1002 | Pari | BNG | 01-01-2014 |
| 1003 | Naveen | HYD | 01-01-2014 |

|  |
| --- |
| **ETL** |
|

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| T.CUST\_HISTORY | | | | | | | | |
| T | B | B | B | B | B | T | T | T |
| **SKEY** | **CID** | **CNAME** | **CITY** | **EFFECTIVE S.DATE** | **EFFECTIVE E.DATE** | **ACC\_ROW**  **\_FLAG** | **VERSION** | **ETL PROC DATE** |
| 5001 | 1001 | Rudra | BNG | 01-01-2014 | 31-12-2099 | A | 1 | 01-01-2014 |
| 5002 | **1002** | **Pari** | **BNG** | **01-01-2014** | **14-04-2014** | **H** | **1** | **01-01-2014** |
| 5003 | 1003 | Naveen | HYD | 01-01-2014 | 31-12-2099 | A | 1 | 01-01-2014 |
| 5004 | **1002** | **Pari** | **CHN** | **15-04-2014** | **31-12-2099** | **A** | **2** | **15-04-2014** |

Where

COLUMN:

T – Technical Attribute

B-Business Attribute COMPOSITE PRIMARY KEY

**FLAG:**

|  |  |
| --- | --- |
| H-HISTORY | N-NOT STAYING |
| A-ACTIVE | Y-STAYING |

**Validations:**

1. ETL PROCESS DATE should not NULL

2. EFFECTIVE S.DATE->HIRE DATE

3. New record version is updated

4. Data column is mandatory should not be NULL

5. Suppose one customer entry is 10 times in the table, I want to check all changes with version.

6. If multiple customer then count record active FLAG will be A , remaining should be H.

Note: For Primary Key we are borrowing from system is called surrogate key-ETL mechanism.

7. Duplicate record count

SELECT CID,EFFECTIVE S.DATE, EFFECTIVE T.DATE,COUNT(\*) FROM CUST\_HISTORY GROUP BY CID,EFFECTIVE S.DATE, EFFECTIVE T.DATE HAVING COUNT(\*)>1;

|  |  |  |
| --- | --- | --- |
|  | **DIFFENCE BETWEEN PRIMARY KEY(PK) AND SURROGATE KEY (SK)** | |
|  | **PRIMARY KEY(PK)** | **SURROGATE KEY (SK)** |
| 1 | PK values are entered by user | SK is generated by system (ETL Mechanism) |
| 2 | PK are used to identify each and every record unique record in SOURCE systems | SK are used to identify each and every record unique record in TARGET systems |
| 3 | PK values can be numeric or alphanumeric | SK values are always numeric |
| 4 | PK are belong to Business Data | SK are not belong to Business Data |

**Advantages:**

* This allows us to accurately keep all historical information.

**Disadvantages:**

* This will cause the size of the table to grow fast, in cases where the number of row for the table is very high to start with, storage and performance can become a concern.

**When to use:**

Should be used when it is necessary for the DWH to track historical changes.

**TYPE-3:** THE ORIGINAL RECORD IS MODIFIED TO REFLECT THE CHANGE. UPDATING PARTIAL HISTORY CURRENT AND PREVIOUS.

S.EMP

|  |  |  |
| --- | --- | --- |
| **EMPNO** | **NAME** | **CITY** |
| 1 | Rudra | BNG |
| 2 | Pari | BNG |

|  |
| --- |
| **ETL** |
|

T.EMP

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| T | B | B | B | B | T | T |
| **SKEY** | **EMPNO** | **CNAME** | **CURENT** | **PREVIOUS** | **EFFECTIVE DATE** | **FLAG** |
| 1001 | 1 | Rudra | BNG |  | 15-04-2014 | I |
| 1002 | **2** | **Pari** | **CHN** | **BNG** | **27-07-2014** | **U** |

**FLAG:**

|  |
| --- |
| I/N-INSERT/NEW |
| U-UPDATE |
| D-DELETE |

Suppose 1000 records are there in source table

900 new records are loaded to TARGET table – I

100 updated records are loaded to TARGET table - U

SELECT \* FROM EMP WHERE FLAG IS NOT I

Surrogate key is PRIMARY key for TARGET table

**Advantages:**

* This does not increase the size of the table, since new information is updated.
* This allows us to keep some part of history

**Disadvantage:**

* Will not able to keep all the history wherein the attribute is changed more than once.

**Usage:**

It is used rarely used in actual practice

**When to use:**

Should be used when it is necessary for the DWH to track the historical changes and when such changes will only occur for a finite number of time.

**What is the difference between Type 1 and 3 SCD?**

* Type 1 is overwriting where as type 3 is updating.
* In real time mostly Type 1 and 2 are used.

**How we come to know whether any update happened on the table?**

There is column called as Flag.

**UPDATE STRATEGY TRANSFORMATION**

It is also one type of transformation which is used to handle update data. Basically used in SCD Type 1.

S: eno, ename, dno, loc, update\_dt

T: SGT\_Key, eno, ename, dno, loc, update\_dt

1. Update source value from Bangalore to Mumbai.
2. Again run the mapping and check if it getting updated means overwritted.

**SORTER TRANSFORMATION**

It is an active and connected transformation used to sort the data. The data can be sorted in ascending or descending order by specifying the sort key. A port can be defined as sort key.

**ACTIVE & PASSIVE TRANSFORMATION**

**Active Transformation**

An Active transformation can change the number of rows that pass through it from source to target. It eliminates rows that do not meet the condition in transformation.  
eg: filter transformation passes the rows from the source to target that which meet the filter condition.

Active Transformations list

* Source Qualifier Transformation
* Aggregator Transformation
* Filter Transformation
* Joiner Transformation
* Router Transformation

**Passive Transformation**

A Passive transformation does not change the number of rows that pass through it i.e., it passes all rows through the transformation.  
eg: an expression transformation that performs the calculation on the data and passes all the rows from source to target.

Passive Transformations list -

* Expression Transformation
* Sequence Generate Transformation

**Tell me one mapping in which you have used multiple transformation and in the order?**

IN SCD’s

Source 🡺 Sequence 🡺 Filter 🡺 Expresssion 🡺Aggregator 🡺 router

Source 🡺 Sequence 🡺 Filter 🡺 joiner 🡺 Expresssion

**INITIAL/FULL LOAD TESTING/FIRST LOAD TESTING**

* First time loading source data into DWH is called Initial load or Full Load.
* Initial /Full load completely erase (truncate) the existing data from all the target tables and reload with fresh data.
* Need to test the data in Dimension/Fact table’s matches what is shown in the source system and Business logic.

**INCREMENTAL LOAD TESTING**

* Second time or modified source data loading into DWH is called as Incremental load or DELTA load.
* Incremental load apply ongoing changes to one or more tables based on a predefined schedule.
* Once a data component is updated in the source system, in the corresponding Dimension/Fact table show the updated value needs to be test.

**DATA QUALITY:**

* We say we have achieved the quality when we successfully fulfilled customer’s requirements.
* Since in Data Warehouse Testing, the test executionmoves around the data, so it is important to achieve the degree of excellence for the data and for that we do the data validation for both the data extracted from the source and then getting loaded into the target table.

**DATA QUALITY** is defined as “how the ETL system handles data rejection, substitution, correction”. To ensure success in testing data quality, includes as many data scenarios as possible.

Note: Typically, data quality roles are defined during design.

**Data Quality Rules:**

* Reject the record if a certain decimal fields has nonnumeric data.
* Substitute Dept Name if a Dept No is given.
* Substitute -1 or 0 if source data has null or empty or spaces.
* Validate and correct the STATE field if necessary based on the ZIP Code.
* Compare product code values in lookup table and if there is no match load anyway but report to users.

Depending on the data quality rules of the application begin tested, scenario to test might include null key values, duplicate records in target data and involved data types in fields (Example Alphabetic characters in a decimal field)

Review the detailed test scenarios with business users and technical designers to ensure that all are on the same page.

Data quality rules applied to the data will usually be invisible to the users once the application is in production ; users will only see what’s loaded to the database .For this reason, it is important to ensure that what is done with invalid data is reported to the business users.

**SYSTEM INTEGRATION TESTING (SIT)**

* It shows how the application fits into the overall flow of all UPSTREAM (DWH) & DOWNSTREAM (DATAMART) system.
* When creating integration test scenarios, consider how overall process can break and focus on touch points between applications rather that with in one application.
* Consider how process failures of each step would be handled and how data would be handled and how data would be recovered if necessary.
* Most issues during integration testing are either data related to or resulting from false assumptions about the design of other applications.
* Therefore, it is important to integration testing with production-like data. Real production data is ideal, but depending on the contents of the data, there could be a privacy or security concern that requires certain fields to be randomized before using it in a test environment.
* Run the overall process from end to end in the same order and with the dependencies as in production.
* Testing carried out by the test teams to make sure Source data corresponds to the final data as appeared in reporting.
* Included in this testing, should be data in the dimension and FACT tables in the DWH as well as the data presented through the presentation layer (BI Reports).

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | **OLTP** |  |  |  | **DWH** |  |  |  | **DATA MART** |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | S.EMP |  |  |  | T.EMP |  |  |  | T.EMPLOYEE |  |  |  |
|  | **ENO** | **NAME** | **SAL** |  | **ENO** | **NAME** | **SAL** |  | **ENO** | **NAME** | **TOT\_SAL** |  |
|  | 1 | Anand | 5000 |  | 1 | Anand | 5000 |  | 1 | Anand | 5100 |  |
|  | 2 | Sudir | 7000 |  | 2 | Sudir | 7000 |  | 2 | Sudir | 7300 |  |
|  | 3 | Kartik | 7500 |  | 3 | Kartik | 7500 |  | 3 | Kartik | 7700 |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | S.EMPBOUNS |  |  |  | EMP.BOUNS\_DIV | |  |  |  |  |  |  |
|  | **ENO** | **BOUNS** |  |  | **ENO** | **BOUNS** |  |  |  |  |  |  |
|  | 1 | 100 |  |  | 1 | 100 |  |  |  |  |  |  |
|  | 2 | 300 |  |  | 2 | 300 |  |  |  |  |  |  |
|  | 3 | 200 |  |  | 3 | 200 |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| W\_EMPLOYEE |  |  | STATUS IN INFORMATICA MONITOR WINDOW |  |  |  |
|  | S.EMP |  | RUNNING | SUCCEEDED | SUCCEEDED | SUCCEEDED |
|  | S.EMPBOUNS |  | WAITING | RUNNING | SUCCEEDED | SUCCEEDED |
|  | T.EMPLOYEE |  | WAITING | WAITING | RUNNING | SUCCEEDED |

**Where would be the mapping dependencies are specified?**

* Technical document

**Regression Testing:**

* Data Warehouse implementation is not a one time job. This requires continual up gradation as the operational data stores changes and also periodic data loads are necessary.
* A well-defined ,baseline regression suit ensure existing functionality remains unchanged each time such a change happens.
* Testcases can also be prioritized as per business risk an importance. This helps in deciding which testcase to be run for each of the new release or changes to the up/down stream application.
* A simple but effective strategy to retest basic functionality is:

1. Take backup of previous version run test result.
2. Result from successful runs of the new code.
3. And compare new test results with previous runs test result.

When doing regression test, it is much quicker to compare results with previous results, than to do an entire data validation again for new code.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| S: Customer | | | |  |  |  | T: Cust\_dim | | |
| id | name | city | mob |  | ETL |  | id | name | city |
|  |  |  |  |  |  |  |  |  |  |

**Strategy for Regression Test**

There is a table “S:Customer” and “T:Cust\_Dim”, the request is that there should be 3 columns in both. Now imagine the client says to add a column “Mob” and perform test.

Instead of completely testing the target. Take the backup of the table before and once the new setup is there then first check/test mob column and then perform a MINUS on the backup and the new data of the three columns, it should return zero if no change.

**UAT (User Acceptance Testing):**

* The main reason for building a Data Warehouse application is to make data available to business users for business decisions.
* Users known the data best and their participation in the testing effort is a key component to the success of a data warehouse implementation.
* UAT typically focus on the data which is loaded to the data warehouse, not the mechanics of how the ETL application works.

**Consider the following strategies for UAT testing:**

* Use data that is from production.
* Users typically find issues once they see the “Real” data.
* UAT test team will plan to take help from system test team/Functional test team to support users during UAT.User will likely have questions about how the data is populated and need to understand details of how the ETL works.
* Functional test team has to discuss with UAT test team, how the end users would require the data loaded during UAT and also discuss how often data will be refreshed.

**What all you test in ETL?**

1. Count
2. Duplicates
3. Column mapping
4. Initial load
5. Incremental load
6. Business logic
7. SCD – Type 1, 2, 3
8. Transformation etc

**What do we doing in Reporting (Cognos)?**

* BRD is different from report
* How to check the Data in the report is coming ok?
* Take the report detail and import it to the excel and from DWH run the query and take the same to excel and compare both the reports
* There are so many users (FIN, MARKETING, HR etc). Login with the HR credential and check whether the user is able to see the details of marketing and fin too. The user should only be able to see the HR details rest should be disabled.

**What is memory leakage?**

* Example, a person went to the room to take a paper, when he went the lights we off, so he switched on the light, took the sheet and came back. He didn’t switch off the light so light keep on, which is like a memory leakage

**What are the categories of projects in ETL?**

* New projects
* Enhancement projects
* Migration projects
* Maintenance/support projects

**FILE COMPARISON**

**APPROACH-1**

**1.** Segregate the columns of Target Table as below:

1. The column which has loaded from Source Table.
2. And the column which has loaded from flat file.

**2.** Write a MINUS query between Source table and Target table the columns which has been loaded from Source table.

**3.** Select the data from the Target table which has loaded from flat file and then export this Target data into a another flat file.

**4.** Then compare the Source flat file and Target flat file. (Which has exported from target data)

**FILE COMPARSION STEPS IN TOAD:**

1. Select UTILITIES MENU from TOAD
2. Click on “Compare File “Option and it will prompt you to select the files.
3. Select the Source file and click on OK.
4. Select the Target file and click on OK.
5. Then it will display a window with the two files data. (SOURCE & TARGET )

Note:

* The Data which has **not matched** will be displayed in “**RED COLOR”**
* The Data which has **matched** will be displayed in “**BLACK COLOR”**

**APPROACH-2**

1. Create a temporary table in the Target Database (Create columns as per requirement)
2. Import flat file data into the temporary table by following the below steps:
3. Select Database Menu
4. Select IMPORT sub menu and then click on import Table Data option
5. Select the table and Schema details from Destination Window.
6. Schema Name
7. Table Type
8. Table Name (Temporary table Name)

And then click on “SHOW DATA” button in the same window and then click on “Next” Button.

1. Select the Source File from “SELECT SOURCE” window and also select the file type (TXT, XLS, ETC..) and then click on “Next” Button.
2. Select the “DELIMITER” (Example-Comma, **TAB, etc**...) option from the next window and then click on “Next” Button.
3. In Next window enter the value as **First Row: 2** and then click on Next Button
4. And then click on “Next” Button, and then click on “Execution” Button.

After above steps is completed with commit then the data will be loaded successfully into the respective table.

**APPROACH-3**

If Source flat file data has loaded into target table by applying some business logic ,how will you validate the data against to the Source flat file including the business logic data (Without importing source flat file data into Target temporary table).

**Steps:-**

1. Export target table data into EXCEL file.
2. If Source flat file is other than EXCEL format then convert the Source File into EXCEL format.
3. Apply the business logic into Source EXCEL file by writing EXCEL formulas.
4. And then compare the Source EXCEL file and Target EXCEL file by using “MACRO” and other EXCEL file comparison options.

**APPROACH-4**

Validating data from two different Databases

*Example:***Source DB:** SQL SERVER and **Target DB:** ORACLE

**Source DB:** SQL SERVER

SELECT EMPNO, NAME, SAL, COMM, SAL\*12/100 PF FROM EMP;

OUTPUT:-COPY TO TXT FILE

**Target DB:** ORACLE

SELECT EMPNO, NAME, SAL, COMM, SAL\*12/100 PF FROM EMP;

OUTPUT:-COPY TO TXT FILE

Then compare the TXT file data by using different file comparison options