Department of Information Technology and Management
ITMD 526 – Data Warehousing

GROUP 4 - PROJECT REPORT

TEAM MEMBERS:

Name	Email ID
Amruta Lele	alele@hawk.iit.edu
Hashmitha Shantharam	hshantharam@hawk.iit.edu
Pooja Kuchani	pkuchani@hawk.iit.edu
Srinath Sunku	ssunku@hawk.iit.edu
Vishnu Pajjuri	vpajjuri1@hawk.iit.edu
Bharath Manjunath	bmanjunath@hawk.iit.edu

AIM:

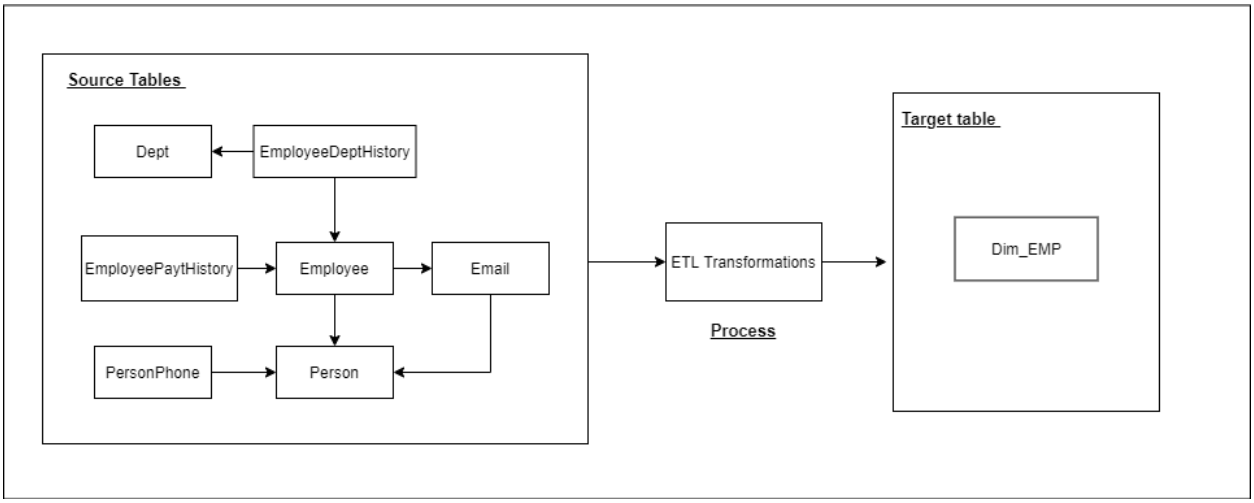
To Perform denormalization of the Adventure Works OLTP dataset to build two-dimension tables as below:

- An Employee dimension.
- A Product dimension.

Step 1:

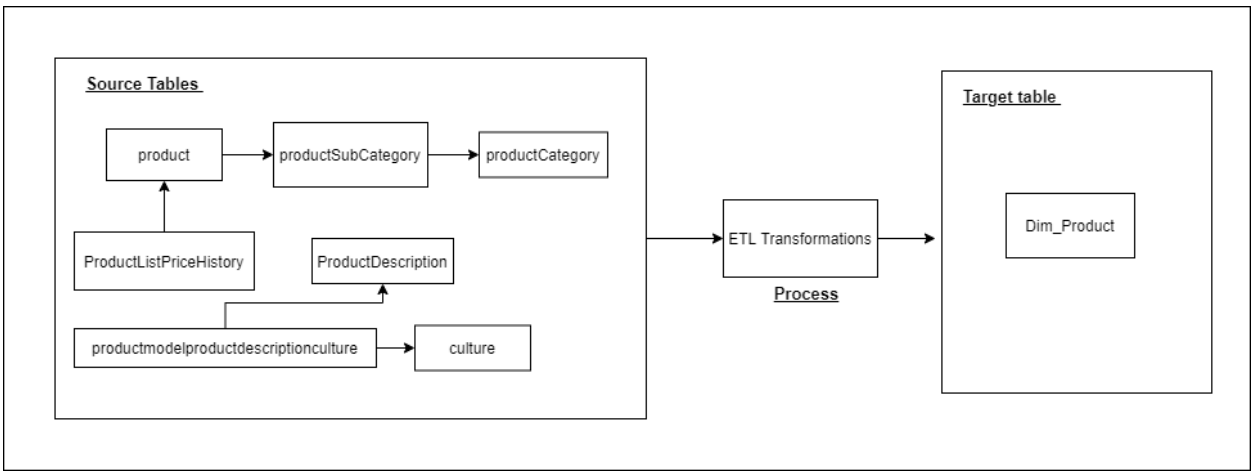
DATA FLOW DIAGRAM:

Employee Dimension:



Department of Information Technology and Management
ITMD 526 – Data Warehousing

Product Dimension:



Step 2:

SOURCE TO TARGET MAPPING:

Employee Dimension:

ETL Package Name	Job.kjb dim_emp.ktr
Target Table	dimemp
Source Tables	department employee personsphone email address person employeedepartmenthistory employeepayhistory
Load Frequency	Whenever a value is being updated
Target Type	SCD Type-2
Mapping Descriptions	FROM t_data a JOIN person p ON a.BusinessEntityID = p.BusinessEntityID LEFT OUTER JOIN t_dept_range b ON a.BusinessEntityID = b.BusinessEntityID AND a.seq >= b.seq_start AND a.seq < b.seq_end LEFT OUTER JOIN t_payr_range c

	ON a.BusinessEntityID = c.BusinessEntityID AND a.seq >= c.seq_start AND a.seq < c.seq_end LEFT OUTER JOIN Emailaddress em ON em.BusinessEntityID = p.BusinessEntityID LEFT OUTER JOIN personphone ph ON ph.BusinessEntityID = p.BusinessEntityID LEFT OUTER JOIN employee e ON e.BusinessEntityID = p.BusinessEntityID
Error Handling	By clearing and disabling the cache in SQL we were able to handle the Serializing-row-to-byte-array error.

TARGET						SOURCE					
Mapping Change Date	Target Table	Target Column	Nullable	PK	Data-type, Length	Source Table	Source Column	Data-type, Length	Expression, Transformation	Default Value	Error Types and Handling
5/5/2021	dimemp	BusinessEntityID FirstName LastName MiddleName Nationalidnumber DepartmentName PayRate jobtitle Hiredate birthdate maritalstatus salariedflag Gender namestyle ModifiedDate emailaddress phonenumber loginid vacationhours sickleavehours employeekey	Except for surrogate key all the keys can be null.	employeekey	1) EmployeeKey - BIGINT AUTO_INCREMENT NOT NULL 2) BusinessEntityid - INT(11) NOT NULL 3) NameStyle, SalariedFlag - TINYINT(1) DEFAULT NULL 4) VacationHours, SickLeaveHours - SMALLINT(6) DEFAULT NULL 5) modifiedDate, HireDate, BirthDate - DATE DEFAULT NULL 6) MaritalStatus, Gender - CHAR(1) DEFAULT NULL 7) FirstName, LastName, MiddleName, Title, LoginID, EmailAddress, DepartmentName, version - VARCHAR(50) DEFAULT NULL	department employee personsphone email address person employeedepartmenthistory employeepayhistory	Person.FirstName person.LastName person.MiddleName person.namestyle employee.Nationalidnumber employee.jobtitle employee.Hiredate employee.birthdate employee.maritalstatus employee.salariedflag employee.Gender employee.loginid employee.vacationhours employee.sickleavehours employeedepartmenthistory. BusinessEntityID t_dept_range.DepartmentName emaimAddress.emailaddress personPhone.phonenumber	BusinessEntityID int(11) FNMAE,LNAME,MNAME varchar(100) Nationalidnumber - Varchar(15) JobTitleVarchar(50) HireDate, birthDate - Date marital Status - char(1) SalariedFlag tinyInt(1) Gender char(1) loginId Varchar(256) VaccationHours smallint sick leave hours smallint	Joins used: Join, Left outer Join	No default value is used	By clearing and disabling the cache in SQL we were able to handle the Serializing-row-to-byte-array error.

Product Dimension:

ETL Package Name	Job.kjb dim_prod.ktr
Target Table	dim_product
Source Tables	product productsubcategory productmodel productdescriptionculture productcategory productdescription culture ProductListPriceHistory
Load Frequency	Whenever a value is being updated
Target Type	SCD Type-2
Mapping Descriptions	FROM t_price_history a JOIN productsubcategory n ON a.ProductSubCategoryID = n.ProductSubCategoryID

	LEFT OUTER JOIN productcategory z ON n.ProductCategoryID = z.ProductCategoryID LEFT OUTER JOIN culture c ON a.CultureID = c.CultureID
Error Handling	By clearing and disabling the cache in SQL we were able to handle the Serializing-row-to-byte-array error.

TARGET						SOURCE				
Mapping Change Da	Target Tabl	Target Column	Nullable	PK	Data-type, Length	Source Table	Source Column	Data-type, Length	Expression, Transformat	Default Valu	Error Types and Handling
5/5/2021	dim_produc	ProductID StartDate EndDate ModifiedDate ListPrice CategoryName SubCategoryName CultureID culture_name Description version productkey	Except for surrogate key all the keys can be null.	productkey	1)ProductID - INT DEFAULT NULL 2) productkey - INT NOT NULL AUTO_INCREMENT 3) StartDate, EndDate, ModifiedDate - DATE DEFAULT NULL 4) ListPrice, CategoryName, SubCategoryName, CultureID, culture_name, Description, version - VARCHAR(256) DEFAULT NULL	product productsubcategory productmodelproductdescriptionculture productcategory productdescription culture ProductListPriceHistory	Product.ProductID product.ProductSubCategoryID productCategory.ProductCategoryID product.modifiedDate ProductListPriceHistory.startdate ProductListPriceHistory.ListPrice ProductCategory.Categoryname ProductsubCategory.SubCategoryname ProductDescription.Description productmodelproductdescriptionculture.CultureId	ProductID - INT ProductSubCategoryID - INT ProductCategoryID - INT modifiedDate - Timestamp startdate - DateTime enddate - DateTime ListPrice Categoryname - VARCHAR(100) SubCategoryname - VARCHAR(100) Description - VARCHAR(400) CultureId - CHAR(6) name - varchar(100)	Joins used: Join, Left out	No default values used	By clearing and disabling the cache in SQL we were able to handle the Serializing-row-to-byte-array error.

Step 3:

ETL PROCESS EXECUTION:

SCD Type 2:

- A Type 2 SCD retains the full history of values. When the value of a chosen attribute changes, the current record is closed.
- A new record is created with the changed data values and this new record becomes the current record.

Kettle Transformation:

Employee Transformation:

SQL Query to create the Dimension Table:

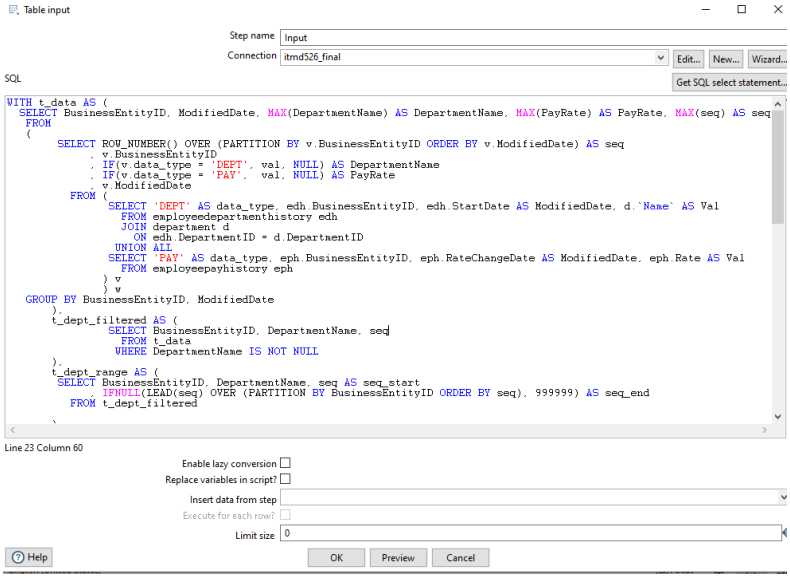
```
CREATE TABLE `dimemp` (
  `EmployeeKey` BIGINT AUTO_INCREMENT NOT NULL,
  `BusinessEntityid` INT(11) NOT NULL,
  `FirstName` VARCHAR(50) DEFAULT NULL,
  `LastName` VARCHAR(50) DEFAULT NULL,
  `MiddleName` VARCHAR(50) DEFAULT NULL,
```

```
 `NameStyle` TINYINT(1) DEFAULT NULL,
 `Title` VARCHAR(50) DEFAULT NULL,
 `HireDate` DATE DEFAULT NULL,
 `BirthDate` DATE DEFAULT NULL,
 `LoginID` VARCHAR(256) DEFAULT NULL,
 `EmailAddress` VARCHAR(50) DEFAULT NULL,
 `Phone` VARCHAR(25) DEFAULT NULL,
 `MaritalStatus` CHAR(1) DEFAULT NULL,
 `SalariedFlag` TINYINT(1) DEFAULT NULL,
 `Gender` CHAR(1) DEFAULT NULL,
 `VacationHours` SMALLINT(6) DEFAULT NULL,
 `SickLeaveHours` SMALLINT(6) DEFAULT NULL,
 `DepartmentName` VARCHAR(50) DEFAULT NULL,
`version` VARCHAR(50) DEFAULT NULL,
 `modifiedDate` DATE DEFAULT NULL,
 PRIMARY KEY (`EmployeeKey`)
) ;
```

Source Input:

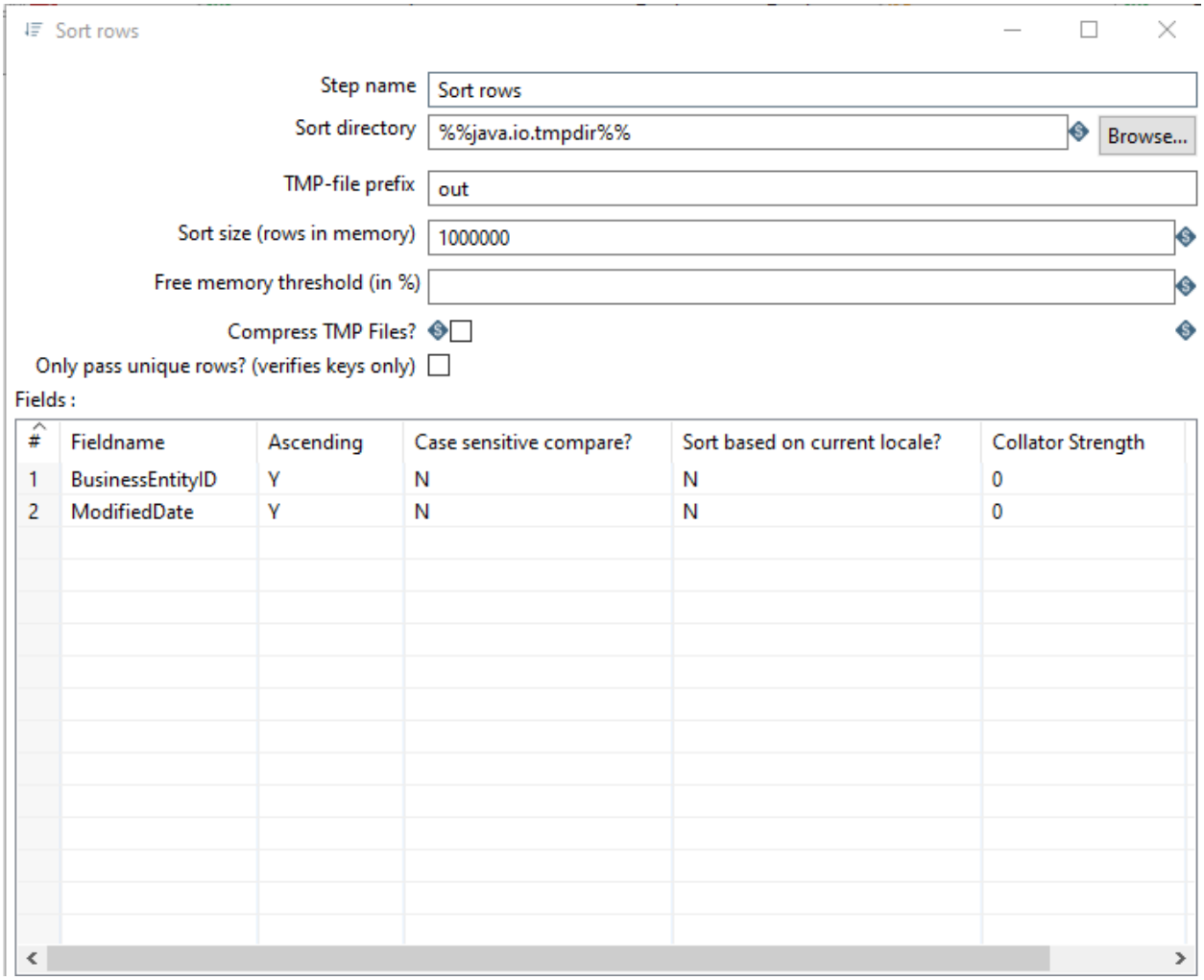
Source table combines multiple tables such as department, employee, personsphone, email, address, person, employeedepartmenthistory, employeepayhistory to select the columns needed to populate the employee dimension table.

Department of Information Technology and Management
ITMD 526 – Data Warehousing



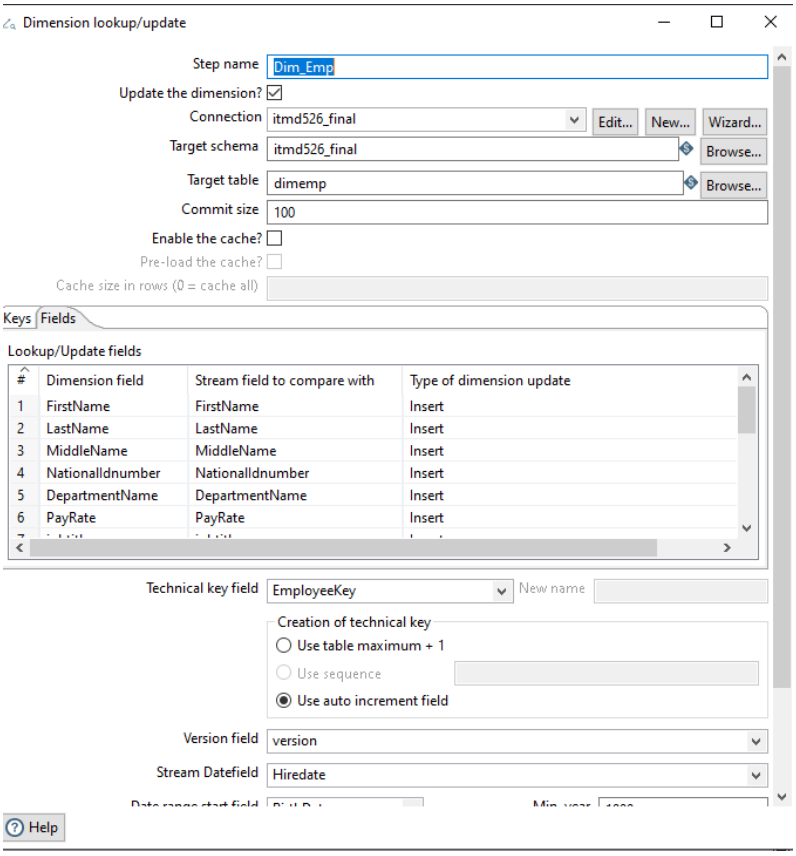

```
WITH t_data AS (
  SELECT BusinessEntityID, ModifiedDate, MAX(DepartmentName) AS DepartmentName, MAX(PayRate) AS PayRate, MAX(seq) AS seq
   FROM
   (
        SELECT ROW_NUMBER() OVER (PARTITION BY v.BusinessEntityID ORDER BY v.ModifiedDate) AS seq
             , v.BusinessEntityID
             , IF(v.data_type = 'DEPT', val, NULL) AS DepartmentName
             , IF(v.data_type = 'PAY',  val, NULL) AS PayRate
             , v.ModifiedDate
          FROM (
                SELECT 'DEPT' AS data_type, edh.BusinessEntityID, edh.StartDate AS ModifiedDate, d.`Name` AS Val
                  FROM employeedepartmenthistory edh
                  JOIN department d
                    ON edh.DepartmentID = d.DepartmentID
                 UNION ALL
                SELECT 'PAY' AS data_type, eph.BusinessEntityID, eph.RateChangeDate AS ModifiedDate, eph.Rate AS Val
                  FROM employeepayhistory eph
               ) v
               ) v
   GROUP BY BusinessEntityID, ModifiedDate
       ),
       t_dept_filtered AS (
                SELECT BusinessEntityID, DepartmentName, seq
                  FROM t_data
                 WHERE DepartmentName IS NOT NULL
       ),
       t_dept_range AS (
        SELECT BusinessEntityID, DepartmentName, seq AS seq_start
             , IFNULL(LEAD(seq) OVER (PARTITION BY BusinessEntityID ORDER BY seq), 999999) AS seq_end
          FROM t_dept_filtered
       `
```

Sort rows:

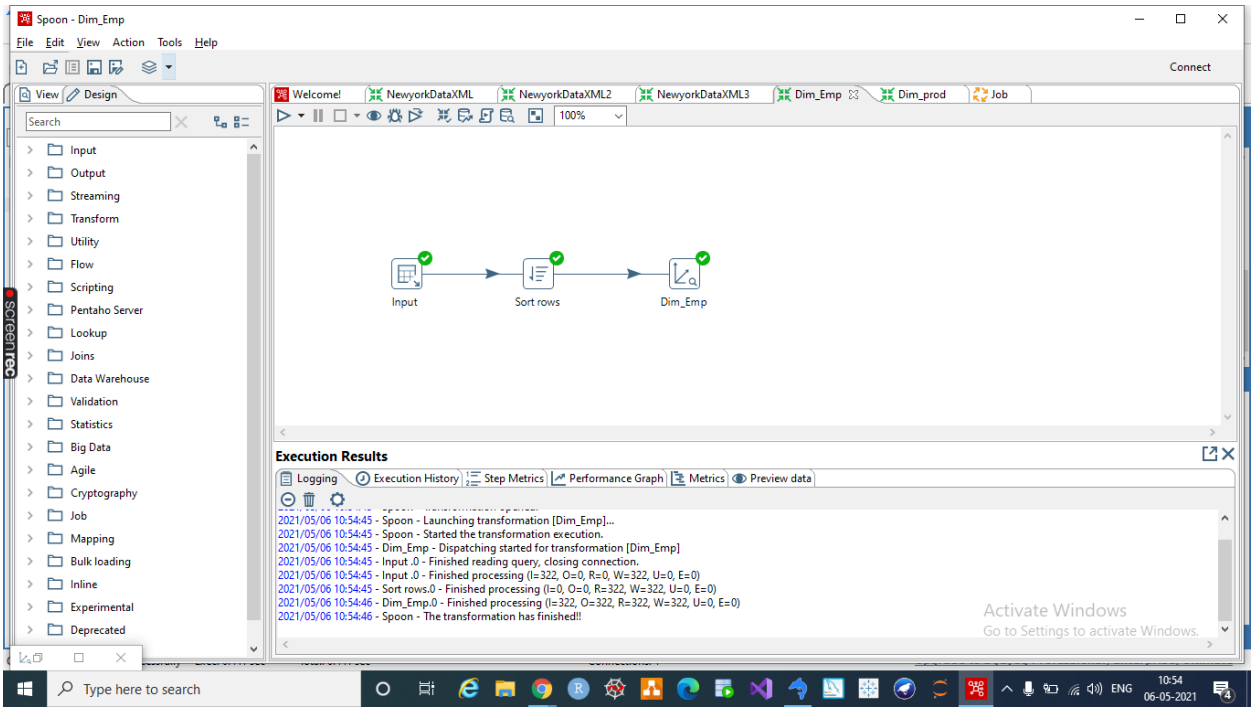


Target output Table:

The selected values from source table are being sorted and mapped to its corresponding target table. The table is being truncated and new value is being inserted on every update.

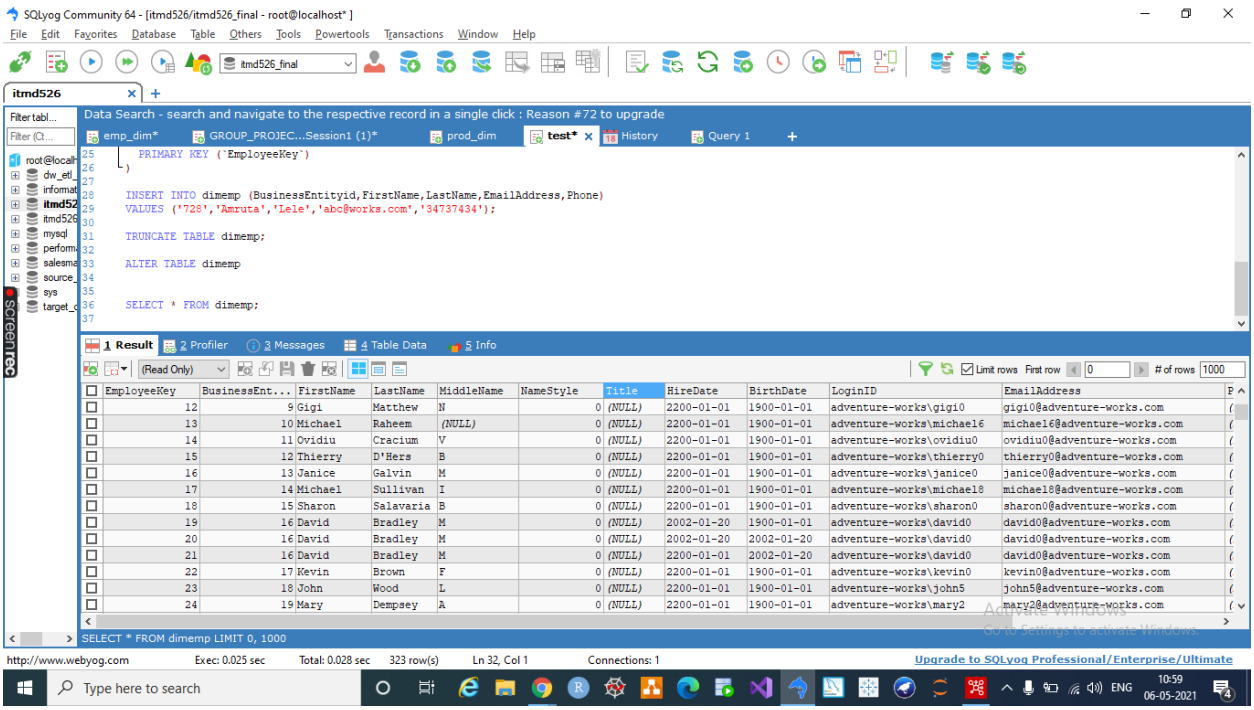

Dimension lookup/update
Step name
Dim_Emp
Update the dimension?
Connection
itmd526_final
Edit...
New...
Wizard...
Target schema
itmd526_final
Browse...
Target table
dimemp
Browse...
Commit size
100
Enable the cache?
Pre-load the cache?
Cache size in rows (0 = cache all)
Keys
Fields
Lookup/Update fields
| Dimension field | Stream field to compare with | Type of dimension update
1 | FirstName | FirstName | Insert
2 | LastName | LastName | Insert
3 | MiddleName | MiddleName | Insert
4 | Nationalldnumber | Nationalldnumber | Insert
5 | DepartmentName | DepartmentName | Insert
6 | PayRate | PayRate | Insert
Technical key field
EmployeeKey
New name
Creation of technical key
Use table maximum + 1
Use sequence
Use auto increment field
Version field
version
Stream Datefield
Hiredate
Help

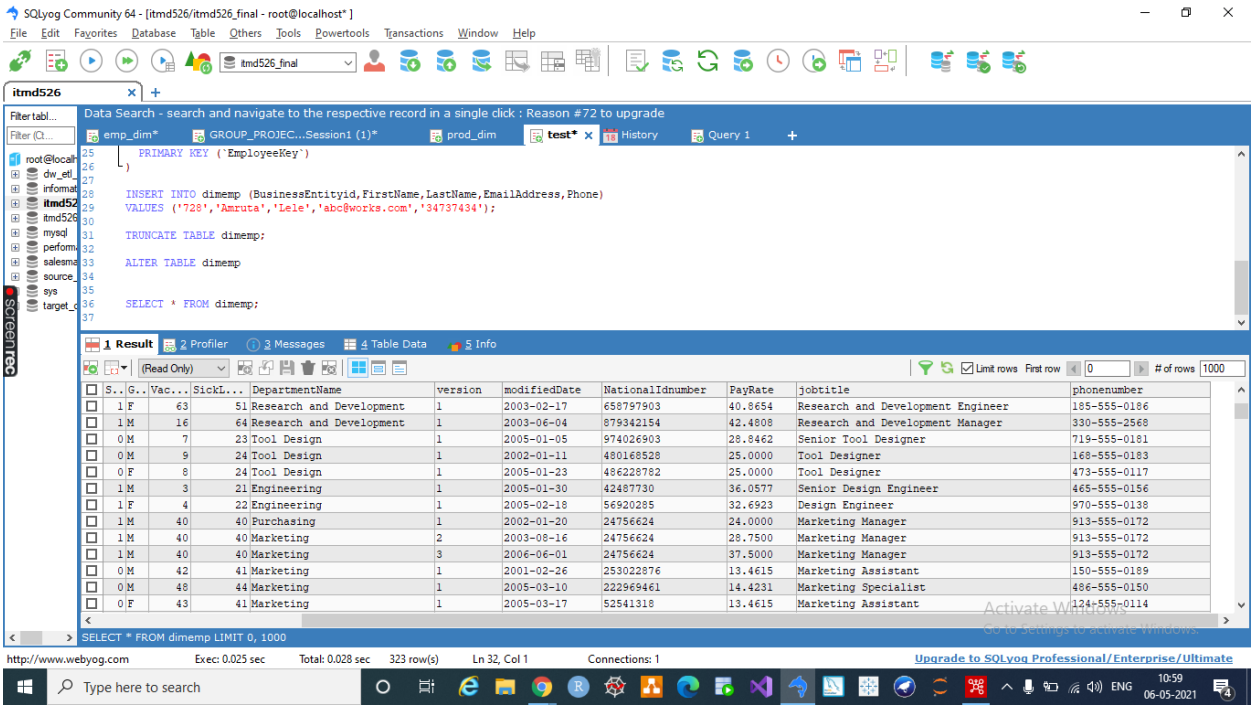
Result:


Spoon - Dim_Emp
File Edit View Action Tools Help
Connect
View
Design
Welcome!
NewyorkDataXML
NewyorkDataXML2
NewyorkDataXML3
Dim_Emp
Dim_prod
Job
Input
Output
Streaming
Transform
Utility
Flow
Scripting
Pentaho Server
Lookup
Joins
Data Warehouse
Validation
Statistics
Big Data
Agile
Cryptography
Job
Mapping
Bulk loading
Inline
Experimental
Deprecated
Input
Sort rows
Dim_Emp
Execution Results
Logging
Execution History
Step Metrics
Performance Graph
Metrics
Preview data
2021/05/06 10:54:45 - Spoon - Launching transformation [Dim_Emp]...
2021/05/06 10:54:45 - Spoon - Started the transformation execution.
2021/05/06 10:54:45 - Dim_Emp - Dispatching started for transformation [Dim_Emp]
2021/05/06 10:54:45 - Input.0 - Finished reading query, closing connection.
2021/05/06 10:54:45 - Input.0 - Finished processing (I=322, O=0, R=0, W=322, U=0, E=0)
2021/05/06 10:54:45 - Sort rows.0 - Finished processing (I=0, O=0, R=322, W=322, U=0, E=0)
2021/05/06 10:54:46 - Dim_Emp.0 - Finished processing (I=322, O=322, R=322, W=322, U=0, E=0)
2021/05/06 10:54:46 - Spoon - The transformation has finished!!
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
10:54
06-05-2021

Department of Information Technology and Management
ITMD 526 – Data Warehousing

Snapshot of Table after running kettle Transformation:





Product Transformation:

SQL Query to create the Dimension Table:

CREATE TABLE `dim_product` (

```
`ProductID` INT DEFAULT NULL,
`StartDate` DATE DEFAULT NULL,
`EndDate` DATE DEFAULT NULL,
`ModifiedDate` DATE DEFAULT NULL,
`ListPrice` VARCHAR(256) DEFAULT NULL,
`CategoryName` VARCHAR(256) DEFAULT NULL,
`SubCategoryName` VARCHAR(256) DEFAULT NULL,
`CultureID` VARCHAR(256) DEFAULT NULL,
`culture_name` VARCHAR(256) DEFAULT NULL,
`Description` VARCHAR(256) DEFAULT NULL,
`version` VARCHAR(256) DEFAULT NULL,
`productkey` INT NOT NULL AUTO_INCREMENT,
PRIMARY KEY (`productkey`));
```

Source Input:

Source table combines multipe tables such as product, Productsubcategory, productmodel, productdescriptionculture, productcategory, productdescription, culture, ProductListPriceHistory to select the columns needed to populate the product dimension table.

Department of Information Technology and Management
ITMD 526 – Data Warehousing

Table input

Step name: Input

Connection: itmd526_final Edit... New... Wizard...

SQL Get SQL select statement...

```
WITH t_data AS (
  SELECT BusinessEntityID, ModifiedDate, MAX(DepartmentName) AS DepartmentName, MAX(PayRate) AS PayRate, MAX(seq) AS seq
   FROM
   (
        SELECT ROW_NUMBER() OVER (PARTITION BY v.BusinessEntityID ORDER BY v.ModifiedDate) AS seq
             , v.BusinessEntityID
             , IF(v.data_type = 'DEPT', val, NULL) AS DepartmentName
             , IF(v.data_type = 'PAY',  val, NULL) AS PayRate
             , v.ModifiedDate
          FROM (
                SELECT 'DEPT' AS data_type, edh.BusinessEntityID, edh.StartDate AS ModifiedDate, d.`Name` AS Val
                  FROM employeedepartmenthistory edh
                  JOIN department d
                    ON edh.DepartmentID = d.DepartmentID
                 UNION ALL
                SELECT 'PAY' AS data_type, eph.BusinessEntityID, eph.RateChangeDate AS ModifiedDate, eph.Rate AS Val
                  FROM employeepayhistory eph
               ) v
               ) w
  GROUP BY BusinessEntityID, ModifiedDate
      ),
      t_dept_filtered AS (
                SELECT BusinessEntityID, DepartmentName, seq
                  FROM t_data
                 WHERE DepartmentName IS NOT NULL
      ),
      t_dept_range AS (
       SELECT BusinessEntityID, DepartmentName, seq AS seq_start
            , IFNULL(LEAD(seq) OVER (PARTITION BY BusinessEntityID ORDER BY seq), 999999) AS seq_end
         FROM t_dept_filtered
```

Line 1 Column 0

Enable lazy conversion ☐

Replace variables in script? ☐

Insert data from step

Execute for each row? ☐

Limit size: 0

Sort Rows:

Sort rows

Step name: Sort rows

Sort directory: %%java.io.tmpdir%% Browse...

TMP-file prefix: out

Sort size (rows in memory): 1000000

Free memory threshold (in %):

Compress TMP Files? ☐

Only pass unique rows? (verifies keys only) ☐

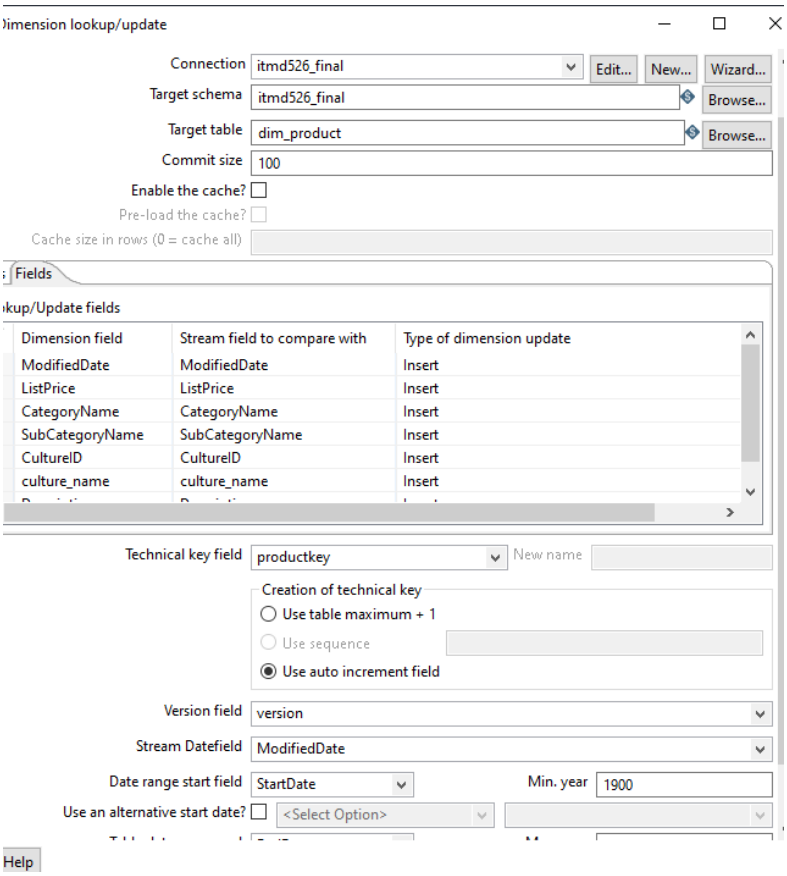
Fields :

#	Fieldname	Ascending	Case sensitive compare?	Sort based on current locale?	Collator Strength	Presorted?
1	ProductID	Y	N	N	0	N
2	ModifiedDate	Y	N	N	0	N

Department of Information Technology and Management
ITMD 526 – Data Warehousing

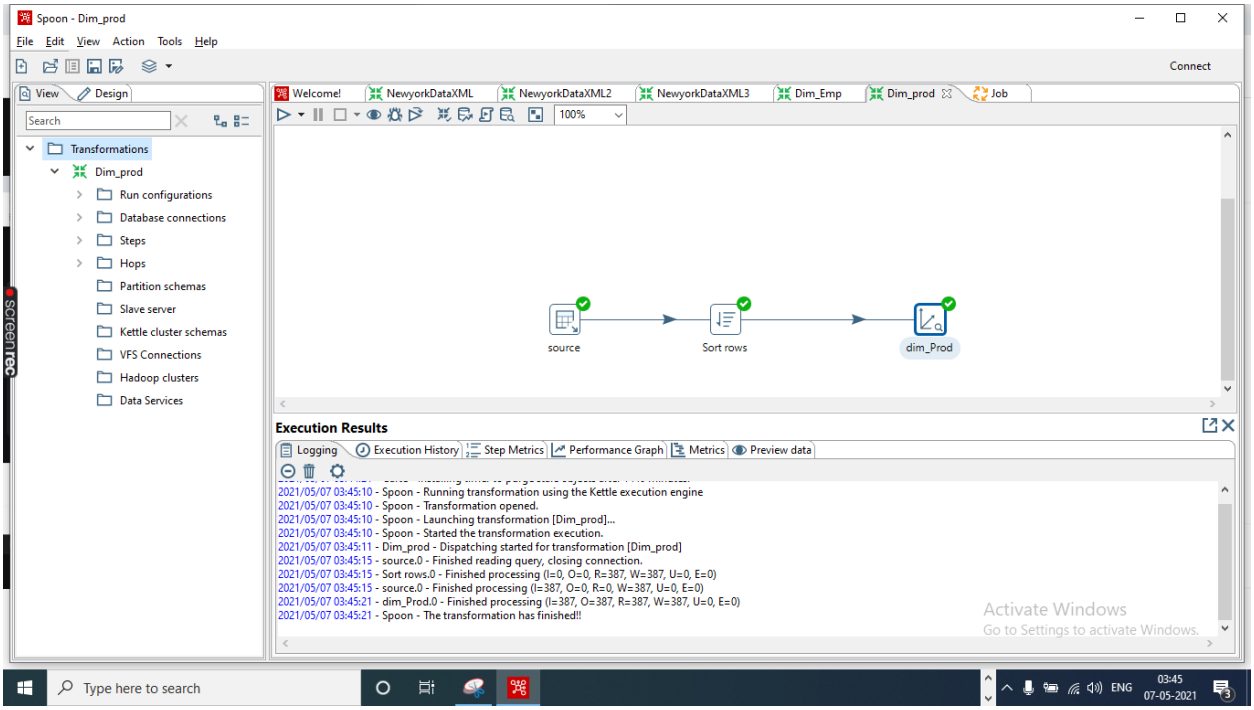
Target output Table:

The selected values from source table are being sorted and mapped to its corresponding target table. The table is being truncated and new value is being inserted on every update.

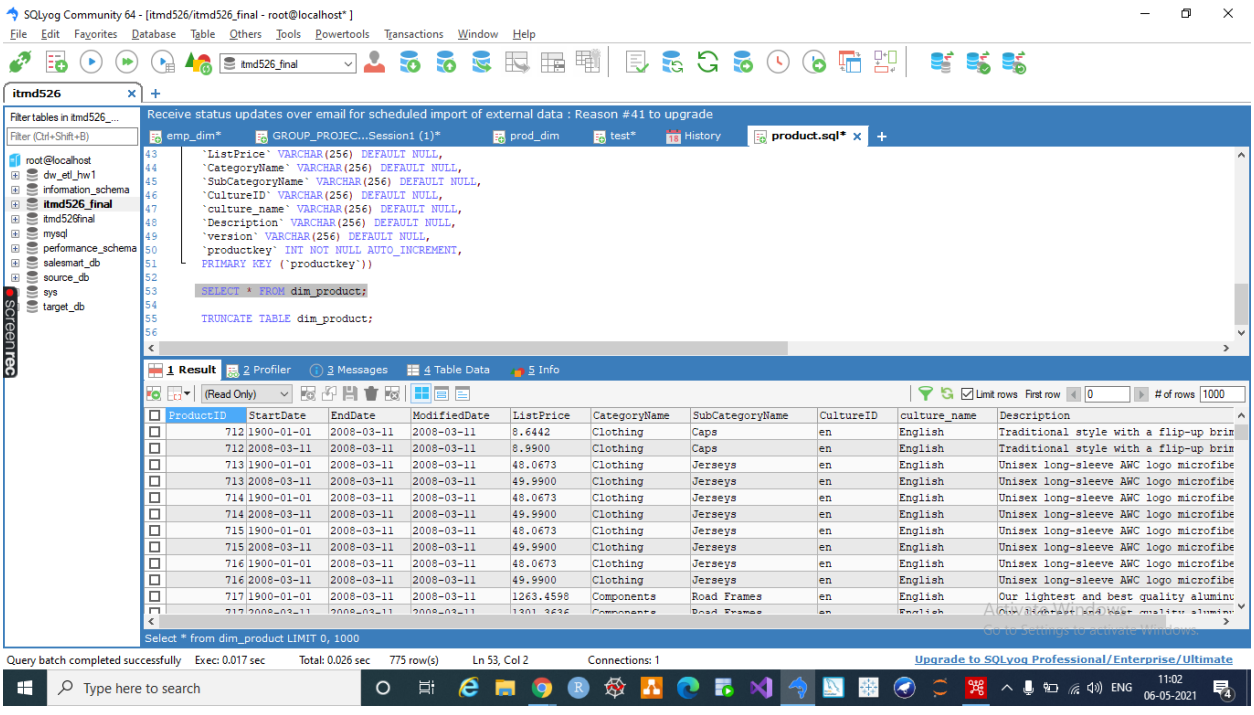


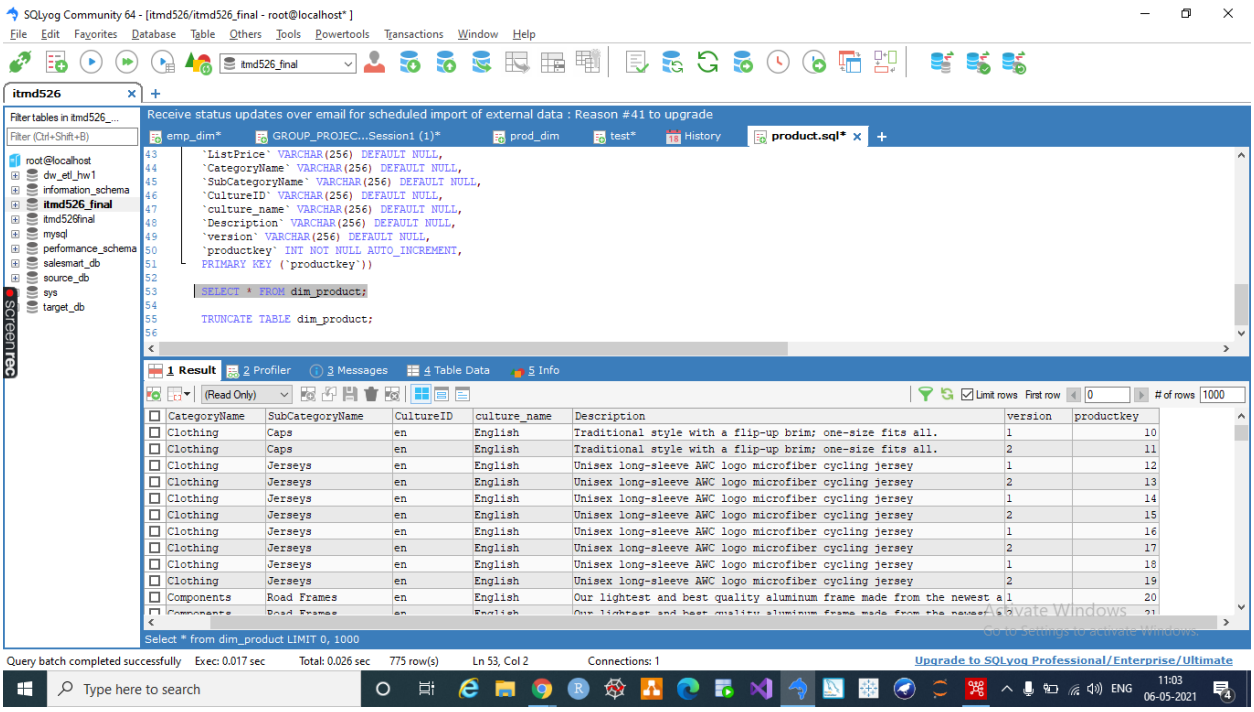
Department of Information Technology and Management
ITMD 526 – Data Warehousing

Result:



Snapshot of Table after running kettle Transformation:

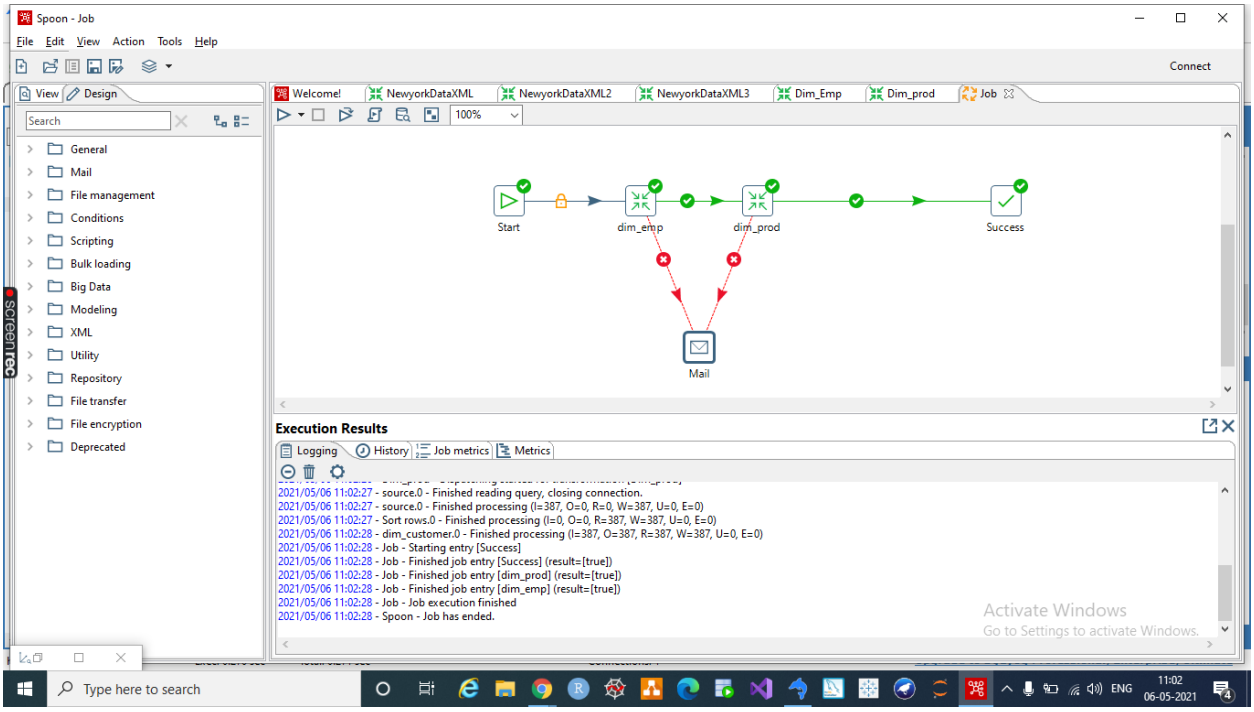




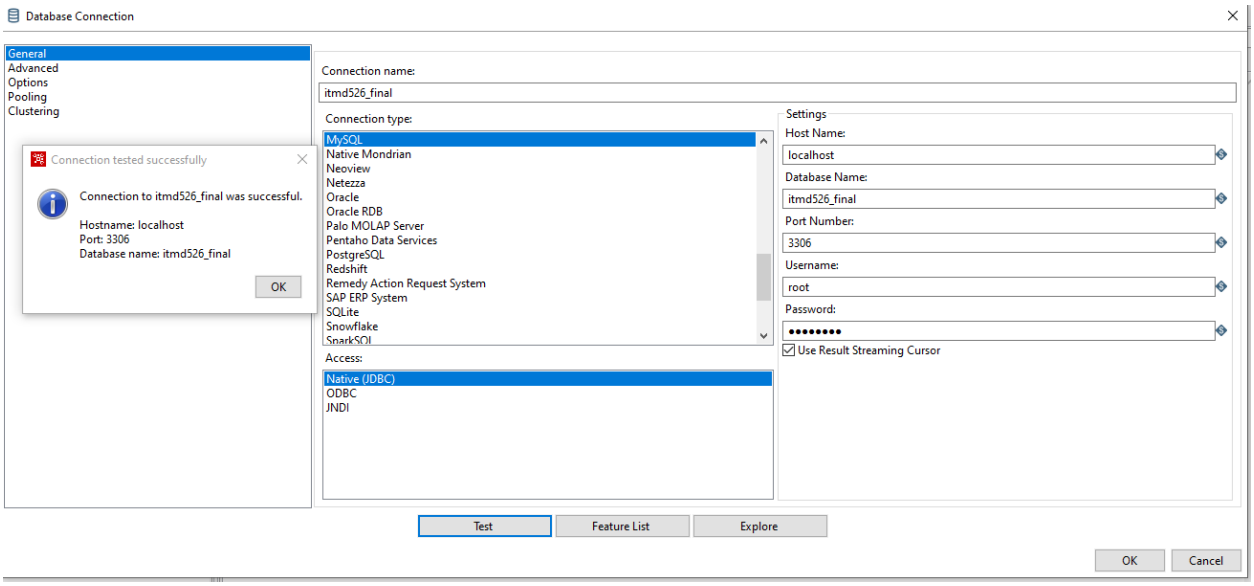
Kettle Job:

Job file runs both the employee and product transformation and provides success message when both the transformation is completed successfully else it automatically sends error message as an email to the corresponding receiver.

Department of Information Technology and Management
ITMD 526 – Data Warehousing



Database Connection: (Database Name: itmd526_final)



CONCLUSION:

Employee and Product dimension has been created and denormalized using SQL queries and SCD Type-2 is implemented using ETL Transformations.