Logo

Description automatically generated

Data Warehousing & Business Intelligence (IT)

3rd Year,1st Semester

**Assignment 1**

Submitted to

Sri Lanka Institute of Information Technology

IT20100698

BRITTO T.A

Weekday Batch

**STEP 01: Data set selection**

This dataset is an extension of MovieLens10M dataset, published by GroupLeans research group. It connects the movies in the MovieLens dataset to their web pages on the Internet Movie Database (IMDb) and the Rotten Tomatoes movie review systems. Only those users with both rating and tagging information were maintained from the original dataset. The dataset is released in the framework of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011).

According to the assignment principles, the data set was started with enough data.

In my data set which I have selected there are transactional data, and it has data of more than a year.

I can receive two sources of data from the data set that I selected (CSV and a text file). And the data set is sufficient to build a data warehouse. I was able to perform ETL functions with this data set.

And I can correctly identify hierarchies, dimensions, and aggregates in this data collection. I also realized that with this data collection, I will be capable of creating correct reports.

So, I selected this MovieLens + IMDb/Rotten Tomatoes details data set for my assignment.

Data Set Link:

<https://grouplens.org/datasets/hetrec-2011/>

The ER Diagram of the Data set

Diagram

Description automatically generated

**STEP 02: Preparation of Data Sources**

There were ten csv files available in the data set. They are movie\_actors.csv, movie\_countries.csv, movie\_directors.csv, movie\_genres.csv, movie\_locations.csv, movie\_tags.csv, movies.csv, tags.csv, user\_ratedmovies.csv and user\_taggedmovies.csv. Among these csv files movie\_countries.csv, movie\_directors.csv, movie\_tags.csv, tags.csv and user\_taggedmovies.csv were not taken to create tables.

I created separate two tables for my source dataset by dividing movies.csv. They are movies table and rottenTomatoesRating table. And I decided to convert movies data csv into text file format as movies.txt. I renamed rottenTomatoesRating table as rotten\_tomatoes.csv.

|  |  |  |
| --- | --- | --- |
| **Data Source Name** | **Data Source Type** | **Description** |
| Movies | TEXT | This file contains basic information about the movies of the database |
| Rotten Tomatoes | CSV | This file contains information about the Rotten Tomatoes ratings of the movies |
| Movie Locations | CSV | This file contains filming locations of the movies |
| Movie Genres | CSV | This file contains the genres of the movies. |
| User Rated Movies | CSV | This file contains the information about how users have rated the movies, and the date that a specific rating has done |
| Movie Actors | CSV | This file contains the main actors and actresses of the movies.  A ranking is given to the actors of each movie according to the order in which they appear on the movie IMDb cast web page. |

Then I imported those csv files into my newly created HetRecMovies\_SourceDB. After I imported my csv files into sourceDB, I created Data warehouse named HetRecMovies\_DW and created my dimension tables and fact tables inside the data warehouse.

The tables which I have imported to my database source is shown below.

Graphical user interface, application, table, Excel

Description automatically generatedTable

Description automatically generated

**Table

Description automatically generated** **movie\_actors.csv movie\_genres.csv**

**Table

Description automatically generated with medium confidence**

**movie\_locations.csv** **user\_rated\_movies.csv**

**Table

Description automatically generatedA picture containing graphical user interface

Description automatically generatedA picture containing text, window, indoor

Description automatically generated movies.csv**

created separate two tables for my source dataset by dividing main movies.csv

**rotten\_tomatoes.csv** **movies.csv**

**Text

Description automatically generated** The movie details were taken as a text file, and I used it as another source type.

**Table

Description automatically generated**

**movies.csv movies.txt**

I have loaded movie\_actors.csv, movie\_genres.csv, movie\_locations.csv, user\_rated\_movies.csv, rotten\_tomatoes.csv files to the DB called HetRecMovies\_SourceDB.

Table

Description automatically generated

Graphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generatedTable

Description automatically generatedData Sources Data Types

Table

Description automatically generatedChart

Description automatically generated with medium confidence

**STEP 03: SOULUTION ARCHITECTURE**

Diagram

Description automatically generatedArchitectural diagram

The data warehouse is the core of the BI system. A data warehouse is a database built for the purpose of data analysis and reporting. This purpose changes the design of this database as well. This architecture shows the high-level BI solution to the warehouse.

**STEP 04: Data Warehouse design and development**

**Diagram

Description automatically generated**

**Assumptions**

I have taken **DimMovieActors as slowly changing dimension**, Actor Ranking can change time to time, and we need to keep track of their historical data.

For my data set the schema is a snowflake schema. In my data set there are 4-dimension tables, date dimension and a fact table. The slowly changing dimension is Movie Actors table.

I have implemented data warehouse tables in the SQL server and the tables that I have created are

shown below.

Diagram, engineering drawing

Description automatically generated

**Data Warehouse Date Types**

Before creating the FactUserRatedMovies fact table and other dimensions, start by creating the Date

Table

Description automatically generatedTable

Description automatically generateddimension. For that I used the code in the file “DateMaster.sql” file.

**Table

Description automatically generated** **FactUserRatedMovies**  **DimMovieGenres**

**DimRottenTomatoes**

Graphical user interface, table

Description automatically generatedTable

Description automatically generated

**DimMovieActors DimMovies**

Table

Description automatically generated

**DimDate**

**Graphical user interface, application

Description automatically generatedSTEP 05: ETL development**

First using the SQL Server Integration Services Software, I have extracted all the data from the tables which were in the HetRecMovies\_SourceDB and Extract movies.txt to separate staging DB called HetRecMovies \_Staging as shown in the below.

Table

Description automatically generated with medium confidence

**HetRecMovies\_Staging**

**Extract Movie Actors Data to Staging**

**Graphical user interface, text, application

Description automatically generated**

Used OLE DB Source as dbo. movie\_actors data table in HetRecMovies\_SourceDB.OLE DB Destination for create new table MovieActors\_Staging in the HetRecMovies \_Staging database.

Graphical user interface, text, application, email

Description automatically generated

Used to execute SQL Task SSIS tool truncate table for SQL command as truncate table [HetRecMovies\_Staging]. [dbo]. [MovieActors\_Staging] in HetRecMovies\_Staging database.

**Extract Movie Locations Data to Staging**

**Graphical user interface, text, application

Description automatically generated**

Used OLE DB Source as dbo. movie\_locations data table in HetRecMovies\_SourceDB.OLE DB Destination for create new table MovieLocations\_Staging in the HetRecMovies \_Staging database.

Graphical user interface, text, application

Description automatically generated

Used to execute SQL Task SSIS tool truncate table for SQL command as truncate table [HetRecMovies\_Staging]. [dbo]. [MovieLocations\_Staging] in HetRecMovies\_Staging database.

**Extract Movie Genres Data to Staging**

**A screenshot of a computer

Description automatically generated**

Used OLE DB Source as dbo. movie\_genres data table in HetRecMovies\_SourceDB.OLE DB Destination for create new table MovieGenres\_Staging in the HetRecMovies \_Staging database.

Graphical user interface, application

Description automatically generated

Used to execute SQL Task SSIS tool truncate table for SQL command as truncate table [HetRecMovies\_Staging]. [dbo]. [MovieGenres\_Staging]in HetRecMovies\_Staging database.

**Extract RottenTomatoesRating Data to Staging**

**Graphical user interface, text, application, Word

Description automatically generated**

Used OLE DB Source as dbo. rotten\_tomatoes data table in HetRecMovies\_SourceDB.OLE DB Destination for create new table RottenTomatoesRatings\_Staging in the HetRecMovies \_Staging database.

Graphical user interface, application

Description automatically generated

Used to execute SQL Task SSIS tool truncate table for SQL command as truncate table [HetRecMovies\_Staging]. [dbo].[RottenTomatoesRatings\_Staging] in HetRecMovies\_Staging database.

**Extract User Rated Movies Data to Staging**

Graphical user interface, text, application, email

Description automatically generated

Used OLE DB Source as dbo. User\_rated\_movies data table in HetRecMovies\_SourceDB.OLE DB Destination for create new table UserRatedMovies\_Staging in the HetRecMovies \_Staging database.

Graphical user interface, application

Description automatically generated

Used to execute SQL Task SSIS tool truncate table for SQL command as truncate table [HetRecMovies\_Staging]. [dbo]. [UserRatedMovies\_Staging] in HetRecMovies\_Staging

database.

**Extract Movies Data to Staging**

**Graphical user interface, text, application, chat or text message

Description automatically generated**

Used Flat file Source SSIS tool, to extract movies.txt data.

Graphical user interface, application

Description automatically generated

Used OLE DO Destination SSIS tool to create new table as movies.txt load next file data to HetRecMovies\_Staging database.

Graphical user interface, text, application

Description automatically generated

Used to execute SQL Task SSIS tool truncate table for SQL command as truncate table [HetRecMovies\_Staging]. [dbo]. [UserRatedMovies\_Staging] in HetRecMovies\_Staging

database.

**Data Profiling**

I used the staging table data to analyze how the data looks like to determine what type of transformations I need to perform on the data.

Graphical user interface, application

Description automatically generated**Graphical user interface, text, application

Description automatically generated**

When double click the Data Profiling task and click on Open Profile Viewer to view the analyzed data.

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generatedGraphical user interface

Description automatically generated

Graphical user interface, table

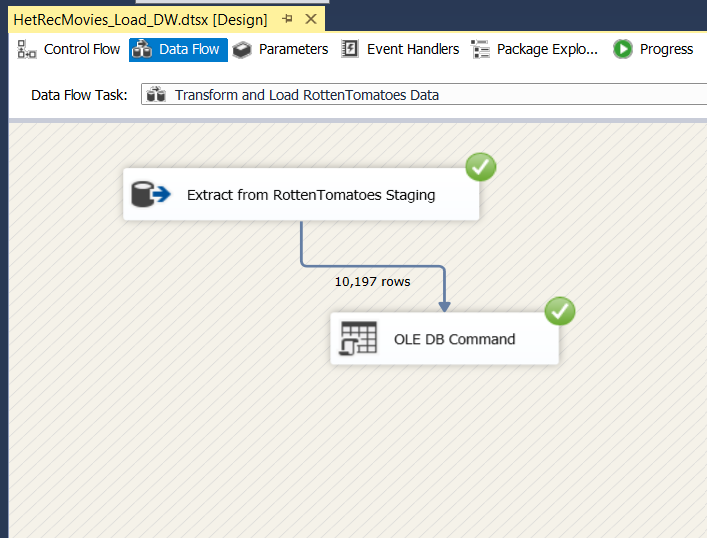
Description automatically generated

**Data Transformation**

**Transform and Load Rotten Tomatoes Data**

I created Rotten Tomatoes data Transformation by below mentioned steps.

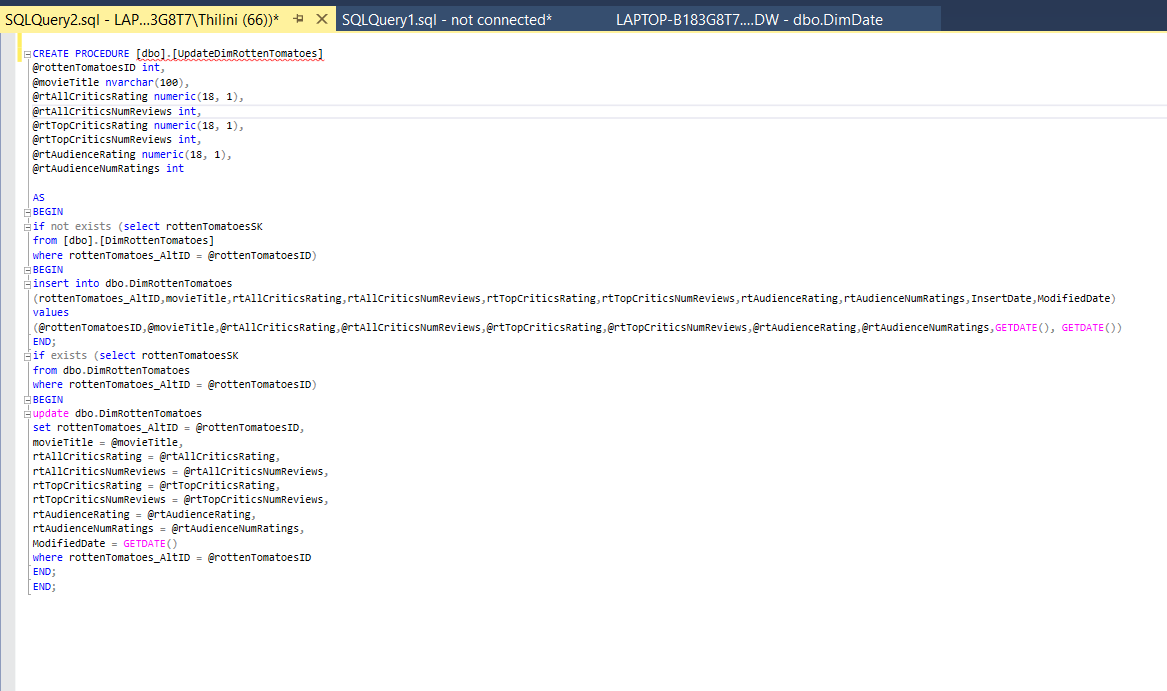
* + - * Created new package called HetRecMovies\_Load\_DW.dtsx.
      * Then Dragged and dropped a Data Flow Task, renamed it as Transform and LoadRotten Tomatoes Dataand go the Data Flow tab.
      * Dragged and dropped OLE DB Source, renamed as Extract from RottenTomatoes Staging and configure it to access the Rotten Tomatoes Staging table**.**
      * After that I dragged and dropped OLE DB Command rotten tomatoes and connect the OLE DB sourceIn the OLE DB Command, I set the configurations as below.



First, I have created a procedure called **UpdateDimRottenTomatoes** and executed in the

HetRecMovies\_DW database.

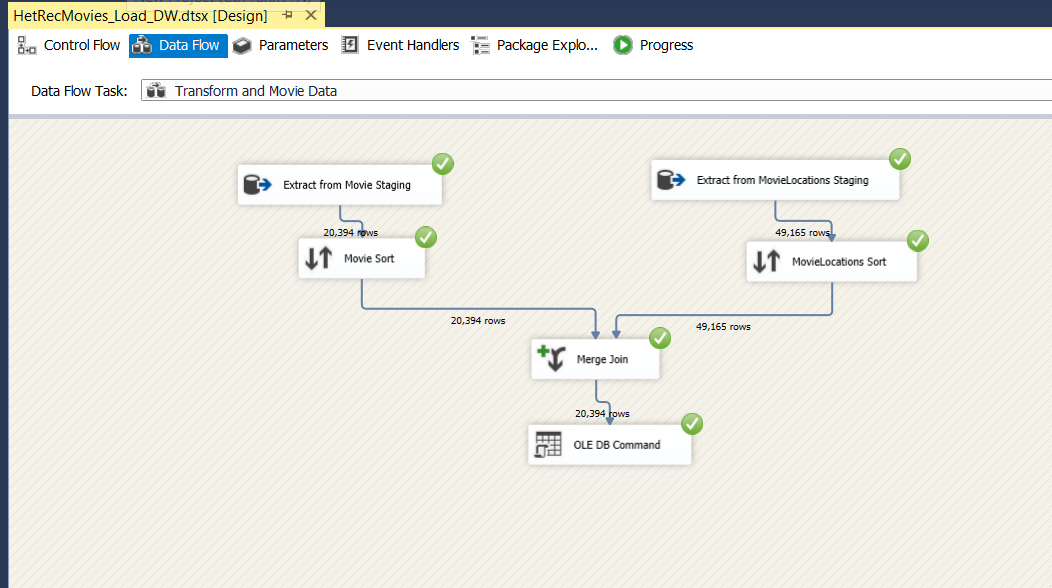
OLE DB Command SSIS tool used to execute, **UpdateDimRottenTomatoes procedure**, it is used to insert data into rotten tomatoes staging to DimRottenTomatoes without data duplication.



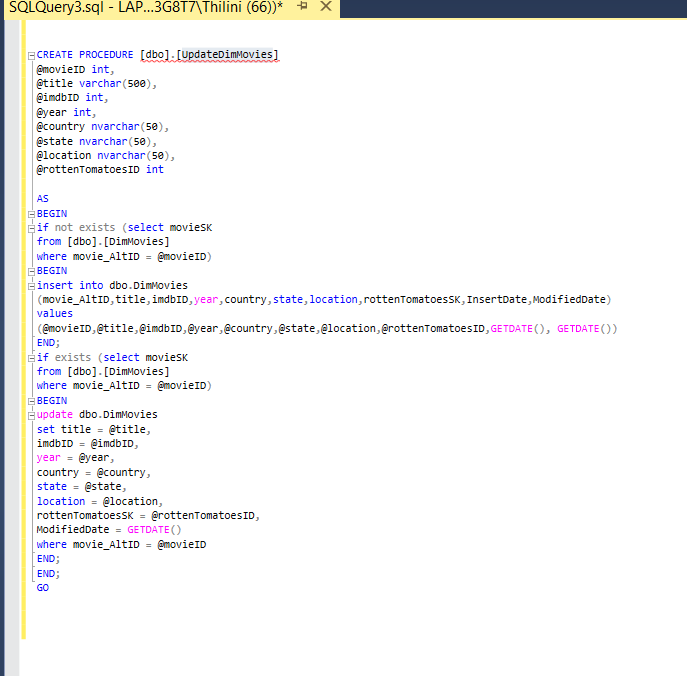
Then I did same process to Movies table and MovieGenres table as well.

**Transform and Movie Data**

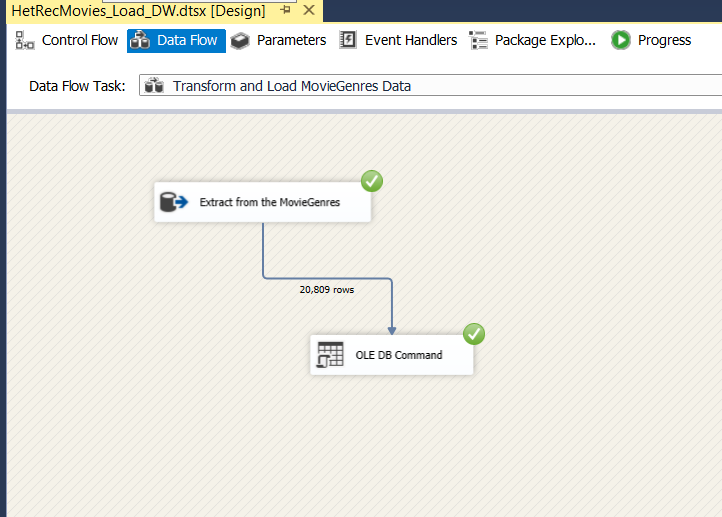
Movies table and movie locations table primary key is movieID therefore**,** I created **DimMovies** dimension table by connecting Movies\_Staging and MovieLocations\_Staging.

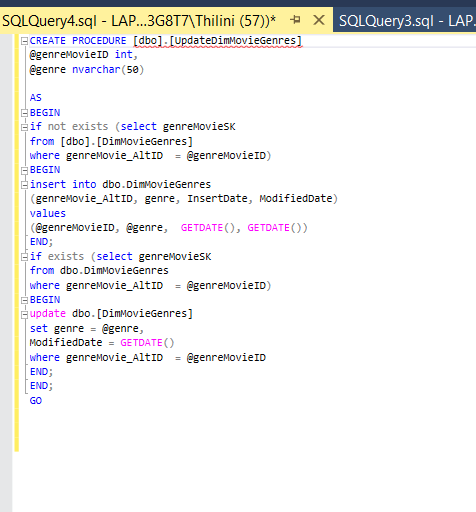
****

**UpdateDimMovies Procedure**



**Transform and Load MovieGenres Data**

****

**UpdateDimMovieGenres Procedure**

**Transform and Load MovieActors Data (Slowly Changing Dimension)**

I created Movie Actors Data Transformation by below mentioned steps.

• Created new package called HetRecMovies\_Load\_DW.dtsx.

• Then dragged and dropped a Data Flow Task, renamed it as Transform and Load Movie Actors data details and go the Data Flow tab.

• Dragged and dropped OLE DB Source, renamed as Extract from Movie Actors Staging

and configure it to access the Movie Actors Staging table.

• After that I dragged and dropped Slowly Changing Dimension movie actors and connect the OLE DB source

Graphical user interface, text, application

Description automatically generatedGraphical user interface, text

Description automatically generated• In the SCD Configuration Wizard I set the configurations as below.

Graphical user interface, text, application, email

Description automatically generated

Once all configurations are done properly, it will automatically create the slowly changing dimension as shown below.

Graphical user interface, text, application

Description automatically generated

**Transform and Load FactUserRatedMovies Data**

**Graphical user interface, text, application, email

Description automatically generated**

**STEP 06: ETL Development -Accumulating Fact Table**

First, I extended my fact table (FactUserRatedMovies Table) with following 03 columns.

accm\_txn\_create\_time

accm\_txn\_complete\_time

txn\_process\_time\_hours

**Text

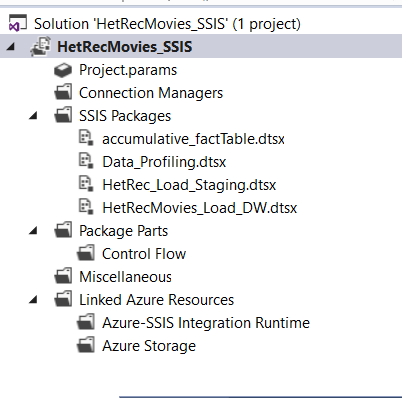
Description automatically generated**

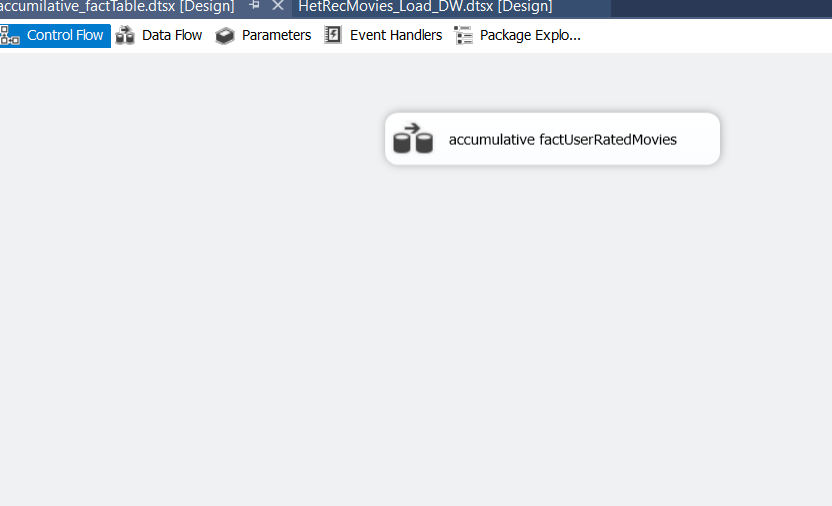
Then I prepared a dataset which contains fact table natural key(userID) and accm\_txn\_complete time.

Table

Description automatically generated

After that, I generated a new ETL SSIS package called accumulative\_factTable.dtsx. Which receives data from this file and updates the accm txn complete time in my DW Fact table accordingly.





Graphical user interface, text, application

Description automatically generated

**Graphical user interface, application

Description automatically generatedFlat File Source**

**Graphical user interface, application

Description automatically generatedOLE DB Source**

Graphical user interface, table

Description automatically generated

**Graphical user interface

Description automatically generatedSort with FactUserRatedMovies natural key. (userID)**

Graphical user interface, application, Word

Description automatically generated

**Merge Join**

**Graphical user interface

Description automatically generated**

Graphical user interface, text, application, email

Description automatically generated

**Graphical user interface, application, table, Excel

Description automatically generatedFinal FactUserRatedMovies Table with Updated txn\_process\_time.**

**Final HetRecMovies\_Load\_DW Control Flow.**

Graphical user interface, text

Description automatically generated