Sri Lanka Institute of Information Technology

Data Warehousing & Business Intelligence

Assignment 1

Gamage M.G.U.D.

IT19169736

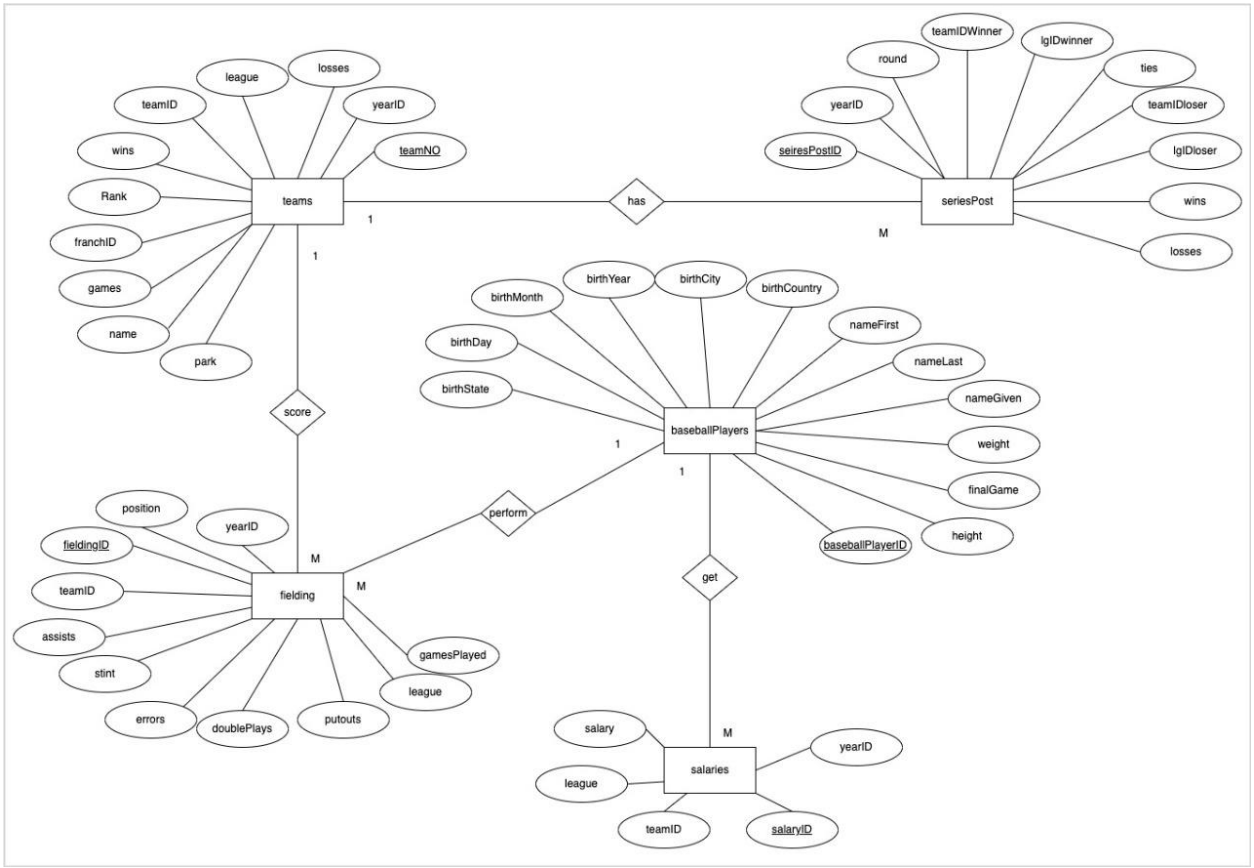
Year 3 Semester 1

Group 05 (Weekday)

Step 1 : Data set selection

Since we need to make rich ETL tasks, I supposed to take a data set which contain lots of data. And they must have at least one years of data. So, I thought to take sports data set since it has years of data and can easily make hierarchies and dimensions etc. The chosen data set is about Baseball tournaments and related areas. This Baseball Databank data set is fulfilling all these requirements. It has various kind of tables, and I am using the tables for my source BaseballPlayers, Fielding, Salaries, SeriesPost, Teams. And these are made by cut as required since there was lots of data. I reduced most of columns and rows to create my final source dataset.

Following is the ER diagram



Step 2 : Preparation of data sources

These customized tables are in two formats which are CSV and Text
BaseballPlayers - All the details about the players (Database table)
Fielding - Data of Fielding stasitics. (CSV file)
Salaries – Details of players' salaries. (CSV file)
SeriesPost – Post-season series information (TEXT file)
Teams – Contain all the details about teams (TEXT file)

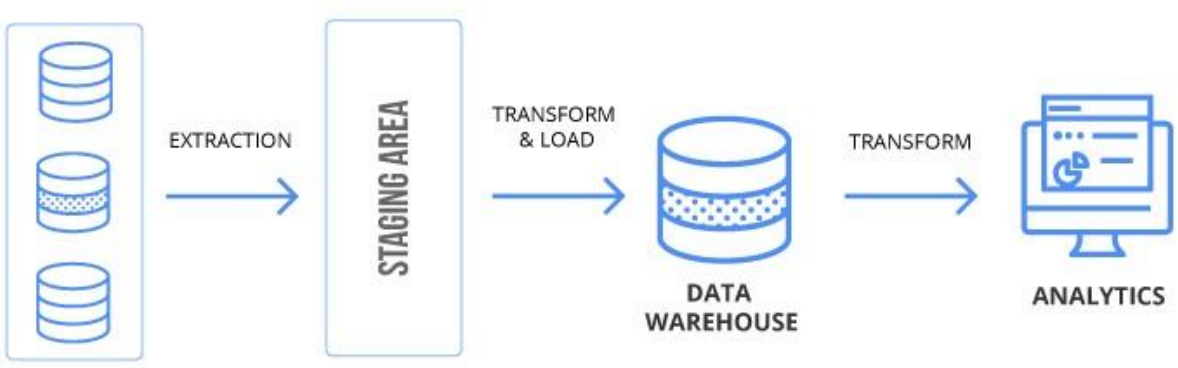
Step 3 : Solution Architecture

In data warehousing ETL process which means Extract, Transform, And Load there are 3 main components.

- Source Files
- Staging Database
- Datawarehouse

The chosen data is customized as requirements which means cut them from the original version since it has huge amount of data. Then using these data, we extract them and change their data types and put them in to the staging area.

Then we use those extracted data as source data from the staging area to transform. In this step data will transform to intelligible set of data and then load them in to the data warehouse database.

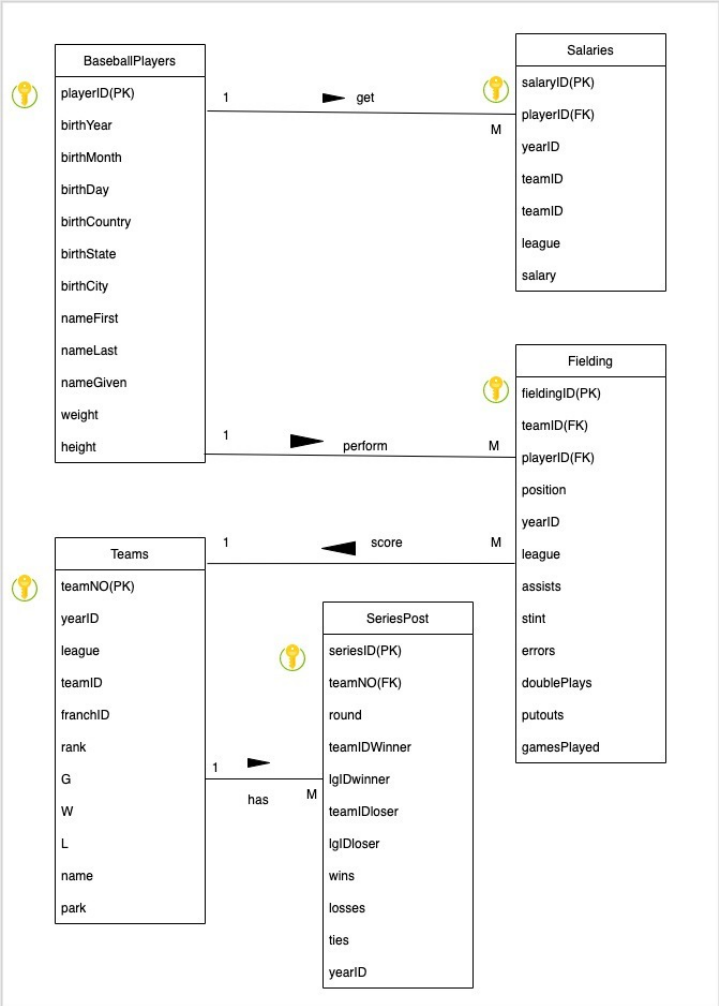


Step 4 : Data warehouse design and development

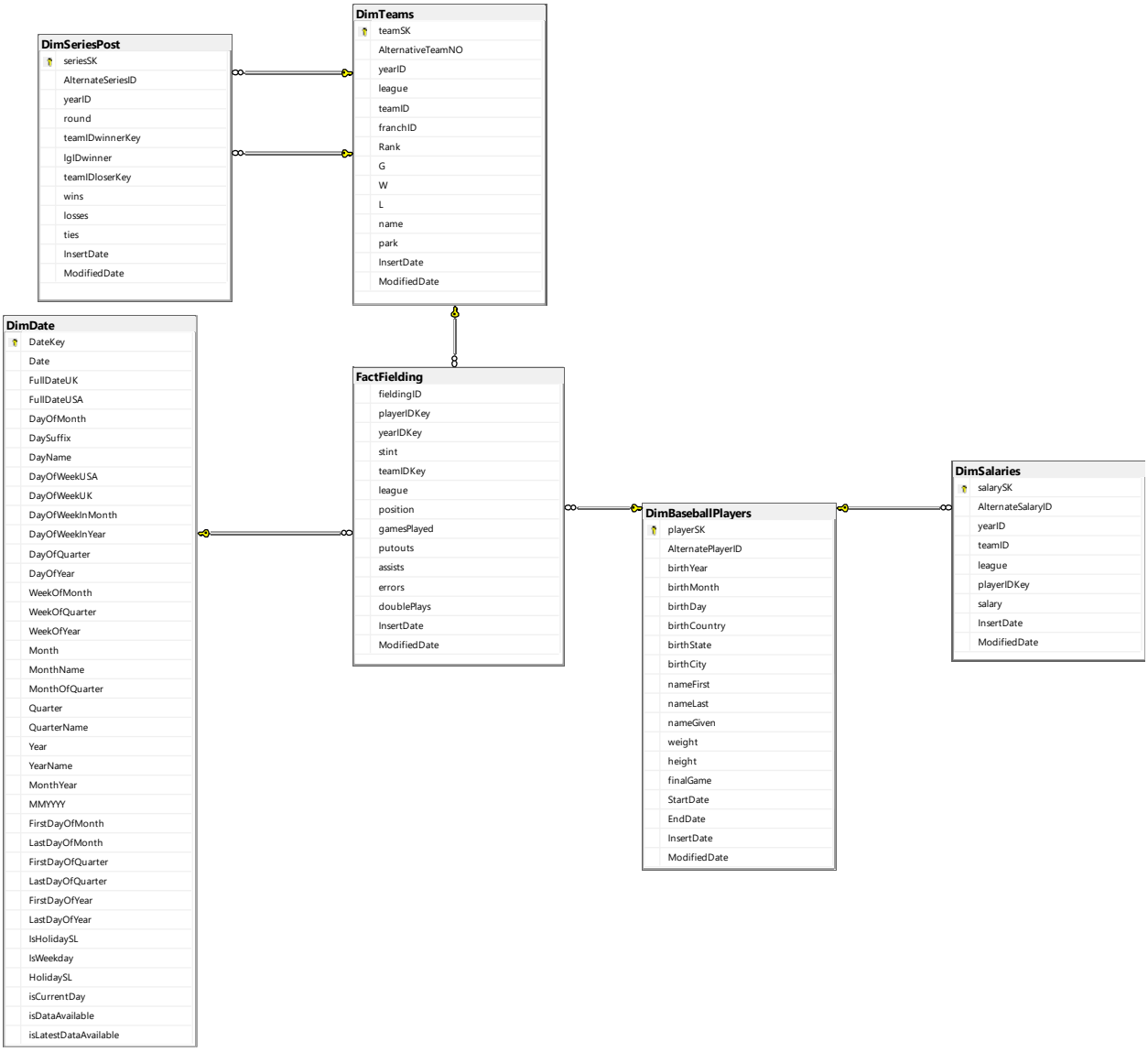
For the Data Warehouse of the Baseball Logs dataset, DimBaseballPlayers, DimSalaries, DimSeriesPost and DimTeams are implemented as the dimensions with DimDate as the Date dimension. FactFielding is implemented as the fact table. DimBaseballPlayers is implemented as the slowly changing dimension. DimBaseballPlayers contains details of Baseball players. DimSalaries contains salary details and has a foreign key reference to DimBaseballPlayers. DimSeriesPost contains details about Baseball series and has two foreign key references to DimTeams. DimTeams contains details about Baseball teams. DimDate contains date and date key. FactFielding contains fielding details and has foreign key reference to DimTeams, DimBaseballPlayers, DimDate.

Assumptions : I implemented DimBaseballPlayers as the slowly changing dimension assuming that the last name, final game, weight and height of the players can be changed over time.

Relational model is as follows

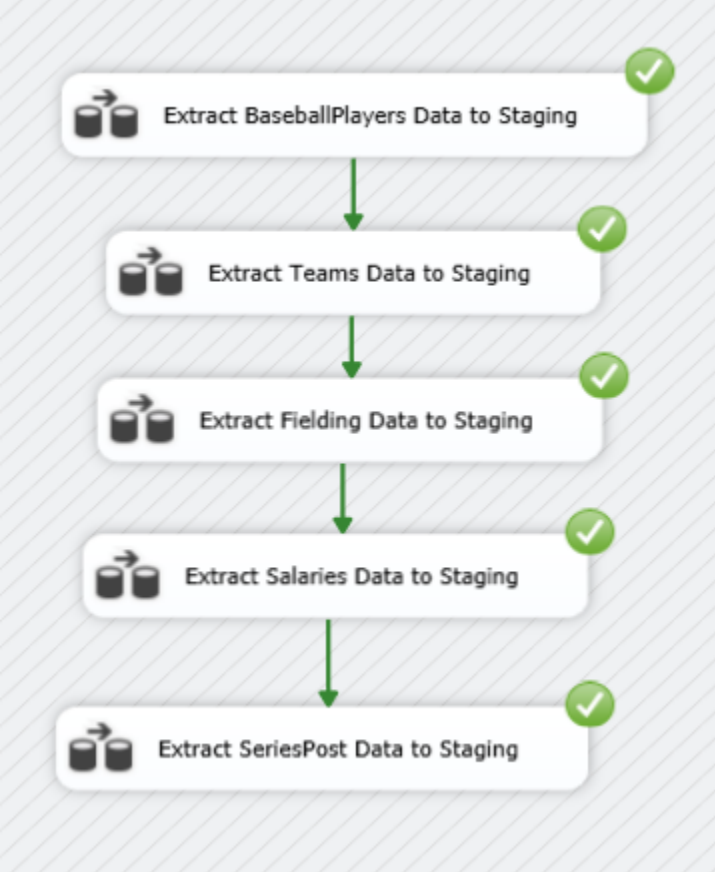


Data warehouse design is a snowflake and dimensional model can be shown as follows.



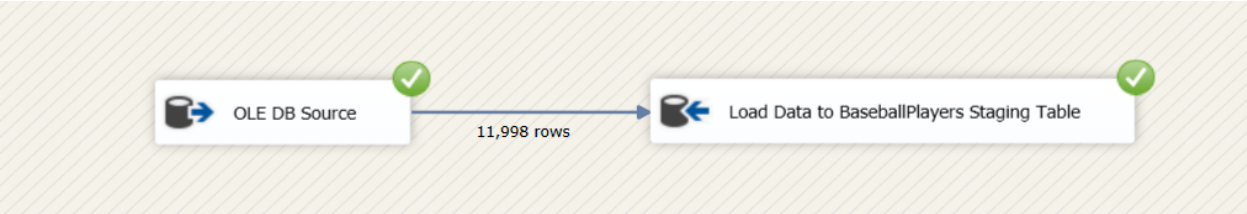
Step 5 : ETL development

Extracting data from source files and loading for staging is done in this step.



1

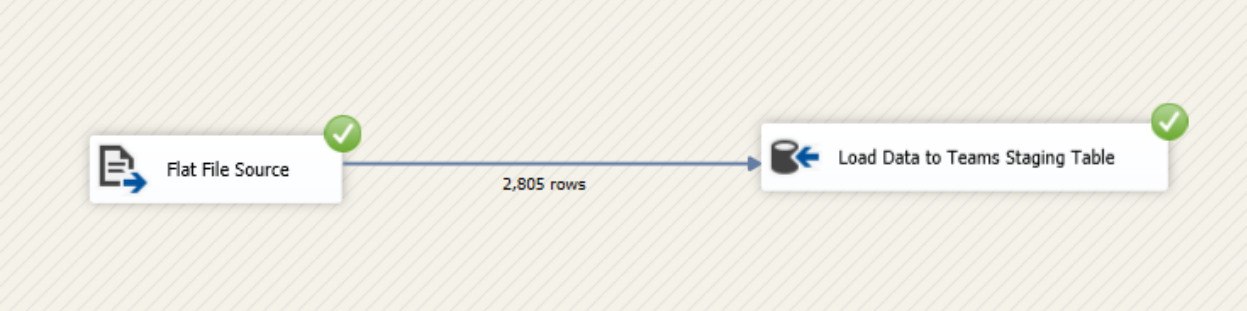
Extract from database table and load to staging table



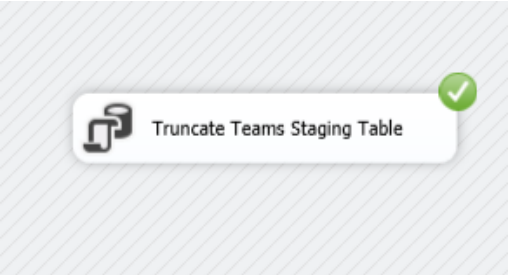
Truncate table to avoid duplication

2

Extract from text file and load to staging table

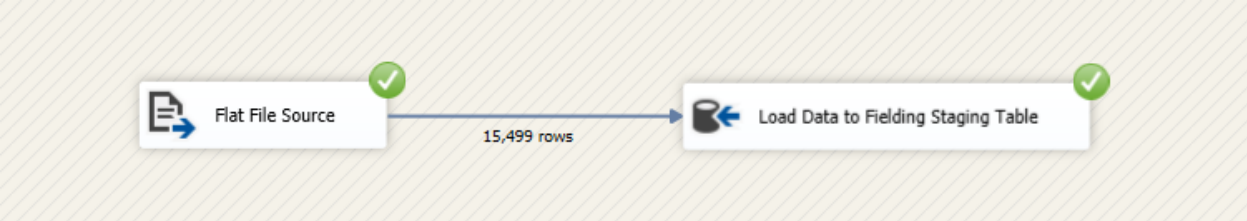


Truncate table to avoid duplication

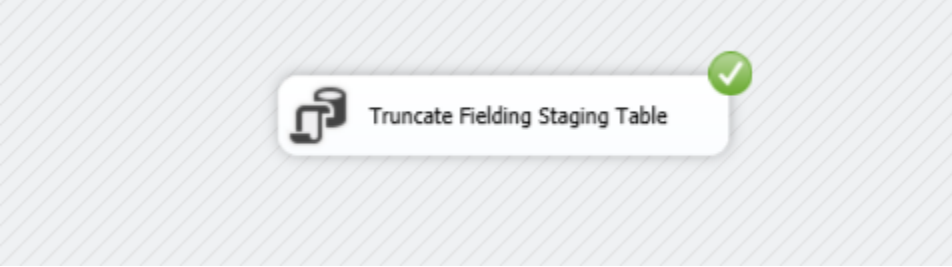


3

Extract from csv file and load to staging table

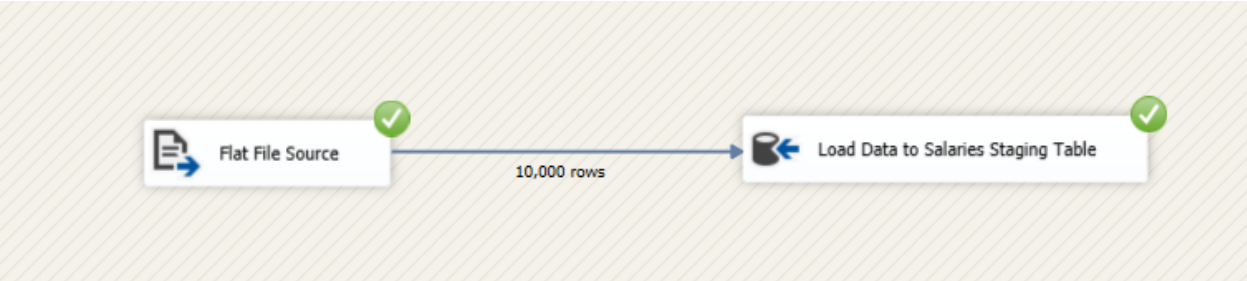


Truncate table to avoid duplication

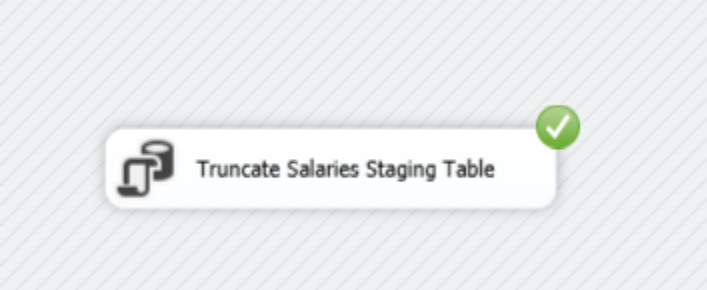


4

Extract from csv file and load to staging table

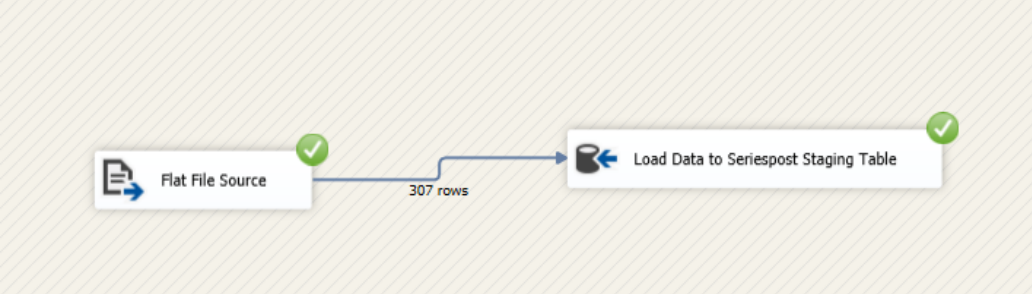


Truncate table to avoid duplication

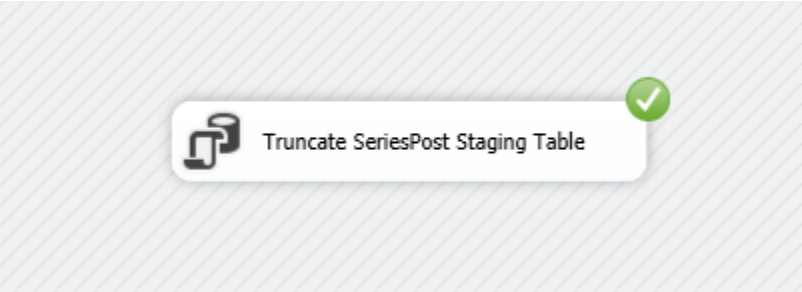


5

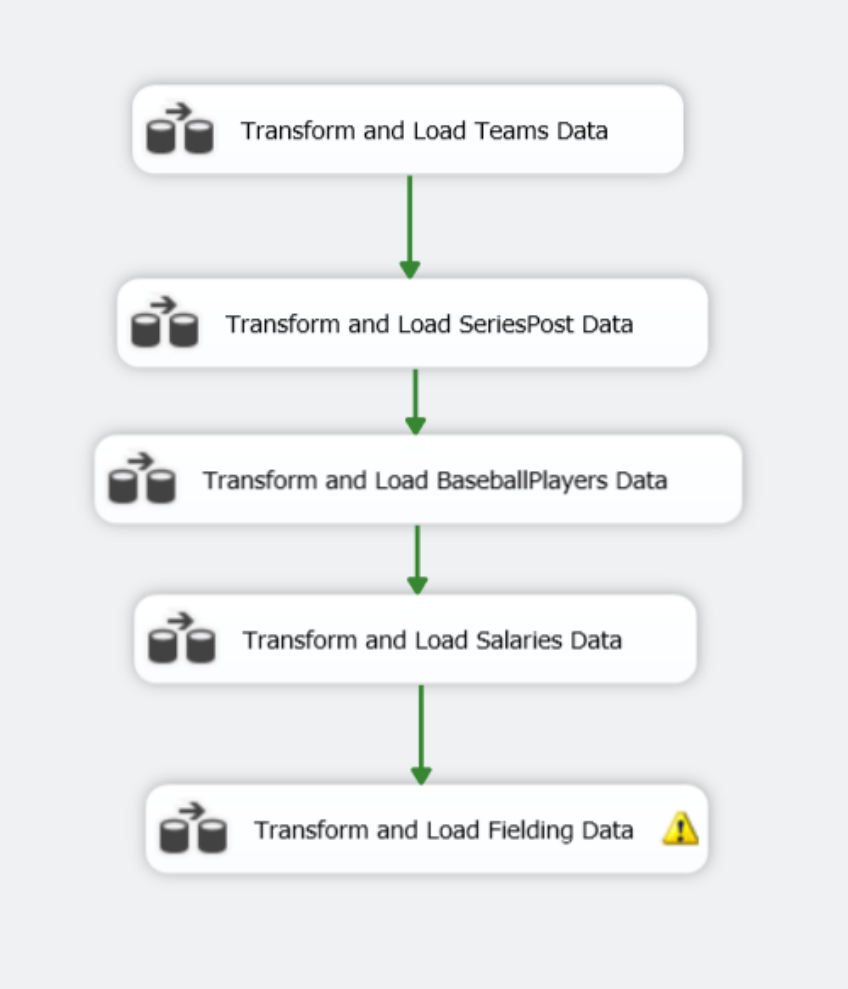
Extract from text file and load to staging table



Truncate table to avoid duplication

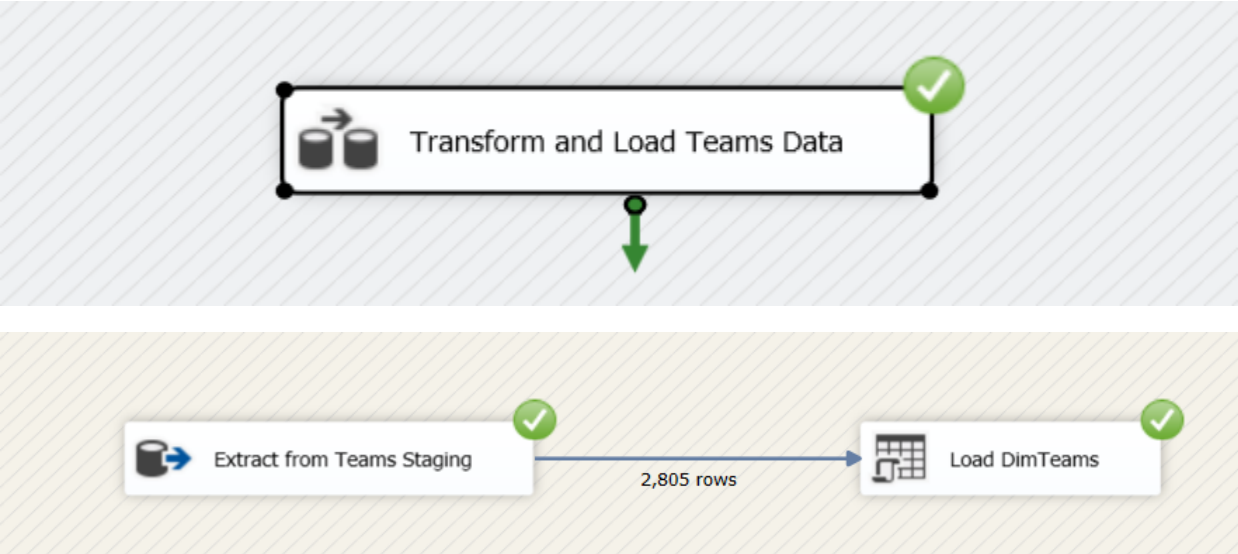


Transform and load is developed in this following step.



1

Extract from Teams staging table and load to DimTeams.

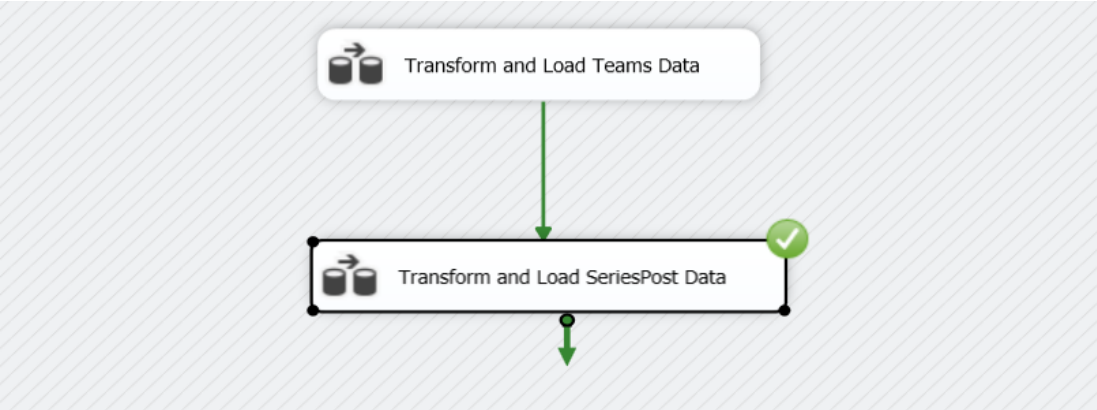


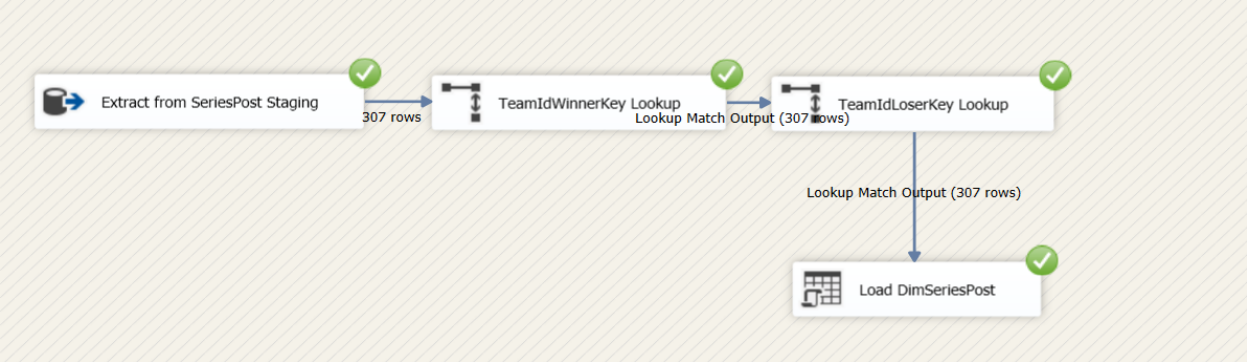
Executing procedure to insert data to DimTeams.

```
create procedure dbo.UpdateDimTeams
@teamNO nvarchar(50),
@yearID nvarchar(50),
@league nvarchar(50),
@teamID nvarchar(50),
@franchID nvarchar(50),
@Rank nvarchar(50),
@G nvarchar(50),
@W nvarchar(50),
@L nvarchar(50),
@name nvarchar(100),
@park nvarchar(100)
AS BEGIN if not exists (select teamSK from dbo.DimTeams where AlternativeTeamNO = @teamNO)
BEGIN insert into dbo.DimTeams(AlternativeTeamNO, yearID,league,teamID,franchID,Rank,G,W,L,name,park,InsertDate,ModifiedDate)
values (
    @teamNO,
    @yearID,
    @league,
    @teamID,
    @franchID,
    @Rank,
    @G,
    @W,
    @L,
    @name,
    @park,
    GETDATE(),
    GETDATE())
END;
if exists (select teamSK from dbo.DimTeams where AlternativeTeamNO = @teamNO)
BEGIN
update dbo.DimTeams set
    yearID = @yearID,
    league = @league,
    teamID = @teamID,
    franchID = @franchID,
    Rank = @Rank,
    G = @G,
    W = @W,
    L = @L,
    name = @name,
    park = @park,
    ModifiedDate = GETDATE()
    where AlternativeTeamNO = @teamNO
    END;
END;
```

2

Extract from SeriesPost staging table and load to DimSeriesPost



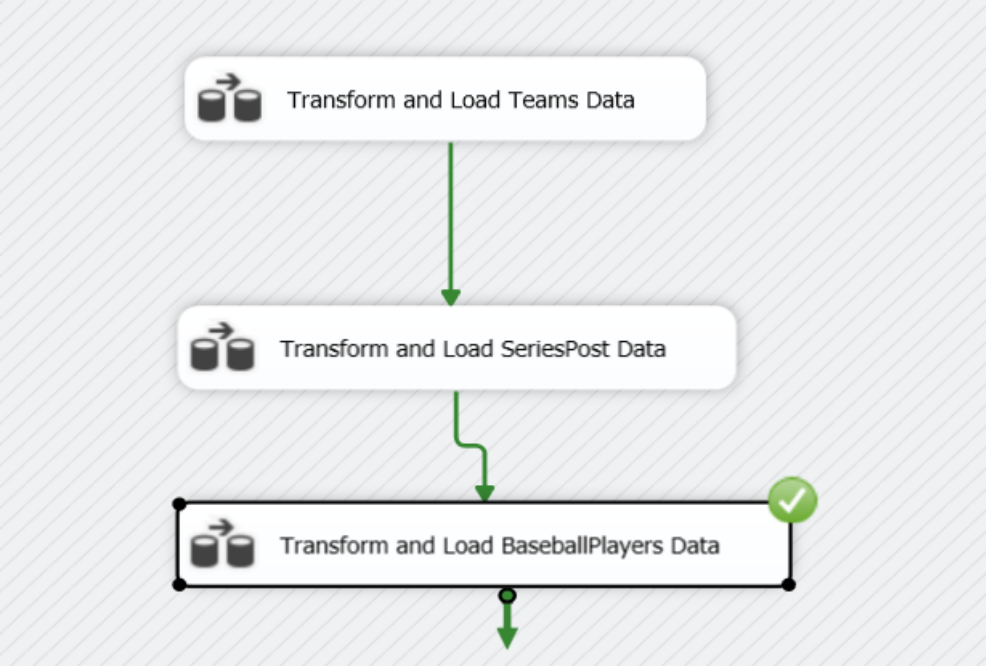


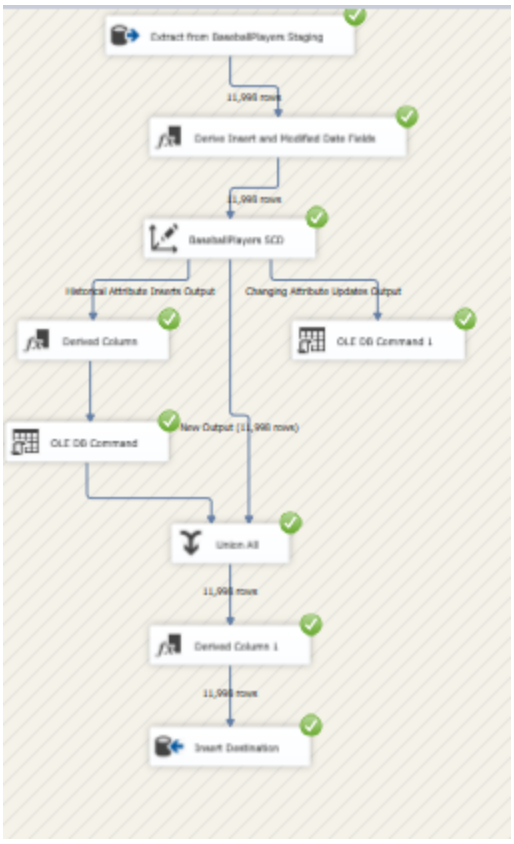
Executing procedure to insert data to DimSeriesPost.

```
create procedure dbo.UpdateDimSeriesPost
    @seriesID nvarchar(50),
    @yearID nvarchar(50),
    @round nvarchar(50),
    @teamIDwinner nvarchar(50),
    @lgIDwinner nvarchar(50),
    @teamIDloser nvarchar(50),
    @wins nvarchar(50),
    @losses nvarchar(50),
    @ties nvarchar(50)

AS BEGIN if not exists (
    select seriesSK from dbo.DimSeriesPost where AlternateSeriesID = @seriesID
)

BEGIN insert into dbo.DimSeriesPost(AlternateSeriesID,yearID,round,teamIDwinnerKey,lgIDwinner,teamIDloserKey,wins,losses,ties,InsertDate,ModifiedDate
)

values (
    @seriesID,
    @yearID,
    @round,
    @teamIDwinner,
    @lgIDwinner,
    @teamIDloser,
    @wins,
    @losses,
    @ties,
    GETDATE(),
    GETDATE()
)
END;

if exists (
    select seriesSK from dbo.DimSeriesPost where AlternateSeriesID = @seriesID
)

BEGIN
update dbo.DimSeriesPost set
    yearID = @yearID,
    round = @round,
    teamIDwinnerKey = @teamIDwinner,
    lgIDwinner = @lgIDwinner,
    teamIDloserKey = @teamIDloser,
    wins = @wins,
    losses = @losses,
    ties = @ties,
    ModifiedDate = GETDATE()
    where AlternateSeriesID = @seriesID
    END;
END;
```

3

Extract from BaseballPlayers staging table and load to DimBaseballPlayers.





I implemented DimBaseballPlayers as the slowly changing dimension. Height, weight, nameLast and finalGame are slowly changing attributes of DimBaseballPlayers dimension. Following types of slowly changing dimension columns are maintained.

Type 1 – Changing attributes

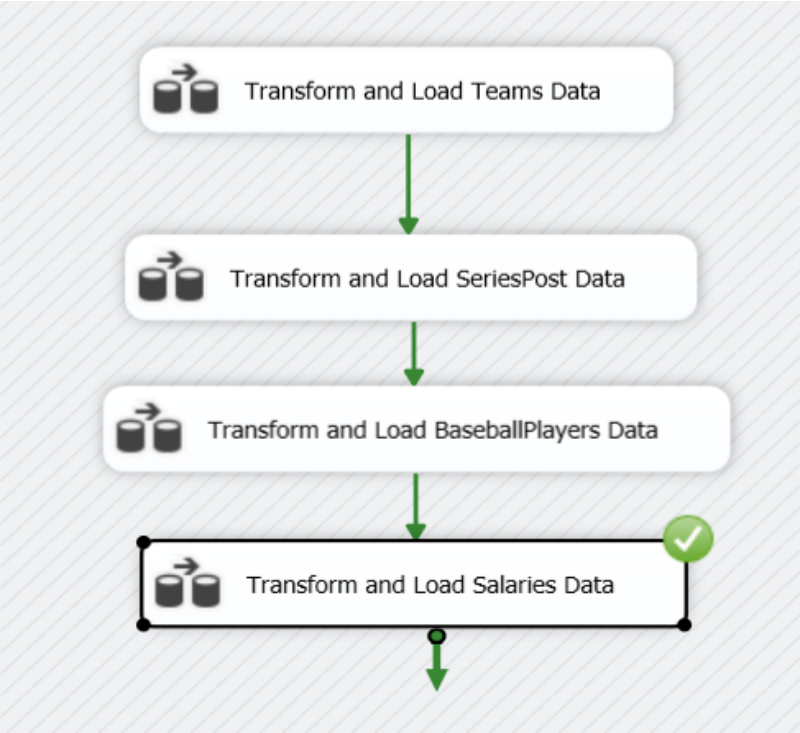
If a player's height or weight is changed, we should replace the old height or weight with the new value.

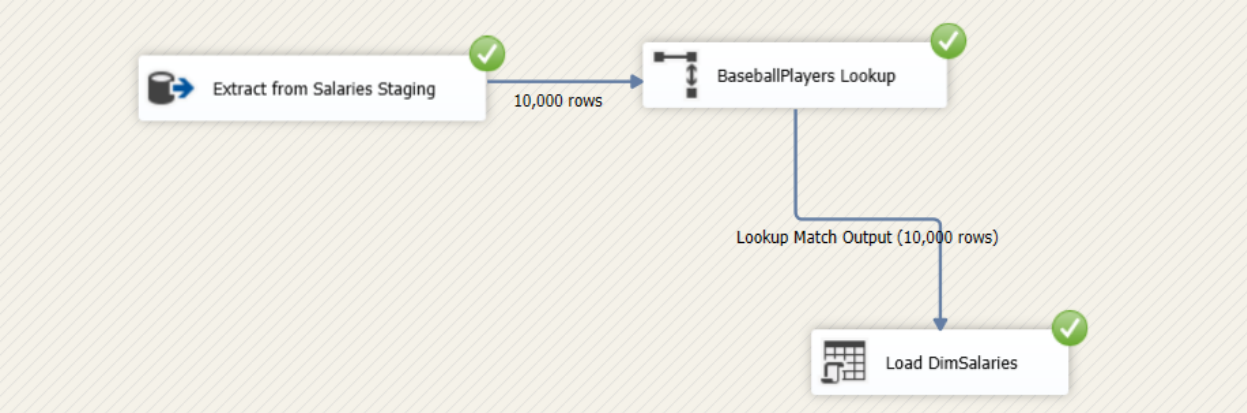
Type 2 – Historical attributes

Any player's last name and date of the final game can be changed by time. It is needed to keep track of those details and new records should be created.

4

Extract from Salaries staging table and load to DimSalaries.



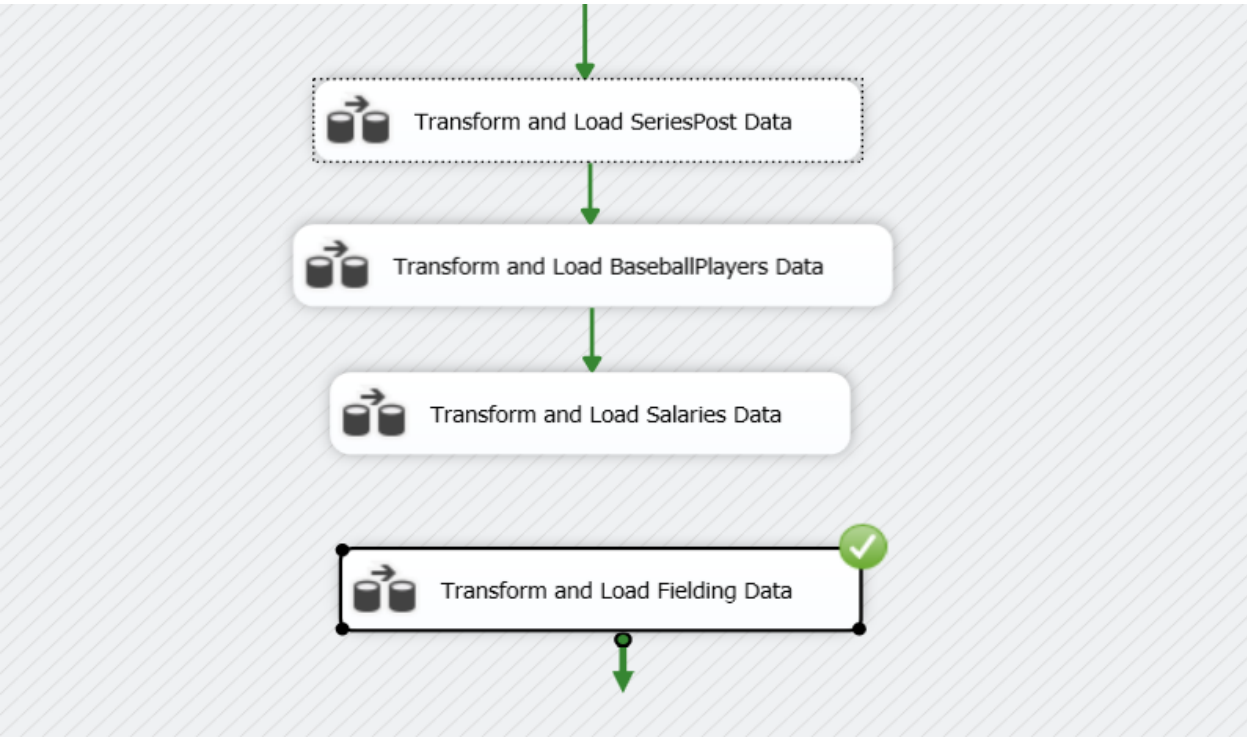


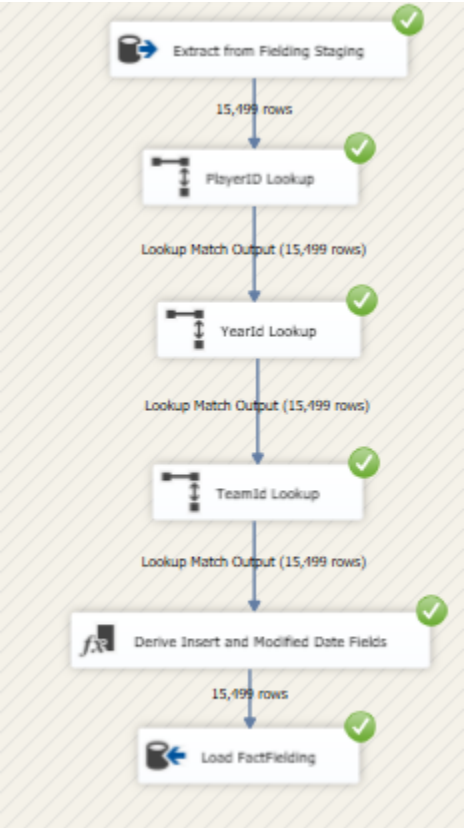
Executing procedure to insert data to DimSalaries

```
create procedure dbo.UpdateDimSalaries
    @salaryID nvarchar(50),
    @yearID nvarchar(50),
    @teamID nvarchar(50),
    @league nvarchar(50),
    @playerID nvarchar(50),
    @salary nvarchar(50)

AS BEGIN if not exists (
    select salarySK from dbo.DimSalaries where AlternateSalaryID = @salaryID
)
BEGIN insert into dbo.DimSalaries(
    AlternateSalaryID,
    yearID,
    teamID,
    league,
    playerIDKey,
    salary,
    InsertDate,
    ModifiedDate
)

values (
    @salaryID,
    @yearID,
    @teamID,
    @league,
    @playerID,
    @salary,
    GETDATE(),
    GETDATE()
)
END;

if exists (
    select salarySK from dbo.DimSalaries where AlternateSalaryID = @salaryID
)

BEGIN
update dbo.DimSalaries set
    yearID = @yearID,
    teamID = @teamID,
    league = @league,
    playerIDKey = @playerID,
    salary = @salary,
    ModifiedDate = GETDATE()
    where AlternateSalaryID = @salaryID
    END;
END;
```

5

Extract from Fielding staging table and load to FactFielding.


Transform and Load SeriesPost Data
Transform and Load BaseballPlayers Data
Transform and Load Salaries Data
Transform and Load Fielding Data


Extract from Fielding Staging
15,499 rows
PlayerID Lookup
Lookup Match Output (15,499 rows)
YearId Lookup
Lookup Match Output (15,499 rows)
TeamId Lookup
Lookup Match Output (15,499 rows)
Derive Insert and Modified Date Fields
15,499 rows
Load FactFielding

FactFielding table is connected to DimTeams, DimDate, DimBaseballPlayer using lookups.

Step 6: ETL development – Accumulating fact tables