Sri Lanka Institute of Information Technology

Assignment - 01

IT3021 - Data Warehousing and Business Intelligence

2025

IT22604644

KAVINDYA K.T.

Data set selection

I have chosen a dataset that includes information about Indian Premier League (IPL) games. This includes information on every ball played in the IPL. The dataset's link is provided below.

https://www.kaggle.com/datasets/patrickb1912/ipl-complete-dataset-20082020?select=IPL+Matches+2008-2020.csv

From this data set, data from 2013 to 2017 was chosen, and additional tables were made taking the relationships into account. To examine player-wise performance, the dataset is converted.

Preparation of data sources

All of the data was initially in "csv" format. They were transformed into several data sources, including a text file, two "csv" files, and a database backup file. The tables that were divided into several sources are as follows:

1. Database (.bak)
    - 1.1. Player
    - 1.2. Venue
    - 1.3. OutType
    - 1.4. ExtraType
    - 1.5. BallingStyle
    - 1.6. BattingStyle
    - 1.7. BallByBall
    - 1.8. BallData
2. Comma Seperated Values(.csv)
    - 2.1. Country
    - 2.2. Team
3. Text(.txt)
    - 3.1. VenueAddress

Data Source Type	Source Name	Column Name	Data Type	Description
Database File (.bak)	dbo. BallByBall	BallDataID	int	Includes facts of the IPL matches ball by ball.
		BallDataID	int	
		TeamBattingID	int	
		TeamBowlingID	int	
		StrikerID	int	
		NonStrikerID	int	
		RunsScored	int	
		ExtraTypeID	int	
		ExtraRuns	int	
		OutTypeID	int	

		OutPlayerID	int	
		IsBowlerWicket	int	
		BowlerID	int	
		FielderID	int	
		MatchDate	datetme	
		VenueID	int	
	dbo.BallData	BallDataID	int	
		MatchNo	int	Contains data about the balls in every match as a hierarchy. Ex: Third ball of second over of the first innings, of the fifth match.
		InningsNo	int	
		OverNo	int	
		BallNo	int	
	dbo.Venue	VenueID	int	Contains details of grounds where matches are played.
		VenueName	nvarchar(255)	
	dbo.ExtraType	ExtraTypeID	int	

		ExtraType	nvarchar(255)	Contains data types extras.
	dbo.OutType	OutTypeID	int	Contains data about types of wickets.
		OutType	nvarchar(255)	
	dbo.Player	PlayerID	int	Contains details of players.
		PlayeName	nvarchar(255)	
		PlayerNameInitials	nvarchar(255)	
		CountryID	int	
		BattingStyle	int	
		BowlingStyleID	int	
	dbo.BattingStyle	BattingStyleID	int	Contains details of batting styles.
		BattingStyle	nvarchar(255)	
	dbo.BowlingStyle	BowlingStyleID	int	Contains details of batting styles.
		BowlingStyle	nvarchar(255)	
CSV File	Country.csv	CountryID	int	Contains details of countries of players.
		CountryName	nvarchar(255)	
	Team.csv	TeamID	int	Contains details of teams of the

		TeanmName	nvarchar(255)	tournament.
Text file	VenueAddress.txt	VenueAddressID	int	Contains details addresses of the venues (grounds)
		CityName	nvarchar(255)	
		CountryName	nvarchar(255)	

Solution architecture

Data Sources

The sources that were used to obtain the data are represented by the data sources. CSV, text, and .bak are the three categories. bak , which stands for database files, text files, and files separated by commas, respectively.

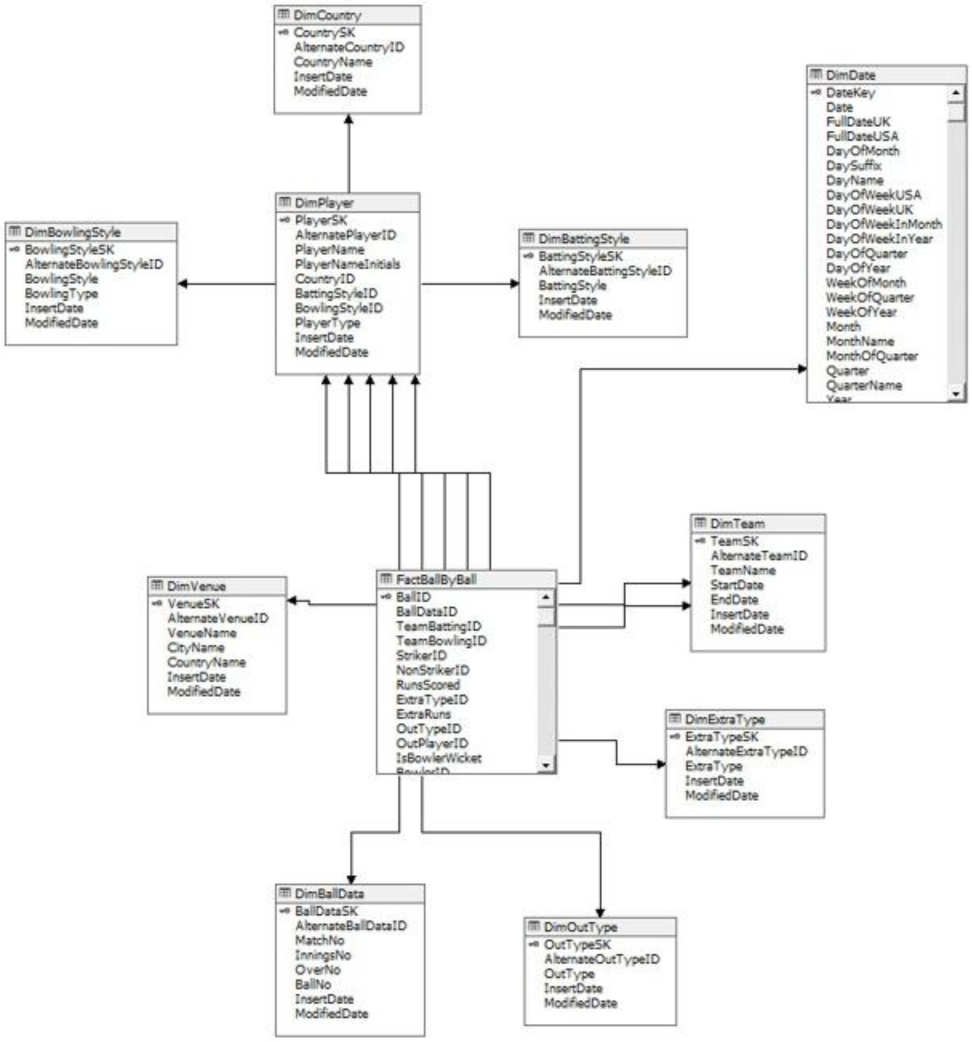
Staging Area

Using the data gathered from various data sources, this level represents constructing staging level tables.

Data Warehouse

Here, information from the staging area is converted and fed into the data warehouse as dimensions and facts, which are then utilized for business intelligence.

Data warehouse design & development



Above diagram shows how the dimension tables and fact table was combined. Following were considered when developing the data warehouse dimensional model;

- Snowflake schema type was used
- **Dimensions**
  - ➢ Hierarchical dimensions
    1. Venue – Country name – City name – Venue name
    2. BowlingStyle – Bowling Type – Bowling Style
    3. BallData – Match number – Innings number – Over number – Ball number 4. Date
  - ➢ **Slowly changing dimensions**
    1. Team – Team name
- **Fact Table**
  - ➢ BallByBall
    This table consists of 12 foreign key columns which are connected to the dimensions of the model.
- **Assumptions**
  - ➢ Since the names of the teams are changed by owners when needed, Team was considered as a slowly changing dimension to tack the details of the team names.

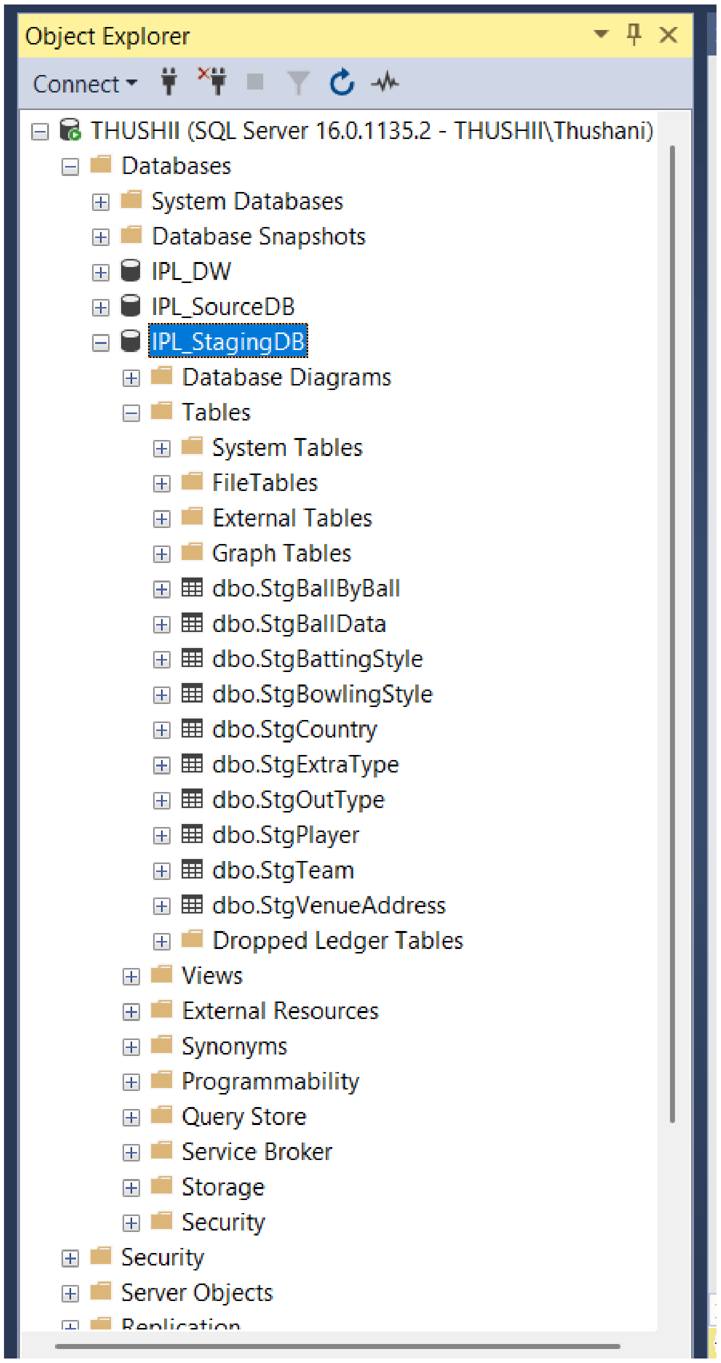
ETL development

Extract

First, all the data which mentioned in the Preparation of Data Sources step were imported to the staging database (IPL_Staging) by using relevant connections and the sources.

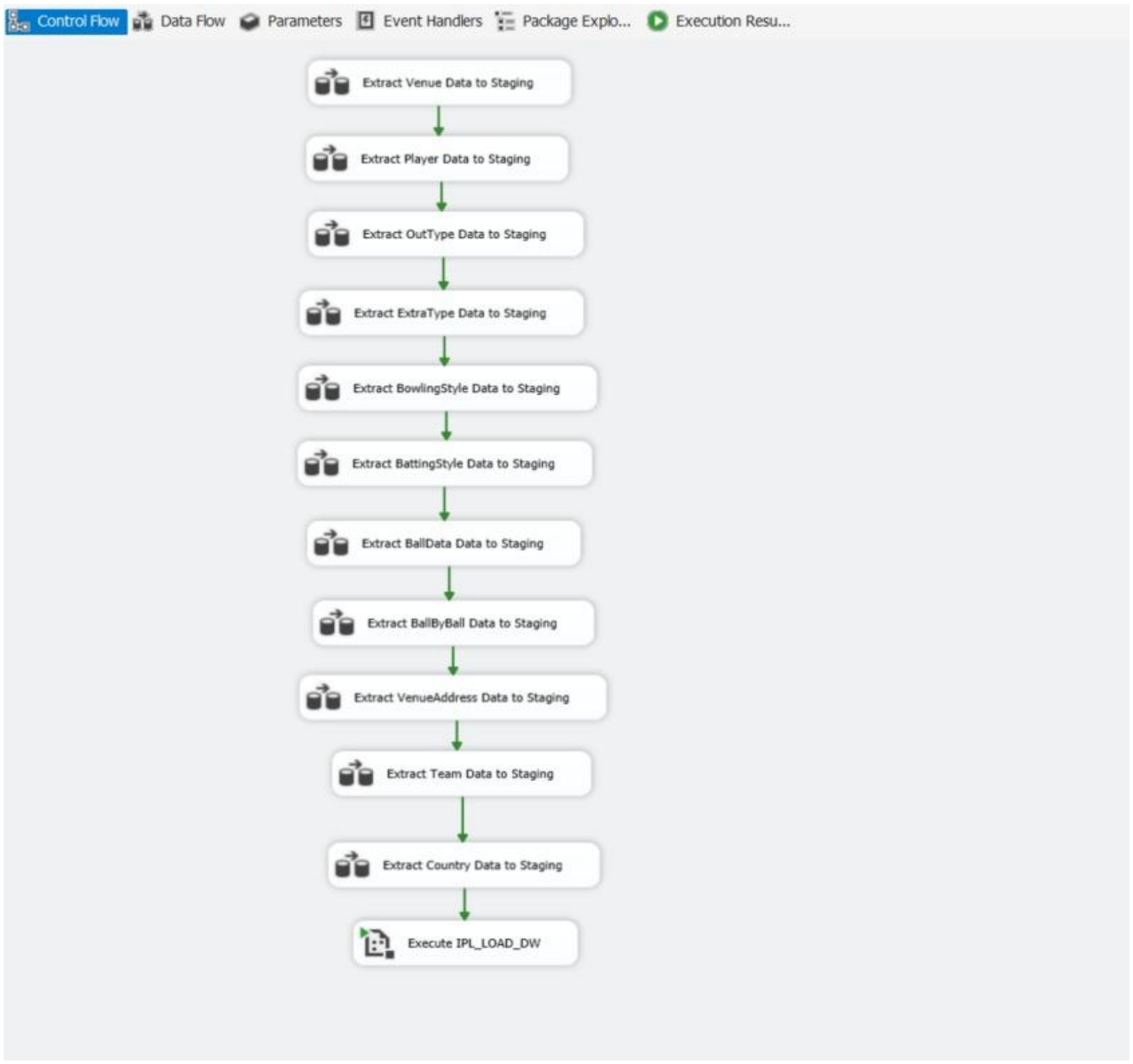
Below image shows the tables of the staging database;

SSMS staging database (IPL_StagingDB)



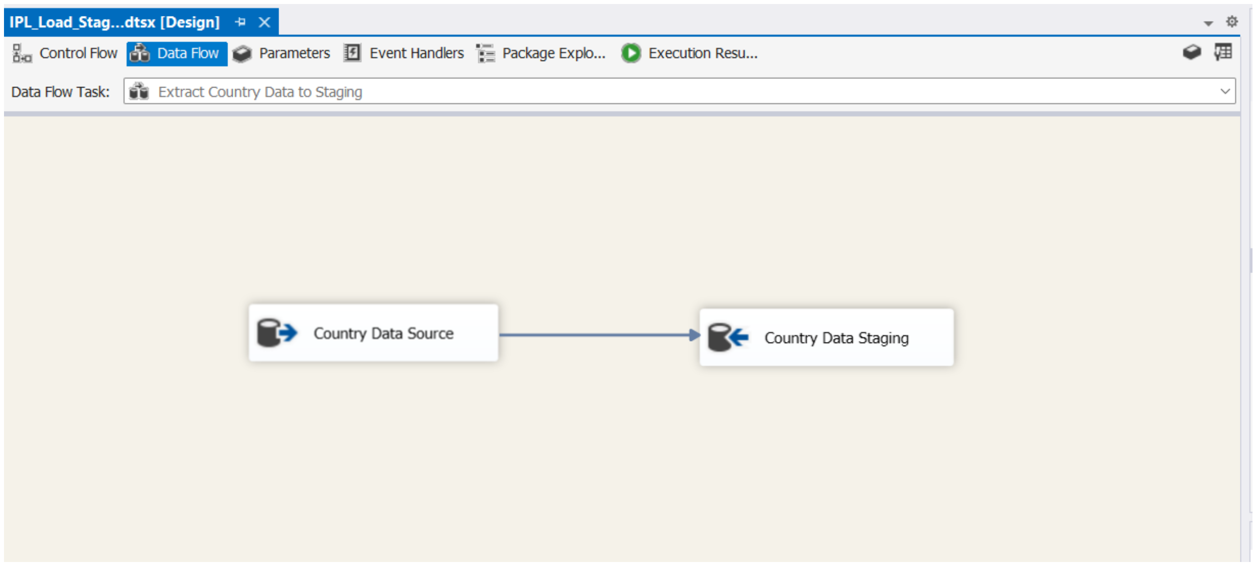
THUSHII (SQL Server 16.0.1135.2 - THUSHII\Thushani)
- Databases
  - System Databases
  - Database Snapshots
  - IPL_DW
  - IPL_SourceDB
  - IPL_StagingDB
    - Database Diagrams
    - Tables
      - System Tables
      - FileTables
      - External Tables
      - Graph Tables
      - dbo.StgBallByBall
      - dbo.StgBallData
      - dbo.StgBattingStyle
      - dbo.StgBowlingStyle
      - dbo.StgCountry
      - dbo.StgExtraType
      - dbo.StgOutType
      - dbo.StgPlayer
      - dbo.StgTeam
      - dbo.StgVenueAddress
      - Dropped Ledger Tables
    - Views
    - External Resources
    - Synonyms
    - Programmability
    - Query Store
    - Service Broker
    - Storage
    - Security
- Security
- Server Objects
- Replication

Control flow of extraction

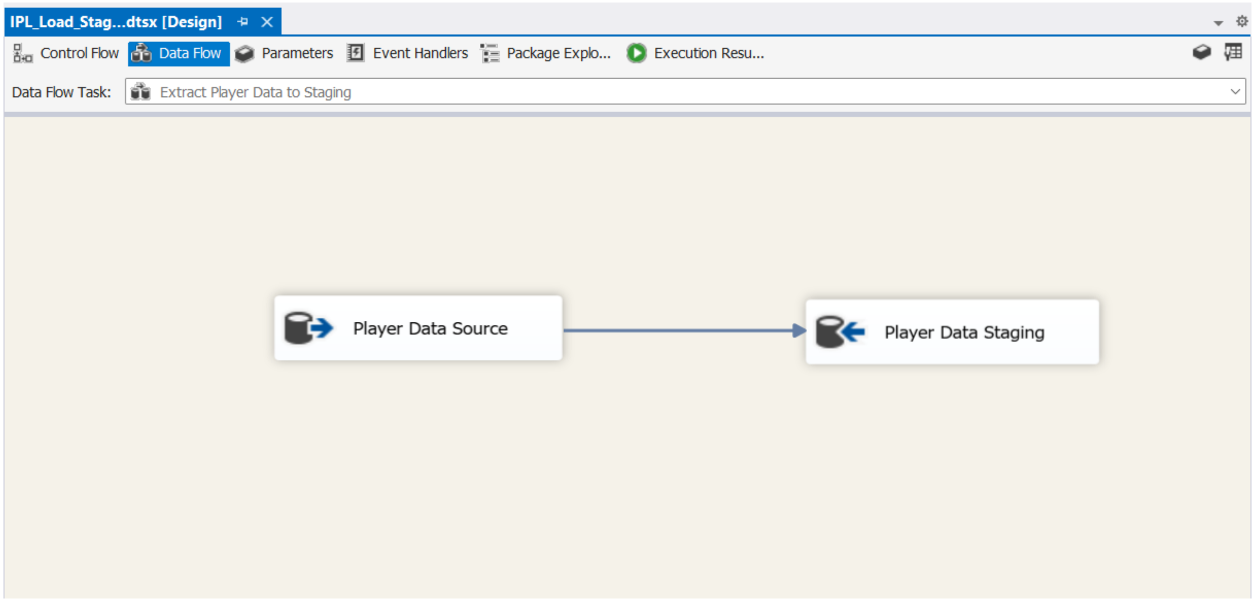

Control Flow
Data Flow
Parameters
Event Handlers
Package Explo...
Execution Resu...
Extract Venue Data to Staging
Extract Player Data to Staging
Extract OutType Data to Staging
Extract ExtraType Data to Staging
Extract BowlingStyle Data to Staging
Extract BattingStyle Data to Staging
Extract BallData Data to Staging
Extract BallByBall Data to Staging
Extract VenueAddress Data to Staging
Extract Team Data to Staging
Extract Country Data to Staging
Execute IPL_LOAD_DW

Data Flows of Staging Tables Screenshots

- **Country Data Flow**

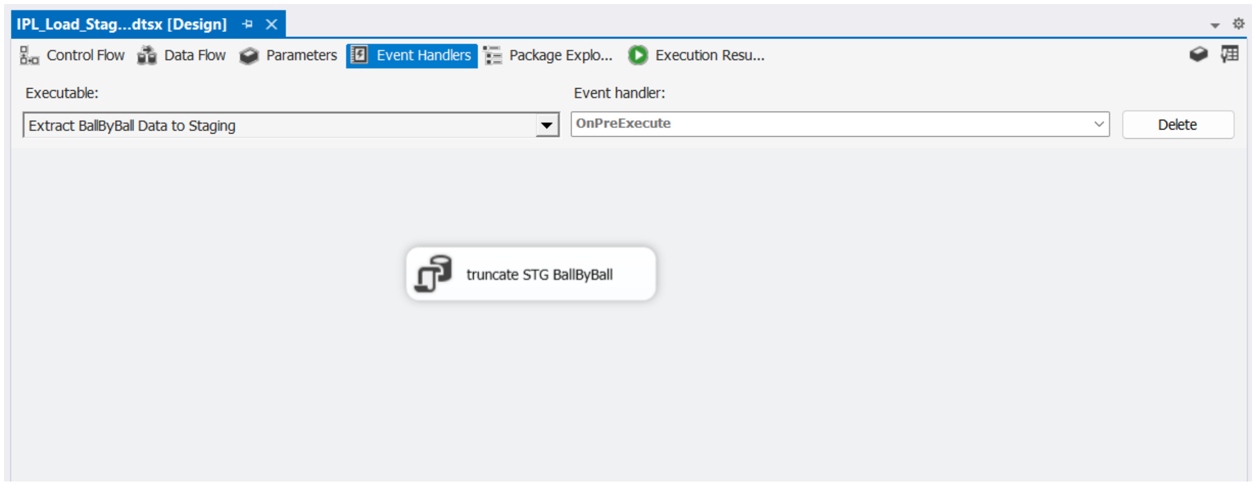


- **Player Data Flow**



Event Handlers of Staging Tables Screenshots

- **Truncate BallByBall Data Staging**

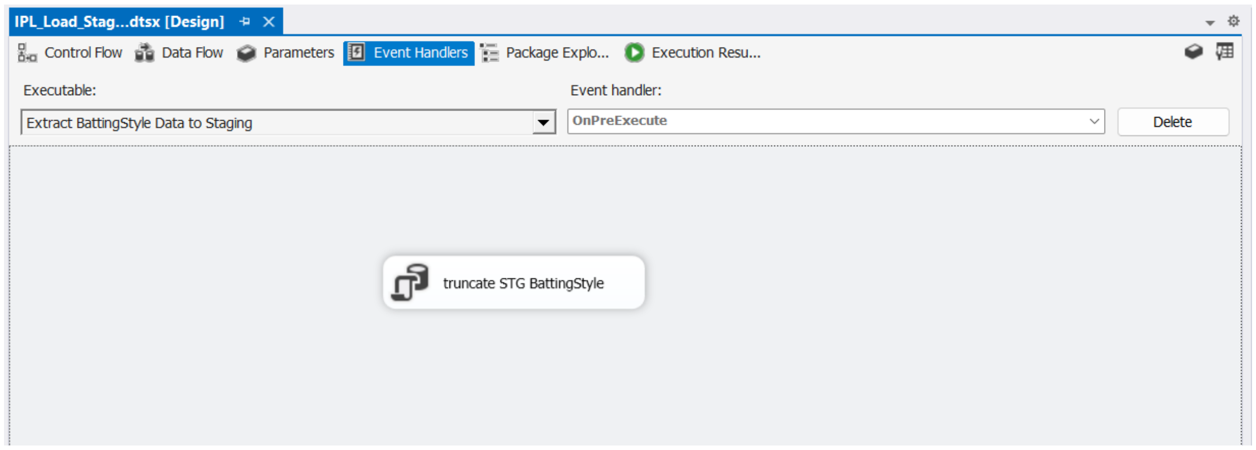


Executable: Extract BallByBall Data to Staging

Event handler: OnPreExecute

truncate STG BallByBall

- **Truncate BattingStyle Staging**



Executable: Extract BattingStyle Data to Staging

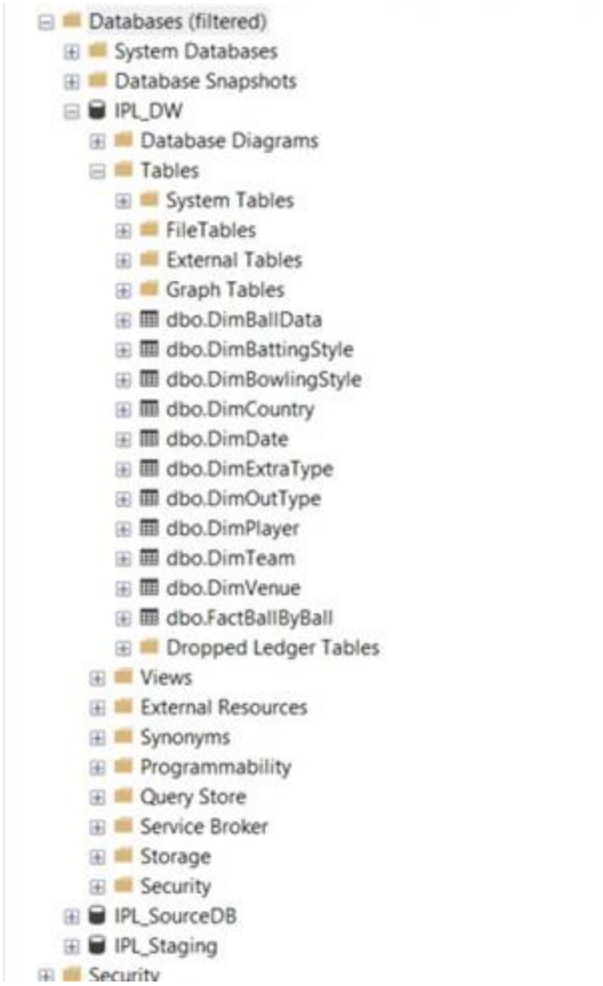
Event handler: OnPreExecute

truncate STG BattingStyle

Transform and load

The staging area's data was then converted and transferred to the data warehouse (IPL_DW). Prior to loading the data in the appropriate order, the dimension tables and fact tables were established.
Data was transformed and loaded into a data warehouse using operations including merge join, lookup, derived columns, and sort.

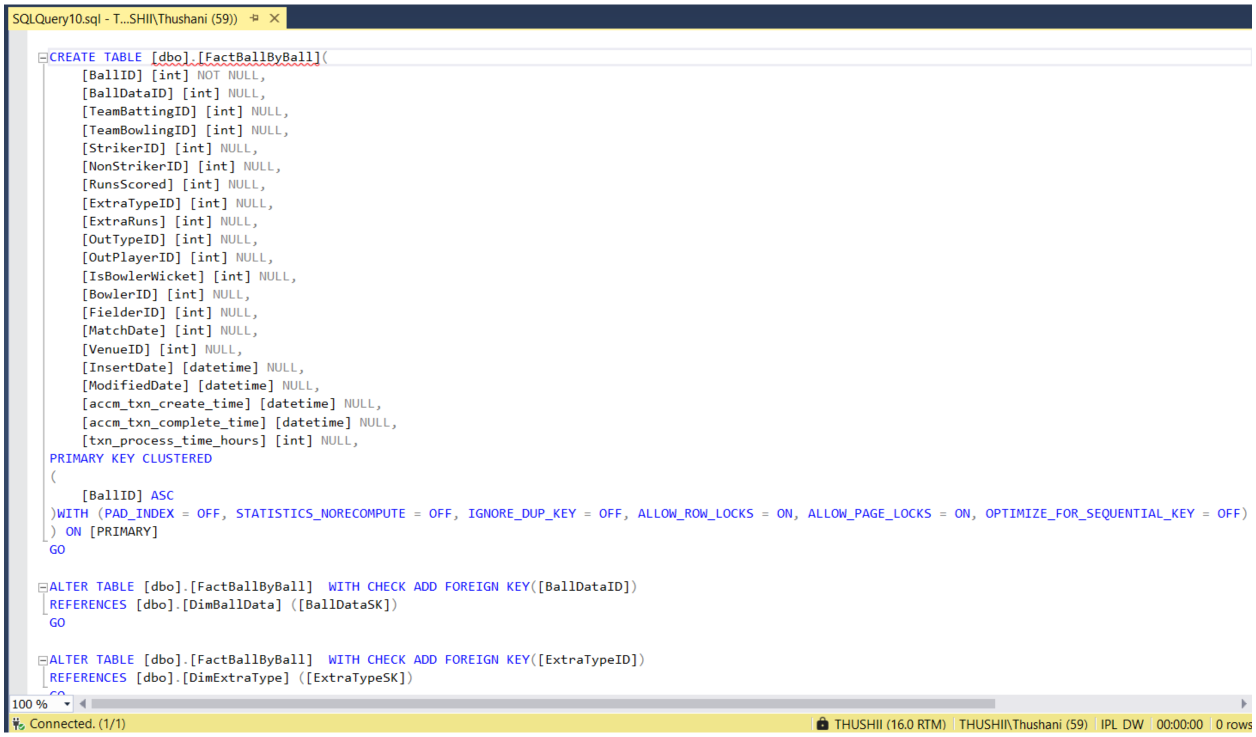
SSMS data warehouse (IPL_DW)



```
⊟ Databases (filtered)
  ⊞ System Databases
  ⊞ Database Snapshots
  ⊟ IPL_DW
    ⊞ Database Diagrams
    ⊟ Tables
      ⊞ System Tables
      ⊞ FileTables
      ⊞ External Tables
      ⊞ Graph Tables
      ⊞ dbo.DimBallData
      ⊞ dbo.DimBattingStyle
      ⊞ dbo.DimBowlingStyle
      ⊞ dbo.DimCountry
      ⊞ dbo.DimDate
      ⊞ dbo.DimExtraType
      ⊞ dbo.DimOutType
      ⊞ dbo.DimPlayer
      ⊞ dbo.DimTeam
      ⊞ dbo.DimVenue
      ⊞ dbo.FactBallByBall
      ⊞ Dropped Ledger Tables
    ⊞ Views
    ⊞ External Resources
    ⊞ Synonyms
    ⊞ Programmability
    ⊞ Query Store
    ⊞ Service Broker
    ⊞ Storage
    ⊞ Security
  ⊞ IPL_SourceDB
  ⊞ IPL_Staging
⊞ Security
```

Sql Queries ScreenShots to create Fact Table and Dimension Tables

- **Fact Table's SQL Query**



SQLQuery10.sql - T...SHII\Thushani (59))

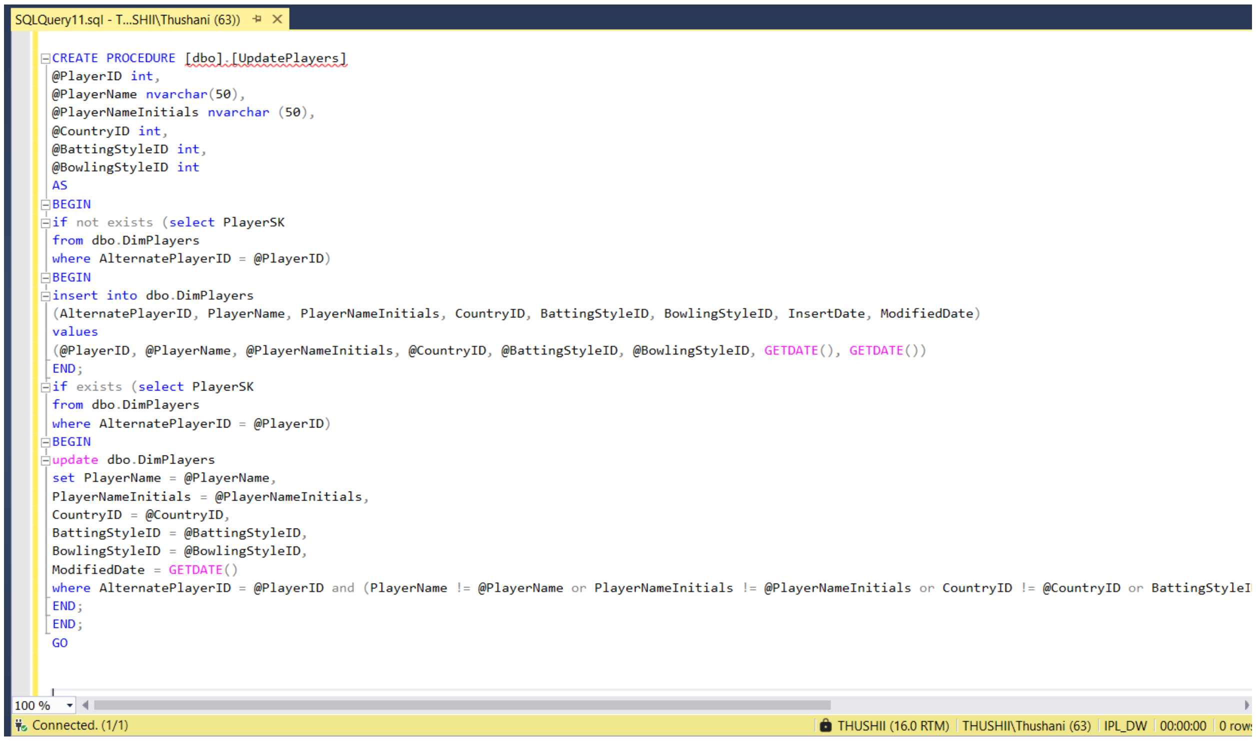
```
CREATE TABLE [dbo].[FactBallByBall](
	[BallID] [int] NOT NULL,
	[BallDataID] [int] NULL,
	[TeamBattingID] [int] NULL,
	[TeamBowlingID] [int] NULL,
	[StrikerID] [int] NULL,
	[NonStrikerID] [int] NULL,
	[RunsScored] [int] NULL,
	[ExtraTypeID] [int] NULL,
	[ExtraRuns] [int] NULL,
	[OutTypeID] [int] NULL,
	[OutPlayerID] [int] NULL,
	[IsBowlerWicket] [int] NULL,
	[BowlerID] [int] NULL,
	[FielderID] [int] NULL,
	[MatchDate] [int] NULL,
	[VenueID] [int] NULL,
	[InsertDate] [datetime] NULL,
	[ModifiedDate] [datetime] NULL,
	[accm_txn_create_time] [datetime] NULL,
	[accm_txn_complete_time] [datetime] NULL,
	[txn_process_time_hours] [int] NULL,
PRIMARY KEY CLUSTERED
(
	[BallID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF)
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[FactBallByBall]  WITH CHECK ADD FOREIGN KEY([BallDataID])
REFERENCES [dbo].[DimBallData] ([BallDataSK])
GO

ALTER TABLE [dbo].[FactBallByBall]  WITH CHECK ADD FOREIGN KEY([ExtraTypeID])
REFERENCES [dbo].[DimExtraType] ([ExtraTypeSK])
```

100 %

Connected. (1/1) | THUSHII (16.0 RTM) | THUSHII\Thushani (59) | IPL DW | 00:00:00 | 0 rows

Sql Procedures ScreenShots

- **Procedure for dimPlayers**



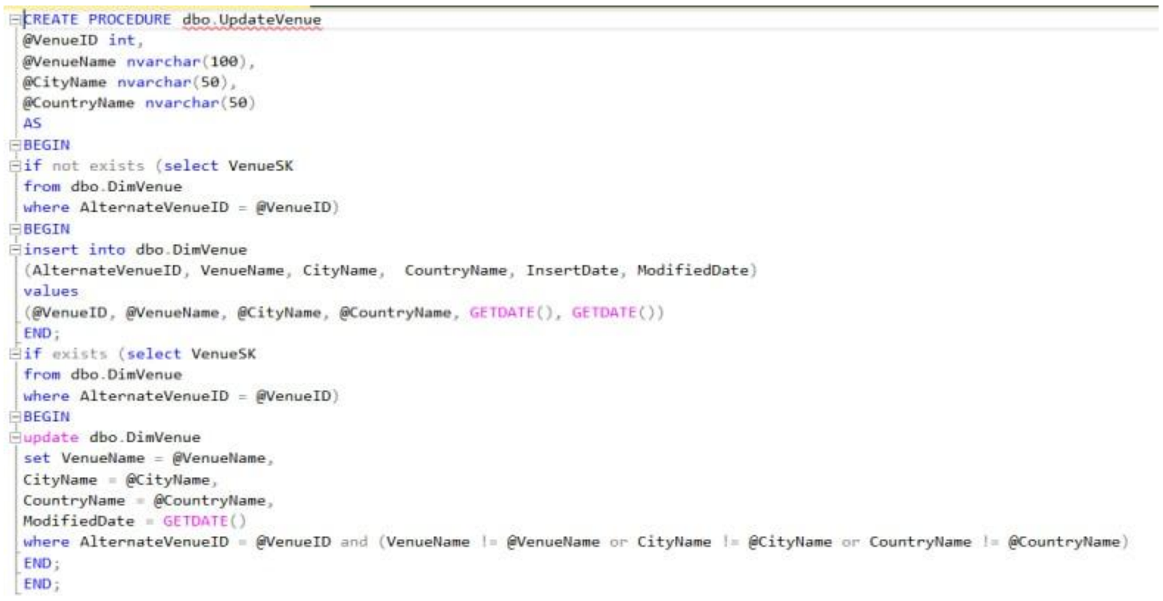
SQLQuery11.sql - T...SHII\Thushani (63))

```
CREATE PROCEDURE [dbo].[UpdatePlayers]
@PlayerID int,
@PlayerName nvarchar(50),
@PlayerNameInitials nvarchar (50),
@CountryID int,
@BattingStyleID int,
@BowlingStyleID int
AS
BEGIN
if not exists (select PlayerSK
from dbo.DimPlayers
where AlternatePlayerID = @PlayerID)
BEGIN
insert into dbo.DimPlayers
(AlternatePlayerID, PlayerName, PlayerNameInitials, CountryID, BattingStyleID, BowlingStyleID, InsertDate, ModifiedDate)
values
(@PlayerID, @PlayerName, @PlayerNameInitials, @CountryID, @BattingStyleID, @BowlingStyleID, GETDATE(), GETDATE())
END;
if exists (select PlayerSK
from dbo.DimPlayers
where AlternatePlayerID = @PlayerID)
BEGIN
update dbo.DimPlayers
set PlayerName = @PlayerName,
PlayerNameInitials = @PlayerNameInitials,
CountryID = @CountryID,
BattingStyleID = @BattingStyleID,
BowlingStyleID = @BowlingStyleID,
ModifiedDate = GETDATE()
where AlternatePlayerID = @PlayerID and (PlayerName != @PlayerName or PlayerNameInitials != @PlayerNameInitials or CountryID != @CountryID or BattingStyleI
END;
END;
GO
```

100 %

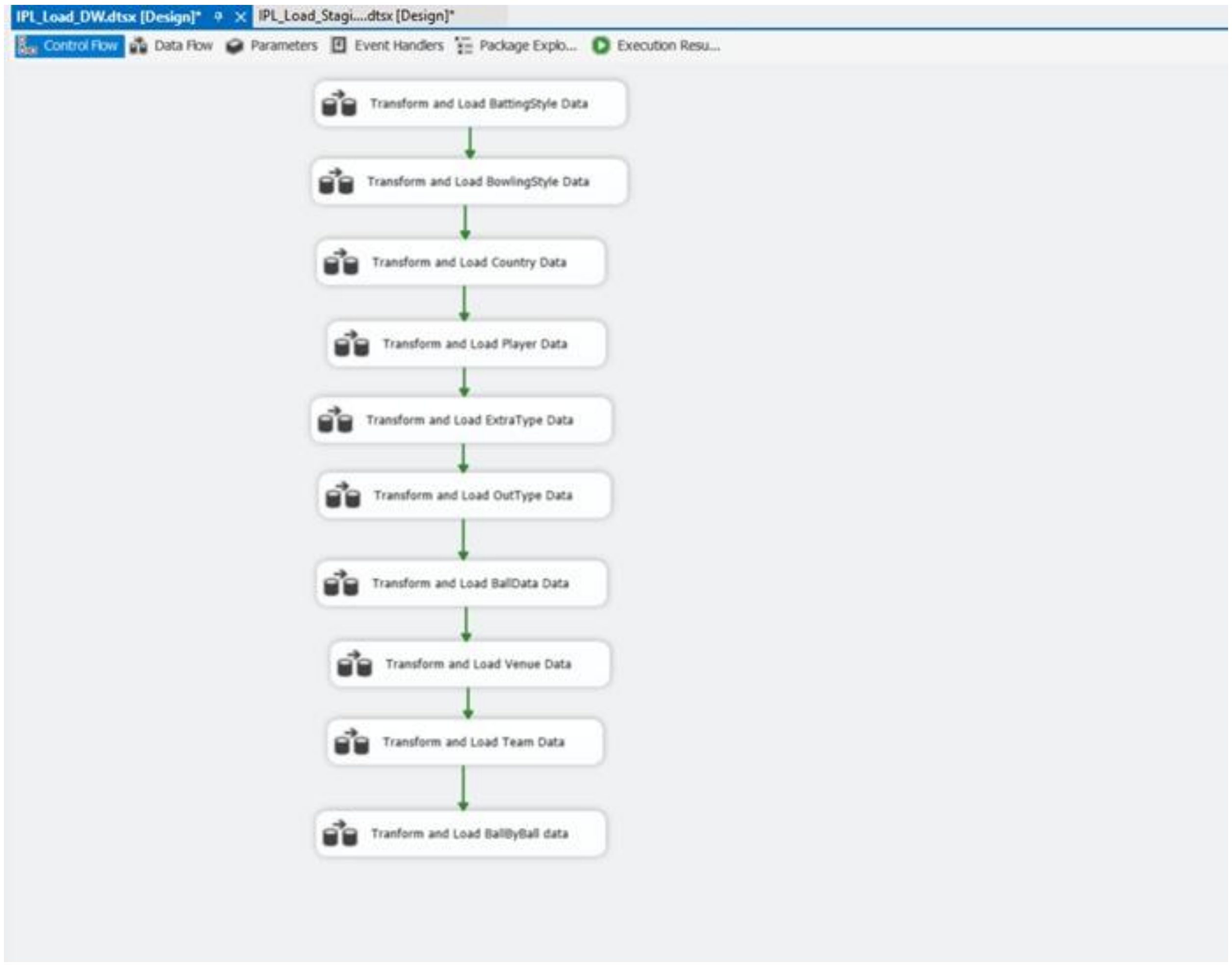
Connected. (1/1) | THUSHII (16.0 RTM) | THUSHII\Thushani (63) | IPL_DW | 00:00:00 | 0 rows

- **Procedure for dimVenue**



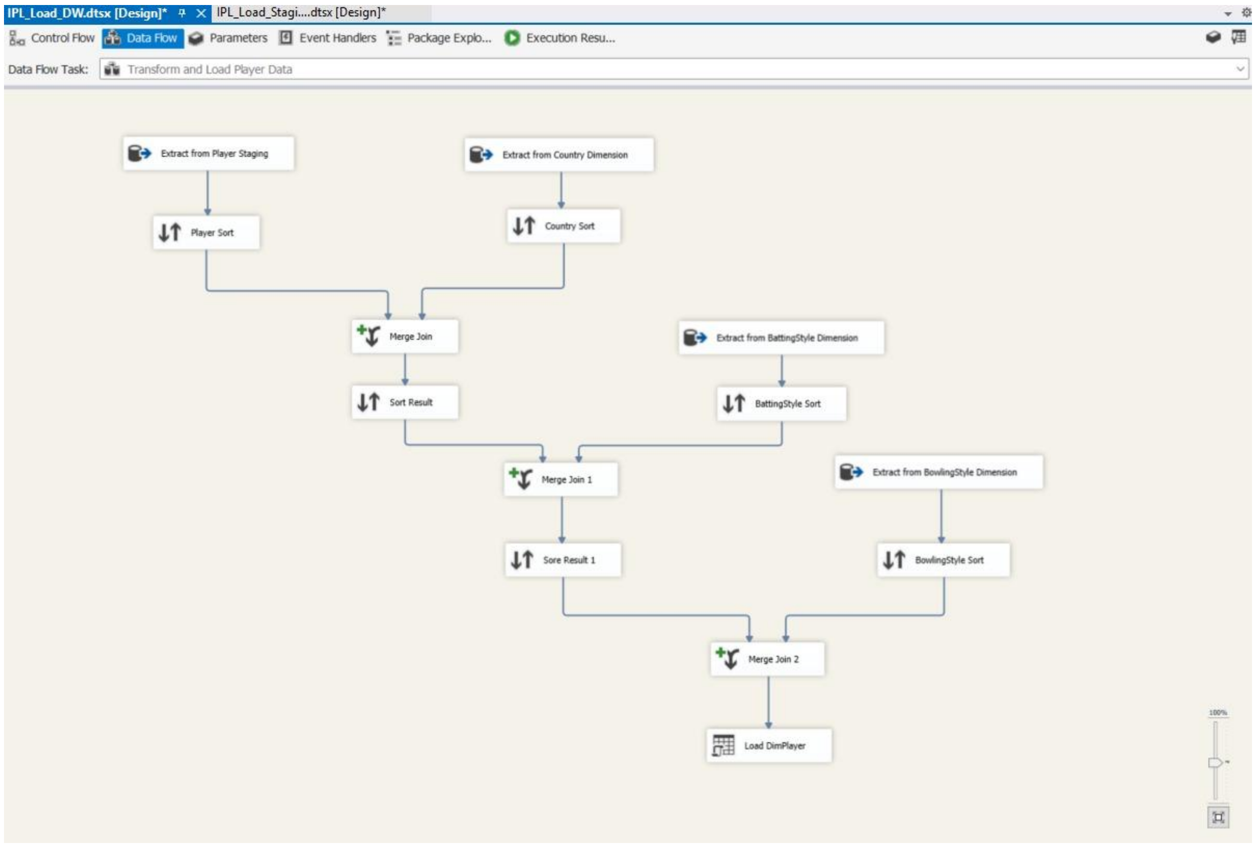
```sql
CREATE PROCEDURE dbo.UpdateVenue
@VenueID int,
@VenueName nvarchar(100),
@CityName nvarchar(50),
@CountryName nvarchar(50)
AS
BEGIN
if not exists (select VenueSK
from dbo.DimVenue
where AlternateVenueID = @VenueID)
BEGIN
insert into dbo.DimVenue
(AlternateVenueID, VenueName, CityName,  CountryName, InsertDate, ModifiedDate)
values
(@VenueID, @VenueName, @CityName, @CountryName, GETDATE(), GETDATE())
END;
if exists (select VenueSK
from dbo.DimVenue
where AlternateVenueID = @VenueID)
BEGIN
update dbo.DimVenue
set VenueName = @VenueName,
CityName = @CityName,
CountryName = @CountryName,
ModifiedDate = GETDATE()
where AlternateVenueID = @VenueID and (VenueName != @VenueName or CityName != @CityName or CountryName != @CountryName)
END;
END;
```

Control flow extraction

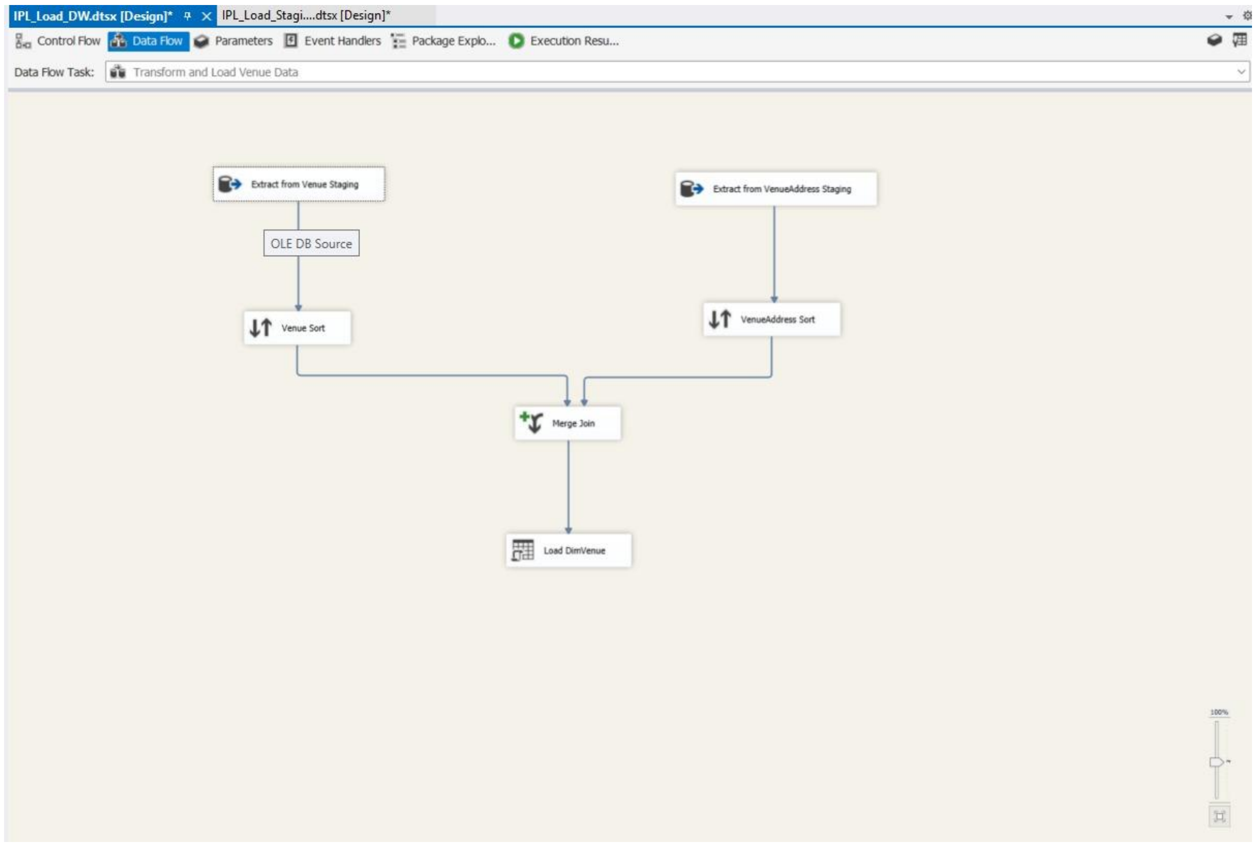

IPL_Load_DW.dtsx [Design]*
IPL_Load_Stagi....dtsx [Design]*
Control Flow
Data Flow
Parameters
Event Handlers
Package Explo...
Execution Resu...
Transform and Load BattingStyle Data
Transform and Load BowlingStyle Data
Transform and Load Country Data
Transform and Load Player Data
Transform and Load ExtraType Data
Transform and Load OutType Data
Transform and Load BallData Data
Transform and Load Venue Data
Transform and Load Team Data
Tranform and Load BallByBall data

Data Flows of Data Warehouse Tables Screenshots

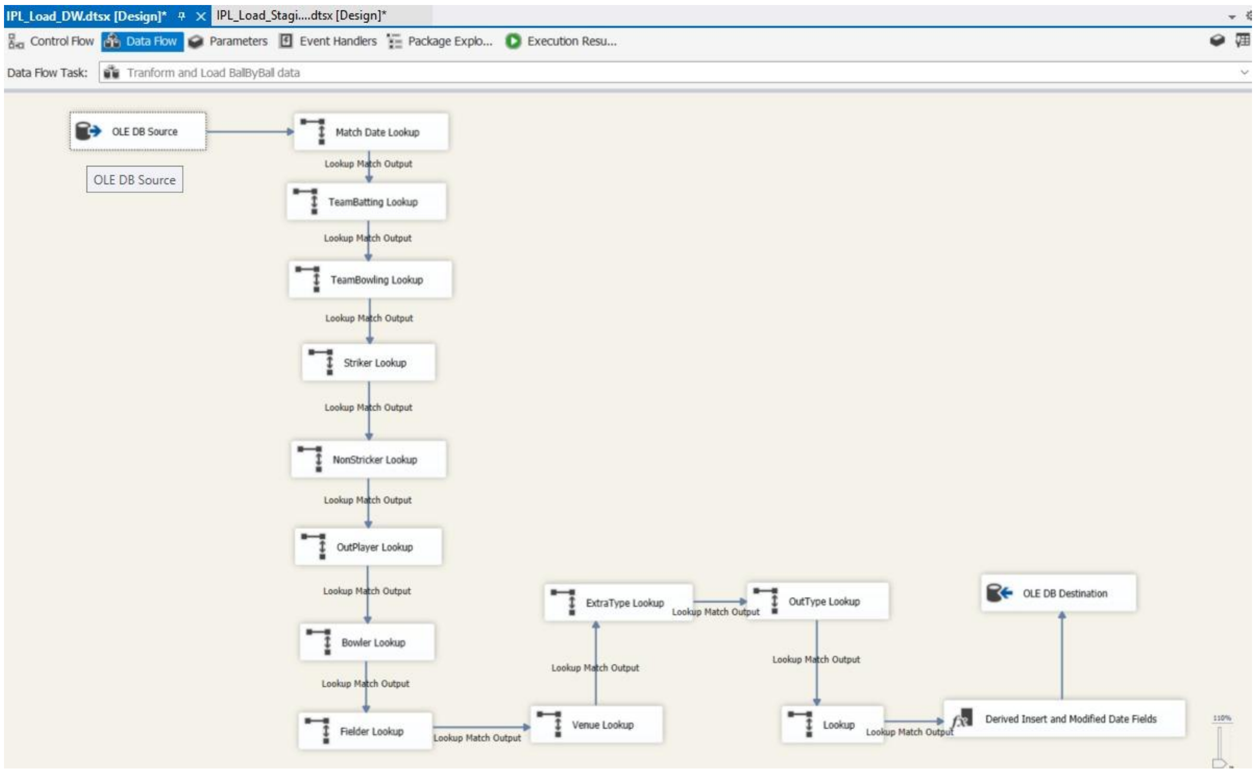
- **DimPlayers Transform and Load**



- **DimVenue Transform and load**



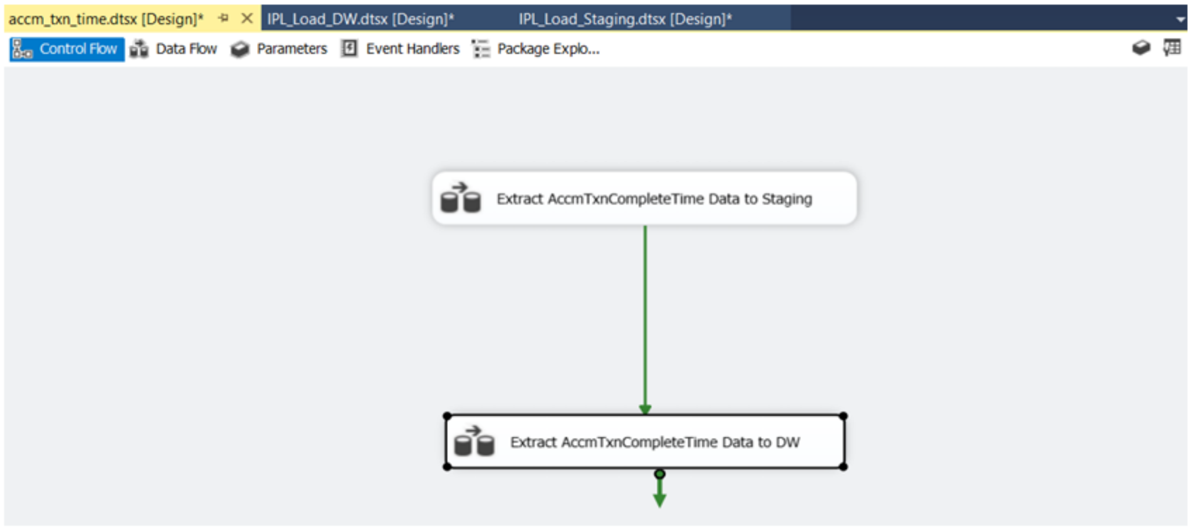
- **Fact BallByBall Transform and load**



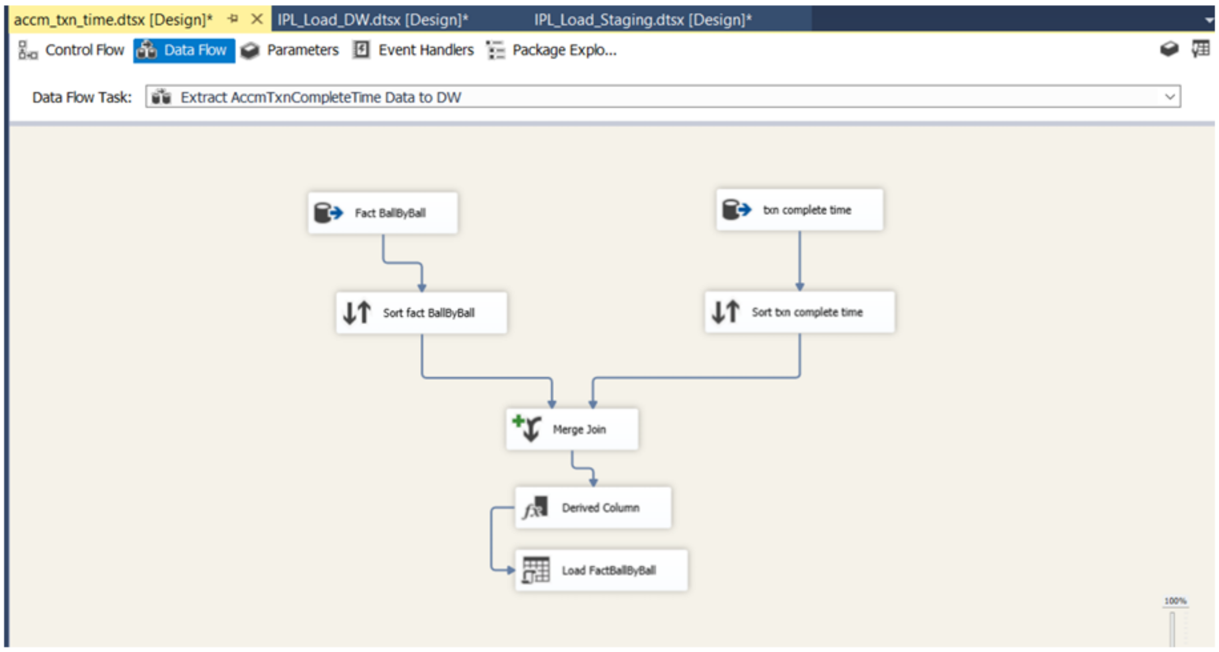
ETL development – Accumulating fact tables

An external csv data source was used for this step and relevant coulombs were created in fact table. Data was transformed and loaded to these fields using separate ETL process.

Control flow extraction


accm_txn_time.dtsx [Design]*
IPL_Load_DW.dtsx [Design]*
IPL_Load_Staging.dtsx [Design]*
Control Flow
Data Flow
Parameters
Event Handlers
Package Explo...
Extract AccmTxnCompleteTime Data to Staging
Extract AccmTxnCompleteTime Data to DW

Transform and load to Fact BallByBall


accm_txn_time.dtsx [Design]*
IPL_Load_DW.dtsx [Design]*
IPL_Load_Staging.dtsx [Design]*
Control Flow
Data Flow
Parameters
Event Handlers
Package Explo...
Data Flow Task:
Extract AccmTxnCompleteTime Data to DW
Fact BallByBall
txn complete time
Sort fact BallByBall
Sort txn complete time
Merge Join
Derived Column
Load FactBallByBall
100%