



Sri Lanka Institute of Information Technology

Assignment I

Data Warehouse & Business

Intelligence 2022

Submitted By:

Rasheed.A

IT20251178

Contents

Data set selection & Preparation.....	1
Solution Architecture.....	5
Data warehouse design and development.....	7
ETL Development.....	8
Data Warehouse Development.....	12
Database Structures.....	24

Description of the dataset

<https://www.kaggle.com/datasets/rohanrao/formula-1-world-championship-1950-2020>

- **Circuits:** Contain details about circuits where F1 races are held.
- **Constructors:** Contain details about Constructors in F1
- **Drivers:** Contain details about Drivers in F1
- **Races:** Contain details about Races in F1
- **Results:** Contain details about Results of F1 races
- **Statuses:** Contain details about Mapping of various statuses.
- **Seasons:** Contain details about Seasons of F1

```
erDiagram
    circuits ||--o{ races : "has"
    races ||--o{ constructors : "has"
    seasons ||--o{ results : "has"
    results ||--o{ drivers : "has"
    results ||--o{ status : "has"
    races ||--o{ results : "has"

    circuits {
        string circuitId PK
        string circuitRef
        string name
        string location
        string country
        float lat
        float lng
        float alt
        string url
    }

    races {
        string raceId PK
        int year
        int round
        string circuitId FK
        string name
        string date
        string time
        string url
    }

    constructors {
        string constructorId PK
        string constructorRef
        string name
        string nationality
        string url
    }

    seasons {
        int year PK
        string url
    }

    results {
        string resultId PK
        string raceId FK
        string driverId FK
        string constructorId FK
        int number
        int grid
        string position
        string positionText
        int positionOrder
        float points
        int laps
        float time
        float milliseconds
        string fastestLap
        int rank
        float fastestLapTime
        float fastestLapSpeed
        string statusId FK
    }

    drivers {
        string driverId PK
        string driverRef
        int number
        string code
        string forename
        string surname
        string dob
        string nationality
        string url
    }

    status {
        string statusId PK
        string status
    }
```

Note

Reasons for selecting this source:

- I found measurable data that can be included in the fact tables.
- I found descriptive data that can be included in the dimension tables.
- There are a rich set of ETL tasks that can be performed using the chosen dataset (explained in this document)
- I will be able of making reports from the chosen dataset

Preparation of data sources

The following table shows the information's available in source and the format of it.

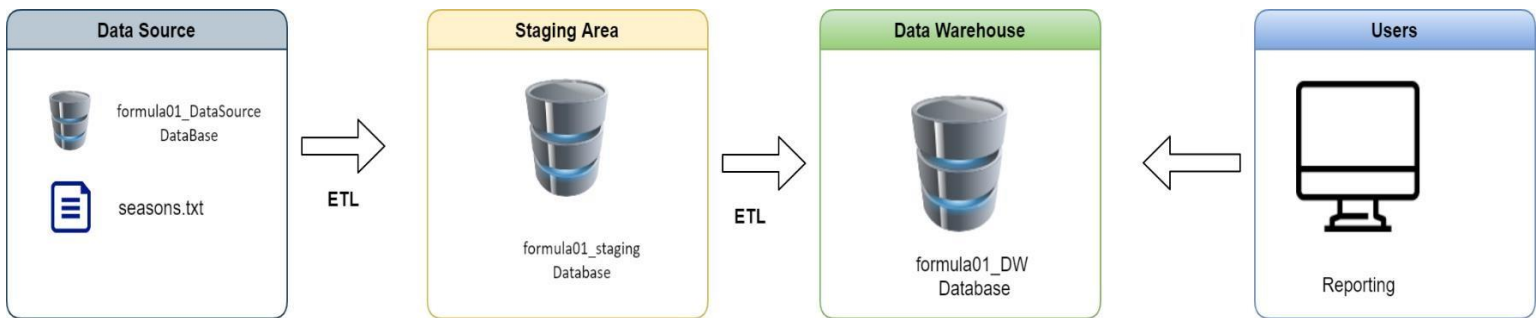
Source	Source Type	Object Name	Schema	Object Type	Description
Formula01_DataSource	SQL Database	circuits	dbo	Table	Contain details about circuits where F1 races are held.
		constructors	dbo	Table	Contain details about Constructors in F1
		drivers	dbo	Table	Contain details about Drivers in F1
		races	dbo	Table	Contain details about Races in F1
		results	dbo	Table	Contain details about Results of F1 races
		status	dbo	Table	Contain details about Mapping of various statuses
Seasons	Text File				Contain details about Seasons of F1

Note

- I have used two source type as shown in the above diagram which are SQL database and text file.

High-level DW & BI solution architecture

The below diagram shows the High-level DW & BI solution architecture,



Explanation of each component:

There are two types of sources that I have used for this assignment.

One is a database which contains source data such as,

- circuits
- constructors
- drivers
- races
- results
- status

The other source file is a text file which contain data on

- seasons

In staging area

I have transformed the data source tables as it is in the source database to staging. Also, I transform the season txt file into the staging database.

The below are the tables available in staging

- CircuitStg
- ConstructorStg
- DriverStg
- RaceStg
- ResultStg
- SessionStg
- StatusStg

In Datawarehouse

The below are the table available in Datawarehouse after I do ETL process (ETL is discuss in later pages).

Dimension Tables

- DimCircuit
- DimConstructor
- DimDate
- DimDriver
- DimSeasons
- DimStatus

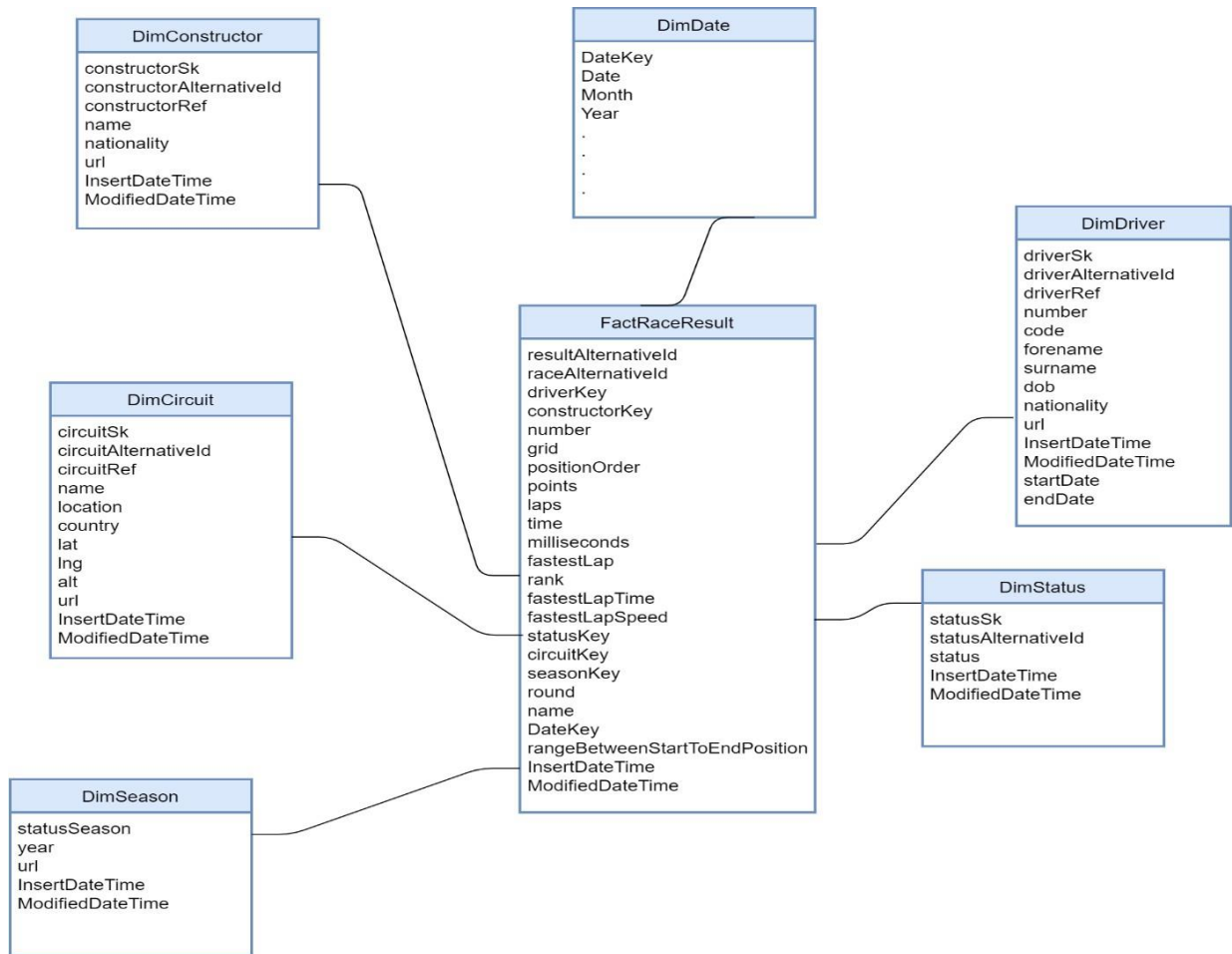
Fact Tables

- FactRaceResult

In User

- We have to create reports (Assignment 2)

Data warehouse design & development



- The above mention diagram has a star schema.
- In my diagram I have 6-dimension tables including the date dimension table.
- It includes one fact table.
- And the Datawarehouse schema has slowing changing dimension in Dim Driver table it consists of startDate and endDate as shown in above diagram.
- Implementation of the Datawarehouse backup file is included in the Datawarehouse folder.

Assumption

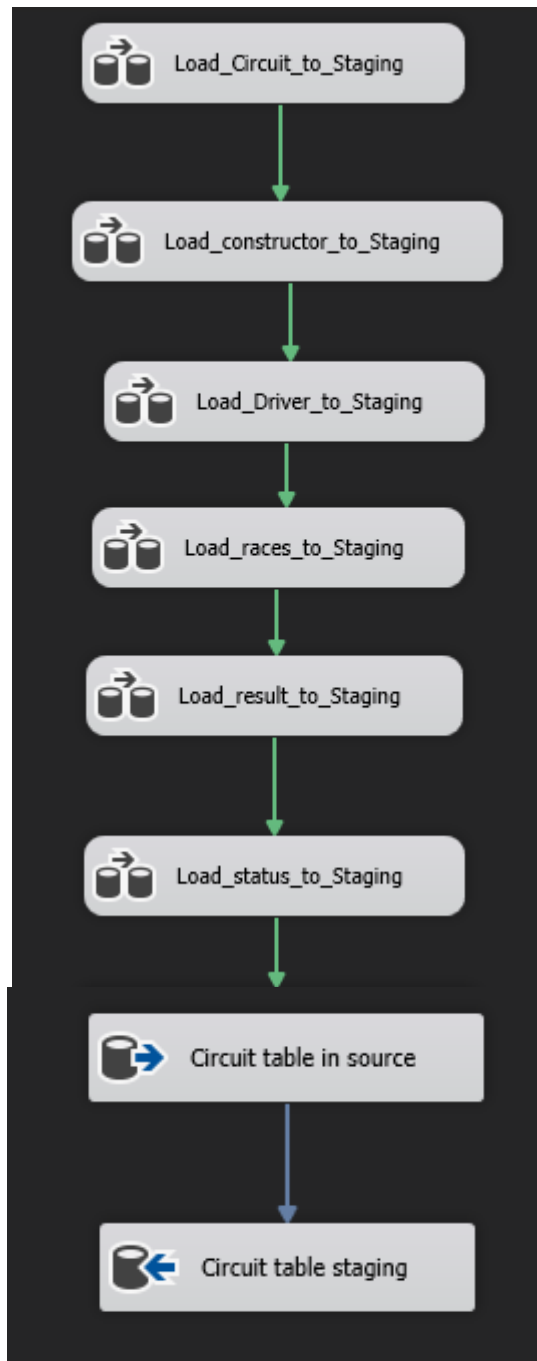
I have assumed that,

- In DimDriver table url as a changing attribute.
- In DimDriver table number and code as an historical attribute.

- In FactRaceResult table I have omitted url and time attribute assuming they are not required for the business
- In all the tables I have added two columns insertDateTime and modifiedDateTime assuming they are important to the business
- In FactRaceResult table I have added a column name “rangeBetweenStartToEndPosition” where it takes the value of “([positionOrder]-[rank])”, assuming that it is a business requirement.

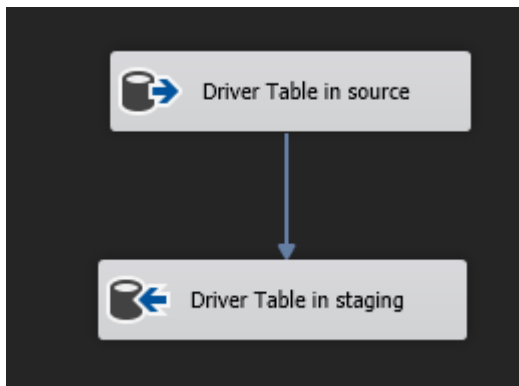
ETL development

In Staging area

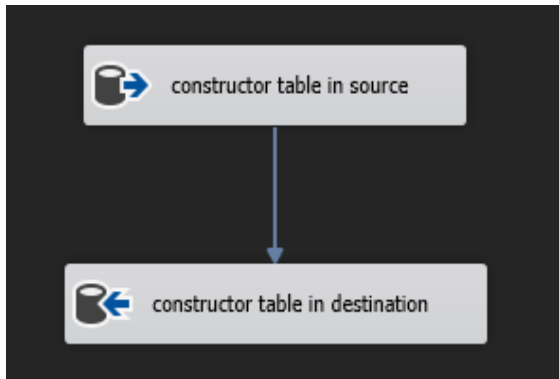


In this diagram it shows the ETL process used to transform source data to staging.

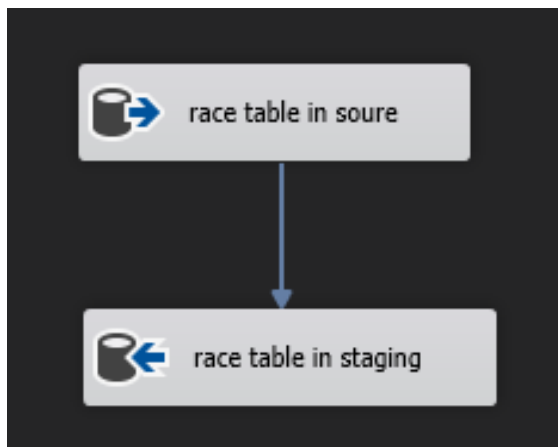
Here data in circuit table in source data database has been tranformed for staging database



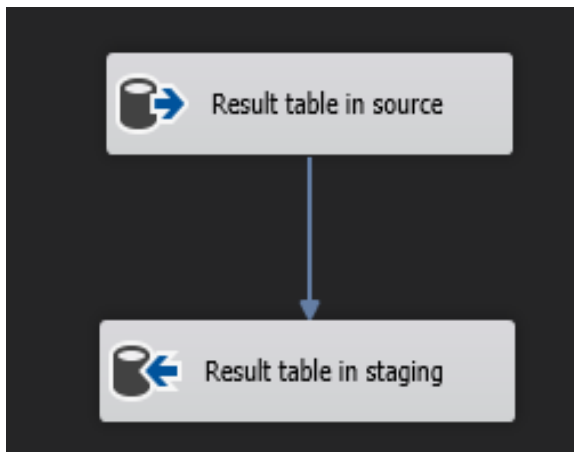
Here data in driver table in data source database are transformed for staging database



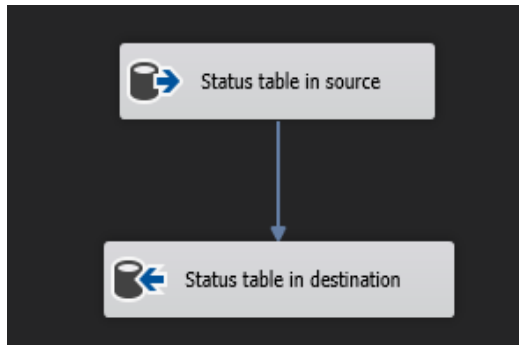
Here data in constructor table in data source database are transformed for staging database



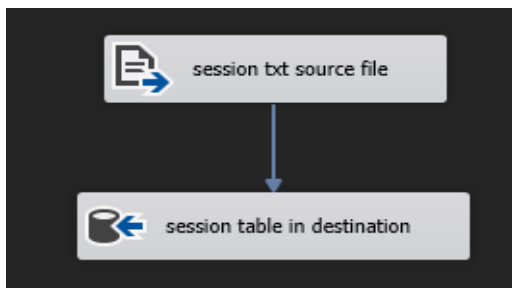
Here data in race table in data source database are transformed for staging database



Here data in result table in data source database are transformed to staging database.

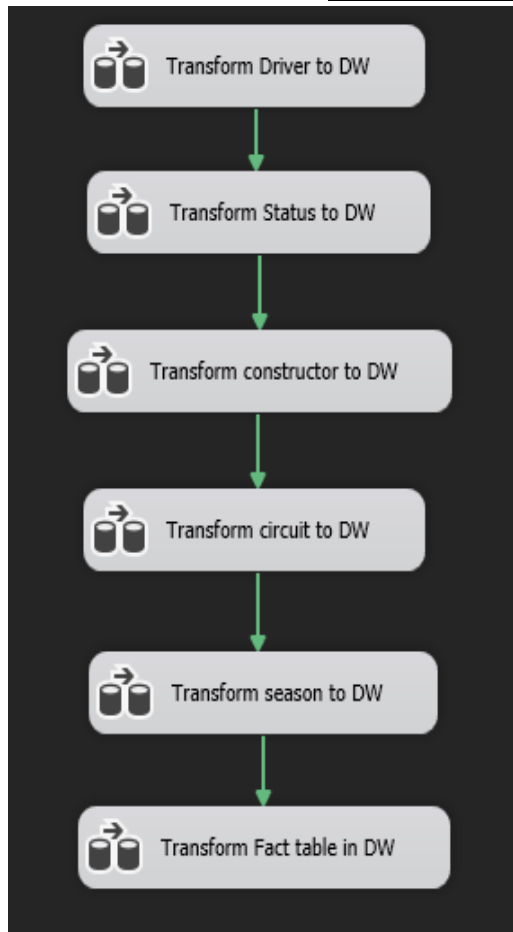


Here data in status table in data source database are transformed to staging database

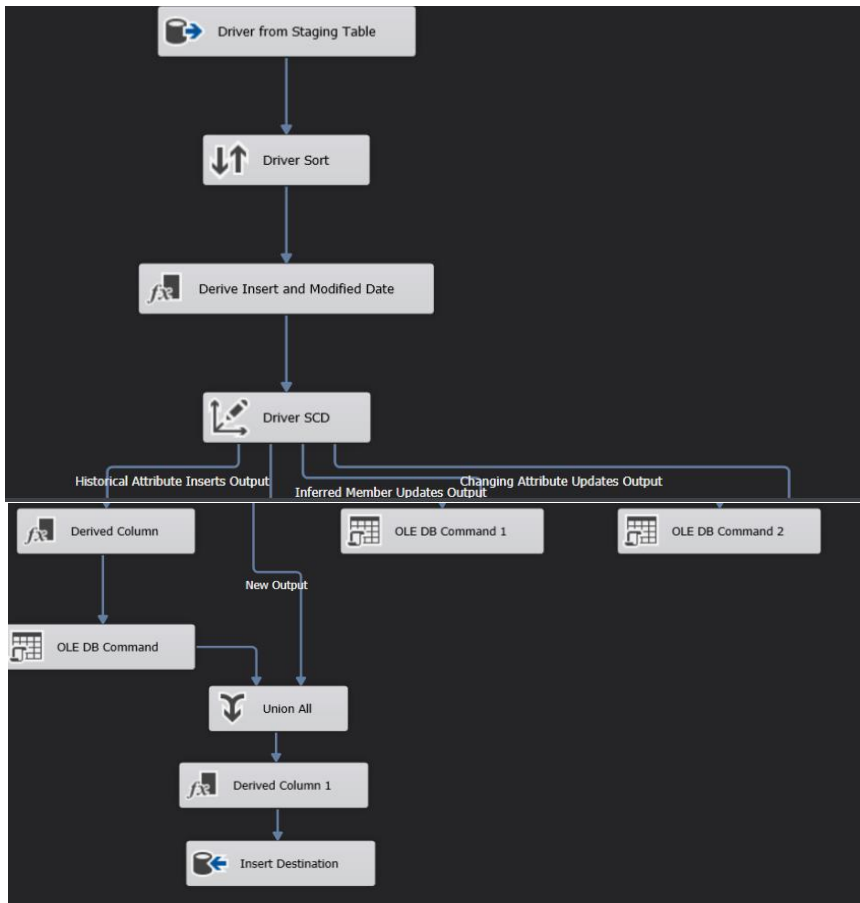


Here data in session txt file are transformed as session table in staging

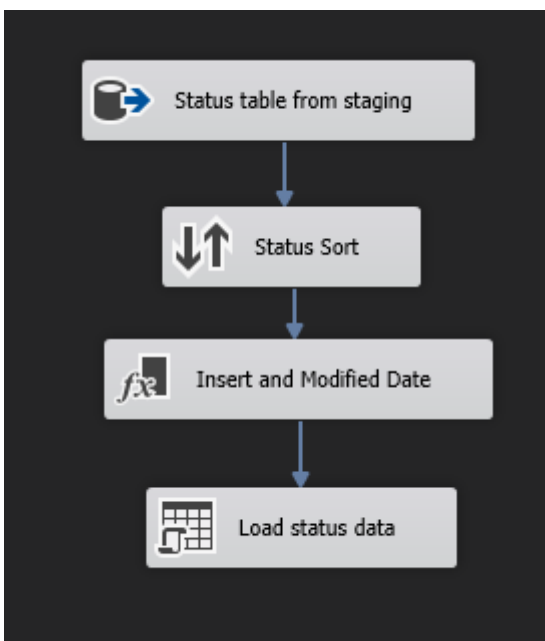
Transform data from staging to data warehouse



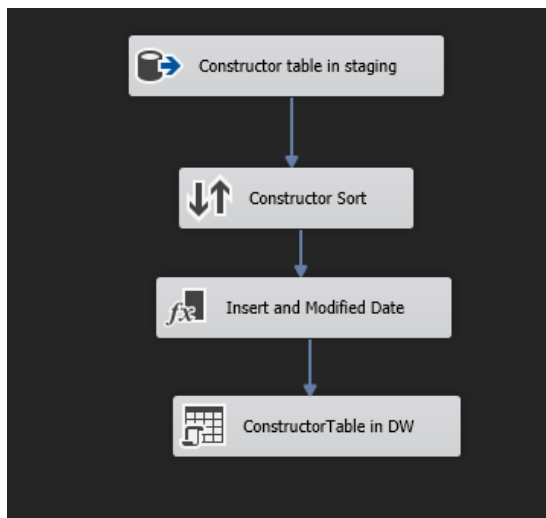
Here data in the staging database are transformed to the data warehouse by using the mentioned data flows



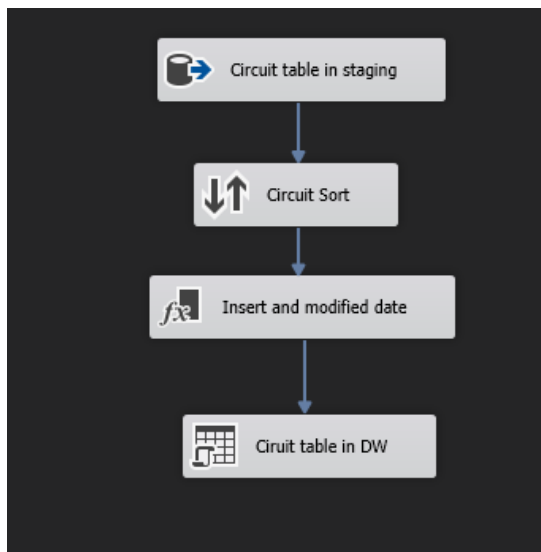
Here I have first loaded the driver table from staging table, after that I sorted it using the driverId, then I created two derived columns as insert date and modified date, after that I have included slowing changing dimension (already explained) finally the data is transformed and load to the driver



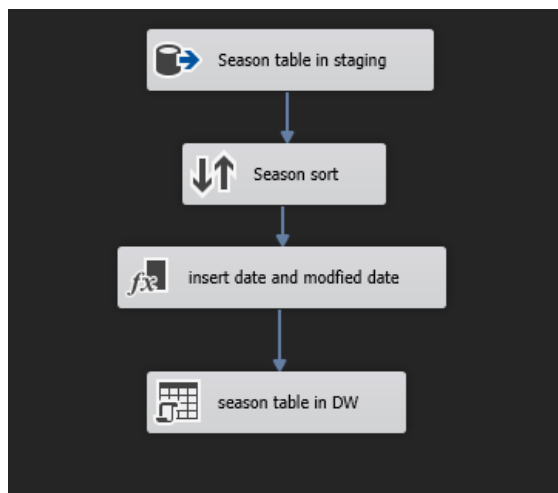
Here I load data from staging database of the table named status and then sorted it. Then I have used dimension columns name as insert date and modified date. Finally, I transform and load data to the status table in Datawarehouse



Here I load data from staging database of the table named constructor and then sorted it. Then I have used dimension columns name as insert date and modified date. Finally, I transform and load data to the constructor table in Datawarehouse



Here I load data from staging database of the table named circuit and then sorted it. Then I have used dimension columns name as insert date and modified date. Finally, I transform and load data to the circuit table in Datawarehouse

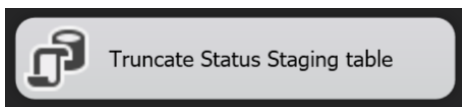
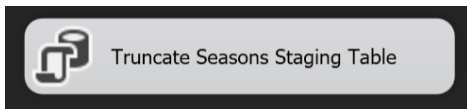
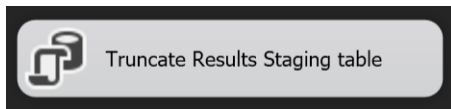
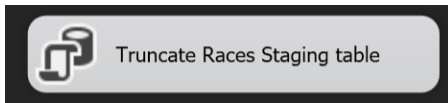
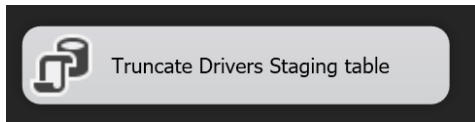
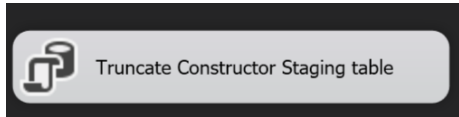
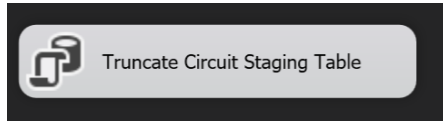


Here I load data from staging database of the table named Season and then sorted it. Then I have used dimension columns name as insert date and modified date. Finally, I transform and load data to the Season table in Datawarehouse



First, I loaded the race table and result table from the staging database then I sorted both according to the race ID, after that I merge the two tables, after that I connect the dimension tables by using lookups. After that I derived two columns named as insert date and modified date. Finally, I load the data to the fact race result table in Datawarehouse database

Below diagrams shows the event handlers I have used in the ETL process to truncate table which are used to avoid duplicate records in table



▼ General	
Name	Truncate Circuit Staging Table
Description	Execute SQL Task
▼ Options	
TimeOut	0
CodePage	1252
TypeConversionMode	Allowed
▼ Result Set	
ResultSet	None
▼ SQL Statement	
ConnectionType	OLE DB
Connection	DESKTOP-NG5FN9Q.formula01_staging
SQLSourceType	Direct input
SQLStatement	truncate table [formula01_staging].[dbo].[StgCircuit]
IsQueryStoredProcedure	False
BypassPrepare	True

▼ General	
Name	Truncate Constructor Staging table
Description	Execute SQL Task
▼ Options	
TimeOut	0
CodePage	1252
TypeConversionMode	Allowed
▼ Result Set	
ResultSet	None
▼ SQL Statement	
ConnectionType	OLE DB
Connection	DESKTOP-NG5FN9Q.formula01_staging
SQLSourceType	Direct input
SQLStatement	truncate table [formula01_staging].[dbo].[StgConstructor]
IsQueryStoredProcedure	False
BypassPrepare	True

▼ General	
Name	Truncate Drivers Staging table
Description	Execute SQL Task
▼ Options	
TimeOut	0
CodePage	1252
TypeConversionMode	Allowed
▼ Result Set	
ResultSet	None
▼ SQL Statement	
ConnectionType	OLE DB
Connection	DESKTOP-NG5FN9Q.formula01_staging
SQLSourceType	Direct input
SQLStatement	truncate table [formula01_staging].[dbo].[StgDrivers]
IsQueryStoredProcedure	False
BypassPrepare	True

▼ General	
Name	Truncate Races Staging table
Description	Execute SQL Task
▼ Options	
TimeOut	0
CodePage	1252
TypeConversionMode	Allowed
▼ Result Set	
ResultSet	None
▼ SQL Statement	
ConnectionType	OLE DB
Connection	DESKTOP-NG5FN9Q.formula01_staging
SQLSourceType	Direct input
SQLStatement	truncate table [formula01_staging].[dbo].[StgRace]
IsQueryStoredProcedure	False
BypassPrepare	True

▼ General	
Name	Truncate Results Staging table
Description	Execute SQL Task
▼ Options	
TimeOut	0
CodePage	1252
TypeConversionMode	Allowed
▼ Result Set	
ResultSet	None
▼ SQL Statement	
ConnectionType	OLE DB
Connection	DESKTOP-NG5FN9Q.formula01_staging
SQLSourceType	Direct input
SQLStatement	truncate table [formula01_staging].[dbo].[StgResults]
IsQueryStoredProcedure	False
BypassPrepare	True

▼ General	
Name	Truncate Status Staging table
Description	Execute SQL Task
▼ Options	
TimeOut	0
CodePage	1252
TypeConversionMode	Allowed
▼ Result Set	
ResultSet	None
▼ SQL Statement	
ConnectionType	OLE DB
Connection	DESKTOP-NG5FN9Q.formula01_staging
SQLSourceType	Direct input
SQLStatement	truncate table [formula01_staging].[dbo].[StgStatus]
IsQueryStoredProcedure	False
BypassPrepare	True

▼ General	
Name	Truncate Seasons Staging Table
Description	Execute SQL Task
▼ Options	
TimeOut	0
CodePage	1252
TypeConversionMode	Allowed
▼ Result Set	
ResultSet	None
▼ SQL Statement	
ConnectionType	OLE DB
Connection	DESKTOP-NG5FN9Q.formula01_staging
SQLSourceType	Direct input
SQLStatement	truncate table [formula01_staging].[dbo].[StgSeason]
IsQueryStoredProcedure	False
BypassPrepare	False
Name Specifies the name of the task.	

So, in the ETL process I have used following transformation task as shown in above diagrams,

- Lookup
- Derived columns
- Merge
- Union
- Sort

The below are the used procedures for the dimension table. The records will be inserted if no existing record available, if an existing record is available, it updates the record with new value

For DimCircuit Table

```
CREATE PROCEDURE [dbo].[UpdateDimCircuit]
@circuitID int,
@circuitRef nvarchar(255),
@name nvarchar(255),
@location nvarchar(255),
@country nvarchar(255),
@lat float,
@lng float,
@alt float,
```

```

@url nvarchar(255)

AS
BEGIN
if not exists (select circuitSk
from [dbo].[DimCircuit]
where circuitAlternativeId = @circuitID)
BEGIN
insert into [dbo].[DimCircuit]
(circuitAlternativeId,circuitRef,name,
location,country,lat,lng,alt,url,insertDateTime,modifiedDateTime)
values
(@circuitID,@circuitRef,@name,@location,@country,@lat,@lng,@alt,@url,GETDATE(),GETDATE(
))
END;
if exists (select circuitSk
from [dbo].[DimCircuit]
where circuitAlternativeId = @circuitID)
BEGIN
update [dbo].[DimCircuit]
set circuitRef = @circuitRef,
name=@name,
location=@location,
country=@country,
lat=@lat,
lng=@lng,
alt=@alt,
url=@url,
modifiedDateTime = GETDATE()
where circuitAlternativeId = @circuitID
END;
END;

```

For DimConstructor

```

CREATE PROCEDURE [dbo].[UpdateDimConstructor]
@constructorID int,
@constructorRef nvarchar(255),
@name nvarchar(255),
@nationality nvarchar(255),
@url nvarchar(255)

AS
BEGIN
if not exists (select constructorSk
from [dbo].[DimConstructor]
where constructorAlternativeId = @constructorID)
BEGIN
insert into [dbo].[DimConstructor]
(constructorAlternativeId,constructorRef,name,
nationality,url,insertDateTime,modifiedDateTime)

```

```

values
(@constructorID,@constructorRef,@name,@nationality,@url,GETDATE(),GETDATE())
END;
if exists (select constructorSk
from [dbo].[DimConstructor]
where constructorAlternativeId = @constructorID)
BEGIN
update [dbo].[DimConstructor]
set constructorRef = @constructorRef,
name=@name,
nationality=@nationality,
url=@url,
modifiedDateTime = GETDATE()
where constructorAlternativeId = @constructorID
END;
END;

```

For DimSeasons

```

create PROCEDURE [dbo].[UpdateDimSeasons]
@year float,
@url nvarchar(255)

AS
BEGIN
if not exists (select seasonSk
from [dbo].[DimSeasons]
where year = @year)
BEGIN
insert into [dbo].[DimSeasons]
(year,url,insertDateTime,modifiedDateTime)
values
(@year,@url,GETDATE(),GETDATE())
END;
if exists (select seasonSk
from [dbo].[DimSeasons]
where year = @year)
BEGIN
update [dbo].[DimSeasons]
set
url=@url,
modifiedDateTime = GETDATE()
where year = @year
END;
END;

```

For DimStatus

```
CREATE PROCEDURE [dbo].[UpdateDimStatus]
@statusID int,
@status nvarchar(50)

AS
BEGIN
if not exists (select statusSk
from dbo.DimStatus
where statusAlternativeId = @statusID)
BEGIN
insert into dbo.DimStatus
(statusAlternativeId, status, insertDateTime, modifiedDateTime)
values
(@statusID, @status, GETDATE(), GETDATE())
END;
if exists (select statusSk
from dbo.DimStatus
where statusAlternativeId = @statusID)
BEGIN
update dbo.DimStatus
set status = @status,
modifiedDateTime = GETDATE()
where statusAlternativeId = @statusID
END;
END;
```

Here are the data source database table and the available fields

formula01_DataSource

Database Diagrams

Tables

System Tables

FileTables

External Tables

Graph Tables

dbo.circuits

dbo.constructors

dbo.drivers

dbo.races

dbo.results

dbo.status

Views

External Resources

Synonyms

Programmability

Service Broker

Storage

Security

dbo.results

Columns

resultId (float, null)

raceId (float, null)

driverId (float, null)

constructorId (float, null)

number (float, null)

grid (float, null)

position (float, null)

positionText (float, null)

positionOrder (float, null)

points (float, null)

laps (float, null)

time (float, null)

milliseconds (float, null)

fastestLap (float, null)

rank (float, null)

fastestLapTime (datetime, null)

fastestLapSpeed (float, null)

statusId (float, null)

dbo.status

Columns

- statusId (float, null)
- status (nvarchar(255), null)

dbo.drivers

Columns

- driverId (float, null)
- driverRef (nvarchar(255), null)
- number (nvarchar(255), null)
- code (nvarchar(255), null)
- forename (nvarchar(255), null)
- surname (nvarchar(255), null)
- dob (datetime, null)
- nationality (nvarchar(255), null)
- url (nvarchar(255), null)

dbo.circuits


Columns


- circuitId (float, null)
- circuitRef (nvarchar(255), null)
- name (nvarchar(255), null)
- location (nvarchar(255), null)
- country (nvarchar(255), null)
- lat (float, null)
- lng (float, null)
- alt (float, null)
- url (nvarchar(255), null)









dbo.constructors

Columns


- constructorId (float, null)
- constructorRef (nvarchar(255), null)
- name (nvarchar(255), null)
- nationality (nvarchar(255), null)
- url (nvarchar(255), null)

 **dbo.races**

 **Columns**

-  raceId (float, null)
-  year (float, null)
-  round (float, null)
-  circuitId (float, null)
-  name (nvarchar(255), null)
-  date (nvarchar(255), null)
-  time (datetime, null)
-  url (nvarchar(255), null)

The txt file of the data source is shown in below

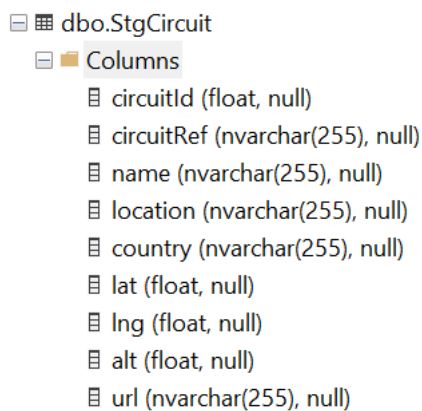
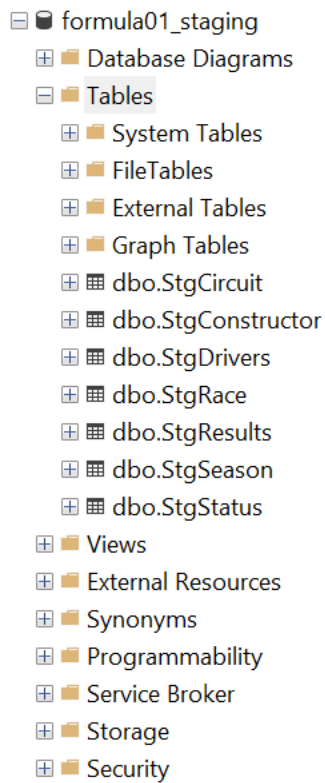
 seasons

5/7/2022 11:11 AM

Text Document

5 KB

The below diagram shows the staging database table and fields available



dbo.StgConstructor

Columns

- constructorId (float, null)
- constructorRef (nvarchar(255), null)
- name (nvarchar(255), null)
- nationality (nvarchar(255), null)
- url (nvarchar(255), null)

dbo.StgDrivers

Columns

- driverId (float, null)
- driverRef (nvarchar(255), null)
- number (nvarchar(255), null)
- code (nvarchar(255), null)
- forename (nvarchar(255), null)
- surname (nvarchar(255), null)
- dob (datetime, null)
- nationality (nvarchar(255), null)
- url (nvarchar(255), null)

dbo.StgRace

Columns

- raceId (float, null)
- year (float, null)
- round (float, null)
- circuitId (float, null)
- name (nvarchar(255), null)
- date (nvarchar(255), null)
- time (datetime, null)
- url (nvarchar(255), null)

dbo.StgSeason

Columns

- year (int, null)
- url (varchar(255), null)

dbo.StgStatus

Columns

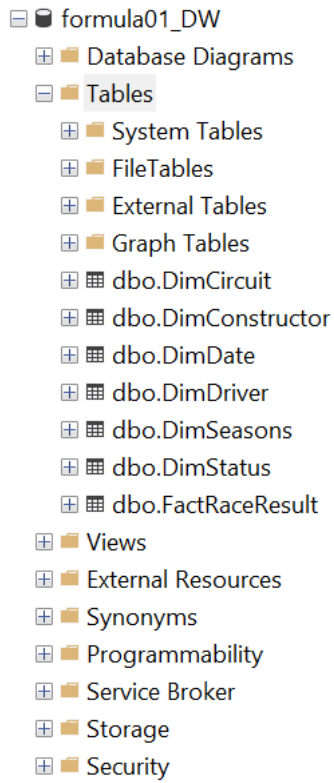
- statusId (float, null)
- status (nvarchar(255), null)

dbo.StgResults

Columns

- resultId (float, null)
- raceId (float, null)
- driverId (float, null)
- constructorId (float, null)
- number (float, null)
- grid (float, null)
- position (float, null)
- positionText (float, null)
- positionOrder (float, null)
- points (float, null)
- laps (float, null)
- time (float, null)
- milliseconds (float, null)
- fastestLap (float, null)
- rank (float, null)
- fastestLapTime (datetime, null)
- fastestLapSpeed (float, null)
- statusId (float, null)

The below diagram shows the data warehouse database table and fields available



```

//Result Fact Table
create table FactRaceResult(
    raceAlternativeId int not null,
    resultAlternativeId int not null,
    driverKey int,
    constructorKey int,
    number float,
    grid int,
    positionOrder int,
    points float,
    laps int,
    time float,
    milliseconds float,
    fastestLap int,
    rank int,
    fastestLapTime datetime,
    fastestLapSpeed float,
    statusKey int,
    circuitKey int,
    seasonKey int,
    round int,
    name nvarchar(255),
    dateKey int,
    rangeBetweenStartToEndPosition float,
    insertDateTime datetime,
    modifiedDateTime datetime,
)

```

```

//Circuit Table
create table DimCircuit(
    CircuitSK int identity(1,1) primary key,
    circuitAlternativeID int not null,
    circuitRef nvarchar(255) not null,
    name nvarchar(255),
    location nvarchar(255),
    country nvarchar(255),
    lat float,
    lng float,
    alt int,
    url nvarchar(255),
    insertDateTime datetime,
    modifiedDateTime datetime
)

//Constructor Table
create table DimConstructor(
    constructorSK int identity(1,1) primary key,
    constructorAlternativeId int not null,
    constructorRef nvarchar(255) not null,
    name nvarchar(255),
    nationality nvarchar(255),
    url nvarchar(255),
    insertDateTime datetime,
    modifiedDateTime datetime
)

//Status Table
create table DimStatus(
    statusSk int identity(1,1) primary key,
    statusAlternativeID float,
    status nvarchar(255),
    insertDateTime datetime,
    modifiedDateTime datetime
)

//Season Table
create table DimSeasons(
    seasonSk int identity(1,1) primary key,
    year float,
    url nvarchar(255),
    insertDateTime datetime,
    modifiedDateTime datetime
)

```


//Driver Table

```
create table DimDriver(
    driverSK int identity(1,1) primary key,
    driverAlternativeId int not null,
    driverRef nvarchar(255) not null,
    number float,
    code nvarchar(255),
    forename nvarchar(255),
    surname nvarchar(255),
    dob date,
    nationality nvarchar(255),
    url nvarchar(255),
    insertDateTime datetime,
    modifiedDateTime datetime,
    startDate datetime,
    endDate datetime
)
```

//Date Table

```
CREATE TABLE [dbo].[DimDate]
(
    [DateKey] VARCHAR(255) primary key,
    [Date] DATETIME,
    [FullDateUK] CHAR(10), -- Date in dd-MM-yyyy format
    [FullDateUSA] CHAR(10), -- Date in MM-dd-yyyy format
    [DayOfMonth] VARCHAR(2), -- Field will hold day number of Month
    [DaySuffix] VARCHAR(4), -- Apply suffix as 1st, 2nd ,3rd etc
    [DayName] VARCHAR(9), -- Contains name of the day, Sunday, Monday
    [DayOfWeekUSA] CHAR(1), -- First Day Sunday=1 and Saturday=7
    [DayOfWeekUK] CHAR(1), -- First Day Monday=1 and Sunday=7
    [DayOfWeekInMonth] VARCHAR(2), --1st Monday or 2nd Monday in Month
    [DayOfWeekInYear] VARCHAR(2),
    [DayOfQuarter] VARCHAR(3),
    [DayOfYear] VARCHAR(3),
    [WeekOfMonth] VARCHAR(1), -- Week Number of Month
    [WeekOfQuarter] VARCHAR(2), --Week Number of the Quarter
    [WeekOfYear] VARCHAR(2), --Week Number of the Year
    [Month] VARCHAR(2), --Number of the Month 1 to 12
    [MonthName] VARCHAR(9), --January, February etc
    [MonthOfQuarter] VARCHAR(2), -- Month Number belongs to Quarter
    [Quarter] CHAR(1),
    [QuarterName] VARCHAR(9), --First,Second..
    [Year] CHAR(4), -- Year value of Date stored in Row
    [YearName] CHAR(7), --CY 2012,CY 2013
    [MonthYear] CHAR(10), --Jan-2013, Feb-2013
    [MMYYYY] CHAR(6),
    [FirstDayOfMonth] DATE,
    [LastDayOfMonth] DATE,
    [FirstDayOfQuarter] DATE,
    [LastDayOfQuarter] DATE,
    [FirstDayOfYear] DATE,
    [LastDayOfYear] DATE,
    [IsHolidaySL] BIT, -- Flag 1=National Holiday, 0=No National Holiday
    [IsWeekday] BIT, -- 0=Week End ,1=Week Day
    [HolidaySL] VARCHAR(50), --Name of Holiday in US
    [isCurrentDay] int, -- Current day=1 else = 0
    [isDataAvailable] int, -- data available for the day = 1, no data available for the day = 0
    [isLatestDataAvailable] int
)
```