Sri Lanka Institute of Information Technology

Assignment I

Data Warehouse & Business Intelligence

2021

Submitted By:

Maduwantha W.W.A.K.

IT19027548

Contents

Data set selection and preparation

The selected data source was collection of transactional data tables obtained from the Kaggle web service. The link of the dataset is given below

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

this dataset consist of all the information of races , drivers, constructors, circuits, qualifying of the formula 01 race contest. There were some cleansing and modifications to be done when loading data to staging layer

the sources are consist of two main source types the list of the sources are given below

TEXT files

- Constructors
- Countries

Few created database table from csv files provided from the Kaggle service

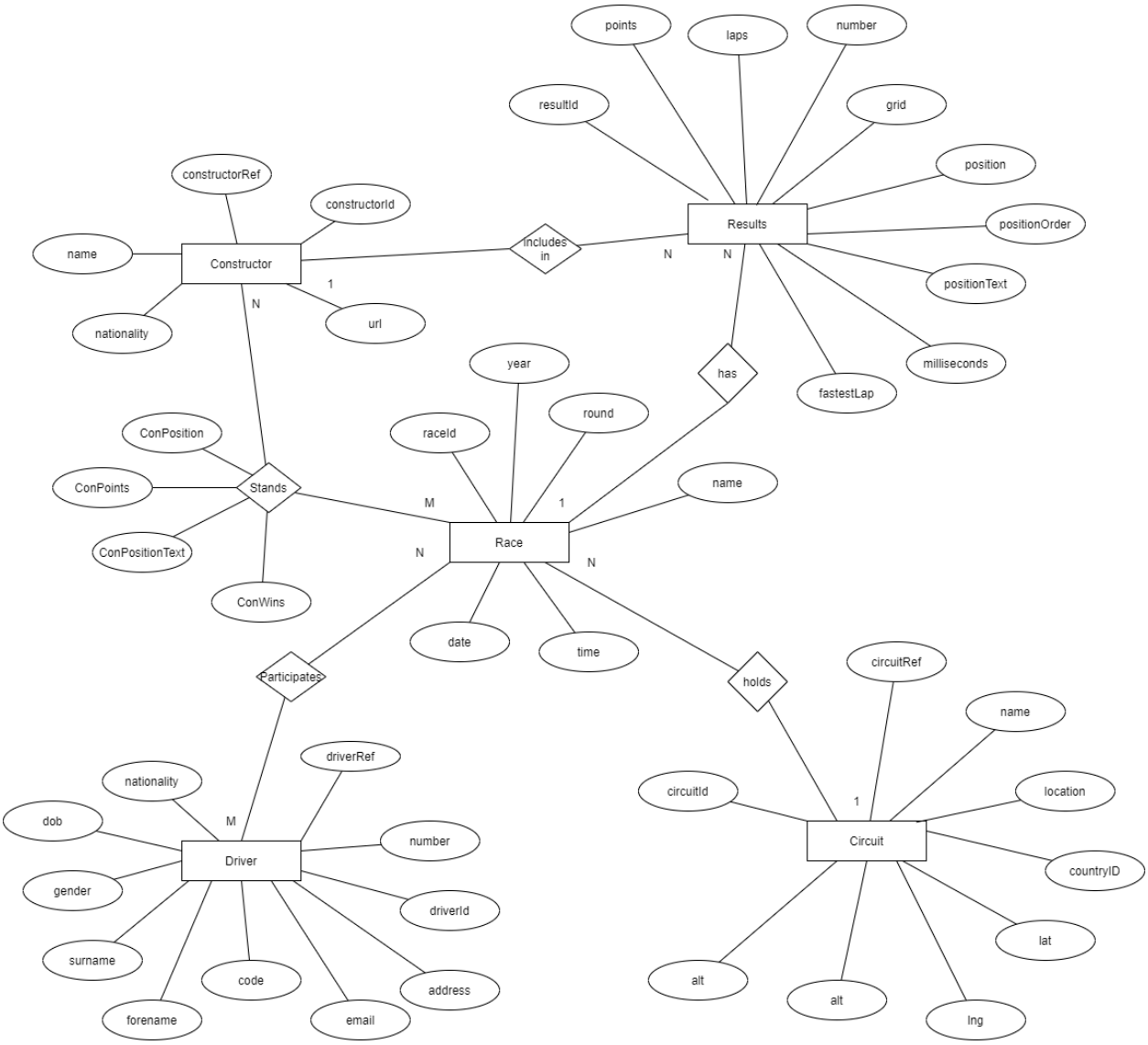
- Circuits
- Constructor_Standings
- Driver
- Race
- Results

Discription of the data set

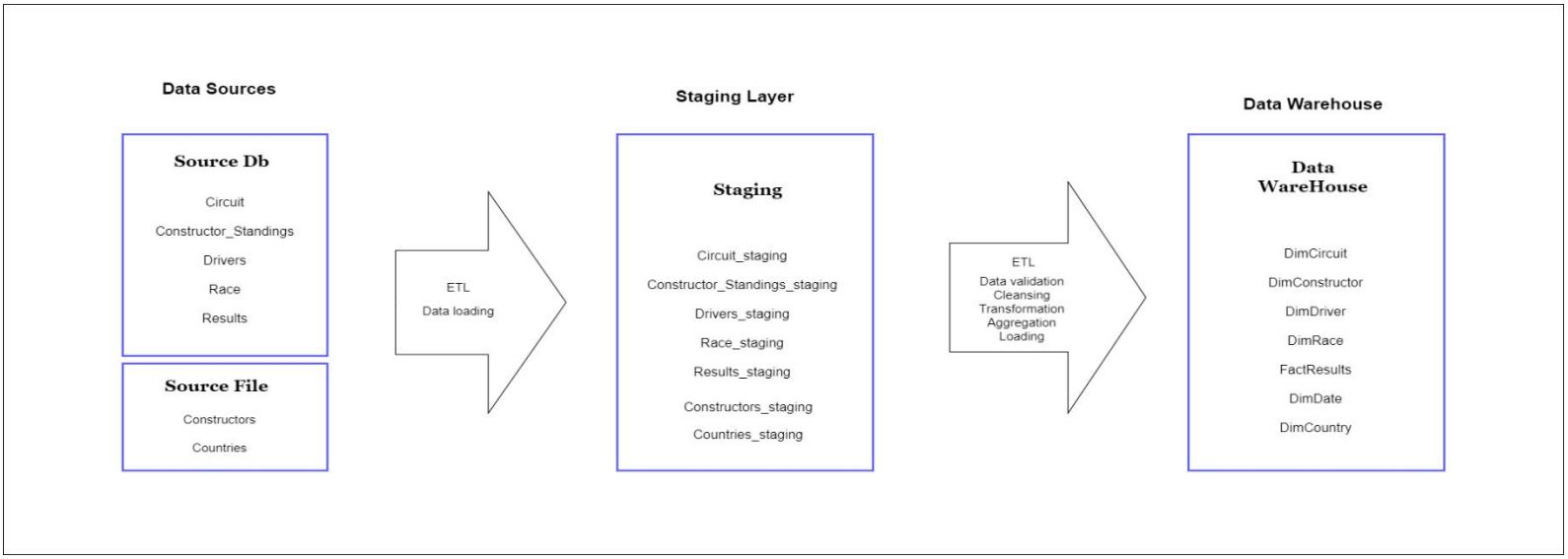
Table name	Column name	Datatypes	description
Circuits	circuitId	int	details of all the circuits where the competition will be held
	circuitRef	nvarchar	
	name	nvarchar	
	location	nvarchar	
	countryID	nvarchar	
	lat	float	
	lng	float	
	alt	float	
	url	nvarchar	
Country	countryID	int	includes the country of the Circuit
	country	nvarchar	
Constructor_standings	constructorStandingsId	int	Includes all the information of Constructor Standings according to the race
	raceId	int	
	constructorId	int	
	ConPoints	int	
	ConPosition	int	
	ConPositionText	nvarchar	
	ConWins	int	
Constructor	constructorId	int	includes all the informations of constructors
	constructorRef	nvarchar	
	name	nvarchar	
	nationality	nvarchar	
	url	nvarchar	
Driver	driverId	int	Includes all the information related to
	driverRef	nvarchar	
	number	int	
	code	nvarchar	

	forename	nvarchar	drivers who are going to participate in the event
	surname	nvarchar	
	gender	nvarchar	
	dob	date	
	nationality	nvarchar	
	address	nvarchar	
	email	nvarchar	
	url	nvarchar	
Race	raceId	int	includes all the details of race events
	year	nvarchar	
	round	int	
	circuitId	int	
	name	nvarchar	
	date	date	
	time	time	
	url	nvarchar	
Results	resultId	int	includes all the result matrix of the races of the competition
	raceId	int	
	driverId	int	
	constructorId	int	
	number	int	
	grid	int	
	position	int	
	positionText	nvarchar	
	positionOrder	int	
	points	int	
	laps	int	
	milliseconds	int	
	fastestLap	int	
	rank	int	
	fastestLapTime	time	
	fastestLapSpeed	decimal	
	statusId	int	

ER diagram


points
laps
number
resultId
grid
position
constructorRef
constructorId
Results
positionOrder
name
Constructor
includes in
N
N
positionText
1
N
nationality
url
has
milliseconds
year
fastestLap
round
ConPosition
raceId
name
ConPoints
Stands
M
1
Race
N
N
ConPositionText
ConWins
date
time
Participates
holds
circuitRef
name
driverRef
nationality
circuitId
location
M
1
dob
Driver
number
Circuit
countryID
gender
driverId
lat
surname
code
address
alt
alt
lng
forename
email

Solution Architecture



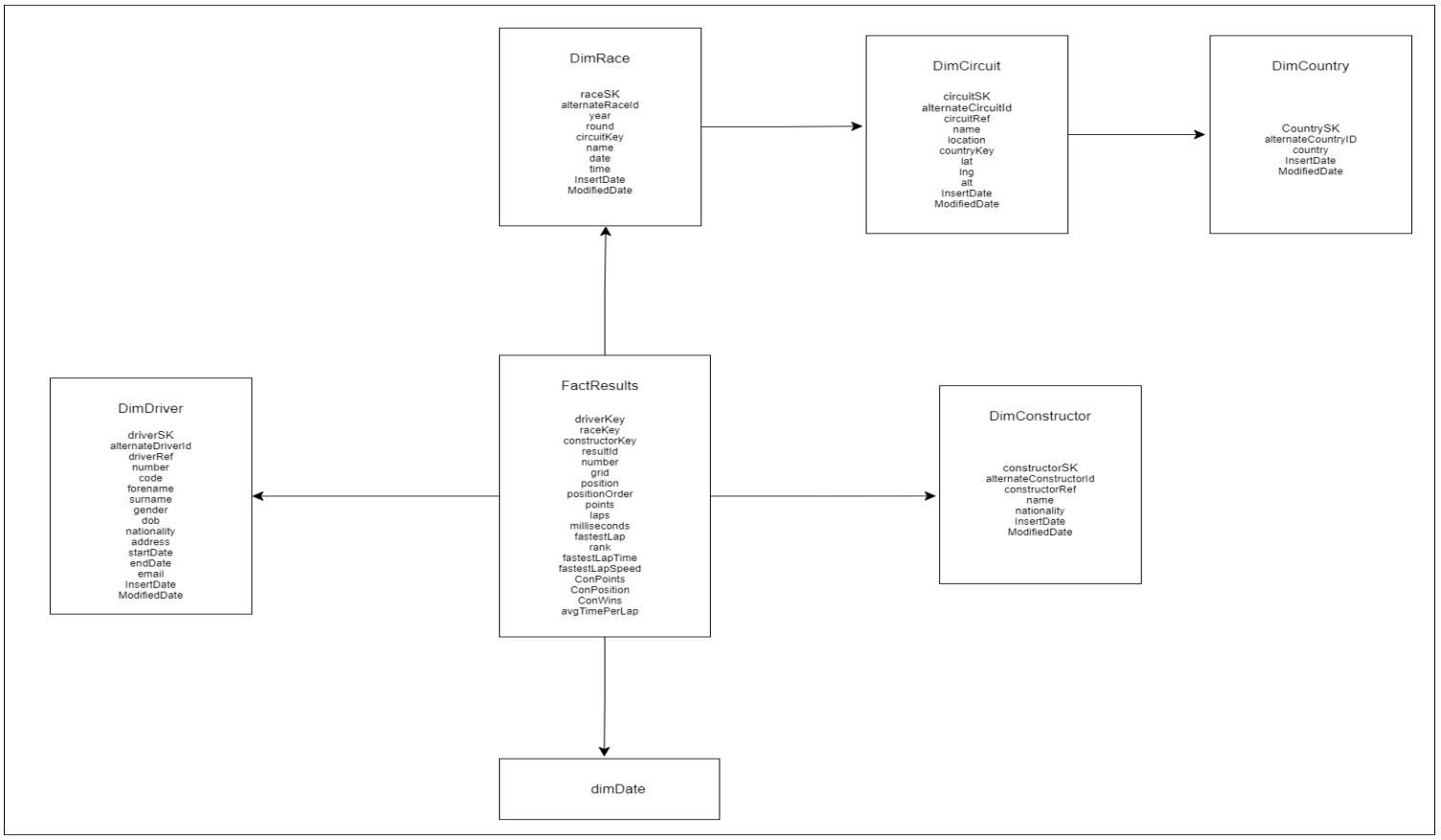
According to the explanation initial step is to create the staging layer to the above data set , after executing the etl of the staging layer fallowing staging tables created

1. Circuit_staging
2. Constructor_standings_staging
3. Driver_staging
4. Race_staging
5. Results_staging
6. Constructor_staging
7. Countries_staging

As the next step staged tables are profiled and checked whether the columns have null values , more duplicates whether aggregations to be done. As the next step data is transformed according to the profiling information and loaded. After completing the described stages, data is tested and validated and load to the Datawarehouse.

After the warehouse is created BI results such as OLAP analysis if necessary, Reports, Data visualization, Data mining can be obtained as results after further modifications.

DataWarehouse Design



Snowflake schema model is used to design this Datawarehouse design. There is one fact table as Results and 6 dimensions. The grain level of this fact table is results record per Driver and race

From all the six diemensions , race -> Circuit -> Country is a hierarchy

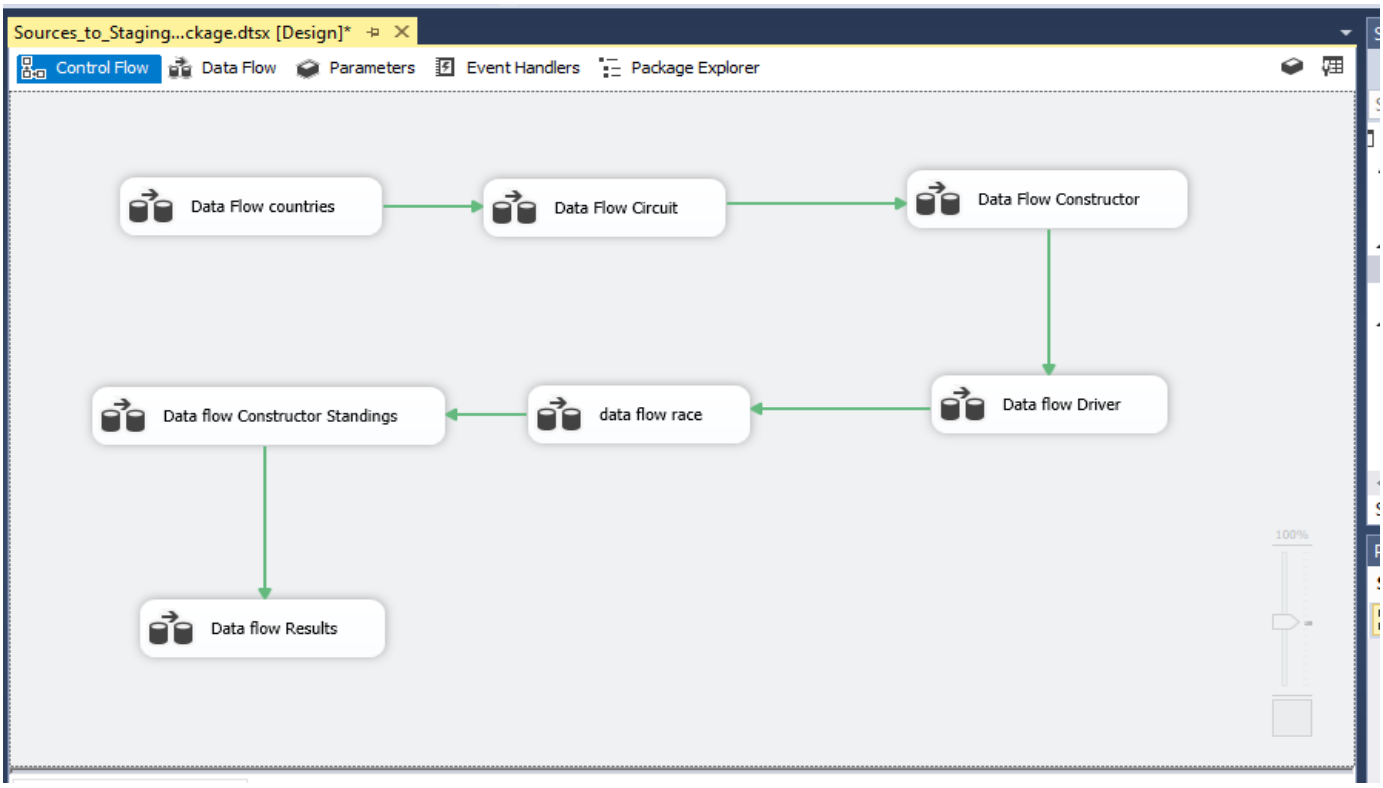
DimDriver assume to be slowly changing dimesion because permanent address or email can be changed.

ETL development

As initial step source data are extracted and loaded to the staging layer before creating the datawarehouse. This staging layer is created to minimize the impact to the datawarehouse. In ETL package to handle above process called Sources_to_Staging_ETL_Package ETL was created in the service integration project and all the extractions from source table to staging table are managed by the data flaw tasks which are created inside the ETL component.

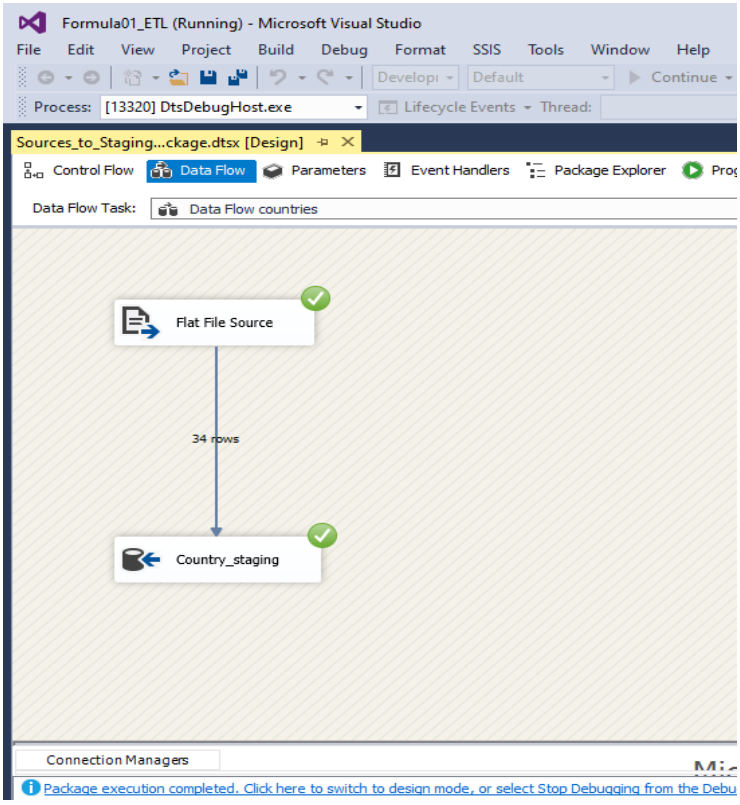
Inside the ETL flow task the source data table , destination staging data table the their mappings are created and truncate sql command event handler are added to the each and every flaw task to truncate the table before execute the ETL flawtask

All the joined ETL flaw tasks of Sources_to_Staging_ETL_Package is given below



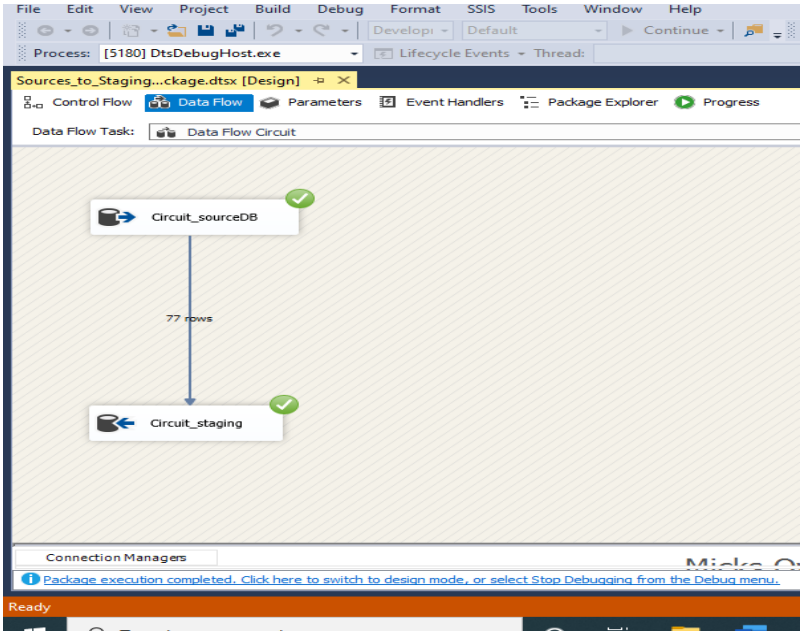
Screenshots of all the data sources that were staged and truncate tables created are attached below:

Staging Country Source data



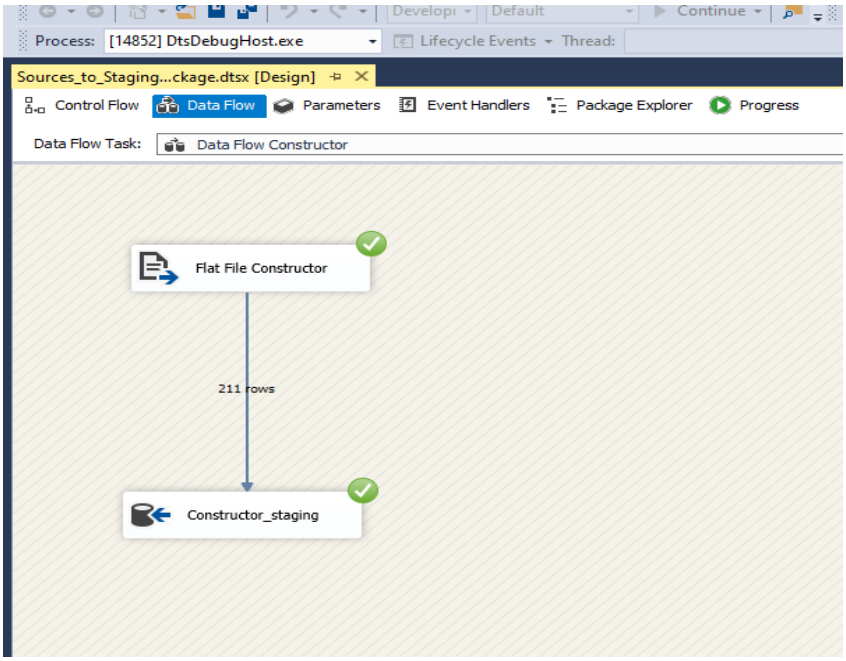
This flaw task depicts the extraction of Country source data from flat file Source(TXT) and loaded to staging layer as Country_staging table

Staging Circuit Source data



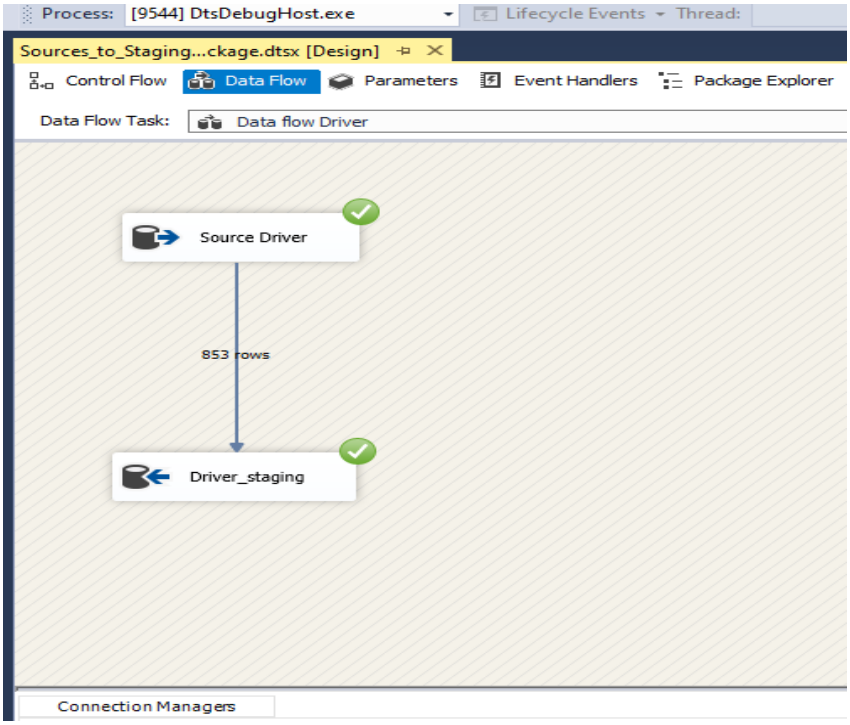
This flaw task depicts the extraction of Circuit source data from database table and loaded to staging layer as Circuit_staging table

Staging Constructor Source data



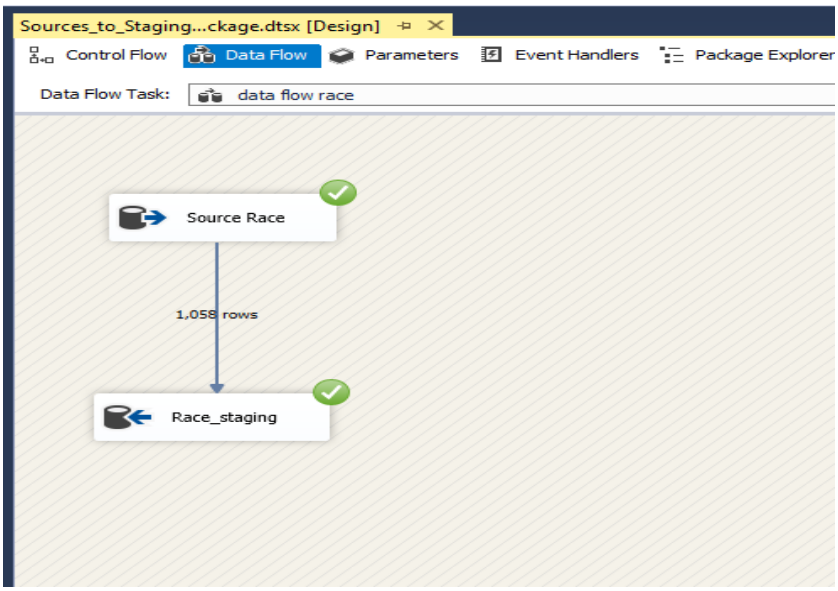
This flaw task depicts the extraction of Constructor source data from flat file Source and loaded to staging layer as Constructor_staging table

Staging Driver Source data



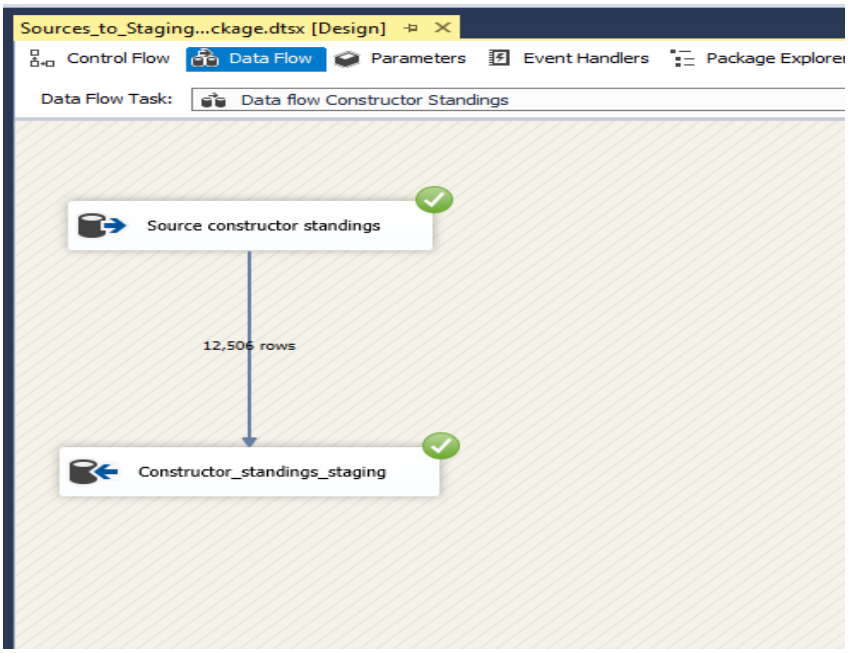
This flaw task depicts the extraction of Driver source data from database table and loaded to staging layer as Driver_staging table

Staging race Source data



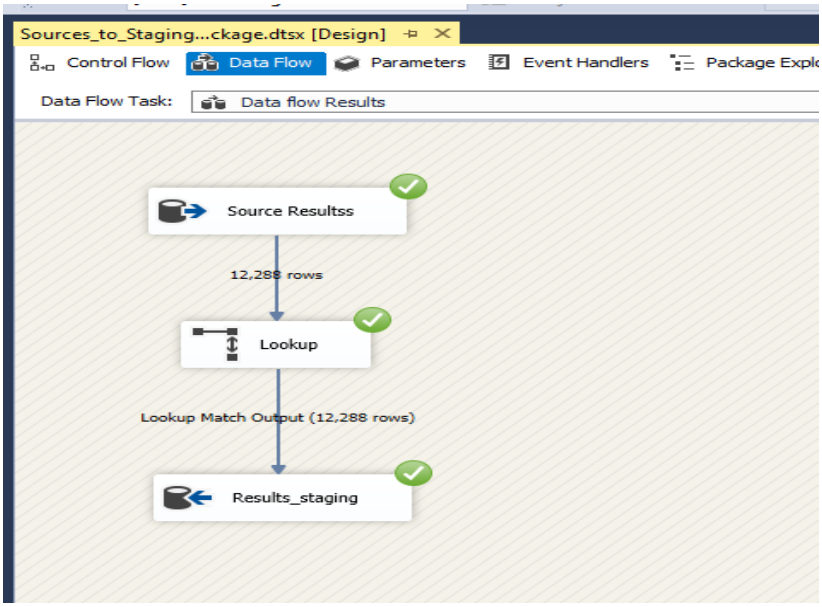
This flaw task depicts the extraction of Race source data from database table and loaded to staging layer as Race_staging table

Staging Constructor standings Source data



This flaw task depicts the extraction of Constructor standings source data from database table and loaded to staging layer as Constructor_standings_staging table

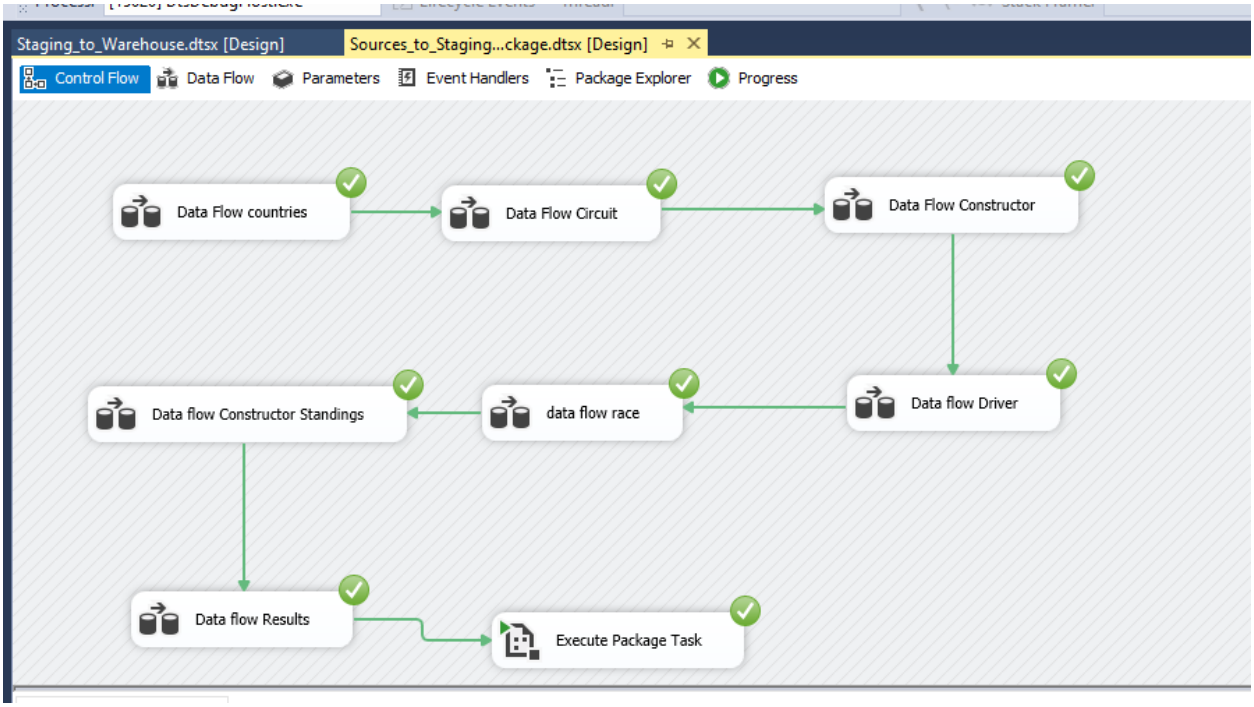
Staging results Source data



This flaw task depicts the extraction of Results source data from database table and loaded to staging layer as Results _staging table

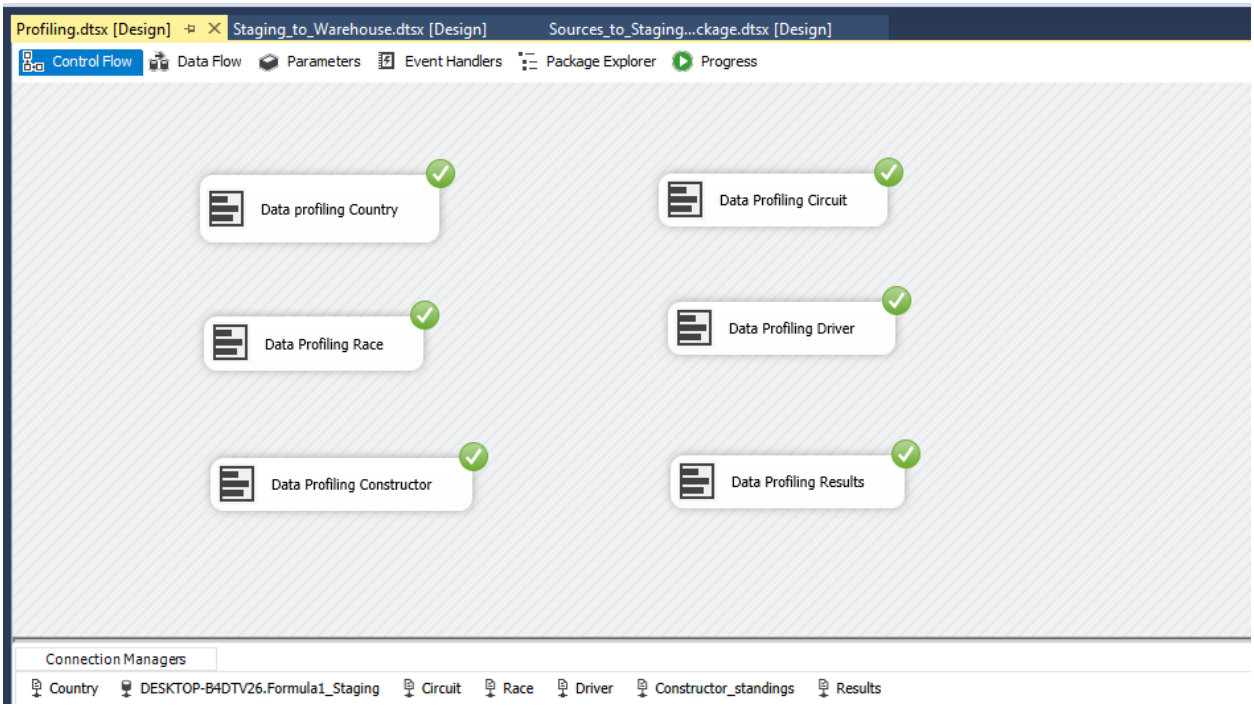
- A lookup column has been added to the results staging to get the date of the race

The execution task connected to the last data flow task is linked to the Staging to datawarehouse package



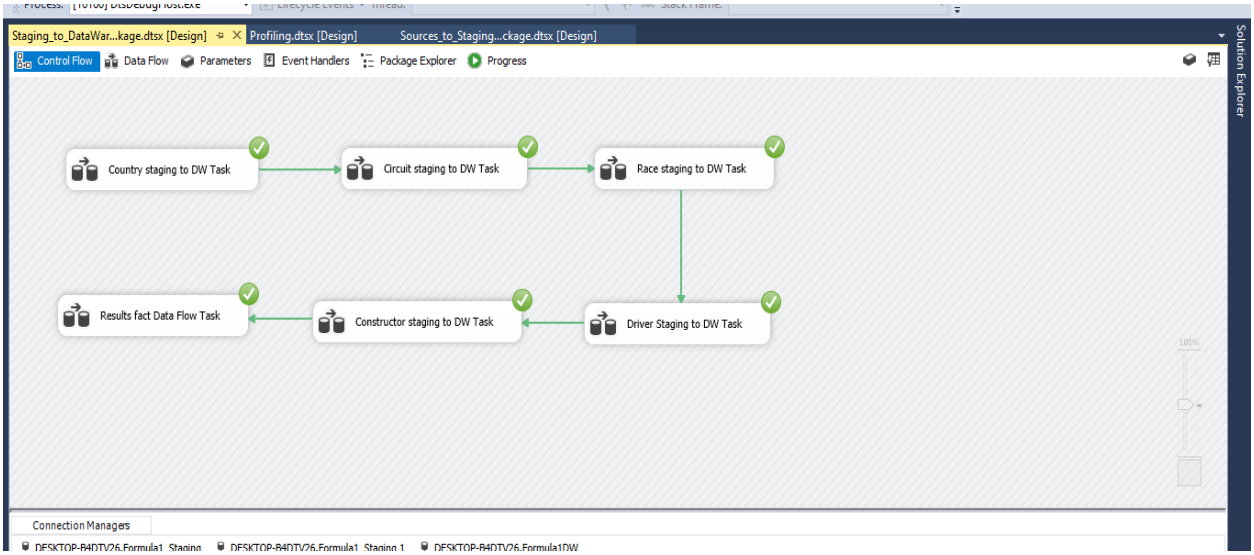
.

Profiling is done as follow



Next step is data transformation (executing staging to warehouse ETL) and as described in a previous step, the execution task connected to the last data flow task of the first package (staging the results) is attached to the transformation package (staging to datawarehouse package).

All the joined ETL flaw tasks of Staging_to_DataWarehouse_Package is given below (data transformation and loading to Formula1DW)



According to the dependency of the database tables the execution order is given below

- Transformation and loading of Country Dimension (First step)
- Transformation and loading of Circuit Dimension (Second step)
- Transformation and loading of Race Dimension (Third step)

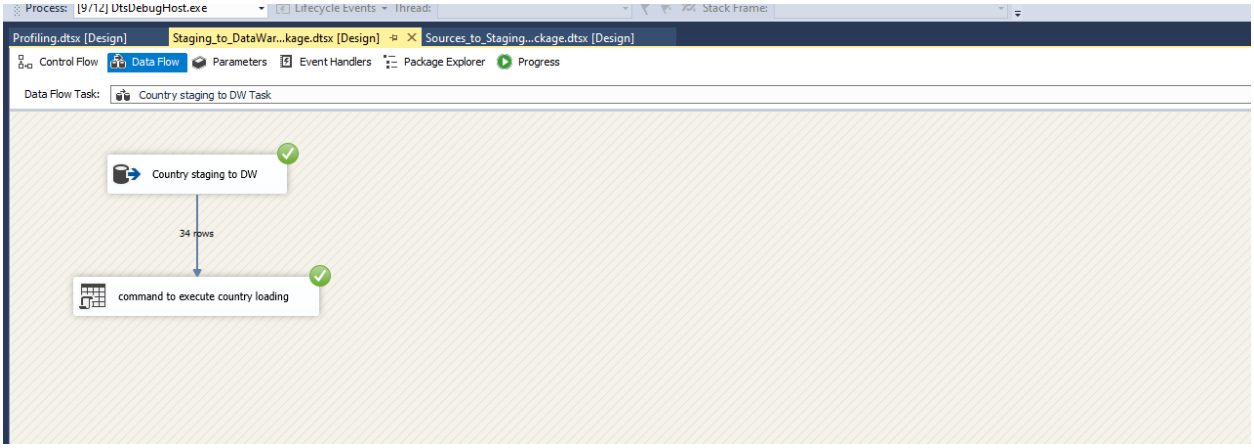
- Transformation and loading of Driver Dimension (Fourth step)

- Transformation and loading of Constructor Dimension (Fifth step)
- Transformation and loading of Results Fact table (Sixth or Last step)

Transformation and loading of the Country Dimension (first step)

Country table doesn't have any references to another table and this doesn't have any transformations

ETL task of the transformation of the country_staging table given below



The data is loaded to the DimCountry using Stored procedure which was stored in the database of the Warehouse

Transformation and loading of the Circuit Dimension (second step)

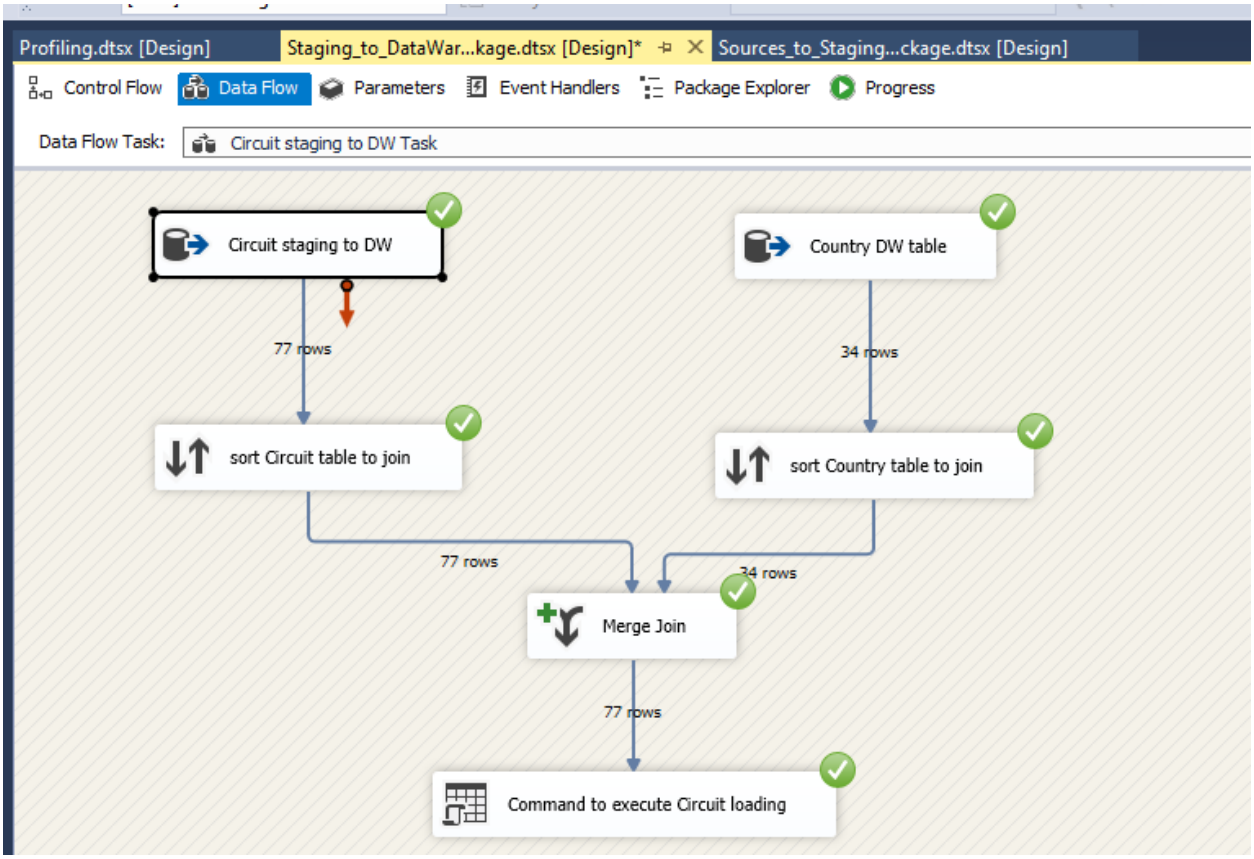
In DimCircuit table there is a reference to the DimCountry(hierarchy country -> circuits) in order to loading of the Circuit staging table to DimCircuit dimension it is necessary to take the surrogate key of the Country table and map it to DimCircuit when loading.

This ETL mapping of the countryKey (CountrySK) done by joining the DimCountry table and Circuit_staging table using countryID and alternateCountryID of DimCountry

Before joining the two table both the tables are sorted according to their referencing attribute and join the two tables using merge join component

After joining the two tables necessary fields are taken from the output of the joined tables and pass the data to the stored procedure to load the data to DimCircuit

ETL task of the transformation of the Circuit_staging table given below



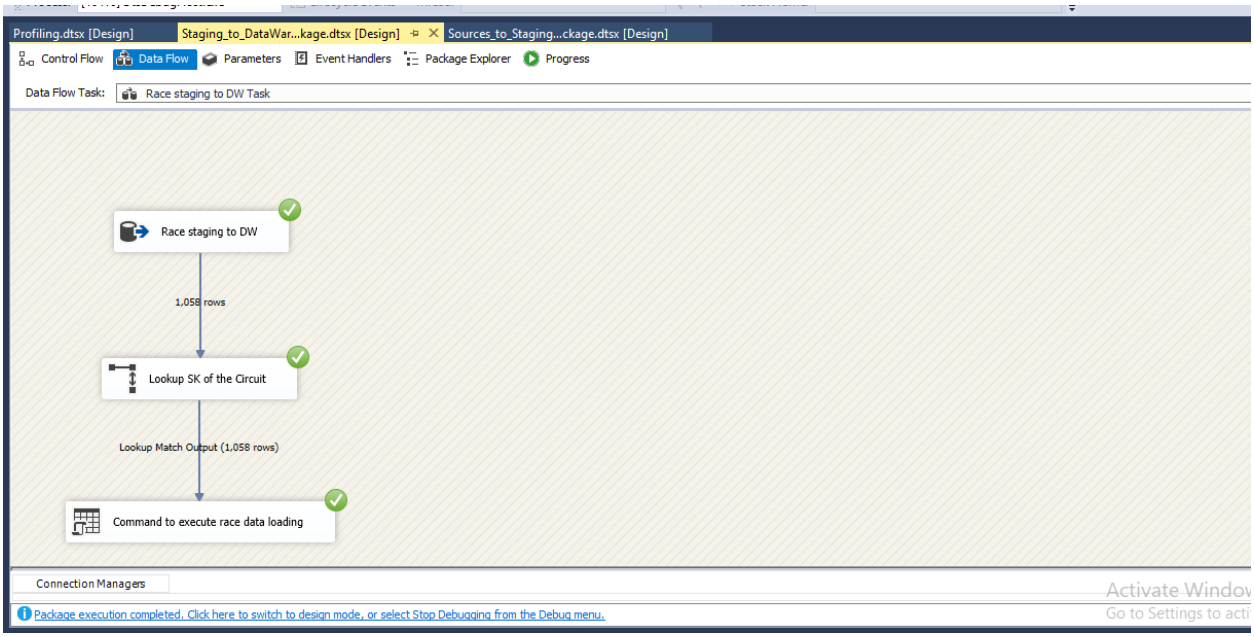
Transformation and loading of the Race Dimension (third step)

As the previous step In DimRace table there is a reference to the DimCircuit(hierarchy Circuit -> Race) in order to loading of the Race_staging table to DimRace dimension it is necessary to take the surrogate key of the Circuit table and map it to DimRace when loading the data.

This ETL task of mapping the surrogate key of the DimCircuit dimension table to the DimRace table is done using lookup component provided by SSIS

The necessary data coming from the Race_staging and lookup column from the DimCircuit(CircuitSK) is sent to the stored procedure using execute OLEDB command.

ETL task of the transformation of the Race_staging table is given below



Transformation and loading of the Driver Dimension (fourth step)

As mentioned earlier under assumptions, Driver dimension details were considered as slowly changing details.

The changing attributes of the slowly changing dimension Driver are given below

- Address (permanent address of the driver) historical attribute
- Email address (personal email address of the driver) historical attribute
- Code (code of the Driver which will be used in races) changing attribute
- Number (the number given to the driver in races) changing attribute

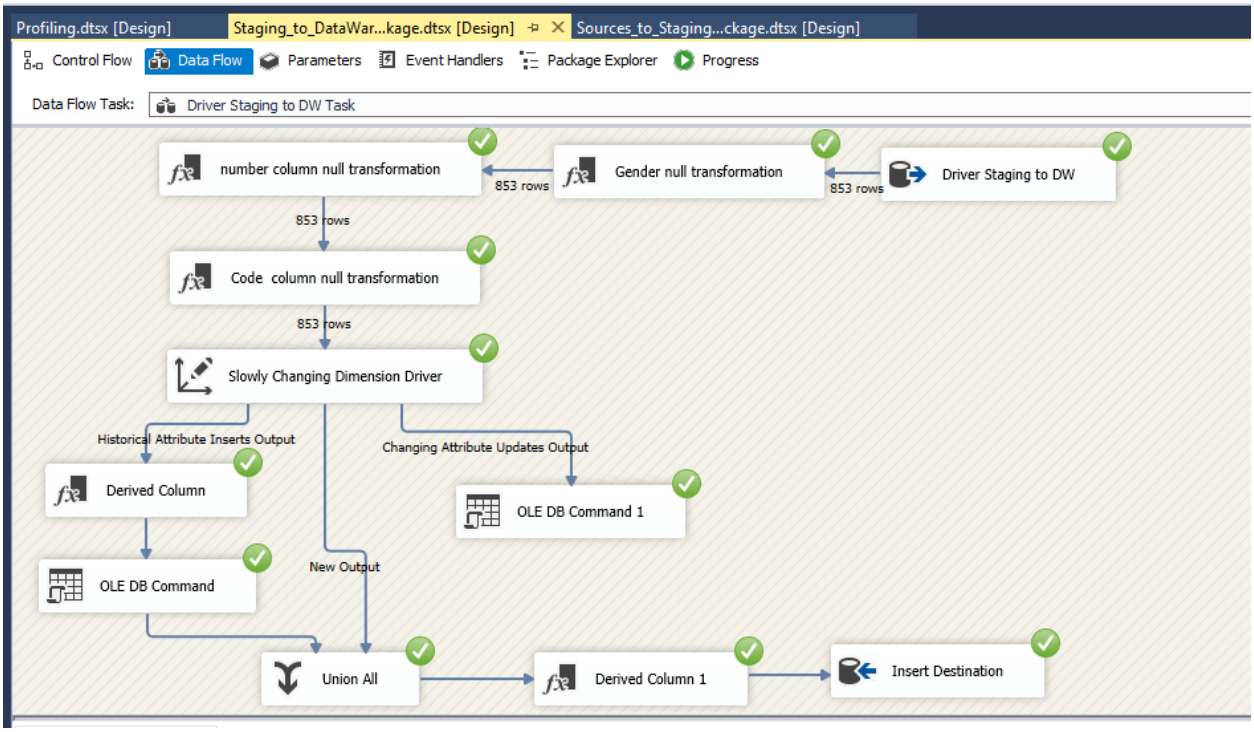
The data coming from the Driver_staging is subjected to some transformations

1. Replace null values of gender column to “N”
2. Replace null values of number column to 0

3. Replace null values of code column first three letters of each Drivers surname

Then set up the changing and historical attributes of the DimDriver dimension and Set up the DimDriver destination table by using OLEDB destination tool in SSIS.

ETL task of the transformation of the Driver_staging table is given below

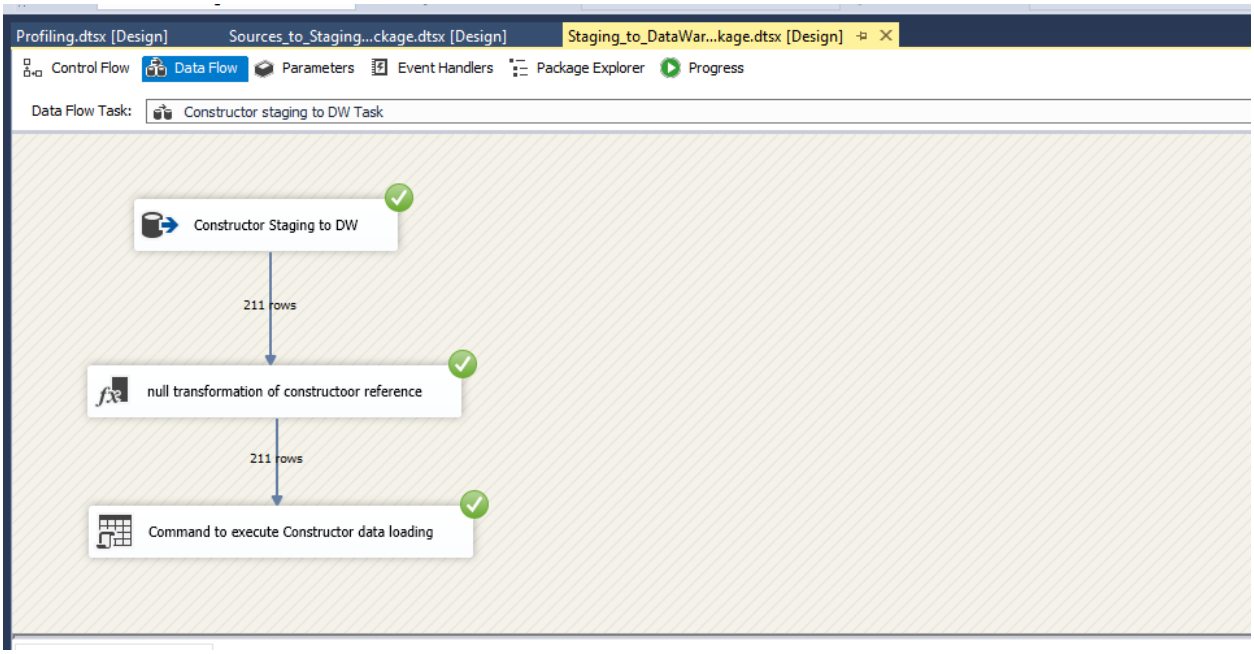


Transformation and loading of the Constructor Standings Dimension (fifth step)

DimConstructor table contains all the informations of constructors who made the parts of the cars in the formula one contest and there is only have one reference to the fact table (according to the snowflake schema) because of that loading the Constructor_staging table can be done as single table to DimConstrutor.

In order to load the data from the Constructor_staging to DimConstructor there is a null transformation on column(constructor ref) which will replace all the nulls with first three letters of the constructor name.

ETL task of the transformation of the Constructor_staging table is given below



Transformation and loading of Results Fact table (Sixth or Last step)

Transforming and loading the fact Results is the final step of the data warehousing ETL. FactResult table consiste results from both Results_staging and Constructor_standings_staging tables.

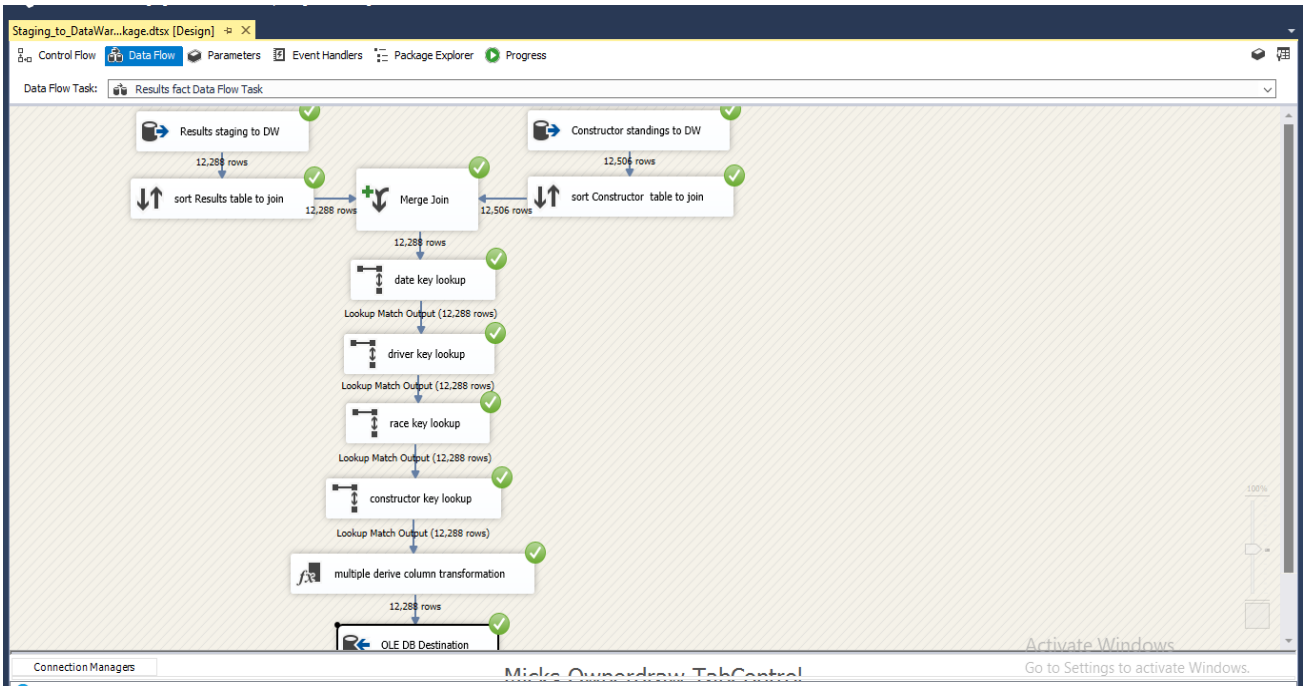
Grain of this datawarehouse is one record per driver and race.

In order to load data to FactResults transformations and lookups given below is applied

- Sort both the tables according to joined attributes
- Join the two tables (Constructor_standing_staging and Result_staging) merg join(left outer)
- Lookup dateKey(Surrogate key of date Dimension)
- Lookup driverKey(Surrogate key of DimDriver Dimension)
- Lookup raceKey(surrogate key of DimRace dimension)

- Lookup constructorKey (surrogate key of DimConstructor dimension)
- Calculate average time taken to finish a lap by a driver in derive column component (multiple derive column transformation component)

ETL task of the transformation and loading of the FactResult table is given below



All the stored procedures use to load staging to warehouse given below

Load data to DimCountry

```
CREATE PROCEDURE [dbo].[UpdateDimCountry]
@alternateCountryID int,
@country nvarchar(50)
as
begin
if not exists (select CountrySK from dbo.DimCountry where
alternateCountryID = @alternateCountryID )
begin
insert into dbo.DimCountry
(alternateCountryID,country,InsertDate,ModifiedDate)
values (@alternateCountryID,@country,GETDATE(),GETDATE())
end;
if exists (select CountrySK from dbo.DimCountry where alternateCountryID =
@alternateCountryID)
begin
update dbo.DimCountry
set alternateCountryID = @alternateCountryID, country = @country,
ModifiedDate = GETDATE()
where alternateCountryID = @alternateCountryID
end;
end;
```

Load data to DimCircuit

```
CREATE PROCEDURE [dbo].[UpdateDimCircuit]
@alternateCircuitId int,
@circuitRef nvarchar(50),
@name nvarchar(50),
@location nvarchar(50),
@countryKey int,
@lat decimal(5,0),
@lng decimal(5,0),
@alt int
as
begin
if not exists (select circuitSK from dbo.DimCircuit where
alternateCircuitId = @alternateCircuitId)
begin
insert into dbo.DimCircuit
(alternateCircuitId,circuitRef,name,location,countryKey,lat,lng,alt,Insert
Date,ModifiedDate)
values
(@alternateCircuitId,@circuitRef,@name,@location,@countryKey,@lat,@lng,@al
t,GETDATE(),GETDATE())
end;
if exists (select circuitSK from dbo.DimCircuit where alternateCircuitId =
@alternateCircuitId)
begin
update dbo.DimCircuit
set alternateCircuitId = @alternateCircuitId, circuitRef = @circuitRef,
name = @name,location = @location,countryKey = @countryKey,lat = @lat,lng
= @lng,alt = @alt,ModifiedDate = GETDATE()
where alternateCircuitId = @alternateCircuitId
end;
end;
```

Load data to DimRace

```
CREATE PROCEDURE [dbo].[UpdateDimRace]
@alternateRaceId int,
@year nvarchar(50),
@round int,
@circuitKey int,
@name nvarchar(50),
@date date,
@time time
as
begin
if not exists (select raceSK from dbo.DimRace where alternateRaceId =
@alternateRaceId)
begin
insert into dbo.DimRace
(alternateRaceId,year_,round_,circuitKey,name_,date_,time_,InsertDate,Modi
fiedDate)
values
(@alternateRaceId,@year,@round,@circuitKey,@name,@date,@time,GETDATE(),GET
DATE())
end;
if exists (select raceSK from dbo.DimRace where alternateRaceId =
@alternateRaceId)
begin
update dbo.DimRace
set alternateRaceId = @alternateRaceId,year_ = @year,round_ =
@round,circuitKey = @circuitKey,name_ = @name,date_ = @date,time_ =
@time,ModifiedDate = GETDATE()
where alternateRaceId = @alternateRaceId
end;
end;
```

Load data to DimConstructor

```
ALTER PROCEDURE [dbo].[UpdateDimConstructor]
@alternateConstructorId int,
@constructorRef nvarchar(50),
@name nvarchar(50),
@nationality nvarchar(50)
as
begin
if not exists (select constructorSK from dbo.DimConstructor where
alternateConstructorId = @alternateConstructorId)
begin
insert into dbo.DimConstructor
(alternateConstructorId,constructorRef,name,nationality,InsertDate,Modifie
dDate)
values
(@alternateConstructorId,@constructorRef,@name,@nationality,GETDATE(),GETD
ATE())
end;
if exists (select constructorSK from dbo.DimConstructor where
alternateConstructorId = @alternateConstructorId)
begin
update dbo.DimConstructor
set alternateConstructorId = @alternateConstructorId,constructorRef =
@constructorRef,name = @name,nationality = @nationality,ModifiedDate =
GETDATE()
where alternateConstructorId = @alternateConstructorId
end;
end;
```