Advanced Databases 1 – Assignment 1

James Byrne – C11393906

# Explanation of how the script works

The database used is a MySQL database. The script is broken into steps, as seen in the code. Each step takes care of a single part of the ETL process. These steps are:

* Create a database (Advanced-Databases)
* Switch to the database
* Delete any existing tables
* Create the initial tables (Relational Model as per the assignment specification)
* Create the HolderTable
* Undergo the first insert (insert statements from assignment specification)
* Load the Premier.csv file into the HolderTable
* Insert the information from the HolderTable into the relevant tables
  + Player
  + Matches
  + Player\_stats
* Create the secondary tables (Dimensional Model as per the assignment specification)
* Create the staging tables
* Load the data from the relational model into the staging tables
* Load the data from the staging tables into the Dimensional Model
* Update the fact\_staging table with the information from the other staging tables
* Insert the information from the facts\_stage table into the Facts\_Stats table
* Undergo Second ETL
* Load the data from the ETL2.csv file into the HolderTable
* Insert the data from the HolderTable into the relevant tables checking to ensure that duplicates are not inserted
* Insert information into the staging tables, ensuring that duplicates are not entered
* Insert information from the staging tables into the Dimensional model checking for duplicates
* Both ETL processes are completed at this point
* Two queries are run at the end of the script

More information on parts of the script are given below.

# Loading from Files

The data is loaded from the files given the files position within the file system. For this example both of the files within the home directory “~/someFile.csv”. You will need to change this to point at your files location e.g. “C:/users/someUser/Documents”. You may also need to change the value of “lines terminated by ‘\r\n’”.

# HolderTable

The HolderTable is used to store all of the data from both of the csv files until the data can be moved to its appropriate table with the correct relationships. From there the data can then be loaded into the Dimensional Model through the staging tables.

# Running the Script

The script can be run using a GUI tool such as Toad, Sequel Pro or SQL Developer by opening the file and running the script using a command. To run the script through the terminal you must first log into the MySQL database. You can then execute the script by using the following command “source ~/path/to/file”.

# Queries

The first Query return the top 5 performances by a player measured by the number of goals scored and display where they where and who the opponent was. The example in the script searches for a player with name “Clint”.

The second Query returns the player with the highest pass rate on a team for a given year. The example below shows the results for the Manchester United team.

# The Script

The entire script is included within this document starting on the next page. The code is also attached as the totalScript.sql file. All of the code can also be found at <https://github.com/James-Byrne/Adv-Data-Assignment-1/>

-- Created as a MySQL database and ran using the Sequel Pro application on OSX

-- MySQL database with the name Advanced-Databases

create database Advanced-Databases;

-- Switch to the Advanced-Databases database

use Advanced-Databases;

-- Drop the tables if they already exist

drop table Stadiums;

drop table Player\_stats;

drop table Matches;

drop table Player;

drop table Teams;

drop table HolderTable;

drop table Fact\_Stats;

drop table DimPlayer;

drop table DimTeam;

drop table DimTime;

drop table DimStadium;

drop table player\_stage;

drop table fact\_stage;

drop table stadium\_stage;

drop table team\_stage;

drop table time\_stage;

-- Create the tables for the relational model

create table Teams (

TeamID int not null,

Team\_name varchar(225),

YearOfFound int,

primary key (TeamID)

);

create table Player (

Player\_ID int,

TeamID int,

PL\_name varchar(225),

PL\_surname varchar(225),

foreign key (TeamID) references Teams(TeamID),

primary key (Player\_ID)

);

create table Stadiums (

StadiumID int,

St\_name varchar(225),

City varchar(225),

Capacity int,

TeamID int,

foreign key (TeamID) references Teams(TeamID),

primary key (StadiumID)

);

create table Matches (

Team\_A varchar(225),

Team\_B varchar(255),

M\_date DATE,

primary key (Team\_A, Team\_B, M\_date)

);

create table Player\_stats (

Team\_A\_ID varchar(225),

Team\_B\_ID varchar(225),

Player\_ID int,

M\_date DATE,

MinPlayed int,

Goals int,

Shot\_on int,

Shot\_off int,

Penalty int,

Pass\_OK int,

Pass\_KO int,

foreign key (Team\_A\_ID, Team\_B\_ID, M\_date) references Matches(Team\_A, Team\_B, M\_date),

foreign key (Player\_ID) references Player(Player\_ID)

);

-- Create the holder table for importing from the CSV files

create table HolderTable (

m\_date DATE,

player\_id int,

player\_surname varchar(255),

player\_name varchar(255),

team\_a\_name varchar(255),

team\_a\_id int,

team\_b\_name varchar(255),

team\_b\_id int,

minutes\_played int,

goal int,

shots\_on\_target int,

shots\_off\_target int,

penalties int,

pass\_success int,

pass\_fail int

);

-- Insert teams

-- team\_id,team\_name,year\_of\_foundation

insert into teams values (11,'Everton',1912);

insert into teams values (1,'Manchester United',1902);

insert into teams values (110,'Stoke City',1908);

insert into teams values (56,'Sunderland',1895);

insert into teams values (54,'Fulham',1887);

insert into teams values (8,'Chelsea',1872);

insert into teams values (111,'Wigan Athletic',1904);

insert into teams values (3,'Arsenal',1888);

insert into teams values (6,'Tottenham Hotspur',1901);

insert into teams values (4,'Newcastle United',1899);

insert into teams values (52,'Queens Park Rangers',1901);

insert into teams values (30,'Bolton Wanderers',1887);

insert into teams values (80,'Swansea City',1900);

insert into teams values (14,'Liverpool',1907);

insert into teams values (35,'West Bromwich Albion',1897);

insert into teams values (39,'Wolverhampton Wanderers',1867);

insert into teams values (45,'Norwich City',1914);

insert into teams values (5,'Blackburn Rovers', 1870 );

insert into teams values (7,'Aston Villa', 1899 );

insert into teams values (43,'Manchester City', 1891 );

-- Insert Stadiums

-- stadiumID,stadium name, city, capacity, team\_id (FK)

insert into stadiums values(1,'Old Trafford','Manchester',76100,1);

insert into stadiums values(2,'Emirates Stadium','London',60432,3);

insert into stadiums values(3,'St James'' Park','Newcastle-upon-Tyne',52401,4);

insert into stadiums values(4,'Stadium of Light','Sunderland',49000,56);

insert into stadiums values(5,'Etihad Stadium','Manchester',48000,43);

insert into stadiums values(6,'Anfield','Liverpool',45362,14);

insert into stadiums values(7,'Villa Park','Birmingham',42785,7);

insert into stadiums values(8,'Stamford Bridge','London',41623,8);

insert into stadiums values(9,'Goodison Park','Liverpool',40569,11);

insert into stadiums values(12,'White Hart Lane','London',36274,6);

insert into stadiums values(13,'Craven Cottage','London',25678,54);

insert into stadiums values(14,'DW Stadium','Wigan',25023,111);

insert into stadiums values(15,'Loftus Road','London',18360,52);

insert into stadiums values(16,'Macron Stadium','Horwich',28723,30);

insert into stadiums values(17,'Britannia Stadium','Stoke-on-Trent',28323,110);

insert into stadiums values(18,'Liberty Stadium','Swansea',20700,80);

insert into stadiums values(19,'Carrow Road','Norwich',27220,45);

insert into stadiums values(20,'The Hawthorns','West Bromwich',26500,35);

insert into stadiums values(21,'Molineux','Wolverhampton',31700,39);

insert into stadiums values(22,'Ewood Park','Blackburn',31367,5);

-- Load the csv data into the holderTable using the

load data local infile '~/Premier.csv'

into table HolderTable

fields terminated by ','

enclosed by '"'

lines terminated by '\r\n'

ignore 1 rows;

-- Move the data from the holder table to the correct tables

-- Insert into Player

insert into Player (Player\_ID, TeamID, PL\_name, PL\_surname)

select distinct player\_id, team\_a\_id, player\_name, player\_surname from HolderTable where player\_id!='0';

-- Insert into Matches

insert into Matches (Team\_A, Team\_B, M\_date)

select distinct team\_a\_name, team\_b\_name, m\_date from HolderTable where team\_a\_name!='';

-- Insert into Player\_stats

insert into Player\_stats (Team\_A\_ID, Team\_B\_ID, Player\_ID, M\_date, MinPlayed, Goals, Shot\_on, Shot\_off, Penalty, Pass\_OK, Pass\_KO)

select team\_a\_name, team\_b\_name, player\_id, m\_date, minutes\_played, goal, shots\_on\_target, shots\_off\_target, penalties, pass\_success, pass\_fail from HolderTable where player\_id!='0' and team\_a\_name!='';

-- Create the Dimensional tables

create table DimPlayer (

player\_sk int not null auto\_increment,

player\_name varchar(255),

player\_surname varchar(255),

primary key (player\_sk)

);

create table DimTeam (

team\_sk int not null auto\_increment,

team\_name varchar(255),

year\_of\_foundation int,

primary key (team\_sk)

);

create table DimTime (

date\_sk int not null auto\_increment ,

year int,

month int,

day int,

primary key (date\_sk)

);

create table DimStadium (

stadium\_sk int not null auto\_increment,

stadium\_name varchar(255),

stadium\_city varchar(255),

capacity int,

primary key (stadium\_sk)

);

-- Fact table

create table Fact\_Stats (

date\_sk int,

player\_sk int,

team\_sk int,

opponent\_sk int,

stadium\_sk int,

min\_played int,

goals int,

shot\_on int,

shot\_off int,

penalty int,

pass\_ok int,

pass\_ko int,

foreign key (date\_sk) references DimTime(date\_sk),

foreign key (player\_sk) references DimPlayer(player\_sk),

foreign key (team\_sk) references DimTeam(team\_sk),

foreign key (stadium\_sk) references DimStadium(stadium\_sk),

foreign key (opponent\_sk) references DimTeam(team\_sk)

);

-- Begin the etl process

-- Create the staging tables

-- Insert the information from the relational model to the Dimensional Model

-- Transfer the player data from Player > player\_stage > DimPlayer

create table player\_stage (

sourceDB int,

player\_sk int not null auto\_increment,

player\_id int,

player\_name varchar(255),

player\_surname varchar(255),

primary key (player\_sk)

);

-- Insert the data into the staging table

insert into player\_stage (sourceDB, player\_id, player\_name, player\_surname) select 1, player\_id, PL\_name, PL\_surname from Player;

-- Insert the staged data into the dimensional model

insert into DimPlayer (player\_name, player\_surname) select player\_name, player\_surname from player\_stage;

-- Transfer the Teams data from Teams > team\_stage > DimTeam

create table team\_stage (

team\_id int,

team\_sk int not null auto\_increment,

sourceDB int,

team\_name varchar(255),

year\_of\_foundation int,

primary key(team\_sk)

);

-- insert the data to the staging table

insert into team\_stage (team\_id, sourceDB, team\_name, year\_of\_foundation) select TeamID, 1, Team\_name, YearOfFound from Teams;

-- insert the data from the staging table into the dimensional model

insert into DimTeam (team\_name, year\_of\_foundation) select team\_name, year\_of\_foundation from team\_stage;

-- Transfer the Stadiums data from Stadiums > stadium\_stage > DimStadium

create table stadium\_stage (

stadiumID int,

stadium\_sk int not null auto\_increment,

sourceDB int,

st\_name varchar(255),

city varchar(255),

capacity int,

team\_id int,

primary key(stadium\_sk)

);

-- Insert the data into the staging table

insert into stadium\_stage (stadiumID, sourceDB, st\_name, city, capacity, team\_id) select StadiumID, 1, St\_name, City, Capacity, TeamID from Stadiums;

-- Insert the data into the Dim Table

insert into DimStadium (stadium\_sk, stadium\_name, stadium\_city, capacity) select stadium\_sk, st\_name, city, capacity from stadium\_stage;

-- Transfer the Time data from Matches > time\_stage > DimTime

create table time\_stage (

sourceDB int,

date\_sk int not null auto\_increment,

team\_a varchar(255),

team\_b varchar(255),

m\_date DATE,

year int,

month int,

day int,

primary key(date\_sk)

);

-- insert data into the staging table

insert into time\_stage (sourceDB, team\_a, team\_b, m\_date, year, month, day) select 1, Team\_A, Team\_B, M\_date, extract(year from m\_date), extract(month from m\_date), extract(day from m\_date) from Matches;

-- insert data into the Dim Table

insert into DimTime (date\_sk, year, month, day) select date\_sk, year, month, day from time\_stage;

-- insert the data into the fact table

-- Transfer data from Player\_stats > fact\_stage > Fact\_Stats

create table fact\_stage (

date\_sk int,

player\_sk int,

team\_sk int,

opponent\_sk int,

stadium\_sk int,

sourceDB int,

team\_a\_id varchar(255),

team\_b\_id varchar(255),

player\_id int,

m\_date date,

stadium\_id int,

min\_played int,

goals int,

shot\_on int,

shot\_off int,

penalty int,

pass\_ok int,

pass\_ko int

);

-- insert into the fact\_stage table

insert into fact\_stage (sourceDB, team\_a\_id, team\_b\_id, player\_ID, m\_date, min\_played, goals, shot\_on, shot\_off, penalty, pass\_ok, pass\_ko) select 1, Team\_A\_ID, Team\_B\_ID, Player\_ID, M\_date, MinPlayed, Goals, Shot\_on, Shot\_off, Penalty, Pass\_OK, Pass\_KO from Player\_stats;

-- Update the Fact stage

update fact\_stage set stadium\_id=(

select stadium\_stage.stadiumID from stadium\_stage

join team\_stage on stadium\_stage.team\_id = team\_stage.team\_id

where (

stadium\_stage.sourceDB = fact\_stage.sourceDB

and stadium\_stage.team\_id = team\_stage.team\_id

and team\_stage.team\_name = fact\_stage.team\_a\_id

)

);

-- Update the surrogate keys within the fact table

-- Update the player stage

update fact\_stage set player\_sk=(

select player\_stage.player\_sk from player\_stage where (

player\_stage.sourceDB = fact\_stage.sourceDB

and player\_stage.player\_id = fact\_stage.player\_id

)

);

-- Update the Teams stage (gets the players current team)

update fact\_stage set team\_sk=(

select team\_stage.team\_sk from team\_stage where (

team\_stage.sourceDB = fact\_stage.sourceDB

and team\_stage.team\_name = fact\_stage.team\_a\_id

)

);

-- Update the Opposition stage (gets the players oppositon teams)

update fact\_stage set opponent\_sk=(

select team\_stage.team\_sk from team\_stage where (

team\_stage.sourceDB = fact\_stage.sourceDB

and team\_stage.team\_name = fact\_stage.team\_b\_id

)

);

-- Update the Stadiums stage

update fact\_stage set stadium\_sk=(

select stadium\_stage.stadium\_sk from stadium\_stage

join team\_stage on stadium\_stage.team\_id = team\_stage.team\_id

where (

stadium\_stage.sourceDB = fact\_stage.sourceDB

and stadium\_stage.team\_id = team\_stage.team\_id

and team\_stage.team\_name = fact\_stage.team\_a\_id

)

);

-- Update the time stage

update fact\_stage set date\_sk=(

select time\_stage.date\_sk from time\_stage where (

time\_stage.sourceDB = fact\_stage.sourceDB

and time\_stage.team\_a = fact\_stage.team\_a\_id

and time\_stage.team\_b = fact\_stage.team\_b\_id

and time\_stage.m\_date = fact\_stage.m\_date

)

);

-- insert into the Fact\_Stats table

insert into Fact\_Stats (date\_sk, player\_sk, team\_sk, opponent\_sk, stadium\_sk, min\_played, goals, shot\_on, shot\_off, penalty, pass\_ok, pass\_ko) select date\_sk, player\_sk, team\_sk, opponent\_sk, stadium\_sk, min\_played, goals, shot\_on, shot\_off, penalty, pass\_ok, pass\_ko from fact\_stage;

-- Undergo the second etl below

-- Load the second set of data

load data local infile '~/ETL2.csv'

into table HolderTable

fields terminated by ','

enclosed by '"'

lines terminated by '\r\n'

ignore 1 rows;

-- Insert into Player

insert into Player (Player\_ID, TeamID, PL\_name, PL\_surname)

select distinct player\_id, team\_a\_id, player\_name, player\_surname

from HolderTable where player\_id!='0' and not exists (

select \* from Player where HolderTable.player\_id=Player.Player\_ID

);

-- Insert into Matches

insert into Matches (Team\_A, Team\_B, M\_date)

select distinct team\_a\_name, team\_b\_name, m\_date

from HolderTable where team\_a\_name!='' and not exists (

select \* from Matches where

HolderTable.team\_a\_name = Matches.Team\_A and

HolderTable.team\_b\_name = Matches.Team\_B and

HolderTable.m\_date = Matches.M\_date

);

-- Insert into Player\_stats

insert into Player\_stats (Team\_A\_ID, Team\_B\_ID, Player\_ID, M\_date, MinPlayed, Goals, Shot\_on, Shot\_off, Penalty, Pass\_OK, Pass\_KO)

select team\_a\_name, team\_b\_name, player\_id, m\_date, minutes\_played, goal, shots\_on\_target, shots\_off\_target, penalties, pass\_success, pass\_fail

from HolderTable where player\_id!='0' and team\_a\_name!='' and not exists (

select \* from Player\_stats where

HolderTable.team\_a\_name = Player\_stats.Team\_A\_ID and

HolderTable.team\_b\_name = Player\_stats.Team\_B\_ID and

HolderTable.m\_date = Player\_stats.M\_date and

HolderTable.player\_id = Player\_stats.Player\_ID

);

-- Insert the data into the staging table checking to make sure the data doesnt already exist

-- Insert the data into the player\_stage table

insert into player\_stage (sourceDB, player\_id, player\_name, player\_surname) select 1, player\_id, PL\_name, PL\_surname from Player

where not exists (

select \* from player\_stage where

Player.Player\_ID=player\_stage.player\_id

);

-- Insert the data into the DimPlayer table

insert into DimPlayer (player\_sk, player\_name, player\_surname)

select player\_sk, player\_name, player\_surname from player\_stage where not exists (

select \* from DimPlayer where

player\_stage.player\_sk=DimPlayer.player\_sk

);

-- Insert the data into the team\_stage table

insert into team\_stage (sourceDB, team\_id, team\_name, year\_of\_foundation) select 1, TeamID, Team\_name, YearOfFound from Teams where not exists (

select \* from team\_stage where

Teams.TeamID=team\_stage.team\_id

);

-- Insert the data into the DimTeam table

insert into DimTeam (team\_sk, team\_name, year\_of\_foundation) select team\_sk, team\_name, year\_of\_foundation from team\_stage where not exists(

select \* from DimTeam where

team\_stage.team\_sk=DimTeam.team\_sk

);

-- insert the data into the stadium\_stage table

insert into stadium\_stage (stadiumID, sourceDB, st\_name, city, capacity, team\_id) select StadiumID, 1, St\_name, City, Capacity, TeamID from Stadiums where not exists (

select \* from stadium\_stage where

Stadiums.StadiumID=stadium\_stage.stadiumID

);

-- insert the data into the DimStadium table

insert into DimStadium (stadium\_sk, stadium\_name, stadium\_city, capacity) select stadium\_sk, stadiumID, city, capacity from stadium\_stage where not exists (

select \* from DimStadium where

stadium\_stage.stadium\_sk=DimStadium.stadium\_sk

);

-- insert data into the staging table

insert into time\_stage (sourceDB, team\_a, team\_b, m\_date, year, month, day) select 1, Team\_A, Team\_B, M\_date, extract(year from m\_date), extract(month from m\_date), extract(day from m\_date) from Matches where not exists (

select \* from time\_stage where

Matches.Team\_A=time\_stage.team\_a and

Matches.Team\_B=time\_stage.team\_b and

Matches.M\_date=time\_stage.m\_date

);

-- insert data into the Dim Table

insert into DimTime (date\_sk, year, month, day) select date\_sk, year, month, day from time\_stage where not exists (

select \* from DimTime where

time\_stage.date\_sk=DimTime.date\_sk

);

-- insert into the fact\_stage table

insert into fact\_stage (sourceDB, team\_a\_id, team\_b\_id, player\_ID, m\_date, min\_played, goals, shot\_on, shot\_off, penalty, pass\_ok, pass\_ko) select 1, Team\_A\_ID, Team\_B\_ID, Player\_ID, M\_date, MinPlayed, Goals, Shot\_on, Shot\_off, Penalty, Pass\_OK, Pass\_KO from Player\_stats where not exists (

select \* from fact\_stage where

Player\_stats.Team\_A\_ID=fact\_stage.team\_a\_id and

Player\_stats.Team\_B\_ID=fact\_stage.team\_b\_id and

Player\_stats.M\_date=fact\_stage.m\_date and

Player\_stats.Player\_ID=fact\_stage.player\_ID

);

-- Update the surrogate keys within the fact table

-- Update the fact\_stage

update fact\_stage set stadium\_id=(

select stadium\_stage.stadiumID from stadium\_stage

join team\_stage on stadium\_stage.team\_id = team\_stage.team\_id

where (

stadium\_stage.sourceDB = fact\_stage.sourceDB

and stadium\_stage.team\_id = team\_stage.team\_id

and team\_stage.team\_name = fact\_stage.team\_a\_id

)

);

-- Update the player stage

update fact\_stage set player\_sk=(

select player\_stage.player\_sk from player\_stage where (

player\_stage.sourceDB = fact\_stage.sourceDB

and player\_stage.player\_id = fact\_stage.player\_id

)

);

-- Update the Teams stage (gets the players current team)

update fact\_stage set team\_sk=(

select team\_stage.team\_sk from team\_stage where (

team\_stage.sourceDB = fact\_stage.sourceDB

and team\_stage.team\_name = fact\_stage.team\_a\_id

)

);

-- Update the Opposition stage (gets the players oppositon teams)

update fact\_stage set opponent\_sk=(

select team\_stage.team\_sk from team\_stage where (

team\_stage.sourceDB = fact\_stage.sourceDB

and team\_stage.team\_name = fact\_stage.team\_b\_id

)

);

-- Update the Stadiums stage

update fact\_stage set stadium\_sk=(

select stadium\_stage.stadium\_sk from stadium\_stage

join team\_stage on stadium\_stage.team\_id = team\_stage.team\_id

where (

stadium\_stage.sourceDB = fact\_stage.sourceDB

and stadium\_stage.team\_id = team\_stage.team\_id

and team\_stage.team\_name = fact\_stage.team\_a\_id

)

);

-- Update the time stage

update fact\_stage set date\_sk=(

select time\_stage.date\_sk from time\_stage where (

time\_stage.sourceDB = fact\_stage.sourceDB

and time\_stage.team\_a = fact\_stage.team\_a\_id

and time\_stage.team\_b = fact\_stage.team\_b\_id

and time\_stage.m\_date = fact\_stage.m\_date

)

);

-- insert into the Fact\_Stats table

insert into Fact\_Stats (date\_sk, player\_sk, team\_sk, opponent\_sk, stadium\_sk, min\_played, goals, shot\_on, shot\_off, penalty, pass\_ok, pass\_ko) select date\_sk, player\_sk, team\_sk, opponent\_sk, stadium\_sk, min\_played, goals, shot\_on, shot\_off, penalty, pass\_ok, pass\_ko from fact\_stage;

-- Second etl completed

-- Query the database

-- Get a players top 5 performances by a player measured by number of goals scored

-- and display where they where and who the opponent was

-- The example below shows the stats for the player with first name "Clint"

select player\_name, stadium\_name, team\_name, goals from Fact\_Stats

join DimStadium on Fact\_Stats.stadium\_sk=DimStadium.stadium\_sk

join DimPlayer on Fact\_Stats.player\_sk=DimPlayer.player\_sk

join DimTeam on Fact\_stats.opponent\_sk=DimTeam.team\_sk

where player\_name="Clint"

order by goals desc

limit 5;

-- Get player with the highest pass rate on a team for a given year

-- Example below shows the results for the Manchester United team

select distinct player\_name, player\_surname, team\_name, pass\_ok, year from Fact\_Stats

join DimTeam on Fact\_Stats.team\_sk=DimTeam.team\_sk

join DimPlayer on Fact\_Stats.player\_sk=DimPLayer.player\_sk

join DimTime on Fact\_Stats.date\_sk=DimTime.date\_sk

where team\_name="Manchester United" and year="2012"

order by pass\_ok desc;

-- Everything above tested and passed