ETL Lab Explanation

# Team Stage table

*drop table team\_stage;*

*create table team\_stage(*

*sourceDB integer,*

*team\_sk integer,*

*team\_id integer,*

*team\_name varchar(255)*

*);*

This is the code to create the team stage. There is no normalization required for the team dimension table so a sequence to create the surrogate keys and a trigger to fill the keys are only necessary the rest of the tables also have identical sequence and trigger tables. As below.

*drop sequence team\_stage\_seq;*

*create sequence team\_stage\_seq*

*start with 1*

*increment by 1*

*nomaxvalue;*

*drop trigger team\_stage\_trigger;*

*create trigger team\_stage\_trigger*

*before insert on team\_stage*

*for each row*

*begin*

*select team\_stage\_seq.nextval into :new.team\_sk from dual;*

*end;*

The two databases are then inserted into the stage as below.

*insert into team\_stage (sourcedb, team\_id, team\_name)*

*select 1, team\_id, team\_name from Team1;*

*insert into team\_stage (sourcedb, team\_id, team\_name)*

*select 2, team\_id, team\_name from Team2;*

The data is then loaded into the dimensional model as follows.

*insert into TeamDim select team\_sk, team\_name from team\_stage;*

# Player Stage Table

*drop table p\_stage;*

*create table p\_stage(*

*sourceDB integer,*

*p\_sk integer,*

*p\_id integer,*

*p\_name varchar(50),*

*p\_sname varchar(50),*

*team\_id integer*

*);*

This is the code to create the player stage table.

*insert into team\_stage (sourcedb, team\_id, team\_name)*

*select 1, team\_id, team\_name from Team1;*

*insert into team\_stage (sourcedb, team\_id, team\_name)*

*select 2, team\_id, team\_name from Team2;*

This is the code to input the two tables into the stage tables.

*insert into TeamDim select team\_sk, team\_name from team\_stage;*

This is the code to input the data into the team dimensional model.

# Player Stage table

*drop table p\_stage;*

*create table p\_stage(*

*sourceDB integer,*

*p\_sk integer,*

*p\_id integer,*

*p\_name varchar(50),*

*p\_sname varchar(50),*

*team\_id integer*

*);*

This is the code to create the player stage table.

*insert into p\_stage (sourcedb, p\_id, p\_name, p\_sname, team\_id)*

*select 1, p\_id, p\_name, p\_sname, team\_id from Players1;*

*insert into p\_stage (sourcedb, p\_id, p\_name, p\_sname, team\_id)*

*select 2, p\_id, p\_name, p\_sname, team\_id from Players2;*

The following is the code to insert the two databases into the player stage table.

insert into PlayerDim

*select p\_stage.p\_sk, p\_stage.p\_name || ' ' || p\_stage.p\_sname from p\_stage where p\_stage.sourceDB = 1;*

*insert into PlayerDim*

*select p\_stage.p\_sk, p\_stage.p\_name || ' ' || p\_stage.p\_sname*

*from p\_stage*

*where not exists (*

*select player\_name*

*from PlayerDim*

*where PlayerDim.player\_name = p\_stage.p\_name || ' ' || p\_stage.p\_sname*

*) and sourcedb = 2;*

There is duplicate players from the two databases so the not exists() function is used to stop the duplicate values.

# Tournament Stage Table

*drop table t\_stage;*

*create table t\_stage(*

*sourceDB integer,*

*t\_sk integer,*

*t\_id integer,*

*t\_descriprion varchar(100),*

*t\_date date,*

*total\_price float*

*);*

This is the code to create the tournament stage table.

*insert into t\_stage (sourcedb, t\_id, t\_descriprion, t\_date, total\_price)*

*select 1, t\_id, t\_descriprion, t\_date, total\_price from Tournament1;*

*insert into t\_stage (sourcedb, t\_id, t\_descriprion, t\_date, total\_price)*

*select 2, t\_id, t\_descriprion, t\_date, total\_price from Tournament2;*

This is the code to insert the data for the two databases into the stage table.

*update t\_stage*

*set total\_price = total\_price \* 0.7*

*where sourcedb = 2;*

This updates the second database table to convert the currency to euro.

*insert into TournamentDim select t\_sk, t\_descriprion, total\_price from t\_stage;*

This inserts it after the normalization into the tournament dimensional model.

# Date Stage Table

*drop table d\_stage;*

*create table d\_stage(*

*d\_sk integer,*

*day integer,*

*month integer,*

*year integer,*

*week integer,*

*quarter integer,*

*dayOfWeek varchar(20),*

*t\_date date*

*);*

This is the code to create the date stage table.

*insert into d\_stage( day, year, month, week, quarter, dayOfWeek,t\_date)*

*select*

*cast(to\_char(t\_date,'DD') as integer),*

*cast(to\_char(t\_date,'YYYY') as integer),*

*cast(to\_char(t\_date,'MM') as integer),*

*to\_number(to\_char(to\_date(t\_date,'DD/MM/YYYY'),'WW')),*

*to\_number(to\_char(to\_date(t\_date,'DD/MM/YYYY'),'Q')),*

*to\_number(to\_char(to\_date(t\_date,'DD/MM/YYYY'),'D')),*

*t\_date from tournament1;*

*insert into d\_stage( day, year, month, week, quarter, dayOfWeek,t\_date)*

*select*

*cast(to\_char(t\_date,'DD') as integer),*

*cast(to\_char(t\_date,'YYYY') as integer),*

*cast(to\_char(t\_date,'MM') as integer),*

*to\_number(to\_char(to\_date(t\_date,'DD/MM/YYYY'),'WW')),*

*to\_number(to\_char(to\_date(t\_date,'DD/MM/YYYY'),'Q')),*

*to\_number(to\_char(to\_date(t\_date,'DD/MM/YYYY'),'D')),*

*t\_date from tournament2;*

This code is used to convert the single full date into day, month, year, week, quarter, and day of the week.

*insert into DateDim select d\_sk, day, month, year, week, quarter, dayOfWeek from d\_stage;*

This code is then used to insert it into the dimensional model.

# Fact Stage Table

*/\*creating fact table\*/*

*drop table facts\_stage;*

*create table facts\_stage(*

*p\_sk integer,*

*t\_sk integer,*

*team\_sk integer,*

*d\_sk integer,*

*p\_id integer,*

*team\_id integer,*

*t\_id integer,*

*rank integer,*

*price float,*

*sourcedb integer,*

*full\_date date*

*);*

This is the table created for the facts stage table. It has both surrogate keys and the old ids so that you can find the correct surrogate keys for each joining table.

*insert into facts\_stage(t\_id, p\_id, team\_id, full\_date, rank, price, sourcedb)*

*select r.t\_id, r.p\_id, p.team\_id, t.t\_date, r.rank, r.price, 1 from Results1 r, Players1 p , Tournament1 t*

*where r.p\_id = p.p\_id and r.t\_id = t.t\_id;*

*insert into facts\_stage(t\_id, p\_id, team\_id, full\_date, rank, price, sourcedb)*

*select r.t\_id, r.p\_id, p.team\_id, t.t\_date, r.rank, r.price, 2 from Results2 r, Players2 p , Tournament2 t*

*where r.p\_id = p.p\_id and r.t\_id = t.t\_id;*

This code is to fill the fact stage table with enough data to get the correct surrogate keys from the dimension tables.

*update facts\_stage*

*set team\_sk=*

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

*team\_stage where (team\_stage.sourceDB=facts\_stage.sourceDB and*

*team\_stage.team\_id = facts\_stage.team\_id));*

*update facts\_stage*

*set p\_sk=*

*(select p\_stage.p\_sk from*

*p\_stage where (p\_stage.sourceDB=facts\_stage.sourceDB and*

*p\_stage.p\_id = facts\_stage.p\_id));*

*/\*Tournament SK\*/*

*update facts\_stage*

*set t\_sk =*

*(select t\_stage.t\_sk from*

*t\_stage where (t\_stage.sourceDB=facts\_stage.sourceDB and*

*t\_stage.t\_id = facts\_stage.t\_id));*

*update facts\_stage*

*set d\_sk=*

*(select d\_stage.d\_sk from*

*d\_stage where (d\_stage.t\_date=facts\_stage.full\_date));*

This code is then used to get the surrogate keys and fill the facts stage.

*update facts\_stage*

*set price = price \* 0.7*

*where sourcedb = 2;*

*update facts\_stage*

*set t\_sk =1*

*where t\_sk = 7 and sourceDB = 2;*

*update facts\_stage*

*set p\_sk = 1*

*where p\_sk = 7 and sourceDB = 2;*

*update facts\_stage*

*set p\_sk = 5*

*where p\_sk = 8 and sourceDB = 2;*

This fixes the duplicate keys in the fact tables to do with teams and players. It also changes the prices to euro for the second database.

*insert into fact\_results(player\_sk, tournament\_sk, team\_sk, date\_sk, rank, price)*

*select p\_sk, t\_sk, team\_sk, d\_sk, rank, price from facts\_stage;*

The data is then inserted into the facts results table.

# Second Load

*INSERT INTO PLAYERS1 (P\_ID, P\_NAME, P\_SNAME, TEAM\_ID) VALUES (7, 'Alan', 'Parker', 1);*

*INSERT INTO PLAYERS1 (P\_ID, P\_NAME, P\_SNAME, TEAM\_ID) VALUES (8, 'Martha', 'Bag', 2);*

*INSERT INTO TOURNAMENT1 (T\_ID, T\_DESCRIPRION, t\_date, TOTAL\_PRICE) VALUES (5, 'Saudi Open', '22-Nov-14', 500000);*

*INSERT INTO RESULTS1 (T\_ID, P\_ID, RANK, PRICE) VALUES (5, 1, 1, 60000);*

*INSERT INTO RESULTS1 (T\_ID, P\_ID, RANK, PRICE) VALUES (5, 7, 5, 20000);*

*INSERT INTO RESULTS1 (T\_ID, P\_ID, RANK, PRICE) VALUES (2, 8, 3, 1000);*

This new data is then loaded into database one.

*insert into p\_stage ( sourcedb, p\_id, p\_name, p\_sname, team\_id)*

*select 1, p\_id, p\_name, p\_sname, team\_id from players1*

*where not exists( select \* from p\_stage p where p.p\_id = players1.p\_id and sourcedb = 1 );*

The above code is used to add only the new data inputted into the player stage. This is done using the not exists() function.

*insert into PlayerDim*

*select p\_stage.p\_sk, p\_stage.p\_name || ' ' || p\_stage.p\_sname from p\_stage*

*where not exists(select \* from PlayerDim where PlayerDim.player\_sk = p\_stage.p\_sk) and sourcedb = 1;*

This code is done in a very similar way to the previous snippet to add the new rows from the player stage to the player dimensional model.

*insert into t\_stage (sourcedb, t\_id, t\_descriprion, total\_price)*

*select 1, t\_id, t\_descriprion, total\_price from TOURNAMENT1*

*where not exists ( select \* from t\_stage t where t.t\_id = tournament1.t\_id and sourcedb = 1 );*

*insert into TournamentDim*

*select ts.t\_sk, ts.t\_descriprion, ts.total\_price from t\_stage ts*

*where not exists (select \* from TournamentDim t where t.tournament\_sk = ts.t\_sk ) and sourcedb = 1;*

The same is done for the tournament table as the player table.

insert into facts\_stage(t\_id, p\_id, team\_id, full\_date, rank, price, sourcedb)

select r.t\_id, r.p\_id, p.team\_id, t.t\_date, r.rank, r.price, 1 from Results1 r, Players1 p , Tournament1 t

where r.p\_id = p.p\_id and r.t\_id = t.t\_id and

NOT EXISTS (SELECT \* FROM facts\_stage fs

WHERE fs.t\_id = r.t\_id and fs.p\_id = r.p\_id and fs.team\_id = p.team\_id and fs.full\_date = t.t\_date and fs.rank =r.rank and fs.price = r.price

and fs.sourcedb = 1 )

This code is to add only new data to the facts stage table.

*update facts\_stage*

*set team\_sk=*

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

*team\_stage where (team\_stage.sourceDB=facts\_stage.sourceDB and*

*team\_stage.team\_id = facts\_stage.team\_id));*

*update facts\_stage*

*set p\_sk=*

*(select p\_stage.p\_sk from*

*p\_stage where (p\_stage.sourceDB=facts\_stage.sourceDB and*

*p\_stage.p\_id = facts\_stage.p\_id));*

*/\*Tournament SK\*/*

*update facts\_stage*

*set t\_sk =*

*(select t\_stage.t\_sk from*

*t\_stage where (t\_stage.sourceDB=facts\_stage.sourceDB and*

*t\_stage.t\_id = facts\_stage.t\_id));*

*update facts\_stage*

*set d\_sk=*

*(select d\_stage.d\_sk from*

*d\_stage where (d\_stage.t\_date=facts\_stage.full\_date));*

The surrogate keys are again gotten through the above keys.

*insert into fact\_results(player\_sk, tournament\_sk, team\_sk, date\_sk, rank, price)*

*select p\_sk, t\_sk, team\_sk, d\_sk, rank, price from facts\_stage fs where*

*not exists(select \* from fact\_results fr where fr.player\_sk = fs.p\_sk*

*and fr.tournament\_sk = fs.t\_sk and fr.team\_sk = fs.t\_sk and fr.date\_sk = fs.d\_sk ) and sourcedb = 1;*

The new data in the facts stage is then inputted into the facts results and the second load is finished.