Assignment 2 : Railway Query

Subhasish Basak

MDS201803

CREATE TABLE filter\_sch (SELECT T1.train,

T1.station,

T1.arr,

T1.arrdate,

T1.dep,

T1.depdate,

terminus name FROM

schedule AS T1,

terminus

WHERE

T1.station = terminus.station);

*#The table Filter\_sch is obtained by filtering over the stations in the schedule table,by the stations in the terminus table. In other words it gives the schedule corresponding those stations present in terminus.*

CREATE TABLE No\_change (SELECT M1.station AS StationA,

M2.station AS StationB,

M1.train AS Train1 FROM

filter\_sch AS M1,

filter\_sch AS M2

WHERE

M1.train = M2.train

AND M1.station != M2.station

AND ((M2.arr IS NOT NULL

AND M1.dep IS NOT NULL)

XOR (M2.arr IS NOT NULL

AND M1.dep IS NOT NULL

AND M1.arr IS NOT NULL

AND M2.dep IS NOT NULL

AND M1.arrdate <= M2.arrdate)));

*#This query creates table of all possible pairs of stations listed in terminus,which can be travelled in a single train journey.*

*Here we take a Cartesian product of filter\_sch with itself, such that the following conditions hold:*

*1. Two of them have different stations*

*2. Two of them have same train*

*And the following two conditions combined with Exclusive OR command..(since we want exactly one of them to occur)*

*3. station in M1 is not the last stop & station in M2 is not the first stop*

*=> exclusive occurance of this case includes all the following journeys:*

1. *from a starting station to a run-through station :*

*M2.arr = not null and M1.dep = not null and M1.arr = null and M2.dep = not null*

1. *from a run-through station to a ending station :*

*M2.arr = not null and M1.dep = not null and M1.arr = not null and M2.dep = null*

1. *from a starting station to a ending station :*

*M2.arr = not null and M1.dep = not null and M1.arr = null and M2.dep = null*

*4. Both the stations in M1 and M2 are middle stations with arrival date if M1.station <= that of M2 counterpart.*

*Together with 3 & 4 we cover all possible combinations*

*We found that there are 93 routes by which one can reach a terminus station from another.*

CREATE TABLE t2Rt (SELECT M1.station AS StationA,

M2.station AS Changest,

M1.train AS Train1,

M2.arrdate AS arrdate,

M2.arr AS arr FROM

filter\_sch AS M1,

schedule AS M2

WHERE

M1.train = M2.train

AND M1.station != M2.station

AND (((M1.depdate = M2.arrdate)

AND (M1.dep < M2.arr))

OR (M1.depdate < M2.arrdate)));

*#This command creates table t2Rt and gives the routes of trains from a terminus station till all possible run-through stations,such that:*

*1.The train starts from a terminus station, stops at a different run through station*

*2.In cartesian product to avoid wrong direction trains there should be a time & date contraint:*

*if the depdate from terminus station and arrival date in the runthrough station is same then M1.dep<M2.arr*

*(M1.dep<M2.arr) => departure time from terminus station is less than arrival time in the destination station*

*otherwise it is sufficient to have M1.depdate<M2.arrdate*

*(M1.depdate<M2.arrdate) => the departure date from terminus station is less than arrival date in the destination station*

CREATE TABLE Rt2t (SELECT M1.station AS Changest1,

M2.station AS StationB,

M1.train AS Train2,

M1.depdate AS depdate,

M1.dep AS dep FROM

schedule AS M1,

filter\_sch AS M2

WHERE

M1.train = M2.train

AND M1.station != M2.station

AND (((M1.depdate = M2.arrdate)

AND (M1.dep < M2.arr))

OR (M1.depdate < M2.arrdate)));

*#This command creates table Rt2t and gives the routes of trains from a run-through station till all possible terminus stations,such that:*

*1.The train starts from a run through station, stops at a different terminus station*

*2.In cartesian product to avoid wrong direction trains there should be a time & date contraint:*

*if the depdate from run through station and arrival date in the terminus station is same then M1.dep<M2.arr*

*(M1.dep<M2.arr) => departure time from run through station is less than arrival time in the destination station*

*otherwise it is sufficient to have M1.depdate<M2.arrdate*

*(M1.depdate<M2.arrdate) => the departure date from run through station is less than arrival date in the destination station*

*#This following query gives the route from one terminus station to another with one change in between.*

CREATE TABLE One\_Change (SELECT T1.StationA,

T2.StationB,

T1.Train1,

T1.changest AS Change1,

T2.train2 FROM

t2Rt AS T1,

Rt2t AS T2

WHERE

T1.changest = T2.changest1

AND T1.StationA != T2.StationB

AND ((T2.dep - T1.arr + 240000 BETWEEN 10000 AND 50000)

OR (T2.dep - T1.arr BETWEEN 10000 AND 50000)));

*#Here we take the cartesian product of t2Rt and Rt2t such that the following conditions are satisfied:*

*1. The Runthrough station of t2Rt should match the Runthrough station of Rt2t i.e. T1.changest=T2.changest1=The Change1 station*

*2. The terminus station of t2Rt should be different from the terminus station of Rt2t i.e. T1.StationA!=T2.StationB*

*3. The gap of min. 1 hour and max. 5 hours between the arrival of Train1 & departure of Train2 in the Change1 station, is obtained by the following conditions:*

*(T2.dep-T1.arr+240000 between 10000 and 50000) or (T2.dep-T1.arr between 10000 and 50000)*

*We found that there are a total 5827 routes to reach one terminus station from another with one change in between.*

CREATE TABLE Rt2Rt (SELECT M1.station AS Change1,

M2.station AS Change2,

M1.train AS Train2,

M1.dep AS dep1,

M1.depdate AS depdate1,

M2.arr AS arr1,

M2.arrdate AS arradate1 FROM

schedule AS M1,

schedule AS M2

WHERE

M1.train = M2.train

AND M1.station != M2.station

AND (((M1.depdate = M2.arrdate)

AND (M1.dep < M2.arr))

OR (M1.depdate < M2.arrdate)));

*#This query gives the routes of trains from a run-through station till all possible run-through stations,such that:*

*1. There is no wrong direction trains, validated by the following conditions:*

*(((M1.depdate=M2.arrdate) and (M1.dep<M2.arr)) or (M1.depdate<M2.arrdate)))*

*2. M1.train=M2.train and M1.station!=M2.station : validates the journey by a single train from one run-through station till all possible run-through stations*

CREATE TABLE One\_Change\_left (SELECT W1.StationA AS StationA,

W1.Train1 AS Train1,

W2.change1 AS Change1,

W2.Train2 AS Train2,

W2.change2 AS Change2 FROM

t2Rt AS W1,

Rt2Rt AS W2

WHERE

W1.changest = W2.change1

AND W1.StationA != W2.Change2

AND W1.Train1 != W2.Train2

AND ((W2.dep1 - W1.arr + 240000 BETWEEN 10000 AND 50000)

OR (W2.dep1 - W1.arr BETWEEN 10000 AND 50000)));

*#This command creates a table by taking cartesian product of t2Rt and Rt2Rt table, which consists of all possible combinations of StationA,Train1,Change1,Train2,Change2*

*1. (W1.changest=W2.change1) => Here we match the destination station of the 1st table with starting station of the 2nd table*

*2. (W1.StationA!=W2.Change2) => The starting station of the 1st table should be different from the destination station of the 2nd table*

*3. (W1.Train1!=W2.Train2) => The train in 1st table should be different from the train in the 2nd table*

CREATE TABLE One\_change\_right (SELECT W3.StationB AS StationB,

W3.Train2 AS Train3,

W2.change1 AS Change11,

W2.Train2 AS Train22,

W2.change2 AS Change22 FROM

Rt2Rt AS W2,

Rt2t AS W3

WHERE

W2.change2 = W3.changest1

AND W2.Change1 != W3.StationB

AND W2.Train2 != W3.Train2

AND ((W3.dep - W2.arr1 + 240000 BETWEEN 10000 AND 50000)

OR (W3.dep - W2.arr1 BETWEEN 10000 AND 50000)));

*#This command creates a table by taking cartesian product of Rt2Rt and Rt2t table, which consists of all possible combinations of StationB,Train3,Change1,Train2,Change2*

*1. (W2.change2=W3.changest1) => Here we match the destination station of the 1st table with starting station of the 2nd table*

*2. (W2.Change1!=W3.StationB) => The starting station of the 1st table should be different from the destination station of the 2nd table*

*3. (W2.Train2!=W3.Train2) => The train in 1st table should be different from the train in the 2nd table*

*#Next we have to join the 2 tables One\_Change\_left and One\_Change\_right such that the following conditions hold:*

*1. The tuple (Change1,Train2,Change2) is same for the 2 table*

*2. StationA should not match StationB*

*3. StationA is different from Change 2;StationB is different from Change 1*

*4. Train1 , Train2 and Train3 are distinct*

*# Now to run the query faster we use indexing on the required columns viz. Change1,Train2,Change2 as follows:*

create index left\_ind on One\_Change\_left(Change1,Train2,Change2);

create index right\_ind on One\_Change\_right(Change11,Train22,Change22);

CREATE TABLE twice\_change (SELECT StationA, StationB, Train1, Change1, Train2, Change2, Train3 FROM

One\_Change\_left AS W1,

One\_Change\_right AS W2

WHERE

W1.StationA != W2.StationB

AND W1.Train1 != W2.Train3

AND (W1.Change1 , W1.Train2, W1.Change2) = (W2.Change11 , W2.Train22, W2.Change22));

*#The above query matches the conditions stated above and gives the required output of all possible ways of reaching from a terminus station to another by changing two trains.*

*We found that there are 401862 routes in which one can reach one terminus station from another by switching 2 trains in between*

*##Now summing up all the cases:*

create table Final\_output(

select \* from twice\_change

union all

select \*,NULL,NULL from one\_change

union all

select \*,NULL,NULL,NULL,NULL from no\_change);

SELECT

\*

FROM

Final\_output;

*#This table gives the required output and we found total 407782 routes are there to reach one terminus station to another by switching atmost 2 trains in between*

*#Apart from these queries a deep inspection of the given data reveals some irregular observations, viz,*

*1. Trains with the departure time is less than the arrival time on the same travel date*

*2. Trains with the departure time is equal to the arrival time on the same travel date*

*Even if we can consider the 2nd trains since they can be halting at a perticular station for less than 59 seconds which is not incorporated in the data*

*But observations of kind 1 are not justified ,which are indentified by the following command*

CREATE TABLE irr\_obs (SELECT train, station FROM

schedule

WHERE

(arrdate = depdate) AND (arr > dep));

SELECT

\*

FROM

irr\_obs

WHERE

station IN (SELECT

station

FROM

terminus);

*#the above query gives 0 rows => None of the irregular stations correspond to terminal stations*

*So these irregular observations does not affect the No\_change table*

*But it affects One\_change & Twice\_change since they include non-terminus stations*

CREATE TABLE One\_change1 (SELECT \* FROM

One\_Change

WHERE

(Train1 , Change1) NOT IN (SELECT

\*

FROM

irr\_obs)

AND (Train2 , Change1) NOT IN (SELECT

\*

FROM

irr\_obs));

*#The above command removes the non-valid routes corresponding to the irregular observatins from One\_change. We get 5813 routes now.*

CREATE TABLE Twice\_change1 (SELECT \* FROM

Twice\_Change

WHERE

(Train1 , Change1) NOT IN (SELECT

\*

FROM

irr\_obs)

AND (Train2 , Change1) NOT IN (SELECT

\*

FROM

irr\_obs)

AND (Train2 , Change2) NOT IN (SELECT

\*

FROM

irr\_obs)

AND (Train3 , Change2) NOT IN (SELECT

\*

FROM

irr\_obs));

*#The above command removes the non-valid routes corresponding to the irregular observatins from Twice\_change. We get 401060 routes now.*

*#Now summing all together we get the following:*

create table Final\_output\_1(

select \* from twice\_change1

union all

select \*,NULL,NULL from one\_change1

union all

select \*,NULL,NULL,NULL,NULL from no\_change);

SELECT

\*

FROM

Final\_output\_1;

*#This table gives the required output after removing non-valid routes and we found total 406966 routes are there to reach one terminus station to another by switching at most 2 trains in between.*