|  |
| --- |
| **CIS 450/550: Database and Information Systems** |
| **Homework 1 ‒ SQL** |

Due by by 23:59:59 EDT on June 4, 2019

**Part 1: Querying an AWS Oracle database (60 points)**

**IMDB Database.** The database you will use consists of a portion of the Internet Movie Database (IMDB.com). It has been uploaded to Amazon's AWS/RDS and conforms to the following schema (where keys are underlined):

* **movie** (movie\_id: number(7), title: varchar(255), runtime: number(7), release\_year: number(7), rating: float(2), num\_ratings: number(7))
* **genre** (name: varchar(255))
* **crew** (id: number(7), gender: number(7), name: varchar(255))
* **movie\_cast**(id: number(7), gender: number(7), name: varchar(255))

-- Think of this as a relation of actors

* **cast\_in** (movie\_id: number(7), cast\_id: number(7), charac: varchar(255))

-- movie\_id REFERENCES movie(movie\_id)

-- cast\_id REFERENCES movie\_cast(id)

* **crew\_in** (movie\_id: number(7), crew\_id: number(7), job: varchar(255))

-- movie\_id REFERENCES movie(movie\_id)

-- crew\_id REFERENCES crew(id)

* **movie\_genre** (movie\_id: number(7), genre\_name: varchar(255))

-- movie\_id REFERENCES movie(movie\_id)

-- genre\_name REFERENCES genre(name)

* **movie\_keyword** (kwd\_name: varchar(255), movie\_id: number(7))

-- movie\_id REFERENCES movie(movie\_id)

**1. (2 points)**

**Connecting to the Oracle instance.** Using the instructions on the homework stub web page, connect to the instance IMDb that we have set up.

Then find the title and release\_year of the first five movies that occur in the table **movie** by running the query:

SELECT title, release\_year

FROM movie

WHERE ROWNUM <= 5;

For question 1 only, please submit the query *and result*. The remaining questions on the homework ask you to **submit the query only**.

ANSWER:

SELECT title, release\_year

FROM movie

WHERE ROWNUM<=5;

TITLE

----------------------------------------------------------------------

RELEASE\_YEAR

------------

Carmencita

1894

Pauvre Pierrot

1892

Blacksmith Scene

1893

TITLE

----------------------------------------------------------------------

RELEASE\_YEAR

------------

Edison Kinetoscopic Record of a Sneeze

1894

Employees Leaving the Lumi???re Factory

1895

**2. (3 points)**

Print the range of release years of movies. Your result should have a single tuple consisting of the year of the oldest movie and the year of the most recent movie.

Schema: (MinYear, MaxYear)

ANSWER:

SELECT MIN(release\_year)AS MinYear, MAX(release\_year)AS MaxYear

FROM movie;

MINYEAR MAXYEAR

---------- ----------

1874 2018

**3. (5 points)**

Print the number of cast who are also crew. You may assume that id identifies someone as a person, and is that same for that person as a member of the cast or crew. Your result should have a single tuple.

Schema: (Num)

ANSWER:

SELECT COUNT(DISTINCT movie\_cast.id)AS Num

FROM movie\_cast,crew

WHERE movie\_cast.id=crew.id;

NUM

----------

14279

**4. (5 points)**

For each genre in the genre relation, print the number (count) of movies of that genre in movie\_genre. Include genres that do not appear in movie\_genre with a count of 0.

Schema: (genre\_name, num)

ANSWER:

SELECT genre.name AS genre\_name, NVL(MG.num, 0) AS num

FROM genre

LEFT JOIN (

SELECT movie\_genre.genre\_name AS genre\_name, COUNT(\*) AS num

FROM movie\_genre

GROUP BY genre\_name

) MG

ON genre.name=MG.genre\_name

ORDER BY num DESC;

GENRE\_NAME NUM

----------------------------------------------------------------

Drama 21425

Comedy 13548

Romance 6447

Action 6038

Crime 5763

Thriller 5084

Horror 4370

Adventure 4260

Documentary 4130

Mystery 2624

Family 2249

Fantasy 2246

Sci-Fi 2245

Biography 2132

Animation 1876

History 1572

Music 1373

War 1194

Short 951

Musical 863

Western 823

Sport 725

Film-Noir 452

News 90

Adult 17

Reality-TV 3

Talk-Show 2

genres 0

Game-Show 0

29 rows selected.

**5. (5 points)**

For the movie 'The Other Side of the Wind' released in 2018, print the name of all actors and their role in the movie.

Schema: (name, charac)

SELECT movie\_cast.name AS name, cast\_in.charac AS charac

FROM cast\_in, movie\_cast, movie

WHERE cast\_in.cast\_id=movie\_cast.id AND cast\_in.movie\_id=movie.movie\_id AND movie.title='The Other Side of the Wind' AND movie.release\_year=2018;

NAME

----------------------------------------------------------------

CHARAC

----------------------------------------------------------------

John Huston

J.J. Jake Hannaford

Peter Bogdanovich

Brooks Otterlake

Susan Strasberg

Juliet Rich

Oja Kodar

The Actress

Joseph McBride

Charles Pister

Mercedes McCambridge

Maggie Fassbender

Cameron Mitchell

Matt Zimmer

Paul Stewart

Matt Costello

Peter Jason

Marvin P. Fassbender

Tonio Selwart

The Baron

Howard Grossman

Charles Higgam

Geoffrey Land

Max David

Dennis Hopper

Lucas Renard

Gregory Sierra

Jack Simon

Benny Rubin

Abe Vogel

Cathy Luvas

Mavis Henscher

Dan Tobin

Dr. Bradley Pease Burroughs

George Jessel

George

Richard Wilson

Dick

Claude Chabrol

Himself

St??phane Audran

Himself

Curtis Harrington

Himself

Henry Jaglom

Himself

Paul Mazursky

Himself

24 rows selected.

**6. (7 points)**

Print the movie title and director name of all movies with genre ‘Comedy’ that appeared in 2015 and were directed by a director whose (first) name starts with ‘L’.

Schema: (title, name)

ANSWER:

SELECT movie.title AS title, crew.name AS name

FROM movie,crew,movie\_genre,crew\_in

WHERE movie\_genre.genre\_name='Comedy' AND movie.release\_year=2015 AND crew\_in.job='Director' AND crew.name LIKE 'L%' AND crew\_in.crew\_id=crew.id AND crew\_in.movie\_id=movie.movie\_id AND movie\_genre.movie\_id=movie.movie\_id;

TITLE

----------------------------------------------------------------

NAME

----------------------------------------------------------------

Kyle Kinane: I Liked His Old Stuff Better

Lance Bangs

Tales of Halloween

Lucky McKee

Two Friends

Louis Garrel

Louis C.K.: Live at the Comedy Store

Louis C.K.

Twenty

Lee Byeong-heon

The Meddler

Lorene Scafaria

Gary Owen: I Agree with Myself

Leslie Small

Bridget Everett: Gynecological Wonder

Lance Bangs

October Kiss

Lynne Stopkewich

A Gift Wrapped Christmas

Lee Friedlander

All About E

Louise Wadley

Glada h???lsningar fr???n Miss???ngertr???sk

Lisa Siwe

Zombieworld

Luke Guidici

Sleeping with Other People

Leslye Headland

Requirements to Be a Normal Person

Leticia Dolera

15 rows selected.

**7. (7 points)**

Print the title, earliest release year, most recent release year, and number of releases of the movie that has been released the most times. You should assume that two movies are the same if they have the same title.

Schema: (title, earliest, recent, num\_releases)

ANSWER:

SELECT \*

FROM (SELECT title, MIN(release\_year) AS earliest, MAX(release\_year) AS recent, COUNT(\*) AS num\_releases

FROM movie

GROUP BY title

ORDER BY num\_releases DESC)

WHERE ROWNUM<=1;

TITLE

----------------------------------------------------------------

EARLIEST RECENT NUM\_RELEASES

---------- ---------- ------------

Cinderella

1899 2015 12

**8. (8 points)**

For each decade in which movies have been released and each genre of movie in that decade, print the number and the average rating of movies in that genre and decade. Your query must be instance independent, i.e. the decades must be calculated from the input.

Schema: (decade, genre\_name, avg\_rating, num)

ANSWER:

SELECT FLOOR(movie.release\_year/10)\*10 AS decade,movie\_genre.genre\_name,AVG(rating) AS avg\_rating, COUNT(\*) AS num

FROM movie,movie\_genre

WHERE movie.movie\_id=movie\_genre.movie\_id

GROUP BY FLOOR(movie.release\_year/10)\*10,movie\_genre.genre\_name

ORDER BY decade DESC,avg\_rating DESC;

DECADE

----------

GENRE\_NAME

----------------------------------------------------------------AVG\_RATING NUM

---------- ----------

2010

Talk-Show

8 1

2010

News

7.39506173 81

……

1870

Short

7 2

1870

Documentary

7 2

293 rows selected.

**9. (8 points)**

Print name of directors who have **only** directed *near-top rated* movies since 2016. By “near-top” we mean in the interval [m-1, m], where m is the max rating of movies released since 2016. Note that directors are crew rather than actors.

Schema: (name)

ANSWER:

SELECT DISTINCT crew.name

FROM crew\_in,crew,

(SELECT DISTINCT crew\_in.crew\_id, COUNT(\*) AS Num

FROM crew\_in,

(SELECT movie\_id

FROM movie,

(SELECT MAX(movie.rating) AS max\_rating

FROM movie

WHERE movie.release\_year>2016) MAX

WHERE release\_year>2016 AND movie.rating >= MAX.max\_rating-1) MVID

WHERE MVID.movie\_id=crew\_in.movie\_id AND crew\_in.job='Director'

GROUP BY crew\_in.crew\_id) TD,

(SELECT DISTINCT crew\_in.crew\_id, COUNT(\*) AS Num

FROM movie,crew\_in

WHERE release\_year>2016 AND crew\_in.movie\_id=movie.movie\_id AND crew\_in.job='Director'

GROUP BY crew\_in.crew\_id) FD

WHERE crew\_in.crew\_id=TD.crew\_id AND FD.crew\_id=TD.crew\_id AND FD.Num=TD.Num AND crew\_in.crew\_id=crew.id

ORDER BY crew.name

NAME

----------------------------------------------------------------------

Akiyuki Shinbo

Alex Timbers

Amir Bar-Lev

Andrey Zvyagintsev

Beth David

Brian Oakes

Buda Guly??s

Chad Stahelski

Christopher Nolan

Christopher Storer

Daniel Lindsay

……

48 rows selected.

**10. (10 points)**

Print the name of directors who have directed top-rated movies in its genre in at least two different genres, where “top-rated” means the *max rating within the genre*. The movie may be the same in the two genres, or they could be different movies. (**Note:** this is not an easy query!)

Schema: (name)

//genre\_max: 27 rows; max\_genre\_movie\_id: 435 rows; crew\_name: 116 rows

ANSWER:

SELECT crew.name

FROM crew,

(SELECT crew\_in.crew\_id

FROM crew\_in,

(SELECT movie.movie\_id,movie\_genre.genre\_name

FROM movie,movie\_genre,

(SELECT genre\_name, MAX(rating) AS rating

FROM movie,movie\_genre

WHERE movie.movie\_id=movie\_genre.movie\_id

GROUP BY movie\_genre.genre\_name) MAX

WHERE movie.movie\_id=movie\_genre.movie\_id AND movie\_genre.genre\_name=MAX.genre\_name AND movie.rating=MAX.rating) MVID

WHERE MVID.movie\_id=crew\_in.movie\_id AND crew\_in.job='Director'

GROUP BY crew\_in.crew\_id

HAVING COUNT(DISTINCT MVID.genre\_name)>=2) CWID

WHERE CWID.crew\_id=crew.id;

NAME

----------------------------------------------------------------

Richard Loncraine

David Fincher

Bryan Singer

Steven Spielberg

Martin Scorsese

Chuck Jones

Danny Cannon

Chris Cunningham

Francis Ford Coppola

Roman Polanski

Ridley Scott

……

116 rows selected.

**Part 2: Setting up and Querying a MySQL Instance (40 points)**

The goal of the second part of the homework is to get experience with MySQL (MariaDB), one of the most popular open source databases, and to understand how to create your own database instance. You have been provided an account for a MySQL database on fling by CETS, and instructions on how to connect to it can be found at the following address:

<https://cets.seas.upenn.edu/answers/mysql.html>

Alternatively, you can set up your own local MySQL instance using a free download.

You will be creating a database with the following three tables:

**Airlines**, primary key is id:

+----------------+----------------+

| Field | Type |

+----------------+----------------+

| id | int(11) |

| name | varchar(20) |

| alias | varchar(20) |

| iata | char(2) |

| icao | char(3) |

| callsign | varchar(20) |

| country | varchar(20) |

| active | tinyint(1) |

+----------------+----------------+

**Airports,** primary key is id:

+---------------+-----------------+

| Field | Type |

+----------+----------------------+

| id | int(11) |

| name | varchar(20) |

| city | varchar(20) |

| country | varchar(20) |

| iata | char(3) |

| icao | char(4) |

| lat | decimal(8,6)|

| lon | decimal(9,6)|

| alt | int(11) |

| timezone | decimal(3,1)|

| dst | char(1) |

| tz | varchar(20) |

+---------------+-----------------+

**Routes,** with three foreign keys:

* airline\_id is a foreign key referencing airline(id)
* source\_id is a foreign key referencing airport(airport\_id)
* target\_id is a foreign key referencing airport(airport\_id))

+------------------------+-----------------+

| Field | Type |

+------------------------+-----------------+

| airline\_iiata | char(3) |

| airline\_id | int(11) |

| src\_iata\_icao | char(4) |

| source\_id | int(11) |

| target\_iata\_icao | char(4) |

| target\_id | int(11) |

| code\_share | char(1) |

| stops | int(11) |

| equipment | char(20) |

+------------------------+-----------------+

**11. (8 pts)**

Write SQL DDL statements to create the database schema above and enforce the primary and foreign key constraints indicated. Also add a constraint in routes to ensure that attribute codeshare can only be "Y" (if this flight is a codeshare, that is, not operated by Airline, but another carrier) or " " (otherwise).

**Note:**  If you want to see what tables are in your database, use “show tables;” . To see the schema of a table “R”, use “describe R;”.

ANSWER：

MariaDB [zty]> CREATE TABLE Airlines (

-> id int(11),

-> name varchar(20),

-> alias varchar(20),

-> iata char(2),

-> icao char(3),

-> callsign varchar(20),

-> country varchar(20),

-> active tinyint(1),

-> PRIMARY KEY (id)

-> );

**Query OK, 0 rows affected (0.01 sec)**

MariaDB [zty]> CREATE TABLE Airports(

-> id int(11),

-> name varchar(20),

-> city varchar(20),

-> country varchar(20),

-> iata char(3),

-> icao char(4),

-> lat decimal(8,6),

-> lon decimal(9,6),

-> alt int(11),

-> timezone decimal(3,1),

-> dst char(1),

-> tz varchar(20),

-> PRIMARY KEY (id)

-> );

**Query OK, 0 rows affected (0.01 sec)**

MariaDB [zty]> CREATE TABLE Routes(

-> airline\_iiata char(3),

-> airline\_id int(11),

-> src\_iata\_icao char(4),

-> source\_id int(11),

-> target\_iata\_icao char(4),

-> target\_id int(11),

-> code\_share char(1),

-> stops int(11),

-> equipment char(20),

-> FOREIGN KEY (airline\_id) REFERENCES Airlines(id),

-> FOREIGN KEY (source\_id) REFERENCES Airports(id),

-> FOREIGN KEY (target\_id) REFERENCES Airports(id),

-> CHECK (code\_share IN ('Y',' '))

-> );

**Query OK, 0 rows affected (0.01 sec)**

MariaDB [zty]> describe Routes;

+------------------+----------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+------------------+----------+------+-----+---------+-------+

| airline\_iiata | char(3) | YES | | NULL | |

| airline\_id | int(11) | YES | MUL | NULL | |

| src\_iata\_icao | char(4) | YES | | NULL | |

| source\_id | int(11) | YES | MUL | NULL | |

| target\_iata\_icao | char(4) | YES | | NULL | |

| target\_id | int(11) | YES | MUL | NULL | |

| code\_share | char(1) | YES | | NULL | |

| stops | int(11) | YES | | NULL | |

| equipment | char(20) | YES | | NULL | |

+------------------+----------+------+-----+---------+-------+

**9 rows in set (0.00 sec)**

**12. (5 points**)

To populate a database using a text file, enter a MySQL (MariaDB) session using the following (substituting your user name for USERNAME):

mysql -p -u zty -h fling.seas.upenn.edu --local-infile zty

LOAD DATA LOCAL INFILE 'data/file.txt' INTO TABLE relation FIELDS TERMINATED BY ',' ENCLOSED BY '"';

Using the text files **airports.dat.txt**, **airlines.dat.txt** and **routes.dat.txt** provided, populate your database. Your (English) answer to this question should be the order in which you loaded these files, and an explanation of why that order is important.

ANSWER：

1. airlines.dat.txt 2. airports.dat.txt 3. routes.dat.txt

Airlines and airports contain primary keys. So, they need to be loaded first. As there is no relation (foreign key refers to primary key) between these two datasets, so the order of loading this two are randomly. Dataset routes has foreign keys refer the above two tables, so, it needs to be loaded after the two.

**13. (5 points)**

For each city in the United States with more than 3 airports, count the number of airports.

Schema: (city, NumAirports)

ANSWER：

MariaDB [zty]> SELECT city, COUNT(\*) AS NumAirports

-> FROM Airports

-> WHERE country='United States'

-> GROUP BY city

-> Having COUNT(\*)>3;

+-----------------+-------------+

| city | NumAirports |

+-----------------+-------------+

| Anchorage | 5 |

| Atlanta | 5 |

| Baltimore | 5 |

| Boston | 5 |

| Burlington | 4 |

| Chicago | 9 |

| Cleveland | 5 |

| Columbus | 5 |

| Dallas | 4 |

| Dayton | 4 |

| Denver | 5 |

| Detroit | 4 |

| Fort Lauderdale | 4 |

| Georgetown | 4 |

| Houston | 6 |

| Jacksonville | 5 |

| Ketchikan | 4 |

| Las Vegas | 4 |

| Los Angeles | 4 |

| Miami | 5 |

| New York | 14 |

| Orlando | 5 |

| Philadelphia | 6 |

| Phoenix | 4 |

| Sacramento | 5 |

| San Diego | 8 |

| Seattle | 9 |

| Springfield | 4 |

| Tampa | 4 |

| Trenton | 4 |

| Washington | 4 |

| Wilmington | 4 |

+-----------------+-------------+

**32 rows in set (0.01 sec)**

**14. (5 points)**

Print the name of airports in the United States that are the source of a route to another airport, but not the target of some route from that airport. Be sure to eliminate the NULL airport (the data is somewhat dirty). Eliminate duplicates and order by name.

Schema: (airport\_name)

ANSWER：

CREATE TEMPORARY TABLE S1

(SELECT R1.source\_id

FROM Routes AS R1, Routes AS R2

WHERE R1.source\_id IS NOT NULL AND R1.target\_id IS NOT NULL AND R1.target\_id=R2.source\_id AND R1.source\_id=R2.target\_id);

CREATE TEMPORARY TABLE R1

(SELECT source\_id

FROM Routes

WHERE Routes.source\_id IS NOT NULL AND Routes.target\_id IS NOT NULL AND Routes.source\_id NOT IN (SELECT source\_id

FROM S1));

SELECT DISTINCT name

FROM Airports,R1

WHERE Airports.country='United States' AND Airports.id=R1.source\_id

ORDER BY name;

+----------------------+

| name |

+----------------------+

| Alliance Municipal A |

| Amook Bay Seaplane B |

| Angoon Seaplane Base |

| Birch Creek Airport |

| …… |

| Takotna Airport |

| Tatalina Lrrs |

| Tenakee Seaplane Bas |

| Tin City LRRS Airpor |

| Toksook Bay Airport |

| Tununak Airport |

+----------------------+

**53 rows in set (0.00 sec)**

SELECT DISTINCT a.name

FROM (SELECT DISTINCT source\_id FROM routes r1 WHERE r1.source\_id IS NOT NULL AND NOT EXISTS (SELECT \* FROM routes r2 WHERE r2.source\_id=r1.target\_id AND r2.target\_id=r1.source\_id) ) s JOIN airports a ON s.source\_id=a.id

WHERE a.country= 'United States'

ORDER BY name;

SELECT DISTINCT name

FROM routes AS R1, Airports

WHERE R1.source\_id IS NOT NULL AND R1.target\_id IS NOT NULL AND Airports.country='United States' AND Airports.id=R1.source\_id AND NOT EXIST

(SELECT \*

FROM routes AS R2

WHERE R1.target\_id=R2.source\_id AND R2.target\_id=R1.source\_id)

ORDER BY name

**15. (7 points)**

Print the name of all airports in the United States for which there are departures to both an airport in San Francisco, USA and an airport in Los Angeles, USA, along with the number of departures for each.

Schema: (airport\_name, num\_SF, num\_LA)

ANSWER：

SELECT Airports.name AS airport\_name, SF\_LA.num\_SF, SF\_LA.num\_LA

FROM Airports,

(SELECT SF.source\_id, SF.num\_SF, LA.num\_LA

FROM

(SELECT source\_id, COUNT(\*) AS num\_SF

FROM Routes

WHERE target\_id IN

(SELECT id

FROM Airports

WHERE country='United States' AND city='San Francisco')

GROUP BY source\_id) SF

INNER JOIN

(SELECT source\_id, COUNT(\*) AS num\_LA

FROM Routes

WHERE target\_id IN

(SELECT id

FROM Airports

WHERE country='United States' AND city='Los Angeles')

GROUP BY source\_id) LA

ON SF.source\_id=LA.source\_id) SF\_LA

WHERE Airports.id=SF\_LA.source\_id AND country='United States';

+----------------------+--------+--------+

| airport\_name | num\_SF | num\_LA |

+----------------------+--------+--------+

| General Edward Lawre | 3 | 7 |

| Kahului | 1 | 7 |

| Kansas City Intl | 2 | 3 |

| Phoenix Sky Harbor I | 4 | 6 |

| Cleveland Hopkins In | 1 | 1 |

| Cincinnati Northern | 1 | 1 |

| Newark Liberty Intl | 2 | 2 |

| Boise Air Terminal | 1 | 1 |

| Kona Intl At Keahole | 1 | 5 |

| Ronald Reagan Washin | 2 | 4 |

| Fort Lauderdale Holl | 4 | 3 |

| Salt Lake City Intl | 2 | 7 |

| George Bush Intercon | 2 | 5 |

| Meadows Fld | 1 | 1 |

| Pittsburgh Intl | 1 | 4 |

| Miami Intl | 3 | 4 |

| Seattle Tacoma Intl | 5 | 5 |

| Indianapolis Intl | 1 | 4 |

| Lihue | 1 | 5 |

| San Antonio Intl | 1 | 3 |

| Raleigh Durham Intl | 2 | 4 |

| Tucson Intl | 1 | 6 |

| Detroit Metro Wayne | 1 | 3 |

| Dallas Fort Worth In | 5 | 6 |

| Austin Bergstrom Int | 3 | 5 |

| Lambert St Louis Int | 1 | 5 |

| Hartsfield Jackson A | 9 | 10 |

| Fresno Yosemite Intl | 1 | 4 |

| Washington Dulles In | 2 | 4 |

| General Mitchell Int | 2 | 2 |

| Portland Intl | 4 | 4 |

| Honolulu Intl | 3 | 9 |

| San Diego Intl | 3 | 5 |

| Chicago Midway Intl | 2 | 2 |

| Denver Intl | 3 | 7 |

| Philadelphia Intl | 4 | 3 |

| John F Kennedy Intl | 7 | 8 |

| Reno Tahoe Intl | 1 | 5 |

| Sacramento Intl | 1 | 7 |

| City Of Colorado Spr | 1 | 1 |

| Chicago Ohare Intl | 5 | 6 |

| Palm Springs Intl | 5 | 1 |

| Baltimore Washington | 1 | 3 |

| Minneapolis St Paul | 5 | 7 |

| Louis Armstrong New | 1 | 9 |

| Will Rogers World | 1 | 4 |

| Charlotte Douglas In | 2 | 2 |

| Mc Carran Intl | 4 | 9 |

| Orlando Intl | 2 | 6 |

| Monterey Peninsula | 1 | 4 |

| Santa Barbara Muni | 1 | 1 |

| Albuquerque Internat | 1 | 5 |

| Mahlon Sweet Fld | 1 | 3 |

| Rogue Valley Intl Me | 1 | 3 |

| Roberts Fld | 1 | 2 |

| San Luis County Regi | 1 | 1 |

+----------------------+--------+--------+

**56 rows in set (0.10 sec)**

**16. (10 points)**

Write a query that shows all non-stop, one-stop, and two-stop routes between pairs of airports, where no airport is visited more than once (e.g. you do not go from X to Y by X-->Y-->Z-->Y). Note that in routes the field “stops” says the number of stops between the source and target airport. Your query should be instance independent, i.e. you should not take into consideration the fact that 11 of the routes have value 1 (i.e. are 1 stop) and the remaining are 0 (i.e. are non-stop) -- it should work even if some of the routes are listed as 2, 3, or 4 stop. Be sure to eliminate the NULL airport.

Schema: (source\_id, num\_connections, target\_id)

Note that to generalize this to take the *transitive closure* of the routes relation (i.e. to calculate routes that are 0, 1, 2, 3, … stops until there are no more results), you would need to write a stored procedure, or use common table expressions (CTE) WITH RECURSIVE (which is not available in this version of MySQL).

ANSWER:

0,1,2 airports 1 airline: **37480 rows**

CREATE TEMPORARY TABLE DRCT

SELECT DISTINCT source\_id, stops AS num\_connections, target\_id

FROM Routes

WHERE stops IN (0,1,2);

1 airport 2 airlines: **646974 rows**

CREATE TEMPORARY TABLE ONE\_AP

SELECT DISTINCT R1.source\_id, 1 AS num\_connections, R2.target\_id

FROM Routes AS R1, Routes AS R2

WHERE R1.target\_id=R2.source\_id AND R1.stops=0 AND R2.stops=0 AND R1.source\_id<>R2.target\_id;

2 airports 3 airlines:1742422252**?**

CREATE TEMPORARY TABLE TWO\_AP\_3\_AL

SELECT DISTINCT R1.source\_id, 2 AS num\_connections, R3.target\_id

FROM Routes AS R1, Routes AS R2, Routes AS R3

WHERE R1.target\_id=R2.source\_id AND R2.target\_id=R3.source\_id AND R1.stops=0 AND R2.stops=0 AND R3.stops=0 AND R1.source\_id<>R2.target\_id AND R1.source\_id<>R3.target\_id AND R2.source\_id<>R3.target\_id;

2 airports 2 airlines: **934 rows**

CREATE TEMPORARY TABLE TWO\_AP\_2\_AL\_1

SELECT DISTINCT R1.source\_id, 2 AS num\_connections, R2.target\_id

FROM Routes AS R1, Routes AS R2

WHERE R1.target\_id=R2.source\_id AND R1.stops=0 AND R2.stops=1 AND R1.source\_id<>R2.target\_id;

2 airports 2 airlines: **441 rows**

CREATE TEMPORARY TABLE TWO\_AP\_2\_AL\_2

SELECT DISTINCT R1.source\_id, 2 AS num\_connections, R2.target\_id

FROM Routes AS R1, Routes AS R2

WHERE R1.target\_id=R2.source\_id AND R1.stops=1 AND R2.stops=0 AND R1.source\_id<>R2.target\_id;

TABLE DRCT

UNION

ONE\_AP

UNION

TWO\_AP\_3\_AL

UNION

TWO\_AP\_2\_AL\_1

UNION

TABLE TWO\_AP\_2\_AL\_2

Using something like R1.stops+R2.stops+1<=2 to test the stops