Top of Form

Bottom of Form

Top of Form

Bottom of Form

|  |  |
| --- | --- |
|  | --Q101: |
|  |  |
|  | CREATE TABLE user\_activity |
|  | ( |
|  | username VARCHAR(25), |
|  | activity VARCHAR(25), |
|  | start\_date DATE, |
|  | end\_date DATE |
|  | ); |
|  |  |
|  | INSERT INTO user\_activity VALUES('Alice', 'Travel', '2020-02-12', '2020-02-20'); |
|  | INSERT INTO user\_activity VALUES('Alice', 'Dancing', '2020-02-21', '2020-02-23'); |
|  | INSERT INTO user\_activity VALUES('Alice', 'Travel', '2020-02-24', '2020-02-28'); |
|  | INSERT INTO user\_activity VALUES('Bob', 'Travel', '2020-02-11', '2020-02-18'); |
|  |  |
|  | WITH activity\_stat\_by\_user AS( |
|  | SELECT |
|  | \*, |
|  | DENSE\_RANK() OVER(PARTITION BY username ORDER BY start\_date) AS activity\_serial, |
|  | COUNT(\*) OVER(PARTITION BY username) AS total\_activity\_count\_by\_user |
|  | FROM |
|  | user\_activity |
|  | ) |
|  | SELECT |
|  | username, |
|  | activity, |
|  | start\_date, |
|  | end\_date |
|  | FROM |
|  | activity\_stat\_by\_user |
|  | WHERE |
|  | CASE |
|  | WHEN total\_activity\_count\_by\_user = 1 |
|  | THEN 1 |
|  | WHEN activity\_serial = 2 |
|  | THEN 1 |
|  | END = 1 |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE user\_activity; |
|  |  |
|  |  |
|  | --Q102: |
|  |  |
|  | CREATE TABLE user\_activity |
|  | ( |
|  | username VARCHAR(25), |
|  | activity VARCHAR(25), |
|  | start\_date DATE, |
|  | end\_date DATE |
|  | ); |
|  |  |
|  | INSERT INTO user\_activity VALUES('Alice', 'Travel', '2020-02-12', '2020-02-20'); |
|  | INSERT INTO user\_activity VALUES('Alice', 'Dancing', '2020-02-21', '2020-02-23'); |
|  | INSERT INTO user\_activity VALUES('Alice', 'Travel', '2020-02-24', '2020-02-28'); |
|  | INSERT INTO user\_activity VALUES('Bob', 'Travel', '2020-02-11', '2020-02-18'); |
|  |  |
|  | WITH activity\_stat\_by\_user AS( |
|  | SELECT |
|  | \*, |
|  | DENSE\_RANK() OVER(PARTITION BY username ORDER BY start\_date) AS activity\_serial, |
|  | COUNT(\*) OVER(PARTITION BY username) AS total\_activity\_count\_by\_user |
|  | FROM |
|  | user\_activity |
|  | ) |
|  | SELECT |
|  | username, |
|  | activity, |
|  | start\_date, |
|  | end\_date |
|  | FROM |
|  | activity\_stat\_by\_user |
|  | WHERE |
|  | CASE |
|  | WHEN total\_activity\_count\_by\_user = 1 |
|  | THEN 1 |
|  | WHEN activity\_serial = 2 |
|  | THEN 1 |
|  | END = 1 |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE user\_activity; |
|  |  |
|  |  |
|  | --Q103: |
|  |  |
|  | CREATE TABLE students |
|  | ( |
|  | id INT, |
|  | name VARCHAR(25), |
|  | marks INT, |
|  | CONSTRAINT pk\_students PRIMARY KEY(id) |
|  | ); |
|  |  |
|  | INSERT INTO students VALUES(1, 'Ashley', 81); |
|  | INSERT INTO students VALUES(2, 'Samantha', 75); |
|  | INSERT INTO students VALUES(4, 'Julia', 76); |
|  | INSERT INTO students VALUES(3, 'Belvet', 84); |
|  |  |
|  | SELECT |
|  | name |
|  | FROM |
|  | students |
|  | WHERE |
|  | marks > 75 |
|  | ORDER BY |
|  | RIGHT(name, 3), |
|  | id |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE students; |
|  |  |
|  |  |
|  | --Q104: |
|  |  |
|  | CREATE TABLE employee |
|  | ( |
|  | employee\_id INT, |
|  | name VARCHAR(25), |
|  | months INT, |
|  | salary INT, |
|  | CONSTRAINT pk\_employee PRIMARY KEY(employee\_id) |
|  | ); |
|  |  |
|  |  |
|  | INSERT INTO employee VALUES(12228, 'Rose', 15, 1968); |
|  | INSERT INTO employee VALUES(33645, 'Angela', 1, 3443); |
|  | INSERT INTO employee VALUES(45692, 'Frank', 17, 1608); |
|  | INSERT INTO employee VALUES(56118, 'Patrick', 7, 1345); |
|  | INSERT INTO employee VALUES(59725, 'Lisa', 11, 2330); |
|  | INSERT INTO employee VALUES(74197, 'Kimberly', 16, 4372); |
|  | INSERT INTO employee VALUES(78454, 'Bonnie', 8, 1771); |
|  | INSERT INTO employee VALUES(83565, 'Michael', 6, 2017); |
|  | INSERT INTO employee VALUES(98607, 'Todd', 5, 3396); |
|  | INSERT INTO employee VALUES(99989, 'Joe', 9, 3573); |
|  |  |
|  | SELECT |
|  | name |
|  | FROM |
|  | employee |
|  | WHERE |
|  | salary > 2000 |
|  | AND months < 10 |
|  | ORDER BY |
|  | employee\_id |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE employee; |
|  |  |
|  |  |
|  | --Q105: |
|  |  |
|  | CREATE TABLE triangles |
|  | ( |
|  | a INT, |
|  | b INT, |
|  | c INT |
|  | ); |
|  |  |
|  |  |
|  | INSERT INTO triangles VALUES(20, 20, 23); |
|  | INSERT INTO triangles VALUES(20, 20, 20); |
|  | INSERT INTO triangles VALUES(20, 21, 22); |
|  | INSERT INTO triangles VALUES(13, 14, 30); |
|  |  |
|  | SELECT |
|  | a, |
|  | b, |
|  | c, |
|  | CASE |
|  | WHEN a + b <= c OR b + c <= a OR c + a <= b |
|  | THEN 'Not A Triangle' |
|  | WHEN a = b AND b = c |
|  | THEN 'Equilateral' |
|  | WHEN a = b |
|  | THEN 'Isosceles' |
|  | ELSE |
|  | 'Scalene' |
|  | END AS triangle\_type |
|  | FROM |
|  | triangles |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE triangles; |
|  |  |
|  |  |
|  | --Q106: |
|  |  |
|  | CREATE TABLE employees |
|  | ( |
|  | id INT, |
|  | name VARCHAR(25), |
|  | salary INT |
|  | ); |
|  |  |
|  | INSERT INTO employees VALUES(1, 'Kristeen', 1420); |
|  | INSERT INTO employees VALUES(2, 'Ashley', 2006); |
|  | INSERT INTO employees VALUES(3, 'Julia', 2210); |
|  | INSERT INTO employees VALUES(4, 'Maria', 3000); |
|  |  |
|  | SELECT |
|  | ROUND(AVG(salary) -AVG(CAST(REPLACE(CAST(salary AS CHAR),'0','') AS UNSIGNED)), 2) AS diff\_average |
|  | FROM |
|  | employees |
|  | WHERE |
|  | salary > 1000 |
|  | AND salary < 100000 |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE employees; |
|  |  |
|  |  |
|  | --Q107: |
|  |  |
|  | CREATE TABLE employee |
|  | ( |
|  | employee\_id INT, |
|  | name VARCHAR(25), |
|  | months INT, |
|  | salary INT |
|  | ); |
|  |  |
|  | INSERT INTO employee VALUES(12228, 'Rose', 15, 1968); |
|  | INSERT INTO employee VALUES(33645, 'Angela', 1, 3443); |
|  | INSERT INTO employee VALUES(45692, 'Frank', 17, 1608); |
|  | INSERT INTO employee VALUES(56118, 'Patrick', 7, 1345); |
|  | INSERT INTO employee VALUES(59725, 'Lisa', 11, 2330); |
|  | INSERT INTO employee VALUES(74197, 'Kimberly', 16, 4372); |
|  | INSERT INTO employee VALUES(78454, 'Bonnie', 8, 1771); |
|  | INSERT INTO employee VALUES(83565, 'Michael', 6, 2017); |
|  | INSERT INTO employee VALUES(98607, 'Todd', 5, 3396); |
|  | INSERT INTO employee VALUES(99989, 'Joe', 9, 3573); |
|  |  |
|  | WITH employee\_max\_sal\_count AS( |
|  | SELECT |
|  | months \* salary AS max\_sal, |
|  | count(\*) AS emp\_count |
|  | FROM |
|  | employee |
|  | WHERE |
|  | months \* salary = ( |
|  | SELECT |
|  | MAX(months \* salary) |
|  | FROM |
|  | employee |
|  | ) |
|  | GROUP BY |
|  | months \* salary |
|  | ) |
|  | SELECT |
|  | CONCAT(max\_sal, ' ', emp\_count) |
|  | FROM |
|  | employee\_max\_sal\_count |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE employee; |
|  |  |
|  |  |
|  | --Q108: |
|  |  |
|  | CREATE TABLE occupations |
|  | ( |
|  | name VARCHAR(25), |
|  | occupation VARCHAR(25) |
|  | ); |
|  |  |
|  | INSERT INTO occupations VALUES('Samantha', 'Doctor'); |
|  | INSERT INTO occupations VALUES('Julia', 'Actor'); |
|  | INSERT INTO occupations VALUES('Maria', 'Actor'); |
|  | INSERT INTO occupations VALUES('Meera', 'Singer'); |
|  | INSERT INTO occupations VALUES('Ashely', 'Professor'); |
|  | INSERT INTO occupations VALUES('Ketty', 'Professor'); |
|  | INSERT INTO occupations VALUES('Christeen', 'Professor'); |
|  | INSERT INTO occupations VALUES('Jane', 'Actor'); |
|  | INSERT INTO occupations VALUES('Jenny', 'Doctor'); |
|  | INSERT INTO occupations VALUES('Priya', 'Singer'); |
|  |  |
|  | SELECT |
|  | CONCAT(name, '(', LEFT(occupation,1), ')') AS name\_with\_occupation |
|  | FROM |
|  | occupations; |
|  |  |
|  | WITH occupations\_stat AS( |
|  | SELECT |
|  | occupation, |
|  | COUNT(\*) AS individual\_count |
|  | FROM |
|  | occupations |
|  | GROUP BY |
|  | occupation |
|  | ) |
|  | SELECT |
|  | CONCAT('There are a total of ', individual\_count, ' ', LOWER(occupation)) |
|  | FROM |
|  | occupations\_stat |
|  | ORDER BY |
|  | individual\_count, |
|  | occupation |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE occupations; |
|  |  |
|  |  |
|  | --Q109: |
|  |  |
|  | CREATE TABLE occupations |
|  | ( |
|  | name VARCHAR(25), |
|  | occupation VARCHAR(25) |
|  | ); |
|  |  |
|  | INSERT INTO occupations VALUES('Samantha', 'Doctor'); |
|  | INSERT INTO occupations VALUES('Julia', 'Actor'); |
|  | INSERT INTO occupations VALUES('Maria', 'Actor'); |
|  | INSERT INTO occupations VALUES('Meera', 'Singer'); |
|  | INSERT INTO occupations VALUES('Ashely', 'Professor'); |
|  | INSERT INTO occupations VALUES('Ketty', 'Professor'); |
|  | INSERT INTO occupations VALUES('Christeen', 'Professor'); |
|  | INSERT INTO occupations VALUES('Jane', 'Actor'); |
|  | INSERT INTO occupations VALUES('Jenny', 'Doctor'); |
|  | INSERT INTO occupations VALUES('Priya', 'Singer'); |
|  |  |
|  | WITH serialized\_ocp AS( |
|  | SELECT |
|  | name, |
|  | occupation, |
|  | row\_number() over(partition by occupation order by name) as serial |
|  | FROM |
|  | occupations |
|  | ) |
|  | SELECT |
|  | MAX(CASE |
|  | WHEN occupation = 'Doctor' |
|  | THEN |
|  | name |
|  | END) AS Doctor, |
|  | MAX(CASE |
|  | WHEN occupation = 'Professor' |
|  | THEN |
|  | name |
|  | END) AS Professor, |
|  | MAX(CASE |
|  | WHEN occupation = 'Singer' |
|  | THEN |
|  | name |
|  | END) AS Singer, |
|  | MAX(CASE |
|  | WHEN occupation = 'Actor' |
|  | THEN |
|  | name |
|  | END) AS Actor |
|  | FROM |
|  | serialized\_ocp |
|  | GROUP BY |
|  | serial |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE occupations; |
|  |  |
|  |  |
|  | --Q110: |
|  |  |
|  | CREATE TABLE bst |
|  | ( |
|  | n INT, |
|  | p INT |
|  | ); |
|  |  |
|  | INSERT INTO bst VALUES(1, 2); |
|  | INSERT INTO bst VALUES(3, 2); |
|  | INSERT INTO bst VALUES(6, 8); |
|  | INSERT INTO bst VALUES(9, 8); |
|  | INSERT INTO bst VALUES(2, 5); |
|  | INSERT INTO bst VALUES(8, 5); |
|  | INSERT INTO bst VALUES(5, null); |
|  |  |
|  |  |
|  | SELECT |
|  | n, |
|  | CASE |
|  | WHEN p IS NULL |
|  | THEN 'Root' |
|  | WHEN n IN ( SELECT p FROM bst) |
|  | THEN 'Inner' |
|  | ELSE |
|  | 'Leaf' |
|  | END as node\_type |
|  | FROM |
|  | bst |
|  | ORDER BY |
|  | n |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE bst; |
|  |  |
|  |  |
|  | --Q111: |
|  |  |
|  | CREATE TABLE company |
|  | ( |
|  | company\_code VARCHAR(25), |
|  | founder VARCHAR(25) |
|  | ); |
|  |  |
|  | CREATE TABLE lead\_manager |
|  | ( |
|  | lead\_manager\_code VARCHAR(25), |
|  | company\_code VARCHAR(25) |
|  | ); |
|  |  |
|  | CREATE TABLE senior\_manager |
|  | ( |
|  | senior\_manager\_code VARCHAR(25), |
|  | lead\_manager\_code VARCHAR(25), |
|  | company\_code VARCHAR(25) |
|  | ); |
|  |  |
|  | CREATE TABLE manager |
|  | ( |
|  | manager\_code VARCHAR(25), |
|  | senior\_manager\_code VARCHAR(25), |
|  | lead\_manager\_code VARCHAR(25), |
|  | company\_code VARCHAR(25) |
|  | ); |
|  |  |
|  | CREATE TABLE employee |
|  | ( |
|  | employee\_code VARCHAR(25), |
|  | manager\_code VARCHAR(25), |
|  | senior\_manager\_code VARCHAR(25), |
|  | lead\_manager\_code VARCHAR(25), |
|  | company\_code VARCHAR(25) |
|  | ); |
|  |  |
|  | INSERT INTO company VALUES('C1', 'Monika'); |
|  | INSERT INTO company VALUES('C2', 'Samantha'); |
|  |  |
|  | INSERT INTO lead\_manager VALUES('LM1', 'C1'); |
|  | INSERT INTO lead\_manager VALUES('LM2', 'C2'); |
|  |  |
|  | INSERT INTO senior\_manager VALUES('SM1', 'LM1', 'C1'); |
|  | INSERT INTO senior\_manager VALUES('SM2', 'LM1', 'C1'); |
|  | INSERT INTO senior\_manager VALUES('SM3', 'LM2', 'C2'); |
|  |  |
|  | INSERT INTO manager VALUES('M1', 'SM1', 'LM1', 'C1'); |
|  | INSERT INTO manager VALUES('M2', 'SM3', 'LM2', 'C2'); |
|  | INSERT INTO manager VALUES('M3', 'SM3', 'LM2', 'C2'); |
|  |  |
|  | INSERT INTO employee VALUES('E1', 'M1', 'SM1', 'LM1', 'C1'); |
|  | INSERT INTO employee VALUES('E2', 'M1', 'SM1', 'LM1', 'C1'); |
|  | INSERT INTO employee VALUES('E3', 'M2', 'SM3', 'LM2', 'C2'); |
|  | INSERT INTO employee VALUES('E4', 'M3', 'SM3', 'LM2', 'C2'); |
|  |  |
|  |  |
|  | SELECT |
|  | c.company\_code, |
|  | c.founder, |
|  | count(distinct lm.lead\_manager\_code) AS lead\_manager\_count, |
|  | count(distinct sm.senior\_manager\_code) AS senior\_manager\_count, |
|  | count(distinct m.manager\_code) AS manager\_count, |
|  | count(distinct e.employee\_code) AS employeee\_count |
|  | FROM |
|  | lead\_manager lm |
|  | LEFT JOIN senior\_manager sm ON lm.lead\_manager\_code = sm.lead\_manager\_code |
|  | LEFT JOIN manager m ON m.senior\_manager\_code = sm.senior\_manager\_code |
|  | LEFT JOIN employee e ON e.manager\_code = m.manager\_code |
|  | LEFT JOIN company c ON c.company\_code = lm.company\_code |
|  | GROUP BY |
|  | c.company\_code, |
|  | c.founder |
|  | ORDER BY |
|  | c.company\_code |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE company; |
|  | DROP TABLE lead\_manager; |
|  | DROP TABLE senior\_manager; |
|  | DROP TABLE manager; |
|  | DROP TABLE employee; |
|  |  |
|  |  |
|  | --Q112: |
|  |  |
|  | WITH RECURSIVE numbers AS( |
|  | SELECT 1 AS n |
|  | UNION |
|  | SELECT n+1 FROM numbers WHERE n < 1000 |
|  | ), |
|  | prime\_numbers AS( |
|  | SELECT |
|  | n1.n |
|  | FROM |
|  | numbers n1 |
|  | JOIN numbers n2 ON n1.n >= n2.n\*2 AND n2.n <> 1 |
|  | WHERE |
|  | n1.n > 3 |
|  | GROUP BY |
|  | n1.n |
|  | HAVING |
|  | MIN(MOD(n1.n,n2.n)) <> 0 |
|  | ORDER BY n1.n |
|  | ), |
|  | all\_prime AS( |
|  | SELECT 1 AS n UNION |
|  | SELECT 2 AS n UNION |
|  | SELECT 3 AS n UNION |
|  | SELECT |
|  | n |
|  | FROM |
|  | prime\_numbers |
|  | ORDER BY n |
|  | ) |
|  | SELECT |
|  | GROUP\_CONCAT(n SEPARATOR '&') |
|  | FROM |
|  | all\_prime |
|  | ; |
|  |  |
|  |  |
|  | --Q113: |
|  |  |
|  | WITH RECURSIVE numbers AS( |
|  | SELECT 1 AS n, '\*' AS star |
|  | UNION |
|  | SELECT n+1,'\*' AS star FROM numbers WHERE n < 20 |
|  | ) |
|  | SELECT |
|  | GROUP\_CONCAT(n1.star SEPARATOR '') AS stars |
|  | FROM |
|  | numbers n1 |
|  | JOIN numbers n2 ON n1.n >= n2.n |
|  | GROUP BY |
|  | n1.n |
|  | ; |
|  |  |
|  | --print in single row |
|  |  |
|  | WITH RECURSIVE numbers AS( |
|  | SELECT 1 AS n, '\*' AS star |
|  | UNION |
|  | SELECT n+1,'\*' AS star FROM numbers WHERE n < 20 |
|  | ), |
|  | multiline\_star AS( |
|  | SELECT |
|  | GROUP\_CONCAT(n1.star SEPARATOR '') AS stars |
|  | FROM |
|  | numbers n1 |
|  | JOIN numbers n2 ON n1.n >= n2.n |
|  | --ORDER BY n1.n |
|  | GROUP BY |
|  | n1.n |
|  | ) |
|  | SELECT |
|  | GROUP\_CONCAT(stars SEPARATOR '\n') AS pattern |
|  | FROM |
|  | multiline\_star |
|  | ; |
|  |  |
|  |  |
|  | --Q114: |
|  |  |
|  | WITH RECURSIVE numbers AS( |
|  | SELECT 1 AS n, '\*' AS star |
|  | UNION |
|  | SELECT n+1,'\*' AS star FROM numbers WHERE n < 20 |
|  | ) |
|  | SELECT |
|  | GROUP\_CONCAT(n1.star SEPARATOR '') AS stars |
|  | FROM |
|  | numbers n1 |
|  | JOIN numbers n2 ON n1.n <= n2.n |
|  | GROUP BY |
|  | n1.n |
|  | ; |
|  |  |
|  | --print in single row |
|  |  |
|  | WITH RECURSIVE numbers AS( |
|  | SELECT 1 AS n, '\*' AS star |
|  | UNION |
|  | SELECT n+1,'\*' AS star FROM numbers WHERE n < 20 |
|  | ), |
|  | multiline\_star AS( |
|  | SELECT |
|  | GROUP\_CONCAT(n1.star SEPARATOR '') AS stars |
|  | FROM |
|  | numbers n1 |
|  | JOIN numbers n2 ON n1.n <= n2.n |
|  | --ORDER BY n1.n |
|  | GROUP BY |
|  | n1.n |
|  | ) |
|  | SELECT |
|  | GROUP\_CONCAT(stars SEPARATOR '\n') AS pattern |
|  | FROM |
|  | multiline\_star |
|  | ; |
|  |  |
|  |  |
|  |  |
|  | --Q115: |
|  |  |
|  | CREATE TABLE students |
|  | ( |
|  | id INT, |
|  | name VARCHAR(25), |
|  | marks INT, |
|  | CONSTRAINT pk\_students PRIMARY KEY(id) |
|  | ); |
|  |  |
|  | INSERT INTO students VALUES(1, 'Ashley', 81); |
|  | INSERT INTO students VALUES(2, 'Samantha', 75); |
|  | INSERT INTO students VALUES(4, 'Julia', 76); |
|  | INSERT INTO students VALUES(3, 'Belvet', 84); |
|  |  |
|  | SELECT |
|  | name |
|  | FROM |
|  | students |
|  | WHERE |
|  | marks > 75 |
|  | ORDER BY |
|  | RIGHT(name, 3), |
|  | id |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE students; |
|  |  |
|  |  |
|  | --Q116: |
|  |  |
|  | CREATE TABLE functions |
|  | ( |
|  | x INT, |
|  | y INT |
|  | ); |
|  |  |
|  | INSERT INTO functions VALUES(20, 20); |
|  | INSERT INTO functions VALUES(20, 20); |
|  | INSERT INTO functions VALUES(20, 21); |
|  | INSERT INTO functions VALUES(23, 22); |
|  | INSERT INTO functions VALUES(22, 23); |
|  | INSERT INTO functions VALUES(21, 20); |
|  |  |
|  | WITH functions\_serialized AS( |
|  | SELECT |
|  | \*, |
|  | ROW\_NUMBER() OVER(ORDER BY x) AS serial |
|  | FROM |
|  | functions |
|  | ) |
|  | SELECT |
|  | DISTINCT f1.x, |
|  | f1.y |
|  | FROM |
|  | functions\_serialized f1 |
|  | WHERE |
|  | EXISTS( |
|  | SELECT |
|  | \* |
|  | FROM |
|  | functions\_serialized f2 |
|  | WHERE |
|  | f1.serial <> f2.serial |
|  | AND f1.x = f2.y |
|  | AND f1.y = f2.x |
|  | ) |
|  | AND f1.x <= f1.y |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE functions; |
|  |  |
|  |  |
|  | --Q116: |
|  |  |
|  | CREATE TABLE employee |
|  | ( |
|  | employee\_id INT, |
|  | name VARCHAR(25), |
|  | months INT, |
|  | salary INT, |
|  | CONSTRAINT pk\_employee PRIMARY KEY(employee\_id) |
|  | ); |
|  |  |
|  |  |
|  | INSERT INTO employee VALUES(12228, 'Rose', 15, 1968); |
|  | INSERT INTO employee VALUES(33645, 'Angela', 1, 3443); |
|  | INSERT INTO employee VALUES(45692, 'Frank', 17, 1608); |
|  | INSERT INTO employee VALUES(56118, 'Patrick', 7, 1345); |
|  | INSERT INTO employee VALUES(59725, 'Lisa', 11, 2330); |
|  | INSERT INTO employee VALUES(74197, 'Kimberly', 16, 4372); |
|  | INSERT INTO employee VALUES(78454, 'Bonnie', 8, 1771); |
|  | INSERT INTO employee VALUES(83565, 'Michael', 6, 2017); |
|  | INSERT INTO employee VALUES(98607, 'Todd', 5, 3396); |
|  | INSERT INTO employee VALUES(99989, 'Joe', 9, 3573); |
|  |  |
|  | SELECT |
|  | name |
|  | FROM |
|  | employee |
|  | ORDER BY |
|  | name |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE employee; |
|  |  |
|  |  |
|  | --Q117: |
|  |  |
|  | CREATE TABLE employee |
|  | ( |
|  | employee\_id INT, |
|  | name VARCHAR(25), |
|  | months INT, |
|  | salary INT, |
|  | CONSTRAINT pk\_employee PRIMARY KEY(employee\_id) |
|  | ); |
|  |  |
|  |  |
|  | INSERT INTO employee VALUES(12228, 'Rose', 15, 1968); |
|  | INSERT INTO employee VALUES(33645, 'Angela', 1, 3443); |
|  | INSERT INTO employee VALUES(45692, 'Frank', 17, 1608); |
|  | INSERT INTO employee VALUES(56118, 'Patrick', 7, 1345); |
|  | INSERT INTO employee VALUES(59725, 'Lisa', 11, 2330); |
|  | INSERT INTO employee VALUES(74197, 'Kimberly', 16, 4372); |
|  | INSERT INTO employee VALUES(78454, 'Bonnie', 8, 1771); |
|  | INSERT INTO employee VALUES(83565, 'Michael', 6, 2017); |
|  | INSERT INTO employee VALUES(98607, 'Todd', 5, 3396); |
|  | INSERT INTO employee VALUES(99989, 'Joe', 9, 3573); |
|  |  |
|  | SELECT |
|  | name |
|  | FROM |
|  | employee |
|  | WHERE |
|  | salary > 2000 |
|  | AND months < 10 |
|  | ORDER BY |
|  | employee\_id |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE employee; |
|  |  |
|  |  |
|  | --Q118: |
|  |  |
|  | CREATE TABLE triangles |
|  | ( |
|  | a INT, |
|  | b INT, |
|  | c INT |
|  | ); |
|  |  |
|  |  |
|  | INSERT INTO triangles VALUES(20, 20, 23); |
|  | INSERT INTO triangles VALUES(20, 20, 20); |
|  | INSERT INTO triangles VALUES(20, 21, 22); |
|  | INSERT INTO triangles VALUES(13, 14, 30); |
|  |  |
|  | SELECT |
|  | a, |
|  | b, |
|  | c, |
|  | CASE |
|  | WHEN a + b <= c OR b + c <= a OR c + a <= b |
|  | THEN 'Not A Triangle' |
|  | WHEN a = b AND b = c |
|  | THEN 'Equilateral' |
|  | WHEN a = b |
|  | THEN 'Isosceles' |
|  | ELSE |
|  | 'Scalene' |
|  | END AS triangle\_type |
|  | FROM |
|  | triangles |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE triangles; |
|  |  |
|  |  |
|  | --Q119: |
|  |  |
|  | CREATE TABLE user\_transactions |
|  | ( |
|  | transaction\_id INT, |
|  | product\_id INT, |
|  | spend DECIMAL(10,2), |
|  | transaction\_date DATE |
|  | ); |
|  |  |
|  |  |
|  | INSERT INTO user\_transactions VALUES(1341, 123424, 1500.60, '2019-12-31'); |
|  | INSERT INTO user\_transactions VALUES(1423, 123424, 1000.20, '2020-12-31'); |
|  | INSERT INTO user\_transactions VALUES(1623, 123424, 1246.44, '2021-12-31'); |
|  | INSERT INTO user\_transactions VALUES(1322, 123424, 2145.32, '2022-12-31'); |
|  |  |
|  |  |
|  | SELECT |
|  | DATE\_FORMAT(transaction\_date,'%Y') AS year, |
|  | product\_id, |
|  | spend AS curr\_year\_spend, |
|  | LAG(spend) OVER(ORDER BY DATE\_FORMAT(transaction\_date,'%Y')) AS prev\_year\_spend, |
|  | ROUND((spend - LAG(spend) OVER(ORDER BY DATE\_FORMAT(transaction\_date,'%Y'))) |
|  | \*100.00/LAG(spend) OVER(ORDER BY DATE\_FORMAT(transaction\_date,'%Y')),2) AS yoy\_rate |
|  | FROM |
|  | user\_transactions |
|  | ORDER BY |
|  | year |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE user\_transactions; |
|  |  |
|  |  |
|  | --Q120: |
|  |  |
|  | CREATE TABLE inventory |
|  | ( |
|  | item\_id INT, |
|  | item\_type VARCHAR(20), |
|  | item\_category VARCHAR(20), |
|  | square\_footage DECIMAL(10,2) |
|  | ); |
|  |  |
|  | INSERT INTO inventory VALUES(1374, 'prime\_eligible' , 'mini refrigerator', 68.00); |
|  | INSERT INTO inventory VALUES(4245, 'not\_prime', 'standing lamp', 26.40); |
|  | INSERT INTO inventory VALUES(2452, 'prime\_eligible', 'television', 85.00); |
|  | INSERT INTO inventory VALUES(3255, 'not\_prime', 'side table', 22.60); |
|  | INSERT INTO inventory VALUES(1672, 'prime\_eligible', 'laptop', 8.50); |
|  |  |
|  | WITH product\_inventory\_summary AS |
|  | ( |
|  | SELECT |
|  | item\_type, |
|  | SUM(square\_footage) as square\_footage\_required, |
|  | COUNT(item\_id) as unique\_item\_count, |
|  | 500000 as total\_space, |
|  | FLOOR(500000/sum(square\_footage))\*sum(square\_footage) as space\_used, |
|  | FLOOR(500000/sum(square\_footage))\*COUNT(item\_id) as item\_count |
|  | FROM |
|  | inventory |
|  | GROUP BY |
|  | item\_type |
|  | ) |
|  | SELECT |
|  | t1.item\_type, |
|  | CASE |
|  | WHEN t1.item\_type = 'prime\_eligible' |
|  | THEN t1.item\_count |
|  | ELSE |
|  | FLOOR((500000-t2.space\_used)/t1.square\_footage\_required)\*t1.unique\_item\_count |
|  | END AS item\_count |
|  | FROM |
|  | product\_inventory\_summary t1 |
|  | JOIN product\_inventory\_summary t2 ON t1.item\_type <> t2.item\_type |
|  | ORDER BY t1.item\_type DESC |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE inventory; |
|  |  |
|  |  |
|  | --Q121: |
|  |  |
|  | CREATE TABLE user\_actions |
|  | ( |
|  | user\_id INT, |
|  | event\_id INT, |
|  | event\_type VARCHAR(20), |
|  | event\_date DATE |
|  | ); |
|  |  |
|  | INSERT INTO user\_actions VALUES(445, 7765 , 'sign-in', '2022-05-31'); |
|  | INSERT INTO user\_actions VALUES(742, 6458, 'sign-in', '2022-06-03'); |
|  | INSERT INTO user\_actions VALUES(445, 3634, 'like', '2022-06-05'); |
|  | INSERT INTO user\_actions VALUES(742, 1374, 'comment', '2022-06-05'); |
|  | INSERT INTO user\_actions VALUES(648, 3124, 'like', '2022-06-18'); |
|  |  |
|  |  |
|  | SELECT |
|  | CAST(DATE\_FORMAT(curr\_month\_ua.event\_date, '%m') AS UNSIGNED) AS month, |
|  | count(distinct curr\_month\_ua.user\_id) AS monthly\_active\_users |
|  | FROM |
|  | user\_actions curr\_month\_ua |
|  | WHERE |
|  | curr\_month\_ua.event\_type IN ('sign-in', 'like', 'comment') |
|  | AND DATE\_FORMAT(curr\_month\_ua.event\_date,'%Y-%m') = '2022-06' |
|  | AND EXISTS( |
|  | SELECT |
|  | \* |
|  | FROM |
|  | user\_actions last\_month\_ua |
|  | WHERE |
|  | curr\_month\_ua.user\_id = last\_month\_ua.user\_id |
|  | AND last\_month\_ua.event\_type IN ('sign-in', 'like', 'comment') |
|  | AND DATE\_FORMAT(curr\_month\_ua.event\_date, '%Y-%m') = |
|  | DATE\_FORMAT(last\_month\_ua.event\_date + INTERVAL '1' MONTH, '%Y-%m') |
|  | ) |
|  | GROUP BY |
|  | CAST(DATE\_FORMAT(curr\_month\_ua.event\_date, '%m') AS UNSIGNED); |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE user\_actions; |
|  |  |
|  |  |
|  | --Q122: |
|  |  |
|  | CREATE TABLE search\_frequency |
|  | ( |
|  | searches INT, |
|  | num\_users INT |
|  | ); |
|  |  |
|  | INSERT INTO search\_frequency VALUES(1, 2); |
|  | INSERT INTO search\_frequency VALUES(2, 2); |
|  | INSERT INTO search\_frequency VALUES(3, 3); |
|  | INSERT INTO search\_frequency VALUES(4, 1); |
|  |  |
|  | WITH cumulative\_sum AS |
|  | ( |
|  | SELECT |
|  | \*, |
|  | SUM(num\_users) OVER(ORDER BY searches) as cum\_sum, |
|  | ROW\_NUMBER() OVER(ORDER BY searches) as row\_num |
|  | FROM |
|  | search\_frequency |
|  | ), |
|  | max\_cumulative\_sum AS |
|  | ( |
|  | SELECT |
|  | MAX(cum\_sum) as max\_cum\_sum |
|  | FROM |
|  | cumulative\_sum |
|  | ), |
|  | odd\_even\_sum AS |
|  | ( |
|  | SELECT |
|  | MIN(row\_num) AS row1 |
|  | FROM |
|  | cumulative\_sum |
|  | WHERE |
|  | cum\_sum >= ( |
|  | SELECT |
|  | CEILING((max\_cum\_sum)\*0.5) |
|  | FROM |
|  | max\_cumulative\_sum |
|  | ) |
|  | ), |
|  | even\_sum AS |
|  | ( |
|  | SELECT |
|  | MIN(row\_num) AS row2 |
|  | FROM |
|  | cumulative\_sum |
|  | WHERE |
|  | cum\_sum >= ( |
|  | SELECT |
|  | CEILING((max\_cum\_sum)\*0.5)+1 |
|  | FROM |
|  | max\_cumulative\_sum |
|  | ) |
|  | ) |
|  | SELECT |
|  | ROUND( |
|  | CASE |
|  | WHEN (SELECT max\_cum\_sum FROM max\_cumulative\_sum) % 2 = 0 |
|  | THEN ( |
|  | (SELECT searches FROM cumulative\_sum WHERE row\_num = (SELECT row1 FROM odd\_even\_sum)) |
|  | + |
|  | (SELECT searches FROM cumulative\_sum WHERE row\_num = (SELECT row2 FROM even\_sum)) |
|  | )/2.0 |
|  | ELSE (SELECT searches FROM cumulative\_sum WHERE row\_num = (SELECT row1 FROM odd\_even\_sum)) |
|  | END |
|  | ,1) as median |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE search\_frequency; |
|  |  |
|  |  |
|  | --Q123: |
|  |  |
|  | CREATE TABLE advertiser |
|  | ( |
|  | user\_id VARCHAR(15), |
|  | status VARCHAR(15) |
|  | ); |
|  |  |
|  | CREATE TABLE daily\_pay |
|  | ( |
|  | user\_id VARCHAR(15), |
|  | paid DECIMAL(10,2) |
|  | ); |
|  |  |
|  | INSERT INTO advertiser VALUES('bing', 'NEW'); |
|  | INSERT INTO advertiser VALUES('yahoo', 'NEW'); |
|  | INSERT INTO advertiser VALUES('alibaba', 'EXISTING'); |
|  |  |
|  | --resuccernt test |
|  | INSERT INTO advertiser VALUES('oracle', 'CHURN'); |
|  |  |
|  | INSERT INTO daily\_pay VALUES('yahoo', '45.00'); |
|  | INSERT INTO daily\_pay VALUES('alibaba', '100.00'); |
|  | INSERT INTO daily\_pay VALUES('target', '13.00'); |
|  |  |
|  | --resuccernt test |
|  | INSERT INTO daily\_pay VALUES('oracle', '13.00'); |
|  |  |
|  |  |
|  |  |
|  | WITH full\_outer\_join\_table AS( |
|  | SELECT |
|  | dp.user\_id AS user\_id\_dp, |
|  | dp.paid, |
|  | a.user\_id AS user\_id\_advertiser, |
|  | a.status |
|  | FROM daily\_pay dp |
|  | LEFT OUTER JOIN advertiser a ON dp.user\_id = a.user\_id |
|  | UNION |
|  | SELECT |
|  | dp.user\_id AS user\_id\_dp, |
|  | dp.paid, |
|  | a.user\_id AS user\_id\_advertiser, |
|  | a.status |
|  | FROM daily\_pay dp |
|  | RIGHT OUTER JOIN advertiser a ON dp.user\_id = a.user\_id |
|  | ) |
|  | SELECT |
|  | CASE |
|  | WHEN user\_id\_dp IS NULL |
|  | THEN user\_id\_advertiser |
|  | ELSE user\_id\_dp |
|  | END AS user\_id, |
|  | CASE |
|  | WHEN user\_id\_dp is NULL |
|  | THEN |
|  | 'CHURN' |
|  | ELSE |
|  | CASE |
|  | WHEN status is null |
|  | THEN 'NEW' |
|  | WHEN status = 'CHURN' |
|  | THEN 'RESURRECT' |
|  | ELSE 'EXISTING' |
|  | END |
|  | END AS new\_status |
|  | FROM full\_outer\_join\_table full\_table |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE advertiser; |
|  | DROP TABLE daily\_pay; |
|  |  |
|  |  |
|  | --Q124: |
|  |  |
|  | CREATE TABLE server\_utilization |
|  | ( |
|  | server\_id INT, |
|  | status\_time TIMESTAMP, |
|  | session\_status VARCHAR(10) |
|  | ); |
|  |  |
|  | INSERT INTO server\_utilization VALUES(1, '2022-08-02 10:00:00', 'start'); |
|  | INSERT INTO server\_utilization VALUES(1, '2022-08-04 10:00:00', 'stop'); |
|  | INSERT INTO server\_utilization VALUES(2, '2022-08-17 10:00:00', 'start'); |
|  | INSERT INTO server\_utilization VALUES(2, '2022-08-24 10:00:00', 'stop'); |
|  |  |
|  | WITH up\_time\_by\_server AS |
|  | ( |
|  | SELECT |
|  | server\_id, |
|  | session\_status, |
|  | status\_time, |
|  | CASE |
|  | WHEN session\_status = 'stop' |
|  | THEN |
|  | TIMESTAMPDIFF(SECOND, status\_time,LAG(status\_time) OVER(PARTITION BY server\_id ORDER BY status\_time))/3600 |
|  | END as up\_time |
|  | FROM server\_utilization |
|  | ) |
|  | SELECT |
|  | ROUND(sum(up\_time)/24) |
|  | FROM |
|  | up\_time\_by\_server |
|  | WHERE |
|  | up\_time is not null |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE server\_utilization; |
|  |  |
|  |  |
|  | --Q125: |
|  |  |
|  | CREATE TABLE transactions |
|  | ( |
|  | transaction\_id INT, |
|  | merchant\_id INT, |
|  | credit\_card\_id INT, |
|  | amount INT, |
|  | transaction\_timestamp TIMESTAMP |
|  | ); |
|  |  |
|  | INSERT INTO transactions VALUES(1, 101, 1, 100, '2022-09-25 12:00:00'); |
|  | INSERT INTO transactions VALUES(2, 101, 1, 100, '2022-09-25 12:08:00'); |
|  | INSERT INTO transactions VALUES(3, 101, 1, 100, '2022-09-25 12:28:00'); |
|  | INSERT INTO transactions VALUES(4, 102, 2, 300, '2022-09-25 12:00:00'); |
|  | INSERT INTO transactions VALUES(5, 102, 2, 400, '2022-09-25 14:00:00'); |
|  |  |
|  | WITH trx\_with\_repeadted AS |
|  | ( |
|  | SELECT |
|  | credit\_card\_id, |
|  | amount, |
|  | transaction\_timestamp, |
|  | count(\*) OVER( |
|  | PARTITION BY credit\_card\_id,amount |
|  | ORDER BY transaction\_timestamp |
|  | RANGE BETWEEN INTERVAL '10' MINUTE PRECEDING AND CURRENT ROW |
|  | ) AS moving\_count |
|  | FROM |
|  | transactions |
|  | ) |
|  | SELECT |
|  | COUNT(\*) as payment\_count |
|  | FROM trx\_with\_repeadted |
|  | WHERE |
|  | moving\_count > 1 |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE transactions; |
|  |  |
|  |  |
|  | --Q126: |
|  |  |
|  | CREATE TABLE orders |
|  | ( |
|  | order\_id INT, |
|  | customer\_id INT, |
|  | trip\_id INT, |
|  | status VARCHAR(30), |
|  | order\_timestamp TIMESTAMP |
|  | ); |
|  |  |
|  | CREATE TABLE trips |
|  | ( |
|  | dasher\_id INT, |
|  | trip\_id INT, |
|  | estimated\_delivery\_timestamp TIMESTAMP, |
|  | actual\_delivery\_timestamp TIMESTAMP |
|  | ); |
|  |  |
|  | CREATE TABLE customers |
|  | ( |
|  | customer\_id INT, |
|  | signup\_timestamp TIMESTAMP |
|  | ); |
|  |  |
|  | INSERT INTO orders VALUES(727424,8472, 100463, 'completed successfully', '2022-06-05 09:12:00'); |
|  | INSERT INTO orders VALUES(242513, 2341, 100482, 'completed incorrectly', '2022-06-05 14:40:00'); |
|  | INSERT INTO orders VALUES(141367, 1314, 100362, 'completed incorrectly', '2022-06-07 15:03:00'); |
|  | INSERT INTO orders VALUES(582193, 5421, 100657, 'never\_received', '2022-07-07 15:22:00'); |
|  | INSERT INTO orders VALUES(253613, 1314, 100213, 'completed successfully', '2022-06-12 13:43:00'); |
|  |  |
|  | INSERT INTO trips VALUES(101, 100463, '2022-06-05 09:42:00', '2022-06-05 09:38:00'); |
|  | INSERT INTO trips VALUES(102, 100482, '2022-06-05 15:10:00', '2022-06-05 15:46:00'); |
|  | INSERT INTO trips VALUES(101, 100362, '2022-06-07 15:33:00', '2022-06-07 16:45:00'); |
|  | INSERT INTO trips VALUES(102, 100657, '2022-07-07 15:52:00',null); |
|  | INSERT INTO trips VALUES(103, 100213, '2022-06-12 14:13:00', '2022-06-12 14:10:00'); |
|  |  |
|  | INSERT INTO customers VALUES(8472, '2022-05-30 00:00:00'); |
|  | INSERT INTO customers VALUES(2341, '2022-06-01 00:00:00'); |
|  | INSERT INTO customers VALUES(1314, '2022-06-03 00:00:00'); |
|  | INSERT INTO customers VALUES(1435, '2022-06-05 00:00:00'); |
|  | INSERT INTO customers VALUES(5421, '2022-06-07 00:00:00'); |
|  |  |
|  | SELECT |
|  | ROUND((COUNT( |
|  | CASE |
|  | WHEN lower(o.status) <> 'completed successfully' |
|  | THEN o.order\_id |
|  | END |
|  | )\*100.00/COUNT(o.order\_id)),2) as bad\_experience\_pct |
|  |  |
|  | FROM orders o |
|  | JOIN customers c ON c.customer\_id = o.customer\_id |
|  | WHERE |
|  | TIMESTAMPDIFF(DAY,o.order\_timestamp,c.signup\_timestamp) < 14 |
|  | AND DATE\_FORMAT(c.signup\_timestamp,'%Y-%m') = '2022-06' |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE customers; |
|  | DROP TABLE trips; |
|  | DROP TABLE orders; |
|  |  |
|  | --Q127: |
|  |  |
|  | CREATE TABLE scores |
|  | ( |
|  | player\_name VARCHAR(25), |
|  | gender VARCHAR(1), |
|  | day DATE, |
|  | score\_points INT, |
|  | CONSTRAINT pk\_scores PRIMARY KEY (gender, day) |
|  | ); |
|  |  |
|  | INSERT INTO scores VALUES('Aron', 'F', '2020-01-01', 17); |
|  | INSERT INTO scores VALUES('Alice', 'F', '2020-01-07', 23); |
|  | INSERT INTO scores VALUES('Bajrang', 'M', '2020-01-07', 7); |
|  | INSERT INTO scores VALUES('Khali' , 'M', '2019-12-25', 11); |
|  | INSERT INTO scores VALUES('Slaman', 'M', '2019-12-30', 13); |
|  | INSERT INTO scores VALUES('Joe', 'M', '2019-12-31', 3); |
|  | INSERT INTO scores VALUES('Jose', 'M', '2019-12-18', 2); |
|  | INSERT INTO scores VALUES('Priya', 'F', '2019-12-31', 23); |
|  | INSERT INTO scores VALUES('Priyanka', 'F', '2019-12-30', 17); |
|  |  |
|  | SELECT |
|  | gender, |
|  | day, |
|  | sum(score\_points) OVER(PARTITION BY gender ORDER BY day) AS total |
|  | FROM |
|  | scores; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE scores; |
|  |  |
|  |  |
|  | --Q128: |
|  |  |
|  | CREATE TABLE person |
|  | ( |
|  | id INT, |
|  | name VARCHAR(25), |
|  | phone\_number VARCHAR(11), |
|  | CONSTRAINT pk\_person PRIMARY KEY (id) |
|  | ); |
|  |  |
|  | CREATE TABLE country |
|  | ( |
|  | name VARCHAR(25), |
|  | country\_code VARCHAR(3), |
|  | CONSTRAINT pk\_country PRIMARY KEY (country\_code) |
|  | ); |
|  |  |
|  | CREATE TABLE calls |
|  | ( |
|  | caller\_id INT, |
|  | callee\_id INT, |
|  | duration INT |
|  | ); |
|  |  |
|  | INSERT INTO person VALUES(3, 'Jonathan', '051-1234567'); |
|  | INSERT INTO person VALUES(12, 'Elvis', '051-7654321'); |
|  | INSERT INTO person VALUES(1, 'Moncef', '212-1234567'); |
|  | INSERT INTO person VALUES(2, 'Maroua', '212-6523651'); |
|  | INSERT INTO person VALUES(7, 'Meir', '972-1234567'); |
|  | INSERT INTO person VALUES(9, 'Rachel', '972-0011100'); |
|  |  |
|  | INSERT INTO country VALUES('Peru', '51'); |
|  | INSERT INTO country VALUES('Israel', '972'); |
|  | INSERT INTO country VALUES('Morocco', '212'); |
|  | INSERT INTO country VALUES('Germany', '49'); |
|  | INSERT INTO country VALUES('Ethiopia', '251'); |
|  |  |
|  | INSERT INTO calls VALUES(1, 9, 33); |
|  | INSERT INTO calls VALUES(2, 9, 4); |
|  | INSERT INTO calls VALUES(1, 2, 59); |
|  | INSERT INTO calls VALUES(3, 12, 102); |
|  | INSERT INTO calls VALUES(3, 12, 330); |
|  | INSERT INTO calls VALUES(12, 3, 5); |
|  | INSERT INTO calls VALUES(7, 9, 13); |
|  | INSERT INTO calls VALUES(7, 1, 3); |
|  | INSERT INTO calls VALUES(9, 7, 1); |
|  | INSERT INTO calls VALUES(1, 7, 7); |
|  |  |
|  | WITH receiver\_caller\_calls AS( |
|  | SELECT |
|  | caller\_id AS caller\_receiver\_id, |
|  | duration |
|  | FROM |
|  | calls |
|  | UNION ALL |
|  | SELECT |
|  | callee\_id AS caller\_receiver\_id, |
|  | duration |
|  | FROM |
|  | calls |
|  | ), |
|  | call\_duration\_avg AS( |
|  | SELECT |
|  | DISTINCT cn.name, |
|  | avg(c.duration) OVER() as global\_average, |
|  | avg(c.duration) OVER(PARTITION BY cn.name) as country\_average |
|  | FROM |
|  | person p |
|  | JOIN country cn |
|  | ON CAST(SUBSTRING\_INDEX(p.phone\_number, '-', 1) AS UNSIGNED) = CAST(cn.country\_code AS UNSIGNED) |
|  | JOIN receiver\_caller\_calls c |
|  | ON c.caller\_receiver\_id = p.id |
|  | ) |
|  | SELECT |
|  | name |
|  | FROM |
|  | call\_duration\_avg |
|  | WHERE |
|  | country\_average > global\_average; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE calls; |
|  | DROP TABLE country; |
|  | DROP TABLE person; |
|  |  |
|  |  |
|  | --Q129: |
|  |  |
|  | CREATE TABLE numbers |
|  | ( |
|  | num INT, |
|  | frequency INT |
|  | ); |
|  |  |
|  | INSERT INTO numbers VALUES(0, 7); |
|  | INSERT INTO numbers VALUES(1, 1); |
|  | INSERT INTO numbers VALUES(2, 3); |
|  | INSERT INTO numbers VALUES(3, 1); |
|  |  |
|  | WITH cumulative\_sum AS |
|  | ( |
|  | SELECT |
|  | \*, |
|  | SUM(frequency) OVER(ORDER BY num) as cum\_sum, |
|  | ROW\_NUMBER() OVER(ORDER BY num) as row\_num |
|  | FROM |
|  | numbers |
|  | ), |
|  | max\_cumulative\_sum AS |
|  | ( |
|  | SELECT |
|  | MAX(cum\_sum) as max\_cum\_sum |
|  | FROM |
|  | cumulative\_sum |
|  | ), |
|  | odd\_even\_sum AS |
|  | ( |
|  | SELECT |
|  | MIN(row\_num) AS row1 |
|  | FROM |
|  | cumulative\_sum |
|  | WHERE |
|  | cum\_sum >= ( |
|  | SELECT |
|  | CEILING((max\_cum\_sum)\*0.5) |
|  | FROM |
|  | max\_cumulative\_sum |
|  | ) |
|  | ), |
|  | even\_sum AS |
|  | ( |
|  | SELECT |
|  | MIN(row\_num) AS row2 |
|  | FROM |
|  | cumulative\_sum |
|  | WHERE |
|  | cum\_sum >= ( |
|  | SELECT |
|  | CEILING((max\_cum\_sum)\*0.5)+1 |
|  | FROM |
|  | max\_cumulative\_sum |
|  | ) |
|  | ) |
|  | SELECT |
|  | ROUND( |
|  | CASE |
|  | WHEN (SELECT max\_cum\_sum FROM max\_cumulative\_sum) % 2 = 0 |
|  | THEN ( |
|  | (SELECT num FROM cumulative\_sum WHERE row\_num = (SELECT row1 FROM odd\_even\_sum)) |
|  | + |
|  | (SELECT num FROM cumulative\_sum WHERE row\_num = (SELECT row2 FROM even\_sum )) |
|  | )/2.0 |
|  | ELSE( |
|  | SELECT num FROM cumulative\_sum WHERE row\_num = (SELECT row1 FROM odd\_even\_sum) |
|  | ) |
|  | END |
|  | ,1) as median |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE numbers; |
|  |  |
|  |  |
|  | --Q130: |
|  |  |
|  | CREATE TABLE employee |
|  | ( |
|  | employee\_id INT, |
|  | department\_id INT, |
|  | CONSTRAINT pk\_employee PRIMARY KEY(employee\_id) |
|  | ); |
|  |  |
|  | CREATE TABLE salary |
|  | ( |
|  | id INT, |
|  | employee\_id INT, |
|  | amount INT, |
|  | pay\_date DATE, |
|  | CONSTRAINT pk\_salary PRIMARY KEY(id), |
|  | CONSTRAINT fk\_employee FOREIGN KEY(employee\_id) |
|  | REFERENCES employee(employee\_id) |
|  | ); |
|  |  |
|  | INSERT INTO employee VALUES(1, 1); |
|  | INSERT INTO employee VALUES(2, 2); |
|  | INSERT INTO employee VALUES(3, 2); |
|  |  |
|  | INSERT INTO salary VALUES(1, 1, 9000, '2017-03-31'); |
|  | INSERT INTO salary VALUES(2, 2, 6000, '2017-03-31'); |
|  | INSERT INTO salary VALUES(3, 3, 10000, '2017-03-31'); |
|  | INSERT INTO salary VALUES(4, 1, 7000, '2017-02-28'); |
|  | INSERT INTO salary VALUES(5, 2, 6000, '2017-02-28'); |
|  | INSERT INTO salary VALUES(6, 3, 8000, '2017-02-28'); |
|  |  |
|  | WITH department\_company\_avg\_monthly AS( |
|  | SELECT |
|  | DISTINCT DATE\_FORMAT(s.pay\_date, '%Y-%m') AS pay\_month, |
|  | department\_id, |
|  | AVG(amount) OVER(PARTITION BY DATE\_FORMAT(s.pay\_date, '%Y-%m')) as company\_avg, |
|  | AVG(amount) OVER(PARTITION BY DATE\_FORMAT(s.pay\_date, '%Y-%m'), department\_id) as department\_avg |
|  | FROM |
|  | salary s |
|  | JOIN employee e ON s.employee\_id = e.employee\_id |
|  | ) |
|  | SELECT |
|  | pay\_month, |
|  | department\_id, |
|  | CASE |
|  | WHEN department\_avg > company\_avg |
|  | THEN 'higher' |
|  | WHEN department\_avg < company\_avg |
|  | THEN 'lower' |
|  | ELSE |
|  | 'same' |
|  | END AS comparison |
|  | FROM |
|  | department\_company\_avg\_monthly |
|  | ORDER BY |
|  | department\_id; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE salary; |
|  | DROP TABLE employee; |
|  |  |
|  |  |
|  | --Q131: |
|  |  |
|  | CREATE TABLE activity |
|  | ( |
|  | player\_id INT, |
|  | device\_id INT, |
|  | event\_date DATE, |
|  | games\_played INT, |
|  | CONSTRAINT pk\_activity PRIMARY KEY(player\_id, event\_date) |
|  | ); |
|  |  |
|  | INSERT INTO activity VALUES(1, 2, '2016-03-01', 5); |
|  | INSERT INTO activity VALUES(1, 2, '2016-03-02', 6); |
|  | INSERT INTO activity VALUES(2, 3, '2017-06-25', 1); |
|  | INSERT INTO activity VALUES(3, 1, '2016-03-01', 0); |
|  | INSERT INTO activity VALUES(3, 4, '2016-07-03', 5); |
|  |  |
|  | WITH retention\_data AS( |
|  | SELECT |
|  | curr\_day.event\_date, |
|  | COUNT(DISTINCT curr\_day.player\_id) AS retention\_player\_count |
|  | FROM |
|  | activity curr\_day |
|  | WHERE |
|  | EXISTS( |
|  | SELECT |
|  | \* |
|  | FROM |
|  | activity next\_day |
|  | WHERE |
|  | curr\_day.player\_id = next\_day.player\_id |
|  | AND next\_day.event\_date = curr\_day.event\_date + INTERVAL '1' DAY |
|  | ) |
|  | GROUP BY |
|  | curr\_day.event\_date |
|  | ), |
|  | player\_signup\_data AS( |
|  | SELECT |
|  | DISTINCT player\_id, |
|  | FIRST\_VALUE(event\_date) OVER(PARTITION BY player\_id ORDER BY event\_date) as signup\_date |
|  | FROM |
|  | activity |
|  | ), |
|  | daily\_player\_data AS( |
|  | SELECT |
|  | signup\_date, |
|  | COUNT(DISTINCT player\_id) AS daily\_player\_count |
|  | FROM |
|  | player\_signup\_data |
|  | GROUP BY |
|  | signup\_date |
|  | ) |
|  | SELECT |
|  | dpd.signup\_date AS install\_dt, |
|  | dpd.daily\_player\_count AS installs, |
|  | ROUND(IFNULL(rd.retention\_player\_count,0)/dpd.daily\_player\_count,2) AS Day1\_retention |
|  |  |
|  | FROM |
|  | daily\_player\_data dpd |
|  | LEFT JOIN retention\_data rd ON dpd.signup\_date = rd.event\_date |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE activity; |
|  |  |
|  |  |
|  | --Q132: |
|  |  |
|  | CREATE TABLE players |
|  | ( |
|  | player\_id INT, |
|  | group\_id INT, |
|  | CONSTRAINT pk\_players PRIMARY KEY(player\_id) |
|  | ); |
|  |  |
|  | CREATE TABLE matches |
|  | ( |
|  | match\_id INT, |
|  | first\_player INT, |
|  | second\_player INT, |
|  | first\_score INT, |
|  | second\_score INT, |
|  | CONSTRAINT pk\_matches PRIMARY KEY(match\_id) |
|  | ); |
|  |  |
|  | INSERT INTO players VALUES(15, 1); |
|  | INSERT INTO players VALUES(25, 1); |
|  | INSERT INTO players VALUES(30, 1); |
|  | INSERT INTO players VALUES(45, 1); |
|  | INSERT INTO players VALUES(10, 2); |
|  | INSERT INTO players VALUES(35, 2); |
|  | INSERT INTO players VALUES(50, 2); |
|  | INSERT INTO players VALUES(20, 3); |
|  | INSERT INTO players VALUES(40, 3); |
|  |  |
|  | INSERT INTO matches VALUES(1, 15, 45, 3, 0); |
|  | INSERT INTO matches VALUES(2, 30, 25, 1, 2); |
|  | INSERT INTO matches VALUES(3, 30, 15, 2, 0); |
|  | INSERT INTO matches VALUES(4, 40, 20, 5, 2); |
|  | INSERT INTO matches VALUES(5, 35, 50, 1, 1); |
|  |  |
|  | WITH player\_score AS( |
|  | SELECT |
|  | p.group\_id, |
|  | p.player\_id, |
|  | SUM(CASE |
|  | WHEN p.player\_id = m.first\_player |
|  | THEN m.first\_score |
|  | WHEN p.player\_id = m.second\_player |
|  | THEN m.second\_score |
|  | END) AS score |
|  | FROM |
|  | players p |
|  | JOIN matches m ON p.player\_id = m.first\_player OR p.player\_id = m.second\_player |
|  | GROUP BY |
|  | p.group\_id, |
|  | p.player\_id |
|  | ), |
|  | ranked\_player AS( |
|  | SELECT |
|  | group\_id, |
|  | player\_id, |
|  | score, |
|  | DENSE\_RANK() OVER (PARTITION BY group\_id ORDER BY score DESC,player\_id) AS player\_rank |
|  | FROM |
|  | player\_score |
|  | ) |
|  | SELECT |
|  | group\_id, |
|  | player\_id |
|  | FROM |
|  | ranked\_player |
|  | WHERE |
|  | player\_rank = 1 |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE players; |
|  | DROP TABLE matches; |
|  |  |
|  |  |
|  | --Q133: |
|  |  |
|  | CREATE TABLE student |
|  | ( |
|  | student\_id INT, |
|  | student\_name VARCHAR(25), |
|  | CONSTRAINT pk\_student PRIMARY KEY(student\_id) |
|  | ); |
|  |  |
|  | CREATE TABLE exam |
|  | ( |
|  | exam\_id INT, |
|  | student\_id INT, |
|  | score INT, |
|  | CONSTRAINT pk\_exam PRIMARY KEY(exam\_id, student\_id) |
|  | ); |
|  |  |
|  | INSERT INTO student VALUES(1, 'Daniel'); |
|  | INSERT INTO student VALUES(2, 'Jade'); |
|  | INSERT INTO student VALUES(3, 'Stella'); |
|  | INSERT INTO student VALUES(4, 'Jonathan'); |
|  | INSERT INTO student VALUES(5, 'Will'); |
|  |  |
|  | INSERT INTO exam VALUES(10, 1, 70); |
|  | INSERT INTO exam VALUES(10, 2, 80); |
|  | INSERT INTO exam VALUES(10, 3, 90); |
|  | INSERT INTO exam VALUES(20, 1, 80); |
|  | INSERT INTO exam VALUES(30, 1, 70); |
|  | INSERT INTO exam VALUES(30, 3, 80); |
|  | INSERT INTO exam VALUES(30, 4, 90); |
|  | INSERT INTO exam VALUES(40, 1, 60); |
|  | INSERT INTO exam VALUES(40, 2, 70); |
|  | INSERT INTO exam VALUES(40, 4, 80); |
|  |  |
|  | WITH exam\_highest\_lowest AS( |
|  | SELECT |
|  | \*, |
|  | FIRST\_VALUE(score) OVER(PARTITION BY exam\_id ORDER BY score) as exam\_lowest, |
|  | FIRST\_VALUE(score) OVER(PARTITION BY exam\_id ORDER BY score DESC) as exam\_highest |
|  | FROM |
|  | exam |
|  | ), |
|  | student\_highest\_lowest AS( |
|  | SELECT |
|  | DISTINCT student\_id |
|  | FROM |
|  | exam\_highest\_lowest |
|  | WHERE |
|  | score = exam\_lowest |
|  | OR score = exam\_highest |
|  | ) |
|  | SELECT |
|  | student\_id, |
|  | student\_name |
|  | FROM |
|  | student s |
|  | WHERE |
|  | EXISTS( |
|  | SELECT |
|  | \* |
|  | FROM |
|  | exam e |
|  | WHERE |
|  | e.student\_id = s.student\_id |
|  | ) |
|  | AND s.student\_id NOT IN( |
|  | SELECT |
|  | student\_id |
|  | FROM |
|  | student\_highest\_lowest |
|  | ) |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE student; |
|  | DROP TABLE exam; |
|  |  |
|  |  |
|  | --Q134: |
|  |  |
|  | CREATE TABLE student |
|  | ( |
|  | student\_id INT, |
|  | student\_name VARCHAR(25), |
|  | CONSTRAINT pk\_student PRIMARY KEY(student\_id) |
|  | ); |
|  |  |
|  | CREATE TABLE exam |
|  | ( |
|  | exam\_id INT, |
|  | student\_id INT, |
|  | score INT, |
|  | CONSTRAINT pk\_exam PRIMARY KEY(exam\_id, student\_id) |
|  | ); |
|  |  |
|  | INSERT INTO student VALUES(1, 'Daniel'); |
|  | INSERT INTO student VALUES(2, 'Jade'); |
|  | INSERT INTO student VALUES(3, 'Stella'); |
|  | INSERT INTO student VALUES(4, 'Jonathan'); |
|  | INSERT INTO student VALUES(5, 'Will'); |
|  |  |
|  | INSERT INTO exam VALUES(10, 1, 70); |
|  | INSERT INTO exam VALUES(10, 2, 80); |
|  | INSERT INTO exam VALUES(10, 3, 90); |
|  | INSERT INTO exam VALUES(20, 1, 80); |
|  | INSERT INTO exam VALUES(30, 1, 70); |
|  | INSERT INTO exam VALUES(30, 3, 80); |
|  | INSERT INTO exam VALUES(30, 4, 90); |
|  | INSERT INTO exam VALUES(40, 1, 60); |
|  | INSERT INTO exam VALUES(40, 2, 70); |
|  | INSERT INTO exam VALUES(40, 4, 80); |
|  |  |
|  | WITH exam\_highest\_lowest AS( |
|  | SELECT |
|  | \*, |
|  | FIRST\_VALUE(score) OVER(PARTITION BY exam\_id ORDER BY score) as exam\_lowest, |
|  | FIRST\_VALUE(score) OVER(PARTITION BY exam\_id ORDER BY score DESC) as exam\_highest |
|  | FROM |
|  | exam |
|  | ), |
|  | student\_highest\_lowest AS( |
|  | SELECT |
|  | DISTINCT student\_id |
|  | FROM |
|  | exam\_highest\_lowest |
|  | WHERE |
|  | score = exam\_lowest |
|  | OR score = exam\_highest |
|  | ) |
|  | SELECT |
|  | student\_id, |
|  | student\_name |
|  | FROM |
|  | student s |
|  | WHERE |
|  | EXISTS( |
|  | SELECT |
|  | \* |
|  | FROM |
|  | exam e |
|  | WHERE |
|  | e.student\_id = s.student\_id |
|  | ) |
|  | AND s.student\_id NOT IN( |
|  | SELECT |
|  | student\_id |
|  | FROM |
|  | student\_highest\_lowest |
|  | ) |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE student; |
|  | DROP TABLE exam; |
|  |  |
|  |  |
|  | --Q135: |
|  |  |
|  | CREATE TABLE user\_activity |
|  | ( |
|  | username VARCHAR(25), |
|  | activity VARCHAR(25), |
|  | start\_date DATE, |
|  | end\_date DATE |
|  | ); |
|  |  |
|  | INSERT INTO user\_activity VALUES('Alice', 'Travel', '2020-02-12', '2020-02-20'); |
|  | INSERT INTO user\_activity VALUES('Alice', 'Dancing', '2020-02-21', '2020-02-23'); |
|  | INSERT INTO user\_activity VALUES('Alice', 'Travel', '2020-02-24', '2020-02-28'); |
|  | INSERT INTO user\_activity VALUES('Bob', 'Travel', '2020-02-11', '2020-02-18'); |
|  |  |
|  | WITH activity\_stat\_by\_user AS( |
|  | SELECT |
|  | \*, |
|  | DENSE\_RANK() OVER(PARTITION BY username ORDER BY start\_date) AS activity\_serial, |
|  | COUNT(\*) OVER(PARTITION BY username) AS total\_activity\_count\_by\_user |
|  | FROM |
|  | user\_activity |
|  | ) |
|  | SELECT |
|  | username, |
|  | activity, |
|  | start\_date, |
|  | end\_date |
|  | FROM |
|  | activity\_stat\_by\_user |
|  | WHERE |
|  | CASE |
|  | WHEN total\_activity\_count\_by\_user = 1 |
|  | THEN 1 |
|  | WHEN activity\_serial = 2 |
|  | THEN 1 |
|  | END = 1 |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE user\_activity; |
|  |  |
|  |  |
|  | --Q136: |
|  |  |
|  | CREATE TABLE user\_activity |
|  | ( |
|  | username VARCHAR(25), |
|  | activity VARCHAR(25), |
|  | start\_date DATE, |
|  | end\_date DATE |
|  | ); |
|  |  |
|  | INSERT INTO user\_activity VALUES('Alice', 'Travel', '2020-02-12', '2020-02-20'); |
|  | INSERT INTO user\_activity VALUES('Alice', 'Dancing', '2020-02-21', '2020-02-23'); |
|  | INSERT INTO user\_activity VALUES('Alice', 'Travel', '2020-02-24', '2020-02-28'); |
|  | INSERT INTO user\_activity VALUES('Bob', 'Travel', '2020-02-11', '2020-02-18'); |
|  |  |
|  | WITH activity\_stat\_by\_user AS( |
|  | SELECT |
|  | \*, |
|  | DENSE\_RANK() OVER(PARTITION BY username ORDER BY start\_date) AS activity\_serial, |
|  | COUNT(\*) OVER(PARTITION BY username) AS total\_activity\_count\_by\_user |
|  | FROM |
|  | user\_activity |
|  | ) |
|  | SELECT |
|  | username, |
|  | activity, |
|  | start\_date, |
|  | end\_date |
|  | FROM |
|  | activity\_stat\_by\_user |
|  | WHERE |
|  | CASE |
|  | WHEN total\_activity\_count\_by\_user = 1 |
|  | THEN 1 |
|  | WHEN activity\_serial = 2 |
|  | THEN 1 |
|  | END = 1 |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE user\_activity; |
|  |  |
|  |  |
|  | --Q137: |
|  |  |
|  | CREATE TABLE employees |
|  | ( |
|  | id INT, |
|  | name VARCHAR(25), |
|  | salary INT |
|  | ); |
|  |  |
|  | INSERT INTO employees VALUES(1, 'Kristeen', 1420); |
|  | INSERT INTO employees VALUES(2, 'Ashley', 2006); |
|  | INSERT INTO employees VALUES(3, 'Julia', 2210); |
|  | INSERT INTO employees VALUES(4, 'Maria', 3000); |
|  |  |
|  | SELECT |
|  | ROUND(AVG(salary) -AVG(CAST(REPLACE(CAST(salary AS CHAR),'0','') AS UNSIGNED)), 2) AS diff\_average |
|  | FROM |
|  | employees |
|  | WHERE |
|  | salary > 1000 |
|  | AND salary < 100000 |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE employees; |
|  |  |
|  |  |
|  | --Q138: |
|  |  |
|  | CREATE TABLE employee |
|  | ( |
|  | employee\_id INT, |
|  | name VARCHAR(25), |
|  | months INT, |
|  | salary INT |
|  | ); |
|  |  |
|  | INSERT INTO employee VALUES(12228, 'Rose', 15, 1968); |
|  | INSERT INTO employee VALUES(33645, 'Angela', 1, 3443); |
|  | INSERT INTO employee VALUES(45692, 'Frank', 17, 1608); |
|  | INSERT INTO employee VALUES(56118, 'Patrick', 7, 1345); |
|  | INSERT INTO employee VALUES(59725, 'Lisa', 11, 2330); |
|  | INSERT INTO employee VALUES(74197, 'Kimberly', 16, 4372); |
|  | INSERT INTO employee VALUES(78454, 'Bonnie', 8, 1771); |
|  | INSERT INTO employee VALUES(83565, 'Michael', 6, 2017); |
|  | INSERT INTO employee VALUES(98607, 'Todd', 5, 3396); |
|  | INSERT INTO employee VALUES(99989, 'Joe', 9, 3573); |
|  |  |
|  | WITH employee\_max\_sal\_count AS( |
|  | SELECT |
|  | months \* salary AS max\_sal, |
|  | count(\*) AS emp\_count |
|  | FROM |
|  | employee |
|  | WHERE |
|  | months \* salary = ( |
|  | SELECT |
|  | MAX(months \* salary) |
|  | FROM |
|  | employee |
|  | ) |
|  | GROUP BY |
|  | months \* salary |
|  | ) |
|  | SELECT |
|  | CONCAT(max\_sal, ' ', emp\_count) |
|  | FROM |
|  | employee\_max\_sal\_count |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE employee; |
|  |  |
|  |  |
|  | --Q139: |
|  |  |
|  | CREATE TABLE occupations |
|  | ( |
|  | name VARCHAR(25), |
|  | occupation VARCHAR(25) |
|  | ); |
|  |  |
|  | INSERT INTO occupations VALUES('Samantha', 'Doctor'); |
|  | INSERT INTO occupations VALUES('Julia', 'Actor'); |
|  | INSERT INTO occupations VALUES('Maria', 'Actor'); |
|  | INSERT INTO occupations VALUES('Meera', 'Singer'); |
|  | INSERT INTO occupations VALUES('Ashely', 'Professor'); |
|  | INSERT INTO occupations VALUES('Ketty', 'Professor'); |
|  | INSERT INTO occupations VALUES('Christeen', 'Professor'); |
|  | INSERT INTO occupations VALUES('Jane', 'Actor'); |
|  | INSERT INTO occupations VALUES('Jenny', 'Doctor'); |
|  | INSERT INTO occupations VALUES('Priya', 'Singer'); |
|  |  |
|  | SELECT |
|  | CONCAT(name, '(', LEFT(occupation,1), ')') AS name\_with\_occupation |
|  | FROM |
|  | occupations; |
|  |  |
|  | WITH occupations\_stat AS( |
|  | SELECT |
|  | occupation, |
|  | COUNT(\*) AS individual\_count |
|  | FROM |
|  | occupations |
|  | GROUP BY |
|  | occupation |
|  | ) |
|  | SELECT |
|  | CONCAT('There are a total of ', individual\_count, ' ', LOWER(occupation)) |
|  | FROM |
|  | occupations\_stat |
|  | ORDER BY |
|  | individual\_count, |
|  | occupation |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE occupations; |
|  |  |
|  | --Q140: |
|  |  |
|  | CREATE TABLE occupations |
|  | ( |
|  | name VARCHAR(25), |
|  | occupation VARCHAR(25) |
|  | ); |
|  |  |
|  | INSERT INTO occupations VALUES('Samantha', 'Doctor'); |
|  | INSERT INTO occupations VALUES('Julia', 'Actor'); |
|  | INSERT INTO occupations VALUES('Maria', 'Actor'); |
|  | INSERT INTO occupations VALUES('Meera', 'Singer'); |
|  | INSERT INTO occupations VALUES('Ashely', 'Professor'); |
|  | INSERT INTO occupations VALUES('Ketty', 'Professor'); |
|  | INSERT INTO occupations VALUES('Christeen', 'Professor'); |
|  | INSERT INTO occupations VALUES('Jane', 'Actor'); |
|  | INSERT INTO occupations VALUES('Jenny', 'Doctor'); |
|  | INSERT INTO occupations VALUES('Priya', 'Singer'); |
|  |  |
|  | WITH serialized\_ocp AS( |
|  | SELECT |
|  | name, |
|  | occupation, |
|  | row\_number() over(partition by occupation order by name) as serial |
|  | FROM |
|  | occupations |
|  | ) |
|  | SELECT |
|  | MAX(CASE |
|  | WHEN occupation = 'Doctor' |
|  | THEN |
|  | name |
|  | END) AS Doctor, |
|  | MAX(CASE |
|  | WHEN occupation = 'Professor' |
|  | THEN |
|  | name |
|  | END) AS Professor, |
|  | MAX(CASE |
|  | WHEN occupation = 'Singer' |
|  | THEN |
|  | name |
|  | END) AS Singer, |
|  | MAX(CASE |
|  | WHEN occupation = 'Actor' |
|  | THEN |
|  | name |
|  | END) AS Actor |
|  | FROM |
|  | serialized\_ocp |
|  | GROUP BY |
|  | serial |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE occupations; |
|  |  |
|  | --Q141: |
|  |  |
|  | CREATE TABLE bst |
|  | ( |
|  | n INT, |
|  | p INT |
|  | ); |
|  |  |
|  | INSERT INTO bst VALUES(1, 2); |
|  | INSERT INTO bst VALUES(3, 2); |
|  | INSERT INTO bst VALUES(6, 8); |
|  | INSERT INTO bst VALUES(9, 8); |
|  | INSERT INTO bst VALUES(2, 5); |
|  | INSERT INTO bst VALUES(8, 5); |
|  | INSERT INTO bst VALUES(5, null); |
|  |  |
|  |  |
|  | SELECT |
|  | n, |
|  | CASE |
|  | WHEN p IS NULL |
|  | THEN 'Root' |
|  | WHEN n IN ( SELECT p FROM bst) |
|  | THEN 'Inner' |
|  | ELSE |
|  | 'Leaf' |
|  | END as node\_type |
|  | FROM |
|  | bst |
|  | ORDER BY |
|  | n |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE bst; |
|  |  |
|  |  |
|  | --Q142: |
|  |  |
|  | CREATE TABLE company |
|  | ( |
|  | company\_code VARCHAR(25), |
|  | founder VARCHAR(25) |
|  | ); |
|  |  |
|  | CREATE TABLE lead\_manager |
|  | ( |
|  | lead\_manager\_code VARCHAR(25), |
|  | company\_code VARCHAR(25) |
|  | ); |
|  |  |
|  | CREATE TABLE senior\_manager |
|  | ( |
|  | senior\_manager\_code VARCHAR(25), |
|  | lead\_manager\_code VARCHAR(25), |
|  | company\_code VARCHAR(25) |
|  | ); |
|  |  |
|  | CREATE TABLE manager |
|  | ( |
|  | manager\_code VARCHAR(25), |
|  | senior\_manager\_code VARCHAR(25), |
|  | lead\_manager\_code VARCHAR(25), |
|  | company\_code VARCHAR(25) |
|  | ); |
|  |  |
|  | CREATE TABLE employee |
|  | ( |
|  | employee\_code VARCHAR(25), |
|  | manager\_code VARCHAR(25), |
|  | senior\_manager\_code VARCHAR(25), |
|  | lead\_manager\_code VARCHAR(25), |
|  | company\_code VARCHAR(25) |
|  | ); |
|  |  |
|  | INSERT INTO company VALUES('C1', 'Monika'); |
|  | INSERT INTO company VALUES('C2', 'Samantha'); |
|  |  |
|  | INSERT INTO lead\_manager VALUES('LM1', 'C1'); |
|  | INSERT INTO lead\_manager VALUES('LM2', 'C2'); |
|  |  |
|  | INSERT INTO senior\_manager VALUES('SM1', 'LM1', 'C1'); |
|  | INSERT INTO senior\_manager VALUES('SM2', 'LM1', 'C1'); |
|  | INSERT INTO senior\_manager VALUES('SM3', 'LM2', 'C2'); |
|  |  |
|  | INSERT INTO manager VALUES('M1', 'SM1', 'LM1', 'C1'); |
|  | INSERT INTO manager VALUES('M2', 'SM3', 'LM2', 'C2'); |
|  | INSERT INTO manager VALUES('M3', 'SM3', 'LM2', 'C2'); |
|  |  |
|  | INSERT INTO employee VALUES('E1', 'M1', 'SM1', 'LM1', 'C1'); |
|  | INSERT INTO employee VALUES('E2', 'M1', 'SM1', 'LM1', 'C1'); |
|  | INSERT INTO employee VALUES('E3', 'M2', 'SM3', 'LM2', 'C2'); |
|  | INSERT INTO employee VALUES('E4', 'M3', 'SM3', 'LM2', 'C2'); |
|  |  |
|  |  |
|  | SELECT |
|  | c.company\_code, |
|  | c.founder, |
|  | count(distinct lm.lead\_manager\_code) AS lead\_manager\_count, |
|  | count(distinct sm.senior\_manager\_code) AS senior\_manager\_count, |
|  | count(distinct m.manager\_code) AS manager\_count, |
|  | count(distinct e.employee\_code) AS employeee\_count |
|  | FROM |
|  | lead\_manager lm |
|  | LEFT JOIN senior\_manager sm ON lm.lead\_manager\_code = sm.lead\_manager\_code |
|  | LEFT JOIN manager m ON m.senior\_manager\_code = sm.senior\_manager\_code |
|  | LEFT JOIN employee e ON e.manager\_code = m.manager\_code |
|  | LEFT JOIN company c ON c.company\_code = lm.company\_code |
|  | GROUP BY |
|  | c.company\_code, |
|  | c.founder |
|  | ORDER BY |
|  | c.company\_code |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE company; |
|  | DROP TABLE lead\_manager; |
|  | DROP TABLE senior\_manager; |
|  | DROP TABLE manager; |
|  | DROP TABLE employee; |
|  |  |
|  | --Q143: |
|  |  |
|  | CREATE TABLE functions |
|  | ( |
|  | x INT, |
|  | y INT |
|  | ); |
|  |  |
|  | INSERT INTO functions VALUES(20, 20); |
|  | INSERT INTO functions VALUES(20, 20); |
|  | INSERT INTO functions VALUES(20, 21); |
|  | INSERT INTO functions VALUES(23, 22); |
|  | INSERT INTO functions VALUES(22, 23); |
|  | INSERT INTO functions VALUES(21, 20); |
|  |  |
|  | WITH functions\_serialized AS( |
|  | SELECT |
|  | \*, |
|  | ROW\_NUMBER() OVER(ORDER BY x) AS serial |
|  | FROM |
|  | functions |
|  | ) |
|  | SELECT |
|  | DISTINCT f1.x, |
|  | f1.y |
|  | FROM |
|  | functions\_serialized f1 |
|  | WHERE |
|  | EXISTS( |
|  | SELECT |
|  | \* |
|  | FROM |
|  | functions\_serialized f2 |
|  | WHERE |
|  | f1.serial <> f2.serial |
|  | AND f1.x = f2.y |
|  | AND f1.y = f2.x |
|  | ) |
|  | AND f1.x <= f1.y |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE functions; |
|  |  |
|  |  |
|  | --Q144: |
|  |  |
|  | CREATE TABLE students |
|  | ( |
|  | id INT, |
|  | name VARCHAR(25) |
|  | ); |
|  |  |
|  | CREATE TABLE friends |
|  | ( |
|  | id INT, |
|  | friend\_id INT |
|  | ); |
|  |  |
|  | CREATE TABLE packages |
|  | ( |
|  | id INT, |
|  | salary DECIMAL(10, 2) |
|  | ); |
|  |  |
|  | INSERT INTO students VALUES(1, 'Ashley'); |
|  | INSERT INTO students VALUES(2, 'Samantha'); |
|  | INSERT INTO students VALUES(3, 'Julia'); |
|  | INSERT INTO students VALUES(4, 'Scarlet'); |
|  |  |
|  | INSERT INTO friends VALUES(1, 2); |
|  | INSERT INTO friends VALUES(2, 3); |
|  | INSERT INTO friends VALUES(3, 4); |
|  | INSERT INTO friends VALUES(4, 1); |
|  |  |
|  | INSERT INTO packages VALUES(1, 15.20); |
|  | INSERT INTO packages VALUES(2, 10.06); |
|  | INSERT INTO packages VALUES(3, 11.55); |
|  | INSERT INTO packages VALUES(4, 12.12); |
|  |  |
|  |  |
|  | SELECT |
|  | s.name |
|  | FROM |
|  | friends f |
|  | JOIN packages p1 ON f.id = p1.id |
|  | JOIN packages p2 ON p2.id = f.friend\_id |
|  | JOIN students s ON s.id = f.id |
|  | WHERE |
|  | p2.salary > p1.salary |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE students; |
|  | DROP TABLE friends; |
|  | DROP TABLE packages; |
|  |  |
|  |  |
|  | --Q145: |
|  |  |
|  | CREATE TABLE hackers |
|  | ( |
|  | hacker\_id INT, |
|  | name VARCHAR(25) |
|  | ); |
|  |  |
|  | CREATE TABLE difficulty |
|  | ( |
|  | difficulty\_level INT, |
|  | score INT |
|  | ); |
|  |  |
|  | CREATE TABLE challenges |
|  | ( |
|  | challenge\_id INT, |
|  | hacker\_id INT, |
|  | difficulty\_level INT |
|  | ); |
|  |  |
|  | CREATE TABLE submissions |
|  | ( |
|  | submission\_id INT, |
|  | hacker\_id INT, |
|  | challenge\_id INT, |
|  | score INT |
|  | ); |
|  |  |
|  | INSERT INTO hackers VALUES(5580, 'Rose'); |
|  | INSERT INTO hackers VALUES(8439, 'Angela'); |
|  | INSERT INTO hackers VALUES(27205, 'Frank'); |
|  | INSERT INTO hackers VALUES(52243, 'Patrick'); |
|  | INSERT INTO hackers VALUES(52348, 'Lisa'); |
|  | INSERT INTO hackers VALUES(57645, 'Kimberly'); |
|  | INSERT INTO hackers VALUES(77726, 'Bonnie'); |
|  | INSERT INTO hackers VALUES(83082, 'Michael'); |
|  | INSERT INTO hackers VALUES(86870, 'Todd'); |
|  | INSERT INTO hackers VALUES(90411, 'Joe'); |
|  |  |
|  | INSERT INTO difficulty VALUES(1, 20); |
|  | INSERT INTO difficulty VALUES(2, 30); |
|  | INSERT INTO difficulty VALUES(3, 40); |
|  | INSERT INTO difficulty VALUES(4, 60); |
|  | INSERT INTO difficulty VALUES(5, 80); |
|  | INSERT INTO difficulty VALUES(6, 100); |
|  | INSERT INTO difficulty VALUES(7, 120); |
|  |  |
|  | INSERT INTO challenges VALUES(4810, 77726, 4); |
|  | INSERT INTO challenges VALUES(21089, 27205, 1); |
|  | INSERT INTO challenges VALUES(36566, 5580, 7); |
|  | INSERT INTO challenges VALUES(66730, 52243, 6); |
|  | INSERT INTO challenges VALUES(71055, 52243, 2); |
|  |  |
|  | INSERT INTO submissions VALUES(68628, 77726, 36566, 30); |
|  | INSERT INTO submissions VALUES(65300, 77726, 21089, 10); |
|  | INSERT INTO submissions VALUES(40326, 52243, 36566, 77); |
|  | INSERT INTO submissions VALUES(8941, 27205, 4810, 4); |
|  | INSERT INTO submissions VALUES(83554, 77726, 66730, 30); |
|  | INSERT INTO submissions VALUES(43353, 52243, 66730, 0); |
|  | INSERT INTO submissions VALUES(55385, 52348, 71055, 20); |
|  | INSERT INTO submissions VALUES(39784, 27205, 71055, 23); |
|  | INSERT INTO submissions VALUES(94613, 86870, 71055, 30); |
|  | INSERT INTO submissions VALUES(45788, 52348, 36566, 0); |
|  | INSERT INTO submissions VALUES(93058, 86870, 36566, 30); |
|  | INSERT INTO submissions VALUES(7344, 8439, 66730, 92); |
|  | INSERT INTO submissions VALUES(2721, 8439, 4810, 36); |
|  | INSERT INTO submissions VALUES(523, 5580, 71055, 4); |
|  | INSERT INTO submissions VALUES(49105, 52348, 66730, 0); |
|  | INSERT INTO submissions VALUES(55877, 57645, 66730, 80); |
|  | INSERT INTO submissions VALUES(38355, 27205, 66730, 35); |
|  | INSERT INTO submissions VALUES(3924, 8439, 36566, 80); |
|  | INSERT INTO submissions VALUES(97397, 90411, 66730, 100); |
|  | INSERT INTO submissions VALUES(84162, 83082, 4810, 40); |
|  | INSERT INTO submissions VALUES(97431, 90411, 71055, 30); |
|  |  |
|  |  |
|  | SELECT |
|  | h.hacker\_id, |
|  | h.name |
|  | FROM |
|  | submissions s |
|  | JOIN challenges c ON c.challenge\_id = s.challenge\_id |
|  | JOIN difficulty d ON d.difficulty\_level = c.difficulty\_level |
|  | JOIN hackers h ON h.hacker\_id = s.hacker\_id |
|  | WHERE |
|  | s.score = d.score |
|  | GROUP BY |
|  | h.hacker\_id, |
|  | h.name |
|  | ORDER BY |
|  | COUNT(\*) DESC, |
|  | h.hacker\_id |
|  | LIMIT 1 |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE hackers; |
|  | DROP TABLE difficulty; |
|  | DROP TABLE challenges; |
|  | DROP TABLE submissions; |
|  |  |
|  |  |
|  | --Q146: |
|  |  |
|  | CREATE TABLE projects |
|  | ( |
|  | task\_id INT, |
|  | start\_date DATE, |
|  | end\_date DATE |
|  | ); |
|  |  |
|  | INSERT INTO projects VALUES(1, '2015-10-01', '2015-10-02'); |
|  | INSERT INTO projects VALUES(1, '2015-10-02', '2015-10-03'); |
|  | INSERT INTO projects VALUES(1, '2015-10-03', '2015-10-04'); |
|  | INSERT INTO projects VALUES(1, '2015-10-13', '2015-10-14'); |
|  | INSERT INTO projects VALUES(1, '2015-10-14', '2015-10-15'); |
|  | INSERT INTO projects VALUES(1, '2015-10-28', '2015-10-29'); |
|  | INSERT INTO projects VALUES(1, '2015-10-30', '2015-10-31'); |
|  |  |
|  |  |
|  | WITH diff\_projects AS( |
|  | SELECT |
|  | \*, |
|  | LEAD(end\_date) OVER(ORDER BY end\_date) AS next\_end\_date, |
|  | DATEDIFF(LEAD(end\_date) OVER(ORDER BY end\_date), end\_date) AS diff\_next\_end\_date, |
|  | DATEDIFF(end\_date, LAG(end\_date) OVER(ORDER BY end\_date)) AS diff\_prev\_end\_date |
|  | FROM |
|  | projects |
|  | ORDER BY |
|  | end\_date |
|  | ), |
|  | projects\_start\_end AS( |
|  | SELECT |
|  | CASE |
|  | WHEN diff\_next\_end\_date = 1 |
|  | THEN start\_date |
|  | WHEN diff\_next\_end\_date <> 1 AND diff\_prev\_end\_date <> 1 |
|  | THEN start\_date |
|  | WHEN diff\_next\_end\_date IS NULL AND diff\_prev\_end\_date <> 1 |
|  | THEN start\_date |
|  | WHEN diff\_next\_end\_date <> 1 AND diff\_prev\_end\_date IS NULL |
|  | THEN start\_date |
|  | END as project\_start\_date, |
|  | CASE |
|  | WHEN LEAD(diff\_prev\_end\_date) OVER(ORDER BY end\_date) = 1 |
|  | THEN LEAD(end\_date) OVER(ORDER BY end\_date) |
|  | WHEN diff\_next\_end\_date <> 1 AND diff\_prev\_end\_date <> 1 |
|  | THEN end\_date |
|  | WHEN diff\_next\_end\_date IS NULL AND diff\_prev\_end\_date <> 1 |
|  | THEN end\_date |
|  | WHEN diff\_next\_end\_date <> 1 AND diff\_prev\_end\_date IS NULL |
|  | THEN end\_date |
|  | END as project\_end\_date |
|  |  |
|  | FROM |
|  | diff\_projects |
|  | WHERE |
|  | NOT (diff\_next\_end\_date = 1 |
|  | AND diff\_prev\_end\_date = 1 ) |
|  | OR ( |
|  | diff\_next\_end\_date IS NULL |
|  | OR diff\_prev\_end\_date IS NULL |
|  | ) |
|  | ) |
|  | SELECT |
|  | \* |
|  | FROM |
|  | projects\_start\_end |
|  | WHERE |
|  | project\_end\_date IS NOT NULL |
|  | ORDER BY |
|  | DATEDIFF(project\_end\_date, project\_start\_date), |
|  | project\_start\_date |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE projects; |
|  |  |
|  |  |
|  | --Q147: |
|  |  |
|  | CREATE TABLE transactions |
|  | ( |
|  | user\_id INT, |
|  | amount DECIMAL(10,2), |
|  | transaction\_date TIMESTAMP |
|  | ); |
|  |  |
|  | INSERT INTO transactions VALUES(1, '9.99', '2022-08-01 10:00:00'); |
|  | INSERT INTO transactions VALUES(1, '55', '2022-08-17 10:00:00'); |
|  | INSERT INTO transactions VALUES(1, '149.5', '2022-08-05 10:00:00'); |
|  | INSERT INTO transactions VALUES(1, '4.89', '2022-08-06 10:00:00'); |
|  | INSERT INTO transactions VALUES(1, '34', '2022-08-07 10:00:00'); |
|  |  |
|  |  |
|  | SELECT |
|  | \*, |
|  | DATEDIFF(LEAD(transaction\_date) OVER(PARTITION BY user\_id ORDER BY transaction\_date), transaction\_date) AS diff1, |
|  | DATEDIFF(transaction\_date, LAG(transaction\_date) OVER(PARTITION BY user\_id ORDER BY transaction\_date)) AS diff2 |
|  | FROM |
|  | transactions; |
|  |  |
|  | SELECT |
|  | distinct user\_id |
|  | FROM( |
|  | SELECT |
|  | \*, |
|  | DATEDIFF(LEAD(transaction\_date) OVER(PARTITION BY user\_id ORDER BY transaction\_date), transaction\_date) AS diff1, |
|  | DATEDIFF(transaction\_date, LAG(transaction\_date) OVER(PARTITION BY user\_id ORDER BY transaction\_date)) AS diff2 |
|  | FROM |
|  | transactions |
|  | ) AS trx\_with\_date\_diff |
|  | WHERE |
|  | trx\_with\_date\_diff.diff1 = 1 |
|  | AND trx\_with\_date\_diff.diff2 = 1 |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE transactions; |
|  |  |
|  |  |
|  | --Q148: |
|  |  |
|  | CREATE TABLE payments |
|  | ( |
|  | payer\_id INT, |
|  | recipient\_id INT, |
|  | amount INT |
|  | ); |
|  |  |
|  | INSERT INTO payments VALUES(101, 201, 30); |
|  | INSERT INTO payments VALUES(201, 101, 10); |
|  | INSERT INTO payments VALUES(101, 301, 20); |
|  | INSERT INTO payments VALUES(301, 101, 80); |
|  | INSERT INTO payments VALUES(201, 301, 70); |
|  |  |
|  | SELECT |
|  | ROUND(COUNT( |
|  | distinct p1.payer\_id, |
|  | p1.recipient\_id |
|  | )/2) AS unique\_relationships |
|  | FROM |
|  | payments p1 |
|  | JOIN payments p2 ON p1. payer\_id = p2.recipient\_id |
|  | AND p1.recipient\_id = p2.payer\_id; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE payments; |
|  |  |
|  |  |
|  | --Q149: |
|  |  |
|  | CREATE TABLE user\_transactions |
|  | ( |
|  | transaction\_id INT, |
|  | user\_id INT, |
|  | spend DECIMAL(10,2), |
|  | transaction\_date TIMESTAMP |
|  | ); |
|  |  |
|  | INSERT INTO user\_transactions VALUES(759274, 111, 49.50, '2022-02-03 00:00:00'); |
|  | INSERT INTO user\_transactions VALUES(850371, 111, 51.00, '2022-03-15 00:00:00'); |
|  | INSERT INTO user\_transactions VALUES(615348, 145, 36.30, '2022-03-22 00:00:00'); |
|  | INSERT INTO user\_transactions VALUES(137424, 156, 151.00, '2022-04-04 00:00:00'); |
|  | INSERT INTO user\_transactions VALUES(248475, 156, 87.00, '2022-02-16 00:00:00'); |
|  |  |
|  | WITH user\_trx\_serialized AS( |
|  | SELECT |
|  | \*, |
|  | DENSE\_RANK() OVER(PARTITION BY user\_id ORDER BY transaction\_date) as serial |
|  | FROM |
|  | user\_transactions |
|  | ) |
|  | SELECT |
|  | COUNT(\*) AS users |
|  | FROM |
|  | user\_trx\_serialized |
|  | WHERE |
|  | serial = 1 |
|  | AND spend >= 50 |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE user\_transactions; |
|  |  |
|  |  |
|  | --Q150: |
|  |  |
|  | CREATE TABLE measurements |
|  | ( |
|  | measurement\_id INT, |
|  | measurement\_value DECIMAL(10,2), |
|  | measurement\_time TIMESTAMP |
|  | ); |
|  |  |
|  | INSERT INTO measurements VALUES(131233, 1109.51, '2022-07-10 09:00:00'); |
|  | INSERT INTO measurements VALUES(135211, 1662.74, '2022-07-10 11:00:00'); |
|  | INSERT INTO measurements VALUES(523542, 1246.24, '2022-07-10 13:15:00'); |
|  | INSERT INTO measurements VALUES(143562, 1124.50, '2022-07-11 15:00:00'); |
|  | INSERT INTO measurements VALUES(346462, 1234.14, '2022-07-11 16:45:00'); |
|  |  |
|  |  |
|  | WITH daily\_measurement AS( |
|  | SELECT |
|  | \*, |
|  | DATE\_FORMAT(measurement\_time, '%m/%d/%Y 00:00:00') AS measurement\_day, |
|  | DENSE\_RANK() OVER(PARTITION BY DATE\_FORMAT(measurement\_time, 'm%/%d/%Y') ORDER BY measurement\_time) AS serial |
|  |  |
|  | FROM |
|  | measurements |
|  | ) |
|  | SELECT |
|  | measurement\_day, |
|  | SUM(CASE |
|  | WHEN serial % 2 = 1 |
|  | THEN measurement\_value |
|  | END) as odd\_sum, |
|  | SUM(CASE |
|  | WHEN serial % 2 = 0 |
|  | THEN measurement\_value |
|  | END) as even\_sum |
|  | FROM |
|  | daily\_measurement |
|  | GROUP BY |
|  | measurement\_day; |
|  |  |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE measurements; |
|  |  |
|  |  |
|  | --Q151: |
|  |  |
|  | CREATE TABLE transactions |
|  | ( |
|  | user\_id INT, |
|  | amount DECIMAL(10,2), |
|  | transaction\_date TIMESTAMP |
|  | ); |
|  |  |
|  | INSERT INTO transactions VALUES(1, '9.99', '2022-08-01 10:00:00'); |
|  | INSERT INTO transactions VALUES(1, '55', '2022-08-17 10:00:00'); |
|  | INSERT INTO transactions VALUES(1, '149.5', '2022-08-05 10:00:00'); |
|  | INSERT INTO transactions VALUES(1, '4.89', '2022-08-06 10:00:00'); |
|  | INSERT INTO transactions VALUES(1, '34', '2022-08-07 10:00:00'); |
|  |  |
|  |  |
|  | SELECT |
|  | \*, |
|  | DATEDIFF(LEAD(transaction\_date) OVER(PARTITION BY user\_id ORDER BY transaction\_date), transaction\_date) AS diff1, |
|  | DATEDIFF(transaction\_date, LAG(transaction\_date) OVER(PARTITION BY user\_id ORDER BY transaction\_date)) AS diff2 |
|  | FROM |
|  | transactions; |
|  |  |
|  | SELECT |
|  | distinct user\_id |
|  | FROM( |
|  | SELECT |
|  | \*, |
|  | DATEDIFF(LEAD(transaction\_date) OVER(PARTITION BY user\_id ORDER BY transaction\_date), transaction\_date) AS diff1, |
|  | DATEDIFF(transaction\_date, LAG(transaction\_date) OVER(PARTITION BY user\_id ORDER BY transaction\_date)) AS diff2 |
|  | FROM |
|  | transactions |
|  | ) AS trx\_with\_date\_diff |
|  | WHERE |
|  | trx\_with\_date\_diff.diff1 = 1 |
|  | AND trx\_with\_date\_diff.diff2 = 1 |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE transactions; |
|  |  |
|  |  |
|  | --Q152: |
|  |  |
|  | CREATE TABLE rental\_amenities |
|  | ( |
|  | rental\_id INT, |
|  | amenity VARCHAR(40) |
|  | ); |
|  |  |
|  | INSERT INTO rental\_amenities VALUES(123, 'pool'); |
|  | INSERT INTO rental\_amenities VALUES(123, 'kitchen'); |
|  | INSERT INTO rental\_amenities VALUES(234, 'hit tub'); |
|  | INSERT INTO rental\_amenities VALUES(234, 'fireplace'); |
|  | INSERT INTO rental\_amenities VALUES(345, 'kitchen'); |
|  | INSERT INTO rental\_amenities VALUES(345, 'pool'); |
|  | INSERT INTO rental\_amenities VALUES(456, 'pool'); |
|  |  |
|  | WITH rental\_amenities AS( |
|  | SELECT |
|  | rental\_id, |
|  | GROUP\_CONCAT(amenity ORDER BY amenity) AS amenities |
|  | FROM |
|  | rental\_amenities |
|  | GROUP BY |
|  | rental\_id |
|  | ) |
|  | SELECT |
|  | count(distinct amenities) AS matching\_airbnb |
|  | FROM |
|  | rental\_amenities |
|  | GROUP BY |
|  | amenities |
|  | HAVING |
|  | COUNT(DISTINCT rental\_id) > 1 |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE rental\_amenities; |
|  |  |
|  |  |
|  | --Q153: |
|  |  |
|  | CREATE TABLE ad\_campaigns |
|  | ( |
|  | campaign\_id INT, |
|  | spend INT, |
|  | revenue DECIMAL(10,2), |
|  | advertiser\_id INT |
|  | ); |
|  |  |
|  | INSERT INTO ad\_campaigns VALUES(1, 5000, 7500, 3); |
|  | INSERT INTO ad\_campaigns VALUES(2, 1000, 900, 1); |
|  | INSERT INTO ad\_campaigns VALUES(3, 3000, 12000, 2); |
|  | INSERT INTO ad\_campaigns VALUES(1, 500, 2000, 4); |
|  | INSERT INTO ad\_campaigns VALUES(1, 100, 400, 4); |
|  |  |
|  | WITH campaign\_per\_advertiser AS( |
|  | SELECT |
|  | advertiser\_id, |
|  | SUM(spend) AS total\_spend, |
|  | SUM(revenue) AS total\_revenue |
|  | FROM |
|  | ad\_campaigns |
|  | GROUP BY |
|  | advertiser\_id |
|  | ) |
|  | SELECT |
|  | advertiser\_id, |
|  | ROUND(total\_revenue/total\_spend, 2) AS ROAS |
|  | FROM |
|  | campaign\_per\_advertiser |
|  | ORDER BY |
|  | advertiser\_id |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE ad\_campaigns; |
|  |  |
|  |  |
|  | --Q154: |
|  |  |
|  | CREATE TABLE employee\_pay |
|  | ( |
|  | employee\_id INT, |
|  | salary INT, |
|  | title VARCHAR(40) |
|  | ); |
|  |  |
|  | INSERT INTO employee\_pay VALUES(101, 80000, 'Data Analyst'); |
|  | INSERT INTO employee\_pay VALUES(102, 90000, 'Data Analyst'); |
|  | INSERT INTO employee\_pay VALUES(103, 100000, 'Data Analyst'); |
|  | INSERT INTO employee\_pay VALUES(104, 30000, 'Data Analyst'); |
|  | INSERT INTO employee\_pay VALUES(105, 120000, 'Data Scientist'); |
|  | INSERT INTO employee\_pay VALUES(106, 100000, 'Data Scientist'); |
|  | INSERT INTO employee\_pay VALUES(107, 80000, 'Data Scientist'); |
|  | INSERT INTO employee\_pay VALUES(108, 310000, 'Data Scientist'); |
|  |  |
|  | WITH employee\_pay\_avg\_by\_title AS( |
|  | SELECT |
|  | \*, |
|  | AVG(salary) OVER(PARTITION BY title) AS avg\_by\_title |
|  | FROM |
|  | employee\_pay |
|  | ) |
|  | SELECT |
|  | employee\_id, |
|  | salary, |
|  | CASE |
|  | WHEN salary > (2 \* avg\_by\_title) |
|  | THEN 'Overpaid' |
|  | WHEN salary < (avg\_by\_title/2) |
|  | THEN 'Underpaid' |
|  | END AS status |
|  | FROM |
|  | employee\_pay\_avg\_by\_title |
|  | WHERE |
|  | CASE |
|  | WHEN salary > (2 \* avg\_by\_title) OR salary < (avg\_by\_title/2) |
|  | THEN 1 |
|  | END = 1 |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE employee\_pay; |
|  |  |
|  |  |
|  | --Q155: |
|  |  |
|  | CREATE TABLE payments |
|  | ( |
|  | payer\_id INT, |
|  | recipient\_id INT, |
|  | amount INT |
|  | ); |
|  |  |
|  | INSERT INTO payments VALUES(101, 201, 30); |
|  | INSERT INTO payments VALUES(201, 101, 10); |
|  | INSERT INTO payments VALUES(101, 301, 20); |
|  | INSERT INTO payments VALUES(301, 101, 80); |
|  | INSERT INTO payments VALUES(201, 301, 70); |
|  |  |
|  | SELECT |
|  | ROUND(COUNT( |
|  | distinct p1.payer\_id, |
|  | p1.recipient\_id |
|  | )/2) AS unique\_relationships |
|  | FROM |
|  | payments p1 |
|  | JOIN payments p2 ON p1. payer\_id = p2.recipient\_id |
|  | AND p1.recipient\_id = p2.payer\_id; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE payments; |
|  |  |
|  |  |
|  | --Q156: |
|  |  |
|  | CREATE TABLE purchases |
|  | ( |
|  | user\_id INT, |
|  | product\_id INT, |
|  | quantity INT, |
|  | purchase\_date TIMESTAMP |
|  | ); |
|  |  |
|  | INSERT INTO purchases VALUES(536, 3223, 6, '2022-01-11 12:33:44'); |
|  | INSERT INTO purchases VALUES(827, 3585, 35, '2022-02-20 14:05:26'); |
|  | INSERT INTO purchases VALUES(536, 3223, 5, '2022-03-02 09:33:28'); |
|  | INSERT INTO purchases VALUES(536, 1435, 10, '2022-03-2 08:40:00'); |
|  | INSERT INTO purchases VALUES(827, 2452, 45, '2022-04-09 00:00:00'); |
|  |  |
|  | SELECT |
|  | COUNT(DISTINCT user\_id) AS repeat\_purchasers |
|  | FROM( |
|  | SELECT |
|  | user\_id |
|  | FROM |
|  | purchases |
|  | GROUP BY |
|  | user\_id, |
|  | product\_id |
|  | HAVING |
|  | COUNT(DISTINCT DATE\_FORMAT(purchase\_date,'%Y-%m-%d')) > 1 |
|  | ) repeat\_purchase\_product |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE purchases; |
|  |  |
|  |  |
|  | --Q157: |
|  |  |
|  | CREATE TABLE transactions |
|  | ( |
|  | transaction\_id INT, |
|  | type VARCHAR(20), |
|  | amount DECIMAL(10, 2), |
|  | transaction\_date TIMESTAMP |
|  | ); |
|  |  |
|  | INSERT INTO transactions VALUES(19153, 'deposit', 65.90, '2022-07-10 10:00:00'); |
|  | INSERT INTO transactions VALUES(53151, 'deposit', 178.55, '2022-07-08 10:00:00'); |
|  | INSERT INTO transactions VALUES(29776, 'withdrawal', 25.90, '2022-07-08 10:00:00'); |
|  | INSERT INTO transactions VALUES(16461, 'withdrawal', 45.99, '2022-07-08 10:00:00'); |
|  | INSERT INTO transactions VALUES(77134, 'deposit', 32.60, '2022-07-10 10:00:00'); |
|  |  |
|  | WITH daily\_balance AS ( |
|  | SELECT |
|  | DATE\_FORMAT(transaction\_date, '%Y-%m') AS transaction\_month, |
|  | DATE\_FORMAT(transaction\_date, '%m/%d/%Y') AS transaction\_date, |
|  | SUM(CASE |
|  | WHEN type = 'deposit' |
|  | THEN amount |
|  | WHEN type = 'withdrawal' |
|  | THEN amount \* (-1) |
|  | END) AS balance |
|  |  |
|  | FROM |
|  | transactions |
|  | GROUP BY |
|  | DATE\_FORMAT(transaction\_date, '%m/%d/%Y'), |
|  | DATE\_FORMAT(transaction\_date, '%Y-%m') |
|  | ) |
|  | SELECT |
|  | CONCAT(transaction\_date , ' 12:00:00') AS transaction\_date, |
|  | SUM(balance) OVER( PARTITION BY transaction\_month ORDER BY transaction\_date) AS balance |
|  | FROM |
|  | daily\_balance |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE transactions; |
|  |  |
|  |  |
|  | --Q158: |
|  |  |
|  | CREATE TABLE product\_spend |
|  | ( |
|  | category VARCHAR(20), |
|  | product VARCHAR(30), |
|  | user\_id INT, |
|  | spend DECIMAL(10, 2), |
|  | transaction\_date TIMESTAMP |
|  | ); |
|  |  |
|  | INSERT INTO product\_spend VALUES('appliance', 'refrigerator', 165, 246.00, '2021-12-26 12:00:00'); |
|  | INSERT INTO product\_spend VALUES('appliance', 'refrigerator', 123, 299.99, '2022-03-02 12:00:00'); |
|  | INSERT INTO product\_spend VALUES('appliance', 'washing machine', 123, 219.80, '2022-03-02 12:00:00'); |
|  | INSERT INTO product\_spend VALUES('electronics', 'vacuum', 178, 152.00, '2022-04-05 12:00:00'); |
|  | INSERT INTO product\_spend VALUES('electronics', 'wireless headset', 156, 249.90, '2022-07-08 12:00:00'); |
|  | INSERT INTO product\_spend VALUES('electronics', 'vacuum', 145, 189.00, '2022-07-15 12:00:00'); |
|  |  |
|  | WITH category\_product\_spend\_2022 AS( |
|  | SELECT |
|  | category, |
|  | product, |
|  | sum(spend) AS total\_spend, |
|  | DENSE\_RANK() OVER(PARTITION BY category ORDER BY sum(spend)) AS serial |
|  | FROM |
|  | product\_spend |
|  | WHERE |
|  | DATE\_FORMAT(transaction\_date, '%Y') = '2022' |
|  | GROUP BY |
|  | category, |
|  | product |
|  | ) |
|  | SELECT |
|  | category, |
|  | product, |
|  | total\_spend |
|  | FROM |
|  | category\_product\_spend\_2022 |
|  | WHERE |
|  | serial <= 2; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE product\_spend; |
|  |  |
|  |  |
|  | --Q159: |
|  |  |
|  | CREATE TABLE users |
|  | ( |
|  | user\_id INT, |
|  | signup\_date TIMESTAMP, |
|  | last\_login TIMESTAMP |
|  | ); |
|  |  |
|  | INSERT INTO users VALUES(1001, '2022-06-01 12:00:00', '2022-07-05 12:00:00'); |
|  | INSERT INTO users VALUES(1002, '2022-06-03 12:00:00', '2022-06-15 12:00:00'); |
|  | INSERT INTO users VALUES(1004, '2022-06-02 12:00:00', '2022-06-15 12:00:00'); |
|  | INSERT INTO users VALUES(1006, '2022-06-15 12:00:00', '2022-06-27 12:00:00'); |
|  | INSERT INTO users VALUES(1012, '2022-06-16 12:00:00', '2022-07-22 12:00:00'); |
|  |  |
|  | WITH users\_churn\_weekly AS ( |
|  | SELECT |
|  | \*, |
|  | CASE |
|  | WHEN DATEDIFF(last\_login, signup\_date) < 28 |
|  | THEN 'yes' |
|  | ELSE |
|  | 'no' |
|  | END AS churn, |
|  | DATEDIFF(last\_login, signup\_date) as diff, |
|  | WEEK(signup\_date) - WEEK('2022-05-28') AS week\_no |
|  | FROM |
|  | users |
|  | WHERE |
|  | DATE\_FORMAT(signup\_date, '%Y-%m') = '2022-06' |
|  | ) |
|  | SELECT |
|  | week\_no, |
|  | ROUND( |
|  | COUNT( |
|  | CASE |
|  | WHEN churn = 'yes' |
|  | THEN user\_id |
|  | END |
|  | )\*100.00/COUNT(\*) |
|  | , 2) AS churn\_rate |
|  | FROM |
|  | users\_churn\_weekly |
|  | GROUP BY |
|  | week\_no |
|  | ; |
|  |  |
|  | --drop tables |
|  |  |
|  | DROP TABLE users; |
|  |  |