SQL Assignments

Database and Table creation

CREATE DATABASE TestDB;

USE TestDB;

CREATE TABLE CITY

(

ID INT,

NAME VARCHAR(17),

COUNTRYCODE VARCHAR(3),

DISTRICT VARCHAR(20),

POPULATION INT

);

insert into CITY values(6, 'Rotterdam','NLD','Zuid-Holland', 593321);

insert into CITY values(3878, 'Scottsdale','USA','Arizona', 202705);

insert into CITY values(3965, 'Corona', 'USA', 'California', 124966);

insert into CITY values(3973,'Concord', 'USA', 'California', 121780);

insert into CITY values(3977, 'Cedar Rapids', 'USA', 'Iowa', 120758);

insert into CITY values(3982, 'Coral Springs','USA', 'Florida', 11754);

insert into CITY values(4054 ,'Fairfield', 'USA', 'California', 92256);

insert into CITY values(4058, 'Boulder', 'USA', 'Colorado', 91238);

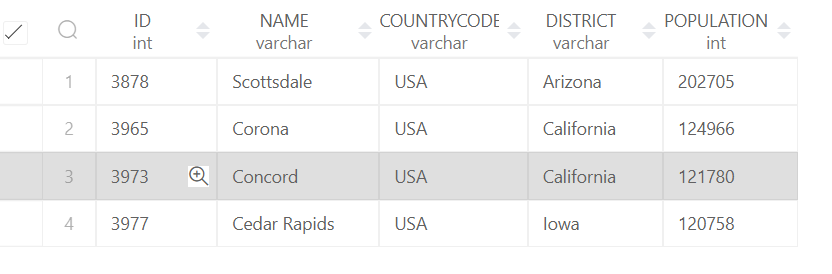
insert into CITY values(4061,'Fall River', 'USA','Massachusetts',90555);

Q1. Query all columns for all American cities in the CITY table with populations larger than 100000. The CountryCode for America is USA.

Solution

SELECT \* FROM CITY WHERE COUNTRYCODE = 'USA' AND POPULATION > 100000;

Output

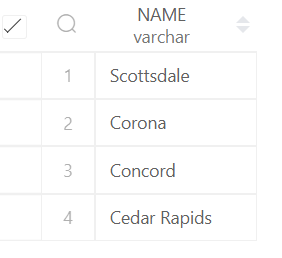


Q2. Query the NAME field for all American cities in the CITY table with populations larger than 120000. The CountryCode for America is USA.

Solution

SELECT NAME FROM CITY WHERE COUNTRYCODE = 'USA' AND POPULATION > 120000;

Output

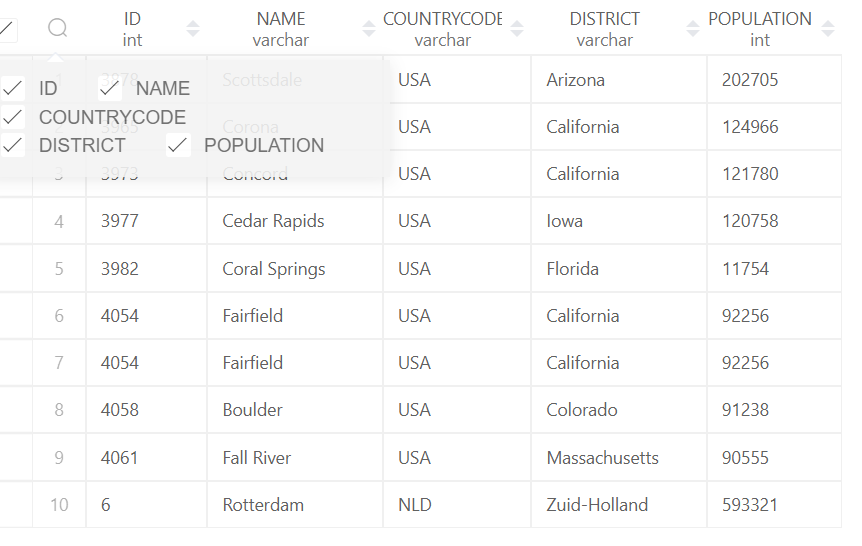


Q3. Query all columns (attributes) for every row in the CITY table.

Solution

SELECT \* FROM CITY;

Output



Q4. Query all columns for a city in CITY with the ID 1661.

Solution

SELECT \* FROM CITY WHERE ID =1661;

Output

No output

Q5. Query all attributes of every Japanese city in the CITY table. The COUNTRYCODE for Japan is JPN

Solution

SELECT \* FROM CITY WHERE COUNTRYCODE=’JPN’;

Q6. Query the names of all the Japanese cities in the CITY table. The COUNTRYCODE for Japan is JPN

Solution

SELECT NAME FROM CITY WHERE COUNTRYCODE=’JPN’;

Station Table

CREATE TABLE STATION

(

ID INT,

CITY VARCHAR(21),

STATE VARCHAR(2),

LAT\_N INT,

LONG\_W INT

);

insert into STATION values(794, 'Kissee Mills', 'MO', 139,73);

insert into STATION values(824, 'Loma Mar', 'CA', 48, 130);

insert into STATION values(603, 'Sandy Hook', 'CT', 72, 148);

insert into STATION values(478, 'Tipton', 'IN', 33, 97);

insert into STATION values(619, 'Arlington', 'CO', 75,92);

insert into STATION values(711, 'Turner', 'AR', 50, 101);

insert into STATION values(839, 'Slidell', 'LA', 85, 151);

insert into STATION values(411, 'Negreet', 'LA', 98, 105);

insert into STATION values(588, 'Glencoe', 'KY', 46, 136);

insert into STATION values(665, 'Chelsea', 'IA', 98, 59);

insert into STATION values(342,

'Chignik

Lagoon' ,'AK', 103, 153);

insert into STATION values(733, 'Pelahatchie', 'MS', 38,28);

insert into STATION values(441, 'Hanna City' ,'IL', 50, 136);

insert into STATION values(811, 'Dorrance', 'KS', 102, 121);

insert into STATION values(698, 'Albany' ,'CA', 49, 80);

insert into STATION values(325, 'Monument', 'KS', 70, 141);

insert into STATION values(414, 'Manchester','MD', 73,37);

insert into STATION values(113 ,'Prescott', 'IA', 39, 65);

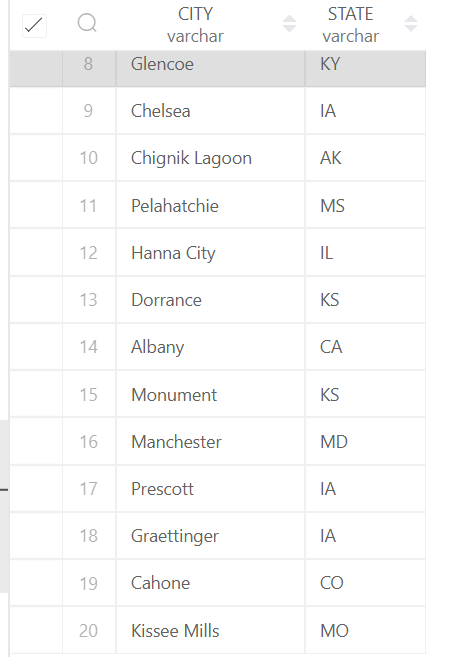
insert into STATION values(971, 'Graettinger', 'IA', 94, 150);

insert into STATION values(266, 'Cahone', 'CO', 116, 127);

Q7. Query a list of CITY and STATE from the STATION table. The STATION table is described as follows:

Solution

SELECT CITY, STATE FROM STATION;

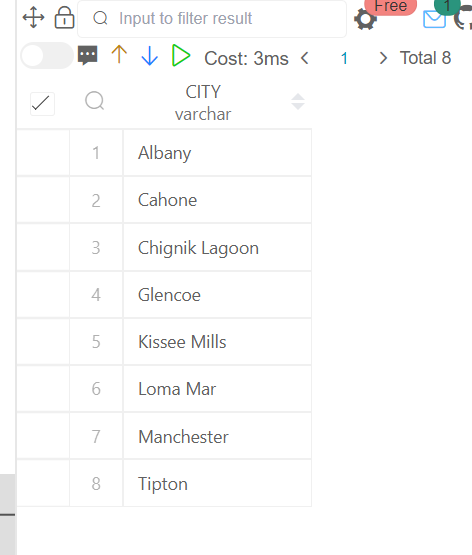


Q8. Query a list of CITY names from STATION for cities that have an even ID number. Print the results in any order, but exclude duplicates from the answer.

Solution

SELECT DISTINCT(CITY) FROM STATION WHERE ID%2 = 0

ORDER BY CITY

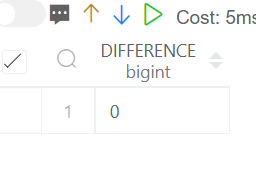


Q9. Find the difference between the total number of CITY entries in the table and the number of distinct CITY entries in the table.

Solution

SELECT COUNT(CITY) - COUNT(DISTINCT(CITY)) AS 'DIFFERENCE'

FROM STATION;



Q10. Query the two cities in STATION with the shortest and longest CITY names, as well as their respective lengths (i.e.: number of characters in the name). If there is more than one smallest or largest city, choose the one that comes first when ordered alphabetically.

Solution

(SELECT CITY, LENGTH(CITY) AS CITY\_LEN

FROM STATION

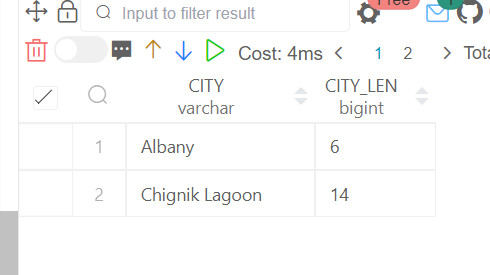
ORDER BY CITY\_LEN, CITY LIMIT 1)

UNION

(SELECT CITY, LENGTH(CITY) AS CITY\_LEN

FROM STATION

ORDER BY CITY\_LEN DESC, CITY LIMIT 1);



Q11. Query the list of CITY names starting with vowels (i.e., a, e, i, o, or u) from STATION. Your result cannot contain duplicates.

Solution

SELECT DISTINCT(CITY)

FROM

STATION

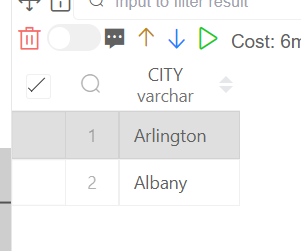
WHERE (CITY LIKE 'a%'

OR CITY LIKE 'e%'

OR CITY LIKE 'i%'

OR CITY LIKE 'o%'

OR CITY LIKE 'u%');



Q13. Query the list of CITY names from STATION that do not start with vowels. Your result cannot contain duplicates

Solution

SELECT DISTINCT(CITY)

FROM

STATION

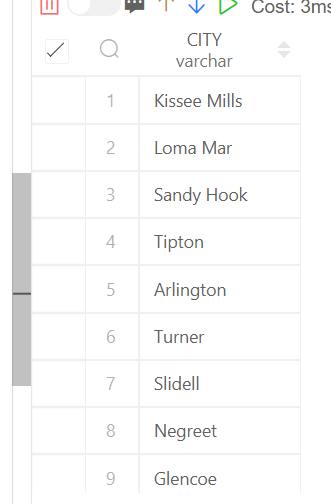
WHERE (CITY NOT LIKE 'a%'

AND CITY NOT LIKE 'e%'

AND CITY NOT LIKE 'i%'

AND CITY NOT LIKE 'o%'

AND CITY NOT LIKE 'u%');



Q14. Query the list of CITY names from STATION that do not end with vowels. Your result cannot contain duplicates.

Solution

SELECT DISTINCT(CITY)

FROM

STATION

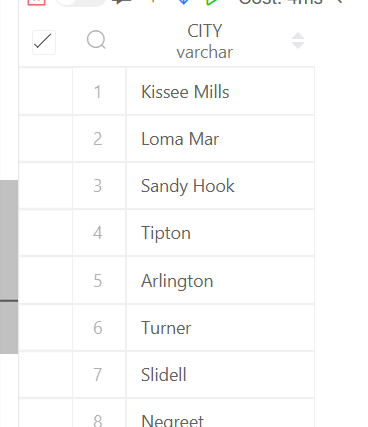
WHERE (CITY NOT LIKE '%a'

AND CITY NOT LIKE '%e'

AND CITY NOT LIKE '%i'

AND CITY NOT LIKE '%o'

AND CITY NOT LIKE '%u');



Q15. Query the list of CITY names from STATION that either do not start with vowels or do not end with vowels. Your result cannot contain duplicates.

Solution

SELECT DISTINCT(CITY)

FROM

STATION

WHERE (CITY NOT LIKE '%a'

AND CITY NOT LIKE '%e'

AND CITY NOT LIKE '%i'

AND CITY NOT LIKE '%o'

AND CITY NOT LIKE '%u'

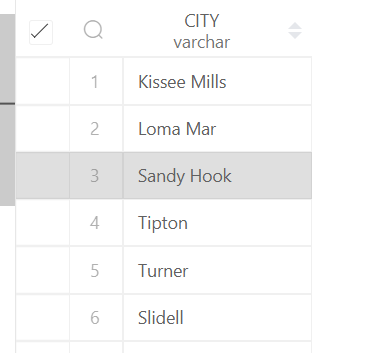
AND CITY NOT LIKE 'a%'

AND CITY NOT LIKE 'e%'

AND CITY NOT LIKE 'i%'

AND CITY NOT LIKE 'o%'

AND CITY NOT LIKE 'u%');



Q16. Query the list of CITY names from STATION that do not start with vowels and do not end with vowels. Your result cannot contain duplicates.

Solution

SELECT DISTINCT(CITY)

FROM

STATION

WHERE (CITY NOT LIKE '%a'

AND CITY NOT LIKE '%e'

AND CITY NOT LIKE '%i'

AND CITY NOT LIKE '%o'

AND CITY NOT LIKE '%u'

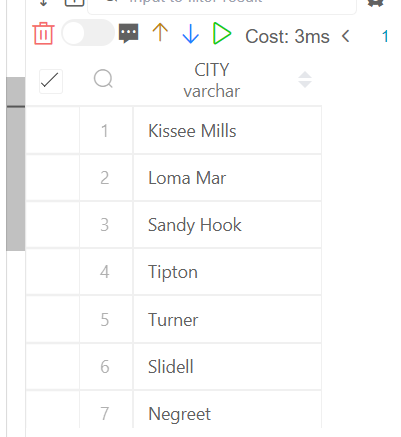
AND CITY NOT LIKE 'a%'

AND CITY NOT LIKE 'e%'

AND CITY NOT LIKE 'i%'

AND CITY NOT LIKE 'o%'

AND CITY NOT LIKE 'u%');



17.Write an SQL query that reports the products that were only sold in the first quarter of 2019. That is, between 2019-01-01 and 2019-03-31 inclusive.

Solution

Creating Tables

CREATE TABLE Product

(

product\_id INT,

product\_name VARCHAR(20),

unit\_price INT,

CONSTRAINT PK PRIMARY KEY (product\_id)

);

CREATE TABLE Sales

(

seller\_id INT,

product\_id INT,

buyer\_id INT,

sale\_date DATE,

quantity INT,

price INT,

constraint fk Foreign Key (product\_id) REFERENCES Product(product\_id)

);

insert into Product values(1, 'S8', 1000);

insert into Product values(2, 'G4', 800);

insert into Product values(3, 'iPhone', 1400);

insert into Sales values(1, 1, 1, '2019-01-21', 2, 2000);

insert into Sales values(1, 2, 2, '2019-02-17', 1, 800);

insert into Sales values(2, 2, 3, '2019-06-02', 1, 800);

insert into Sales values(3, 3, 4, '2019-05-13', 2, 2800);

18. Write an SQL query to find all the authors that viewed at least one of their own articles. Return the result table sorted by id in ascending order

Solution

CREATE DATABASE TestDB;

USE TestDB;

CREATE TABLE Views

(

article\_id INT,

author\_id INT,

viewer\_id INT,

view\_date DATE

);

insert into Views values(1, 3, 5,'2019-08-01');

insert into Views values(1, 3, 6,'2019-08-02');

insert into Views values(2, 7, 7, '2019-08-01');

insert into Views values(2, 7, 6, '2019-08-02');

insert into Views values(4, 7, 1, '2019-07-22');

insert into Views values(3, 4, 4, '2019-07-21');

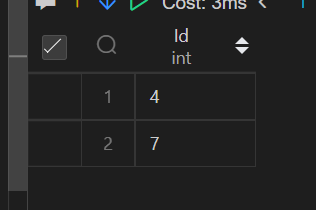
insert into Views values(3, 4, 4, '2019-07-21');

SELECT DISTINCT(A.author\_id) AS Id

FROM Views A, Views B

WHERE A.author\_id = B.viewer\_id

ORDER BY A.author\_id;



19. Write an SQL query to find the percentage of immediate orders in the table, rounded to 2 decimal places.

Solution

CREATE TABLE Delivery

(

delivery\_id INT,

customer\_id INT,

order\_date DATE,

customer\_pref\_delivery\_date DATE,

CONSTRAINT PK PRIMARY KEY (delivery\_id)

);

insert into Delivery values(1, 1, '2019-08-01', '2019-08-02');

insert into Delivery values(2, 5, '2019-08-02', '2019-08-02');

insert into Delivery values(3, 1, '2019-08-11', '2019-08-11');

insert into Delivery values(4, 3, '2019-08-24', '2019-08-26');

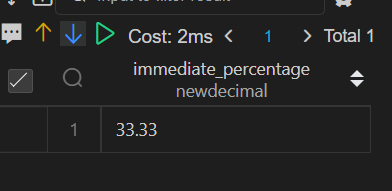
insert into Delivery values(5, 4, '2019-08-21', '2019-08-22');

insert into Delivery values(6, 2, '2019-08-11', '2019-08-13');

SELECT ROUND((SELECT COUNT(\*) FROM Delivery WHERE order\_date = customer\_pref\_delivery\_date) \* 100/ COUNT(\*),2)

AS 'immediate\_percentage'

FROM Delivery;



20. Not able to understand the question

21. Write an SQL query to find the team size of each of the employees.

Solution

Proper data set is not available

22. Write an SQL query to find the type of weather in each country for November 2019. The type of weather is: ● Cold if the average weather\_state is less than or equal 15, ● Hot if the average weather\_state is greater than or equal to 25, and ● Warm otherwise. Return result table in any order.

Solution (Not able to do it)

CREATE TABLE Countries

(

country\_id INT,

country\_name VARCHAR(17),

CONSTRAINT PK PRIMARY KEY (country\_id)

);

CREATE TABLE Weather

(

country\_id INT,

weather\_state INT,

day DATE,

CONSTRAINT PK PRIMARY KEY (country\_id, day)

);

insert into Countries values(2, 'USA');

insert into Countries values(3, 'Australia');

insert into Countries values(7, 'Peru');

insert into Countries values(5, 'China');

insert into Countries values(8, 'Morocco');

insert into Countries values(9, 'Spain');

insert into Weather values(2,15, '2019-11-01');

insert into Weather values(2,12, '2019-10-28');

insert into Weather values(2,12, '2019-10-27');

insert into Weather values(3,-2, '2019-11-10');

insert into Weather values(3,0, '2019-11-11');

insert into Weather values(3,3, '2019-11-12');

insert into Weather values(5,16, '2019-11-07');

insert into Weather values(5,18, '2019-11-09');

insert into Weather values(5,21, '2019-11-23');

insert into Weather values(7,25, '2019-11-28');

insert into Weather values(7,22, '2019-12-01');

insert into Weather values(7,20, '2019-12-02');

insert into Weather values(8,25, '2019-11-05');

insert into Weather values(8,27, '2019-11-15');

insert into Weather values(8,31, '2019-11-25');

insert into Weather values(9,7, '2019-10-23');

insert into Weather values(9,3, '2019-12-23');

select C.country\_name,

case

when (SELECT country\_id, AVG(weather\_state) FROM Weather WHERE day BETWEEN '2019-11-01' AND '2019-11-30'

GROUP BY country\_id) < = 15 then 'Cold'

when (SELECT country\_id, AVG(weather\_state) FROM Weather WHERE day BETWEEN '2019-11-01' AND '2019-11-30'

GROUP BY country\_id) > = 25 then 'Hot'

else 'Warm'

end as weather\_type

Countries C INNER JOIN Weather W

ON C.country\_id = W.country\_id;

23. Write an SQL query to find the average selling price for each product. average\_price should be rounded to 2 decimal places

Solution

CREATE TABLE Prices

(

product\_id INT,

start\_date DATE,

end\_date DATE,

price INT,

CONSTRAINT PK PRIMARY KEY (product\_id, start\_date, end\_date)

);

CREATE TABLE UnitsSold

(

product\_id INT,

purchase\_date DATE,

units INT

);

insert into Prices values(1, '2019-02-17', '2019-02-28',5);

insert into Prices values(1, '2019-03-01', '2019-03-22',20);

insert into Prices values(2 ,'2019-02-01' ,'2019-02-20' ,15);

insert into Prices values(2, '2019-02-21', '2019-03-31' ,3);

insert into UnitsSold values(1 ,'2019-02-25', 100);

insert into UnitsSold values(1, '2019-03-01', 15);

insert into UnitsSold values(2, '2019-02-10' ,200);

insert into UnitsSold values(2 ,'2019-03-22', 30);

SELECT \* FROM UnitsSold;

24. Write an SQL query to report the first login date for each player. Return the result table in any order

Solution

CREATE TABLE Activity

(

player\_id INT,

device\_id INT,

event\_date DATE,

games\_played INT,

CONSTRAINT PK PRIMARY KEY (player\_id, event\_date)

);

insert into Activity values(1, 2, '2016-03-01', 5);

insert into Activity values(1, 2, '2016-05-02', 6);

insert into Activity values(2, 3, '2017-06-25', 1);

insert into Activity values(3, 1, '2016-03-02', 0);

insert into Activity values(3, 4, '2018-07-03', 5);

select

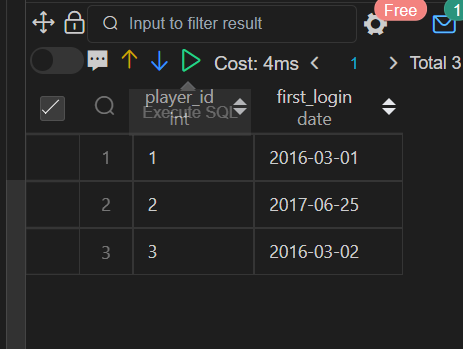
tmp.player\_id, tmp.event\_date as first\_login

from (select \*,

row\_number() over(partition by player\_id order by event\_date ) as row\_num

from Activity) tmp

where tmp.row\_num = 1;



25. Write an SQL query to report the device that is first logged in for each player. Return the result table in any order.

Solution

select

tmp.player\_id, tmp.device\_id

from (select \*,

row\_number() over(partition by player\_id order by event\_date desc) as row\_num

from Activity) tmp

where tmp.row\_num = 1;

