//Creating database

**CREATE DATABASE JALAASSIGNMENTSDATABASE**;

//Using the database

**USE JALAASSIGNMENTSDATABASE;**

//Create Tables

CREATE TABLE SALESPEOPLE (

SNUM INT PRIMARY KEY,

SNAME VARCHAR(50),

CITY VARCHAR(50),

COMM DECIMAL(4,2)

);

show tables;

show columns from SALESPEOPLE;

CREATE TABLE CUST (

CNUM INT PRIMARY KEY,

CNAME VARCHAR(50),

CITY VARCHAR(50),

RATING INT,

SNUM INT,

FOREIGN KEY (SNUM) REFERENCES SALESPEOPLE(SNUM)

);

show tables;

show columns from CUST;

CREATE TABLE ORDERS (

ONUM INT PRIMARY KEY,

AMT DECIMAL(10,2),

ODATE DATE,

CNUM INT,

SNUM INT,

FOREIGN KEY (CNUM) REFERENCES CUST(CNUM),

FOREIGN KEY (SNUM) REFERENCES SALESPEOPLE(SNUM)

);

show tables;

show columns from ORDERS;

SET SQL\_SAFE\_UPDATES = 0;

DELETE FROM SALESPEOPLE WHERE SNUM IN (1001, 1002, 1003, 1004, 1007);

SET SQL\_SAFE\_UPDATES = 1;

SET SQL\_SAFE\_UPDATES = 0;

DELETE FROM CUST WHERE SNUM IN (1001, 1002, 1003, 1004, 1007);

DELETE FROM SALESPEOPLE WHERE SNUM IN (1001, 1002, 1003, 1004, 1007);

SET SQL\_SAFE\_UPDATES = 1;

INSERT INTO SALESPEOPLE (SNUM, SNAME, CITY, COMM) VALUES

(1001, 'Peel', 'London', 0.12),

(1002, 'Serres', 'San Jose', 0.13),

(1003, 'Axelrod', 'New York', 0.10),

(1004, 'Motika', 'London', 0.11),

(1007, 'Rafkin', 'Barcelona', 0.15);

SELECT \* FROM SALESPEOPLE;

SET SQL\_SAFE\_UPDATES = 0;

DELETE FROM ORDERS WHERE CNUM IN (2001, 2002, 2003, 2004, 2006, 2007, 2008);

DELETE FROM CUST WHERE CNUM IN (2001, 2002, 2003, 2004, 2006, 2007, 2008);

INSERT INTO CUST (CNUM, CNAME, CITY, RATING, SNUM) VALUES

(2001, 'Hoffman', 'London', 100, 1001),

(2002, 'Giovanni', 'Rome', 200, 1003),

(2003, 'Liu', 'San Jose', 300, 1002),

(2004, 'Grass', 'Berlin', 100, 1002),

(2006, 'Clemens', 'London', 300, 1007),

(2007, 'Pereira', 'Rome', 100, 1004),

(2008, 'James', 'London', 200, 1007);

SELECT \* FROM CUST;

DELETE FROM ORDERS WHERE ONUM IN (3001, 3002, 3003, 3005, 3006, 3007, 3008, 3009, 3010);

INSERT INTO ORDERS (ONUM, AMT, ODATE, CNUM, SNUM) VALUES

(3001, 18.69, '1994-10-03', 2008, 1007),

(3002, 1900.10, '1994-10-03', 2007, 1004),

(3003, 767.19, '1994-10-03', 2001, 1001),

(3005, 5160.45, '1994-10-03', 2003, 1002),

(3006, 1098.16, '1994-10-04', 2008, 1007),

(3007, 75.75, '1994-10-05', 2004, 1002),

(3008, 4723.00, '1994-10-05', 2006, 1001),

(3009, 2500.00, '1994-10-06', 2002, 1003),

(3010, 1290.50, '1994-10-06', 2001, 1001);

SELECT \* FROM ORDERS;

**Queries**

-- 1. Display snum,sname,city and comm of all salespeople.

SELECT snum, sname, city, comm FROM SALESPEOPLE;

-- 2. Display all snum without duplicates from all orders.

SELECT DISTINCT snum FROM ORDERS;

-- 3. Display names and commissions of all salespeople in London.

SELECT sname, comm FROM SALESPEOPLE WHERE city = 'London';

-- 4. All customers with rating of 100.

SELECT \* FROM CUST WHERE rating = 100;

-- 5. Produce orderno, amount and date from all rows in the order table.

SELECT onum, amt, odate FROM ORDERS;

-- 6. All customers in San Jose, who have rating more than 200.

SELECT \* FROM CUST WHERE city = 'San Jose' AND rating > 200;

-- 7. All customers who were either located in San Jose or had a rating above 200.

SELECT \* FROM CUST WHERE city = 'San Jose' OR rating > 200;

-- 8. All orders for more than $1000.

SELECT \* FROM ORDERS WHERE amt > 1000;

-- 9. Names and cities of all salespeople in London with commission above 0.10.

SELECT sname, city FROM SALESPEOPLE WHERE city = 'London' AND comm > 0.10;

-- 10. All customers excluding those with rating <= 100 unless they are located in Rome.

SELECT \* FROM CUST WHERE rating > 100 OR city = 'Rome';

-- 11. All salespeople either in Barcelona or in London.

SELECT \* FROM SALESPEOPLE WHERE city IN ('London', 'Barcelona');

-- 12. All salespeople with commission between 0.10 and 0.12. (Boundary values should be excluded)

SELECT \* FROM SALESPEOPLE WHERE comm > 0.10 AND comm < 0.12;

-- 13. All customers with NULL values in city column.

SELECT \* FROM CUST WHERE city IS NULL;

-- 14. All orders taken on Oct 3rd and Oct 4th, 1994.

SELECT \* FROM ORDERS WHERE odate IN ('03-OCT-94', '04-OCT-94');

-- 15. All customers serviced by Peel or Motika.

SELECT \* FROM CUST WHERE snum IN (

SELECT snum FROM SALESPEOPLE WHERE sname IN ('Peel', 'Motika')

);

-- 16. All customers whose names begin with a letter from A to B.

SELECT \* FROM CUST WHERE cname LIKE 'A%' OR cname LIKE 'B%';

-- 17. All orders except those with 0 or NULL value in amt field.

SELECT \* FROM ORDERS WHERE amt IS NOT NULL AND amt <> 0;

-- 18. Count the number of salespeople currently listing orders in the order table.

SELECT COUNT(DISTINCT snum) FROM ORDERS;

-- 19. Largest order taken by each salesperson, datewise.

SELECT snum, odate, MAX(amt) AS largest\_order

FROM ORDERS

GROUP BY snum, odate;

-- 20. Largest order taken by each salesperson with order value more than $3000.

SELECT snum, MAX(amt) AS largest\_order

FROM ORDERS

WHERE amt > 3000

GROUP BY snum;

-- 21. Which day had the highest total amount ordered.

SELECT odate, SUM(amt) AS total\_amount

FROM orders

GROUP BY odate

ORDER BY total\_amount DESC

LIMIT 1;

-- 22. Count all orders for Oct 3rd.

SELECT COUNT(\*) FROM ORDERS WHERE odate = '03-OCT-94';

-- 23. Count the number of different non-NULL city values in customers table.

SELECT COUNT(DISTINCT city) FROM CUST WHERE city IS NOT NULL;

-- 24. Select each customer’s smallest order.

SELECT cnum, MIN(amt) AS smallest\_order

FROM ORDERS

GROUP BY cnum;

-- 25. First customer in alphabetical order whose name begins with G.

SELECT \*

FROM cust

WHERE cname LIKE 'G%'

ORDER BY cname ASC

LIMIT 1;

-- 26. Get the output like “For dd/mm/yy there are \_\_\_ orders.”

SELECT 'For ' || odate || ' there are ' || COUNT(\*) || ' orders' AS summary

FROM ORDERS

GROUP BY odate;

-- 27. Assume that each salesperson has a 12% commission. Produce order no., salesperson no., and amount of salesperson’s commission for that order.

SELECT onum, snum, amt \* 0.12 AS commission

FROM ORDERS;

-- 28. Find highest rating in each city. Output: For the city (city), the highest rating is: (rating).

SELECT city, MAX(rating) AS highest\_rating

FROM CUST

GROUP BY city;

-- 29. Display the totals of orders for each day and place the results in descending order.

SELECT odate, SUM(amt) AS total

FROM ORDERS

GROUP BY odate

ORDER BY total DESC;

-- 30. All combinations of salespeople and customers who shared a city.

SELECT S.sname, C.cname, S.city

FROM SALESPEOPLE S

JOIN CUST C ON S.city = C.city;

-- 31. Name of all customers matched with the salespeople serving them.

SELECT C.cname, S.sname

FROM CUST C

JOIN SALESPEOPLE S ON C.snum = S.snum;

-- 32. List each order number followed by the name of the customer who made the order.

SELECT O.onum, C.cname

FROM ORDERS O

JOIN CUST C ON O.cnum = C.cnum;

-- 33. Names of salesperson and customer for each order after the order number.

SELECT O.onum, S.sname, C.cname

FROM ORDERS O

JOIN SALESPEOPLE S ON O.snum = S.snum

JOIN CUST C ON O.cnum = C.cnum;

-- 34. Produce all customers serviced by salespeople with a commission above 12%.

SELECT \* FROM CUST

WHERE snum IN (

SELECT snum FROM SALESPEOPLE WHERE comm > 0.12

);

-- 35. Calculate the amount of the salesperson’s commission on each order with a rating above 100.

SELECT O.onum, O.amt \* S.comm AS commission

FROM ORDERS O

JOIN CUST C ON O.cnum = C.cnum

JOIN SALESPEOPLE S ON O.snum = S.snum

WHERE C.rating > 100;

-- 36. Find all pairs of customers having the same rating.

SELECT A.cname, B.cname, A.rating

FROM CUST A, CUST B

WHERE A.rating = B.rating AND A.cnum < B.cnum;

-- 37. Find all pairs of customers having the same rating, each pair coming once only.

SELECT DISTINCT A.cname AS customer1, B.cname AS customer2, A.rating

FROM CUST A

JOIN CUST B ON A.rating = B.rating AND A.cnum < B.cnum;

-- 38. Policy is to assign three salespeople to each customer. Display all such combinations.

SELECT C.cname, S1.sname, S2.sname, S3.sname

FROM CUST C

CROSS JOIN SALESPEOPLE S1

CROSS JOIN SALESPEOPLE S2

CROSS JOIN SALESPEOPLE S3

WHERE S1.snum < S2.snum AND S2.snum < S3.snum;

-- 39. Display all customers located in cities where salesman Serres has customer.

SELECT \* FROM CUST

WHERE city IN (

SELECT city FROM CUST WHERE snum = (SELECT snum FROM SALESPEOPLE WHERE sname = 'Serres')

);

-- 40. Find all pairs of customers served by single salesperson.

SELECT A.cname, B.cname, A.snum

FROM CUST A

JOIN CUST B ON A.snum = B.snum AND A.cnum < B.cnum;

-- 41. Produce all pairs of salespeople living in the same city. Exclude combinations of salespeople with themselves as well as duplicates with the order reversed.

SELECT A.sname, B.sname, A.city

FROM SALESPEOPLE A

JOIN SALESPEOPLE B ON A.city = B.city AND A.snum < B.snum;

-- 42. Produce all pairs of orders by given customer, name that customer and eliminate duplicates.

SELECT DISTINCT O1.onum, O2.onum, C.cname

FROM ORDERS O1

JOIN ORDERS O2 ON O1.cnum = O2.cnum AND O1.onum < O2.onum

JOIN CUST C ON O1.cnum = C.cnum;

-- 43. Produce names and cities of all customers with the same rating as Hoffman.

SELECT cname, city

FROM CUST

WHERE rating = (SELECT rating FROM CUST WHERE cname = 'Hoffman');

-- 44. Extract all the orders of Motika.

SELECT \* FROM ORDERS

WHERE snum = (SELECT snum FROM SALESPEOPLE WHERE sname = 'Motika');

-- 45. All orders credited to the same salesperson who services Hoffman.

SELECT \* FROM ORDERS

WHERE snum = (SELECT snum FROM CUST WHERE cname = 'Hoffman');

-- 46. All orders that are greater than the average for Oct 4.

SELECT \* FROM ORDERS

WHERE odate = '04-OCT-94' AND amt > (

SELECT AVG(amt) FROM ORDERS WHERE odate = '04-OCT-94'

);

-- 47. Find average commission of salespeople in London.

SELECT AVG(comm) FROM SALESPEOPLE WHERE city = 'London';

-- 48. Find all orders attributed to salespeople servicing customers in London.

SELECT \* FROM ORDERS

WHERE cnum IN (SELECT cnum FROM CUST WHERE city = 'London');

-- 49. Extract commissions of all salespeople servicing customers in London.

SELECT DISTINCT comm FROM SALESPEOPLE

WHERE snum IN (SELECT snum FROM CUST WHERE city = 'London');

-- 50. Find all customers whose cnum is 1000 above the snum of Serres.

SELECT \* FROM CUST

WHERE cnum = (SELECT snum FROM SALESPEOPLE WHERE sname = 'Serres') + 1000;

-- 51. Count the customers with rating above San Jose’s average.

SELECT COUNT(\*) FROM CUST

WHERE rating > (SELECT AVG(rating) FROM CUST WHERE city = 'San Jose');

-- 52. Obtain all orders for the customer named Cisnerous (without using cnum).

SELECT \* FROM ORDERS

WHERE cnum = (SELECT cnum FROM CUST WHERE cname = 'Cisnerous');

-- 53. Produce the names and rating of all customers who have above average orders.

SELECT cname, rating

FROM CUST

WHERE cnum IN (

SELECT cnum FROM ORDERS

GROUP BY cnum

HAVING AVG(amt) > (SELECT AVG(amt) FROM ORDERS)

);

-- 54. Find total amount in orders for each salesperson whose total is greater than the amount of the largest order in the table.

SELECT snum, SUM(amt) AS total\_amt

FROM ORDERS

GROUP BY snum

HAVING SUM(amt) > (SELECT MAX(amt) FROM ORDERS);

-- 55. Find all customers with orders on 3rd Oct.

SELECT \* FROM ORDERS WHERE odate = '03-OCT-94';

-- 56. Find names and numbers of all salespeople who have more than one customer.

SELECT snum, COUNT(\*)

FROM CUST

GROUP BY snum

HAVING COUNT(\*) > 1;

-- 57. Check if the correct salesperson was credited with each sale.

SELECT \* FROM ORDERS O

WHERE NOT EXISTS (

SELECT 1 FROM CUST C WHERE C.cnum = O.cnum AND C.snum = O.snum

-- 58. Find all orders with above average amounts for their customers.

SELECT \*

FROM orders o

WHERE o.amt > (

SELECT AVG(o2.amt)

FROM orders o2

WHERE o2.cnum = o.cnum

);

-- 59. Find sums of amounts from order table grouped by date, eliminating those dates where the sum was not at least 2000 above the max amount.

SELECT odate, SUM(amt) AS total\_amt

FROM ORDERS

GROUP BY odate

HAVING SUM(amt) >= (

SELECT MAX(amt) FROM ORDERS

) + 2000;

-- 60. Find names and numbers of all customers with ratings equal to the max for their city.

SELECT cnum, cname

FROM CUST A

WHERE rating = (

SELECT MAX(rating) FROM CUST B WHERE A.city = B.city

);

-- 61. Find all salespeople who have customers in their cities whom they don’t service (JOIN and Correlated Subquery).

SELECT \* FROM SALESPEOPLE S

WHERE EXISTS (

SELECT 1 FROM CUST C

WHERE C.city = S.city AND C.snum <> S.snum

);

-- 62. Extract cnum, cname and city from customer table if and only if one or more customers are located in San Jose.

SELECT cnum, cname, city FROM CUST

WHERE EXISTS (SELECT 1 FROM CUST WHERE city = 'San Jose');

-- 63. Find salespeople numbers who have multiple customers.

SELECT snum FROM CUST GROUP BY snum HAVING COUNT(\*) > 1;

-- 64. Find salesperson number, name and city who have multiple customers.

SELECT snum, sname, city FROM SALESPEOPLE

WHERE snum IN (SELECT snum FROM CUST GROUP BY snum HAVING COUNT(\*) > 1);

-- 65. Find salespeople who serve only one customer.

SELECT snum FROM SALESPEOPLE

WHERE snum IN (SELECT snum FROM CUST GROUP BY snum HAVING COUNT(\*) = 1);

-- 66. Extract rows of all salespeople with more than one current order.

SELECT snum FROM ORDERS GROUP BY snum HAVING COUNT(\*) > 1;

-- 67. Find all salespeople who have customers with a rating of 300. (Use EXISTS)

SELECT \* FROM SALESPEOPLE S

WHERE EXISTS (

SELECT 1 FROM CUST C WHERE C.rating = 300 AND C.snum = S.snum

);

-- 68. Find all salespeople who have customers with a rating of 300. (Use JOIN)

SELECT DISTINCT S.\*

FROM SALESPEOPLE S

JOIN CUST C ON S.snum = C.snum

WHERE C.rating = 300;

-- 69. Select all salespeople with customers located in their cities who are not assigned to them. (Use EXISTS)

SELECT \* FROM SALESPEOPLE S

WHERE EXISTS (

SELECT 1 FROM CUST C

WHERE C.city = S.city AND C.snum <> S.snum

);

-- 70. Extract customers assigned to a salesperson who has at least one other customer with orders in the order table.

SELECT \* FROM CUST C

WHERE EXISTS (

SELECT 1 FROM ORDERS O

WHERE O.cnum <> C.cnum AND O.snum = C.snum

);

-- 71. Find salespeople with customers located in their cities (Using ANY and IN).

SELECT \* FROM SALESPEOPLE

WHERE city IN (SELECT city FROM CUST)

OR city = ANY (SELECT city FROM CUST);

-- 72. Find all salespeople for whom there are customers that follow them in alphabetical order. (Using ANY and EXISTS)

SELECT \* FROM SALESPEOPLE S

WHERE EXISTS (

SELECT 1 FROM CUST C

WHERE C.sname > S.sname

);

-- 73. Select customers who have a greater rating than any customer in Rome.

SELECT \* FROM CUST

WHERE rating > ANY (SELECT rating FROM CUST WHERE city = 'Rome');

-- 74. Select all orders with amounts greater than at least one of the orders from Oct 6.

SELECT \* FROM ORDERS

WHERE amt > ANY (SELECT amt FROM ORDERS WHERE odate = '06-OCT-94');

-- 75. Find all orders with amounts smaller than any amount for a customer in San Jose. (Using ANY and without ANY)

SELECT \* FROM ORDERS

WHERE amt < ALL (

SELECT amt FROM ORDERS

WHERE cnum IN (SELECT cnum FROM CUST WHERE city = 'San Jose')

);

-- 76. Select customers with ratings higher than every customer in Paris. (Using ALL and NOT EXISTS)

SELECT \* FROM CUST

WHERE rating > ALL (SELECT rating FROM CUST WHERE city = 'Paris');

-- 77. Select all customers with ratings equal to or greater than ANY of Serres.

SELECT \* FROM CUST

WHERE rating >= ANY (SELECT rating FROM SALESPEOPLE WHERE sname = 'Serres');

-- 78. Find all salespeople who have no customers located in their city. (Using ANY and ALL)

SELECT \* FROM SALESPEOPLE

WHERE snum NOT IN (

SELECT DISTINCT snum FROM CUST WHERE city = SALESPEOPLE.city

);

-- 79. Find all orders with amounts greater than any for customers in London.

SELECT \* FROM ORDERS

WHERE amt > ANY (

SELECT amt FROM ORDERS WHERE cnum IN (SELECT cnum FROM CUST WHERE city = 'London')

);

-- 80. Find all salespeople and customers located in London.

SELECT \* FROM SALESPEOPLE WHERE city = 'London'

UNION

SELECT \* FROM CUST WHERE city = 'London';

-- 81. For every salesperson, show dates on which highest and lowest orders were brought.

SELECT snum, odate, MAX(amt) AS max\_order, MIN(amt) AS min\_order

FROM ORDERS

GROUP BY snum, odate;

-- 82. List all salespeople and indicate those with and without customers in their cities.

SELECT sname, city,

CASE

WHEN EXISTS (SELECT 1 FROM CUST WHERE CUST.city = SALESPEOPLE.city AND CUST.snum = SALESPEOPLE.snum) THEN 'Has customer'

ELSE 'No customer'

END AS status

FROM SALESPEOPLE;

-- 83. Append strings to fields indicating whether a salesperson was matched to a customer in their city.

SELECT snum, sname || ' - ' ||

CASE

WHEN EXISTS (SELECT 1 FROM CUST WHERE CUST.snum = SALESPEOPLE.snum AND CUST.city = SALESPEOPLE.city) THEN 'Matched'

ELSE 'Not Matched'

END AS status

FROM SALESPEOPLE;

-- 84. Union of two queries showing names, cities, and ratings of customers, labeled as ‘High Rating’ or ‘Low Rating’.

SELECT cname, city, rating, 'High Rating' AS rating\_status

FROM CUST WHERE rating >= 200

UNION

SELECT cname, city, rating, 'Low Rating'

FROM CUST WHERE rating < 200;

-- 85. Names and numbers of salespeople and customers with more than one order, in alphabetical order.

SELECT sname, snum FROM SALESPEOPLE

WHERE snum IN (SELECT snum FROM ORDERS GROUP BY snum HAVING COUNT(\*) > 1)

UNION

SELECT cname, cnum FROM CUST

WHERE cnum IN (SELECT cnum FROM ORDERS GROUP BY cnum HAVING COUNT(\*) > 1)

ORDER BY sname;

-- 86. Union of three queries: (1) snums of salespeople in San Jose, (2) cnums of customers in San Jose, (3) onums of orders on Oct 3.

SELECT snum FROM SALESPEOPLE WHERE city = 'San Jose'

UNION

SELECT cnum FROM CUST WHERE city = 'San Jose'

UNION ALL

SELECT onum FROM ORDERS WHERE odate = '03-OCT-94';

-- 87. Salespeople in London who had at least one customer there.

SELECT \* FROM SALESPEOPLE

WHERE city = 'London' AND snum IN (

SELECT snum FROM CUST WHERE city = 'London'

);

-- 88. Salespeople in London who did not have customers there.

SELECT \* FROM SALESPEOPLE

WHERE city = 'London' AND snum NOT IN (

SELECT snum FROM CUST WHERE city = 'London'

);

-- 89. Match salespeople to their customers without excluding unassigned salespeople. (Use OUTER JOIN and UNION)

SELECT S.snum, S.sname, C.cname

FROM SALESPEOPLE S

LEFT JOIN CUST C ON S.snum = C.snum

UNION

SELECT S.snum, S.sname, NULL

FROM SALESPEOPLE S

WHERE S.snum NOT IN (SELECT DISTINCT snum FROM CUST);