-- Create a Database with your name. CREATE DATABASE YourNameDB;

-- Use the database USE YourName;

-- 3. create salesman table. create table salesman (snum int(4) unique, sname varchar(15), city varchar(10), commission int(3));

-- 4. Insert data into the salesman table. INSERT INTO salesman (snum, sname, city, commission) VALUES (1001, 'Piyush', 'London', 12);

INSERT INTO salesman (snum, sname, city, commission) VALUES (1002, 'Niraj', 'Surat', 13);

-- 5. Create customer table. create table customer (cnum int(4) unique, cname varchar(15), city varchar(10), rating int(3), snum int(4) references salesman(snum));

-- 6. Insert data into customer table. INSERT INTO customer (cnum, cname, city, rating, snum) VALUES (2001, 'Hardik', 'London', 100, 1001);

-- 7. create orders table. create table orders (onum int(4) primary key, amount int(6), odate date, cnum int(4) references customer(cnum), snum int(4) references salesman(snum));

-- 8. Insert data into orders table. INSERT INTO orders (onum, ammount, odate, cnum, snum) VALUES (3001, 18.69, '10/03/99', 2006, 1007);

-- Practicals -- 1. Produce the orders no, amount and date of all orderss. SELECT onum, ammount, odate FROM orders;

-- 2. Give all the information about all the customers with salesman number 1001. SELECT * FROM customer WHERE snum = 1001;

-- 3. Display the information in the sequence of city, sname, snum, and Commission. SELECT city, sname, snum, commission FROM salesman;

-- 4. List of snum of all salesmen with orders in orders table without duplicates. select DISTINCT snum from orders;

-- 5. List of all orders for more than Rs.1000. SELECT * FROM orders WHERE ammount > 1000;

-- 6. List out names and cities of all salesmen in London with commission above 10%. SELECT sname, city FROM salesman WHERE city = 'London' AND commission > 10;

-- 7. List all customers excluding those with rating <= 100 or they are located in Rome. SELECT * FROM customer WHERE NOT (rating <= 100 OR city = 'Rome');

-- 8. List all orders for more than Rs.1000 except the orders of snum, 1006 of 10/03/99. SELECT * FROM orders WHERE ammount > 1000 AND NOT (odate = '10/03/99' AND snum = 1006);

-- 9. List all orders taken on 10th March, April and June 1999. SELECT * FROM orders WHERE odate IN ('1999/03/10', '1999/04/10', '1999/06/10');

-- 10. List all customers whose names begin with a letter 'C'. SELECT * FROM customer WHERE cname LIKE 'C%';

-- 11. List all customers whose names begins with letter 'A' to 'G'. select * from customer where cname between ('a%') and ('h%');

-- OR SELECT * FROM customer WHERE cname LIKE 'A%' OR cname LIKE 'B%' OR cname LIKE 'C%' OR cname LIKE 'D%' OR cname LIKE 'E%' OR cname LIKE 'F%' OR cname LIKE 'G%';

-- 12.List all orders with zero or NULL amount. SELECT * FROM orders WHERE ammount = 0 OR ammount = NULL;

-- 13. Find out the largest order of salesman 1002 and 1007. SELECT * FROM orders WHERE snum = 1002 OR snum = 1007 order BY ammount DESC LIMIT 1;

-- 14. Calculate the Average and Sum of amount orders. SELECT AVG(ammount), SUM(ammount) FROM orders;

-- 15. Count the no. of salesmen currently having orders. SELECT COUNT(snum), snum FROM orders GROUP by snum;

-- 16. Find the largest orders taken by each salesman on each date. select snum, odate, max(ammount) from orders group by snum;

-- 17. Find out each customer's smallest orders. SELECT cnum, MIN(ammount) AS "Smallest Order" FROM orders GROUP BY cnum;

-- 18. Find out the customer in alphabetical orders whose name begins with 'G'. SELECT * FROM customer WHERE cname LIKE 'G%' ORDER BY cname;

-- 19 Display the no. of order for each day in the following format. -- There are "X"(No. of Orders) Orders on "Y"(Date in dd - mon - yy). SELECT onum as X, DATE_FORMAT(odate, ' %d - %m - %y') as Y FROM orders;

-- 20 Assume each salesperson has a 12% commission. -- Write a query on the order table that will Produce the Order number, salesman no. and amount of commission for that order. SELECT onum, snum, ammount * 0.12 FROM orders;

-- 21 List all customers in descending order of rating. SELECT * FROM customer ORDER BY rating DESC;

-- 22 Show the name of all customers with their salesman's name. SELECT salesman.sname, customer.cname FROM customer, salesman WHERE customer.snum = salesman.snum;

-- OR SELECT cname, sname FROM customer INNER JOIN salesman ON customer.snum = salesman.snum;

-- 23. List all orders with the names of their customer and salesman. SELECT onum, sname, cname, FROM orders INNER JOIN customer ON orders.cnum = customer.cnum INNER JOIN salesman ON orders.snum = salesman.snum;

-- 24. List all orders by the customers not located in the same city as their salesman. SELECT onum, sname, cname FROM orders INNER JOIN customer ON orders.cnum = customer.cnum INNER JOIN salesman ON orders.snum = salesman.snum WHERE customer.city != salesman.city;

-- 25. List all customers serviced by salesman with commission above 12%. SELECT cname, sname, commission FROM customer INNER JOIN salesman ON customer.snum = salesman.snum WHERE commission > 12;

-- 26. Find all pairs of customers having the same rating without duplication. -- Pending SELECT c1.cname, c1.rating, c2.cname, c2.rating FROM customer c1, customer c2 WHERE c1.rating = c2.rating AND c1.cnum != c2.cnum AND c2.cnum != c1.cnum;

-- 27 List all customers located in cities where salesman Niraj has customers. select c.cname, c.city from customer c where c.snum = (select snum from salesman where sname like 'Niraj');

-- 28. List all salesmen who are living in the same city without duplicate rows. SELECT s1.sname, s1.city, s2.sname, s2.city FROM salesman s1, salesman s2 WHERE s1.city = s2.city AND s1.snum != s2.snum;

-- 29. Produce the name and city of all the customers with the same rating as Hardik'. SELECT cname, city FROM customer WHERE rating = (SELECT rating FROM customer WHERE cname = 'Hardik');

-- 30. Extract all orders of Miti. select * from orders where snum = (select snum from salesman where sname = 'Miti');

-- 31. Find all orders of the salesman who services 'Hardik'. SELECT * FROM orders WHERE snum = (SELECT snum FROM customer WHERE cname = 'Hardik');

-- 32. List all orders that are greater than the average of April 10, 1999. select * from orders where ammount > (select avg(ammount) from orders where odate = '10-apr-99' group by odate);

-- 33. Count the no.of customers with the rating above than the average rating of 'Surat'. SELECT COUNT(cnum) FROM customer WHERE rating > (SELECT AVG(rating) FROM customer WHERE city = 'Surat');

-- 34. Using correlated sub query find the name and number of all customer with rating equal to Maximum for their city. SELECT cname, cnum FROM customer WHERE rating = (SELECT MAX(rating) FROM customer WHERE city = customer.city);

-- 35 Find all customers having rating greater than any customer in 'Rome'. SELECT cname, cnum FROM customer WHERE rating > (SELECT MAX(rating) FROM customer WHERE city = 'Rome');

-- 36 Find all the customers who have greater rating than every customer in 'Rome'. SELECT cname, cnum FROM customer WHERE rating > (SELECT MIN(rating) FROM customer WHERE city = 'Rome');

-- 37 Select all customers whose rating doesn't match with any rating customer of 'Surat'. SELECT cname, cnum FROM customer WHERE rating NOT IN (SELECT rating FROM customer WHERE city = 'Surat');

-- 38 Create a union of two queries that shows the names, cities and ratings of all customers.Those with rating of >= 200 should display 'HIGH RATING' and those with < 200 should Display 'LOW RATING' SELECT cname, city, rating, 'HIGH RATING' as rating_type FROM customer WHERE rating >= 200 UNION SELECT cname, city, rating, 'LOW RATING' as rating_type FROM customer WHERE rating < 200;

-- 39. Insert a row into salesmen table with the values snum is 1008 salesman name is Rakesh, City is unknown and commission is 14%. INSERT INTO salesman VALUES (1008, 'Rakesh', 'unknown', 14);

-- 40. Insert a row in to customer table with values London, Pratik a 2008 for the columns city, Name and number. INSERT INTO customer VALUES ('London', 'Pratik', 2008);

-- 41. Create another table Londonstaff having same structure as salesman table. CREATE TABLE londonstaff (snum int, sname varchar(20), city varchar(20), commission int);

-- 42. Insert all the rows of salesmen table with city London in the London staff table. INSERT INTO londonstaff SELECT * FROM salesman WHERE city = 'London';

-- 43. Create another table Day totals with two attributes date and total and insert rows into this Table from order table. CREATE TABLE day_totals (date varchar(20), total int);

-- 44. Remove all orders from customer Chandu. DELETE FROM orders WHERE cnum = (SELECT cnum FROM customer WHERE cname = 'Chandu');

-- 45. Increase the rating of all customers in Rome by 100. UPDATE customer SET rating = rating + 100 WHERE city = 'Rome';

-- 46. Double the commission of all salesmen of London. UPDATE salesman SET commission = commission * 2 WHERE city = 'London';

-- 47. Delete the salesmen who produce the lowest order for each day. DELETE FROM salesman WHERE snum IN (SELECT snum FROM orders WHERE odate IN (SELECT odate FROM orders GROUP BY odate HAVING SUM(ammount) = (SELECT MIN(SUM(ammount)) FROM orders GROUP BY odate)));

-- 48. Delete all customers with no current orders. DELETE FROM customer WHERE cnum NOT IN (SELECT cnum FROM orders);

-- 49. Write a command to add the item-name column to the order table. ALTER TABLE orders ADD item_name varchar(20);

-- 50. Give the commands to create our sample tables (salesmen, customer, orders) with all the Necessary constraints like PRIMARY KEY, NOT NULL UNIQUE, FOREING KEY. CREATE TABLE salesman (snum int NOT NULL, sname varchar(20) NOT NULL, city varchar(20) NOT NULL, commission int NOT NULL, PRIMARY KEY (snum));

-- 51. Create a view called Big orders which stores all orders larger than Rs.4000. CREATE VIEW big_orders AS SELECT * FROM orders WHERE ammount > 4000;

-- 52. Create a view that shows all the customers who have the highest ratings. CREATE VIEW highest_rating AS SELECT * FROM customer WHERE rating = (SELECT MAX(rating) FROM customer);

-- 53. Create a view that shows all the number of salesman in each city. CREATE VIEW salesman_city AS SELECT city, COUNT(snum) as no_of_salesman FROM salesman GROUP BY city;

-- 54. Create a view that shows the average and total orders for each salesmen after his name And number. CREATE VIEW salesman_orders AS SELECT sname, snum, AVG(ammount) as avg_order, SUM(ammount) as total_order FROM orders GROUP BY snum;

-- 55. Create a view Show name that shows for each order the order no, amount, salesman name And the customer name. CREATE VIEW order_details AS SELECT o.order_no, o.ammount, s.sname, c.cname FROM orders o, salesman s, customer c WHERE o.snum = s.snum AND o.cnum = c.cnum;