q1) Display total number of orders placed by AAKASH

select count(\*)

from Customer c, Orders o

where

c.cus\_id= o.cus\_id

and

cus\_name like 'AAKASH';

SELECT COUNT( \* ) AS 'Number Of Orders'

FROM Customer c, Orders o

WHERE c.cus\_id = o.cus\_id

AND cus\_name LIKE 'AAKASH'

SELECT COUNT( \* ) AS 'Number Of Orders'

FROM Customer c, Orders o

WHERE c.cus\_id = o.cus\_id

AND cus\_name

IN ('AAKASH')

SELECT COUNT( \* ) AS 'Number Of Orders'

FROM Customer

join

Orders

on

customer.cus\_id=orders.cus\_id

and

cus\_name like 'AAKASH';

q2) total count of orders by Akash and Aman Collectively

SELECT COUNT( \* ) AS 'Number Of Orders'

FROM Customer

join

Orders

on

customer.cus\_id=orders.cus\_id

and

cus\_name in('AAKASH','AMAN');

q) display names of customers staying in either Delhi or Mumbai

SELECT cus\_name

FROM customer

WHERE cus\_city

IN (

'Delhi', 'Mumbai'

)

q3) q1 using joins

q4) q3 using "using" NOTE using followed by ()

q5) Name of supplier and total prices of products delivered

by that supplier

SELECT supp\_name, sum(supp\_price)

FROM supplier

JOIN supplier\_pricing

USING ( supp\_id )

group by supplier.supp\_id;

q) display total number of products shipped from Delhi

SELECT supp\_city, COUNT( \* )

FROM supplier

JOIN supplier\_pricing

USING ( supp\_id )

where supp\_city like 'Delhi'

GROUP BY supp\_city;

q6) q5 + have supplied products worth more than 30,000

SELECT supp\_name, SUM( supp\_price )

FROM supplier

JOIN supplier\_pricing

USING ( supp\_id )

GROUP BY supplier.supp\_id

having SUM( supp\_price )>30000;

q7) display name and price of products supplied by 'Rajesh Retails;

SELECT pro\_name, supp\_price

FROM Supplier

JOIN Supplier\_pricing

USING ( supp\_id )

JOIN Product

USING ( pro\_id )

WHERE supp\_name LIKE 'Rajesh Retails'

q8) q7 using joins ( demo of joinging 3 tables)

q9) display name of customers who have placed more than 3 orders

a)without join keyword

b)with join+ on keyword

SELECT cus\_name, COUNT( \* ) as 'Number of orders'

FROM customer

JOIN orders

on customer.cus\_id = orders.cus\_id

GROUP BY cus\_name

having COUNT( \* ) >3;

c)with join+using keyword

SELECT cus\_name, COUNT( \* ) as 'Number of orders'

FROM customer

JOIN orders

USING ( cus\_id )

GROUP BY cus\_name

having COUNT( \* ) >3;

d) nested query

SELECT cus\_name

FROM customer

WHERE cus\_id

IN (

SELECT c.cus\_id

FROM customer c, orders o

WHERE c.cus\_id = o.cus\_id

GROUP BY cus\_id

HAVING COUNT( \* ) >3

)

LIMIT 0 , 30

q10 display name of customers who have not placed any order

SELECT cus\_name

FROM customer

LEFT JOIN orders

USING ( cus\_id )

WHERE ord\_id IS NULL

q11 display all the orders along with product name ordered by a customer 'AAKASH'

SELECT ord\_id,pro\_name,pro\_desc

FROM customer

JOIN orders

USING ( cus\_id )

JOIN supplier\_pricing

USING ( pricing\_id )

JOIN product

USING ( pro\_id )

WHERE cus\_name LIKE 'AAKASH';

q12 Display the Supplier details of who is supplying more than two product

SELECT supp\_name, COUNT( \* )

FROM supplier

JOIN supplier\_pricing

USING ( supp\_id )

JOIN product

USING ( pro\_id )

GROUP BY supp\_name

having COUNT( \* ) > 2;

q13 Display customer name and gender whose names start or end with character 'A'.

select cus\_name

from customer

where cus\_name like 'A%' or cus\_name like '%A';

q14 Display the Id and Name of the Product ordered after “2021-10-05”.

SELECT pro\_name

FROM product

JOIN supplier\_pricing

USING ( pro\_id )

JOIN orders

USING ( pricing\_id )

WHERE ord\_date > "2021-10-05"

q15 Display the total number of customers based on gender who have placed

orders of worth at least Rs 3000

SELECT cus\_gender, count(\*)

FROM customer

JOIN orders

USING ( cus\_id )

JOIN supplier\_pricing

USING ( pricing\_id )

WHERE supp\_price >3000

group by cus\_gender;

q16 the least expensive product from each category and print the table

with category id, name, and price of the product

SELECT cat\_name, pro\_name, pro\_desc, MIN( supp\_price )

FROM category

JOIN product

USING ( cat\_id )

JOIN supplier\_pricing

USING ( pro\_id )

GROUP BY cat\_name

q17 Create a stored procedure to display supplier id, name,

rating and Type\_of\_supplier. If rating >4 then “Genuine Supplier” if rating >2

“Average Supplier” else “Supplier should not be considered

step 1:

SELECT supp\_name, AVG( stars )

FROM supplier

JOIN supplier\_pricing

USING ( supp\_id )

JOIN orders

USING ( pricing\_id )

JOIN rating

USING ( ord\_id )

GROUP BY supp\_name

step 2:

SELECT supp\_name, AVG( stars ),

Case

when AVG( stars )=5 then 'Excellent Service'

when AVG( stars )>4 then 'Good Service'

when AVG( stars )>3 then 'Average Service'

else 'Poor Service'

end as 'Supplier Category'

FROM supplier

JOIN supplier\_pricing

USING ( supp\_id )

JOIN orders

USING ( pricing\_id )

JOIN rating

USING ( ord\_id )

GROUP BY supp\_name

step 3

DELIMITER &&

CREATE PROCEDURE supplierValue()

BEGIN

SELECT supp\_name, AVG( stars ),

Case

when AVG( stars )=5 then 'Excellent Service'

when AVG( stars )>4 then 'Good Service'

when AVG( stars )>3 then 'Average Service'

else 'Poor Service'

end as 'Supplier Category'

FROM supplier

JOIN supplier\_pricing

USING ( supp\_id )

JOIN orders

USING ( pricing\_id )

JOIN rating

USING ( ord\_id )

GROUP BY supp\_name;

END &&

DELIMITER ;

call supplierValue()