#Q1. Print product, price, sum of quantity more than 5 sold during all three months.

Select product, round(sum(price),2) sales from bank\_inventory\_pricing group by product having sum(Quantity) >5

#Q2.Print product, quantity , month and count of records for which estimated\_sale\_price is less than purchase\_cost

Select product, month, sum(quantity), count(\*) from bank\_inventory\_pricing where estimated\_sale\_price < purchase\_cost group by product, month

1. Where
2. Group
3. Sum and count

#Q3. Extarct the 3rd highest value of column Estimated\_sale\_price from bank\_inventory\_pricing dataset

Select \* from bank\_inventory\_pricing order by Estimated\_sale\_price desc limit 2,1

#Q4. Count all duplicate values of column Product from table bank\_inventory\_pricing

Select product from bank\_inventory\_pricing group by product having count(\*) >1

#Q5. Create a view 'bank\_details' for the product 'PayPoints' and Quantity is greater than 2

Create view ‘bank\_details’ as select \* from bank\_inventory\_pricing where product = “PayPoints” and quantity > 2

#Q6 Update view bank\_details1 and add new record in bank\_details1.

-- --example(Producct=PayPoints, Quantity=3, Price=410.67)

#Q7.Real Profit = revenue - cost Find for which products, branch level real profit is more than the estimated\_profit in Bank\_branch\_PL.

Select branch, product, sum(estimated\_profit) EP, sum(revenue -cost) RP from bank\_branch\_pl group by branch, product having RP >EP

#Q8.Find the least calculated profit earned during all 3 periods

Select month, sum(revenue-cost) profit from bank\_branch\_pl group by month order by profit limit 1

#Q9. In Bank\_Inventory\_pricing,

-- a) convert Quantity data type from numeric to character

-- b) Add then, add zeros before the Quantity field.

#Q10. Write a MySQL Query to print first\_name , last\_name of the employee\_details whose first\_name Contains ‘U’

Select \* from employee\_details where first\_name like “%U%”

#Q11.Reduce 30% of the cost for all the products and print the products whose calculated profit at branch is exceeding estimated\_profit .

Select branch, product, sum(revenue-(cost\*0.7)) CP, sum(estimated\_profit) EP from bank\_branch\_pl group by branch,product having CP >EP

#Q12.Write a MySQL query to print the observations from the Bank\_Inventory\_pricing table excluding the values “BusiCard” And “SuperSave” from the column Product

Select \* from bank\_inventory\_pricing where product NOT IN (“BusiCard”, “SuperSave”)

#Q13. Extract all the columns from Bank\_Inventory\_pricing where price between 220 and 300

Select \* from Bank\_Inventory\_pricing where price between 220 and 300

Select \* from Bank\_Inventory\_pricing where price >= 220 and price <=300

#Q14. Display all the non duplicate fields in the Product form Bank\_Inventory\_pricing table and display first 5 records.

Select distinct product from Bank\_Inventory\_pricing order by product limit 5

Q15.Update price column of Bank\_Inventory\_pricing with an increase of 15% when the quantity is more than 3.

Update Bank\_Inventory\_pricing set price = Price\*1.15 where quantity > 3 and price is not null

#Q16. Show Round off values of the price without displaying decimal scale from Bank\_Inventory\_pricing

Select round(price,0) from Bank\_Inventory\_pricing

#Q17.Increase the length of Product size by 30 characters from Bank\_Inventory\_pricing.

#Q18. Add '100' in column price where quantity is greater than 3 and dsiplay that column as 'new\_price'

Select product, price, (price+100) new\_price from bank\_inventory\_pricing where quantity>3 and price is not null

#Q19. Display all saving account holders have “Add-on Credit Cards" and “Credit cards"

Select \* from bank\_account\_details a where a.account\_type IN ("Add-on Credit Card" ,"Credit Card") and a.customer\_id IN (SELECT distinct b.Customer\_id FROM bank.bank\_account\_details b where b.Account\_type = "SAVINGS")

#Q20.

# a) Display records of All Accounts , their Account\_types, the transaction amount.

Select a.account\_number, account\_type, transaction\_amount from bank\_account\_transaction a left outer join bank\_account\_details b on a.account\_number = b.account\_number

# b) Along with first step, Display other columns with corresponding linking account number, account types

Select a.account\_number, b.account\_type, transaction\_amount, linking\_account\_number from

bank\_account\_transaction a

left outer join bank\_account\_details b

on a.account\_number = b.account\_number

left outer join bank\_account\_relationship\_details c

on a.account\_number = c.account\_number

# c) After retrieving all records of accounts and their linked accounts, display the transaction amount of accounts appeared in another column.

#Q21.Display all type of “Credit cards” accounts including linked “Add-on Credit Cards"

# type accounts with their respective aggregate sum of transaction amount.

# Ref: Check linking relationship in bank\_transaction\_relationship\_details.

# Check transaction\_amount in bank\_account\_transaction.

Select a.account\_number, account\_type, sum(transaction\_amount )

from bank\_account\_transaction a

left outer join bank\_account\_details b

on a.account\_number = b.account\_number

where account\_type IN (“Credit Card”, "Add-on Credit Card")

group by a.account\_number

#Q22. Compare the aggregate transaction amount of current month versus aggregate transaction with previous months.

# Display account\_number, transaction\_amount ,

-- sum of current month transaction amount ,

-- current month transaction date ,

-- sum of previous month transaction amount ,

-- previous month transaction date.

Select \* from

(SELECT account\_number, sum(transaction\_amount) from bank\_account\_transaction y where month(transaction\_date) != month(current\_timestamp()) group by account\_number) a

Full outer join

(SELECT account\_number, sum(transaction\_amount) from bank\_account\_transaction x where month(transaction\_date) = month(current\_timestamp()) group by account\_number) b

On a.account\_number = b.account\_number

#Q23.Display individual accounts absolute transaction of every next month is greater than the previous months .

#Q24. Find the no. of transactions of credit cards including add-on Credit Cards

#Q25.From employee\_details retrieve only employee\_id , first\_name ,last\_name phone\_number ,salary, job\_id where department\_name is Contracting (Note

#Department\_id of employee\_details table must be other than the list within IN operator.

#Q26. Display savings accounts and its corresponding Recurring deposits transactions are more than 4 times.

#Q27. From employee\_details fetch only employee\_id, ,first\_name, last\_name , phone\_number ,email, job\_id where job\_id should not be IT\_PROG.

#Q29.From employee\_details retrieve only employee\_id , first\_name ,last\_name phone\_number ,salary, job\_id where manager\_id is '60' (Note

#Department\_id of employee\_details table must be other than the list within IN operator.

#Q30.Create a new table as emp\_dept and insert the result obtained after performing inner join on the two tables employee\_details and department\_details.