Lab 7

Relational Algebra & SQL Queries

1. Find the ordered items which are in stock.

SQL Statement:

select s.itemid,orderQuantity.orderid from stock as s join (select oc.itemid,oc.quantity,o.orderid from orders as o join ordercontains as oc on oc.orderid= o.orderid) as orderQuantity on orderQuantity.itemid = s.itemid where orderQuantity.quantity<=s.quantity;

Relational Algebra:

4	itemid character (10)		orderid character (10)	
1	ITEM000005		ORDER00001	
2	ITEM000001		ORDER00001	
3	ITEM000001	ORDER00002		

2. Find the customers whose orders are in stock.

SQL Statement:

Select s.itemid,orderQuantity.orderid,orderQuantity.name from stock as s JOIN (select oc.itemid,oc.quantity,o.orderid,c.name from orders as o JOIN ordercontains as oc on oc.orderid= o.orderid JOIN customer as c on c.customerid= o.customerid o.customerid o.customerid o.customerid orderQuantity on orderQuantity.itemid = s.itemid where orderQuantity.quantity<=s.quantity;

Relational Algebra:

TT (Stock

stock.item.id,orderQuantity.orderid,orderQuantity.name (orderQuantity,(orderQuantity,(orderQuantity,quantity,quantity,quantity,quantity) (orderContains.itemid,orderContains.quantity,quantity,quantity,quantity) (orders.orderid,customer.name) (ordercontains.orderid=orders.orderid),(orders.customerid=customer.customerid)>

4	name character varying (20)	orderid character (10)	<u> </u>	itemid character (10)	
1	Akhilesh	ORDER00001		ITEM000005	
2	Akhilesh	ORDER00001		ITEM000001	
3	Mitesh	ORDER00002		ITEM000001	

3. Display the category names with their ordered quantities from all the orders placed.

SQL Statement:

Select name, sum (result. quantity) from

(Select itemcategory.name, order contains. quantity from itemcategory

JOIN item on item.categoryid = itemcategory.categoryid

JOIN ordercontains on item.itemid = ordercontains.itemid

JOIN orders on ordercontains.orderid = orders.orderid

) as result

GROUP BY name



	(r1)			
resul	t=∏(name SUM(quantity)) name character varying (20)	sum numeric		
1	Jewellery	2		
2	Grooming	19		
3	Fashion	9		

4. Show the number of warehouses present in each city.

SQL Statement:

Select city.name,count(*) from warehouse JOIN city on city.cityid=warehouse.cityid GROUP BY city.name

Relational Algebra:



5. List all the managers of different warehouses.

SQL Statement:

Select emp.* from employee as emp JOIN warehouse as w on w.managerid = emp.emp_id

Relational Algebra:

 $\pi_{emp,*}(employee \quad {\scriptstyle \text{ \ \ } \text{ \ \ \ } \text{ \$

4	emp_id [PK] character (10)	warehouseid character (10)	name character (20)	salary numeric	contact character (10)
1	EMP0000001	WRHSE00001	Suresh	10000	9123456789
2	EMP0000006	WRHSE00004	Akshay	15000	9132456789
3	EMP0000003	WRHSE00002	Shrey	11000	9123456879

6. In which warehouses the employee count is below the average count.

SQL Statement:

Select warehouseid ,count(*) as Employee_count from employee as e GROUP BY e.warehouseid having count(*) > (select Trunc(avg(ware_count.count),2)from (select warehouseid ,count(*) as count from employee as e GROUP BY e.warehouseid) as ware_count)

Relational Algebra:

$$\begin{array}{ll} \text{r1} = & \text{warehouseid}, & \text{$F_{\text{count}(")->\text{count}}$ (employee)} \\ \text{r2} = & \text{$F_{\text{AVG(count)}}$ (r1)} \\ \text{(r1...count>r2.AVG(count))} \\ \text{result} = & \pi & \text{(r3)} \\ \text{(warehouseid,count)} \end{array}$$

4	warehouseid character (10)	employee_count bigint	<u></u>
1	WRHSE00002		2
2	WRHSE00001		2

7. List all the pending orders.

SQL Statement:

Select * from orders where order_delivery_date is null

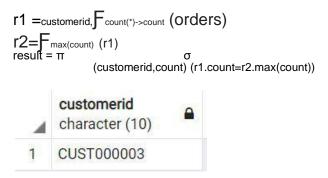
4	orderid [PK] character (10)	transportid character (10)	customerid character (10)	order_placed_date date	order_delivery_date date
1	ORDER00004	TRANS00002	CUST000003	2020-08-02	[null]

8. List the customerid with the highest number of orders placed.

SQL Statement:

Select customerid from (select customerid ,count(*) from orders GROUP BY customerid) as maxcount where count in (select max(count) from (select customerid ,count(*) from orders GROUP BY customerid)as maxc)

Relational Algebra:



9. Name the vehicle no/id which is having the highest number of orders to transport.

SQL Statement:

Select vehicleid, countfrom (select vehicleid, count(*) from transport GROUP BY vehicleid) as maxcount where count in (select max(count) from (select vehicleid, count(*) from transport GROUP BY vehicleid) as maxc)

$$\begin{array}{l} \text{r1 =} \text{vehicleid,} \\ \text{F}_{\text{count(*)-}\text{-}\text{count}} \text{ (transport)} \\ \text{r2=} \\ \text{F}_{\text{max(count)}} \text{ (r1)} \\ \text{result = } \pi \qquad \sigma \\ \text{(vehicleid,count)} \text{ (r1.count=r2.max(count))} \end{array}$$

4	vehicleid character (10)	<u></u>	count bigint	
1	VEHIC00001			1
2	VEHIC00003			1
3	VEHIC00006			1

10. Get the most popular item from each category

```
select ic2.name,i2.name from itemCategory ic2
      join item i2 on i2.itemCategoryID =
      ic2.itemCategoryID join (
            select tmp.ici ici ,max(cnt) mx from (
                select i.itemCategoryID ici,i.itemID ii,count(oc.quantity) cnt from
item i
                join orderContains oc on oc.itemID = i.itemID
                group by i.itemCategoryID,i.itemID
                ) tmp
                group by tmp.ici
      ) tmp2
      on ic2.itemCategoryID = tmp2.ici
      where tmp2.mx = (select count(*) from orderContains where itemID
= i2.itemID);
Relational Algebra:
        ρ (ic2,itemCategoryID) ⋈
                                     i2.itemCategoryID = ic2.itemCategoryID
        ρ (i2,item) ⋈
        ρ (tmp2 (tmp.ici \xrightarrow{\text{measure}} ici, max(cnt) \rightarrow mx), (
                        \rho (tmp(i.itemCategoryID \stackrel{\text{\tiny mess}}{	o} ici, i.itemID \rightarrow
ii,count(oc.quantity) \rightarrow cnt),(
                                                         ρ (i,item)⋈
(oc,orderContains)
                                                                     oc.itemID = i.itemID
                                                )
                                        )
                        ))
               ))
        ))
)
```

name	name
toys	Item 1
toys	Item 2
home appliance	Item 3
home appliance	Item 5
device	Item 6
vehicle	Item 7
vehicle	Item 8
vehicle	Item 9
cosmetic	Item 10

^{**(}Item1 and Item2 are having same Number of orders they are of toys category, similarly item3,item5 they are of home appliance category and Item7, Item8, Item9 they are of vehicle category)

11. List all customers who met the given employee in last week

```
select distinct c.name,o.orderID from employee e
join transport t on t.employeeID = e.employeeID
join _order o on o.transportID = t.transportID
join customer c on c.customerID =
o.customerID where e.employeeID = 1
and o.dileveryDate between current_date -7 and current_date
```

```
name | orderid
```

12. The average salary of employees at each warehouse

SQL Statement:

select (select name from warehouse where warehouseID=w.warehouseID)as name,avg(e.salary) avg_salary from warehouse w join employee e on e.warehouseID=w.warehouseID group by w.warehouseID

IMP - * x * means subscript – used to express subscript of a subscript Relational Algebra:

name	avg_salary
WareHouse 4	25400.0000000000000
WareHouse 6	25400.0000000000000
WareHouse 2	50400.0000000000000
WareHouse 3	38466.66666666667
WareHouse 5	25600.0000000000000
WareHouse 7	3175400.0000000000000
WareHouse 1	30400.0000000000000

13. List of items ordered by a given customer in chronological order

```
select i.name from customer c
      join _order o on o.customerID=c.customerID
      join orderContains oc on oc.orderID=o.orderID
      join item i on i.itemID=oc.itemID
      where c.customerID = 1
      order by o.orderDate;
Relational Algebra:
i.name
         σ
           c.customerID =1
                  ρ (c,customer) ⋈
                                         o.customerID=c.customerID
                  ρ (o,_order) ⋈
                   \begin{array}{c} \text{oc.orderID=0.orderID} \\ \rho \text{ (oc,orderContains)} \bowtie \end{array} 
                                                 i.itemID=oc.itemID
                  ρ (i,item)
)
    name
  Item 6
  Item 1
  Item 6
```

14. List suppliers in order of their popularity

SQL Statement:

```
select (select name from supplier where supplierID = s.supplierID) as name from supplier s
join suppliedBy sb on sb.supplierID = s.supplierID
join orderContains oc on (oc.orderID=sb.orderID and oc.itemID=sb.itemID)
group by s.supplierID
order by sum(oc.quantity) desc;
```

Relational Algebra:

IMP - * x * means subscript – used to express subscript of a subscript IMP – Can't completely express it in relational algebra because it doesn't support ordering

Supplier 2 Supplier 9 Supplier 4 Supplier 6 Supplier 5 Supplier 1 Supplier 3 Supplier 7 Supplier 8

15. Calculate the average delivery time in each city

```
select (select name from city where cityID = c.cityID) as name
,avg(o.dileveryDate-o.orderDate) days from city c
     join warehouse w on w.cityID = c.cityID
     join employee e on e.warehouseID = w.warehouseID
     join transport t on t.employeeID=e.employeeID
     join _order o on o.transportID = t.transportID
     group by c.cityID
       order by avg(o.dileveryDate-o.orderDate)
Relational Algebra:
IMP - * x * means subscript – used to express subscript of a subscript
   \prod *name* (\sigma* cityID = c.cityID* (city) ), avg(o.dileveryDate-o.orderDate)
       w.cityID = c.cityID
                     ρ (w,warehouse) ⋈
                                             e.warehouseID = w.warehouseID
                     ρ (e,employee) ⋈
                                           t.employeeID=e.employeeID
                     ρ (t,transport) ⋈
                                         o.transportID = t.transportID
                     ρ (o,_order)
 ahmedabad | 3.10000000000000000
```

16. Calculate the average number of items in a single order

17. List drivers ordered by their 'Bang for Buck'

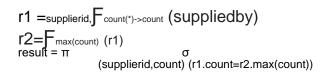
SQL Statement:

```
select (select name from employee where employeeID=e.employeeID) as name from
   employee e
       join transport t on t.employeeID = e.employeeID
       group by e.employeeID,e.salary
       order by (e.salary/count(*))
.Relational Algebra:
   IMP - * x * means subscript - used to express subscript of a subscript
   IMP - Can't completely express it in relational algebra because it doesn't
   support ordering
                                              (
     Π *name * (σ * employeeID=e.employeeID* (employee))
          e.employeeID,e.salary F (
                        ρ (e,employee) κ
                                            t.employeeID = e.employeeID
                        ρ (t,transport)
        name
    Employee 1
    Employee 2
     Employee 3
    Employee 4
```

18. List the supplier, who is supplying the highest count of items.

```
Select supplierid, count from (select supplierid, count(*) from suppliedby GROUP BY supplierid) as maxcount where count in (select max(count) from (select supplierid, count(*) from suppliedby GROUP BY supplierid) as maxc)
```

Relational Algebra:



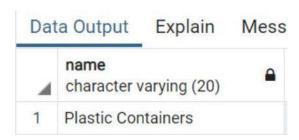


19. List name of item having ordered quantity between 15 & 45 and the name should start with the character 'P' having third character 'a'

SQL Statement:

select name from item
join stock
on stock.itemid=item.itemid
where quantity between 15 and 45 and name LIKE 'P_a%'





20. List the warehouse names and item names with stock of items with quantity 15 to 45 **SQL Statement :**

select warehouse.name,itemNameWarehouseld.name from
warehouse join (select name,itemWarehouse.warehouseid from item
join (select itemid,warehouseid from stock
where quantity between 15 and 45) as itemWarehouse
on item.itemid = itemWarehouse.itemid) as itemNameWarehouseld on
warehouse.warehouseid = itemNameWarehouseld.warehouseid

Relational Algebra:



21. List all customer who ordered all the items

```
select customerid from orders
except
(
select customerid from
(
select o.customerid, items.itemid from orders as o
join ordercontains as oc
on oc.orderid=o.orderid
cross join
(select * from item) as items
except
```

```
select customerid, itemid from orders as o join ordercontains as oc
on oc.orderid=o.orderid
) as r2
```

Relational Algebra:

)

```
 r_0 = \pi(\text{items}) 
 r_1 = \pi_{\text{o.customerid,r0.itemid}} 
 r_2 = r_1 - (\pi(\text{customerid,itemid})) (\rho(\text{o.orders}) < \text{o.orderid=oc.orderid>}) 
 result = \pi_{\text{customerid}} 
 (\rho(\text{oc,ordercontains}) \times r_0) 
 result = \pi_{\text{customerid}} 
 (\text{orders}) - r_2
```



22. List items that are available at all warehouses.

23. List warehouses that have all the items.

SQL Statement:

24. List highest salary of employees in each warehouse.

SQL Statement:

SELECT w.name, MAX(Salary) FROM Employee as e JOIN WareHouse as w ON e.warehouseid = w.warehouseid GROUP BY w.name;

Relational Algebra:

w.name,

	[⋈] ρ(W,warehouse)				
Π	name $max(salary)(\rho(e,employee) < e.warehouseid=v$ character (30)		,		
1	Vadodara Warehouse 1	15000			
2	Surat Warehouse 1	20000			
3	Surat Warehouse 2	12000			
4	Vadodara Warehouse 2	9000			

25. Find the maximum price of each category

SQL Statement:

```
SELECT name,maxprice
FROM item a

JOIN
(

SELECT Categoryid, MAX(cost) as maxPrice
FROM item
GROUP BY Categoryid
) b

ON a.Categoryid = b.Categoryid AND a.cost = b.maxPrice

Relational Algebra:

r1 = \rho(a, (\Pi_{name, maxprice}(item)))
r2 = \rho(b, (\Pi_{name, maxprice}(item)))
r_{name, maxprice}(item))
```

 $Result = r1 \\ \text{<a.categoryid} \quad \text{= b.categoryid AND a.cost = b.maxPrice>} \\ r2$

4	name character varying (20) □	maxprice integer
1	Earrings	3000
2	Facewash	150
3	Plastic Containers	50
4	Hoodie	450

26. List the count of warehouses in each city.

SQL Statement:

Select city.name,count(*) from warehouse JOIN city on city.cityid=warehouse.cityid Group by city.name

Relational Algebra:



27. List all the items details listed under the category 'Fashion'

SQL Statement:

Select * from item

JOIN itemcategory

π.*(itemcategory

On Item.Categoryid = itemcategory.categoryid

Where itemcategory.name = 'Fashion'

