Lab 7

Relational Algebra & SQL Queries

1. Find the ordered items which are in stock.

```
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;
```

$$\pi_{item.id,orderid}(stock_{<orderQuantity.itemid=stock.itemid\ AND\ orderQuantity.quantity<=stock.quantity>} \bowtie (orderQuantity,(\sigma_{(ordercontains.itemid,ordercontains.quantity)}(orders))_{<ordercontains.orderid=orders.orderid>} \bowtie ordercontains))$$

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

2. Find the customers whose orders are in stock.

```
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
) as orderQuantity
on orderQuantity.itemid = s.itemid
where orderQuantity.quantity<=s.quantity;
```

$\pi_{stock.item.id,orderQuantity.orderid,orderQuantity.name}(stock_{<orderQuantity.itemid=stock.itemid\ AND\ orderQuantity.quantity<=stock.quantity>} \bowtie (orderQuantity,(\sigma_{(ordercontains.itemid,ordercontains.quantity,orders.orderid,customer.name)}(orders))_{<(ordercontains.orderid=orders.orderid),(orders.customerid=customer.customerid)>} \bowtie ordercontains \bowtie customer))$

	name character varying (20)	orderid character (10)	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.

```
Select name,sum(result.quantity) from
(Select itemcategory.name,ordercontains.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
```

$r1 = \pi_{(itemcategory.name,ordercontains.quantity)}(itemcategory \bowtie_{<items.categoryid=itemcategory.categoryid>} item \bowtie_{<items.itemsid=ordercontains.itemid>} ordercontains \bowtie_{<ordercontains.itemsid=orders.itemid>} orders)$

$result = \pi_{(name,\mathcal{F}_{SUM(quantity))}}(r1)$

	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.

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

$\pi_{city.name,}\mathcal{F}_{count(*)}(warehouse \bowtie_{<warehouse.cityid=city.cityid>} city)$

	name character varying (20)	count bigint
1	Gandhinagar	1
2	Vadodara	2
3	Surat	2

5. List all the managers of different warehouses.

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

$\pi_{emp,*}(employee \bowtie_{<warehouse.managerid=employee.employeeid>} warehouse)$

	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.

```
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)
```

$r1 = {}_{warehouseid,}\mathcal{F}_{count(*)->count}$ (employee)

$r2 = \mathcal{F}_{AVG(count)}$ (r1)

$r3 = \sigma_{(r1..count>r2.AVG(count))}$

result = $\pi_{(warehouseid,count)}$(r3)

	warehouseid character (10)	employee_count bigint
1	WRHSE00002	2
2	WRHSE00001	2

7. List all the pending orders.

```
Select * from orders where order_delivery_date is null
```

$\sigma_{(order\_delivery\_date = NULL)}$(orders)

	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.

```
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)
```

r1 = $_{customerid,}\mathcal{F}_{count(*)->count}$ (orders)

r2= $\mathcal{F}_{max(count)}$ (r1)

result = $\pi_{(customerid,count)}\sigma_{(r1.count=r2.max(count))}$

	customerid character (10)
1	CUST000003

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

```
Select vehicleid,count from
(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)
```

r1 = $_{vehicleid,}\mathcal{F}_{count(*)->count}$ (transport)

r2= $\mathcal{F}_{max(count)}$ (r1)

result = $\pi_{(vehicleid,count)}\sigma_{(r1.count=r2.max(count))}$

	vehicleid character (10)	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);
```

$$\Pi_{ic2.name,i2.name} (\rho(ic2,itemCategoryID) \bowtie_{i2.itemCategoryID = ic2.itemCategoryID} \rho(i2,item) \bowtie_{temCategoryID = tmp2.ici} \rho(tmp2(tmp.ici \rightarrow ici, max(cnt) \rightarrow mx), (\Pi_{tmp.ici\ ici\ ,max(cnt)\ mx} ({}_{tmp.ici}F_{max(cnt)} (\rho(tmp(i.itemCategoryID \rightarrow ici, i.itemID \rightarrow ii,count(oc.quantity) \rightarrow cnt),(\Pi_{i.itemCategoryID\ ,i.itemID\ ,count(oc.quantity)} ({}_{i.itemCategoryID,i.itemID}F_{count(oc.quantity)} (\rho(i,item) \bowtie_{oc.itemID = i.itemID} \rho(oc,orderContains)))))))))$$

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

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
```

$$\Pi_{\text{c.name,o.orderID}} (\sigma_{\text{e.employeeID = 1 and o.dileveryDate between current\_date -7 and current\_date}} (\rho(e,\text{employee}) \bowtie_{\text{t.employeeID = e.employeeID}} \rho(t,\text{transport}) \bowtie_{\text{o.transportID = t.transportID}} \rho(o,\text{\_order}) \bowtie_{\text{c.customerID = o.customerID}} \rho(c,\text{customer})))$$

name	orderid
Customer 1	1

12. The average salary of employees at each warehouse

```
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

$$\Pi_{\Pi\ *name*\ (\sigma\ *warehouseID=w.warehouseID*\ (warehouse))\ ,\ avg(salary)}\ (\ {}_{w.warehouseID}F_{avg(e.salary)}\ (\ \rho\ (w,warehouse) \bowtie_{e.warehouseID=w.warehouseID} \rho\ (e,employee)\ )\ )$$

name	avg_salary
WareHouse 4	25400.000000000000
WareHouse 6	25400.000000000000
WareHouse 2	50400.000000000000
WareHouse 3	38466.666666666667
WareHouse 5	25600.000000000000
WareHouse 7	3175400.000000000000
WareHouse 1	30400.000000000000

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;
```

$$\Pi_{\text{i.name}} \left(\sigma_{\text{c.customerID}=1} \left(\rho_{(c,\text{customer})} \bowtie_{\text{o.customerID=c.customerID}} \rho_{(o,\text{\_order})} \bowtie_{\text{oc.orderID=o.orderID}} \rho_{(oc,\text{orderContains})} \bowtie_{\text{i.itemID=oc.itemID}} \rho_{(i,\text{item})} \right) \right)$$

```
  name
--------
 Item 6
 Item 1
 Item 6
```

14. List suppliers in order of their popularity

```
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;
```

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

$$\Pi_{\Pi\ *name*\ (\sigma\ *\ supplierID\ =\ s.supplierID*\ supplier)} \Big({}_{s.supplierID}F \big(\rho\ (s,supplier) \bowtie_{sb.supplierID\ =\ s.supplierID} \rho\ (sb,suppliedBy) \bowtie_{(oc.orderID=sb.orderID\ and\ oc.itemID=sb.itemID)} \rho\ (oc,orderContains) \big) \Big)$$

```
    name
------------
 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)
```

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

$$\Pi_{\Pi\ *name*\ (\sigma*\ cityID = c.cityID*\ (city))\ ,\ avg(o.dileveryDate-o.orderDate)}\ (\ {}_{c.cityID}F_{avg(o.dileveryDate-o.orderDate)}\ (\ \rho\ (c,city) \bowtie_{w.cityID = c.cityID} \rho\ (w,warehouse) \bowtie_{e.warehouseID = w.warehouseID} \rho\ (e,employee) \bowtie_{t.employeeID=e.employeeID} \rho\ (t,transport) \bowtie_{o.transportID = t.transportID} \rho\ (o,\_order)\ )\ )$$

name	days
ahmedabad	3.1000000000000000

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

```
select avg(cnt) from (
      select count(*) as cnt from orderContains
      group by orderID
) tmp ;
```

$$F_{avg(cnt)} (\rho ((count(*) \rightarrow cnt), {}_{orderID}F_{count(*)} (orderContains)))$$

```
        avg
--------------------
 1.5555555555555556
```

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

```
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(*))
;
```

.

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

$$\prod_{\prod \text{*name * } (\sigma \text{ * employeeID=e.employeeID* (employee)) }} (e.employeeID,e.salary\ F\ (\rho\ (e,employee) \bowtie_{t.employeeID = e.employeeID} \rho\ (t,transport)))$$

```
    name
------------
 Employee 1
 Employee 2
 Employee 3
 Employee 4
```