Lab-8
Relation Algebra and SQL Queries for Functional Requirements
Prepared By: Group 22

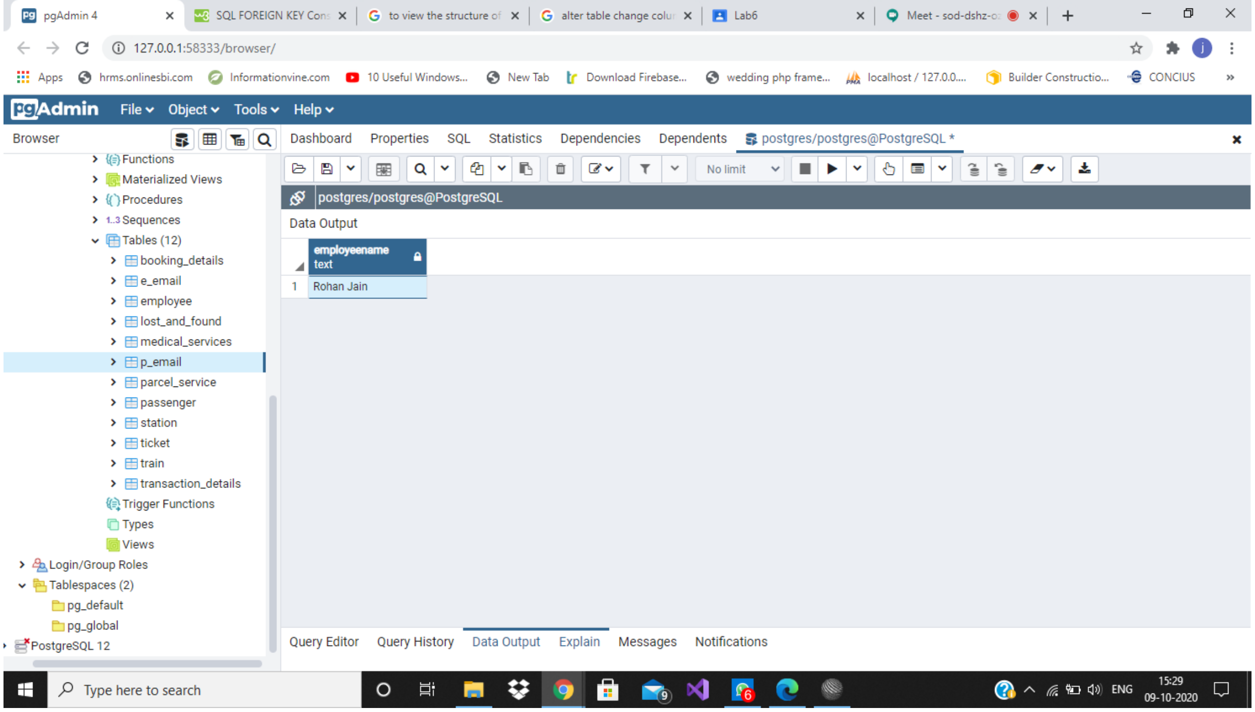
1)Query to retrieve employees with age more than 50 working in safety department

Relational Algebra:

Π $(fname, lname)$ **->**
EmployeeName **(**$\boldsymbol{\sigma} < (age > 50)\ \ AND\ department = 'Safety' >$ **)**

SQL query:

select fname ||' '|| lname as EmployeeName from employee where age > 50 and department='Safety';



	employeename text
1	Rohan Jain

2)Query to retrieve total employees working in each department

Relational Algebra:
$\Pi_{\text{department}}({}_{\text{department}}\mathscr{F}_{\text{count(eid)}})$

SQL QUERY:
select department, count(eid) from employee group by department;

3)Query to retrieve train name that will transport parcel from source AHMEDABAD to destination GANDHINAGAR

RELATIONAL ALGEBRA:
Π<par_id,train_id,train>(parcel⋈**<**s.sid=pa.sid **>**station⋈**<**s.train_id=tr.train_id**>**train)***σ*****<**pa.source_t='Ahmedabad' and pa.destination_t='Gandhinagar' and tr.t_source='Ahmedabad' and tr.t_destination='Gandhinagar'**>**

SQL query:

```
select pa.par_id, tr.train_id, tr.train from parcel pa inner join station s on
s.sid = pa.sid inner join train tr on s.train_id = tr.train_id where
pa.source_t='Ahmedabad' and pa.destination_t='Gandhinagar' and
tr.t_source='Ahmedabad' and tr.t_destination='Gandhinagar';
```

4)RETRIEVE EMPLOYEE WORKING ON TRAIN ID 12005 OR 12115

Relational Algebra:Π <e_id>(employee)(σ train.e_id=’12005’ or train.e_id=’12115’ (employee)⋈<employee.e_id=train.e_id>train).

SQL query:

```
SELECT employee.e_id FROM employee , train  WHERE
employee.e_id=train.e_id
OR ( train.train_id='12005' OR train.train_id='12115')
```

5)Retrieve lost and found item details that are submitted at station id's 1 or 2

Relational Algebra:
Π<item_id, item_name, description, sid**>**(lost_and_found)⋈**<**s.sid = lf.sid **>*σ*<**s.sid=1 or s.sid=2**>**

SQL query:
select lf.item_id, lf.item_name, lf.description, lf.sid from lost_and_found lf inner join station s on s.sid = lf.sid where s.sid=1 or s.sid=2;

6)Retrieve all the station names where patients are suffering from covid 19.

Relational Algebra:
Π<station_name,patient_name>(station,medical_services) ⋈ <s.sid = medical_services.sid> σ<medical_services.description like '%COVID%'>

SQL Query:
select station.station_name,medical_services.patient_name from station,medical_services inner join station s on s.sid = medical_services.sid where medical_services.description like '%COVID%';

7)show the first and last name of the passenger with their ticket details of train id12135.

Relational Algebra:
Π<fname,lname,ticket_id,pid,tra_id,train_id>(passenger)⋈**<**passenger.pid=ticket.pid**>** (ticket) ***σ*****<**ticket.train_id='12135'**>**

SQL QUERY:

```
select
passenger.fname,passenger.lname,ticket.ticket_id,ticket.pid,ticket.tra_id,ticket.train_id from passenger  inner join ticket on passenger.pid=ticket.pid where
ticket.train_id='12135';
```

8)Retrieve the first and last name of an employee who works on a train and whose name starts with “R” and is older than 20.

Relational Algebra:

Π<e_id , fname , lname**>(**employee**)σ<**employee.FNAME like 'R%' And employee.age > 20>(employee)⋈<employee.e_id=train.e_id>(train)
>

SQL QUERY:

```
SELECT employee.e_id , fname , lname
FROM employee
WHERE employee.e_id IN (SELECT train.e_id
FROM train
WHERE employee.FNAME like 'R%') And employee.age > 20
```

9)Retrieve passenger name and all ticket details of passenger with the help of transaction details table and train table.

Relational Algebra:
Π<ticket_id,fname,lname -> PassengerName,amount,train,t_source,t_destination>(ticket) ⋈**<**tc.pid = ps.pid**>(**passenger**)** ⋈**<**tc.tra_id = td.tra_id**>(**transaction_details**)**⋈**<**ti.train_id = tc.train_id**>(**train**)**

SQL Query:

```
select tc.ticket_id,ps.fname ||' '|| ps.lname as PassengerName, td.amount,
ti.train_id, ti.t_source, ti.t_destination from ticket tc inner join passenger ps on
tc.pid = ps.pid inner join transaction_details td on tc.tra_id = td.tra_id inner join
train ti on ti.train_id = tc.train_id;
```

10)Retrieve station’s name where food and waiting rooms facilities are available.

Relational Algebra:

Π<sid,station_name>(station) ***σ***<station.food_availablity=true AND station.waiting_room_availablity=true>

SQL QUERY:

```
SELECT station.sid,station.station_name
FROM station
WHERE ( station.food_availablity=true AND
station.waiting_room_availablity=true)
```

11)Retrive the ticket details of the passengers of the train named karnavati express.

Relational Algebra:
𝝈<train_name='KARNAVATI EXPRESS'>(train)⋈<ticket.train_id=train.train_id>(ticket) ⋈ ticket.pid=passenger.pid(passenger)

SQL QUERY:

```
select pa.* from passenger pa, train tr, ticket tc
where tc.train_id = tr.train_id and tc.pid = pa.pid and tr.train_name = 'KARNAVATI
EXPRESS';
```

12)count the number of tickets of train id 12115 and display the name of the train.

Relational Algebra:
Π<train_id,train_name>(**Π**<ticket_id>)(ticket)***σ***<tc.train_id = tr.train_id>
Π<train.train_id,train.train_name>(***σ***train_id=’12115’ (ticket ⋈**<**ticket.train_id = train.train_id> train **))**

SQL QUERY:

```
select tr.train_id, tr.train_name, (Select COUNT(tc.ticket_id) from ticket tc where
tc.train_id = tr.train_id)
as Total from train tr where tr.train_id = '12115';
```

13) Retrieve the passenger details who booked a ticket in train 12115.

Relational Algebra:
σ<train.train_id = '12115'>(train)⋈**<**train.train_id = ticket.train_id>(ticket)⋈**<**passenger.pid = ticket.pid>(passenger)

SQL QUERY:

```
SELECT passenger.* FROM passenger LEFT JOIN ticket ON passenger.pid =
ticket.pid LEFT JOIN train
ON train.train_id = ticket.train_id WHERE train.train_id = '12115';
```

14) Retrieve the transaction details and passenger details of the passenger who booked ticket of train id 12155

Relational Algebra:
Π<passenger>(transaction_details)(σtrain_id = ‘12155’ (ticket)
⋈<transaction_details.pid = ticket.pid> Transaction_details ⋈ **<**passenger.pid = transaction_details.pid> passengers))

SQL QUERY:

```
SELECT transaction_details,passenger.* FROM transaction_details right JOIN
passenger ON
passenger.pid = transaction_details.pid right JOIN ticket
ON transaction_details.pid = ticket.pid WHERE ticket.train_id = ‘12155’;
```

15) List out the station name of those stations where lost items are not found or food availability or waiting room availability is not there.

Relational Algebra:
Π<station_name,sid,food_availablity,waiting_room_availablity**>(**station**)**⋈ **<**station.sid=lost_and_found.sid**>(**lost_and_found**)** ***σ*<**lost_and_found.status='no' or station.food_availablity=false or station.waiting_room_availablity=false**>**

SQL QUERY:

```
select lost_and_found.*
,station.station_name,station.sid,station.food_availablity,station.waiting_room_availablity from station join lost_and_found on station.sid=lost_and_found.sid
where lost_and_found.status='no' or station.food_availablity=false  or
station.waiting_room_availablity=false;
```

16)show minimum age of employee and count number of employees whose age is greater than 18 and less than 60 count should be atleast one.

Relational Algebra:
Π<sid,MIN**> ->** minage (employee) $\boldsymbol{\sigma}_{\textbf{employee}}$**(**<age>= 18 AND age <=60>)$\mathscr{F}_{\textbf{count(*)>1}}$

SQL QUERY:
SELECT sid, MIN(age) AS minage FROM employee WHERE age >= 18 AND age <=60 GROUP BY sid HAVING COUNT (*) > 1;

17)count the number of lost items whose status is still not found.

Relational Algebra:
$\boldsymbol{\Pi}$**<**$\mathscr{F}_{\textbf{count<item\_id>}}$**,**item_name**>(**lost_and_found**)**$\boldsymbol{\sigma}_{\textbf{item\_id}}$(status='no')

SQL QUERY:
select count(item_id),item_name from lost_and_found where status='no' group by item_id;

18) Retrieve station name and id details with parcel service details and medical service details of patient name starts with x.

Relational Algebra:
Π<sid,station_name,pracel_id,sid>(station) ⋈
<station.sid=parcel_service.sid>(parcel_services) ⋈
<medical_services.sid=station.sid>(medical_services)
σ<medical_services.patient_name like 'X%'>

SQL QUERY:

```
Select station.sid,station.station_name,parcel_service.par_id,medical_services.*
from station join parcel_service on station.sid=parcel_service.sid
join medical_services on medical_services.sid=station.sid where
medical_services.patient_name like 'X%';
```

19)Retrieve the details of passenger and there ticket whose transaction amount is more than 500

Relational Algebra:

Π<amount,transaction_date_time**>**(passenger) ⋈ **<**passenger.pid=ticket.pid**>(**ticket**)**⋈**<**transaction_details.pid=ticket.pid**>(**transaction_details**)** ***σ*<**transaction_details.amount>500**>**

SQL QUERY:

```
select
passenger.*,ticket.*,transaction_details.amount,transaction_details.transaction_date_time
from passenger join ticket on passenger.pid=ticket.pid join transaction_details on transaction_details.pid=ticket.pid
where transaction_details.amount>500;
```

20)count the employees and average salary of employees of station id 1 or 2 or 4.

Relational Algebra:
$\Pi <\text{sid},\mathcal{F}_{\text{count}<\text{e\_id}>,\text{AVERAGE}}(\text{SALARY})>(\text{employee})\sigma_{\text{sid}}<\text{sid=1 or sid=2 or sid=4}>$

SQL QUERY:
SELECT sid, COUNT(e_id), AVG(SALARY) FROM employee WHERE sid=1 or sid=2 or sid=4 GROUP BY sid;

21)Find out employees of departments who draw more salary then average salary of all employees of that department.

Relational Algebra:
R1 -> π(department, ℱavg(salary)->avg_sal)(employee)
σ(e.salary>department_avg_sal.avg_sal) (ρ(e,employee)
⋈<e.department=department_avg_salary.department>
ρ(department_avg_salary,r1))

SQL QUERY:

```
SELECT * FROM employee AS e JOIN (
SELECT department, AVG(salary) AS avg_sal FROM employee
GROUP BY department
ORDER BY department
) AS department_avg_sal
ON (e.department = department_avg_sal.department)
WHERE e.salary > department_avg_sal.avg_sal;
```