

Date-September 2,2022

Name-Divya Kirtikumar Patel

Student ID-202001420

Lab Group- 6 Section - 8

1: Display ticket id for maximum ticket amount.

Ans:

```
SELECT ticket_id FROM ticket
WHERE amount = (SELECT MAX(amount) FROM ticket);
```

The screenshot shows a PostgreSQL query editor interface. The top bar indicates the connection to '202001420_db/postgres@PostgreSQL 14'. Below the toolbar, the 'Query' tab is active, displaying the following SQL code:

```
1 SET search_path to rail_db;
2
3 -- 1: Display ticket id for maximum ticket amount.
4 SELECT ticket_id FROM ticket
5 WHERE amount = (SELECT MAX(amount) FROM ticket);
6
```

Below the query editor, the 'Data output' tab is active, showing the result of the query. The result is a single row with the ticket ID 1000087.

ticket_id
1000087

2: Print the Name of female passengers in descending order of passenger_id.

Ans:

```
SELECT name, passenger_id FROM passengerdetails
WHERE gender = 'female'
ORDER BY passenger_id DESC;
```

The screenshot shows a PostgreSQL query editor interface. The query is as follows:

```
-- 2: Print the Name of female passengers in descending order of passenger_id.
SELECT name, passenger_id FROM passengerdetails
WHERE gender = 'female'
ORDER BY passenger_id DESC;
```

The results are displayed in a table with two columns: name and passenger_id. The data is sorted by passenger_id in descending order.

	name	passenger_id
1	Iola	348
2	Kenneth	346
3	Hyatt	345
4	Cameron	343
5	Deirdre	340
6	Channing	339
7	Amethyst	338
8	Micah	337
9	Brenda	336
10	Thane	335
11	Xandra	333
12	Myles	328

At the bottom, a status bar indicates "Total rows: 134 of 134" and "Query complete 00:00:00.083". A green message box on the right says "Successfully run. Total query r".

3: Print ticket id of the highest 2 amounts of tickets.

Ans:

```
SELECT ticket_id from ticket
ORDER BY amount DESC LIMIT 2;
```

The screenshot shows a PostgreSQL query editor interface. The query is as follows:

```
-- 3: Print ticket id of the highest 2 amounts of tickets.
SELECT ticket_id from ticket
ORDER BY amount DESC LIMIT 2;
```

The results are displayed in a table with one column: ticket_id. The data is sorted by amount in descending order.

	ticket_id
1	1000087
2	1000039

At the bottom, a status bar indicates "Query complete 00:00:00.083".

4: Print count of passengers who did the payment of food in cash.

Ans:

```
SELECT COUNT(DISTINCT(passenger_id))
FROM pantry WHERE payment_mode = 'cash';
```

The screenshot shows a PostgreSQL query editor interface. The query is: `SELECT COUNT(DISTINCT(passenger_id)) FROM pantry WHERE payment_mode = 'cash';`. The query is highlighted in blue. Below the query, there are tabs for 'Data output', 'Messages', and 'Notifications'. The 'Data output' tab is active, showing a table with two columns: 'count' and 'bigint'. The table has one row with the value '4'.

count	bigint
1	4

5: Print Passenger id and gender of the passengers whose ticket amount is greater than avg amount value.

Ans:

```
SELECT passenger_id ,gender from passengerdetails NATURAL JOIN ticket
WHERE amount > (SELECT AVG(amount) FROM ticket);
```

The screenshot shows a PostgreSQL query editor interface. The query is: `SELECT passenger_id ,gender from passengerdetails NATURAL JOIN ticket WHERE amount > (SELECT AVG(amount) FROM ticket);`. The query is highlighted in blue. Below the query, there are tabs for 'Data output', 'Messages', and 'Notifications'. The 'Data output' tab is active, showing a table with two columns: 'passenger_id' and 'gender'. The table has 10 rows of data.

passenger_id	gender
99	male
100	female
102	male
108	male
109	female
111	female
112	male
115	female
117	female
119	female

Successfully run. Total query runtime: 00:00:00.137

6: Find the station id and the train name arriving at that station.

Ans:

```
SELECT station_id, train_name FROM rail_db.train JOIN rail_db.station on
station.train_id = train.train_id;
```

```
4
5 -- 6: Find the station id and the train name arriving at that station.
6 SELECT station_id, train_name FROM rail_db.train JOIN rail_db.station on station.train
7
8 -- 7: Print the count of different types of transactions
9 SELECT COUNT(payment_mode), payment_mode FROM pantry
0 GROUP BY payment_mode;
1
2 -- 8: Find the id of ticket collectors whose age is less than avg age.
3 SELECT tc_id FROM ticketcollector
```

Data output Messages Notifications

station_id	train_name
1	elementum pu...
2	Suspendisse ...
3	di luctus lobor...
4	fermentum m...
5	Integer mollis...
6	enim condime

Total rows: 103 of 103 Query complete 00:00:00.565 Successfully run. Total que

7: Print the count of different types of transactions

Ans:

```
SELECT COUNT(payment_mode), payment_mode FROM pantry
GROUP BY payment_mode;
```

202001420_db/postgres@PostgreSQL 14

Query Query History

```
28 -- 7: Print the count of different types of transactions
29 SELECT COUNT(payment_mode), payment_mode FROM pantry
30 GROUP BY payment_mode;
31
```

Data output Messages Notifications

count	payment_mode
4	cash
6	online

8: Find the id of ticket collectors whose age is less than avg age.

Ans:

```
SELECT tc_id FROM ticketcollector
WHERE age < (SELECT AVG(age) FROM ticketcollector);
```

The screenshot shows a PostgreSQL query editor interface. The query is: `SELECT tc_id FROM ticketcollector WHERE age < (SELECT AVG(age) FROM ticketcollector);`. The results are displayed in a table with 13 rows. A status bar at the bottom indicates 'Total rows: 13 of 13' and 'Query complete 00:00:00.067'. A green 'Successful' message is visible on the right.

tc_id [PK] integer
1
2
3
4
5
6
7
8
9
10
11
12
13

9: Find the names of trains which are taking a halt at a station and sort in order of station id.

Ans:

```
SELECT train_name FROM train JOIN station ON station.train_id = train.train_id
WHERE halt = 'Yes' ORDER BY station.station_id DESC;
```

The screenshot shows a PostgreSQL query editor interface. The query is: `SELECT train_name FROM train JOIN station ON station.train_id = train.train_id WHERE halt = 'Yes' ORDER BY station.station_id DESC;`. The results are displayed in a table with 13 rows. A status bar at the bottom indicates 'Total rows: 56 of 56' and 'Query complete 00:00:00.067'. A green 'Successful' message is visible on the right.

train_name character (50)
1 nulla at sem ...
2 lacus
3 sodales at
4 Cras vehicula ...
5 eu tempor era...
6 tempor
7 aliquet sem
8 arcu
9 tellus faucibu...
10 elit pellentesq...
11 sociis natoque
12 est. Nunc laor...
13 vel sapien imp...

10: Create a view of the maximum amount.

Ans:

```
CREATE or REPLACE VIEW maximum_amount AS (SELECT max(amount) from rail_db.ticket);  
SELECT * from maximum_amount
```

Query

```
40 -- 10: Create a view of the maximum amount.  
41 CREATE or REPLACE VIEW maximum_amount AS (SELECT max(amount) from rail_db.ticket);  
42 SELECT * from maximum_amount  
43  
44 -- 11: Create a view for station details then insert and display some new station ids to the
```

Data output

	max integer
1	4984

11: Create a view for station details then insert and display some new station ids to the recently created view.

Ans:

```
CREATE OR REPLACE view station_details AS (select * from rail_db.station);  
insert into station_details VALUES('101','ASD_EFT','08:21:00 PM', '101', 'Yes') ;  
insert into station_details VALUES('106','HUTRYF','10:21:00 PM', '109', 'No') ;  
insert into station_details VALUES('103','NUGPYF','10:01:00 PM', '104', 'No') ;  
select * from station_details
```

```
44 -- 11: Create a view for station details then insert and display some new station ids  
45 -- recently created view.  
46 CREATE OR REPLACE view station_details AS (select * from rail_db.station);  
47 -- select * from station_details;  
48 insert into station_details VALUES('101','ASD_EFT','08:21:00 PM', '101', 'Yes') ;  
49 insert into station_details VALUES('106','HUTRYF','10:21:00 PM', '109', 'No') ;  
50 insert into station_details VALUES('103','NUGPYF','10:01:00 PM', '104', 'No') ;  
51  
52 select * from station_details  
53
```

Data output

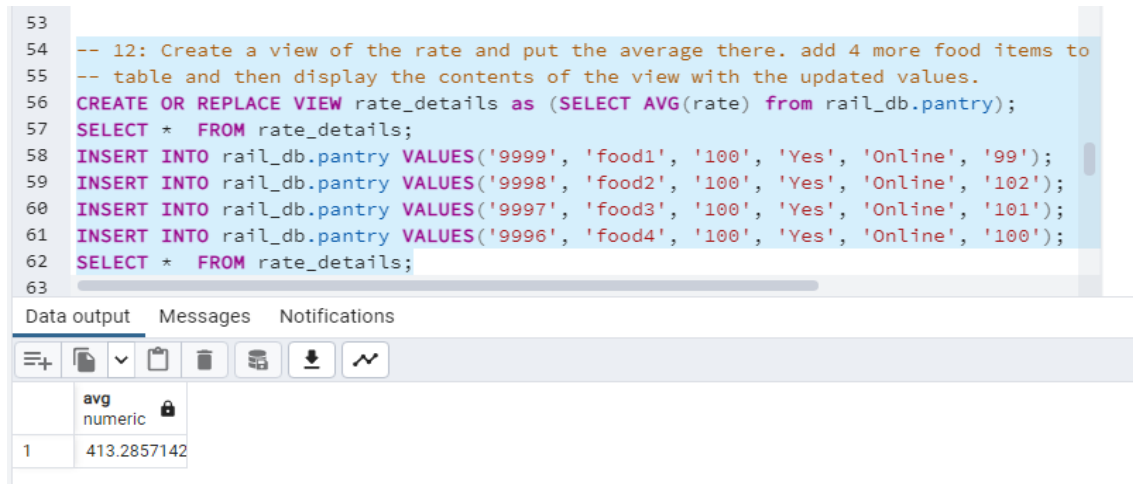
	station_id integer	name character (50)	arrival_time time without time zone	train_id integer	hault character (30)
98	98	PUS 2F9	20:43:00	198	NO
99	99	W6P 5U8	16:53:00	199	Yes
100	100	J5C 4K7	05:23:00	200	No
101	101	ASD_EFT	20:21:00	101	Yes
102	106	HUTRYF	22:21:00	109	No
103	103	NUGPYF	22:01:00	104	No

Total rows: 103 of 103 Query complete 00:00:00.161

12: Create a view of the rate and put the average there. add 4 more food items to the pantry table and then display the contents of the view with the updated values

Ans:

```
CREATE OR REPLACE VIEW rate_details as (SELECT AVG(rate) from rail_db.pantry);
SELECT * FROM rate_details;
INSERT INTO rail_db.pantry VALUES('9999', 'food1', '100', 'Yes', 'Online', '99');
INSERT INTO rail_db.pantry VALUES('9998', 'food2', '100', 'Yes', 'Online', '102');
INSERT INTO rail_db.pantry VALUES('9997', 'food3', '100', 'Yes', 'Online', '101');
INSERT INTO rail_db.pantry VALUES('9996', 'food4', '100', 'Yes', 'Online', '100');
SELECT * FROM rate_details;
```

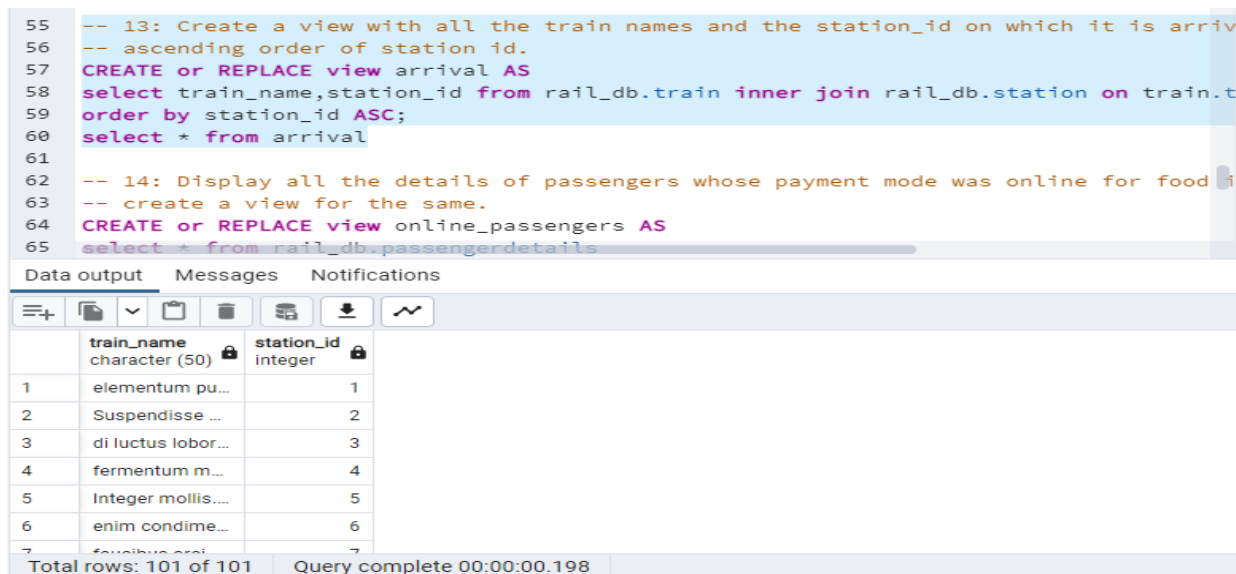


	avg numeric
1	413.2857142

13: Create a view with all the train names and the station_id on which it is arriving in ascending order of station id.

Ans:

```
CREATE or REPLACE view arrival AS
select train_name,station_id from rail_db.train inner join rail_db.station on
train.train_id = station.train_id
order by station_id ASC;
select * from arrival
```



	train_name character (50)	station_id integer
1	elementum pu...	1
2	Suspendisse ...	2
3	di luctus lobor...	3
4	fermentum m...	4
5	Integer mollis....	5
6	enim condime...	6
7	faucibus erat...	7

Total rows: 101 of 101 Query complete 00:00:00.198

14: Display all the details of passengers whose payment mode was online for food items and create a view for the same.

Ans:

```
CREATE or REPLACE view online_passengers AS
select * from rail_db.passengerdetails
where passenger_id in ( select passenger_id from rail_db.pantry where
payment_mode = 'online' );
select * from online_passengers;
```

```
62 -- 14: Display all the details of passengers whose payment mode was online for food i
63 -- create a view for the same.
64 CREATE or REPLACE view online_passengers AS
65 select * from rail_db.passengerdetails
66 where passenger_id in ( select passenger_id from rail_db.pantry where payment_mode =
67
68 select * from online_passengers;
69
70 -- 15: Display the name, ticket id and age of the passengers who are travelling in
```

	passenger_id integer	pnr_no integer	age integer	gender character (30)	seat_no integer	ticket_id integer	name character (50)	reservation_status character (40)	train_id integer
1	107	207	59	male	9	1000009	Giselle	confirm	109
2	102	202	41	male	4	1000004	Ina	confirm	104
3	101	201	77	female	3	1000003	Levi	confirm	103
4	99	199	34	male	1	1000001	Serena	confirm	101
5	108	208	49	male	10	1000010	Tate	waiting	110
6	104	204	75	female	6	1000006	Hadassah	waiting	106

Total rows: 6 of 6 Query complete 00:00:00.170 ✓ Successfully run. Total query runtime: 170 msec. 6 rows aff

15: Display the name, ticket id and age of the passengers who are travelling in a train with the number of seats available is greater than 10 in decreasing order of the ticket id.

Ans:

```
CREATE or replace view passenger_train AS
select name,ticket_id,age from rail_db.passengerdetails where train_id in (
    select train_id from rail_db.train where seats_available>10 )
order by ticket_id desc;
select * from passenger_train;
```

```
70 -- 15: Display the name, ticket id and age of the passengers who are travelling in a
71 -- the number of seats available is greater than 10 in decreasing order of the ticket
72 CREATE or replace view passenger_train AS
73 select name,ticket_id,age from rail_db.passengerdetails where train_id in (
74     select train_id from rail_db.train where seats_available>10
75 )
76 order by ticket_id desc;
77
78 select * from passenger_train;
79
```

	name character (50)	ticket_id integer	age integer
121	Chadwick	1000010	55
122	Linus	1000012	71
123	Tate	1000010	49
124	Giselle	1000009	59
125	Prescott	1000007	51
126	Ina	1000004	41
127	Philip	1000002	56

Total rows: 127 of 127 Query complete 00:00:00.093