

## BDA ASSIGNMENT - 1

Name - Tanishq (2018199), Vipul (2018118)

### QUERIES -

1( a).

```
select date,event,count(pull_requestid)"No. of pull request"  
from PUBLIC.pull_request  
group by event,date  
having event='opened'  
order by date;
```

postgres/postgres@PostgreSQL 13

Query Editor

Query History

1

select date,event,count(pull\_requestid)"No. of pull request"

2

from PUBLIC.pull\_request

3

group by event,date

4

having event='opened'

5

order by date;

6

Data Output

Explain

Messages

Notifications

	date timestamp without time zone	event character (50)	No. of pull request bigint	
1	2010-09-02 00:00:00	opened	2	
2	2010-09-06 00:00:00	opened	1	
3	2010-09-08 00:00:00	opened	1	
4	2010-09-09 00:00:00	opened	4	
5	2010-09-10 00:00:00	opened	3	
6	2010-09-11 00:00:00	opened	3	
7	2010-09-12 00:00:00	opened	3	
8	2010-09-13 00:00:00	opened	3	
9	2010-09-15 00:00:00	opened	2	
10	2010-09-16 00:00:00	opened	2	
11	2010-09-18 00:00:00	opened	6	
12	2010-09-19 00:00:00	opened	4	
13	2010-09-20 00:00:00	opened	2	

### Methodology:

Here we selected three columns named date,events which contain the status (example : opened , discussed , merged) and no.of pull request, we counted all the pull requests of every date which was having the comment opened.

1 (b).

```
select date,event,count(pull_requestid)"No. of pull request"
from PUBLIC.pull_request
group by event,date
having event='discussed'
order by date;
```

postgres/postgres@PostgreSQL 13

Query Editor

Query History

1

2

3

4

5

6

```
select date,event,count(pull_requestid)"No. of pull request"
from PUBLIC.pull_request
group by event,date
having event='discussed'
order by date;
```

Data Output

Explain

Messages

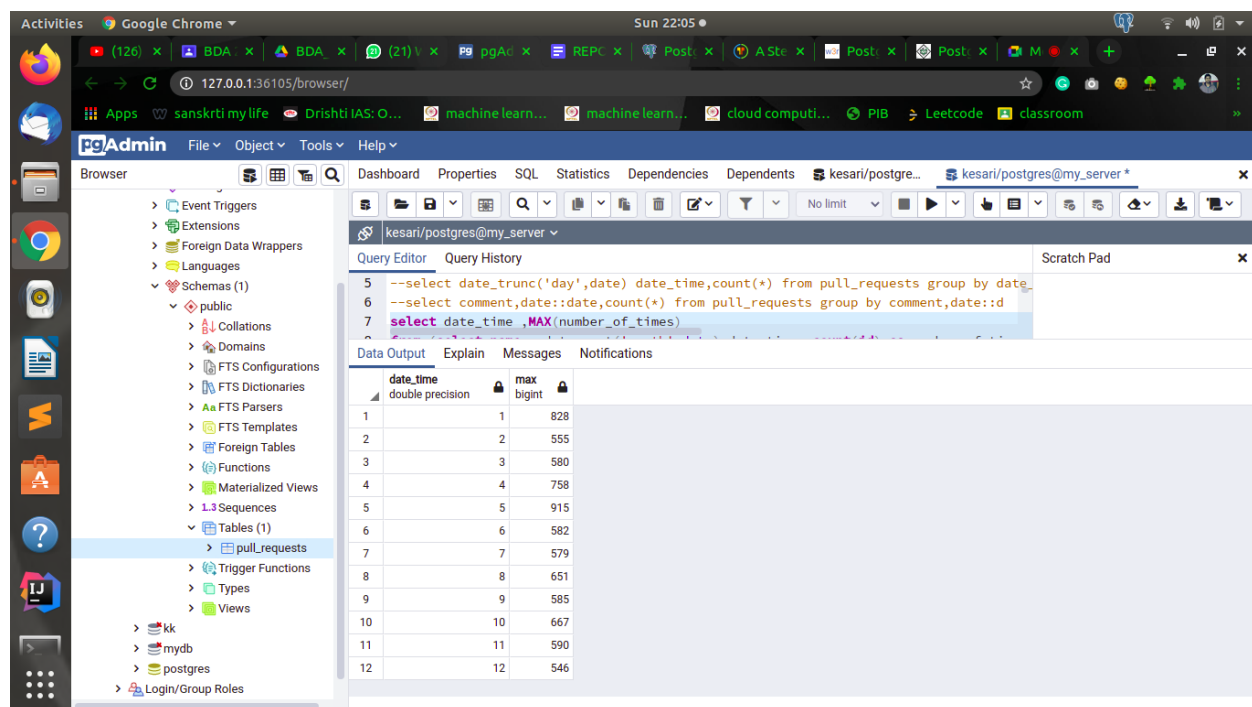
Notifications

	date timestamp without time zone	event character (50)	No. of pull request bigint
1	2010-09-09 00:00:00	discussed	6
2	2010-09-10 00:00:00	discussed	10
3	2010-09-11 00:00:00	discussed	13
4	2010-09-12 00:00:00	discussed	5
5	2010-09-13 00:00:00	discussed	7
6	2010-09-14 00:00:00	discussed	1
7	2010-09-15 00:00:00	discussed	3
8	2010-09-16 00:00:00	discussed	2
9	2010-09-17 00:00:00	discussed	1
10	2010-09-21 00:00:00	discussed	6
11	2010-09-22 00:00:00	discussed	3
12	2010-09-23 00:00:00	discussed	5
13	2010-09-24 00:00:00	discussed	3
14	2010-09-25 00:00:00	discussed	5

**Methodology:** Here we selected three columns named date, event which contains the status (example : opened , discussed , merged) and no.of pull request, we counted all the pull requests of every date which was having the comment discussed.

2.

```
select date_time ,MAX(number_of_times)
from (select name , date_part('month',date) date_time ,count(id) as number_of_times
      from pull_requests
      where comment='discussed'
      group by name, date_time) as deliveer
group by date_time order by date_time;
```



The screenshot shows the pgAdmin interface with a SQL query executed in the Query Editor. The query is a nested select statement designed to find the maximum number of pull requests for a specific comment ('discussed') grouped by month and then ordered by date. The results are displayed in the Data Output tab.

date_time	max
1	828
2	555
3	580
4	758
5	915
6	582
7	579
8	651
9	585
10	667
11	590
12	546

**Methodology:-** Here i use the nested select command to get the result. I first counted the comments in a month and then i counted the max among them.

3.

```
select date_time ,MAX(number_of_times)
from (select name , date_part('week',date) date_time ,count(id) as number_of_times
      from pull_requests
      where comment='discussed'
      group by name, date_time) as deliveer
group by date_time order by date_time;
```

The screenshot shows the pgAdmin 4 interface with the following SQL query in the Query Editor:

```

4 --select comment,date_trunc('month',date) date_time,count(*) from pull_requests grou
5 --select date_trunc('day',date) date_time,count(*) from pull_requests group by date_
6 --select comment,date::date,count(*) from pull_requests group by comment,date::d
7 select date_time ,MAX(number_of_times)
8 from (select name , date_part('week',date) date_time ,count(1d) as number_of_times
9 from pull_requests
10 where comment='discussed'
11 group by name, date_time) as deliver
12 group by date_time order by date_time;
13
14
15

```

The Data Output pane displays the following results:

date_time	max
1	298
2	134

The screenshot shows the pgAdmin 4 interface with the following SQL query in the Query Editor:

```

4 --select comment,date_trunc('month',date) date_time,count(*) from pull_requests grou
5 --select date_trunc('day',date) date_time,count(*) from pull_requests group by date_
6 --select comment,date::date,count(*) from pull_requests group by comment,date::d
7 select date_time ,MAX(number_of_times)

```

The Data Output pane displays the following results:

date_time	max
1	298
2	134
3	119
4	210
5	158
6	124
7	141
8	150
9	151

**Methodology:-** Here i use the nested select command to get the result. I first counted the comments in a month and then i counted the max among them.

4.

*SELECT date\_trunc('week',date) weeks,count(pull\_requestid)"No. of pull request"  
from public.pull\_request*

*group by event, weeks  
having event='opened'  
order by weeks;*

## Output -

postgres/postgres@PostgreSQL 13

Query Editor

Query History

```
1 SELECT date_trunc('week',date) weeks,count(pull_requestid)"No. of pull request"
2 from public.pull_request
3 group by event,weeks
4 having event='opened'
5 order by weeks;
6
```

Data Output

Explain

Messages

Notifications

	<div>weeks</div> <div>timestamp without time zone</div>	<div>No. of pull request</div> <div>bigint</div>
1	2010-08-30 00:00:00	2
2	2010-09-06 00:00:00	15
3	2010-09-13 00:00:00	17
4	2010-09-20 00:00:00	17
5	2010-09-27 00:00:00	13
6	2010-10-04 00:00:00	10
7	2010-10-11 00:00:00	5
8	2010-10-18 00:00:00	5
9	2010-10-25 00:00:00	3
10	2010-11-01 00:00:00	4
11	2010-11-08 00:00:00	9
12	2010-11-15 00:00:00	8
13	2010-11-22 00:00:00	9
14	2010-11-29 00:00:00	6

**Methodology:** Here I used the `date_trunc()` function to convert all the dates in a week, I imported that column with the name of date and truncated it to weeks. I checked for all the comments which were opened per week and counted that.

5.

```
select date_trunc('month',date) months,count(pull_requestid)"No. of pull request"
from public.pull_request
where event='merged' and EXTRACT(YEAR FROM date)='2010'
group by months
order by months;
```

**Output-**

postgres/postgres@PostgreSQL 13

Query Editor

Query History

1

select date\_trunc('month',date) months,count(pull\_requestid)"No. of pull request"

2

from public.pull\_request

3

where event='merged' and EXTRACT(YEAR FROM date)='2010'

4

group by months

5

order by months;

6

Data Output

Explain

Messages

Notifications

months

timestamp without time zone

No. of pull request

bigint

postgres/postgres@PostgreSQL 13 ▾

Query Editor

Query History

```
1 select date_trunc('month',date) months,count(pull_requestid)"No. of pull request"
2 from public.pull_request
3 where event='merged' and EXTRACT(YEAR FROM date)='2015'
4 group by months
5 order by months;
6
```

Data Output

Explain

Messages

Notifications

	<div>months</div> <div>timestamp without time zone</div>	<div>No. of pull request</div> <div>bigint</div>
1	2015-01-01 00:00:00	28
2	2015-02-01 00:00:00	2
3	2015-03-01 00:00:00	27
4	2015-04-01 00:00:00	11
5	2015-05-01 00:00:00	37
6	2015-06-01 00:00:00	17
7	2015-07-01 00:00:00	81
8	2015-08-01 00:00:00	132
9	2015-09-01 00:00:00	148
10	2015-10-01 00:00:00	8
11	2015-11-01 00:00:00	17
12	2015-12-01 00:00:00	2

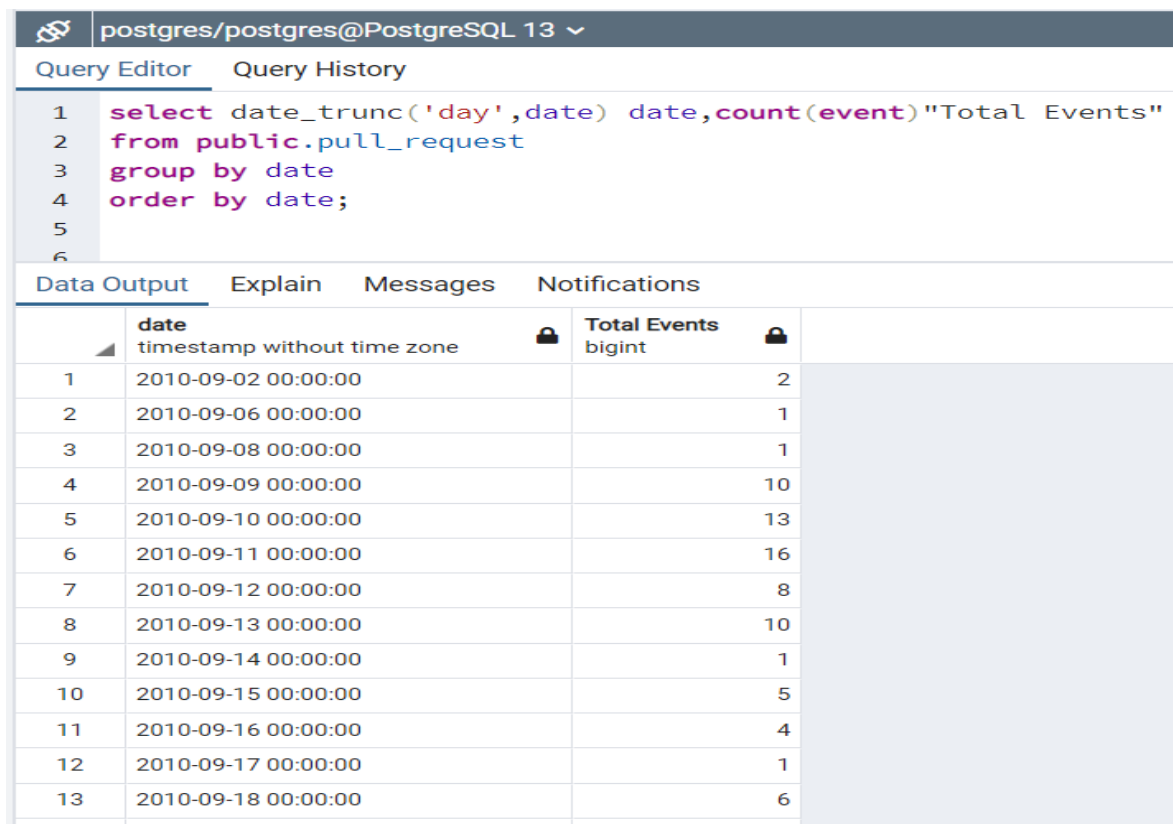
**Methodology:** Here I used the `date_trunc()` function to convert all the dates as month. I also used the `extract()` function to extract the year from date and then I checked for the year 2010. And for the comments that are merged.

**Observation:-** There were not any merged pull requests in the year 2010 so that gave me nothing in the output but when I checked for the year 2015 that time I got the above output.

6.

```
select date_trunc('day',date) date,count(event)"Total Events"  
from public.pull_request  
group by date  
order by date;
```

**Output -**



The screenshot shows a PostgreSQL query editor interface. At the top, there's a tab labeled "postgres/postgres@PostgreSQL 13". Below it, there are two tabs: "Query Editor" and "Query History". The "Query Editor" tab is active, displaying a SQL query. The query is as follows:

```
1 select date_trunc('day',date) date,count(event)"Total Events"  
2 from public.pull_request  
3 group by date  
4 order by date;  
5  
6
```

Below the query editor, there are four tabs: "Data Output", "Explain", "Messages", and "Notifications". The "Data Output" tab is active, showing the results of the query. The results are displayed in a table with two columns: "date" (timestamp without time zone) and "Total Events" (bigint). The table contains 13 rows of data, representing dates from 2010-09-02 to 2010-09-18.

	date timestamp without time zone	Total Events bigint
1	2010-09-02 00:00:00	2
2	2010-09-06 00:00:00	1
3	2010-09-08 00:00:00	1
4	2010-09-09 00:00:00	10
5	2010-09-10 00:00:00	13
6	2010-09-11 00:00:00	16
7	2010-09-12 00:00:00	8
8	2010-09-13 00:00:00	10
9	2010-09-14 00:00:00	1
10	2010-09-15 00:00:00	5
11	2010-09-16 00:00:00	4
12	2010-09-17 00:00:00	1
13	2010-09-18 00:00:00	6

**Methodology:** Here I used the `date_trunc()` function to convert all the dates as days. And then group by dates all events are counted.

7.

```
select distinct(name) ,count(*)  
from pull_requests  
where comment='opened' and EXTRACT(YEAR FROM date)='2011'  
group by name  
order by count desc;
```

## Output -

The screenshot shows the pgAdmin 4 web interface in a Google Chrome browser. The left sidebar displays the database structure, including Schemas (1), public, and Tables (1). The main panel shows the Query Editor with the following SQL query:

```
--copy public.pull_requests from '/home/vipul/Downloads/pullreq_events.csv' with csv  
--select comment,date::date,count(*) from pull_requests group by comment,date::date  
--select comment,date_trunc('month',date) date_time,count(*) from pull_requests grou  
--select date_trunc('day',date) date_time,count(*) from pull_requests group by date  
select distinct(name) ,count(*)  
from pull_requests  
where comment='opened' and EXTRACT(YEAR FROM date)='2011'  
group by name  
order by count desc;
```

The Query History tab shows the query was executed successfully. The Data Output tab displays the results in a table:

	name character varying (50)	count bigint
1	arunagw	228
2	gullelgwaran	118
3	dmathieu	58
4	sikachu	58
5	amatsuda	49
6	lest	49
7	joshk	46

**Methodology:** Here I counted all the person who had opened a pull request in the year 2011. And for getting the person with highest request i just made it in the decreasing order so i got the output as **"arunagw" 228** . Here i used extract function to extract the year from the date column.

## Output -

## Learning:

1. Firstly I was able to revise my dbms course concept.
2. I got to know how to work with postgresql and pgadmin4.



3. I got to know some amazing functions named `date_trunc`, `datepart`, `extract`.
4. I got to know how to import the whole csv file into the db.