BDA Assignment-1 Report

Garvita Jain 2018032 | Arnav Tandon 2018278

Table Schema :
As given in the pullreq_events dataset we have taken the following naming for the assumptions for the csv file provided to us.

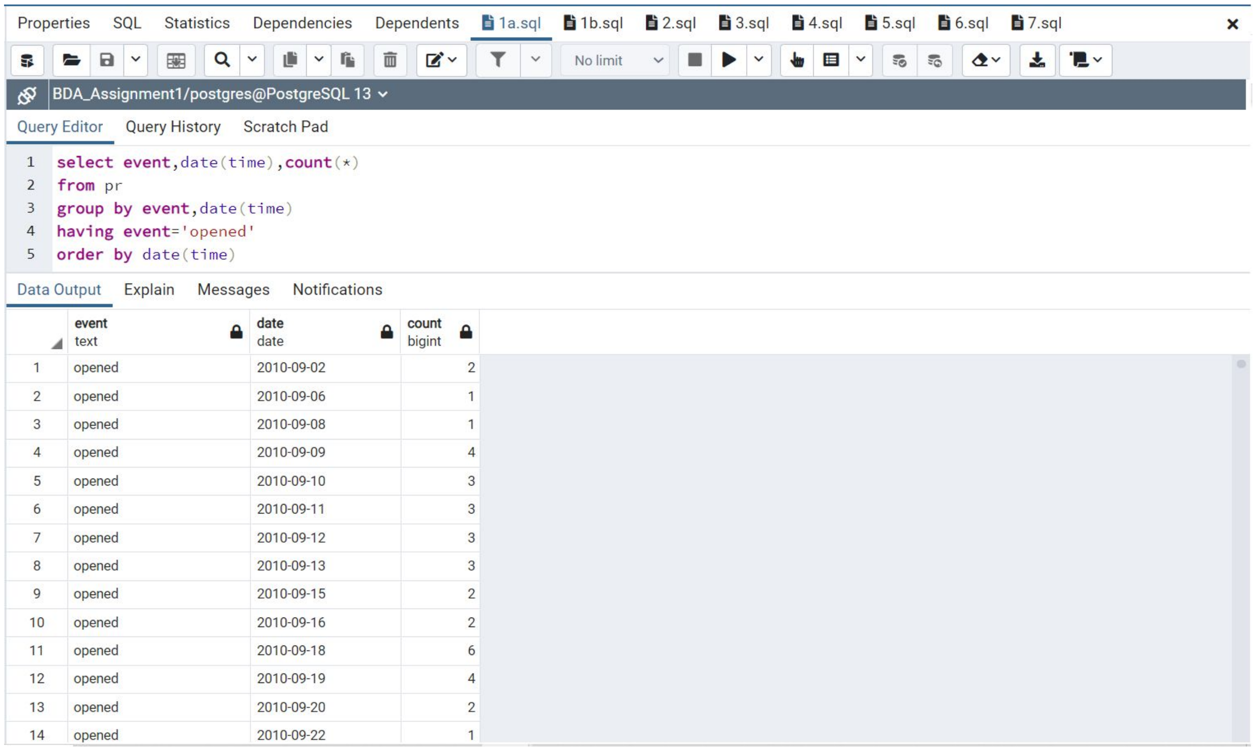
- 1st column : pull_requestId
- 2nd column : author
- 3rd column : event
- 4th column : time

1) Report the number of pull requests
1a) "opened" per day

Ans)
Query :

```
select event,date(time),count(*)
from pr
group by event,date(time)
having event='opened'
order by date(time)
```

Methodology :
To report the number of pull requests "opened" each day, we selected all such data points which had the event value as "opened" and counted their instances grouped by the days. The date of each timestamp was extracted using the date() function for timestamp variables. The output was arranged in increasing order of dates.


Properties SQL Statistics Dependencies Dependents 1a.sql 1b.sql 2.sql 3.sql 4.sql 5.sql 6.sql 7.sql

BDA_Assignment1/postgres@PostgreSQL 13

Query Editor Query History Scratch Pad

```
select event,date(time),count(*)
from pr
group by event,date(time)
having event='opened'
order by date(time)
```

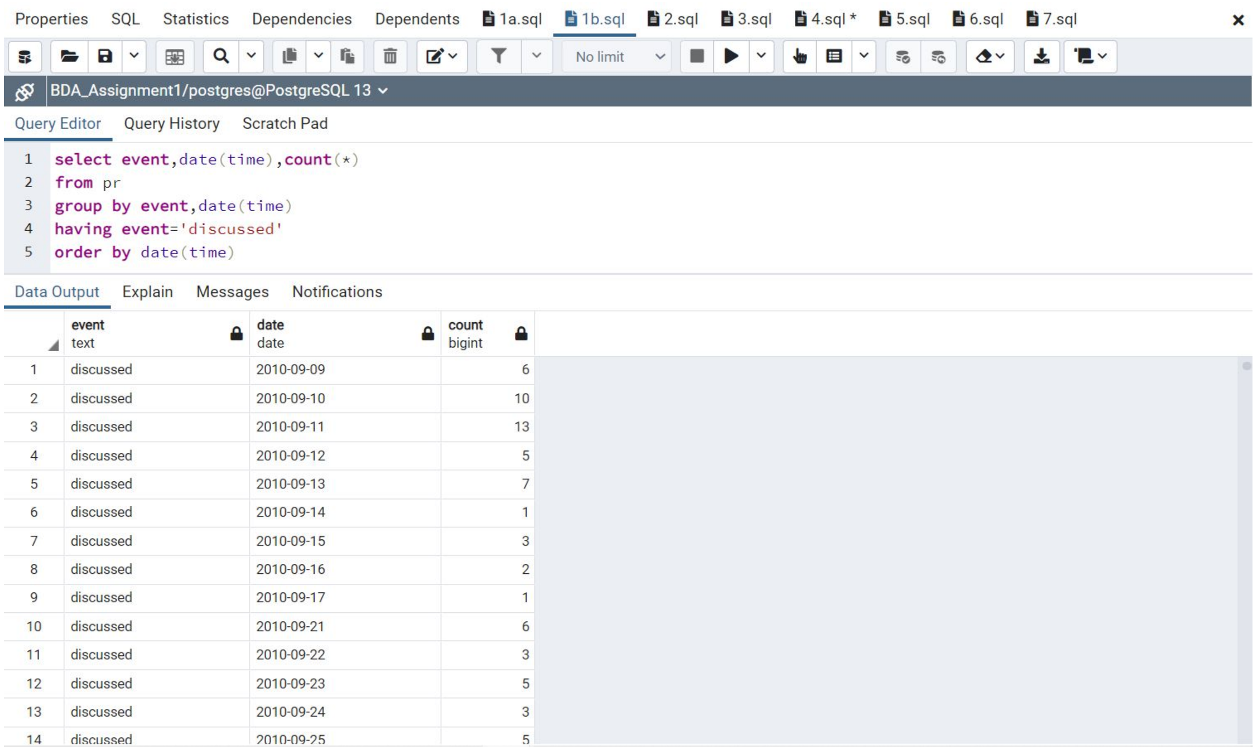
Data Output Explain Messages Notifications

	event text	date date	count bigint
1	opened	2010-09-02	2
2	opened	2010-09-06	1
3	opened	2010-09-08	1
4	opened	2010-09-09	4
5	opened	2010-09-10	3
6	opened	2010-09-11	3
7	opened	2010-09-12	3
8	opened	2010-09-13	3
9	opened	2010-09-15	2
10	opened	2010-09-16	2
11	opened	2010-09-18	6
12	opened	2010-09-19	4
13	opened	2010-09-20	2
14	opened	2010-09-22	1

1b) “discussed” per day

Ans)
Query :
select event,date(time),count(*)
from pr
group by event,date(time)
having event='discussed'
order by date(time)

Methodology :
To report the number of pull requests “discussed” each day, we selected all such data points which had the event value as “discussed” and counted their instances grouped by the days. The date of each timestamp was extracted using the date() function for timestamp variables. The output was arranged in increasing order of dates.



```
select event,date(time),count(*)
from pr
group by event,date(time)
having event='discussed'
order by date(time)
```

	event text	date date	count bigint
1	discussed	2010-09-09	6
2	discussed	2010-09-10	10
3	discussed	2010-09-11	13
4	discussed	2010-09-12	5
5	discussed	2010-09-13	7
6	discussed	2010-09-14	1
7	discussed	2010-09-15	3
8	discussed	2010-09-16	2
9	discussed	2010-09-17	1
10	discussed	2010-09-21	6
11	discussed	2010-09-22	3
12	discussed	2010-09-23	5
13	discussed	2010-09-24	3
14	discussed	2010-09-25	5

2) Output the person who has the highest number of comments per month. Assume a month has 30 days. The comments refer to events of the type "discussed".

Ans 2)
Query Assumption 1 :
select a.name,b.monthnumber,b.countscore
from(select hello.name, max(events) even, hello.monthnumber
from (select author as name,to_char(time, 'MM') monthnumber, count(event) events
from pr where event='discussed'
group by to_char(time, 'MM'), author) as hello

```
group by hello.monthnumber,hello.name
order by monthnumber,even desc)as a,
(select hello.monthnumber, max(events) countscore
from (select to_char(time, 'MM') monthnumber, count(event) events
from pr where event='discussed'
group by to_char(time, 'MM'), author) as hello
group by hello.monthnumber)as b
where a.monthnumber=b.monthnumber and a.even=b.countscore
```

Query Assumption 2 :

```
select a.name,b.monthnumber,b.countscore
from(select hello.name, max(events) even, hello.monthnumber
from (select author as name,to_char(time, 'YYYY-MM') monthnumber, count(event) events
from pr where event='discussed'
group by to_char(time, 'YYYY-MM'), author) as hello
group by hello.monthnumber,hello.name
order by monthnumber,even desc)as a,
(select hello.monthnumber, max(events) countscore
from (select to_char(time, 'YYYY-MM') monthnumber, count(event) events
from pr where event='discussed'
group by to_char(time, 'YYYY-MM'), author) as hello
group by hello.monthnumber)as b
where a.monthnumber=b.monthnumber and a.even=b.countscore
```

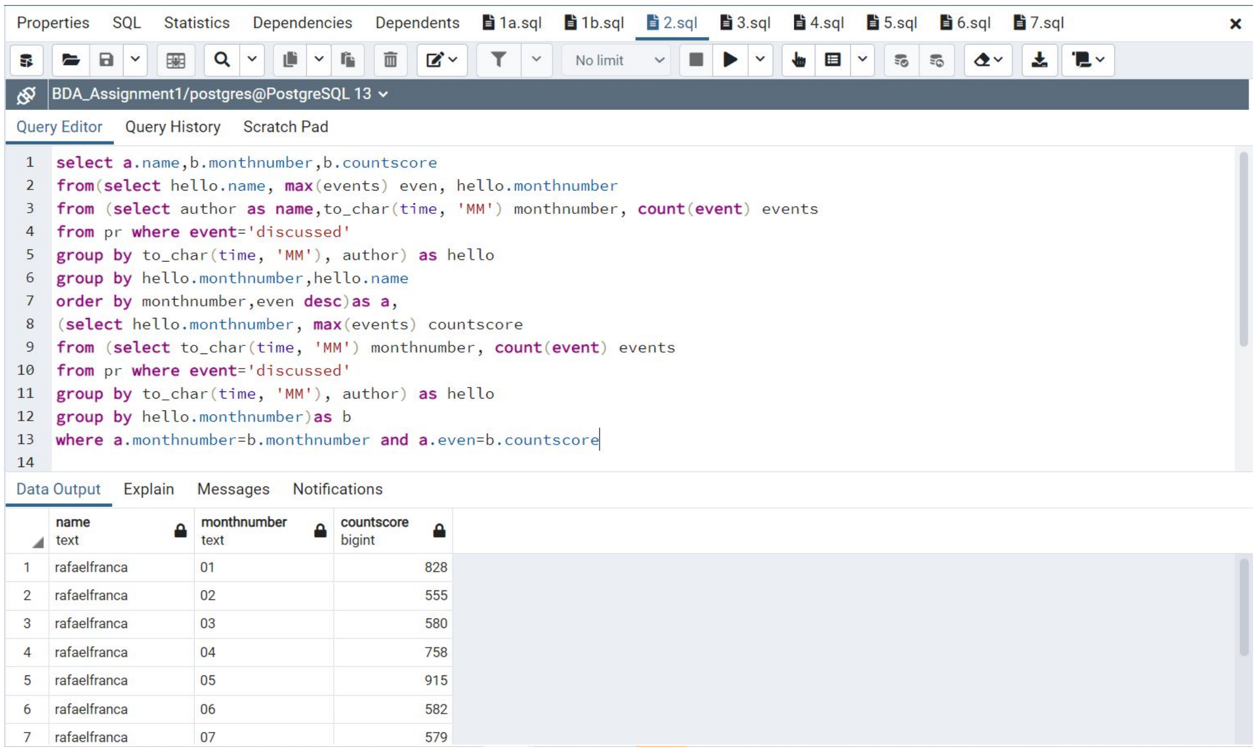
Methodology: All information is collected from by creating two tables as follows:
One table consists of the maximum count of discussed PRs for each month which is calculated from another SELECT subquery which calculates the count of discussed PRs for every month grouped by the months and author name. This table is assigned alias “a”.

The second table contains the list of maximum count of discussed PRs for every author grouped by month which is calculated from a SELECT subquery which stores the count of these PRs grouped by the months and then author name. This table is assigned alias “b”.

Our final query uses the above two queries as sub-queries to select the authors which have the maximum count of PRs for every month by matching the monthnumber and countscore columns of both tables.

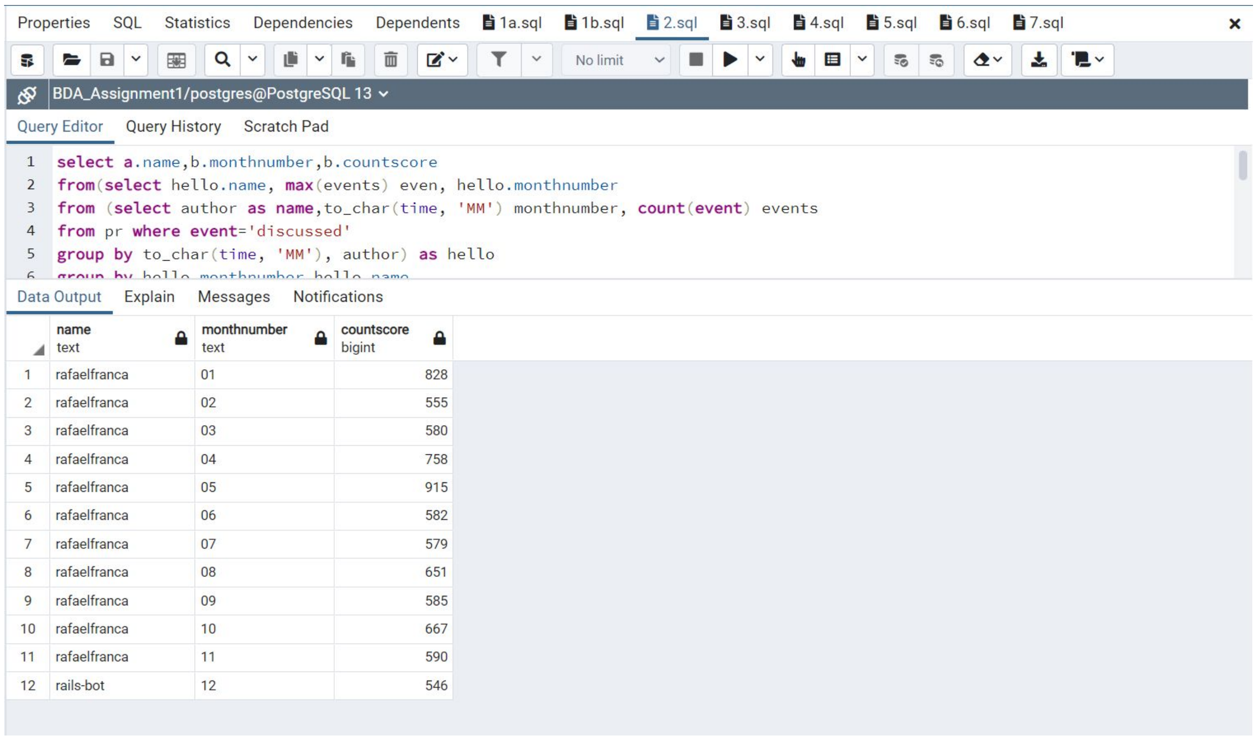
In assumption 1 we are taking to_char(time,’MM’) where we are showing the cumulative result for the same month but in different years together whereas in assumption 2 we are taking to_char(time,’YYYY-MM’) for the same month but different years separately.

Assumption 1: Count of highest number of “discussed” PRs is calculated by including counts of the same month but in different years together. For example, counts of March 2010 and March 2011 will be counted together for the month of March.


Properties SQL Statistics Dependencies Dependents 1a.sql 1b.sql 2.sql 3.sql 4.sql 5.sql 6.sql 7.sql
No limit
BDA_Assignment1/postgres@PostgreSQL 13
Query Editor Query History Scratch Pad
Data Output Explain Messages Notifications

```sql
select a.name,b.monthnumber,b.countscore
from(select hello.name, max(events) even, hello.monthnumber
from (select author as name,to_char(time, 'MM') monthnumber, count(event) events
from pr where event='discussed'
group by to_char(time, 'MM'), author) as hello
group by hello.monthnumber,hello.name
order by monthnumber,even desc)as a,
(select hello.monthnumber, max(events) countscore
from (select to_char(time, 'MM') monthnumber, count(event) events
from pr where event='discussed'
group by to_char(time, 'MM'), author) as hello
group by hello.monthnumber)as b
where a.monthnumber=b.monthnumber and a.even=b.countscore
```

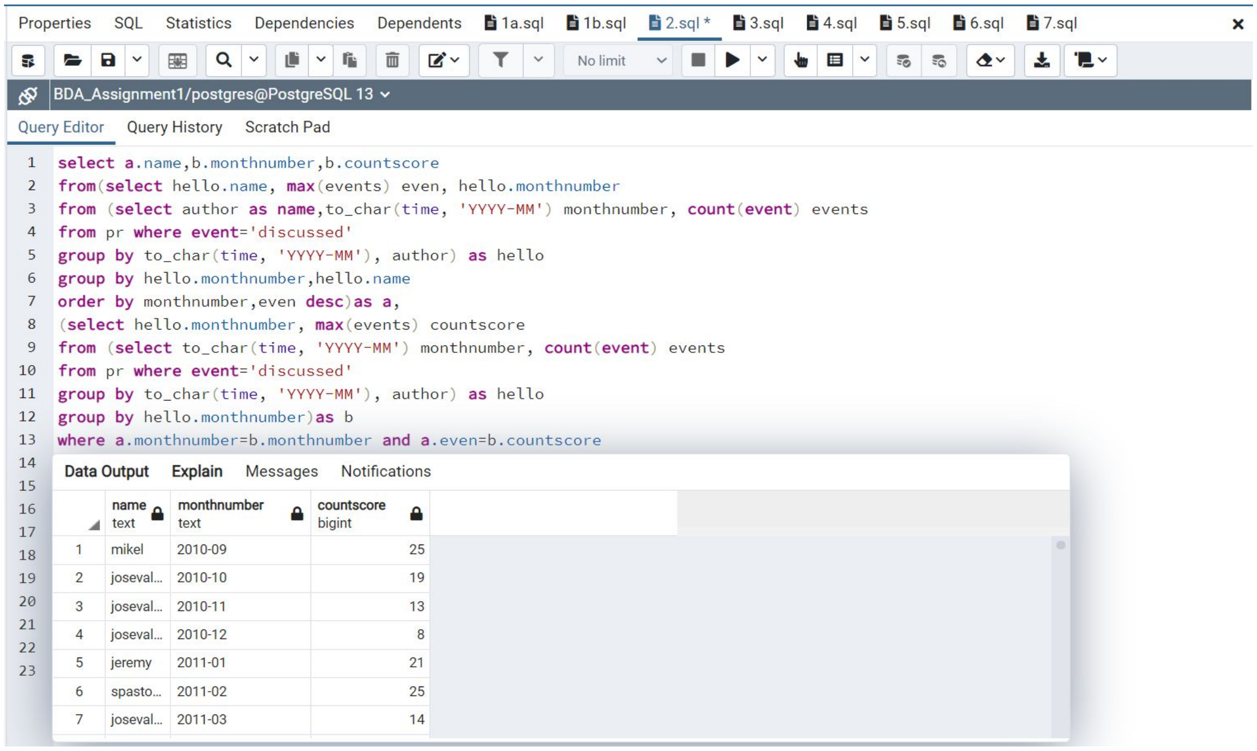
	name text	monthnumber text	countscore bigint
1	rafaelfranca	01	828
2	rafaelfranca	02	555
3	rafaelfranca	03	580
4	rafaelfranca	04	758
5	rafaelfranca	05	915
6	rafaelfranca	06	582
7	rafaelfranca	07	579


Properties SQL Statistics Dependencies Dependents 1a.sql 1b.sql 2.sql 3.sql 4.sql 5.sql 6.sql 7.sql
No limit
BDA_Assignment1/postgres@PostgreSQL 13
Query Editor Query History Scratch Pad
Data Output Explain Messages Notifications

```sql
select a.name,b.monthnumber,b.countscore
from(select hello.name, max(events) even, hello.monthnumber
from (select author as name,to_char(time, 'MM') monthnumber, count(event) events
from pr where event='discussed'
group by to_char(time, 'MM'), author) as hello
```

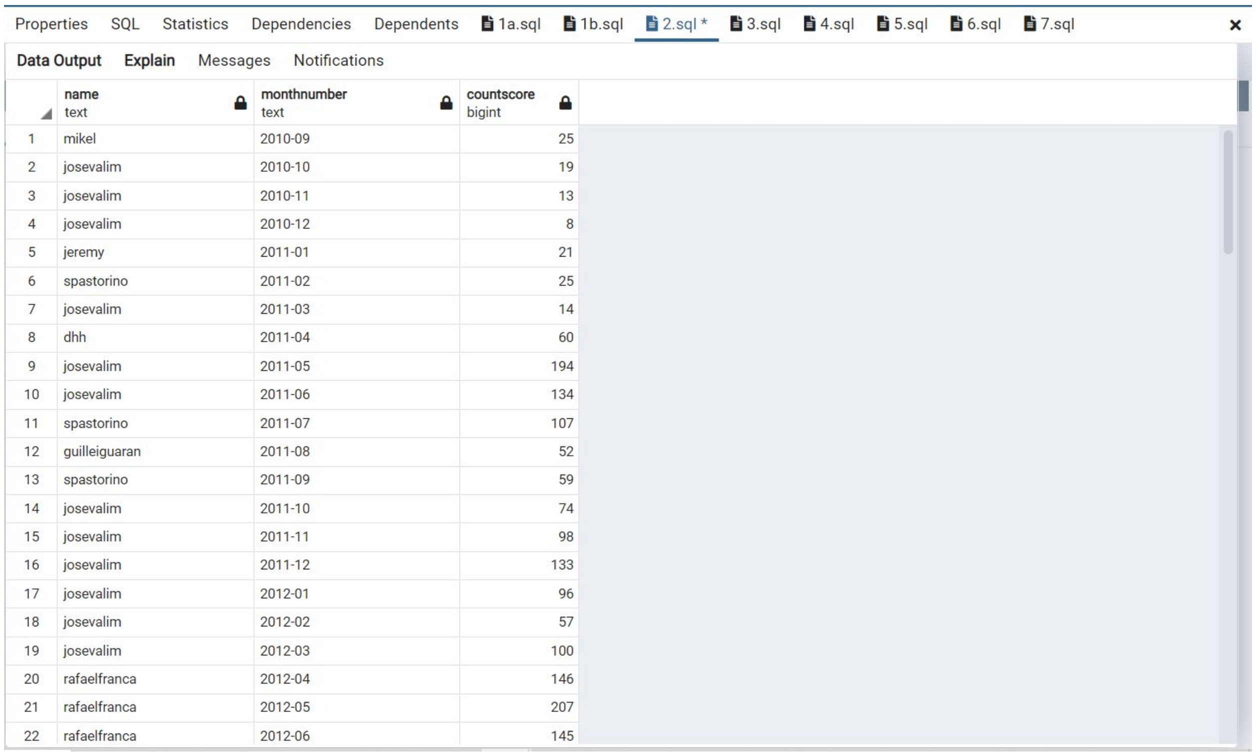
	name text	monthnumber text	countscore bigint
1	rafaelfranca	01	828
2	rafaelfranca	02	555
3	rafaelfranca	03	580
4	rafaelfranca	04	758
5	rafaelfranca	05	915
6	rafaelfranca	06	582
7	rafaelfranca	07	579
8	rafaelfranca	08	651
9	rafaelfranca	09	585
10	rafaelfranca	10	667
11	rafaelfranca	11	590
12	rails-bot	12	546

Assumption 2: Count of highest number of “discussed” PRs is calculated for the same month but different years separately. For example, counts of March 2010 and March 2011 are counted separately and as different entities.



```sql
select a.name,b.monthnumber,b.countscore
from(select hello.name, max(events) even, hello.monthnumber
from (select author as name,to_char(time, 'YYYY-MM') monthnumber, count(event) events
from pr where event='discussed'
group by to_char(time, 'YYYY-MM'), author) as hello
group by hello.monthnumber,hello.name
order by monthnumber,even desc)as a,
(select hello.monthnumber, max(events) countscore
from (select to_char(time, 'YYYY-MM') monthnumber, count(event) events
from pr where event='discussed'
group by to_char(time, 'YYYY-MM'), author) as hello
group by hello.monthnumber)as b
where a.monthnumber=b.monthnumber and a.even=b.countscore
```

	name text	monthnumber text	countscore bigint
1	mikel	2010-09	25
2	joseval...	2010-10	19
3	joseval...	2010-11	13
4	joseval...	2010-12	8
5	jeremy	2011-01	21
6	spasto...	2011-02	25
7	joseval...	2011-03	14



	name text	monthnumber text	countscore bigint
1	mikel	2010-09	25
2	josevalim	2010-10	19
3	josevalim	2010-11	13
4	josevalim	2010-12	8
5	jeremy	2011-01	21
6	spastorino	2011-02	25
7	josevalim	2011-03	14
8	dhh	2011-04	60
9	josevalim	2011-05	194
10	josevalim	2011-06	134
11	spastorino	2011-07	107
12	guilleiguaran	2011-08	52
13	spastorino	2011-09	59
14	josevalim	2011-10	74
15	josevalim	2011-11	98
16	josevalim	2011-12	133
17	josevalim	2012-01	96
18	josevalim	2012-02	57
19	josevalim	2012-03	100
20	rafaelfranca	2012-04	146
21	rafaelfranca	2012-05	207
22	rafaelfranca	2012-06	145

Q 3) Output the person who has the highest number of comments per week. Assume a week has 7 days. The comments refer to events of the type “discussed”.

Ans 3)
Query :

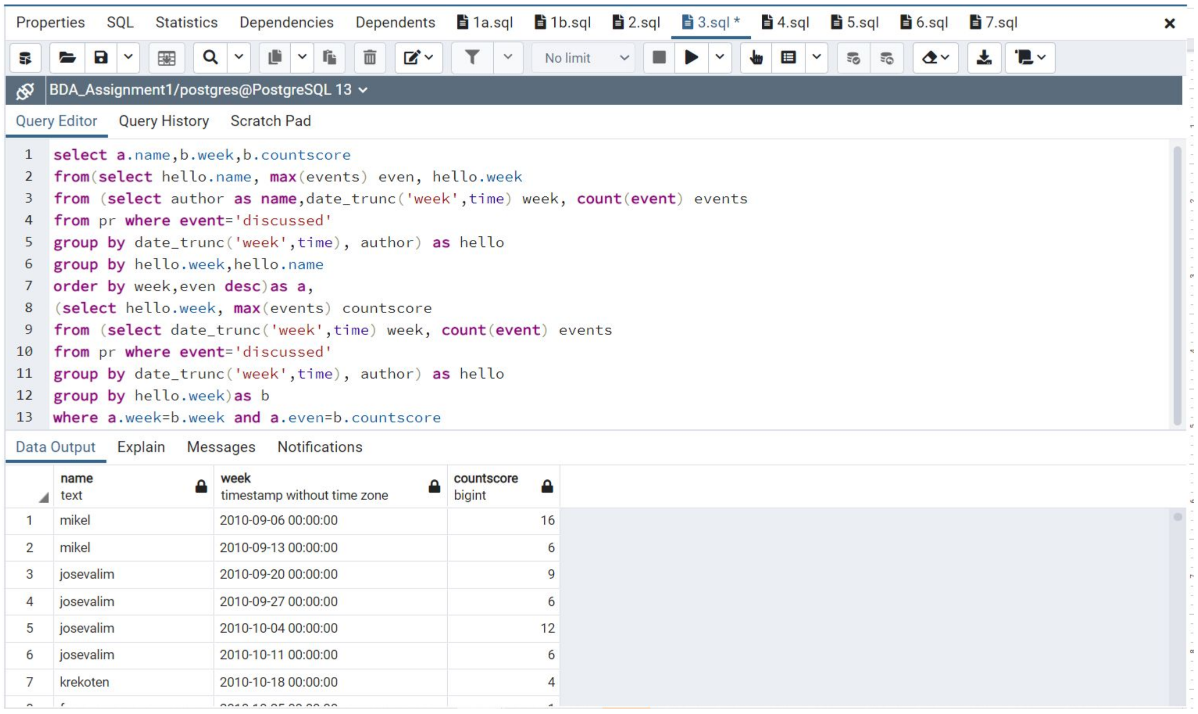
```
select a.name,b.week,b.countscore
from(select hello.name, max(events) even, hello.week
from (select author as name,date_trunc('week',time) week, count(event) events
from pr where event='discussed'
group by date_trunc('week',time), author) as hello
group by hello.week,hello.name
order by week,even desc)as a,
(select hello.week, max(events) countscore
from (select date_trunc('week',time) week, count(event) events
from pr where event='discussed'
group by date_trunc('week',time), author) as hello
group by hello.week)as b
where a.week=b.week and a.even=b.countscore
```

Methodology: All information is collected from by creating two tables as follows:
One table consists of the maximum count of discussed PRs for each week which is calculated from another SELECT subquery which calculates the count of discussed PRs for every week grouped by the weeks and author name. This table is assigned alias “a”.

The second table contains the list of maximum count of discussed PRs for every author grouped by week which is calculated from a SELECT subquery which stores the count of these PRs grouped by the weeks and then author name. This table is assigned alias “b”.

Our final query uses the above two queries as sub-queries to select the authors which have the maximum count of PRs for every week by matching the week and countscore columns of both tables.

Assumption: The weeks are assumed to start on a Monday. As observed, the first row of our data output mentions 2010-09-06 (i.e. Monday) as the start of the week.
The week column only contains the start date of each week.
(A week is assumed to be 7 days long.)



```
select a.name,b.week,b.countscore
from(select hello.name, max(events) even, hello.week
from (select author as name,date_trunc('week',time) week, count(event) events
from pr where event='discussed'
group by date_trunc('week',time), author) as hello
group by hello.week,hello.name
order by week,even desc)as a,
(select hello.week, max(events) countscore
from (select date_trunc('week',time) week, count(event) events
from pr where event='discussed'
group by date_trunc('week',time), author) as hello
group by hello.week)as b
where a.week=b.week and a.even=b.countscore
```

	name text	week timestamp without time zone	countscore bigint
1	mikel	2010-09-06 00:00:00	16
2	mikel	2010-09-13 00:00:00	6
3	josevalim	2010-09-20 00:00:00	9
4	josevalim	2010-09-27 00:00:00	6
5	josevalim	2010-10-04 00:00:00	12
6	josevalim	2010-10-11 00:00:00	6
7	krekoten	2010-10-18 00:00:00	4

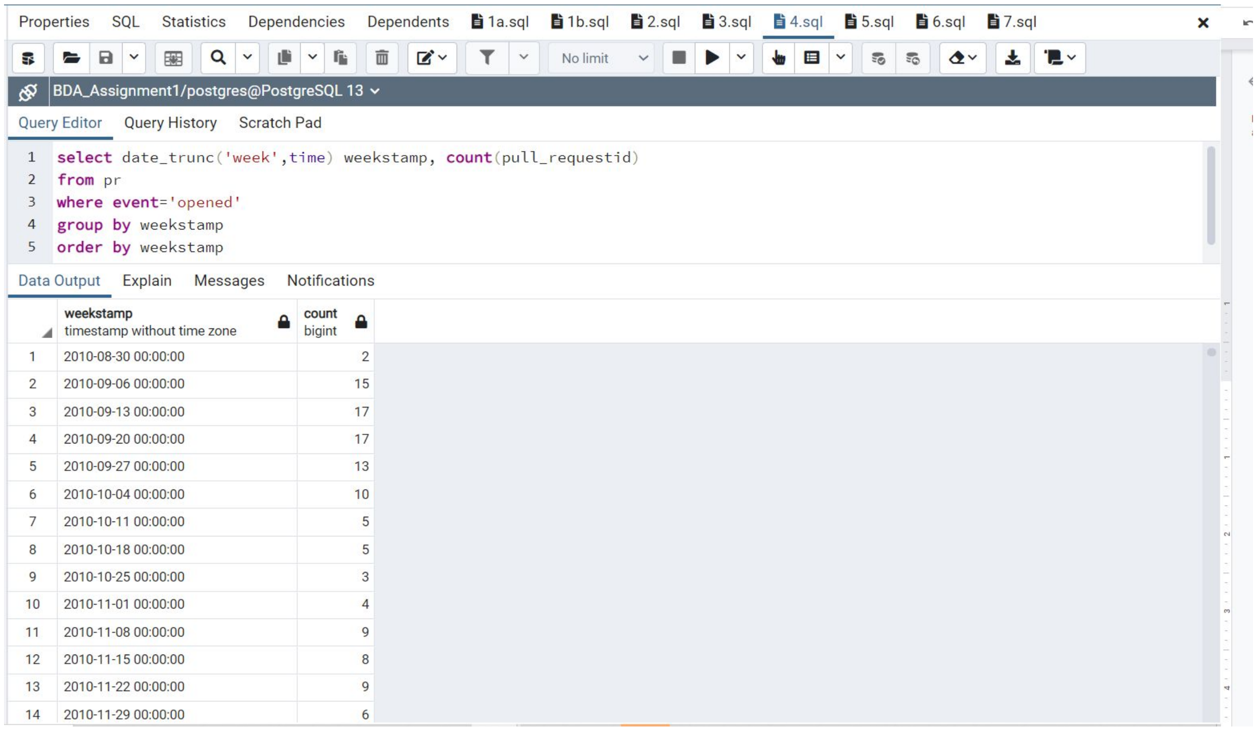
Q 4) Output the number of Pull Requests (PRs) opened per week.

Ans 4)
Query :

```
select date_trunc('week',time) weekstamp, count(pull_requestid)
from pr
where event='opened'
group by event,weekstamp
order by weekstamp
```

Methodology: We selected the count of data points with the event as “opened” and grouped by the week stamp. Weekstamp refers to the start date of each week. The data output is ordered by increasing the start date of weeks.

Assumptions: The weeks are assumed to start on a Monday. As observed, the first row of our data output mentions 2010-09-06 (i.e. Monday) as the start of the week.
The week column only contains the start date of each week.
(A week is assumed to be 7 days long.)



```sql
select date_trunc('week',time) weekstamp, count(pull_requestid)
from pr
where event='opened'
group by weekstamp
order by weekstamp
```

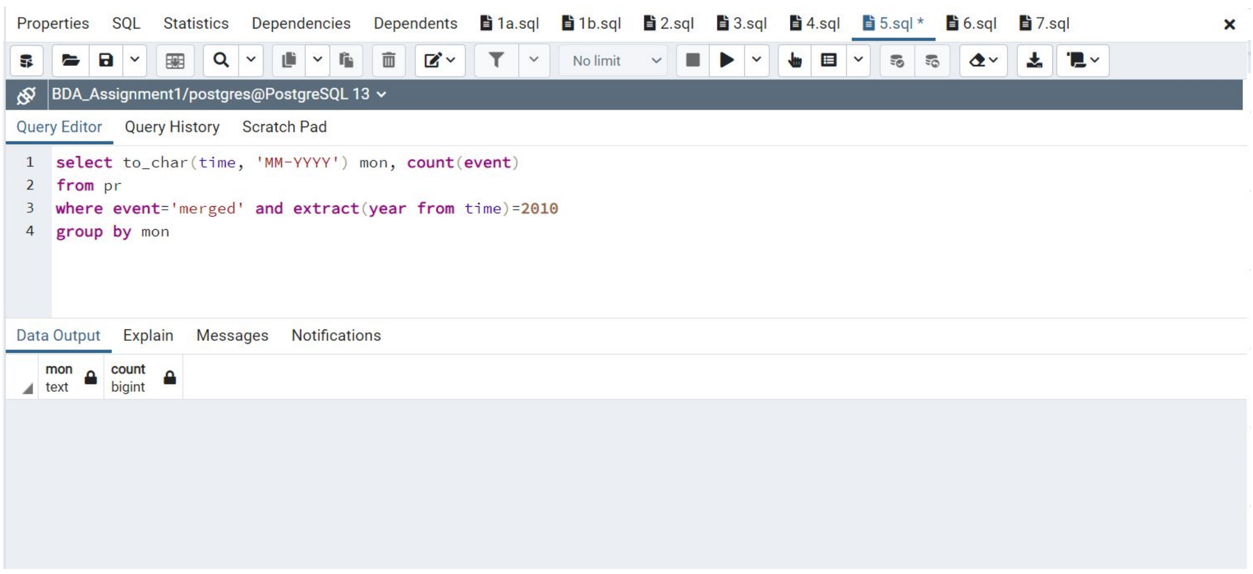
	weekstamp timestamp without time zone	count bigint
1	2010-08-30 00:00:00	2
2	2010-09-06 00:00:00	15
3	2010-09-13 00:00:00	17
4	2010-09-20 00:00:00	17
5	2010-09-27 00:00:00	13
6	2010-10-04 00:00:00	10
7	2010-10-11 00:00:00	5
8	2010-10-18 00:00:00	5
9	2010-10-25 00:00:00	3
10	2010-11-01 00:00:00	4
11	2010-11-08 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

Q 5) Count the number of Pull Requests merged per month in the year 2010.

Ans 5)
Query :

```
select to_char(time, 'MM-YYYY') mon, count(event)
from pr
where event='merged' and extract(year from time)=2010
group by mon
```

Methodology: We selected the count of PRs with the event of type “merged” and the year in timestamp as 2010 with the count of PRs grouped by each month.



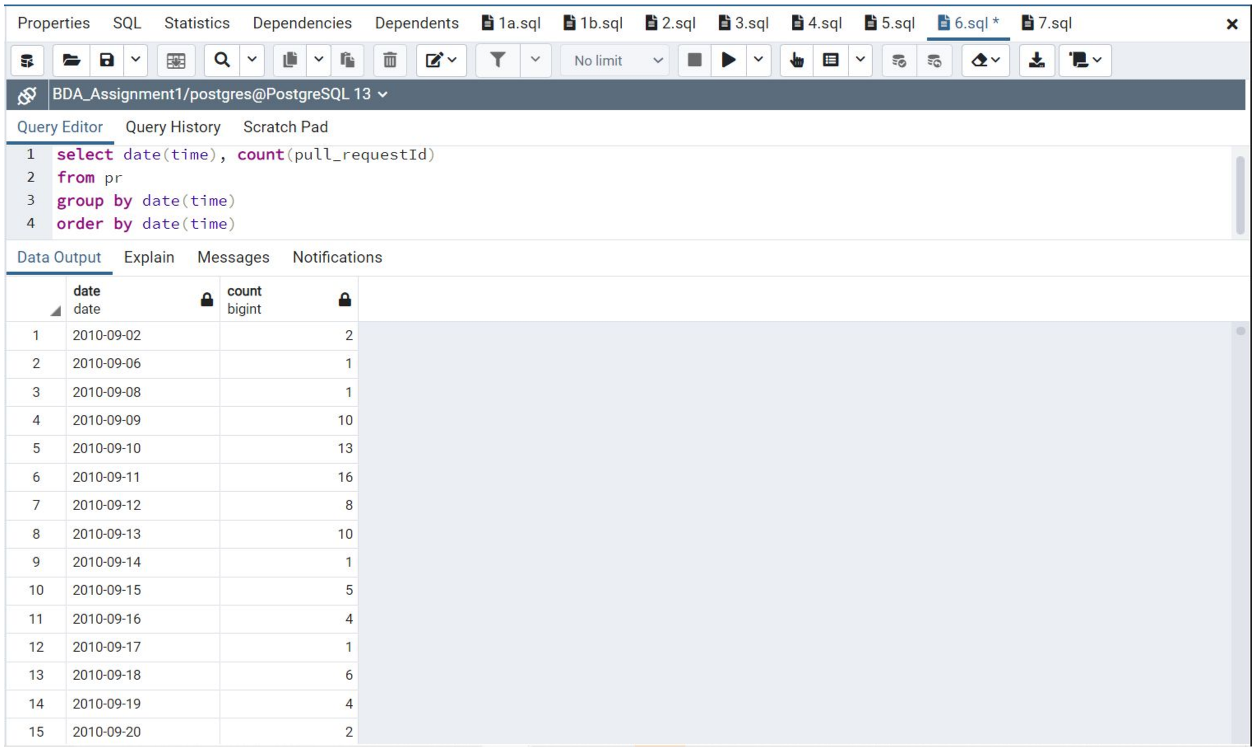
```
select to_char(time, 'MM-YYYY') mon, count(event)
from pr
where event='merged' and extract(year from time)=2010
group by mon
```

mon text	count bigint

Q 6) Per day report the total number of events.

Ans 6)
Query :
select date(time), count(pull_requestId)
from pr
group by date(time)
order by date(time)

Methodology: We selected the count of PRs for every date by grouping the PRs using the date component of timestamp. The data output was also ordered by the oldest date first.



Properties SQL Statistics Dependencies Dependents 1a.sql 1b.sql 2.sql 3.sql 4.sql 5.sql 6.sql * 7.sql
No limit
BDA_Assignment1/postgres@PostgreSQL 13
Query Editor Query History Scratch Pad
Data Output Explain Messages Notifications

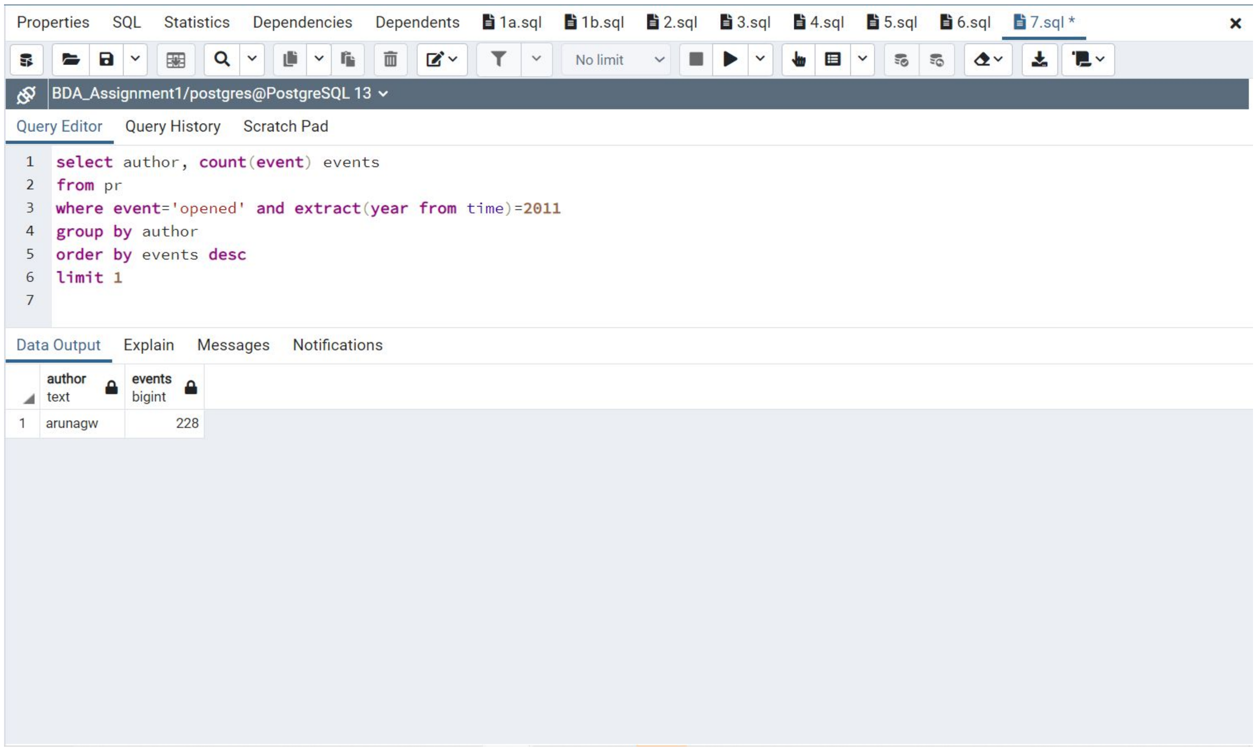
```
select date(time), count(pull_requestId)
from pr
group by date(time)
order by date(time)
```

	date date	count bigint
1	2010-09-02	2
2	2010-09-06	1
3	2010-09-08	1
4	2010-09-09	10
5	2010-09-10	13
6	2010-09-11	16
7	2010-09-12	8
8	2010-09-13	10
9	2010-09-14	1
10	2010-09-15	5
11	2010-09-16	4
12	2010-09-17	1
13	2010-09-18	6
14	2010-09-19	4
15	2010-09-20	2

Q7)Report the user who has opened the highest number of PRs in the year 2011.

Ans 7) :
Query :
select author, count(event) events
from pr
where event='opened' and extract(year from time)=2011
group by author
order by events desc
limit 1

Methodology: We selected the author name and the count of PRs grouped by the name of authors where the event type of PRs was "opened" and the year in timestamp as 2011.
The list of authors were ordered by their count of PRs and the table was limited to top 1 entry so as to get the name of author with maximum number of opened PRs in 2011.



Properties SQL Statistics Dependencies Dependents 1a.sql 1b.sql 2.sql 3.sql 4.sql 5.sql 6.sql 7.sql *
No limit
BDA_Assignment1/postgres@PostgreSQL 13
Query Editor Query History Scratch Pad
Data Output Explain Messages Notifications

```
select author, count(event) events
from pr
where event='opened' and extract(year from time)=2011
group by author
order by events desc
limit 1
```

```
   author   events
   text     bigint
1  arunagw     228
```

Learnings from the assignment :

1. Using PostgreSQL and understanding its pgadmin client side and terminal interfaces.
2. How to structure SQL queries
3. Aggregate SQL functions - count, max
4. Different methods of deriving various information from the SQL data-type timestamp - extract(), year(), date(), date_trunc(), to_char() .
5. Nested subqueries
6. WHERE, GROUP BY, ORDER BY, HAVING clauses
7. Understanding the scope of database available
8. Difference in commands for PostgreSQL(psql) and MySql(mysql)
