

fetch data with count and group by in postgres

Asked 4 years, 7 months ago Modified 4 years, 7 months ago Viewed 57 times

I have a table `mail_details`

mail_id (bigint)	sent_time (timestamp)	failed_time (timestamp)	mail_type (varvharng)
1	2020-02-05	null(default)	type-t
2	2020-02-05	null(default)	type-t
3	2020-02-05	null(default)	type-m
4	2020-02-05	null(default)	type-p
5	null(default)	2020-02-05	type-p
6	2020-02-05	null(default)	type-m
8	2020-02-05	null(default)	type-m
9	null(default)	2020-02-05	type-m
10	2020-02-05	null(default)	type-n
11	2020-02-05	null(default)	type-n

Whenever the mail sent to the user I updated `sent_time` or mail sent failed I update `failed_time`

Now I want to fetch total number of mail sent (count) and the total number of mail failed (count) with the respective date where mail_type=(type_t or type_p or type_m or type_n)

output will be like

```
+-----+-----+-----+
|date          |tatal_sent_mail |tatal_failed_mail|
+-----+-----+-----+
|2020-05-05    |8               |2               |
+-----+-----+-----+
```

I have tried with count but did not work. Any help will be appriciated.

sql postgresql

Share Improve this question Follow

asked May 7, 2020 at 18:47

 **bala**
125 ● 8

what did you try? – [zealous](#) May 7, 2020 at 18:52

Hi @zealous I am beginner and tried with count and group by but it did not work – [bala](#)
May 7, 2020 at 18:55

@bala - Could you try all answers and comment what works and what doesn't? You should try for a data for which an entry for a date exists sent_time and doesn't for failed_time and vice versa to have a 100% coverage on your test. – [VN'sCorner](#) May 7, 2020 at 19:19 ✎

4 Answers

Sorted by: Highest score (default) ▾



1



You can summarise using aggregate filters, and coalesce the dates between columns.

```
SELECT
  coalesce(sent_time, failed_time) AS date,
  count(*) FILTER (WHERE sent_time IS NOT NULL) AS total_sent_mail,
  count(*) FILTER (WHERE failed_time IS NOT NULL) AS total_failed_mail
FROM mail_details
WHERE mail_type IN ('type_t', 'type_p', 'type_m', 'type_n')
GROUP BY 1
ORDER BY 1;
```



If you need rows for missing dates, you would need to resort to using `generate_series()`.

Share Improve this answer Follow

answered May 7, 2020 at 19:03



[Thom Brown](#)

2,019 ● 6 ● 13

why it gives an extra row with blank date and 0 counts @Thom Brown – [bala](#) May 7, 2020 at 19:45

Do you have a row where both sent_mail and failed_mail are NULL? If you do, you can just filter that out in the WHERE clause. – [Thom Brown](#) May 7, 2020 at 19:48

Ok got it. One more thing is that sent_time is a timestamp so for sent time 2020-05-08 01:10:53.847912 and 2020-05-08 01:30:53.847912 it returns two rows with count 1 and 1 but I want it to consider only date not time(h:s:i) and returns one row with count 2. @Thom Brown – [bala](#) May 7, 2020 at 19:57 ✎

You can just cast them to dates, so use sent_time::date, failed_time::date in the coalesce parameters. No need to do that in the FILTER clause though. – [Thom Brown](#) May 7, 2020 at 20:01



1

I would unpivot using a lateral `join` and aggregate:

```
select time::date, sum(is_sent), sum(is_fail)
from mail_details md cross join lateral
  (values (sent_time, (sent_time is not null)::int, 0))
```



```
(failed_time, 0, (failed is not null)::int)
) v(time, is_sent, is_fail)
where t.time is not null and
      md.mail_type in ('type_t', 'type_p', 'type_m', 'type_n')
group by time::date
```

Share

edited May 8, 2020 at 0:14

answered May 7, 2020 at 19:05

Improve this answer



Gordon Linoff

1.3m ● 60 ● 686 ● 832

Follow

I did not got this query i think you have missed where mail_type – bala May 7, 2020 at 19:18

I did not got this query i think you have missed where mail_type @Gordon Linoff – bala May 7, 2020 at 19:46

@bala . . . It was not clear to me if that really required filtering. You listed all the mail types in your sample data, so no filtering seemed necessary. – Gordon Linoff May 8, 2020 at 0:15



Another one using FULL OUTER JOIN on the inline view,

0



```
select sent_time as date, tatal_sent_mail, tatal_failed_mail
from
(select sent_time, count(1) tatal_sent_mail
where mail_type in ('type_t', 'type_p', 'type_m', 'type_n')
group by sent_time) sent_data
FULL OUTER JOIN
(select failed_time Final_date, count(1) tatal_failed_mail
where mail_type in ('type_t', 'type_p', 'type_m', 'type_n')
group by failed_time) failed_Data
ON sent_data.sent_time = failed_Data.failed_time
```

Share Improve this answer Follow

answered May 7, 2020 at 19:15



VN'sCorner

1,552 ● 1 ● 9 ● 13



Using a union, this should do it:

0



```
SELECT dt,
       SUM(CASE WHEN event == 'SENT' then 1 else 0 end) total_sent_mail,
       SUM(CASE WHEN event == 'FAIL' then 1 else 0 end) total_failed_mail,
FROM
(
  SELECT 'SENT' as event,
         sent_time as dt
  FROM   mail_details
  WHERE  sent_time IS NOT NULL
)
UNION ALL
(
  SELECT 'FAIL' as event,
         failed_time as dt
  FROM   mail_details
)
```

```
WHERE failed_time IS NOT NULL )  
GROUP BY dt
```

Share

edited May 7, 2020 at 19:26

answered May 7, 2020 at 18:59

Improve this answer

Follow



timhealz

94 ● 5
