

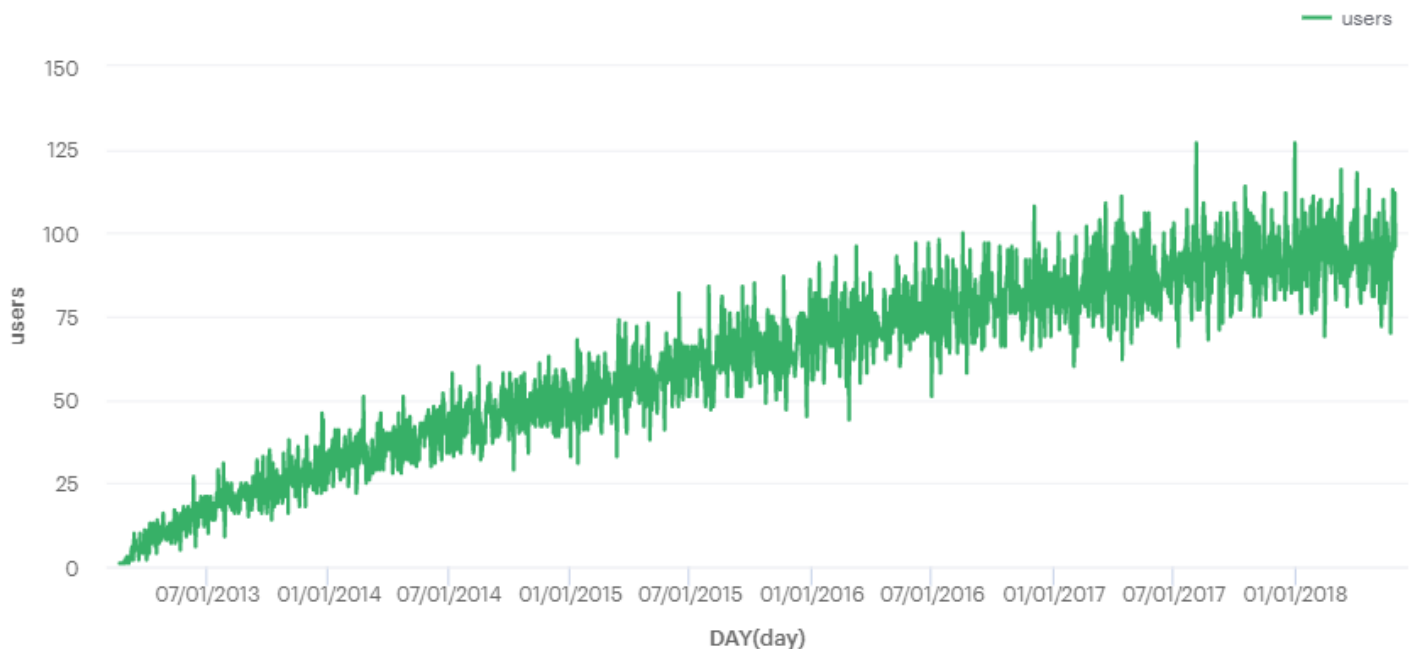
# Counting Users

Exercise 1: We'll be using the users table to answer the question "How many new users are added each day?". Start by making sure you understand the columns in the table.

```
select * from dsv1069.users
```

Exercise 2: Without worrying about deleted user or merged users, count the number of users added each day.

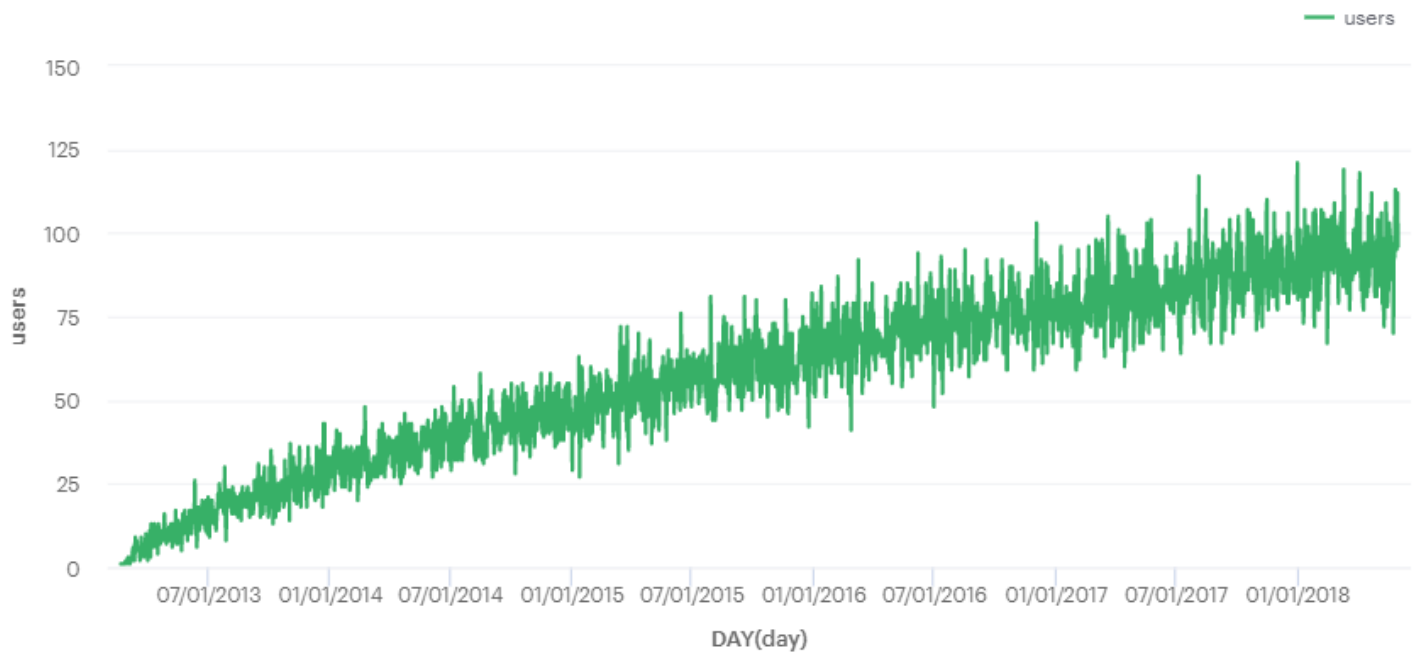
```
1. select
2.   date(created_at) as day,
3.   count(*)         as users
4. from
5.   dsv1069.users
6. group by
7.   date(created_at)
```



Exercise 3: Consider the following query. Is this the right way to count merged or deleted users? If all of our users were deleted tomorrow what would the result look like?

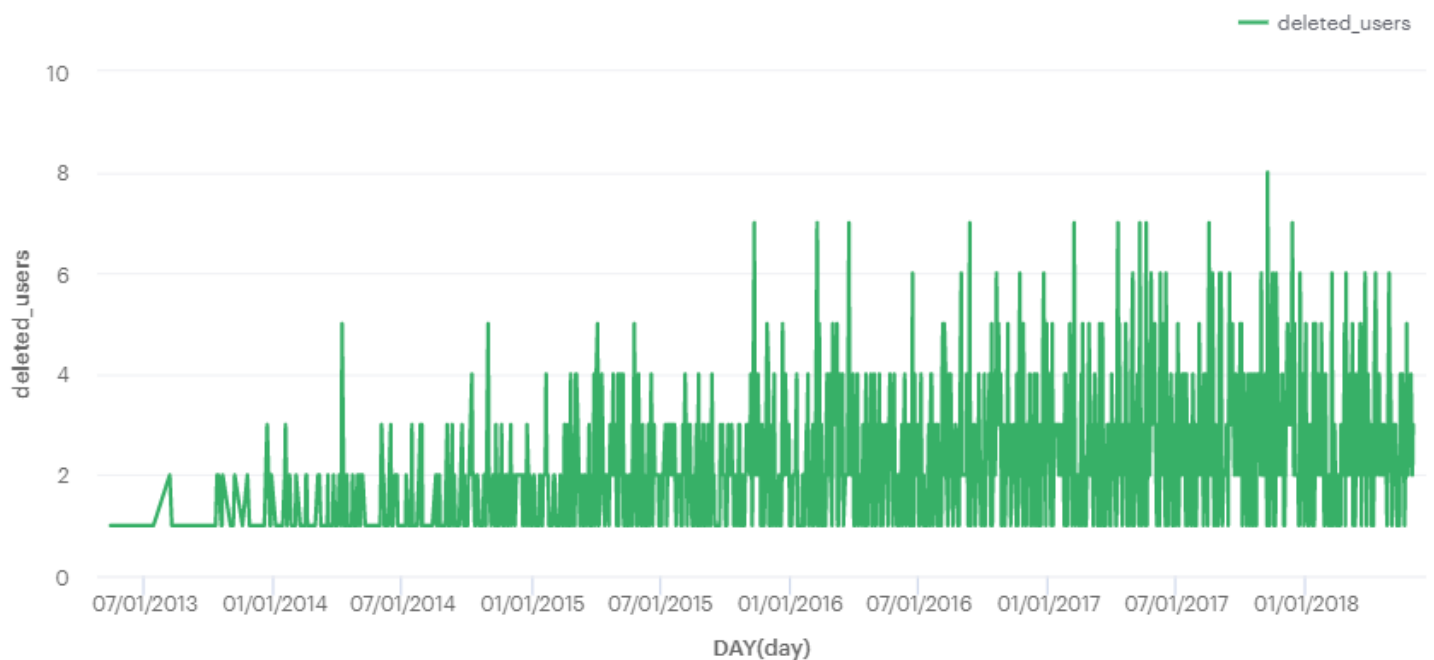
```
1. select
2.   date(created_at) as day,
3.   count(*)         as users
4. from
5.   dsv1069.users
6. where
7.   deleted_at is null
8. and
9.   (id <> parent_user_id or parent_user_id is null)
10. group by
11.   date(created_at)
```

Yes.



Exercise 4: Count the number of users deleted each day. Then count the number of users removed due to merging in a similar way.

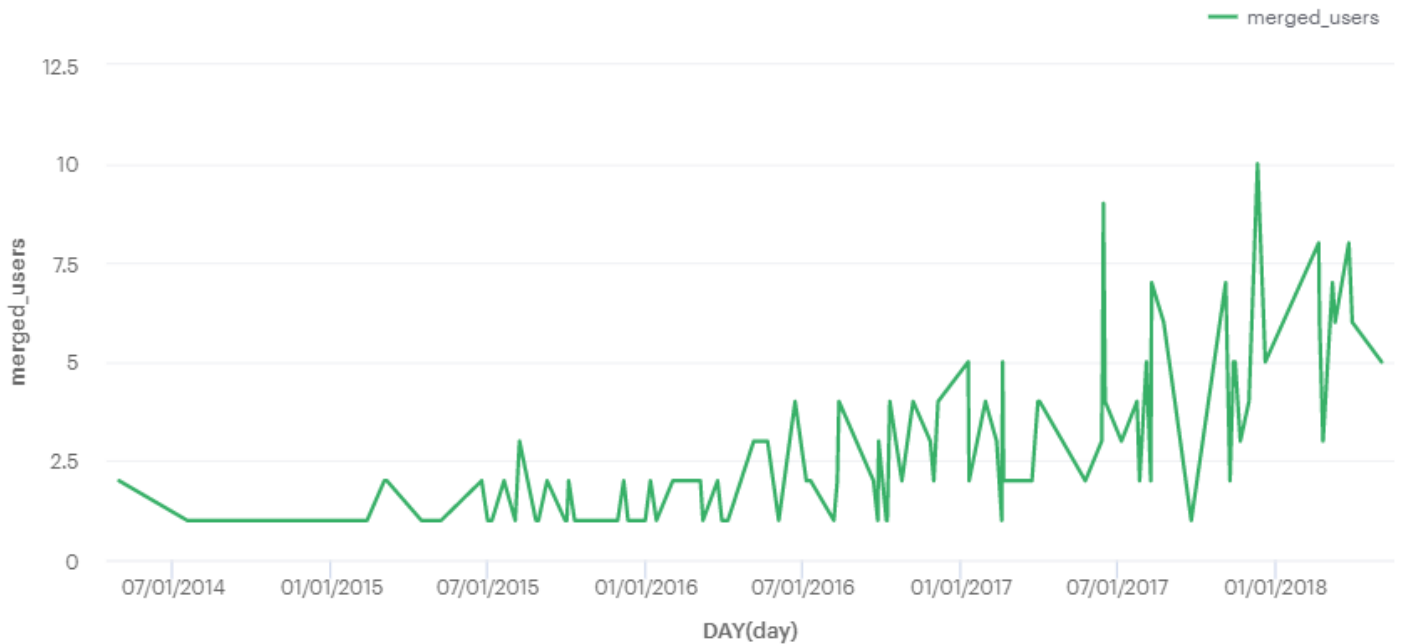
```
1. select
2.   date(deleted_at) as day,
3.   count(*)        as deleted_users
4. from
5.   dsv1069.users
6. where
7.   deleted_at is not null
8. group by
9.   date(deleted_at)
```



```

1. select
2.     date(merged_at) as day,
3.     count(*)         as merged_users
4. from
5.     dsv1069.users
6. where
7.     id <> parent_user_id
8. and
9.     parent_user_id is not null
10. group by
11.     date(merged_at)

```



Exercise 5: Use the pieces you've built as sub-tables and create a table that has a column for the date, the number of users created, the number of users deleted and the number of users merged that day.

```

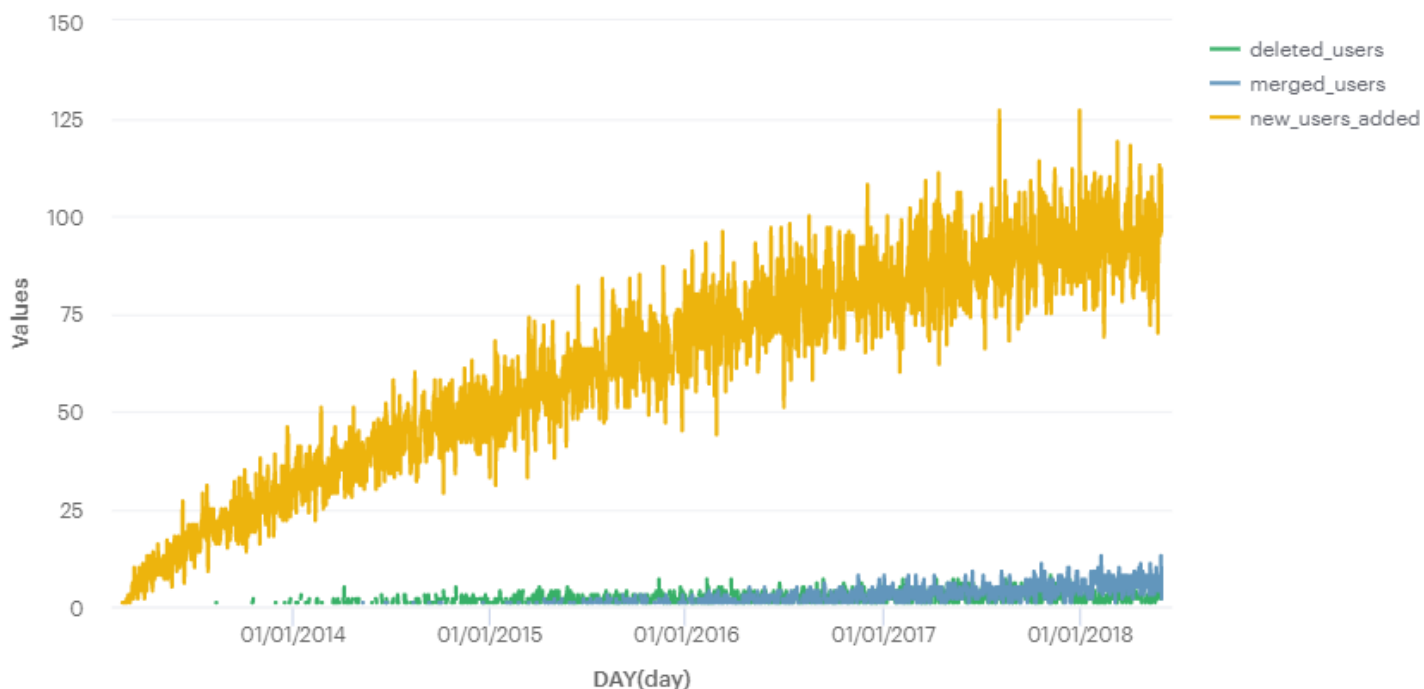
1. SELECT
2.     new.day,
3.     new.new_users_added,
4.     deleted.deleted_users,
5.     merged.merged_users
6. FROM
7.     (SELECT
8.         date(created_at) AS day,
9.         COUNT(*)         AS new_users_added
10.    FROM
11.        dsv1069.users
12.    GROUP BY
13.        date(created_at)
14.    ) new
15. LEFT JOIN
16.     (SELECT
17.         date(deleted_at) AS day,
18.         COUNT(*)         AS deleted_users
19.    FROM
20.        dsv1069.users
21.    WHERE

```

```

22.     deleted_at IS NOT NULL
23. GROUP BY
24.     date(deleted_at)
25. ) deleted
26. ON deleted.day = new.day
27. LEFT JOIN
28. (SELECT
29.     date(merged_at) AS day,
30.     COUNT(*)        AS merged_users
31. FROM
32.     dsv1069.users
33. WHERE
34.     id <> parent_user_id
35. AND
36.     parent_user_id IS NOT NULL
37. GROUP BY
38.     date(merged_at)
39. ) merged
40. ON merged.day = new.day

```



Exercise 6: Refine your query from #5 to have informative column names and so that null columns return 0.

```

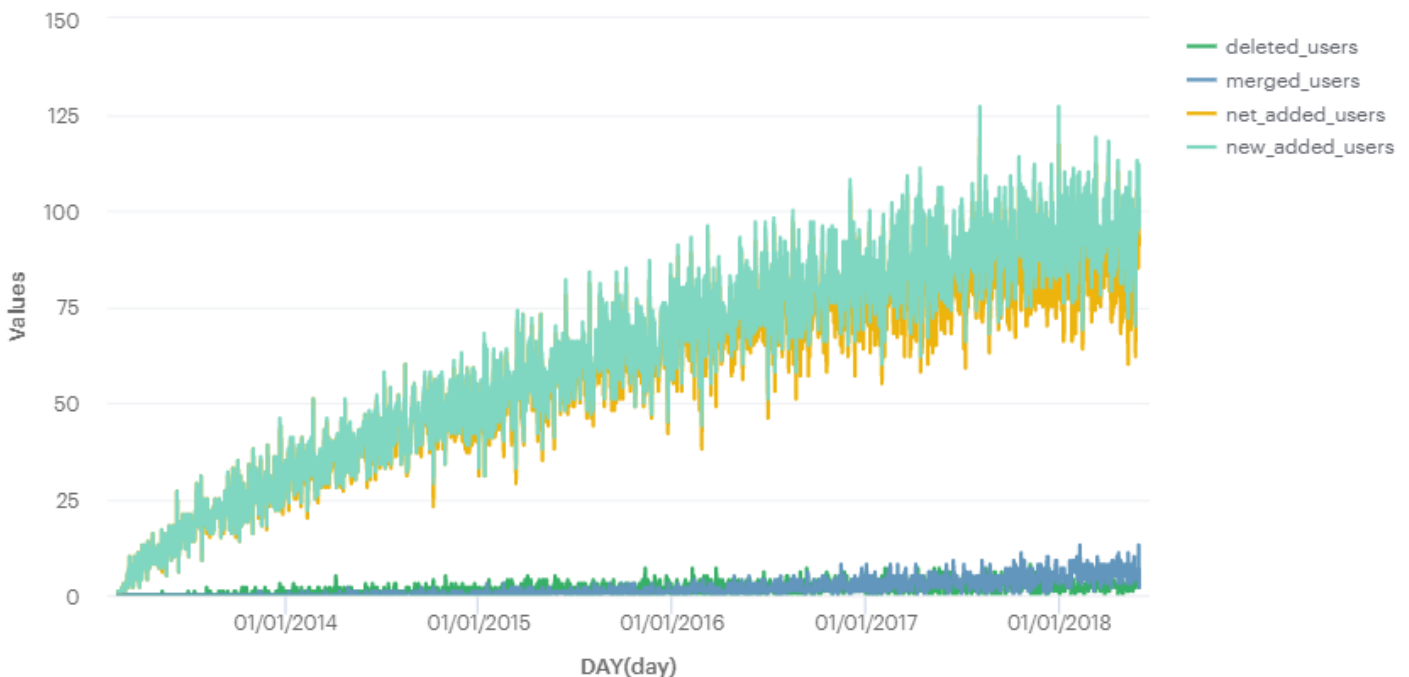
1. SELECT
2.     new.day,
3.     new.new_added_users,
4.     COALESCE(deleted.deleted_users,0) AS deleted_users,
5.     COALESCE(merged.merged_users,0) AS merged_users,
6.     (new.new_added_users - COALESCE(deleted.deleted_users,0) - COALESCE(merged.merged_users,0))
7.     AS net_added_users
8. FROM
9.     (SELECT
10.        date(created_at) AS day,
11.        COUNT(*)        AS new_added_users
12. FROM
13.     dsv1069.users
14. GROUP BY

```

```

15.     date(created_at)
16. ) new
17. LEFT OUTER JOIN
18. (SELECT
19.     date(deleted_at) AS day,
20.     COUNT(*)         AS deleted_users
21. FROM
22.     dsv1069.users
23. WHERE
24.     deleted_at IS NOT NULL
25. GROUP BY
26.     date(deleted_at)
27. ) deleted
28. ON deleted.day = new.day
29. LEFT OUTER JOIN
30. (SELECT
31.     date(merged_at) AS day,
32.     COUNT(*)         AS merged_users
33. FROM
34.     dsv1069.users
35. WHERE
36.     merged_at IS NOT NULL
37. AND
38.     id <> parent_user_id
39. GROUP BY
40.     date(merged_at)
41. ) merged
42. ON
43. merged.day = new.day

```



Exercise 7: What if there were days where no users were created, but some users were deleted or merged. Does the previous query still work? No, it doesn't. Use the `dates_rollup` as a backbone for this query, so that we won't miss any dates.

```

1. SELECT
2.     dates_rollup.date,
3.     new.new_added_users,
4.     COALESCE(deleted.deleted_users,0) AS deleted_users,
5.     COALESCE(merged.merged_users,0) AS merged_users,
6.     (new.new_added_users - COALESCE(deleted.deleted_users,0) - COALESCE(merged.merged_users,0))
7.     AS net_added_users
8. FROM
9.     dsv1069.dates_rollup
10. LEFT OUTER JOIN
11.     (SELECT
12.         date(created_at) AS day,
13.         COUNT(*) AS new_added_users
14.     FROM
15.         dsv1069.users
16.     GROUP BY
17.         date(created_at)
18.     ) new
19. ON
20.     new.day = date(dates_rollup.date)
21. LEFT OUTER JOIN
22.     (SELECT
23.         date(deleted_at) AS day,
24.         COUNT(*) AS deleted_users
25.     FROM
26.         dsv1069.users
27.     WHERE
28.         deleted_at IS NOT NULL
29.     GROUP BY
30.         date(deleted_at)
31.     ) deleted
32. ON deleted.day = date(dates_rollup.date)
33. LEFT OUTER JOIN
34.     (SELECT
35.         date(merged_at) AS day,
36.         COUNT(*) AS merged_users
37.     FROM
38.         dsv1069.users
39.     WHERE
40.         merged_at IS NOT NULL
41.     AND
42.         id <> parent_user_id
43.     GROUP BY
44.         date(merged_at)
45.     ) merged
46. ON
47.     merged.day = date(dates_rollup.date)

```

