CTE practice:

Exploring variable distributions with CTEs

Business problem:

A manager on the marketing team comes to you to ask about the performance of their recent email campaign. Now that the campaign has been launched, the marketing manager wants to know how many users have clicked the link in the email.

While this project was being planned, you collaborated with the front-end engineers to create tracking events for the front end of the product located in the `frontendeventlog` table. One of these events, `eventid = 5`, is logged when the user reaches a unique landing page that is only accessed from this campaign email. Since the event tracks when a user views the unique landing page from the email, tracking `eventid = 5` will be the best method to count how many users have clicked the link in the campaign email.

Since an overall aggregate metric like an average can hide outliers and further insights under the hood, you decide it's best to calculate the distribution of **the number of email link clicks per user**. In other words, how many users clicked the email link one time, two times, three times, and so on?

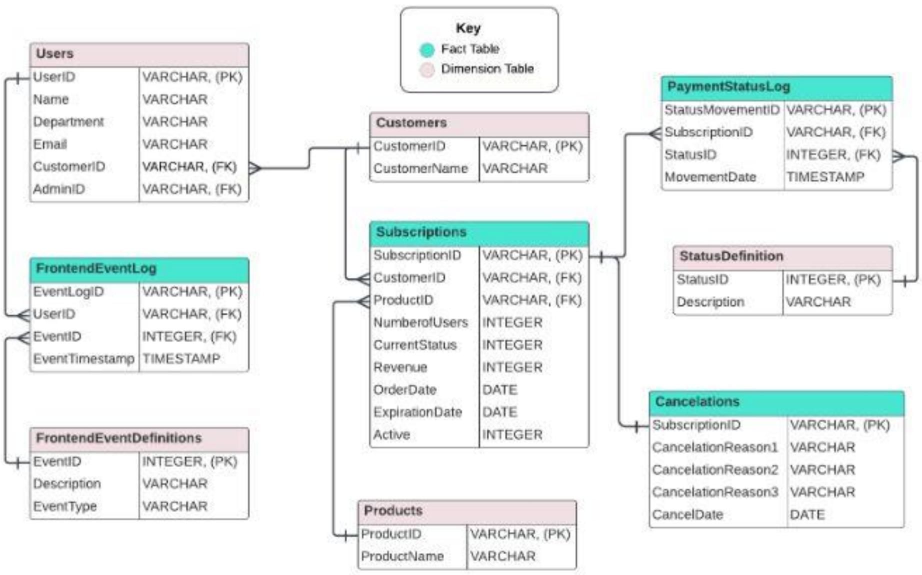
Since the `frontendeventlog` only logs events that *do* happen, we can't count the number of users that received the email but *didn't* click on the link unless we bring in the data from somewhere else. For simplicity, let's ignore this group and only count the events, or email clicks, in the `frontendeventlog`.

Task:

Using a CTE and the `frontendeventlog` table, find the distribution of users across the number of times the email link was clicked per user.

In other words, count the number of users, `num_users`, in each `num_link_clicks` category (one click, two clicks, three clicks,...) using `eventid = 5` to track link clicks.

Main Data Model



Users	
UserID	VARCHAR, (PK)
Name	VARCHAR
Department	VARCHAR
Email	VARCHAR
CustomerID	VARCHAR, (FK)
AdminID	VARCHAR, (FK)

FrontendEventLog	
EventLogID	VARCHAR, (PK)
UserID	VARCHAR, (FK)
EventID	INTEGER, (FK)
EventTimestamp	TIMESTAMP

FrontendEventDefinitions	
EventID	INTEGER, (PK)
Description	VARCHAR
EventType	VARCHAR

Customers	
CustomerID	VARCHAR, (PK)
CustomerName	VARCHAR

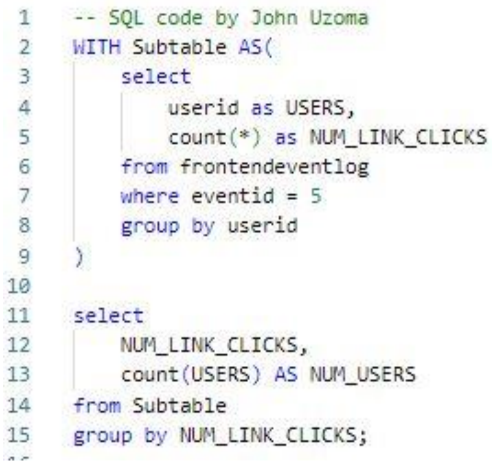
Subscriptions	
SubscriptionID	VARCHAR, (PK)
CustomerID	VARCHAR, (FK)
ProductID	VARCHAR, (FK)
NumberofUsers	INTEGER
CurrentStatus	INTEGER
Revenue	INTEGER
OrderDate	DATE
ExpirationDate	DATE
Active	INTEGER

Products	
ProductID	VARCHAR, (PK)
ProductName	VARCHAR

PaymentStatusLog	
StatusMovementID	VARCHAR, (PK)
SubscriptionID	VARCHAR, (FK)
StatusID	INTEGER, (FK)
MovementDate	TIMESTAMP

StatusDefinition	
StatusID	INTEGER, (PK)
Description	VARCHAR

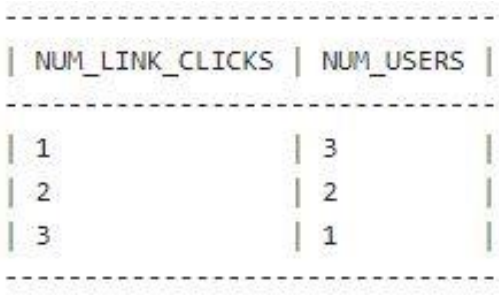
Cancelations	
SubscriptionID	VARCHAR, (PK)
CancelationReason1	VARCHAR
CancelationReason2	VARCHAR
CancelationReason3	VARCHAR
CancelDate	DATE



```
-- SQL code by John Uzoma
WITH Subtable AS(
    select
        userid as USERS,
        count(*) as NUM_LINK_CLICKS
    from frontendeventlog
    where eventid = 5
    group by userid
)

select
    NUM_LINK_CLICKS,
    count(USERS) AS NUM_USERS
from Subtable
group by NUM_LINK_CLICKS;
```

Result



```
-----------------------------------
| NUM_LINK_CLICKS | NUM_USERS |
-----------------------------------
| 1               | 3         |
| 2               | 2         |
| 3               | 1         |
-----------------------------------
```