*Note: Please refer to Table\_data excel for data populated in 3 tables .*

Depending on the amount of data and availability of HTTP request, we can decide to have one job or to create separate job for each HTTP request. For now I created one code to process both the requests.

To schedule the python script to run every day 8:30AM we can run the below cron command.

**crontab -e**

**30 8 \* \* \* /home/melissa/python\_job.py**

**Problem: How many total messages are being sent every day?**

**SQL:** select DATE("createdAt") as "createdAt", count(1) as "total messages" from "PUBLIC".messages group by DATE("createdAt")

**SQL Output:**

|  |  |
| --- | --- |
| createdAt | total messages |
| 25-11-2021 | 3 |
| 26-11-2021 | 2 |
| 27-11-2021 | 5 |
| 28-11-2021 | 1 |
| 29-11-2021 | 1 |
| 30-11-2021 | 1 |
| 02-12-2021 | 1 |
| 03-12-2021 | 1 |
| 05-12-2021 | 1 |
| 06-12-2021 | 1 |
| 04-12-2021 | 3 |

**Problem: Are there any users that did not receive any message?**

**Assumption:** Irrespective of the subscription status, users who did not receive any message

**SQL:** select "ID" from "PUBLIC".user minus select "receiverId" from "PUBLIC".messages group by "receiverId"

**SQL Output:**

|  |
| --- |
| ID |
| 4 |

**Problem:How many active subscriptions do we have today?**

**SQL:** select count(1) as "active subscriptions" from "PUBLIC".subscription where "STATUS" = 'Active'

**SQL Output:**

|  |
| --- |
| active subscriptions |
| 4 |

**Problem:Are there users sending messages without an active subscription? (some extra**

**context for you: in our apps only premium users can send messages).**

**Assumption:**

For "STATUS" = ‘Active’: user can send messages between startDate and EndDate

For "STATUS" = ‘Inactive’: User was active for date between startDate and EndDate and so was able to send message.

For "STATUS" = ‘Rejected’: Ignore the subscription info for such records.

**SQL:**

select distinct A."senderId"

from "PUBLIC".messages A

left join "PUBLIC".subscription B on A."senderId" = B."ID" and A."createdAt" >= B."startDate" and A."createdAt" <= B."endDate" and "STATUS" <> 'Rejected'

where B."ID" is null

**SQL Output:**

|  |
| --- |
| senderId |
| 6 |

**Problem: Did you identified any inaccurate/noisy record that somehow could prejudice**

**the data analyses? How to monitor it (SQL query)? Please explain how do you**

**suggest to handle with this noisy data?**

**Noisy record 1:**

**Problem:** For any given day we should not have any “ID” with multiple “STATUS”

**SQL to Identify Noisy record:**

select A."senderId", count(distinct "STATUS" ) from "PUBLIC".messages A

left outer join "PUBLIC".subscription B on A."senderId" = B."ID"

and A."createdAt" >= B."startDate" and A."createdAt" <= B."endDate"

group by A."senderId" having count(distinct "STATUS" )>1

**Sample data from source (for 2021-11-24 we have two STATUS for ID=3):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **createdAt** | **startDate** | **endDate** | **STATUS** | **AMOUNT** | **ID** |
| 2021-11-22T23:41:32.927Z | 2021-11-23T14:42:04.416Z | 2022-07-26T17:06:45.413Z | Rejected | 64.75 | 3 |
| 2021-11-23T18:57:20.540Z | 2021-11-24T18:04:41.908Z | 2022-03-03T09:47:26.916Z | Active | 88.6 | 3 |

**Solution:**

For the record with status as “Rejected”, the “endDate” should be marked as date present in the “startDate” column for the next record (in this case the record with status as “Active”)

select "createdAt" ,"startDate","endDate", lead ("startDate",1,"endDate") over (partition by "ID" order by "startDate") as "new\_endDate",

"STATUS","AMOUNT","ID" from "PUBLIC".subscription

And use "new\_endDate" for all calculations.

**Expected result set:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **createdAt** | **startDate** | **endDate** | new **\_endDate** | **STATUS** | **AMOUNT** | **ID** |
| 2021-11-22T23:41:32.927Z | 2021-11-23T14:42:04.416Z | 2022-07-26T17:06:45.413Z | 2021-11-24T18:04:41.908Z | Rejected | 64.75 | 3 |
| 2021-11-23T18:57:20.540Z | 2021-11-24T18:04:41.908Z | 2022-03-03T09:47:26.916Z | 2022-03-03T09:47:26.916Z | Active | 88.6 | 3 |

**Noisy record 2:**

In user table age should not be less than 18years (or the minimum age limit)

select \* from "PUBLIC".user where DATEDIFF('YEAR',"birthDate" , CURRENT\_DATE() ) < 18

At front end we should put logic where input birthdate < 18 years (example) should not be accepted.