Given the three tables below, construct a SQL query that would return:

* Monthly spend totals by user’s most recent email
* A field showing their count of droplets used that month
* A field that gives a value of “Low Value” if the customer spent <$500 and “High Value” if the customer spent >=$500 in a given month.

**Invoice Items**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| user\_id | invoice\_month | invoice\_item\_type | resource\_id | amount |
| 19505 | 12/1/2017 | DROPLETS\_TYPE | 392058496 | 10.00 |
| 19505 | 12/1/2017 | BACKUPS\_TYPE | null | 2.00 |
| 19505 | 1/1/2018 | DROPLETS\_TYPE | 392058496 | 10.00 |
| 19505 | 1/1/2018 | DROPLETS\_TYPE | 405849382 | 2.17 |
| 19505 | 1/1/2018 | BACKUPS\_TYPE | null | 2.00 |
| 19506 | 12/1/2017 | DROPLETS\_TYPE | 239483829 | 640.00 |
| 19506 | 12/1/2017 | BLOCK\_STORAGE\_TYPE | 582938 | 200.00 |

**Users**

|  |  |  |
| --- | --- | --- |
| user\_id | email | created\_at |
| 19505 | [john@hotmail.com](mailto:john@hotmail.com) | 2016-04-02 18:40:50 |
| 19505 | [john@gmail.com](mailto:john@gmail.com) | 2017-09-15 13:10:22 |

**Example Output**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| email | invoice\_month | count\_droplets | total\_spend | spend\_segment |
| [john@gmail.com](mailto:john@gmail.com) | 12/1/2017 | 1 | 12.00 | Low Value |
| [john@gmail.com](mailto:john@gmail.com) | 1/1/2018 | 2 | 14.17 | Low Value |

With this dataset, I made the following assumptions:

* All droplets are represented in the Invoice Items table by invoice\_item\_type = ‘DROPLETS\_TYPE’
* The resource\_id will be unique to each droplet used by the user.
* The invoice\_month column has a consistent format of MM/DD/YYYY.
* The Invoice Items table and the Users table have a many-to-many relationship on user\_id.

I created two CTEs. I prefer to use CTEs when possible to avoid using subqueries, as subqueries and complex joins are not usually efficient, and CTEs provide better readability. The first one reduces the users table down to just the most recent email of each user, by using a window function and ignoring records that are not the most recent. The second CTE derives the total amount spent per user per month, and ensures no duplication via the group by clause (as opposed to a window function). I then query from the latter CTE with the addition of a subquery that counts the number of distinct droplets used per each user per month, as specified by invoice\_item\_type. The total spend column is reformatted into a currency. Per the instructions, I labeled a spending segment by using a simple case expression that measures the total spend amount against the provided threshold of $500. Lastly, I join onto the first CTE to grab the most recent email, and then I order the results first by email, then by month.

with most\_recent\_emails as (

select t.user\_id, t.email

from (

select user\_id,

email,

row\_number() over (partition by user\_id order by created\_at desc) as rn

from Users

) t

where t.rn = 1

),

invoice\_amounts as (

Select ii.user\_id,

ii.invoice\_month,

sum(ii.amount) as total\_spend

from Invoice\_Items ii

group by ii.user\_id, ii.invoice\_month

)

select mre.email,

ia.invoice\_month,

(select count(distinct resource\_id)

from Invoice\_Items ii

where ii.user\_id = ia.user\_id and ii.invoice\_month = ia.invoice\_month

and ii.invoice\_item\_type = 'DROPLETS\_TYPE') as count\_droplets,

format(ia.total\_spend, 'C') as total\_spend,

case when ia.total\_spend >= 500 then 'High Value'

else 'Low Value' end as spend\_segment

from invoice\_amounts ia

join most\_recent\_emails mre on mre.user\_id = ia.user\_id

order by mre.email, ia.invoice\_month asc