Project instructions

Athena queries

1. Create a table called **order_products_prior** by using the last SQL query you created from the previous assignment. It should be similar to below (note you need to replace the s3 bucket name "imba" to yours own bucket name):

   ```
   CREATE TABLE order_products_prior WITH (external_location =
   's3://imba/features/order_products_prior/', format = 'parquet')

   as (SELECT a.*,

       b.product_id,

       b.add_to_cart_order,

       b.reordered

   FROM   orders a

       JOIN order_products b

         ON a.order_id = b.order_id

   WHERE  a.eval_set = 'prior' )
   ```

2. Create a table called **user_features_1** as shown below, replace the <sql here> to the desired SQL query. Based on table **order_products_prior**, for each user, calculate the max order_number, the sum of days_since_prior_order and the average of days_since_prior_order. (note you need to replace the s3 bucket name "imba" to yours own bucket name)

   ```
   CREATE TABLE user_features_1 WITH (external_location =
   's3://imba/features/user_features_1/', format = 'parquet') as (

   SELECT user_id,

       Max(order_number)         AS user_orders,

       Sum(days_since_prior_order) AS user_period,

       Avg(days_since_prior_order) AS user_mean_days_since_prior

   FROM   orders

   GROUP  BY user_id)
   ```

3. Create a table called **user_features_2**, similar to above, based on table order_products_prior, for each user calculate the total number of products, total number of distinct products, and user reorder ratio(number of reordered = 1 divided by number of order_number > 1, hint: Cast(Sum(CASE WHEN order_number > 1 THEN 1 ELSE 0 END) AS DOUBLE)
(note you need to replace the s3 bucket name “imba” to yours own bucket name)

```
CREATE TABLE user_features_2 WITH (external_location =
's3://imba/features/user_features_2/', format = 'parquet') as (

SELECT user_id,

    Count(*)                    AS user_total_products,

    Count(DISTINCT product_id)      AS user_distinct_products ,

    Sum(CASE WHEN reordered = 1 THEN 1 ELSE 0 END) / Cast(Sum(CASE WHEN
order_number > 1 THEN 1 ELSE 0 END) AS DOUBLE) AS user_reorder_ratio

FROM   order_products_prior

GROUP  BY user_id )
```

4. Create a table called **up_features**, based on table order_products_prior, for each user and product(hint: group by user_id and product_id), calculate the total number of orders, minimum order_number, maximum order_number and average add_to_cart_order.
(note you need to replace the s3 bucket name “imba” to yours own bucket name)

```
CREATE TABLE up_features WITH (external_location = 's3://imba/features/up_features/', format
= 'parquet') as (

SELECT user_id,

    product_id,

    Count(*)          AS up_orders,

    Min(order_number)     AS up_first_order,

    Max(order_number)     AS up_last_order,

    Avg(add_to_cart_order) AS up_average_cart_position

FROM   order_products_prior

GROUP  BY user_id,

      product_id)
```

5. Create a table called **prd_features**, based on table order_products_prior, first write a sql query to calculate the sequence of product purchase for each user(hint: you should use window

function rank() over (partition by user_id, product_id order by user_id, order_number)) and name it product_seq_time. Then on top of this query, for each product, calculate the count, sum of reordered, sum of product_seq_time = 1 and sum of product_seq_time = 2.
(note you need to replace the s3 bucket name “imba” to yours own bucket name)

```
CREATE TABLE prd_features WITH (external_location = 's3://imba/features/prd_features/',
format = 'parquet')
as (
SELECT product_id,
    Count(*)      AS prod_orders,
    Sum(reordered) AS prod_reorders,
    Sum(CASE WHEN product_seq_time = 1 THEN 1 ELSE 0 END)  AS prod_first_orders,
    Sum(CASE WHEN product_seq_time = 2 THEN 1 ELSE 0 END)  AS prod_second_orders
FROM   (SELECT *,
           Rank()
            OVER (
              partition BY user_id, product_id
              ORDER BY user_id, order_number) AS product_seq_time
      FROM   order_products_prior)
GROUP  BY product_id )
```