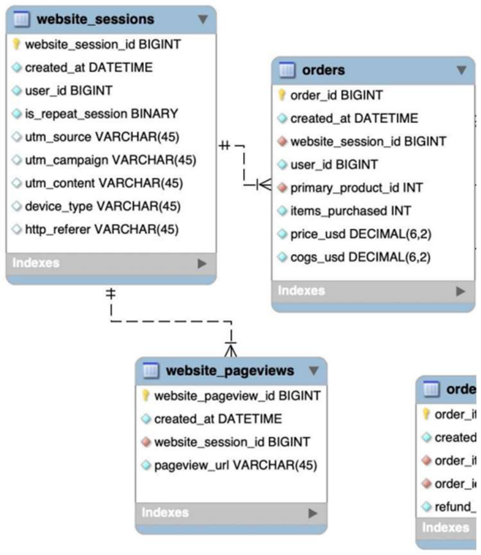
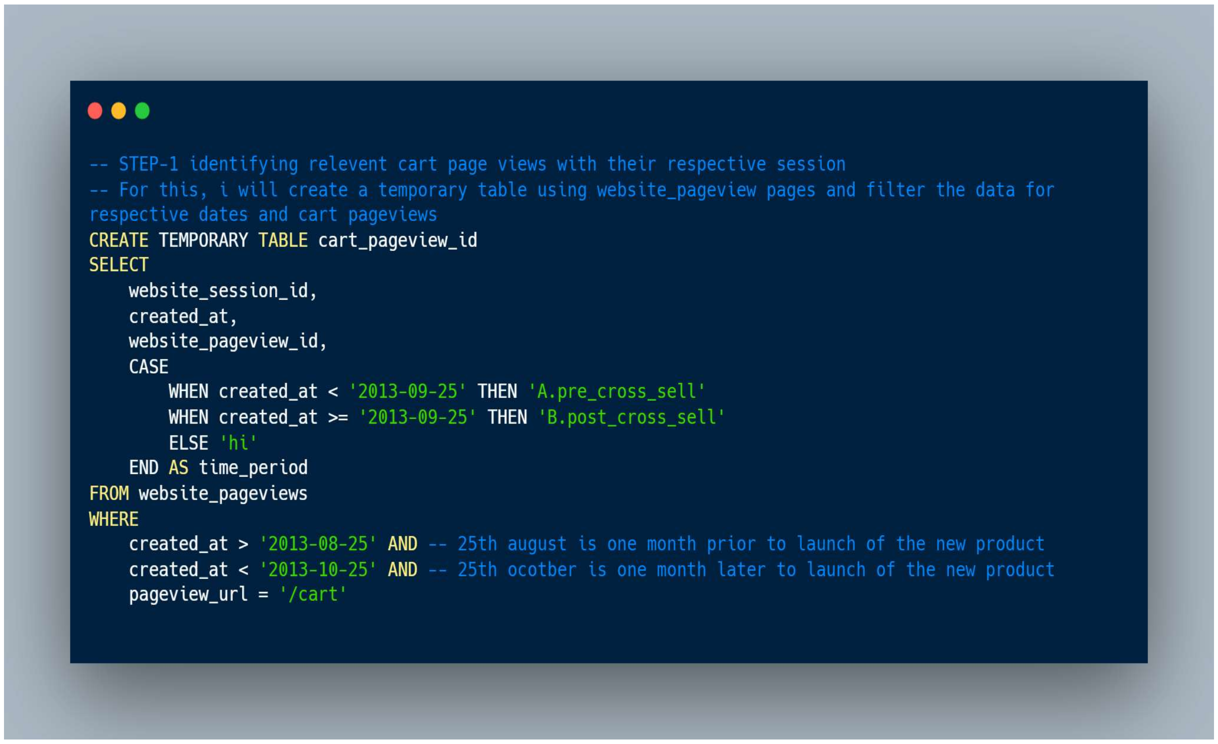
Problem statement: On September 25th company started giving customers the option to add a 2nd product while on the '/cart' page. Write a query to compare the month before vs the month after the change? Result should include CTR (Conversion rate) from the /cart page, Avg Products per Order, AOV (Average order value), and overall revenue per /cart page view.

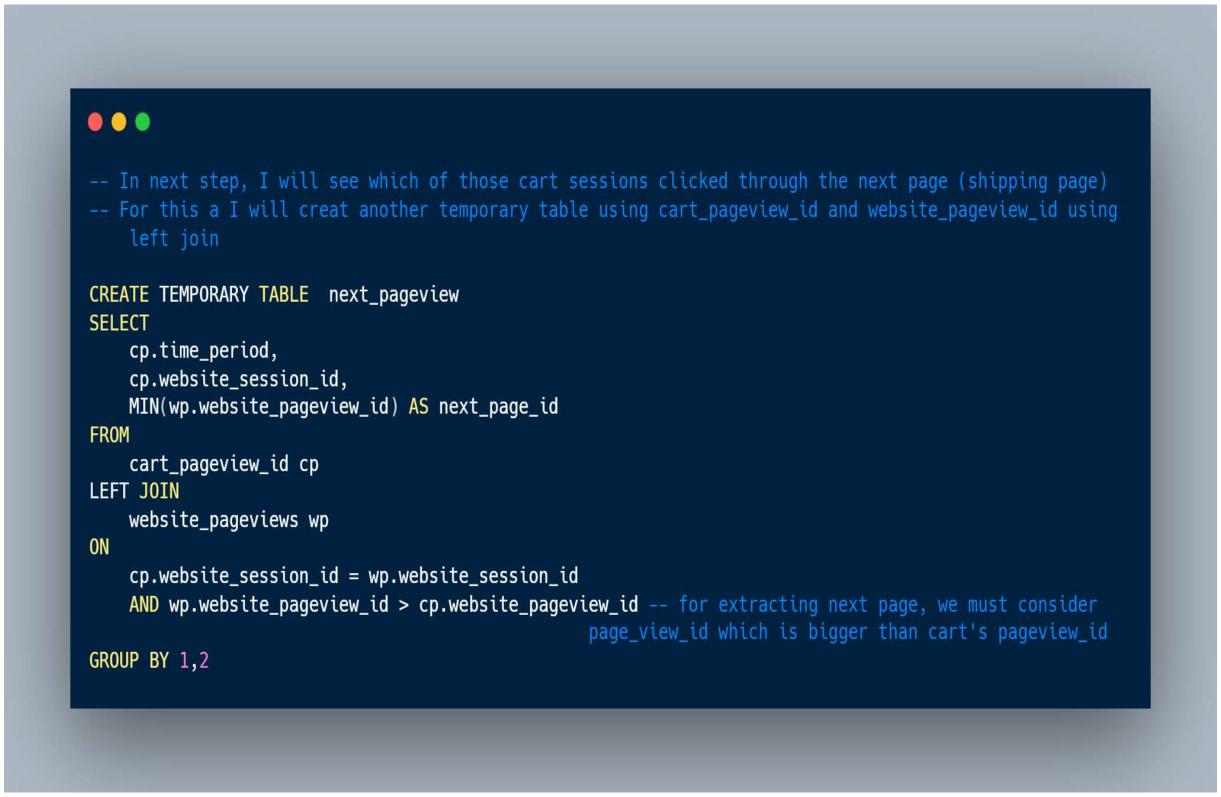
Dataset Schema



Tools Used: MySQL

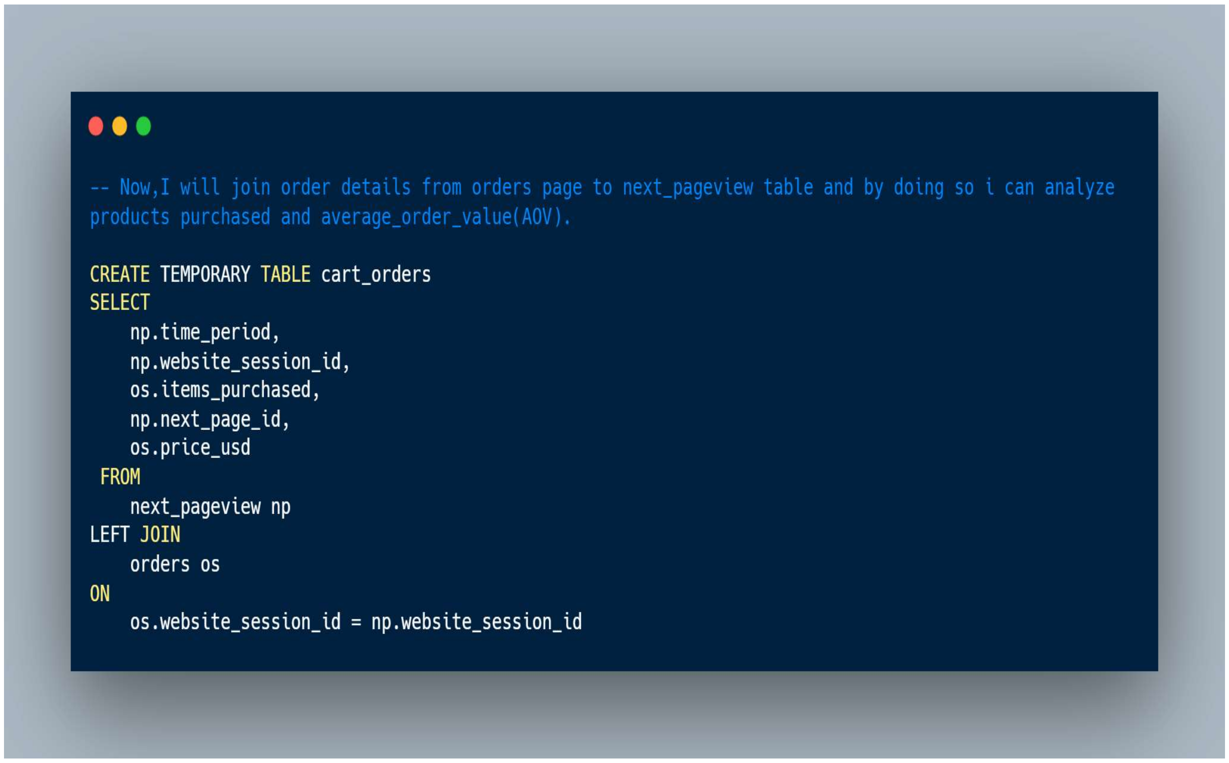


```
-- STEP-1 identifying relevent cart page views with their respective session
-- For this, i will create a temporary table using website_pageview pages and filter the data for
respective dates and cart pageviews
CREATE TEMPORARY TABLE cart_pageview_id
SELECT
    website_session_id,
    created_at,
    website_pageview_id,
    CASE
        WHEN created_at < '2013-09-25' THEN 'A.pre_cross_sell'
        WHEN created_at >= '2013-09-25' THEN 'B.post_cross_sell'
        ELSE 'hi'
    END AS time_period
FROM website_pageviews
WHERE
    created_at > '2013-08-25' AND -- 25th august is one month prior to launch of the new product
    created_at < '2013-10-25' AND -- 25th ocotber is one month later to launch of the new product
    pageview_url = '/cart'
```



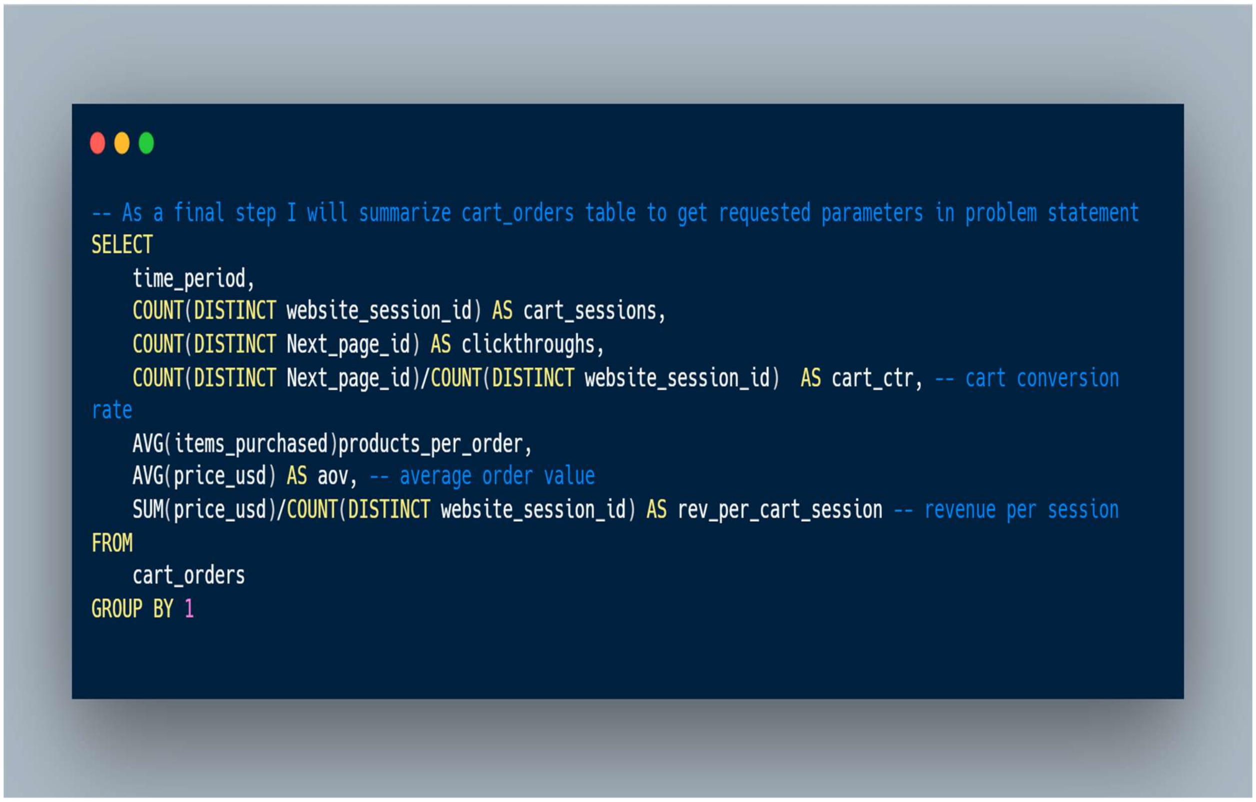
```sql
-- In next step, I will see which of those cart sessions clicked through the next page (shipping page)
-- For this a I will creat another temporary table using cart_pageview_id and website_pageview_id using
    left join

CREATE TEMPORARY TABLE  next_pageview
SELECT
    cp.time_period,
    cp.website_session_id,
    MIN(wp.website_pageview_id) AS next_page_id
FROM
    cart_pageview_id cp
LEFT JOIN
    website_pageviews wp
ON
    cp.website_session_id = wp.website_session_id
    AND wp.website_pageview_id > cp.website_pageview_id -- for extracting next page, we must consider
                                                  page_view_id which is bigger than cart's pageview_id
GROUP BY 1,2
```

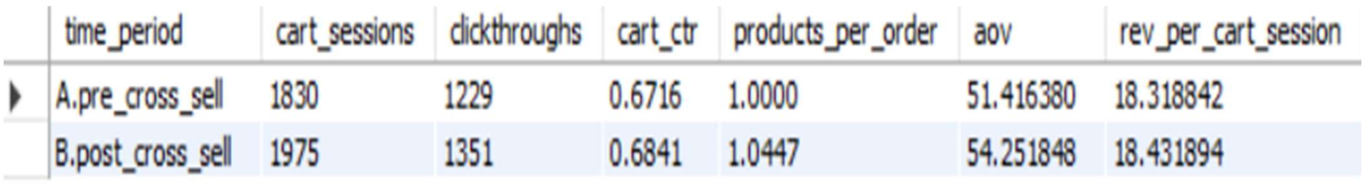


```sql
-- Now,I will join order details from orders page to next_pageview table and by doing so i can analyze
products purchased and average_order_value(AOV).

CREATE TEMPORARY TABLE cart_orders
SELECT
    np.time_period,
    np.website_session_id,
    os.items_purchased,
    np.next_page_id,
    os.price_usd
 FROM
    next_pageview np
LEFT JOIN
    orders os
ON
    os.website_session_id = np.website_session_id
```



```
-- As a final step I will summarize cart_orders table to get requested parameters in problem statement
SELECT
    time_period,
    COUNT(DISTINCT website_session_id) AS cart_sessions,
    COUNT(DISTINCT Next_page_id) AS clickthroughs,
    COUNT(DISTINCT Next_page_id)/COUNT(DISTINCT website_session_id)  AS cart_ctr, -- cart conversion
rate
    AVG(items_purchased)products_per_order,
    AVG(price_usd) AS aov, -- average order value
    SUM(price_usd)/COUNT(DISTINCT website_session_id) AS rev_per_cart_session -- revenue per session
FROM
    cart_orders
GROUP BY 1
```



time_period	cart_sessions	clickthroughs	cart_ctr	products_per_order	aov	rev_per_cart_session
A.pre_cross_sell	1830	1229	0.6716	1.0000	51.416380	18.318842
B.post_cross_sell	1975	1351	0.6841	1.0447	54.251848	18.431894

Conclusion: It looks like the CTR from the /cart page didn't go down and that products per order, AOV, and revenue per /cart session are all up slightly since the cross-sell feature was added.