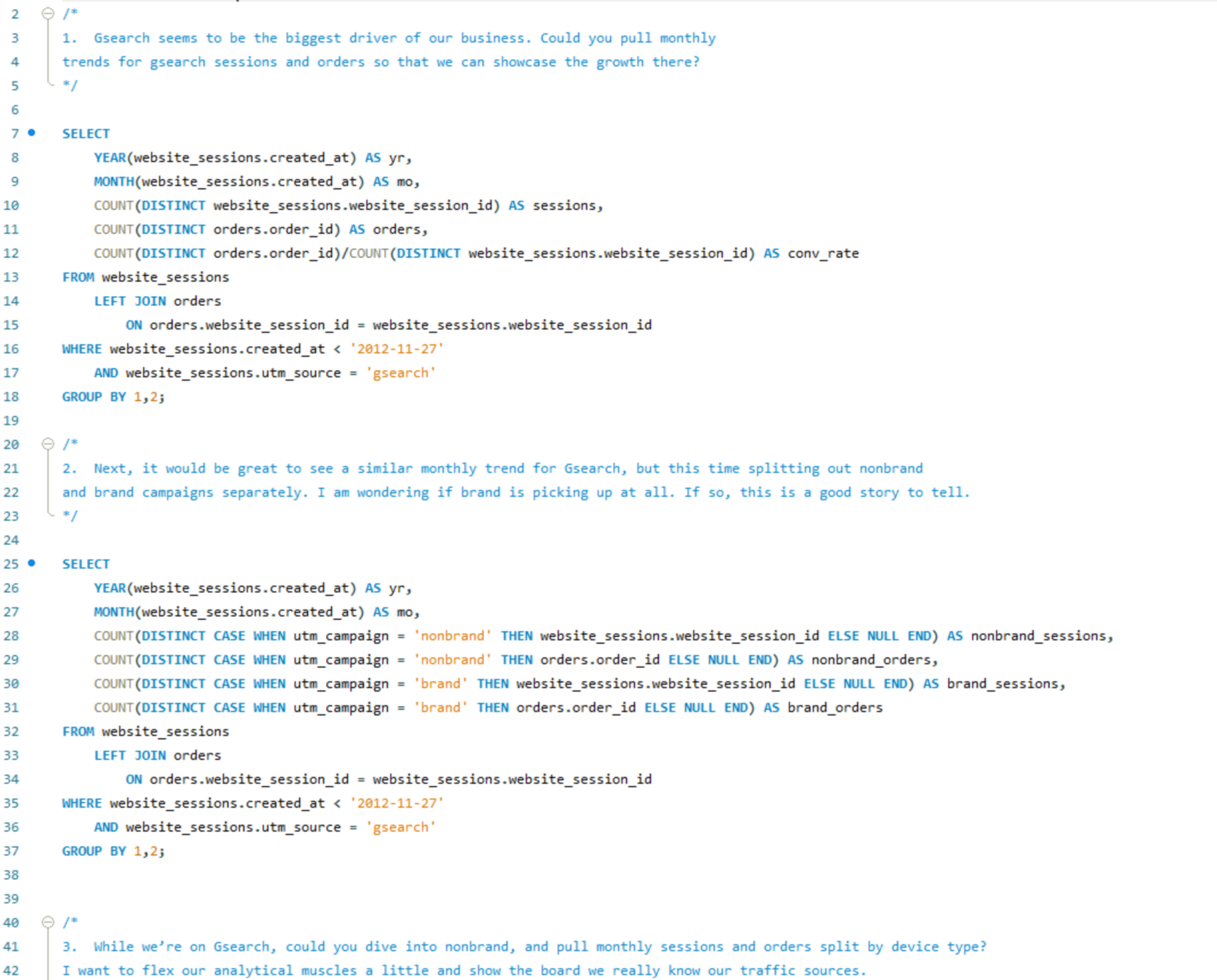
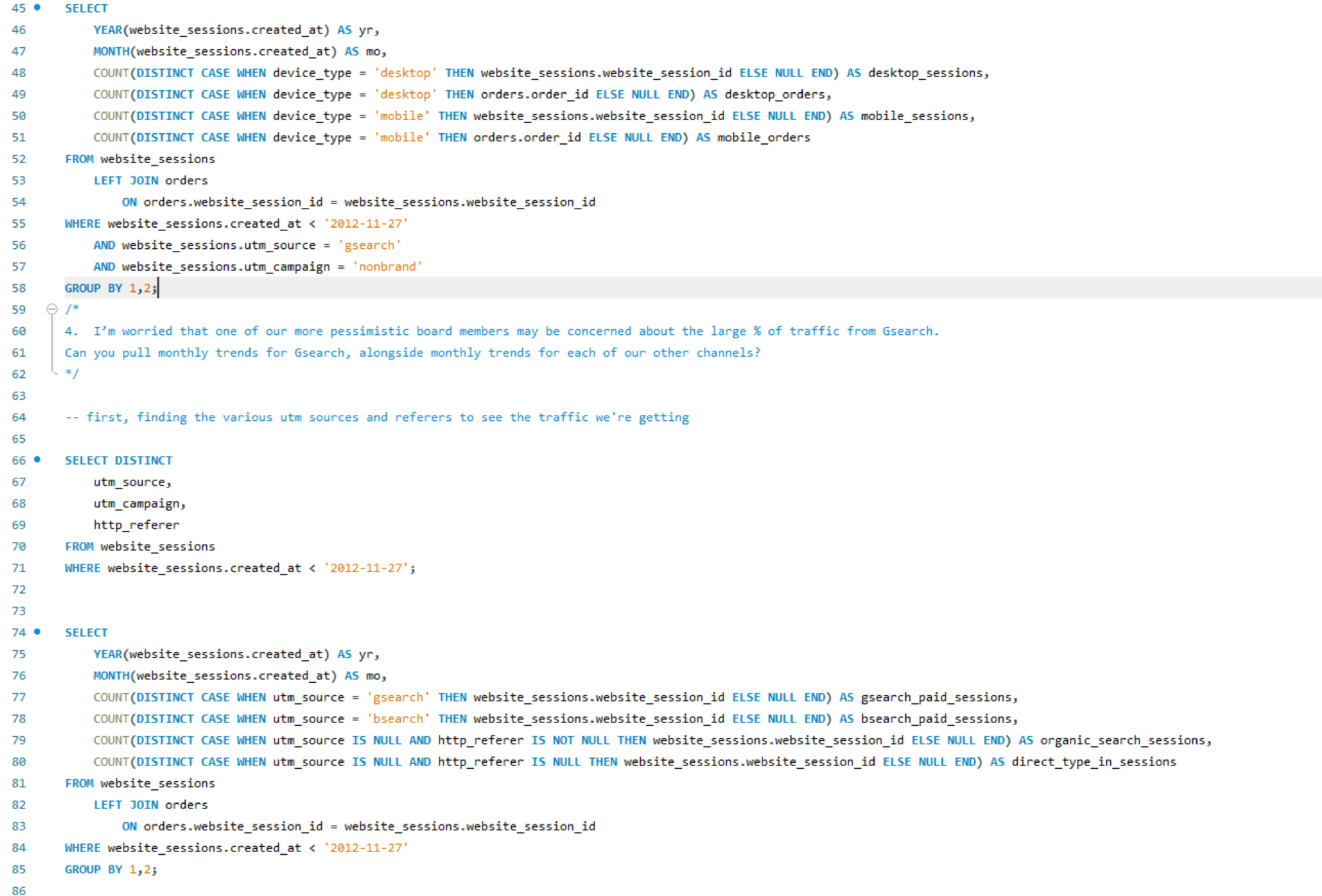
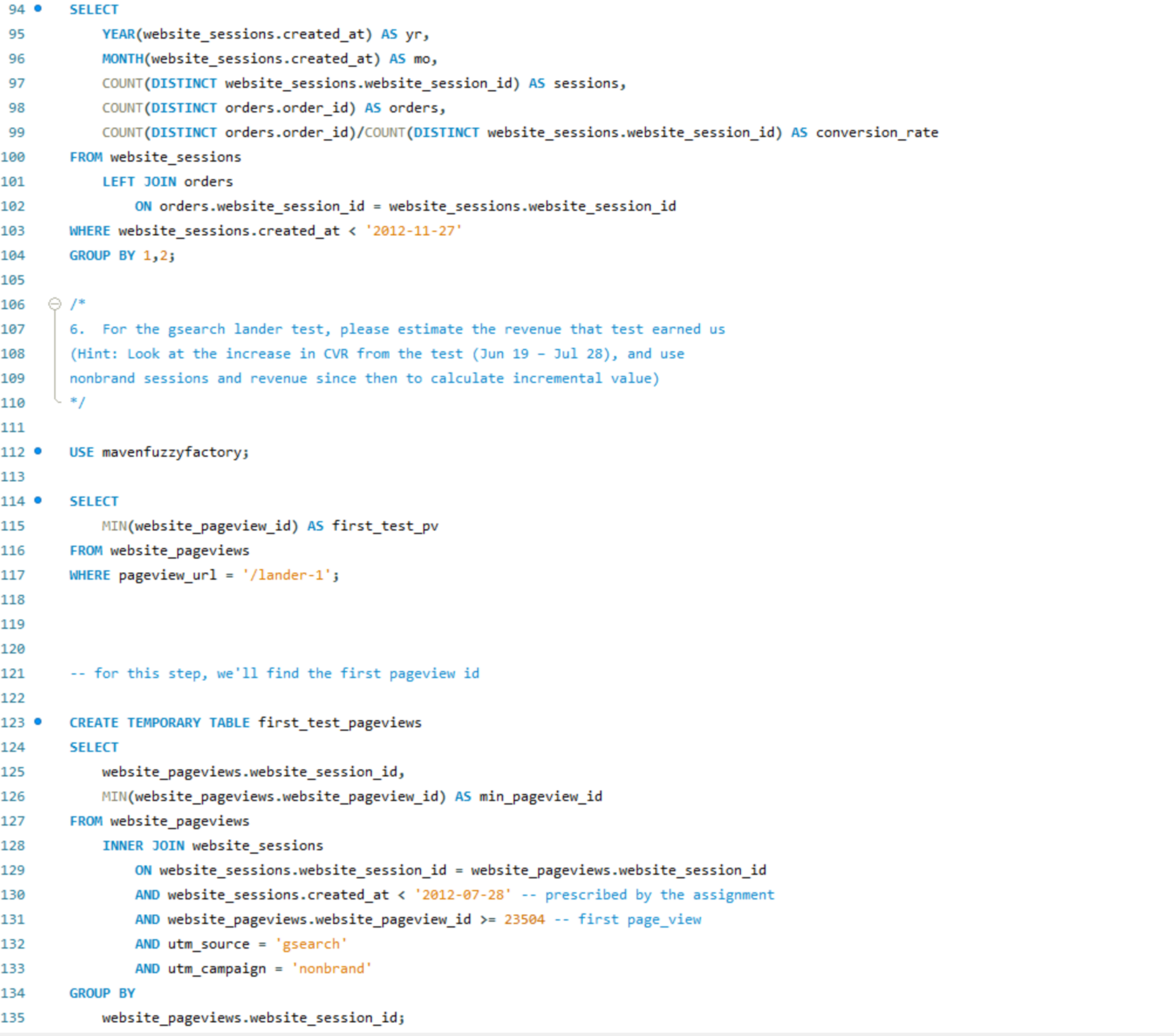


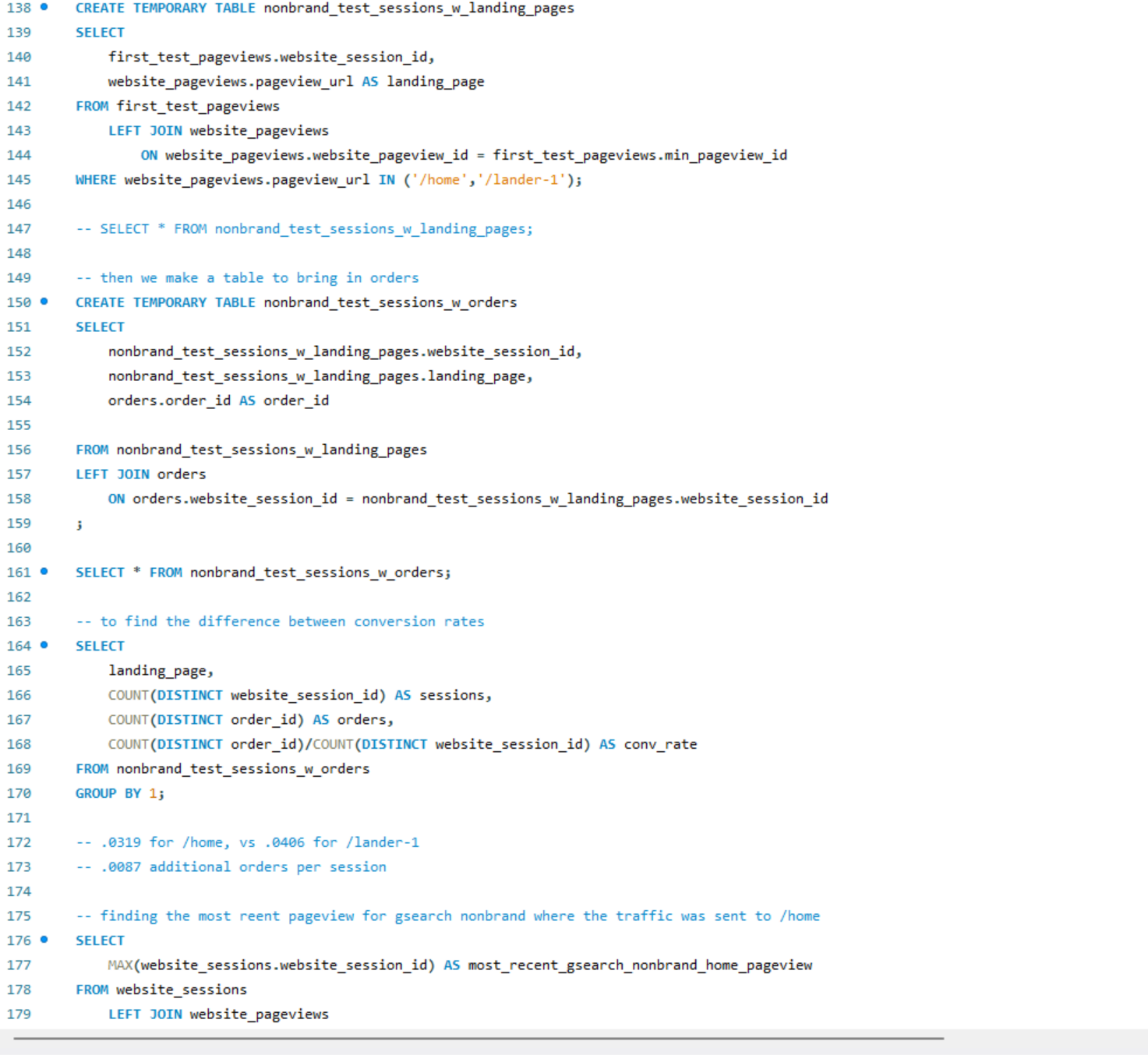
```sql
/*
1.  Gsearch seems to be the biggest driver of our business. Could you pull monthly
trends for gsearch sessions and orders so that we can showcase the growth there?
*/

SELECT
    YEAR(website_sessions.created_at) AS yr,
    MONTH(website_sessions.created_at) AS mo,
    COUNT(DISTINCT website_sessions.website_session_id) AS sessions,
    COUNT(DISTINCT orders.order_id) AS orders,
    COUNT(DISTINCT orders.order_id)/COUNT(DISTINCT website_sessions.website_session_id) AS conv_rate
FROM website_sessions
    LEFT JOIN orders
        ON orders.website_session_id = website_sessions.website_session_id
WHERE website_sessions.created_at < '2012-11-27'
    AND website_sessions.utm_source = 'gsearch'
GROUP BY 1,2;

/*
2.  Next, it would be great to see a similar monthly trend for Gsearch, but this time splitting out nonbrand
and brand campaigns separately. I am wondering if brand is picking up at all. If so, this is a good story to tell.
*/

SELECT
    YEAR(website_sessions.created_at) AS yr,
    MONTH(website_sessions.created_at) AS mo,
    COUNT(DISTINCT CASE WHEN utm_campaign = 'nonbrand' THEN website_sessions.website_session_id ELSE NULL END) AS nonbrand_sessions,
    COUNT(DISTINCT CASE WHEN utm_campaign = 'nonbrand' THEN orders.order_id ELSE NULL END) AS nonbrand_orders,
    COUNT(DISTINCT CASE WHEN utm_campaign = 'brand' THEN website_sessions.website_session_id ELSE NULL END) AS brand_sessions,
    COUNT(DISTINCT CASE WHEN utm_campaign = 'brand' THEN orders.order_id ELSE NULL END) AS brand_orders
FROM website_sessions
    LEFT JOIN orders
        ON orders.website_session_id = website_sessions.website_session_id
WHERE website_sessions.created_at < '2012-11-27'
    AND website_sessions.utm_source = 'gsearch'
GROUP BY 1,2;


/*
3.  While we're on Gsearch, could you dive into nonbrand, and pull monthly sessions and orders split by device type?
I want to flex our analytical muscles a little and show the board we really know our traffic sources.
```



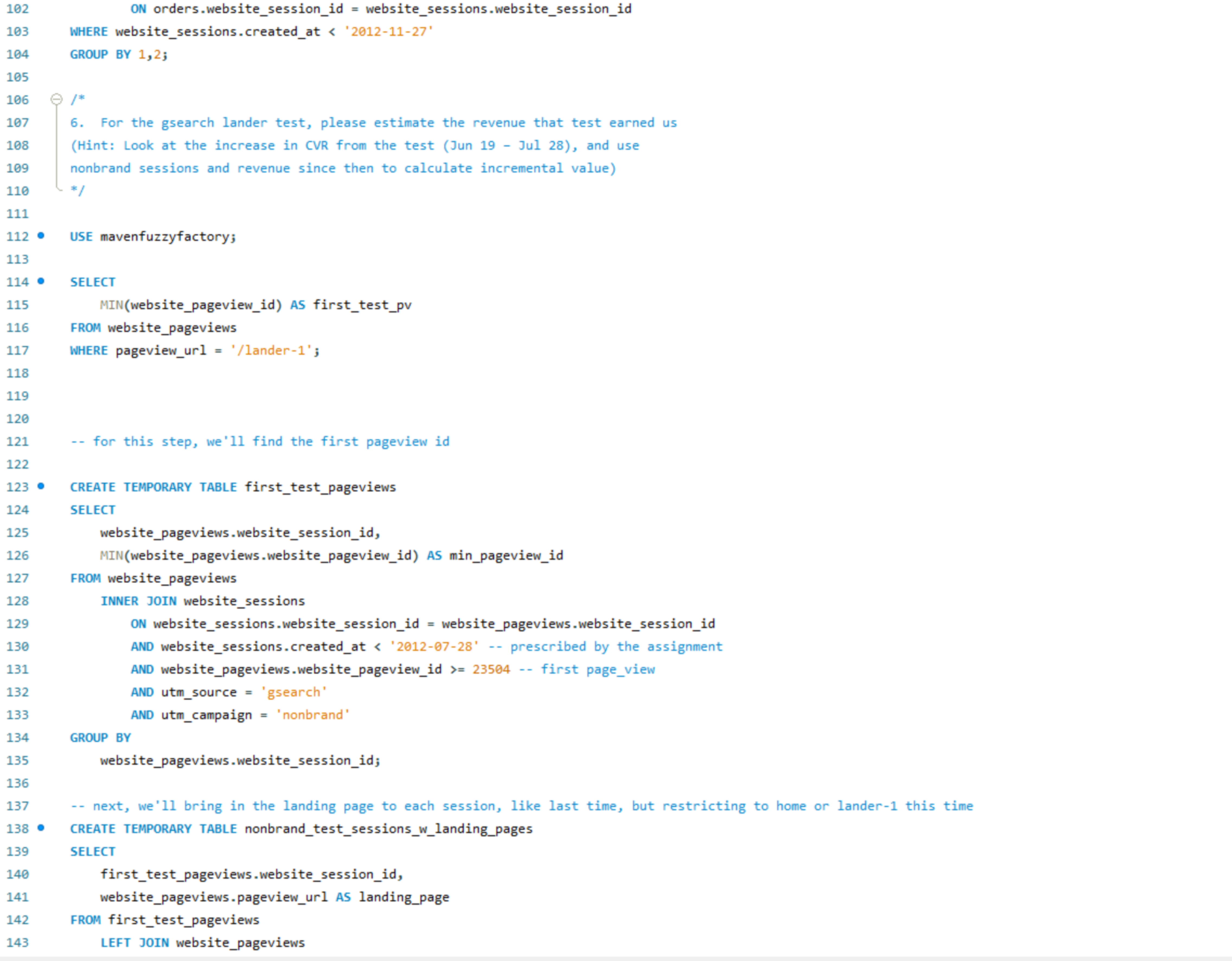
```sql
SELECT
    YEAR(website_sessions.created_at) AS yr,
    MONTH(website_sessions.created_at) AS mo,
    COUNT(DISTINCT CASE WHEN device_type = 'desktop' THEN website_sessions.website_session_id ELSE NULL END) AS desktop_sessions,
    COUNT(DISTINCT CASE WHEN device_type = 'desktop' THEN orders.order_id ELSE NULL END) AS desktop_orders,
    COUNT(DISTINCT CASE WHEN device_type = 'mobile' THEN website_sessions.website_session_id ELSE NULL END) AS mobile_sessions,
    COUNT(DISTINCT CASE WHEN device_type = 'mobile' THEN orders.order_id ELSE NULL END) AS mobile_orders
FROM website_sessions
    LEFT JOIN orders
        ON orders.website_session_id = website_sessions.website_session_id
WHERE website_sessions.created_at < '2012-11-27'
    AND website_sessions.utm_source = 'gsearch'
    AND website_sessions.utm_campaign = 'nonbrand'
GROUP BY 1,2;
/*
4.  I’m worried that one of our more pessimistic board members may be concerned about the large % of traffic from Gsearch.
Can you pull monthly trends for Gsearch, alongside monthly trends for each of our other channels?
*/

-- first, finding the various utm sources and referers to see the traffic we're getting

SELECT DISTINCT
    utm_source,
    utm_campaign,
    http_referer
FROM website_sessions
WHERE website_sessions.created_at < '2012-11-27';


SELECT
    YEAR(website_sessions.created_at) AS yr,
    MONTH(website_sessions.created_at) AS mo,
    COUNT(DISTINCT CASE WHEN utm_source = 'gsearch' THEN website_sessions.website_session_id ELSE NULL END) AS gsearch_paid_sessions,
    COUNT(DISTINCT CASE WHEN utm_source = 'bsearch' THEN website_sessions.website_session_id ELSE NULL END) AS bsearch_paid_sessions,
    COUNT(DISTINCT CASE WHEN utm_source IS NULL AND http_referer IS NOT NULL THEN website_sessions.website_session_id ELSE NULL END) AS organic_search_sessions,
    COUNT(DISTINCT CASE WHEN utm_source IS NULL AND http_referer IS NULL THEN website_sessions.website_session_id ELSE NULL END) AS direct_type_in_sessions
FROM website_sessions
    LEFT JOIN orders
        ON orders.website_session_id = website_sessions.website_session_id
WHERE website_sessions.created_at < '2012-11-27'
GROUP BY 1,2;

```



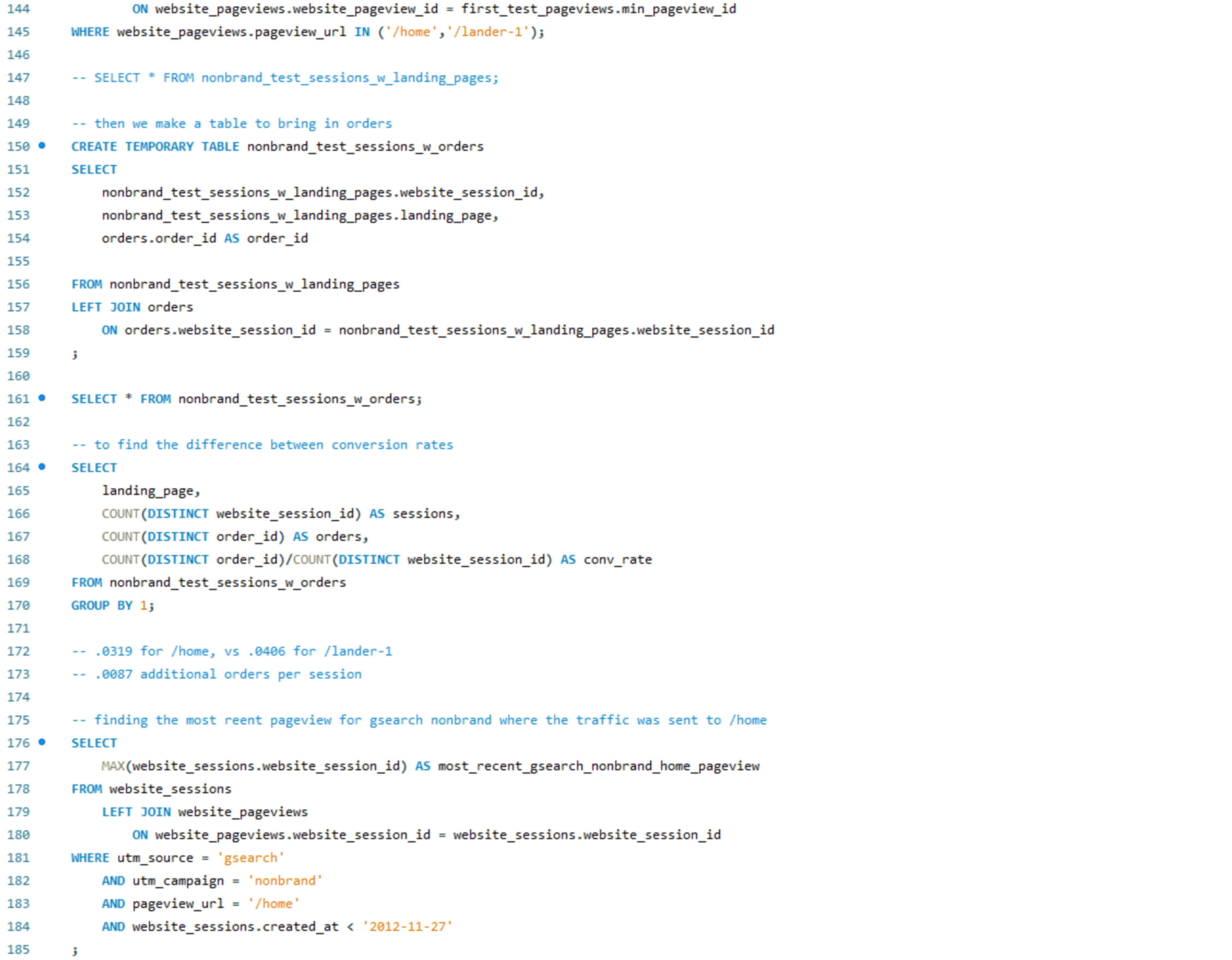
```sql
SELECT
    YEAR(website_sessions.created_at) AS yr,
    MONTH(website_sessions.created_at) AS mo,
    COUNT(DISTINCT website_sessions.website_session_id) AS sessions,
    COUNT(DISTINCT orders.order_id) AS orders,
    COUNT(DISTINCT orders.order_id)/COUNT(DISTINCT website_sessions.website_session_id) AS conversion_rate
FROM website_sessions
    LEFT JOIN orders
        ON orders.website_session_id = website_sessions.website_session_id
WHERE website_sessions.created_at < '2012-11-27'
GROUP BY 1,2;

/*
6.  For the gsearch lander test, please estimate the revenue that test earned us
(Hint: Look at the increase in CVR from the test (Jun 19 - Jul 28), and use
nonbrand sessions and revenue since then to calculate incremental value)
*/

USE mavenfuzzyfactory;

SELECT
    MIN(website_pageview_id) AS first_test_pv
FROM website_pageviews
WHERE pageview_url = '/lander-1';



-- for this step, we'll find the first pageview id

CREATE TEMPORARY TABLE first_test_pageviews
SELECT
    website_pageviews.website_session_id,
    MIN(website_pageviews.website_pageview_id) AS min_pageview_id
FROM website_pageviews
    INNER JOIN website_sessions
        ON website_sessions.website_session_id = website_pageviews.website_session_id
        AND website_sessions.created_at < '2012-07-28' -- prescribed by the assignment
        AND website_pageviews.website_pageview_id >= 23504 -- first page_view
        AND utm_source = 'gsearch'
        AND utm_campaign = 'nonbrand'
GROUP BY
    website_pageviews.website_session_id;
```



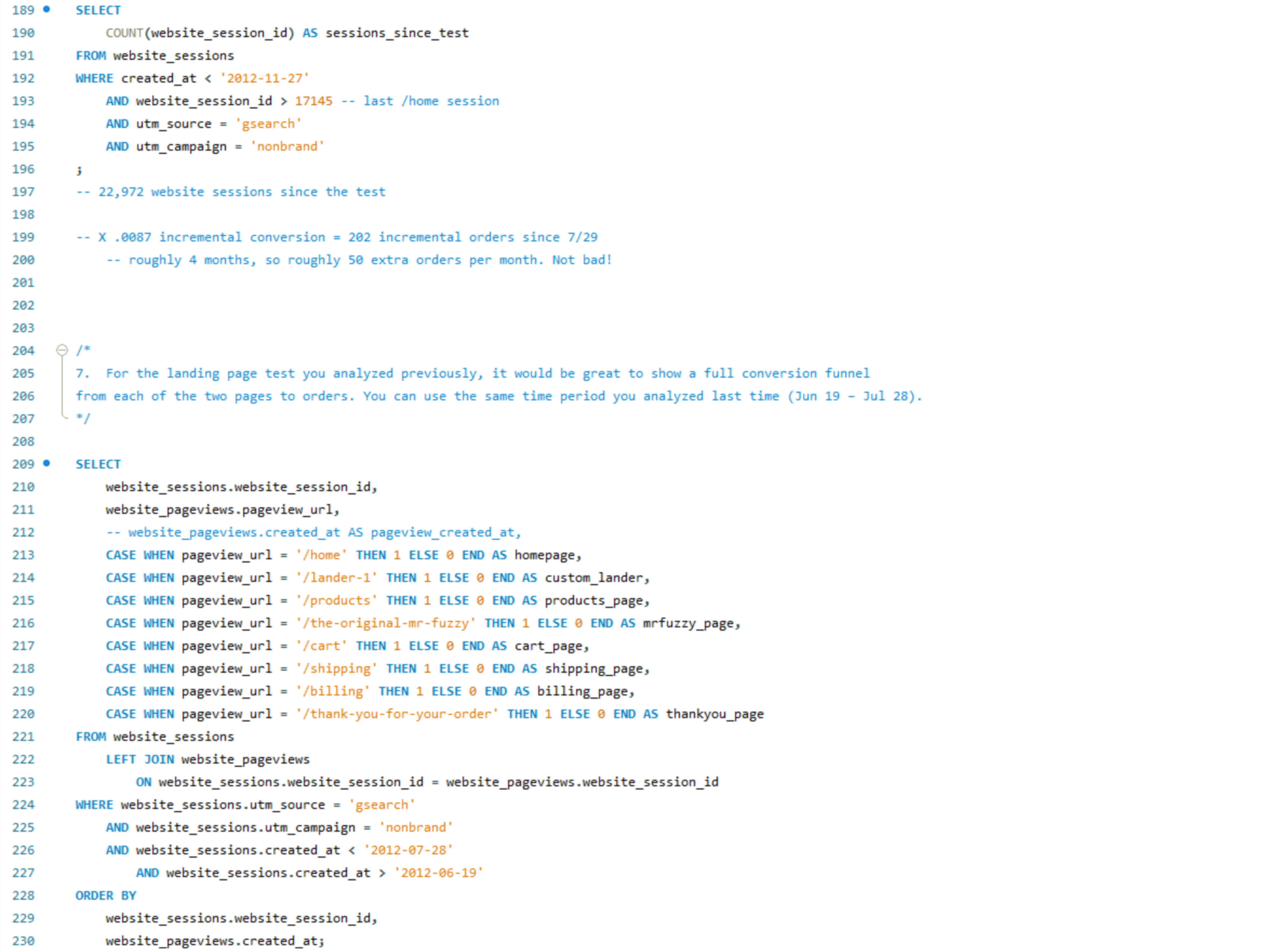
```sql
CREATE TEMPORARY TABLE nonbrand_test_sessions_w_landing_pages
SELECT
    first_test_pageviews.website_session_id,
    website_pageviews.pageview_url AS landing_page
FROM first_test_pageviews
    LEFT JOIN website_pageviews
        ON website_pageviews.website_pageview_id = first_test_pageviews.min_pageview_id
WHERE website_pageviews.pageview_url IN ('/home','/lander-1');

-- SELECT * FROM nonbrand_test_sessions_w_landing_pages;

-- then we make a table to bring in orders
CREATE TEMPORARY TABLE nonbrand_test_sessions_w_orders
SELECT
    nonbrand_test_sessions_w_landing_pages.website_session_id,
    nonbrand_test_sessions_w_landing_pages.landing_page,
    orders.order_id AS order_id

FROM nonbrand_test_sessions_w_landing_pages
LEFT JOIN orders
    ON orders.website_session_id = nonbrand_test_sessions_w_landing_pages.website_session_id
;

SELECT * FROM nonbrand_test_sessions_w_orders;

-- to find the difference between conversion rates
SELECT
    landing_page,
    COUNT(DISTINCT website_session_id) AS sessions,
    COUNT(DISTINCT order_id) AS orders,
    COUNT(DISTINCT order_id)/COUNT(DISTINCT website_session_id) AS conv_rate
FROM nonbrand_test_sessions_w_orders
GROUP BY 1;

-- .0319 for /home, vs .0406 for /lander-1
-- .0087 additional orders per session

-- finding the most reent pageview for gsearch nonbrand where the traffic was sent to /home
SELECT
    MAX(website_sessions.website_session_id) AS most_recent_gsearch_nonbrand_home_pageview
FROM website_sessions
    LEFT JOIN website_pageviews
```



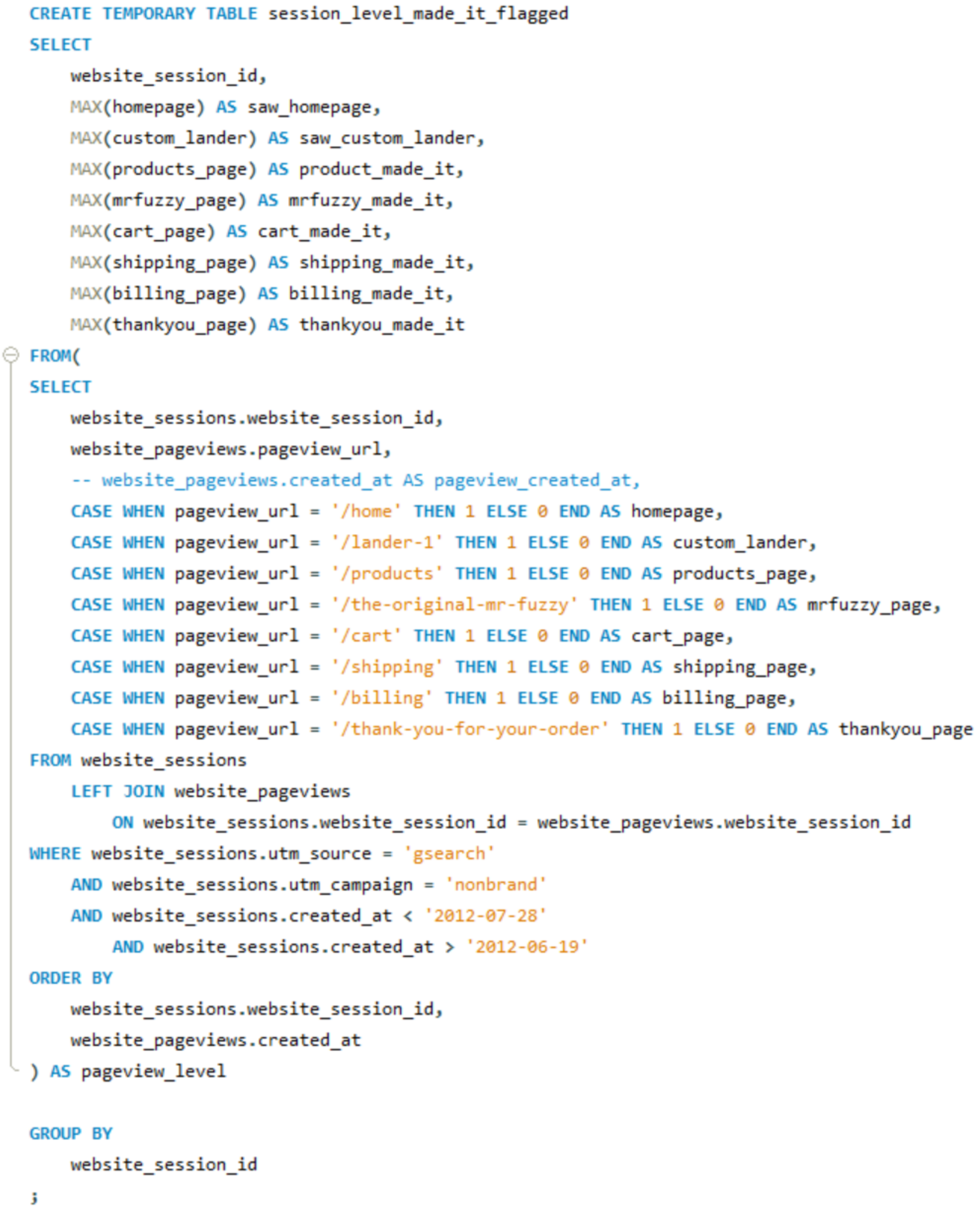
```sql
        ON orders.website_session_id = website_sessions.website_session_id
WHERE website_sessions.created_at < '2012-11-27'
GROUP BY 1,2;

/*
6.  For the gsearch lander test, please estimate the revenue that test earned us
(Hint: Look at the increase in CVR from the test (Jun 19 – Jul 28), and use
nonbrand sessions and revenue since then to calculate incremental value)
*/

USE mavenfuzzyfactory;

SELECT
    MIN(website_pageview_id) AS first_test_pv
FROM website_pageviews
WHERE pageview_url = '/lander-1';



-- for this step, we'll find the first pageview id

CREATE TEMPORARY TABLE first_test_pageviews
SELECT
    website_pageviews.website_session_id,
    MIN(website_pageviews.website_pageview_id) AS min_pageview_id
FROM website_pageviews
    INNER JOIN website_sessions
        ON website_sessions.website_session_id = website_pageviews.website_session_id
        AND website_sessions.created_at < '2012-07-28' -- prescribed by the assignment
        AND website_pageviews.website_pageview_id >= 23504 -- first page_view
        AND utm_source = 'gsearch'
        AND utm_campaign = 'nonbrand'
GROUP BY
    website_pageviews.website_session_id;

-- next, we'll bring in the landing page to each session, like last time, but restricting to home or lander-1 this time
CREATE TEMPORARY TABLE nonbrand_test_sessions_w_landing_pages
SELECT
    first_test_pageviews.website_session_id,
    website_pageviews.pageview_url AS landing_page
FROM first_test_pageviews
    LEFT JOIN website_pageviews
```



```sql
        ON website_pageviews.website_pageview_id = first_test_pageviews.min_pageview_id
WHERE website_pageviews.pageview_url IN ('/home','/lander-1');

-- SELECT * FROM nonbrand_test_sessions_w_landing_pages;

-- then we make a table to bring in orders
CREATE TEMPORARY TABLE nonbrand_test_sessions_w_orders
SELECT
    nonbrand_test_sessions_w_landing_pages.website_session_id,
    nonbrand_test_sessions_w_landing_pages.landing_page,
    orders.order_id AS order_id

FROM nonbrand_test_sessions_w_landing_pages
LEFT JOIN orders
    ON orders.website_session_id = nonbrand_test_sessions_w_landing_pages.website_session_id
;

SELECT * FROM nonbrand_test_sessions_w_orders;

-- to find the difference between conversion rates
SELECT
    landing_page,
    COUNT(DISTINCT website_session_id) AS sessions,
    COUNT(DISTINCT order_id) AS orders,
    COUNT(DISTINCT order_id)/COUNT(DISTINCT website_session_id) AS conv_rate
FROM nonbrand_test_sessions_w_orders
GROUP BY 1;

-- .0319 for /home, vs .0406 for /lander-1
-- .0087 additional orders per session

-- finding the most reent pageview for gsearch nonbrand where the traffic was sent to /home
SELECT
    MAX(website_sessions.website_session_id) AS most_recent_gsearch_nonbrand_home_pageview
FROM website_sessions
    LEFT JOIN website_pageviews
        ON website_pageviews.website_session_id = website_sessions.website_session_id
WHERE utm_source = 'gsearch'
    AND utm_campaign = 'nonbrand'
    AND pageview_url = '/home'
    AND website_sessions.created_at < '2012-11-27'
;
```



```sql
SELECT
    COUNT(website_session_id) AS sessions_since_test
FROM website_sessions
WHERE created_at < '2012-11-27'
    AND website_session_id > 17145 -- last /home session
    AND utm_source = 'gsearch'
    AND utm_campaign = 'nonbrand'
;
-- 22,972 website sessions since the test

-- X .0087 incremental conversion = 202 incremental orders since 7/29
    -- roughly 4 months, so roughly 50 extra orders per month. Not bad!



/*
7.  For the landing page test you analyzed previously, it would be great to show a full conversion funnel
from each of the two pages to orders. You can use the same time period you analyzed last time (Jun 19 - Jul 28).
*/

SELECT
    website_sessions.website_session_id,
    website_pageviews.pageview_url,
    -- website_pageviews.created_at AS pageview_created_at,
    CASE WHEN pageview_url = '/home' THEN 1 ELSE 0 END AS homepage,
    CASE WHEN pageview_url = '/lander-1' THEN 1 ELSE 0 END AS custom_lander,
    CASE WHEN pageview_url = '/products' THEN 1 ELSE 0 END AS products_page,
    CASE WHEN pageview_url = '/the-original-mr-fuzzy' THEN 1 ELSE 0 END AS mrfuzzy_page,
    CASE WHEN pageview_url = '/cart' THEN 1 ELSE 0 END AS cart_page,
    CASE WHEN pageview_url = '/shipping' THEN 1 ELSE 0 END AS shipping_page,
    CASE WHEN pageview_url = '/billing' THEN 1 ELSE 0 END AS billing_page,
    CASE WHEN pageview_url = '/thank-you-for-your-order' THEN 1 ELSE 0 END AS thankyou_page
FROM website_sessions
    LEFT JOIN website_pageviews
        ON website_sessions.website_session_id = website_pageviews.website_session_id
WHERE website_sessions.utm_source = 'gsearch'
    AND website_sessions.utm_campaign = 'nonbrand'
    AND website_sessions.created_at < '2012-07-28'
        AND website_sessions.created_at > '2012-06-19'
ORDER BY
    website_sessions.website_session_id,
    website_pageviews.created_at;
```



```sql
CREATE TEMPORARY TABLE session_level_made_it_flagged
SELECT
    website_session_id,
    MAX(homepage) AS saw_homepage,
    MAX(custom_lander) AS saw_custom_lander,
    MAX(products_page) AS product_made_it,
    MAX(mrfuzzy_page) AS mrfuzzy_made_it,
    MAX(cart_page) AS cart_made_it,
    MAX(shipping_page) AS shipping_made_it,
    MAX(billing_page) AS billing_made_it,
    MAX(thankyou_page) AS thankyou_made_it
FROM(
SELECT
    website_sessions.website_session_id,
    website_pageviews.pageview_url,
    -- website_pageviews.created_at AS pageview_created_at,
    CASE WHEN pageview_url = '/home' THEN 1 ELSE 0 END AS homepage,
    CASE WHEN pageview_url = '/lander-1' THEN 1 ELSE 0 END AS custom_lander,
    CASE WHEN pageview_url = '/products' THEN 1 ELSE 0 END AS products_page,
    CASE WHEN pageview_url = '/the-original-mr-fuzzy' THEN 1 ELSE 0 END AS mrfuzzy_page,
    CASE WHEN pageview_url = '/cart' THEN 1 ELSE 0 END AS cart_page,
    CASE WHEN pageview_url = '/shipping' THEN 1 ELSE 0 END AS shipping_page,
    CASE WHEN pageview_url = '/billing' THEN 1 ELSE 0 END AS billing_page,
    CASE WHEN pageview_url = '/thank-you-for-your-order' THEN 1 ELSE 0 END AS thankyou_page
FROM website_sessions
    LEFT JOIN website_pageviews
        ON website_sessions.website_session_id = website_pageviews.website_session_id
WHERE website_sessions.utm_source = 'gsearch'
    AND website_sessions.utm_campaign = 'nonbrand'
    AND website_sessions.created_at < '2012-07-28'
        AND website_sessions.created_at > '2012-06-19'
ORDER BY
    website_sessions.website_session_id,
    website_pageviews.created_at
) AS pageview_level

GROUP BY
    website_session_id
;
```