Kapil Verma

8WEEKSQLCHALLENGE.COM

CASE STUDY #6

CliqueBait

ATTENTION CAPTURING

DATAWITHDANNY.COM

Introduction:

Welcome to the Clique Bait Case Study!
Founded by CEO Danny, a former member of a digital data analytics team,In this project, we'll support Danny's vision by analyzing the dataset and devising creative solutions to calculate funnel fallout rates.

In this project, we'll delve into our dataset to uncover valuable insights and calculate funnel fallout rates. By understanding where customers drop off in their journey, we can optimize our processes, enhance conversions, and ultimately elevate Clique Bait to new heights.

Challenge :

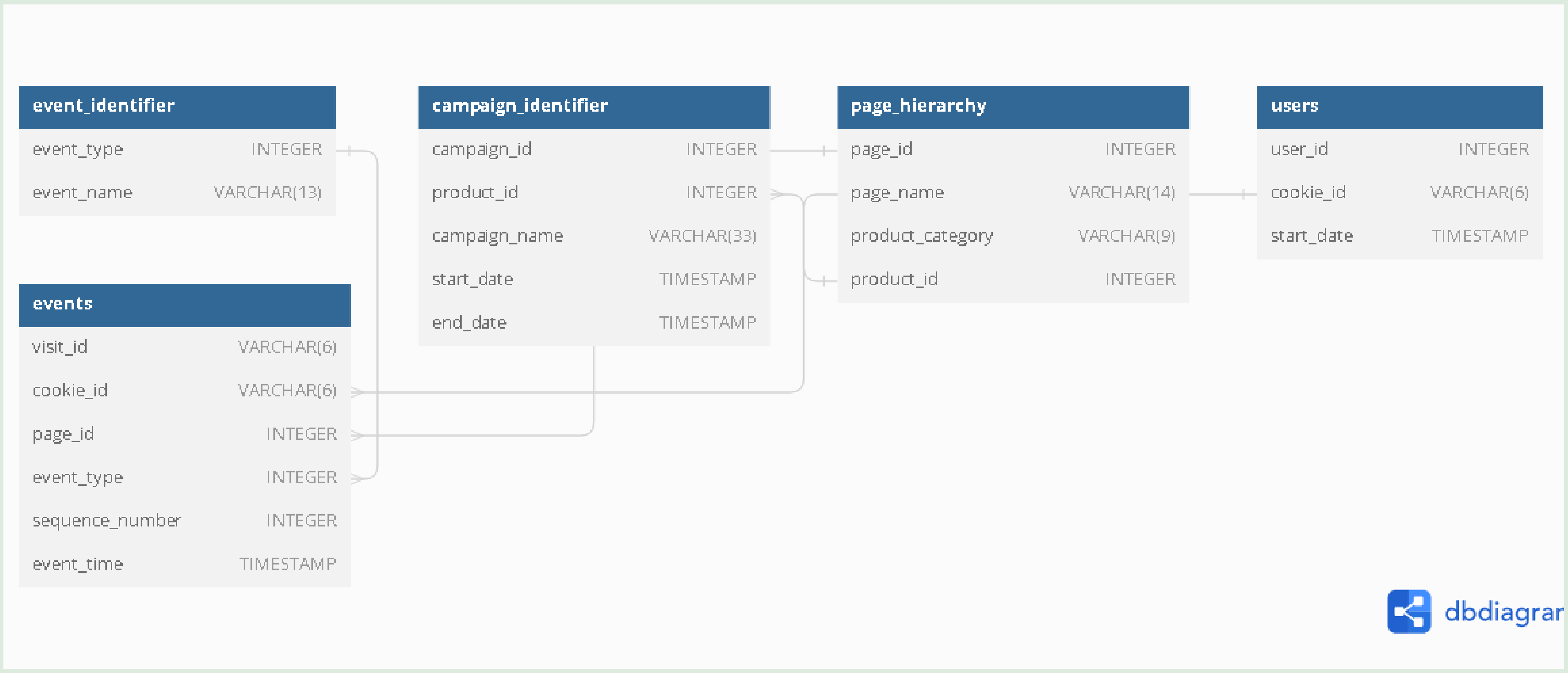
- Digital Analysis
- Product Analysis
- Campaigns Analysis
- Temporal Analysis

Kapil Verma

Tables:

- users
- events
- page hierarchy
- event identifier
- campaign identifier

ER Diagram:



event_identifier	
event_type	INTEGER
event_name	VARCHAR(13)

campaign_identifier	
campaign_id	INTEGER
product_id	INTEGER
campaign_name	VARCHAR(33)
start_date	TIMESTAMP
end_date	TIMESTAMP

page_hierarchy	
page_id	INTEGER
page_name	VARCHAR(14)
product_category	VARCHAR(9)
product_id	INTEGER

users	
user_id	INTEGER
cookie_id	VARCHAR(6)
start_date	TIMESTAMP

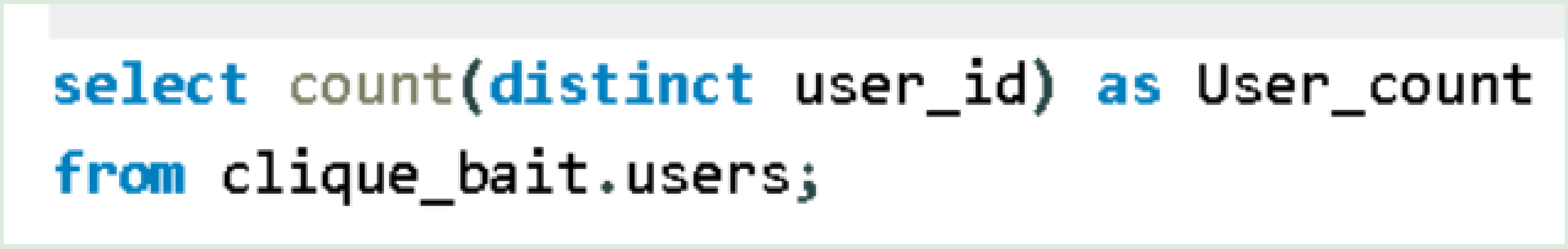
events	
visit_id	VARCHAR(6)
cookie_id	VARCHAR(6)
page_id	INTEGER
event_type	INTEGER
sequence_number	INTEGER
event_time	TIMESTAMP

Part A. Digital Analysis

Kapil Verma

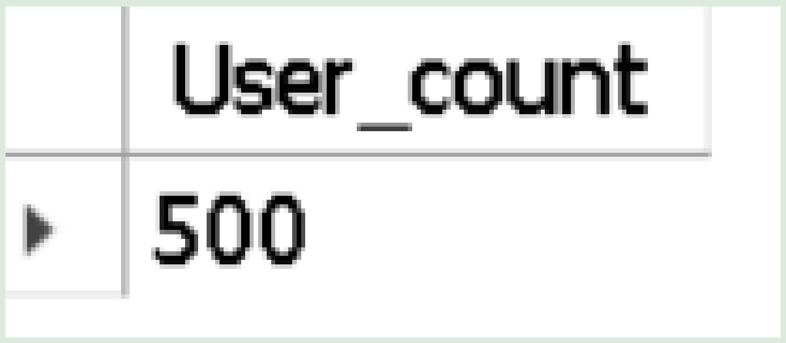
Q1. How many users are there?

Query:



```
select count(distinct user_id) as User_count
from clique_bait.users;
```

Output:

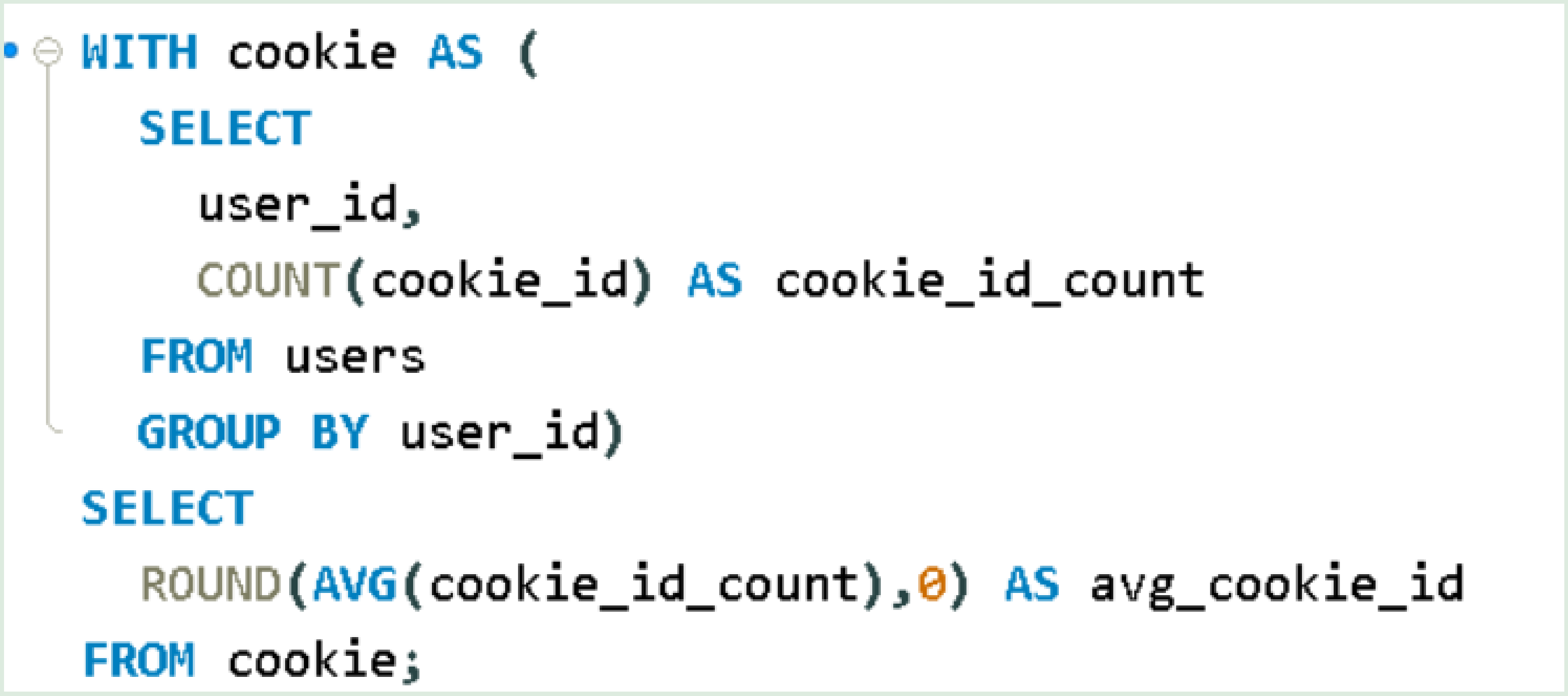


	User_count
▸	500

Kapil Verma

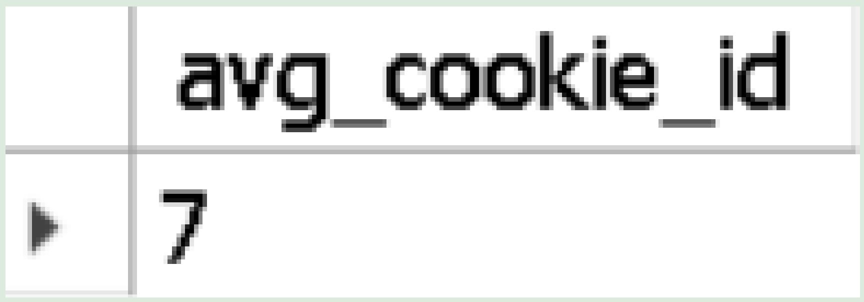
Q2. How many cookies does each user have on average?

Query:



```sql
WITH cookie AS (
   SELECT
      user_id,
      COUNT(cookie_id) AS cookie_id_count
   FROM users
   GROUP BY user_id)
SELECT
   ROUND(AVG(cookie_id_count),0) AS avg_cookie_id
FROM cookie;
```

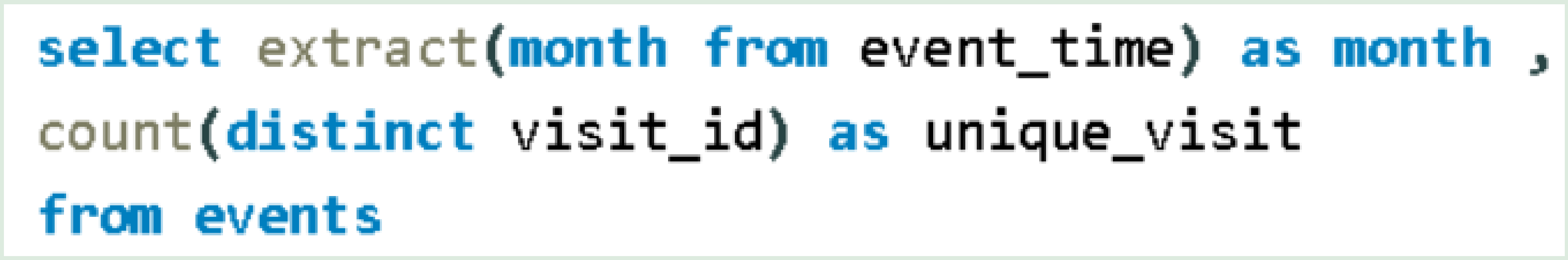
Output:



	avg_cookie_id
▸	7

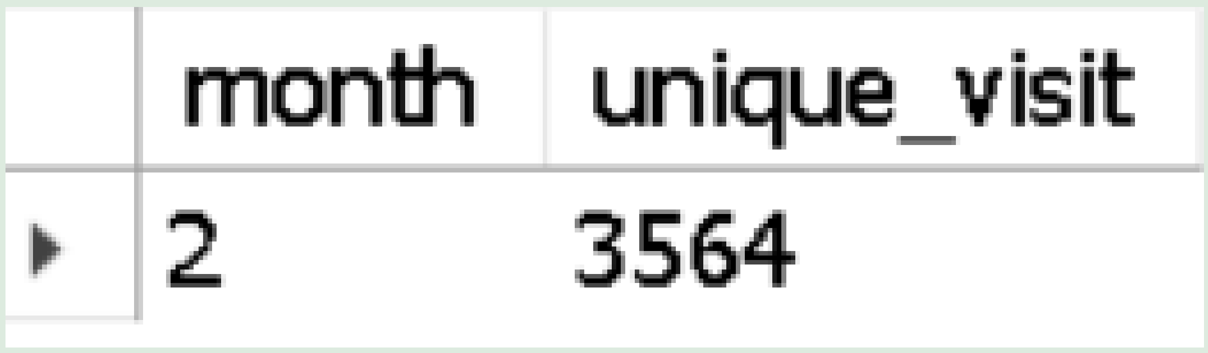
Q3. What is the unique number of visits by all users per month?

Query:



```
select extract(month from event_time) as month ,
count(distinct visit_id) as unique_visit
from events
```

Output:

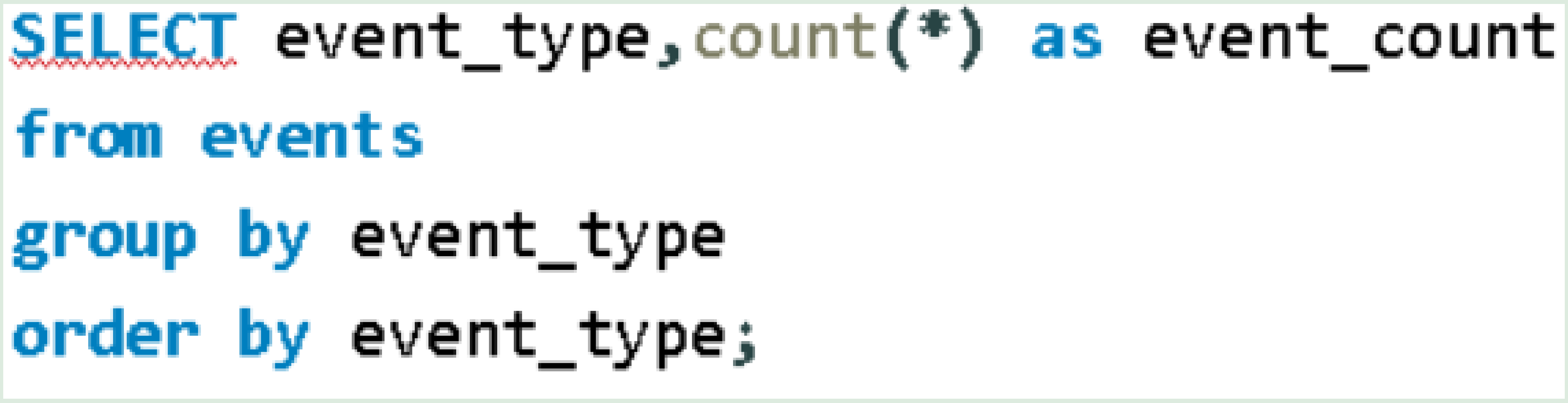


	month	unique_visit
▶	2	3564

Kapil Verma

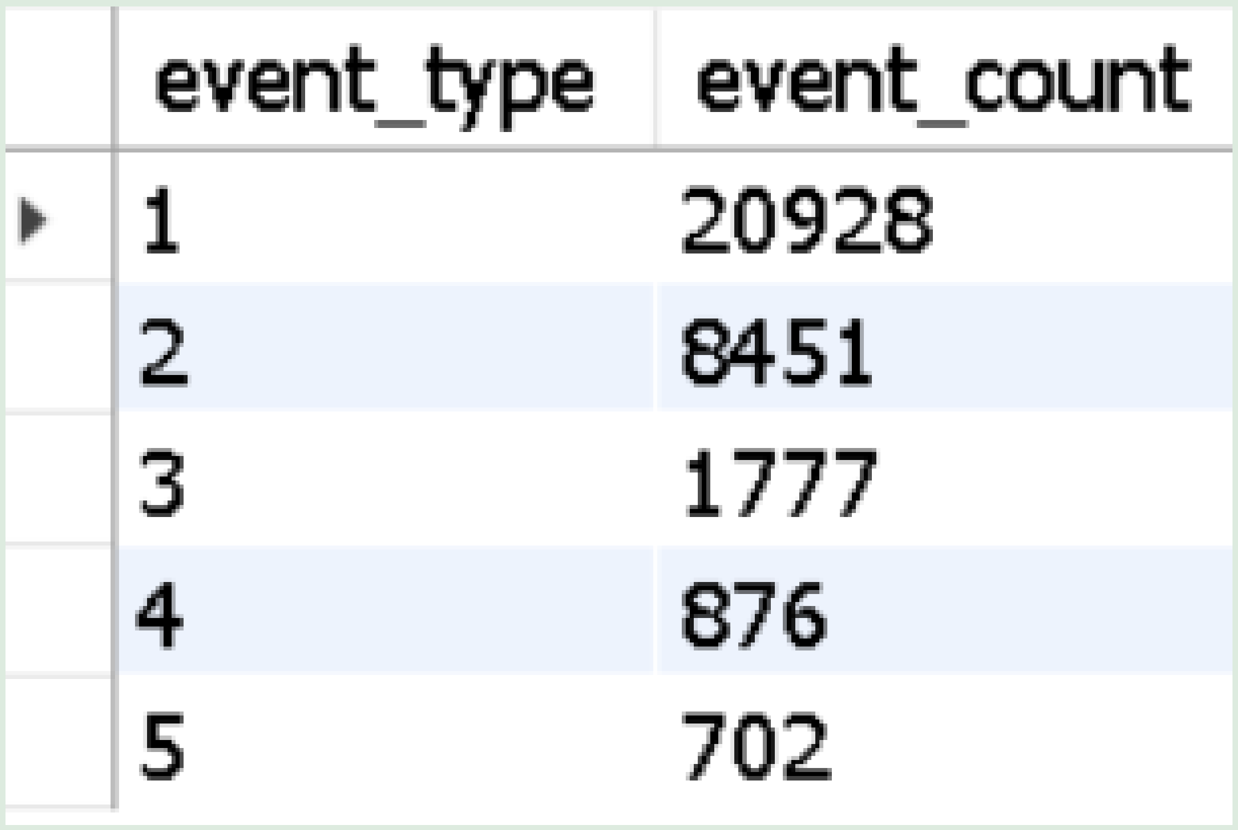
Q4. What is the number of events for each event type?

Query:



```
SELECT event_type,count(*) as event_count
from events
group by event_type
order by event_type;
```

Output:

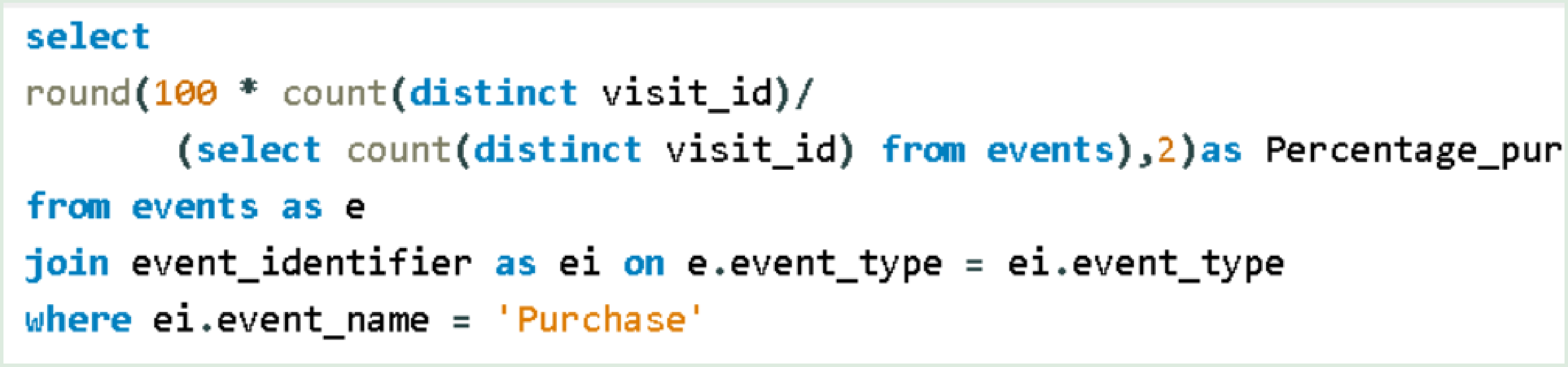


event_type	event_count
1	20928
2	8451
3	1777
4	876
5	702

Kapil Verma

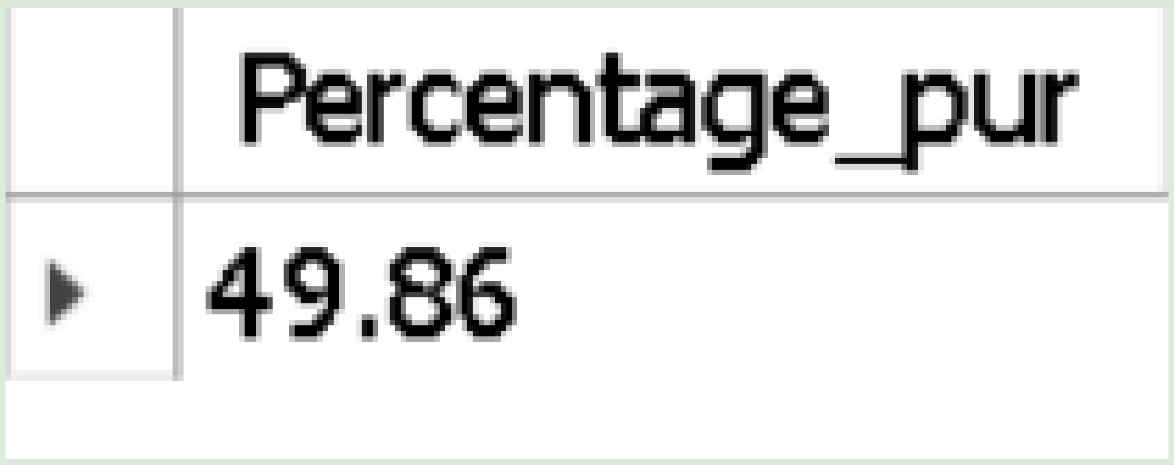
Q5. What is the percentage of visits which have a purchase event?

Query:



```
select
round(100 * count(distinct visit_id)/
      (select count(distinct visit_id) from events),2)as Percentage_pur
from events as e
join event_identifier as ei on e.event_type = ei.event_type
where ei.event_name = 'Purchase'
```

Output:

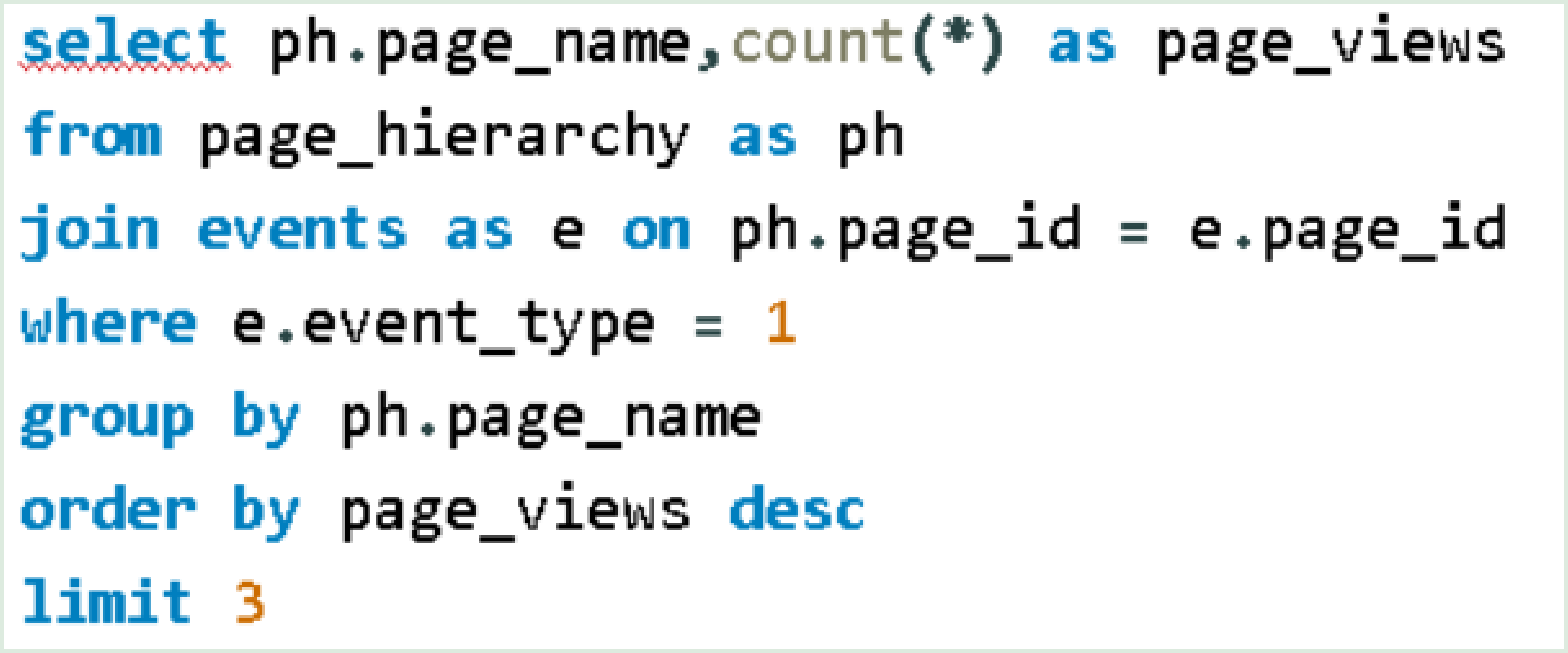


	Percentage_pur
▸	49.86

Kapil Verma

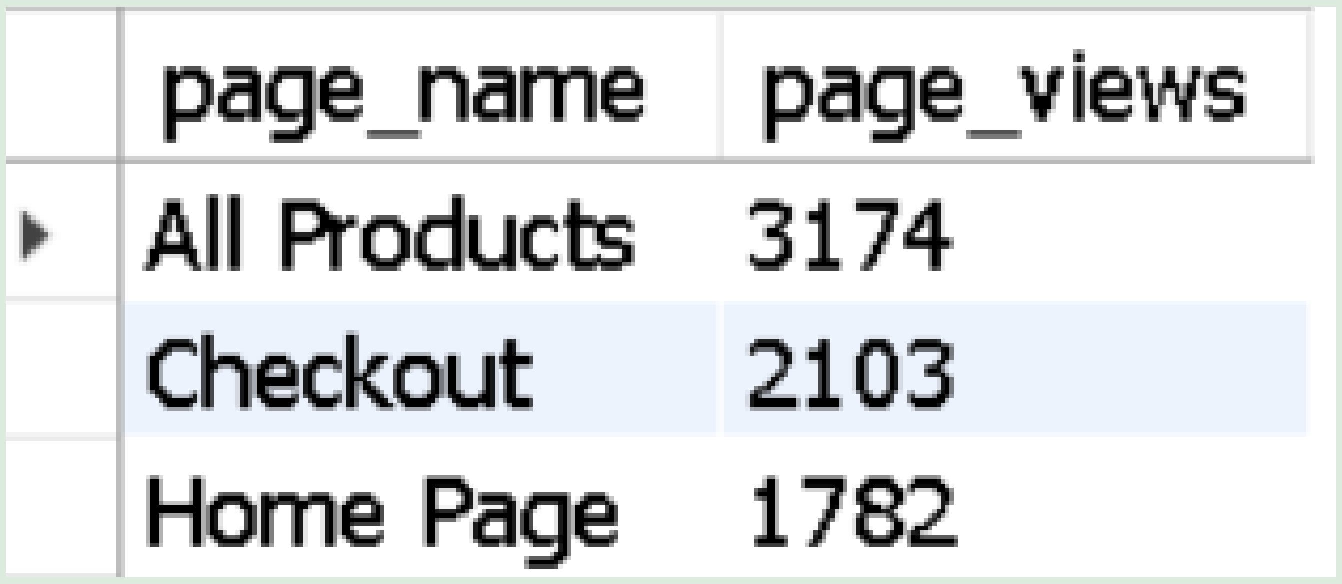
Q6. What are the top 3 pages by number of views?

Query:



```
select ph.page_name,count(*) as page_views
from page_hierarchy as ph
join events as e on ph.page_id = e.page_id
where e.event_type = 1
group by ph.page_name
order by page_views desc
limit 3
```

Output:

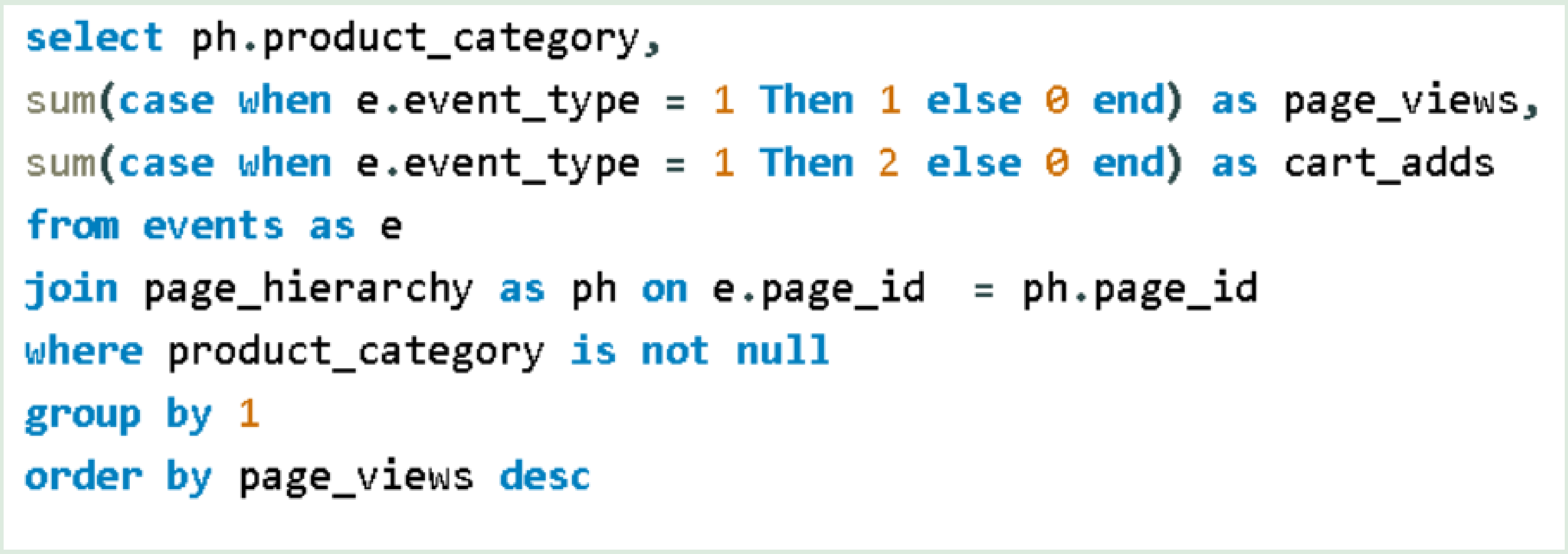


page_name	page_views
All Products	3174
Checkout	2103
Home Page	1782

Kapil Verma

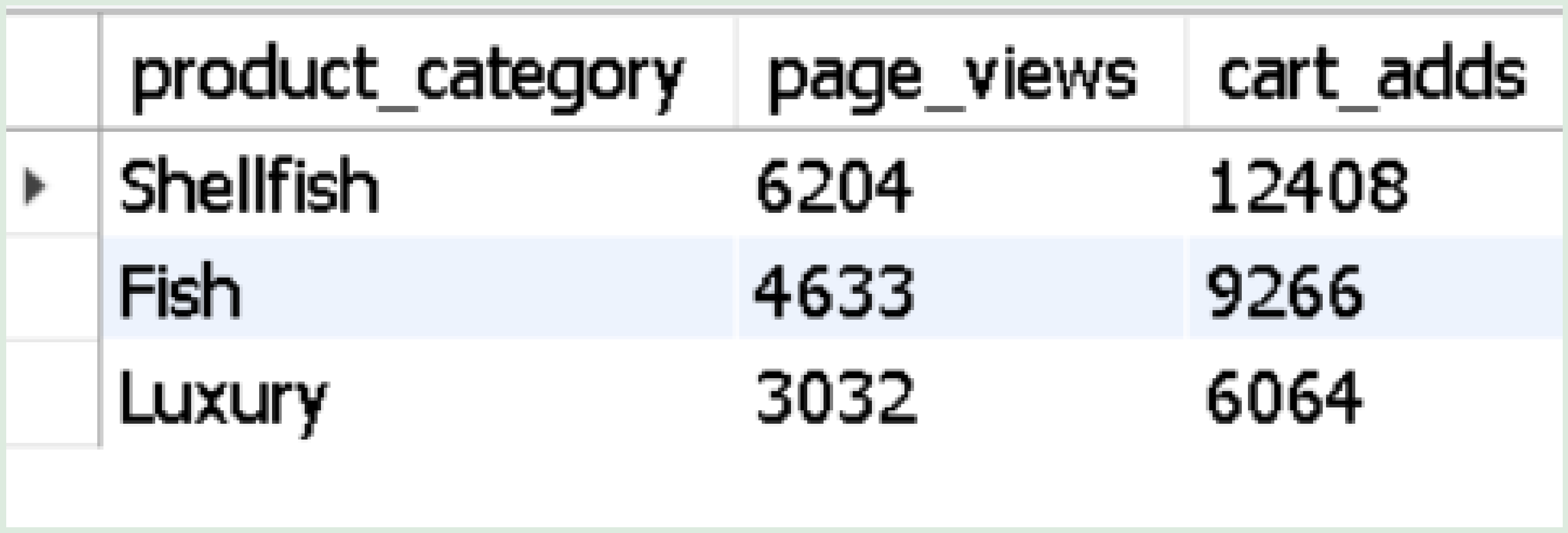
Q7. What is the number of views and cart adds for each product category?

Query:



```
select ph.product_category,
sum(case when e.event_type = 1 Then 1 else 0 end) as page_views,
sum(case when e.event_type = 1 Then 2 else 0 end) as cart_adds
from events as e
join page_hierarchy as ph on e.page_id  = ph.page_id
where product_category is not null
group by 1
order by page_views desc
```

Output:

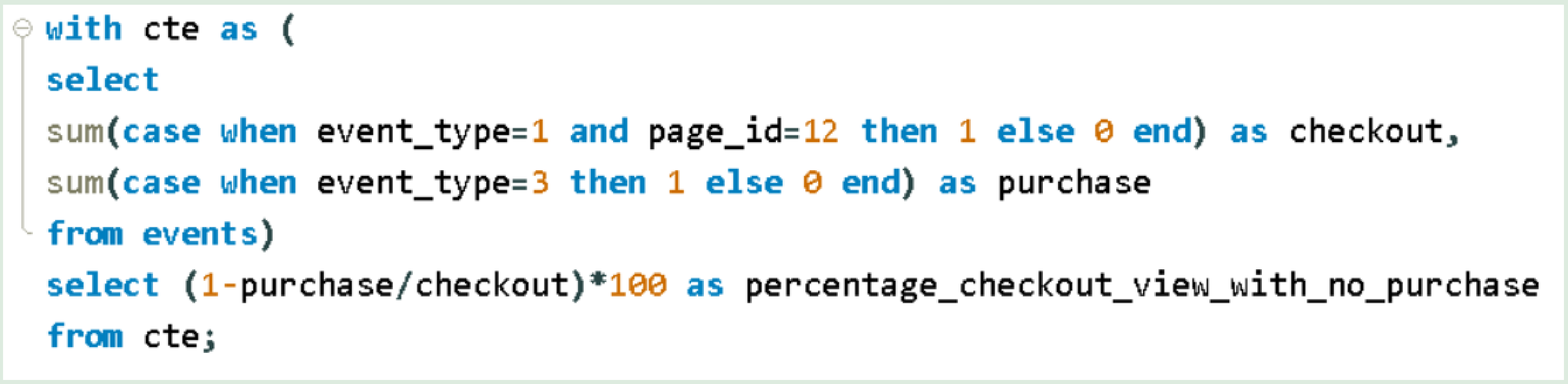


product_category	page_views	cart_adds
Shellfish	6204	12408
Fish	4633	9266
Luxury	3032	6064

Kapil Verma

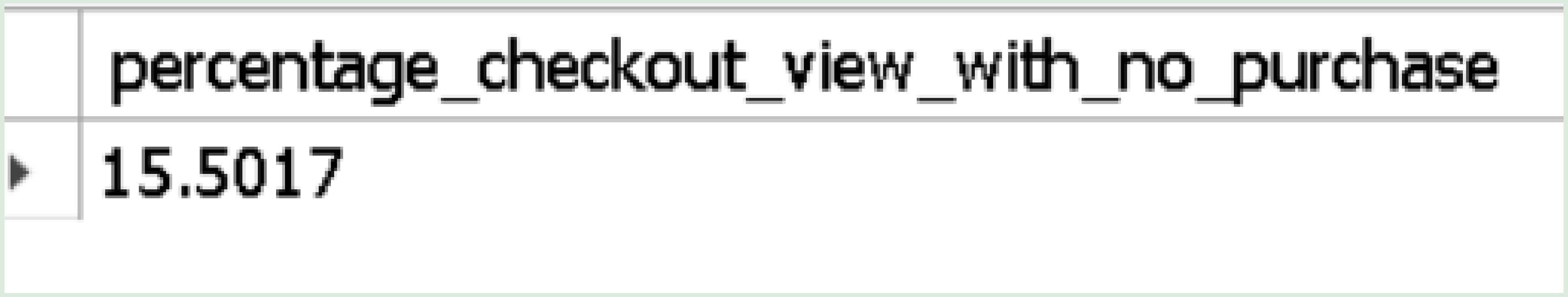
Q8. What is the percentage of visits which view the checkout page but do not have a purchase event?

Query:



```
with cte as (
select
sum(case when event_type=1 and page_id=12 then 1 else 0 end) as checkout,
sum(case when event_type=3 then 1 else 0 end) as purchase
from events)
select (1-purchase/checkout)*100 as percentage_checkout_view_with_no_purchase
from cte;
```

Output:



```
percentage_checkout_view_with_no_purchase
15.5017
```

Kapil Verma

Part B. Product Funnel Analysis

Using a single SQL query - create a new output table which has the following details:

1. How many times was each product viewed?
2. How many times was each product added to cart?
3. How many times was each product added to a cart but not purchased (abandoned)?
4. How many times was each product purchased?

Planning Our Strategy

Column	Description
product	Name of the product
views	Number of views for each product
cart_adds	Number of cart adds for each product
abandoned	Number of times product was added to a cart, but not purchased
purchased	Number of times product was purchased

These information would come from these 2 tables.

- events table - visit_id, page_id, event_type
- page_hierarchy table - page_id, product_category

Solution:

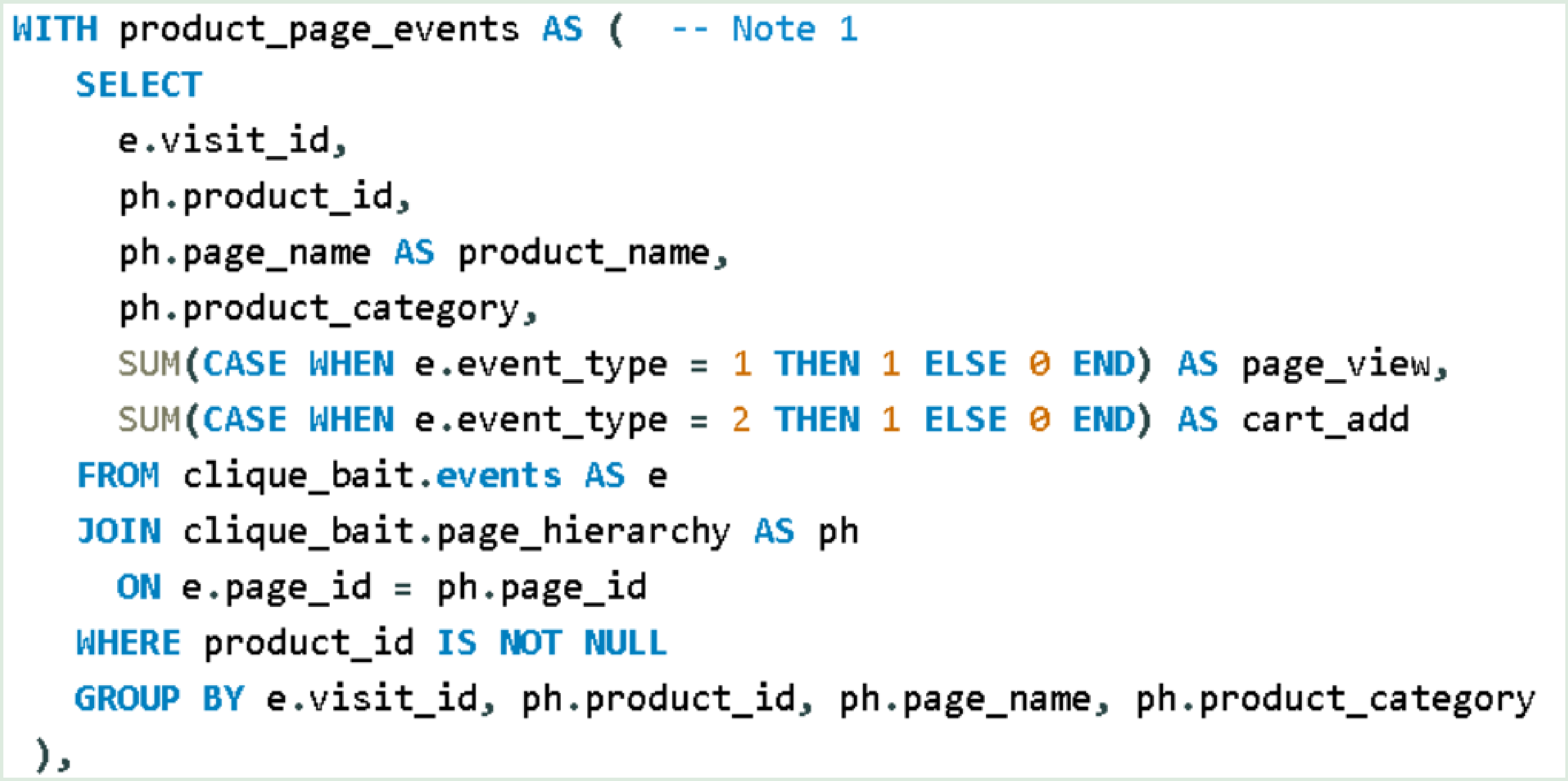
- **Note 1** - In product_page_events CTE, find page views and cart adds for individual visit ids by wrapping SUM around CASE statements so that we do not have to group the results by event_type as well.

- **Note 2** - In purchase_events CTE, get only visit ids that have made purchases.

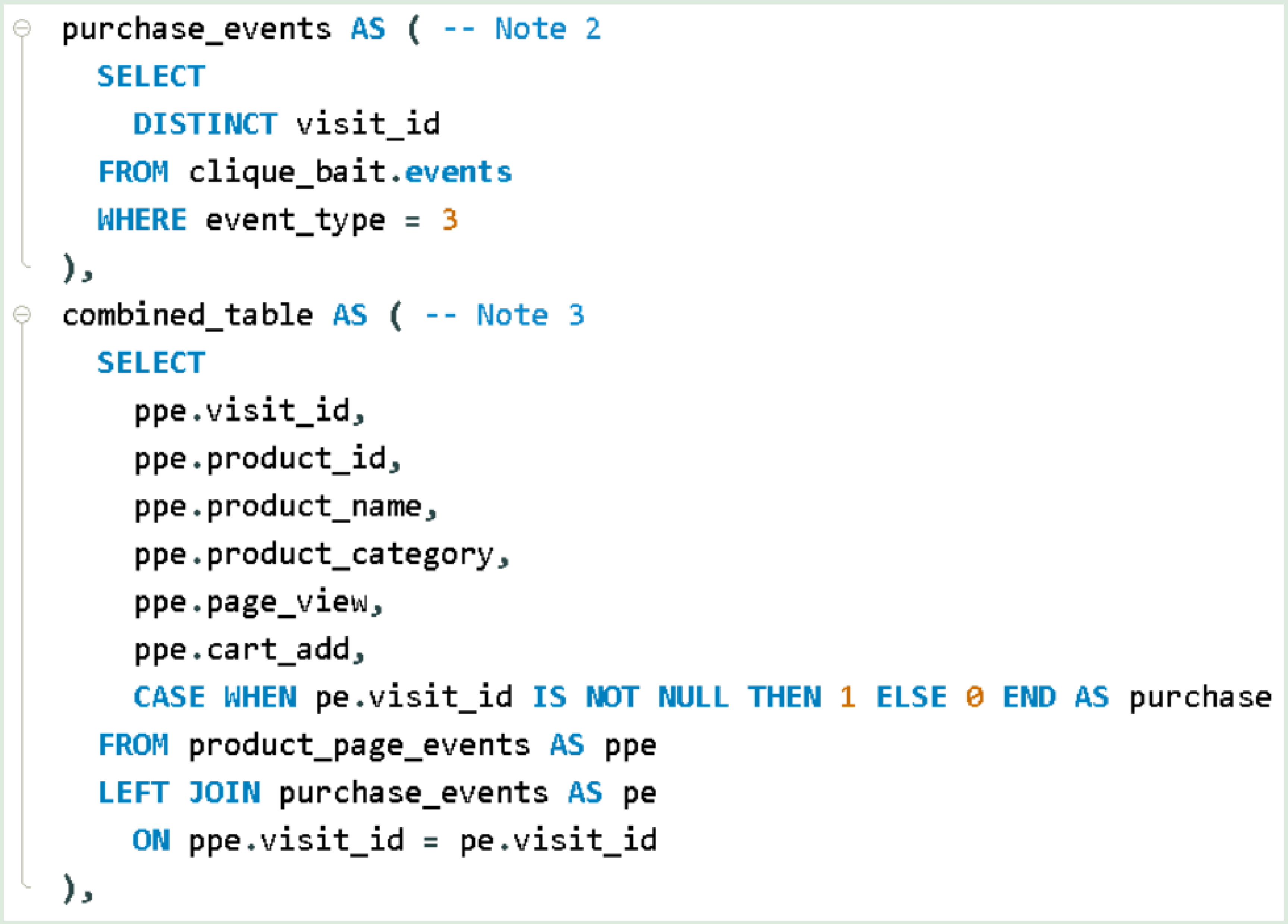
- **Note 3** - In combined_table CTE, merge roduct_page_events and purchase_events using LEFT JOIN. Take note of the table sequence. In order to filter for visit ids with purchases, we use a CASE statement and where visit id is not null, it means the visit id is a purchase.

Kapil Verma

Query:



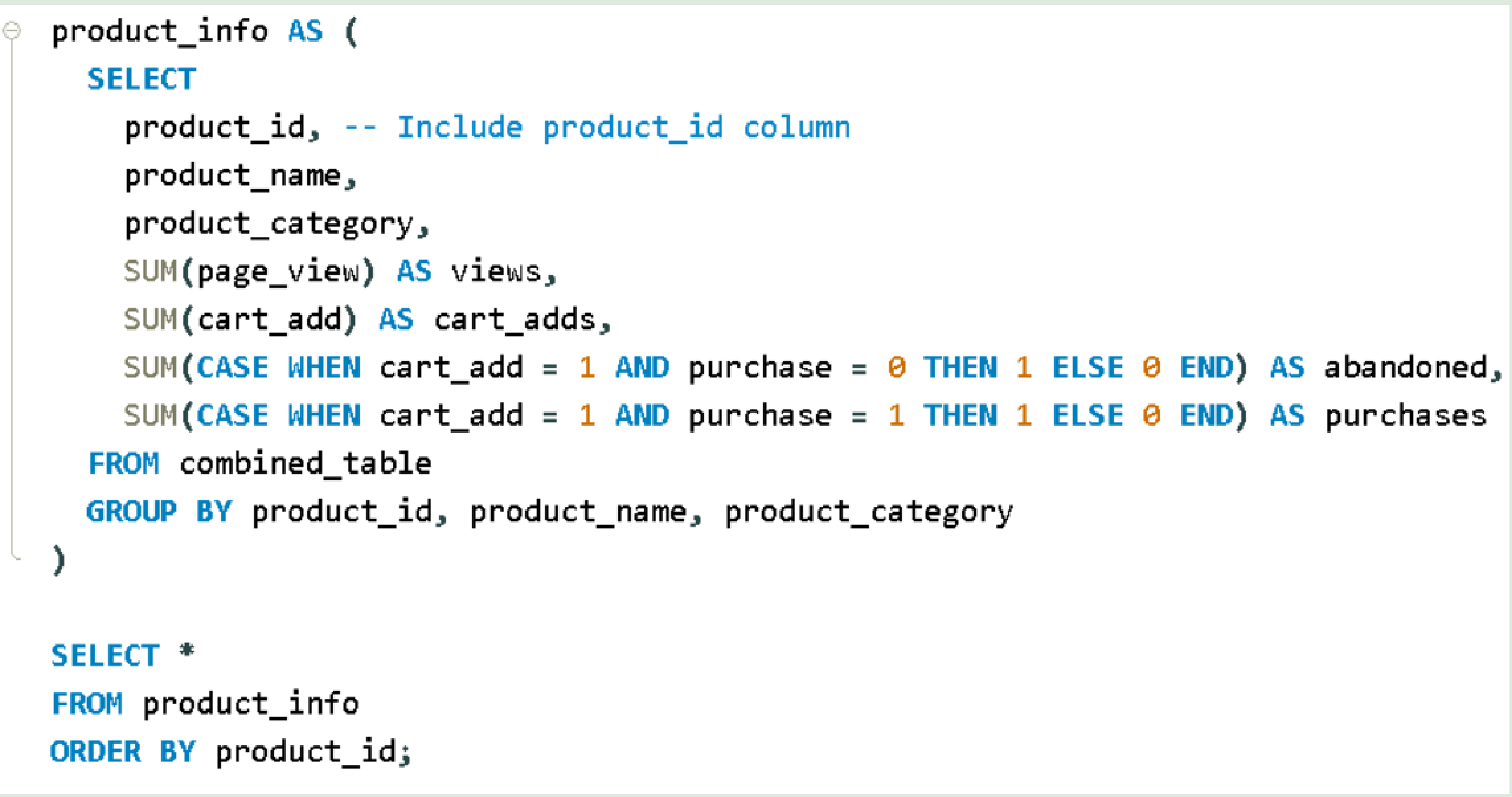
```sql
WITH product_page_events AS (  -- Note 1
  SELECT
    e.visit_id,
    ph.product_id,
    ph.page_name AS product_name,
    ph.product_category,
    SUM(CASE WHEN e.event_type = 1 THEN 1 ELSE 0 END) AS page_view,
    SUM(CASE WHEN e.event_type = 2 THEN 1 ELSE 0 END) AS cart_add
  FROM clique_bait.events AS e
  JOIN clique_bait.page_hierarchy AS ph
    ON e.page_id = ph.page_id
  WHERE product_id IS NOT NULL
  GROUP BY e.visit_id, ph.product_id, ph.page_name, ph.product_category
),
```



```sql
purchase_events AS ( -- Note 2
  SELECT
    DISTINCT visit_id
  FROM clique_bait.events
  WHERE event_type = 3
),
combined_table AS ( -- Note 3
  SELECT
    ppe.visit_id,
    ppe.product_id,
    ppe.product_name,
    ppe.product_category,
    ppe.page_view,
    ppe.cart_add,
    CASE WHEN pe.visit_id IS NOT NULL THEN 1 ELSE 0 END AS purchase
  FROM product_page_events AS ppe
  LEFT JOIN purchase_events AS pe
    ON ppe.visit_id = pe.visit_id
),
```

Kapil Verma

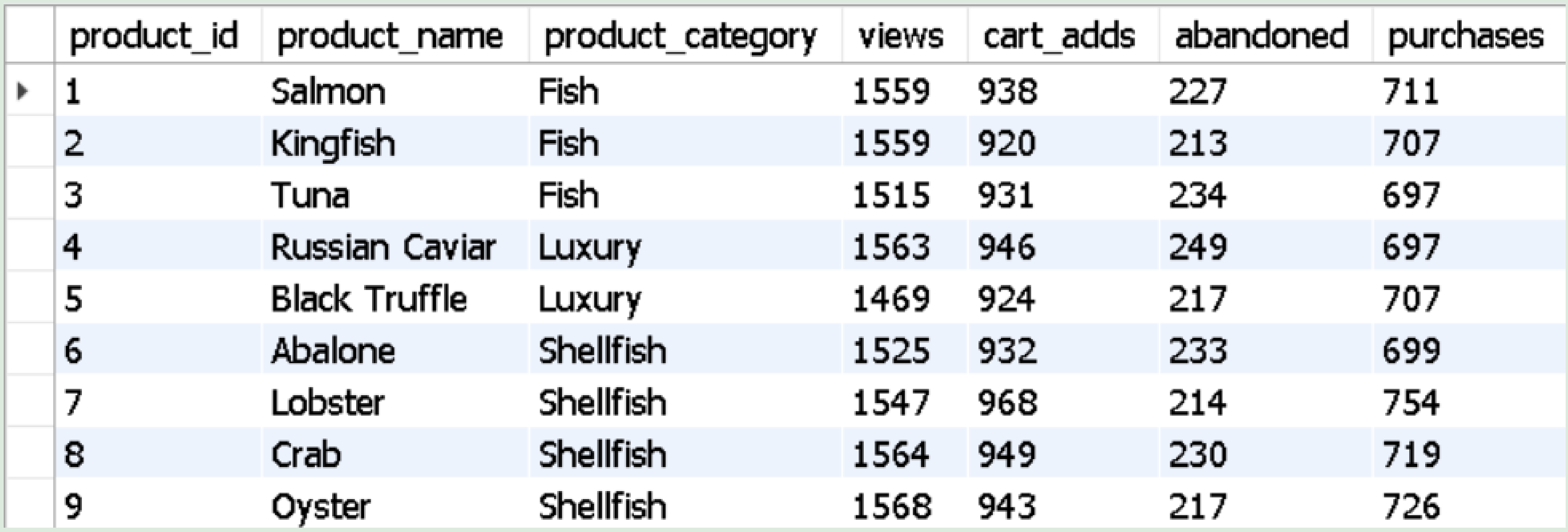
Continue Query of above slide:



```
product_info AS (
  SELECT
    product_id, -- Include product_id column
    product_name,
    product_category,
    SUM(page_view) AS views,
    SUM(cart_add) AS cart_adds,
    SUM(CASE WHEN cart_add = 1 AND purchase = 0 THEN 1 ELSE 0 END) AS abandoned,
    SUM(CASE WHEN cart_add = 1 AND purchase = 1 THEN 1 ELSE 0 END) AS purchases
  FROM combined_table
  GROUP BY product_id, product_name, product_category
)

SELECT *
FROM product_info
ORDER BY product_id;
```

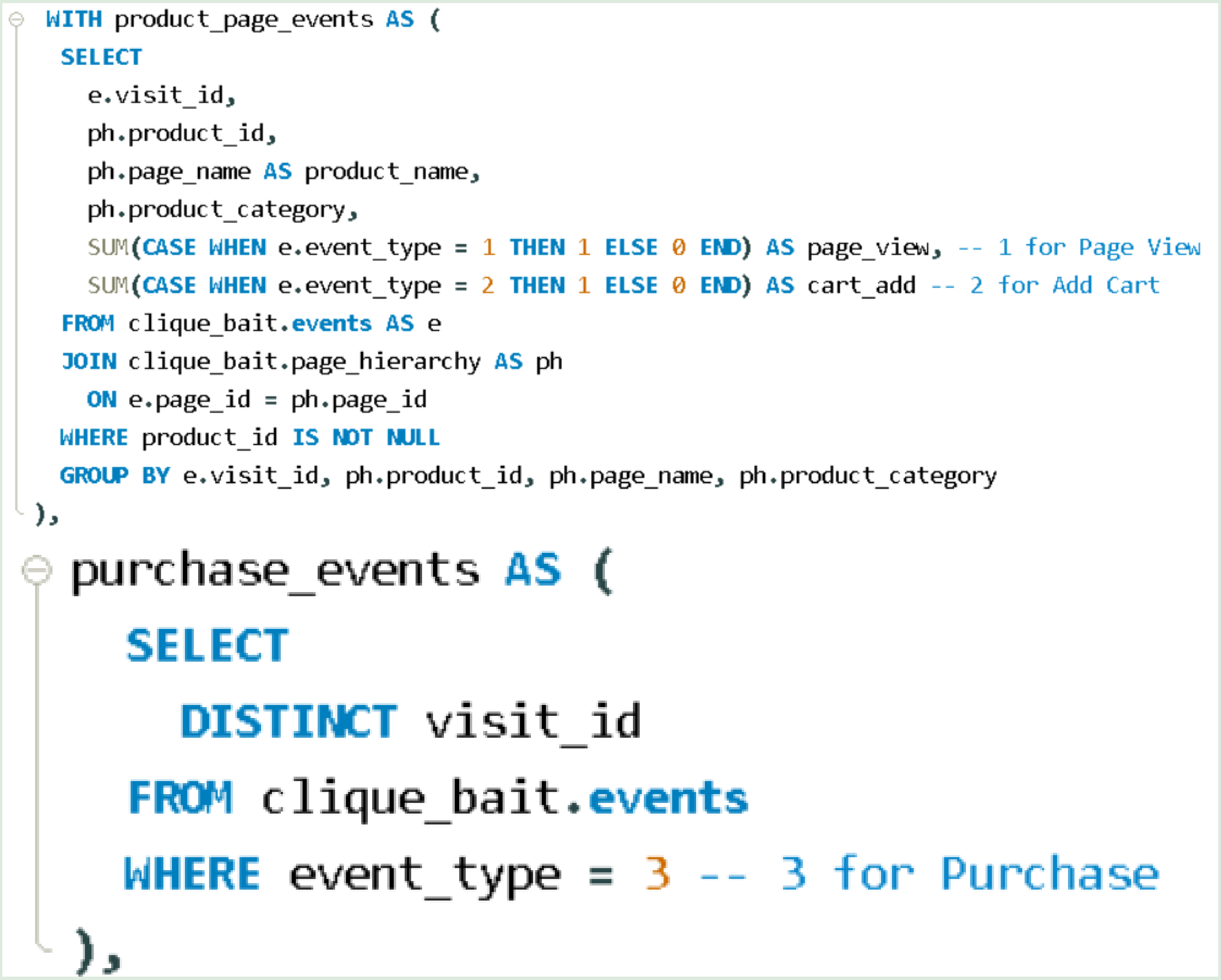
Output:



product_id	product_name	product_category	views	cart_adds	abandoned	purchases
1	Salmon	Fish	1559	938	227	711
2	Kingfish	Fish	1559	920	213	707
3	Tuna	Fish	1515	931	234	697
4	Russian Caviar	Luxury	1563	946	249	697
5	Black Truffle	Luxury	1469	924	217	707
6	Abalone	Shellfish	1525	932	233	699
7	Lobster	Shellfish	1547	968	214	754
8	Crab	Shellfish	1564	949	230	719
9	Oyster	Shellfish	1568	943	217	726

Additionally, create another table which further aggregates the data for the above points but this time for each product category instead of individual products.

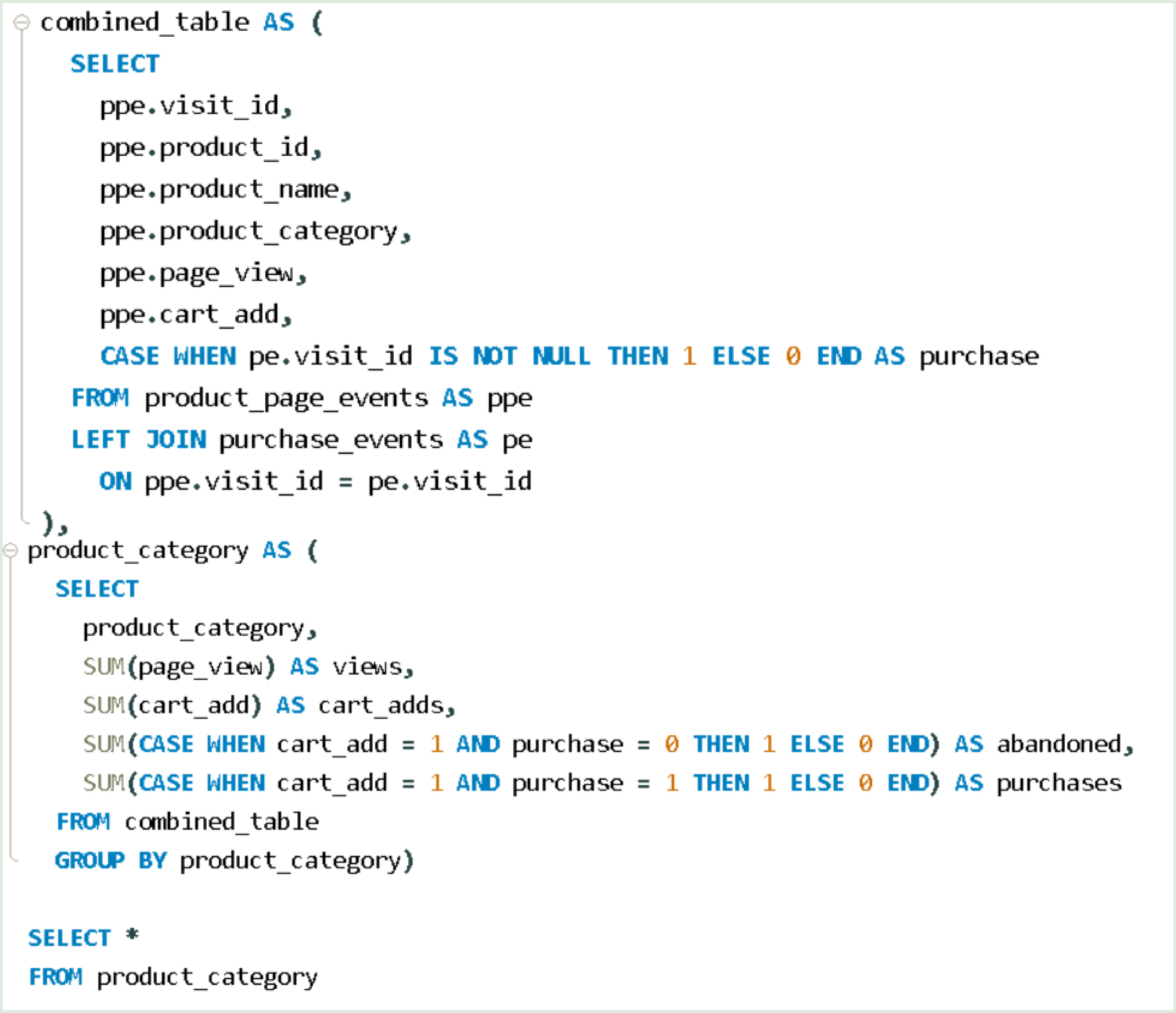
Query:



```sql
WITH product_page_events AS (
  SELECT
    e.visit_id,
    ph.product_id,
    ph.page_name AS product_name,
    ph.product_category,
    SUM(CASE WHEN e.event_type = 1 THEN 1 ELSE 0 END) AS page_view, -- 1 for Page View
    SUM(CASE WHEN e.event_type = 2 THEN 1 ELSE 0 END) AS cart_add -- 2 for Add Cart
  FROM clique_bait.events AS e
  JOIN clique_bait.page_hierarchy AS ph
    ON e.page_id = ph.page_id
  WHERE product_id IS NOT NULL
  GROUP BY e.visit_id, ph.product_id, ph.page_name, ph.product_category
),
purchase_events AS (
  SELECT
    DISTINCT visit_id
  FROM clique_bait.events
  WHERE event_type = 3 -- 3 for Purchase
),
```

Kapil Verma

Continue Query of above slide:

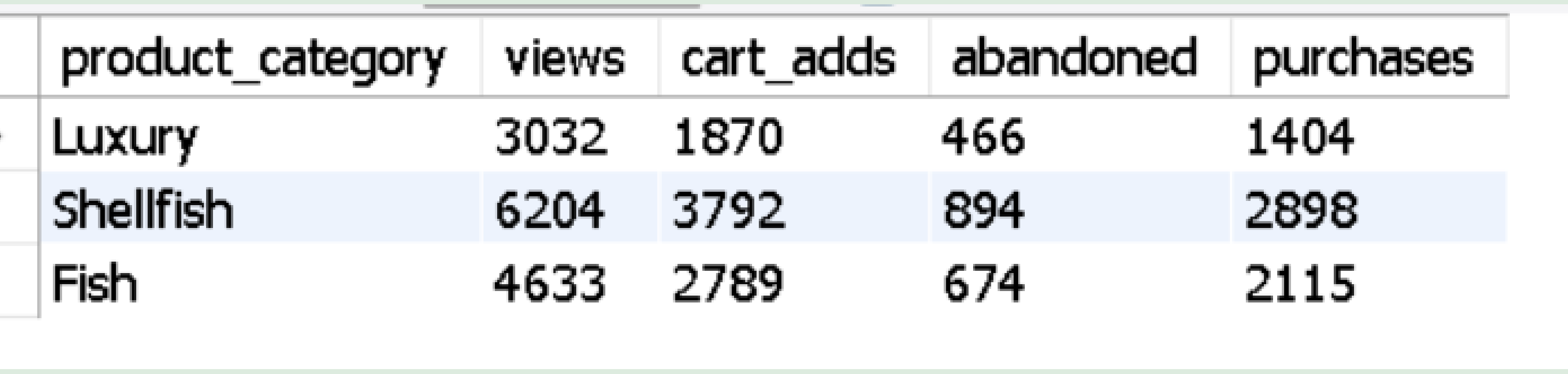


```
combined_table AS (
  SELECT
    ppe.visit_id,
    ppe.product_id,
    ppe.product_name,
    ppe.product_category,
    ppe.page_view,
    ppe.cart_add,
    CASE WHEN pe.visit_id IS NOT NULL THEN 1 ELSE 0 END AS purchase
  FROM product_page_events AS ppe
  LEFT JOIN purchase_events AS pe
    ON ppe.visit_id = pe.visit_id
),
product_category AS (
  SELECT
    product_category,
    SUM(page_view) AS views,
    SUM(cart_add) AS cart_adds,
    SUM(CASE WHEN cart_add = 1 AND purchase = 0 THEN 1 ELSE 0 END) AS abandoned,
    SUM(CASE WHEN cart_add = 1 AND purchase = 1 THEN 1 ELSE 0 END) AS purchases
  FROM combined_table
  GROUP BY product_category)

SELECT *
FROM product_category
```

Kapil Verma

Output:



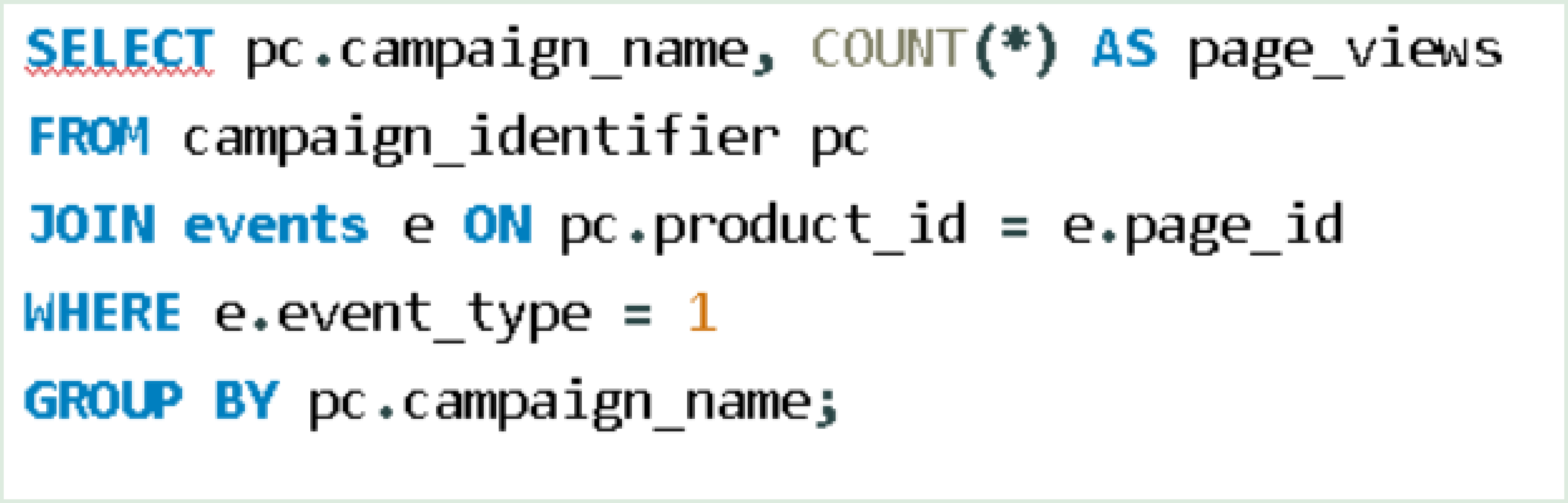
product_category	views	cart_adds	abandoned	purchases
Luxury	3032	1870	466	1404
Shellfish	6204	3792	894	2898
Fish	4633	2789	674	2115

Kapil Verma

Part C. Campaigns Analysis

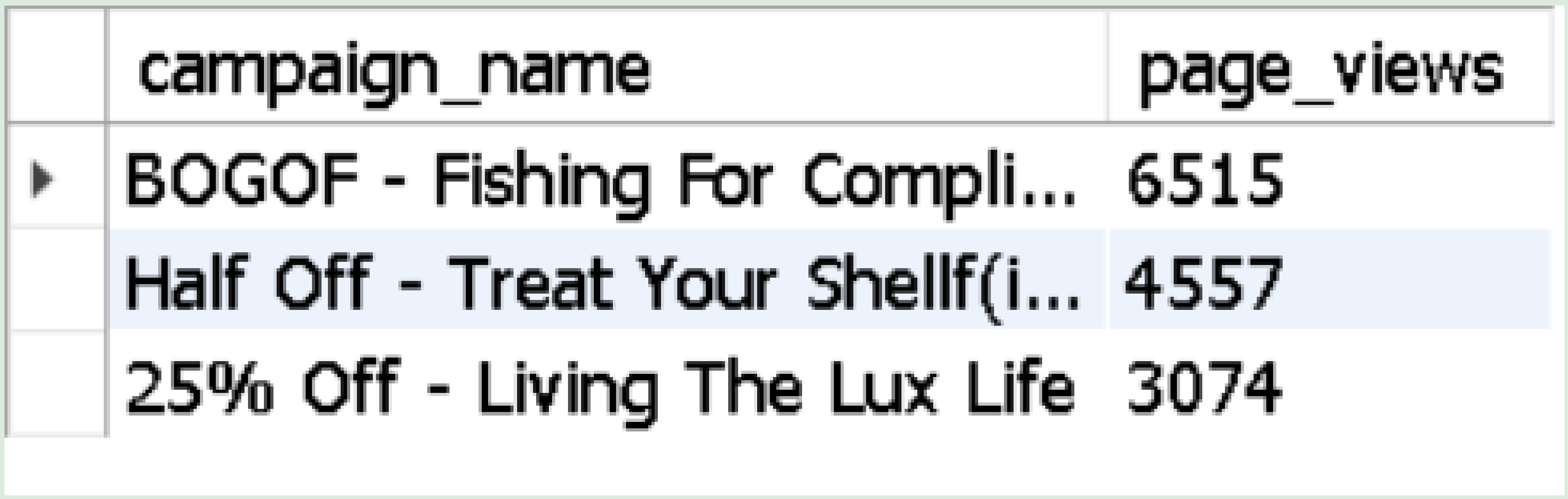
Q1. As the entrusted Data Analyst of Clique Bait, your task is to identify the number of page views each campaign generated.

Query:



```sql
SELECT pc.campaign_name, COUNT(*) AS page_views
FROM campaign_identifier pc
JOIN events e ON pc.product_id = e.page_id
WHERE e.event_type = 1
GROUP BY pc.campaign_name;
```

Output:



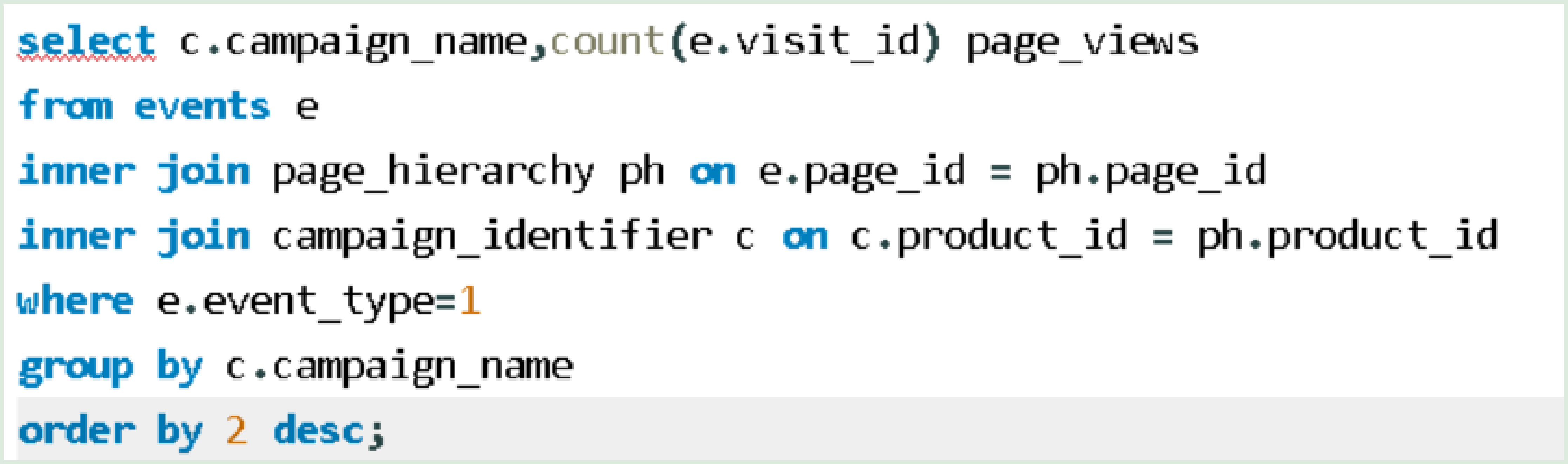
campaign_name	page_views
BOGOF - Fishing For Compli...	6515
Half Off - Treat Your Shellf(i...	4557
25% Off - Living The Lux Life	3074

Kapil Verma

Part C. Campaigns Analysis

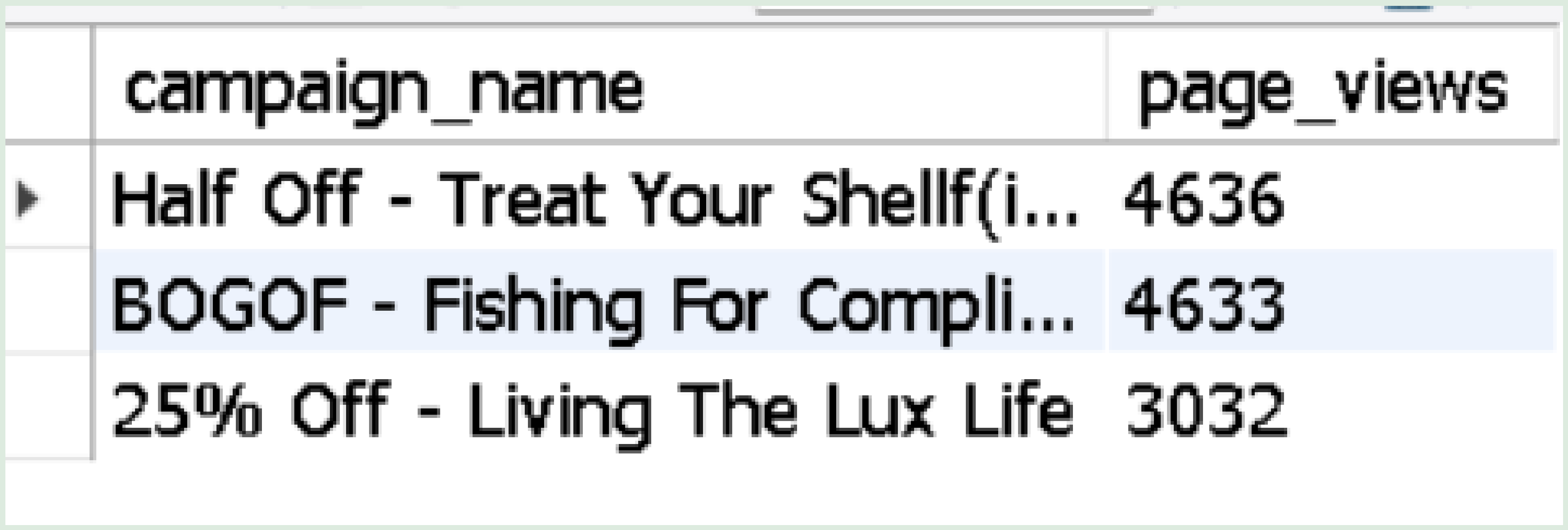
Q2. Your task is to identify the number of page views each campaign generated.

Query:



```
select c.campaign_name,count(e.visit_id) page_views
from events e
inner join page_hierarchy ph on e.page_id = ph.page_id
inner join campaign_identifier c on c.product_id = ph.product_id
where e.event_type=1
group by c.campaign_name
order by 2 desc;
```

Output:



campaign_name	page_views
Half Off - Treat Your Shellf(i...	4636
BOGOF - Fishing For Compli...	4633
25% Off - Living The Lux Life	3032

Kapil Verma

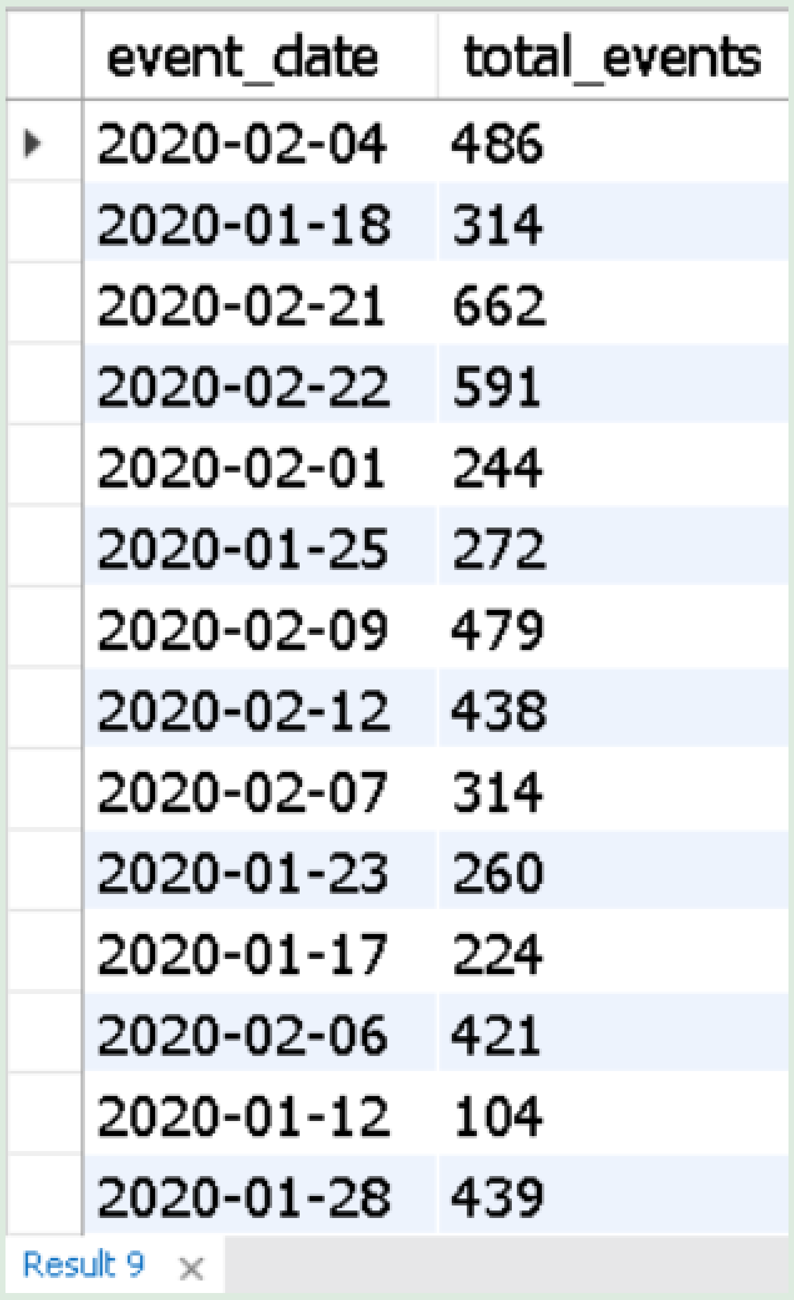
Part C.Temporal Analysis

Q1.Calculate the daily count of events recorded in the "events" table to understand user interaction distribution over different days.

Query:

```
SELECT DATE(event_time) AS event_date, COUNT(*) AS total_events
FROM events
GROUP BY event_date;
```

Output:



event_date	total_events
2020-02-04	486
2020-01-18	314
2020-02-21	662
2020-02-22	591
2020-02-01	244
2020-01-25	272
2020-02-09	479
2020-02-12	438
2020-02-07	314
2020-01-23	260
2020-01-17	224
2020-02-06	421
2020-01-12	104
2020-01-28	439