

Data Warehouse Modeling with DBT

Detail Assignment

Latar Belakang:

Kamu telah diberikan dataset dari e-commerce yang menyediakan wawasan tentang penjualan di Amazon. Dataset ini mencakup informasi seperti SKU Code, Design Number, Stock, Category, Size, dan Color yang digunakan untuk mengoptimalkan profitabilitas produk. Informasi lebih lanjut mengenai dataset bisa kamu temukan di:

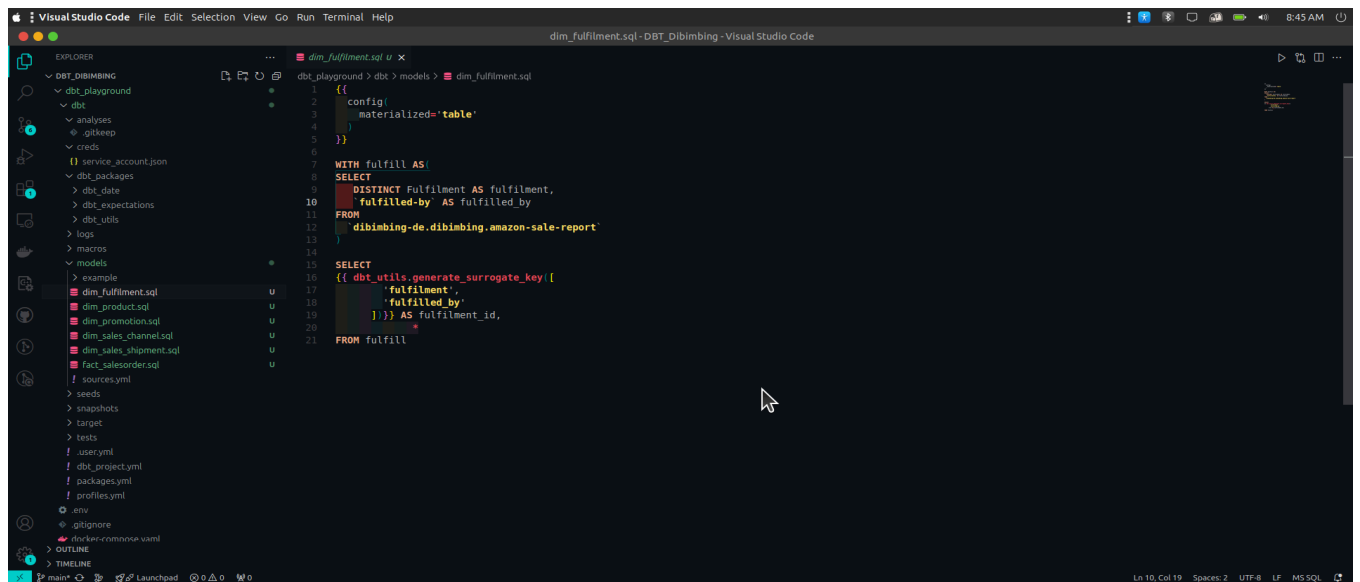
[E-Commerce Sales Dataset.](#)

Bangun Tabel Dimensi:

Buat beberapa tabel dimensi dengan menggunakan DBT (Data Build Tool):

- dim_fulfilment (Hands-on)

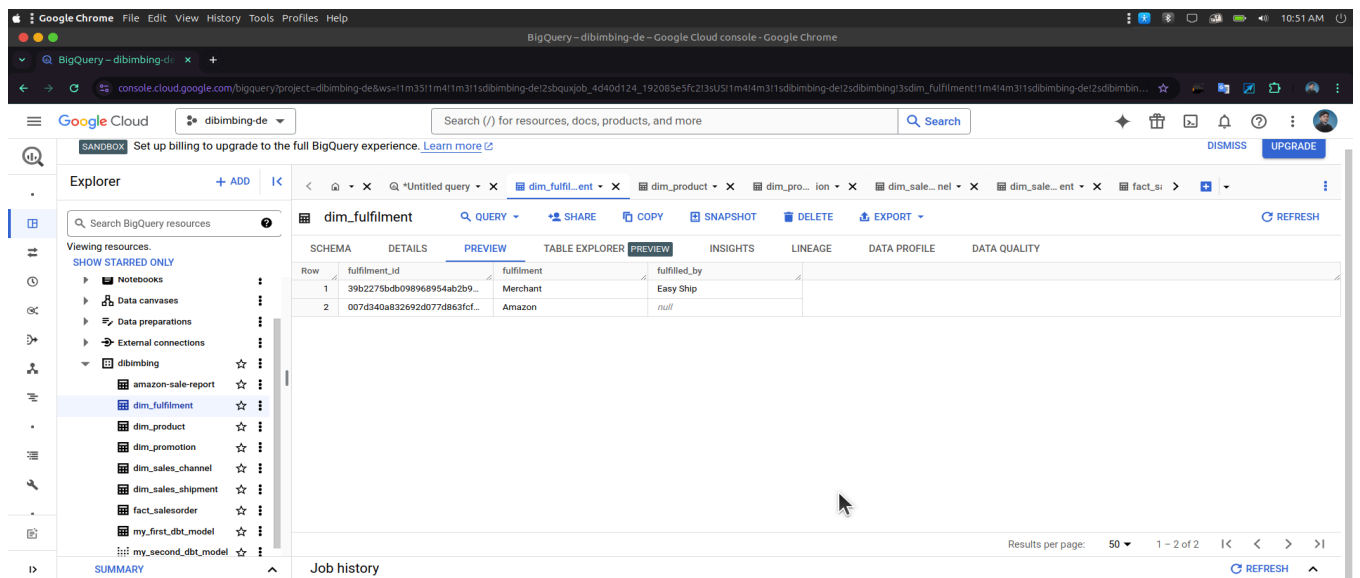
local



The screenshot shows the Visual Studio Code interface with a DBT project. The Explorer panel on the left shows the project structure, including the 'models' directory. The main editor displays the 'dim_fulfilment.sql' file, which contains the following SQL code:

```
1 {{
2   config(
3     materialized='table'
4   )
5 }}
6
7 WITH fulfill AS
8 SELECT
9   DISTINCT Fulfillment AS fulfillment,
10   fulfilled-by AS fulfilled_by
11 FROM
12   dibimbing-de.dibimbing.amazon-sale-report
13
14
15 SELECT
16   ({{ dbt_utils.generate_surrogate_key([
17     'fulfillment',
18     'fulfilled_by'
19   ]) }} AS fulfillment_id,
20   ) AS fulfillment_id,
21 FROM fulfill
```

Cloud

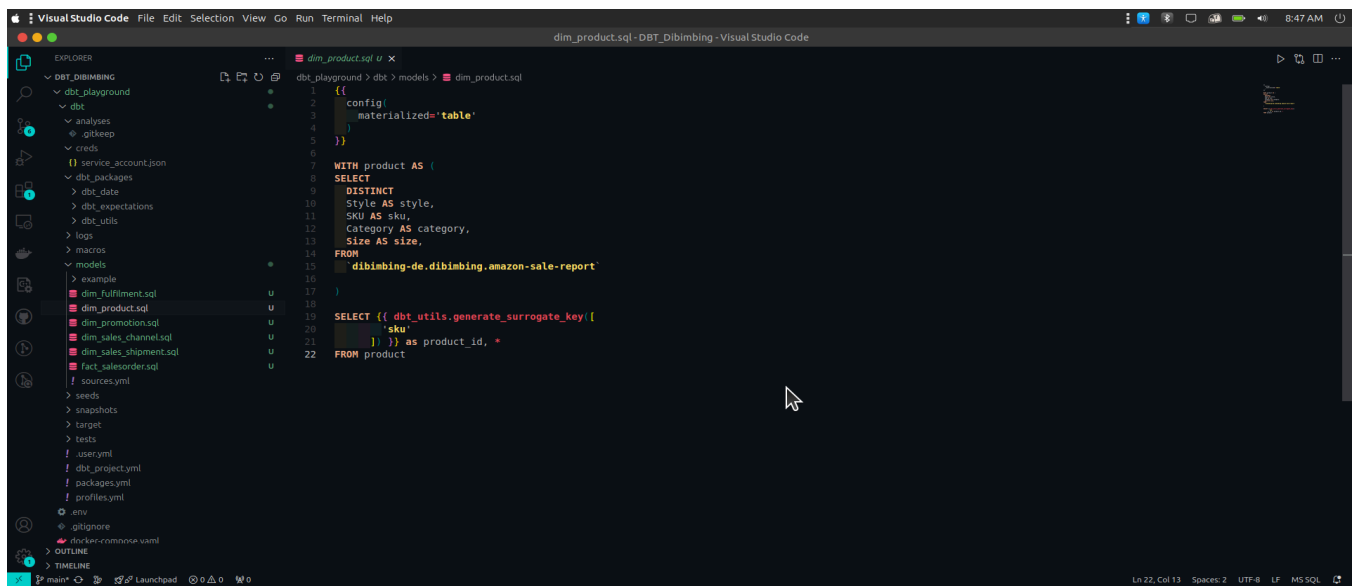


The screenshot shows the Google Cloud BigQuery console. The 'Explorer' panel on the left lists the project's resources, including the 'dim_fulfilment' table. The main panel displays the 'dim_fulfilment' table in the 'PREVIEW' tab, showing the following data:

| Row | fulfillment_id | fulfillment | fulfilled_by |
|-----|----------------------------|-------------|--------------|
| 1 | 39b2275bdb09896954ab2b9... | Merchant | Easy Ship |
| 2 | 007d340a832692d077d863f... | Amazon | null |

- dim_product (Hands-on)

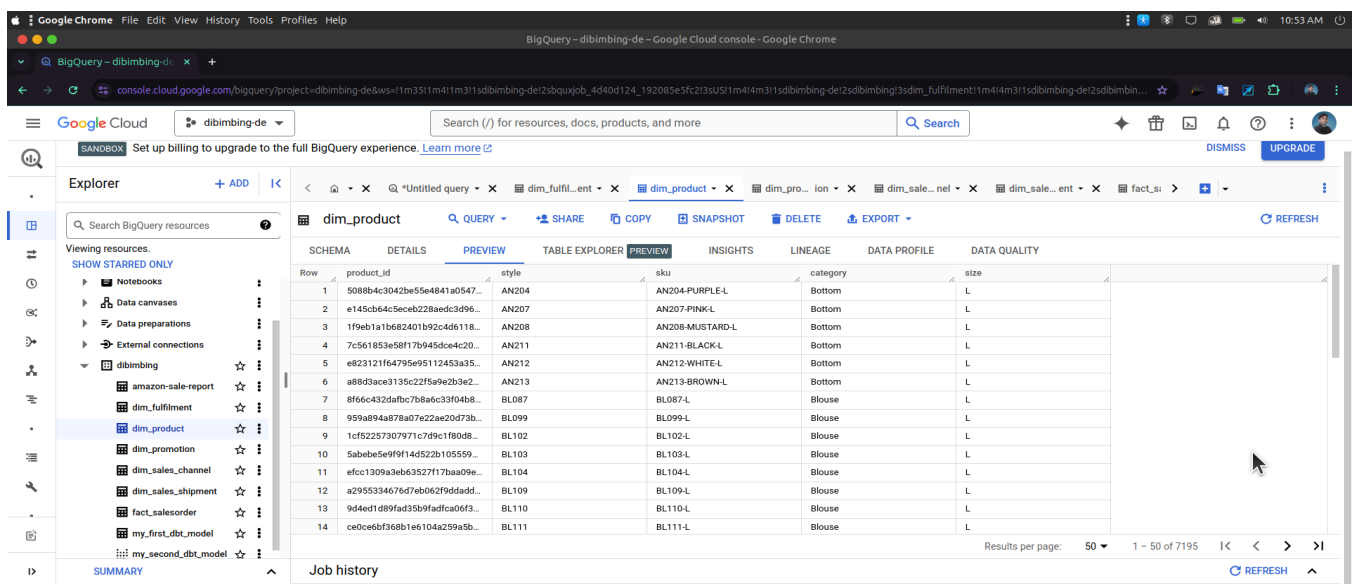
local



The screenshot shows the Visual Studio Code editor with a DBT project open. The file explorer on the left shows the project structure, including the 'models' directory. The main editor displays the 'dim_product.sql' file, which contains a SQL query for creating a dimension table. The query uses a CTE named 'product' to select distinct values for style, sku, category, and size from the 'amazon-sale-report' table. It then uses a 'generate_surrogate_key' function to create a unique product_id for each row.

```
1 {{
2   config(
3     materialized='table'
4   )
5 }}
6
7 WITH product AS (
8   SELECT
9     DISTINCT
10    style AS style,
11    sku AS sku,
12    category AS category,
13    size AS size,
14  FROM
15    dbt_bimbi-de.dbt_bimbi.amazon-sale-report
16 )
17
18 SELECT {{ dbt_utils.generate_surrogate_key([
19   'sku'
20 ]) }} as product_id, *
21 FROM product
22
```

Cloud

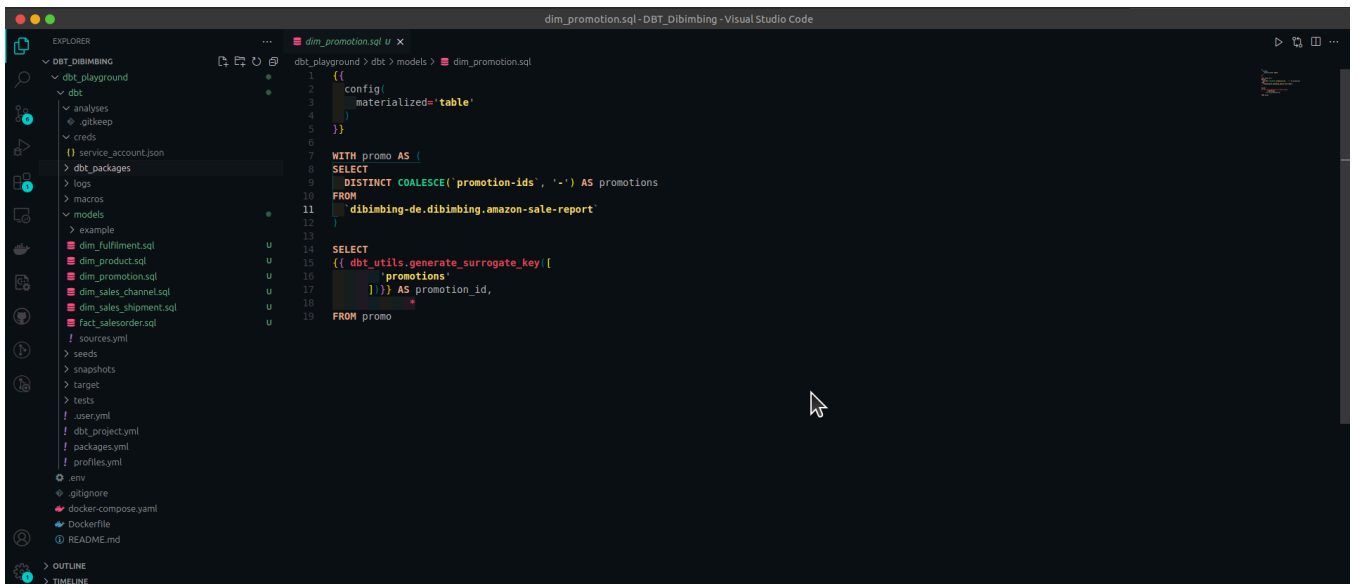


The screenshot shows the Google Cloud BigQuery console. The left sidebar displays the project hierarchy, with 'dim_product' selected under the 'models' directory. The main panel shows the 'dim_product' table in the 'SCHEMA' tab. The table has 14 rows and 7 columns: product_id, style, sku, category, size, and two unnamed columns. The data is as follows:

| Row | product_id | style | sku | category | size | | |
|-----|-------------------------------|-------|-----------------|----------|------|--|--|
| 1 | 5088b4c3042be55e4841a0547... | AN204 | AN204-PURPLE-L | Bottom | L | | |
| 2 | e145cb64c5e6eb228aedc3d96... | AN207 | AN207-PINK-L | Bottom | L | | |
| 3 | 199eb1a1b682401b92c4d6118... | AN208 | AN208-MUSTARD-L | Bottom | L | | |
| 4 | 7c561853e58f17b945dce4c20... | AN211 | AN211-BLACK-L | Bottom | L | | |
| 5 | e823121f64795e95112453a35... | AN212 | AN212-WHITE-L | Bottom | L | | |
| 6 | a88d3ace3135c22f5a9e2b3e2... | AN213 | AN213-BROWN-L | Bottom | L | | |
| 7 | 8f6ec432dafbc7b8a6c3f04b8... | BL087 | BL087-L | Blouse | L | | |
| 8 | 959a894b878a07e22ae20d73b... | BL099 | BL099-L | Blouse | L | | |
| 9 | 1cf52257307971c7d9c1f80d8... | BL102 | BL102-L | Blouse | L | | |
| 10 | 5abeb5e9f9f14d522b105559... | BL103 | BL103-L | Blouse | L | | |
| 11 | efcc1309a3eb63527f17baa09e... | BL104 | BL104-L | Blouse | L | | |
| 12 | a2955334676d7eb062f9ddadd... | BL109 | BL109-L | Blouse | L | | |
| 13 | 9d4ed1d89fad35b9fadfca06f3... | BL110 | BL110-L | Blouse | L | | |
| 14 | ee0ce6bf368b1e6104a259a5b... | BL111 | BL111-L | Blouse | L | | |

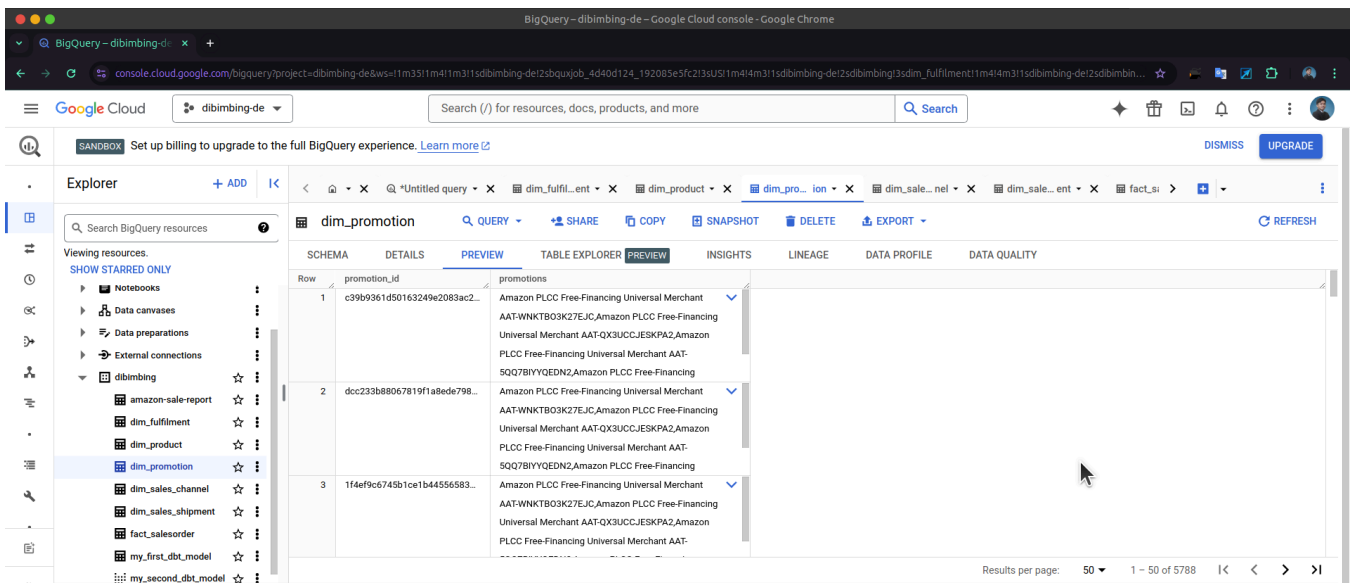
- dim_promotion

local



```
1 {{
2   config(
3     materialized='table'
4   )
5 }}
6
7 WITH promo AS (
8   SELECT
9     DISTINCT COALESCE('promotion-ids', '-') AS promotions
10  FROM
11    dibimbing-de.dibimbing.amazon-sale-report
12 )
13
14 SELECT
15   (( dbt_utils.generate_surrogate_key([
16     'promotions'
17   ])) AS promotion_id,
18   *
19  FROM promo
```

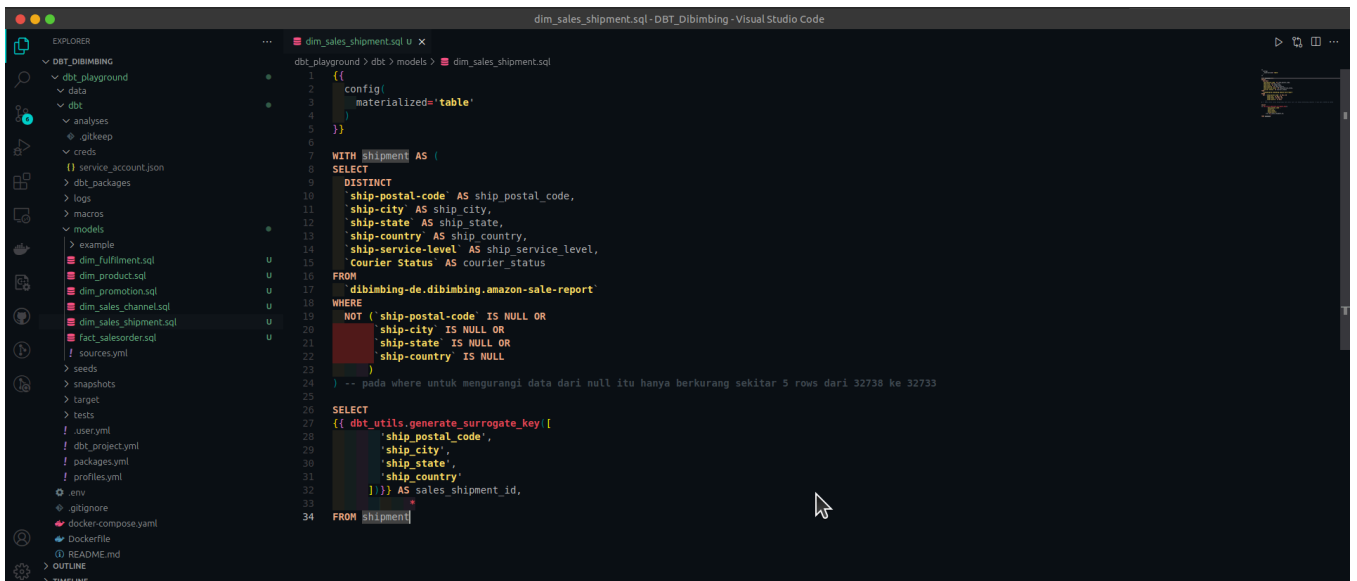
Cloud



| Row | promotion_id | promotions |
|-----|------------------------------|---|
| 1 | c99b9361d50163249e2083ac2... | Amazon PLCC Free-Financing Universal Merchant AAT-WNKTBO3K27EJC Amazon PLCC Free-Financing Universal Merchant AAT-QX3UCCJESKPA2, Amazon PLCC Free-Financing Universal Merchant AAT- 5QO7BIYYGEDN2, Amazon PLCC Free-Financing |
| 2 | dc233b88067819f1a8ede798... | Amazon PLCC Free-Financing Universal Merchant AAT-WNKTBO3K27EJC Amazon PLCC Free-Financing Universal Merchant AAT-QX3UCCJESKPA2, Amazon PLCC Free-Financing Universal Merchant AAT- 5QO7BIYYGEDN2, Amazon PLCC Free-Financing |
| 3 | 1f4ef9c6745b1ce1b44556583... | Amazon PLCC Free-Financing Universal Merchant AAT-WNKTBO3K27EJC Amazon PLCC Free-Financing Universal Merchant AAT-QX3UCCJESKPA2, Amazon PLCC Free-Financing Universal Merchant AAT- |

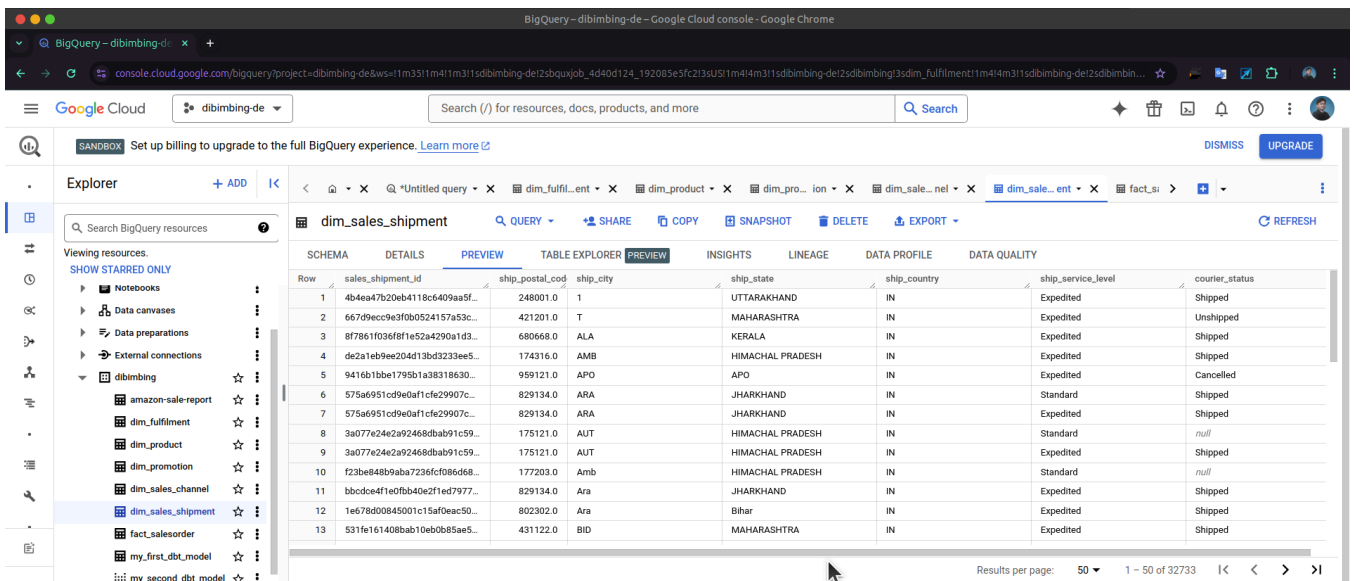
- dim_sales_shipment

local



```
1 {{
2   config(
3     materialized='table'
4   )
5 }}
6
7 WITH shipment AS (
8   SELECT
9     DISTINCT
10    'ship-postal-code' AS ship_postal_code,
11    'ship-city' AS ship_city,
12    'ship-state' AS ship_state,
13    'ship-country' AS ship_country,
14    'ship-service-level' AS ship_service_level,
15    'Courier Status' AS courier_status
16  FROM
17    dibimbing-de.dibimbing.amazon-sale-report
18  WHERE
19    NOT ('ship-postal-code' IS NULL OR
20         'ship-city' IS NULL OR
21         'ship-state' IS NULL OR
22         'ship-country' IS NULL
23        )
24  ) -- pada where untuk mengurangi data dari null itu hanya berkurang sekitar 5 rows dari 32738 ke 32733
25
26 SELECT
27   {{ dbt_utils.generate_surrogate_key([
28     'ship_postal_code',
29     'ship_city',
30     'ship_state',
31     'ship_country'
32   ]) }} AS sales_shipment_id,
33
34 FROM shipment
```

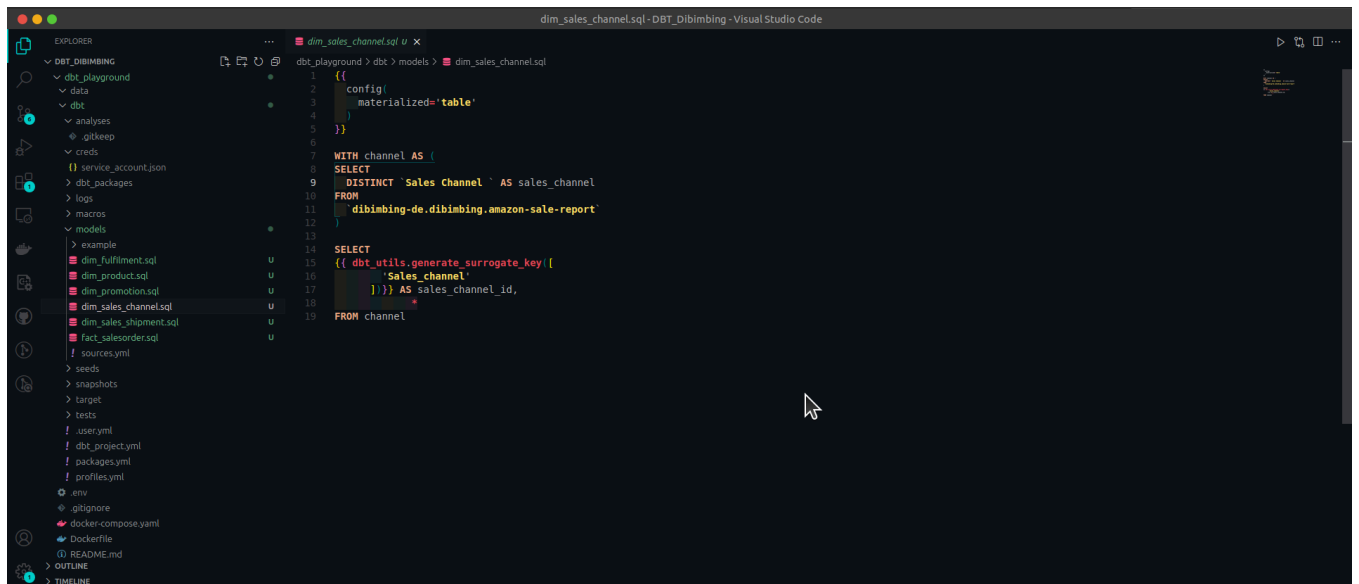
Cloud



| Row | sales_shipment_id | ship_postal_code | ship_city | ship_state | ship_country | ship_service_level | courier_status |
|-----|-------------------------------|------------------|-----------|------------------|--------------|--------------------|----------------|
| 1 | 4b4ea47b20eb4118c6409aa5f... | 248001.0 | 1 | UTTARAKHAND | IN | Expedited | Shipped |
| 2 | 667d9ecc9e3f0b0524157a53c... | 421201.0 | T | MAHARASHTRA | IN | Expedited | Unshipped |
| 3 | 8f7861f036f81e52a4290a1d3... | 680668.0 | ALA | KERALA | IN | Expedited | Shipped |
| 4 | de2a1eb9ee204d13bd3232ee5... | 174316.0 | AMB | HIMACHAL PRADESH | IN | Expedited | Shipped |
| 5 | 9416b1b5b1795b1a38318630... | 999121.0 | APO | APO | IN | Expedited | Cancelled |
| 6 | 575ae951cd9e0af1cfe29907c... | 829134.0 | ARA | JHARKHAND | IN | Standard | Shipped |
| 7 | 575ae951cd9e0af1cfe29907c... | 829134.0 | ARA | JHARKHAND | IN | Expedited | Shipped |
| 8 | 3a077e24e2a92468dbab91c59... | 175121.0 | AUT | HIMACHAL PRADESH | IN | Standard | null |
| 9 | 3a077e24e2a92468dbab91c59... | 175121.0 | AUT | HIMACHAL PRADESH | IN | Expedited | Shipped |
| 10 | f23be848b9aba7236fcf086d68... | 177203.0 | Amb | HIMACHAL PRADESH | IN | Standard | null |
| 11 | bbdc0e4f1e0fb40e2f1ed7977... | 829134.0 | Ara | JHARKHAND | IN | Expedited | Shipped |
| 12 | 1e678d00845001c15af0eac50... | 802302.0 | Ara | Bihar | IN | Expedited | Shipped |
| 13 | 531fe161408bab10eb0b85ae5... | 431122.0 | BID | MAHARASHTRA | IN | Expedited | Shipped |

- Dim_sales_channel

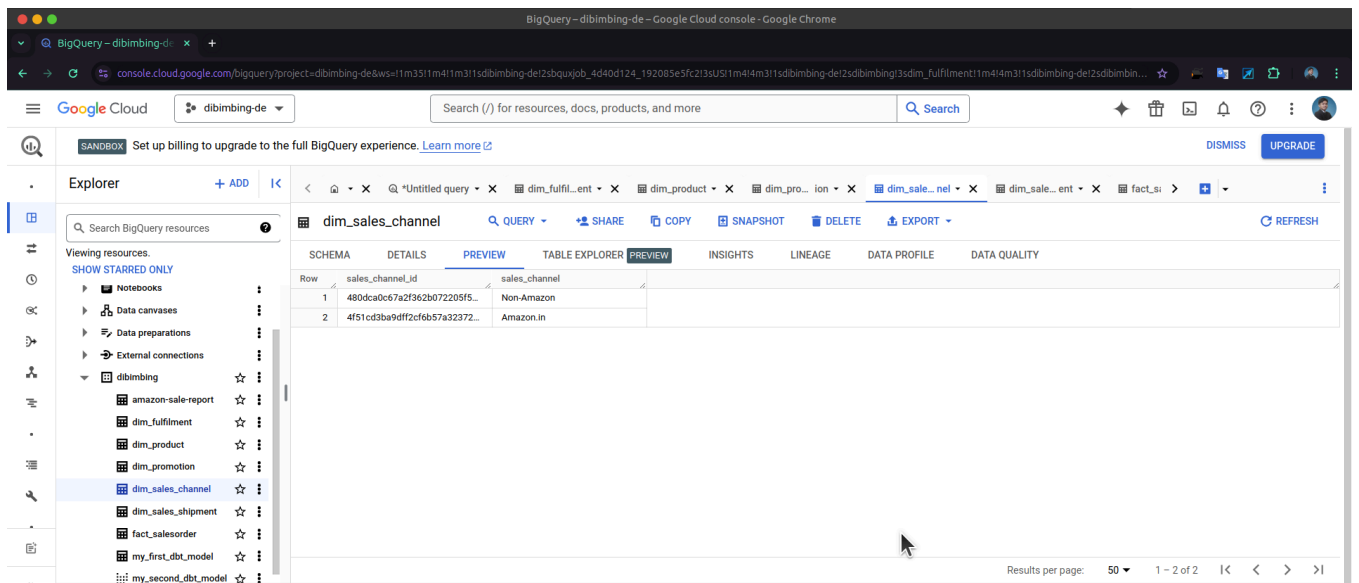
local



The screenshot shows the Visual Studio Code editor with the file `dim_sales_channel.sql` open. The file is located in the `dbt_playground > dbt > models > dim_sales_channel` directory. The SQL code defines a materialized view for the `dim_sales_channel` table, which is populated from the `amazon-sale-report` table in the `dibimbing-de` database. The code includes a `WITH` clause for the `channel` table and a `SELECT` statement that uses `dbt_utils.generate_surrogate_key` to create a surrogate key for the `sales_channel` column.

```
1 {{
2   config(
3     materialized='table'
4   )
5 }}
6
7 WITH channel AS (
8   SELECT
9     DISTINCT 'Sales Channel' AS sales_channel
10    FROM
11      dibimbing-de.dibimbing.amazon-sale-report
12 )
13
14 SELECT
15   ({{ dbt_utils.generate_surrogate_key([
16     'Sales_channel'
17   ])}} AS sales_channel_id,
18   *
19  FROM channel
```

Cloud



The screenshot shows the Google Cloud BigQuery console for the project `dibimbing-de`. The `dim_sales_channel` table is selected, and the `TABLE EXPLORER` tab is active. The table has two columns: `sales_channel_id` and `sales_channel`. The data is as follows:

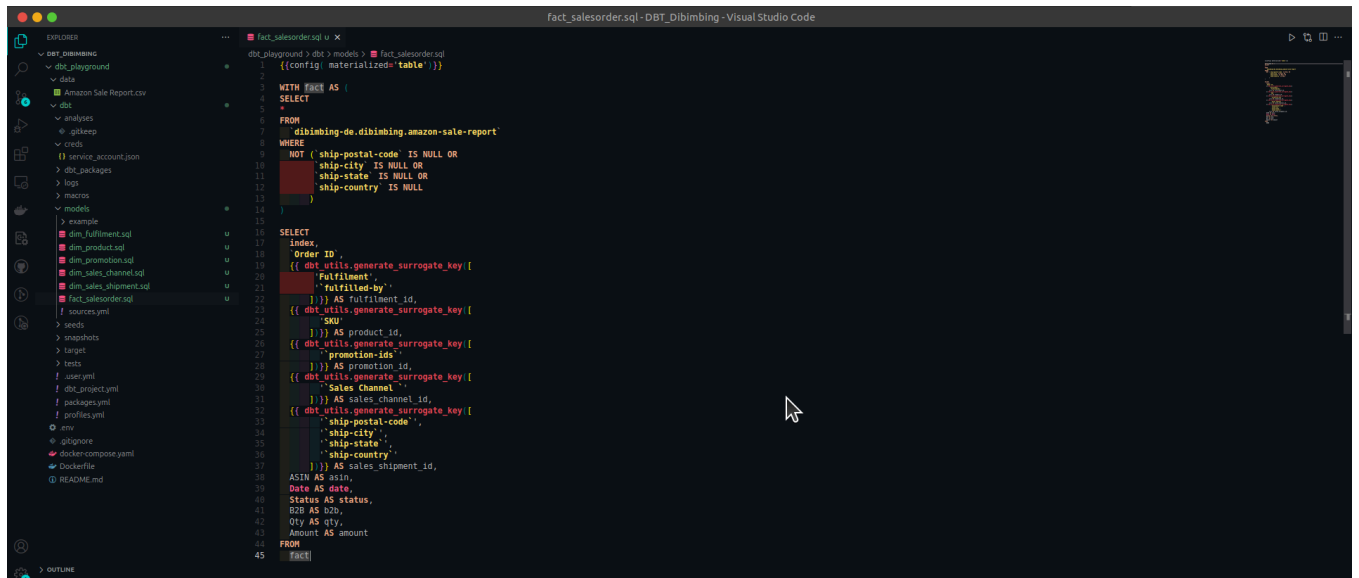
| Row | sales_channel_id | sales_channel |
|-----|------------------------------|---------------|
| 1 | 480dca0c67a2f362b072205f5... | Non-Amazon |
| 2 | 4f51cd3ba9df2cf6b57a32372... | Amazon.in |

The console also shows a list of other tables in the `dibimbing` dataset, including `amazon-sale-report`, `dim_fulfillment`, `dim_product`, `dim_promotion`, `dim_sales_channel`, `dim_sales_shipment`, `fact_salesorder`, `my_first_dbt_model`, and `my_second_dbt_model`.

Bangun Tabel Fakta:

- Buat tabel fakta utama yang disebut fact_salesorder dengan menggunakan DBT.

local



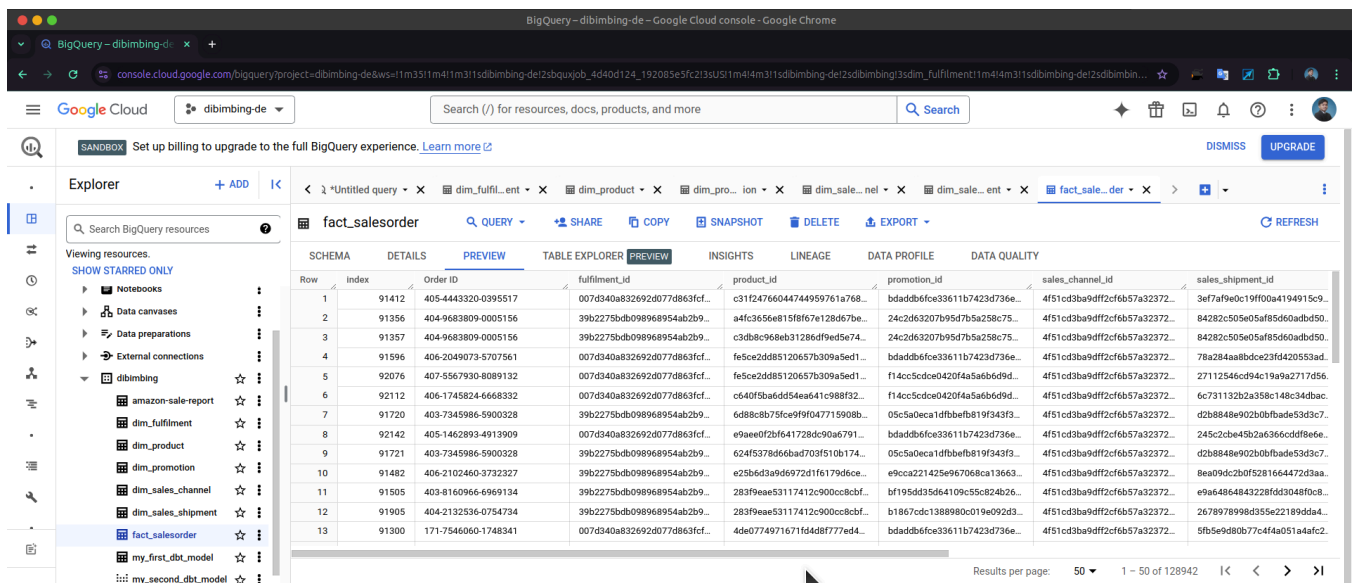
```
fact_salesorder.sql - DBT_Dibimbing - Visual Studio Code

dbt_playground > dbt > models > fact_salesorder.sql
{{config(materialized='table')}}

WITH amazon AS (
  SELECT
    *
  FROM
    dibimbing-de.dibimbing.amazon-sale-report
  WHERE
    NOT ('ship-postal-code' IS NULL OR
        ship-city IS NULL OR
        ship-state IS NULL OR
        ship-country IS NULL)
)

SELECT
  Index
  Order ID,
  {{ dbt_utils.generate_surrogate_key([
    'fulfillment_id',
    'fulfilled-by',
    ]) }} AS fulfillment_id,
  {{ dbt_utils.generate_surrogate_key([
    'SKU'
    ]) }} AS product_id,
  {{ dbt_utils.generate_surrogate_key([
    'promotion-id',
    ]) }} AS promotion_id,
  {{ dbt_utils.generate_surrogate_key([
    'Sales Channel',
    ]) }} AS sales_channel_id,
  {{ dbt_utils.generate_surrogate_key([
    'ship-postal-code',
    'ship-city',
    'ship-state',
    'ship-country'
    ]) }} AS sales_shipment_id,
  ASIN AS asin,
  Date AS date,
  Status AS status,
  Qty AS qty,
  Amount AS amount
FROM
  amazon
```

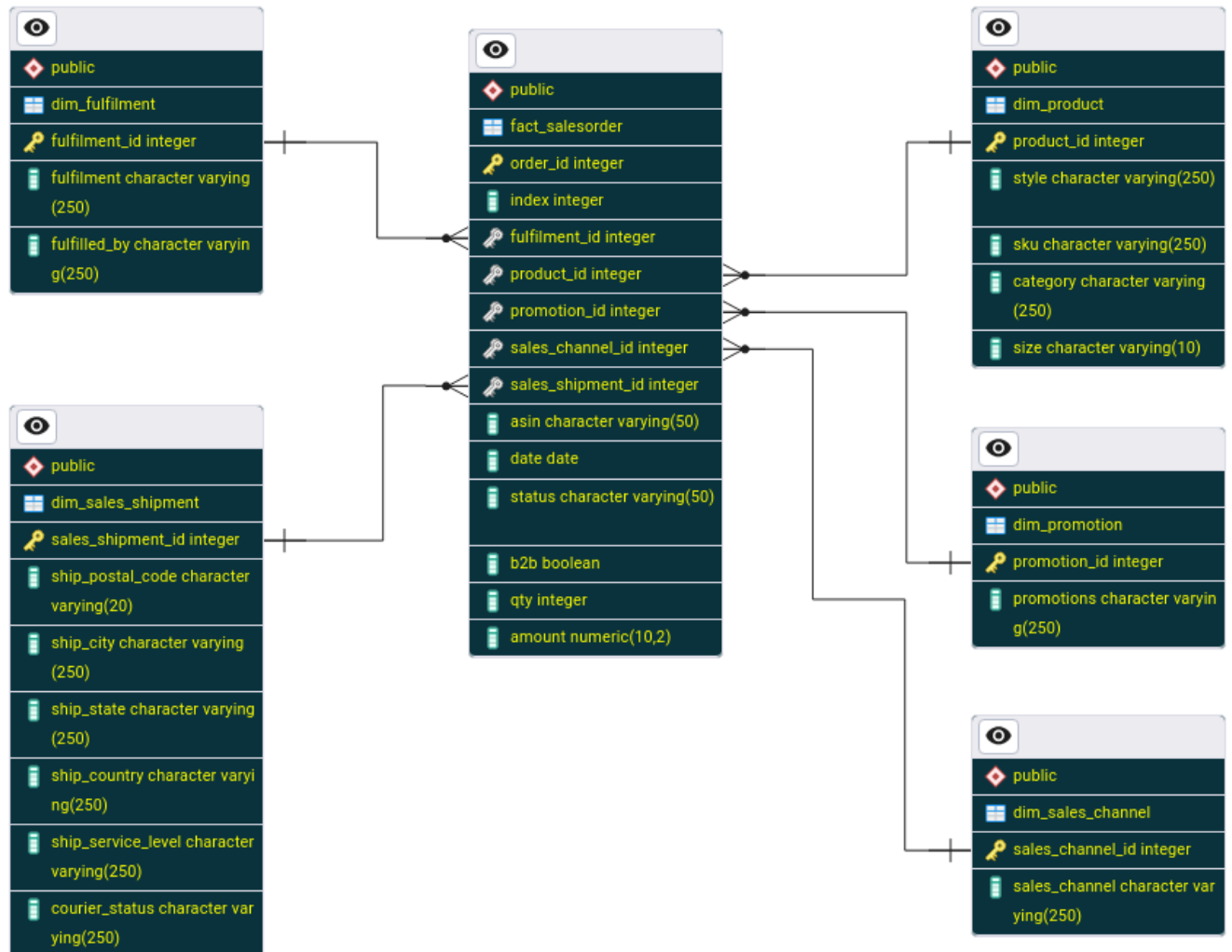
Cloud



| Row | Index | Order ID | fulfillment_id | product_id | promotion_id | sales_channel_id | sales_shipment_id |
|-----|-------|---------------------|------------------------------|-------------------------------|-------------------------------|------------------------------|-------------------------------|
| 1 | 91412 | 405-4443320-0395517 | 007d340a832692d077d863fcf... | c31f24766044744959761a768... | bdaddb6fce33611b7423d736e... | 4f51cd3ba9df2cf6b57a32372... | 3ef7a79e0c19ff00a4194915c9... |
| 2 | 91356 | 404-9683809-0005156 | 39b2275bdc098968954ab2b9... | a4fc3656e815f8f67e128d67be... | 24c2d63207b95d7b5a258c75... | 4f51cd3ba9df2cf6b57a32372... | 84282c505e05af8560adb50... |
| 3 | 91357 | 404-9683809-0005156 | 39b2275bdc098968954ab2b9... | c3db8c968b31286df9ed5e74... | 24c2d63207b95d7b5a258c75... | 4f51cd3ba9df2cf6b57a32372... | 84282c505e05af8560adb50... |
| 4 | 91596 | 406-2049073-5707561 | 007d340a832692d077d863fcf... | fe5ce2dd85120657b309a5ed1... | bdaddb6fce33611b7423d736e... | 4f51cd3ba9df2cf6b57a32372... | 78a284aa8bdce23fd420553ad... |
| 5 | 92076 | 407-5567930-8089132 | 007d340a832692d077d863fcf... | fe5ce2dd85120657b309a5ed1... | f14cc5cdce0420f4a5a6b6d9d... | 4f51cd3ba9df2cf6b57a32372... | 27112546cd94c19a9a2717d56... |
| 6 | 92112 | 406-1745824-6668332 | 007d340a832692d077d863fcf... | c640f5ba6dd54eae41c988f32... | f14cc5cdce0420f4a5a6b6d9d... | 4f51cd3ba9df2cf6b57a32372... | 6c731132b2a358c148c34dbac... |
| 7 | 91720 | 403-7345986-5900328 | 39b2275bdc098968954ab2b9... | 6d88c8b75fce9f9f047715908b... | 05c5a0eca1dfbfbef819f343f3... | 4f51cd3ba9df2cf6b57a32372... | d2b8848e902b0fbade53d3c7... |
| 8 | 92142 | 405-1462893-4913909 | 007d340a832692d077d863fcf... | e9aee0f2bf614728dc90a6791... | bdaddb6fce33611b7423d736e... | 4f51cd3ba9df2cf6b57a32372... | 245c2cbe45b2a636dcdf8e6e... |
| 9 | 91721 | 403-7345986-5900328 | 39b2275bdc098968954ab2b9... | 624f5378d66bad703f510b174... | 05c5a0eca1dfbfbef819f343f3... | 4f51cd3ba9df2cf6b57a32372... | d2b8848e902b0fbade53d3c7... |
| 10 | 91482 | 406-2102460-3723227 | 39b2275bdc098968954ab2b9... | e25b6d3af9d6972d1f6179dce... | e9cca221425e967068ca13663... | 4f51cd3ba9df2cf6b57a32372... | 8ea09dc2b0f5281664472d3aa... |
| 11 | 91505 | 403-8160966-6969134 | 39b2275bdc098968954ab2b9... | 283f9eae53117412c900cc8cbf... | bf195dd35d64109c55c824b26... | 4f51cd3ba9df2cf6b57a32372... | e9a64864843228fdd3048f0c8... |
| 12 | 91905 | 404-2132536-0754734 | 39b2275bdc098968954ab2b9... | 283f9eae53117412c900cc8cbf... | b1867cdc1388900c019e092d3... | 4f51cd3ba9df2cf6b57a32372... | 2678978998d355e22189dda4... |
| 13 | 91300 | 171-7546060-1748341 | 007d340a832692d077d863fcf... | Ade0774971671f5d4d8f77ed4... | bdaddb6fce33611b7423d736e... | 4f51cd3ba9df2cf6b57a32372... | 5fb5e9d80b77c4fa051a4af4c2... |

Buat ERD (Entity-Relationship Diagram):

- Buat diagram ERD yang menggambarkan hubungan antara tabel dimensi dan tabel fakta dalam skema bintang (Star Schema).



Berikut adalah penjelasan mengenai ERD (Entity Relationship Diagram) dengan Star Schema :

- Star Schema Overview

Fact Table: Tabel utama yang berisi data transaksi atau pengukuran (measures) seperti `order_id`, `qty`, `amount` dan lain-lain.

Dimension Tables: Tabel dimensi yang berisi informasi terkait. Tabel-tabel ini memberikan deskripsi tambahan untuk membantu analisis data dari fact table.

Dalam star schema, fact table terhubung langsung ke setiap dimension table melalui foreign key, yang membentuk bentuk bintang (star) jika digambar dalam diagram.

- Relasi dalam Star Schema

Relasi antar tabel didasarkan pada kunci utama (primary key) di setiap tabel dimensi dan kunci asing (foreign key) yang ada di fact table.

Struktur Relasi:

Fact Table: fact_salesorder

Fulfilment_id → Foreign key ke tabel dim_fulfilment (fulfilment_id)
Product_id → Foreign key ke tabel dim_product (product_id)
Promotion_id → Foreign key ke tabel dim_promotion (promotion_id)
Sales_channel_id → Foreign key ke tabel dim_sales_channel (sales_channel_id)
Sales_shipment_id → Foreign key ke tabel dim_sales_shipment (sales_shipment_id)

- Penjelasan Elemen-elemen ERD

Fact Table (fact_salesorder):

Tabel ini berada di tengah ERD dan berisi data faktual dari transaksi penjualan seperti jumlah pesanan (Qty), total penjualan (Amount), status, dan informasi lainnya. Fact table terhubung ke setiap dimension table melalui foreign key.

Dimension Tables:

Tabel-tabel dimensi (dim_fulfilment, dim_product, dim_promotion, dim_sales_channel, dan dim_sales_shipment) berfungsi sebagai deskripsi tambahan untuk memberikan konteks bagi data dalam fact table. Setiap dimension table terhubung ke fact table melalui relasi satu ke banyak (one-to-many). Misalnya, satu produk dapat muncul di banyak pesanan penjualan (satu product_id bisa ada di banyak fact_salesorder).

- Kelebihan Star Schema

1. Sederhana dan Mudah Dimengerti: Dengan star schema, relasi antara fact dan dimension tables sangat sederhana sehingga mudah dipahami oleh pengguna dan analis bisnis.
2. Efisiensi Query: Query analisis seperti perhitungan penjualan, promosi, atau pengiriman dapat dilakukan dengan mudah karena struktur tabelnya sudah dioptimalkan untuk performa analitik.
3. Desain Optimal untuk OLAP (Online Analytical Processing): Star schema sangat cocok untuk operasi OLAP, di mana data agregasi dan analisis membutuhkan akses cepat ke dimensi dan fakta.

Analisis Data:

Gunakan star schema yang telah kamu buat untuk melakukan analisis. Misalnya, buatlah analisis Top Selling Products atau analisis lain yang relevan.

-- Top 10 Selling Products Berdasarkan Jumlah Produk Terjual dan Pendapatan

SELECT

```
dp.style AS product_style,  
dp.category AS product_category,  
SUM(fs.qty) AS total_quantity_sold,  
SUM(fs.amount) AS total_sales_amount
```

FROM

```
`dibimbing-de.dibimbing.fact_salesorder` fs
```

LEFT JOIN

```
`dibimbing-de.dibimbing.dim_product` dp ON fs.product_id = dp.product_id
```

GROUP BY

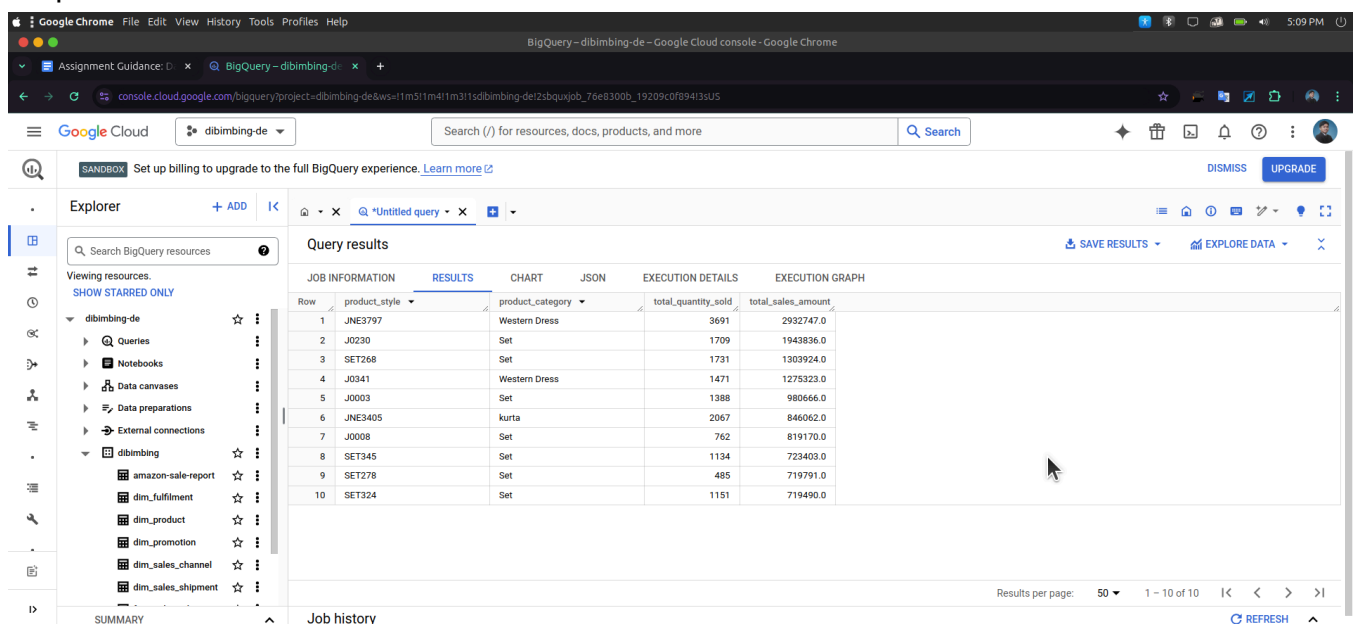
```
dp.style, dp.category
```

ORDER BY

```
total_sales_amount DESC, total_quantity_sold DESC
```

LIMIT 10;

Output :



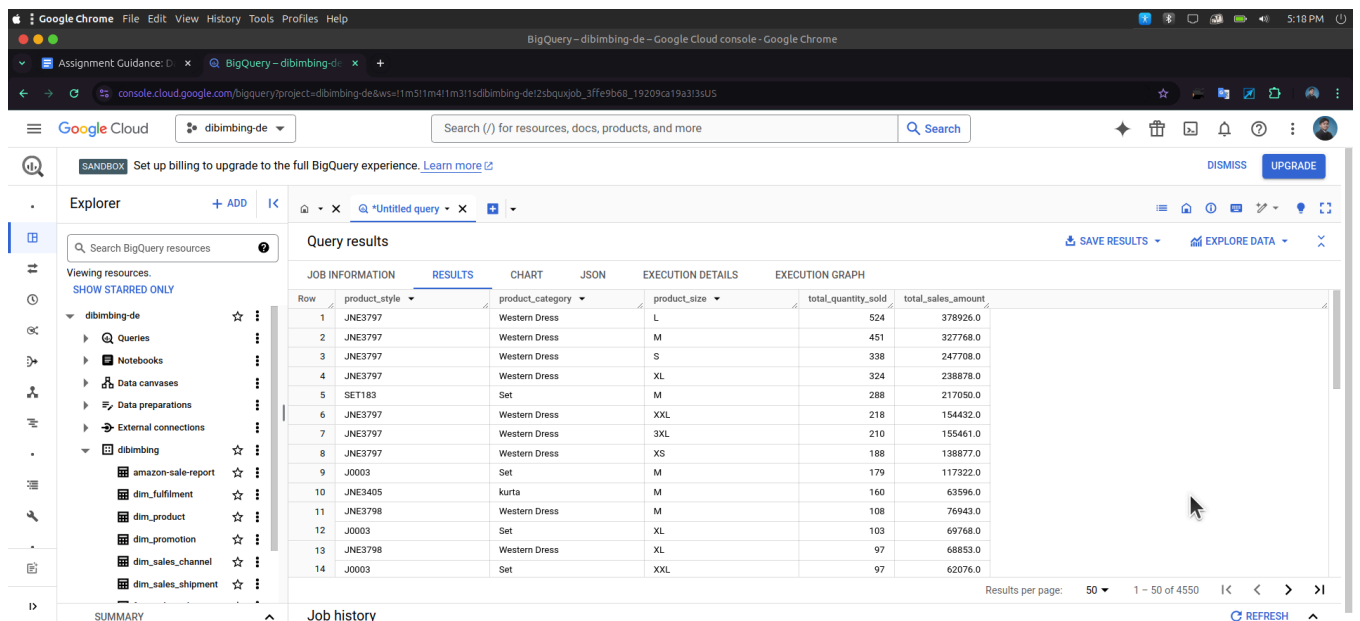
The screenshot shows the Google Cloud BigQuery console interface. On the left is the Explorer panel with a search bar and a list of resources under the 'dibimbing-de' project, including 'Queries', 'Notebooks', 'Data canvases', 'Data preparations', 'External connections', and a folder named 'dibimbing' containing various dimension tables like 'amazon-sale-report', 'dim_fulfillment', 'dim_product', 'dim_promotion', 'dim_sales_channel', and 'dim_sales_shipment'. The main area displays the 'Query results' for an 'Untitled query'. The results are shown in a table with columns: Row, product_style, product_category, total_quantity_sold, and total_sales_amount. The table lists the top 10 selling products, sorted by total_sales_amount in descending order. At the bottom right, there are controls for 'Results per page' (set to 50) and '1 - 10 of 10' results, along with navigation arrows and a 'REFRESH' button.

| Row | product_style | product_category | total_quantity_sold | total_sales_amount |
|-----|---------------|------------------|---------------------|--------------------|
| 1 | JNE3797 | Western Dress | 3691 | 2932747.0 |
| 2 | J0230 | Set | 1709 | 1943836.0 |
| 3 | SET268 | Set | 1731 | 1303924.0 |
| 4 | J0341 | Western Dress | 1471 | 1275323.0 |
| 5 | J0003 | Set | 1388 | 980666.0 |
| 6 | JNE3405 | kurta | 2067 | 846062.0 |
| 7 | J0008 | Set | 762 | 819170.0 |
| 8 | SET345 | Set | 1134 | 723403.0 |
| 9 | SET278 | Set | 485 | 719791.0 |
| 10 | SET324 | Set | 1151 | 719490.0 |

-- Produk Terlaris Berdasarkan Promosi yang Digunakan

```
SELECT
  dp.style AS product_style,
  dp.category AS product_category,
  dp.size AS product_size,
  SUM(fs.qty) AS total_quantity_sold,
  SUM(fs.amount) AS total_sales_amount
FROM
  `dibimbing-de.dibimbing.fact_salesorder` fs
LEFT JOIN
  `dibimbing-de.dibimbing.dim_product` dp
ON
  fs.product_id = dp.product_id
WHERE
  fs.promotion_id IN (
    SELECT
      promotion_id
    FROM
      `dibimbing-de.dibimbing.dim_promotion`
    WHERE
      promotions LIKE '% Free-Financing%'
  )
-- contoh saya pakai Free-Financing atau Pembiayaan Gratis Bisa diganti sesuai jenis
-- promosi yang diinginkan
)
GROUP BY
  dp.style, dp.category, dp.size
ORDER BY
  total_quantity_sold DESC;
```

Output :



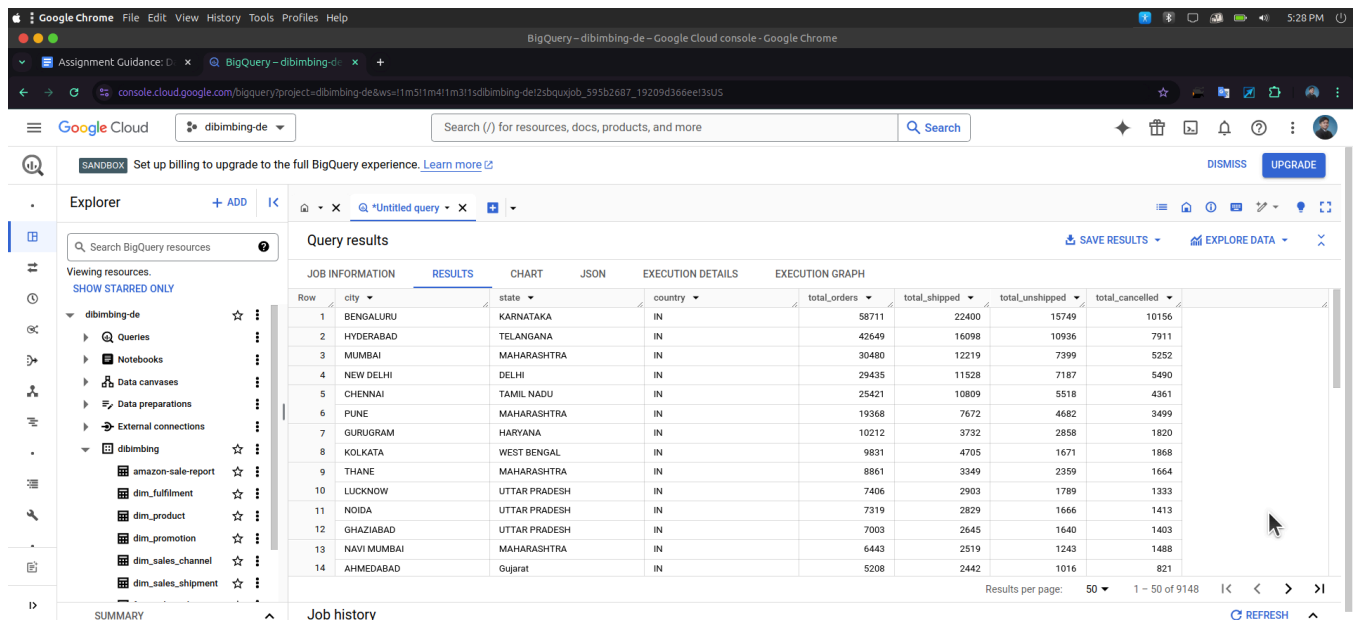
The screenshot shows the Google Cloud BigQuery console interface. The left sidebar displays the Explorer view with a tree structure of resources under the project 'dibimbing-de'. The main area shows the 'Query results' tab for an 'Untitled query'. The results are displayed in a table with columns: Row, product_style, product_category, product_size, total_quantity_sold, and total_sales_amount. The table contains 14 rows of data, sorted by total_quantity_sold in descending order. The bottom of the interface shows the 'Job history' tab and a 'REFRESH' button.

| Row | product_style | product_category | product_size | total_quantity_sold | total_sales_amount |
|-----|---------------|------------------|--------------|---------------------|--------------------|
| 1 | JNE3797 | Western Dress | L | 524 | 378926.0 |
| 2 | JNE3797 | Western Dress | M | 451 | 327768.0 |
| 3 | JNE3797 | Western Dress | S | 338 | 247708.0 |
| 4 | JNE3797 | Western Dress | XL | 324 | 238878.0 |
| 5 | SET183 | Set | M | 288 | 217050.0 |
| 6 | JNE3797 | Western Dress | XXL | 218 | 154432.0 |
| 7 | JNE3797 | Western Dress | 3XL | 210 | 155461.0 |
| 8 | JNE3797 | Western Dress | XS | 188 | 138877.0 |
| 9 | J0003 | Set | M | 179 | 117322.0 |
| 10 | JNE3405 | kurta | M | 160 | 63596.0 |
| 11 | JNE3798 | Western Dress | M | 108 | 76943.0 |
| 12 | J0003 | Set | XL | 103 | 69768.0 |
| 13 | JNE3798 | Western Dress | XL | 97 | 68853.0 |
| 14 | J0003 | Set | XXL | 97 | 62076.0 |

-- Kinerja Pengiriman Berdasarkan Kota, Provinsi dan Negara dari jumlah pesanan

```
SELECT
  dss.ship_city AS city,
  dss.ship_state AS state,
  dss.ship_country AS country,
  COUNT(fs.order_id) AS total_orders,
  SUM(CASE WHEN dss.courier_status = 'Shipped' THEN 1 ELSE 0 END) AS
total_shipped,
  SUM(CASE WHEN dss.courier_status = 'Unshipped' THEN 1 ELSE 0 END) AS
total_unshipped,
  SUM(CASE WHEN dss.courier_status = 'Cancelled' THEN 1 ELSE 0 END) AS
total_cancelled
FROM
  `dibimbing-de.dibimbing.fact_salesorder` fs
LEFT JOIN
  `dibimbing-de.dibimbing.dim_sales_shipment` dss
ON
  fs.sales_shipment_id = dss.sales_shipment_id
GROUP BY
  dss.ship_city, dss.ship_state, dss.ship_country
ORDER BY
  total_orders DESC;
```

Output :



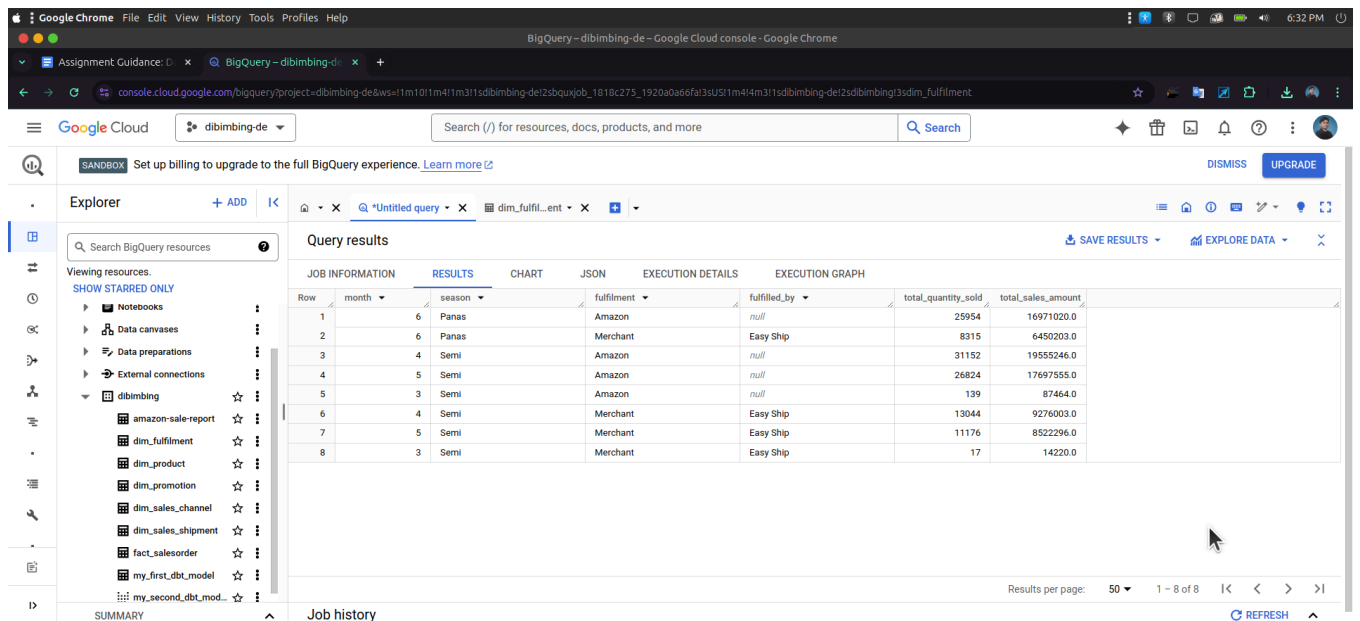
The screenshot displays the Google Cloud BigQuery console interface. The left sidebar shows the project hierarchy for 'dibimbing-de', including various data sources like 'amazon-sale-report', 'dim_fulfillment', 'dim_product', 'dim_promotion', 'dim_sales_channel', and 'dim_sales_shipment'. The main area shows the 'Query results' tab for an 'Untitled query'. The results are presented in a table with columns for city, state, country, total_orders, total_shipped, total_unshipped, and total_cancelled. The data is sorted by total_orders in descending order.

| Row | city | state | country | total_orders | total_shipped | total_unshipped | total_cancelled |
|-----|-------------|---------------|---------|--------------|---------------|-----------------|-----------------|
| 1 | BENGALURU | KARNATAKA | IN | 58711 | 22400 | 15749 | 10156 |
| 2 | HYDERABAD | TELANGANA | IN | 42649 | 16098 | 10936 | 7911 |
| 3 | MUMBAI | MAHARASHTRA | IN | 30480 | 12219 | 7399 | 5252 |
| 4 | NEW DELHI | DELHI | IN | 29435 | 11528 | 7187 | 5490 |
| 5 | CHENNAI | TAMIL NADU | IN | 25421 | 10809 | 5518 | 4361 |
| 6 | PUNE | MAHARASHTRA | IN | 19368 | 7672 | 4682 | 3499 |
| 7 | GURUGRAM | HARYANA | IN | 10212 | 3732 | 2858 | 1820 |
| 8 | KOLKATA | WEST BENGAL | IN | 9831 | 4705 | 1671 | 1868 |
| 9 | THANE | MAHARASHTRA | IN | 8861 | 3349 | 2359 | 1664 |
| 10 | LUCKNOW | UTTAR PRADESH | IN | 7406 | 2903 | 1789 | 1333 |
| 11 | NOIDA | UTTAR PRADESH | IN | 7319 | 2829 | 1666 | 1413 |
| 12 | GHAZIABAD | UTTAR PRADESH | IN | 7003 | 2645 | 1640 | 1403 |
| 13 | NAVI MUMBAI | MAHARASHTRA | IN | 6443 | 2519 | 1243 | 1488 |
| 14 | AHMEDABAD | Gujarat | IN | 5208 | 2442 | 1016 | 821 |

-- Penjualan Bulanan Berdasarkan Musim dan Metode Fulfillment

```
SELECT
  EXTRACT(MONTH FROM fs.date) AS month,
  CASE
    WHEN EXTRACT(MONTH FROM fs.date) IN (12, 1, 2) THEN 'Dingin'
    WHEN EXTRACT(MONTH FROM fs.date) IN (3, 4, 5) THEN 'Semi'
    WHEN EXTRACT(MONTH FROM fs.date) IN (6, 7, 8) THEN 'Panas'
    WHEN EXTRACT(MONTH FROM fs.date) IN (9, 10, 11) THEN 'Gugur'
  END AS season,
  df.fulfilment,
  df.fulfilled_by,
  SUM(fs.qty) AS total_quantity_sold,
  ROUND(SUM(fs.amount)) AS total_sales_amount
FROM
  `dibimbing-de.dibimbing.fact_salesorder` fs
LEFT JOIN
  `dibimbing-de.dibimbing.dim_fulfilment` df
ON
  fs.fulfilment_id = df.fulfilment_id
GROUP BY
  month, season, df.fulfilment, df.fulfilled_by;
```

Output :



The screenshot shows the Google Cloud BigQuery console interface. The 'Query results' tab is active, displaying a table with 8 rows of data. The table columns are: Row, month, season, fulfillment, fulfilled_by, total_quantity_sold, and total_sales_amount. The data is grouped by month and season, with each group having 2 rows corresponding to different fulfillment methods (Amazon and Merchant) and their respective fulfillment agents (null and Easy Ship).

| Row | month | season | fulfilment | fulfilled_by | total_quantity_sold | total_sales_amount |
|-----|-------|--------|------------|--------------|---------------------|--------------------|
| 1 | 6 | Panas | Amazon | null | 25954 | 16971020.0 |
| 2 | 6 | Panas | Merchant | Easy Ship | 8315 | 6450203.0 |
| 3 | 4 | Semi | Amazon | null | 31152 | 19555246.0 |
| 4 | 5 | Semi | Amazon | null | 26824 | 17697555.0 |
| 5 | 3 | Semi | Amazon | null | 139 | 87464.0 |
| 6 | 4 | Semi | Merchant | Easy Ship | 13044 | 9276003.0 |
| 7 | 5 | Semi | Merchant | Easy Ship | 11176 | 8522296.0 |
| 8 | 3 | Semi | Merchant | Easy Ship | 17 | 14220.0 |

Upload Kode ke GitHub:

Untuk mengakses repository github, silakan klik gambar dibawah ini atau [Github](#) :

