

Explored columnar databases by implementing data warehouse solutions by analyzing the sales data of an e-commerce platform stored in BigQuery to gain insights into product performance.

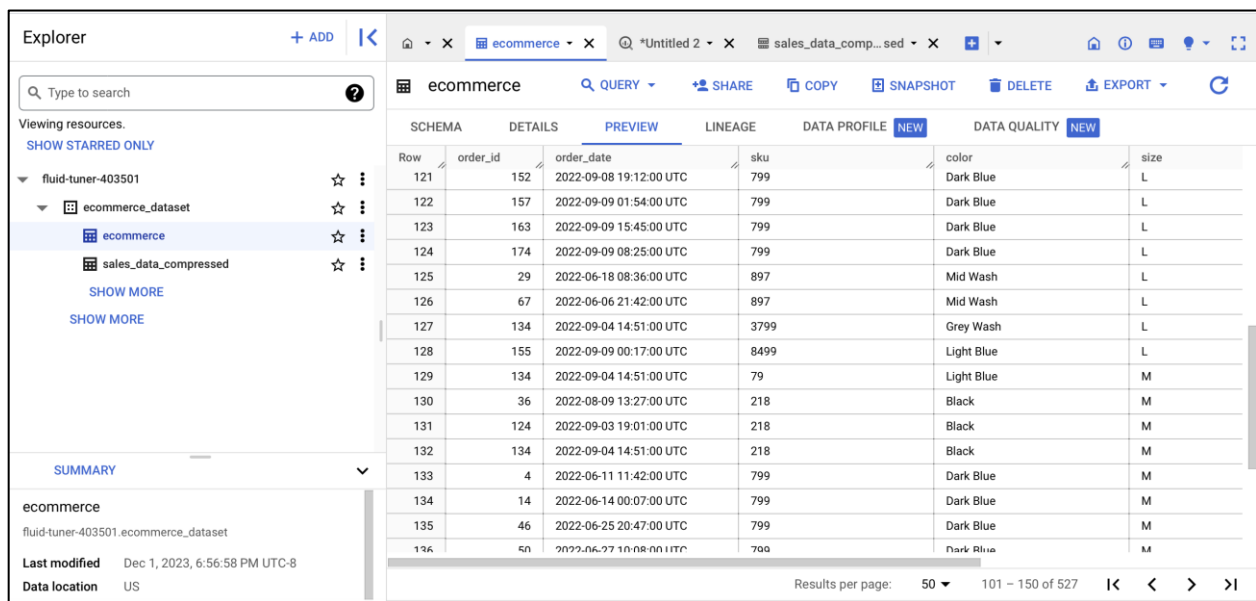
Took e-commerce dataset from Kaggle, <https://www.kaggle.com/datasets/shilongzhuang/-women-clothing-ecommerce-sales-data>

Used BigQuery as it is a fully managed, serverless data warehouse provided by Google Cloud. It stores data in a columnar format, making it well-suited for analytical workloads.

Step 1: Uploaded csv in Google Cloud Storage.

Step 2: Created BigQuery dataset as 'ecommerce_dataset' and loaded the csv uploaded to Google Cloud Storage.

Step 3: Created table as ecommerce and inserted values from storage to table.



The screenshot displays the Google Cloud BigQuery Explorer interface. On the left, the 'Explorer' pane shows the project 'fluid-tuner-403501' with a dataset 'ecommerce_dataset' containing a table 'ecommerce'. The main pane shows the 'ecommerce' table preview with columns: Row, order_id, order_date, sku, color, and size. The table contains 15 rows of data. The bottom pane shows the 'SUMMARY' for the 'ecommerce' table, including the last modified date and data location.

Row	order_id	order_date	sku	color	size
121	152	2022-09-08 19:12:00 UTC	799	Dark Blue	L
122	157	2022-09-09 01:54:00 UTC	799	Dark Blue	L
123	163	2022-09-09 15:45:00 UTC	799	Dark Blue	L
124	174	2022-09-09 08:25:00 UTC	799	Dark Blue	L
125	29	2022-06-18 08:36:00 UTC	897	Mid Wash	L
126	67	2022-06-06 21:42:00 UTC	897	Mid Wash	L
127	134	2022-09-04 14:51:00 UTC	3799	Grey Wash	L
128	155	2022-09-09 00:17:00 UTC	8499	Light Blue	L
129	134	2022-09-04 14:51:00 UTC	79	Light Blue	M
130	36	2022-08-09 13:27:00 UTC	218	Black	M
131	124	2022-09-03 19:01:00 UTC	218	Black	M
132	134	2022-09-04 14:51:00 UTC	218	Black	M
133	4	2022-06-11 11:42:00 UTC	799	Dark Blue	M
134	14	2022-06-14 00:07:00 UTC	799	Dark Blue	M
135	46	2022-06-25 20:47:00 UTC	799	Dark Blue	M
136	50	2022-06-27 10:08:00 UTC	799	Dark Blue	M

Step 3: Ran below analytical queries.

1. This query provides a ranked list of SKUs based on their total revenue, helping to identify the most financially successful products in the specified e-commerce dataset.

```
SELECT sku, SUM(revenue) AS total_revenue
FROM `fluid-tuner-403501.ecommerce_dataset.ecommerce`
GROUP BY sku
ORDER BY total_revenue DESC;
```

3	SELECT sku, SUM(revenue) AS total_revenue
4	FROM `fluid-tuner-403501.ecommerce_dataset.ecommerce`
5	GROUP BY sku
6	ORDER BY total_revenue DESC;
7	

← Query results	
JOB INFORMATION	RESULTS
Row	sku
16	897
17	127
18	8499
19	1499
20	539
21	229
22	628
23	439
24	29

2. This query provides a detailed breakdown of e-commerce sales, showing the total revenue, average unit price, and total quantity sold for each SKU and color combination, with results ordered by total revenue and grouped by sku and color.

```
SELECT sku, color,
SUM(unit_price * quantity) AS total_revenue,
AVG(unit_price) AS avg_unit_price,
SUM(quantity) AS total_quantity_sold
FROM `fluid-tuner-403501.ecommerce_dataset.ecommerce`
GROUP BY sku, color
ORDER BY total_revenue DESC;
```

3. Made use of efficient compression techniques to optimize performance with partitioning and clustering. Industry level use- it can help reduce storage costs and improve query performance.

```
CREATE TABLE ecommerce_dataset.sales_data_compressed
PARTITION BY DATE(order_date)
CLUSTER BY sku AS
SELECT order_id, order_date, sku, color, size, unit_price, quantity, revenue
FROM `fluid-tuner-403501.ecommerce_dataset.ecommerce`;
```

This query partitions the table by date and clusters it by SKU, potentially improving compression and query performance.

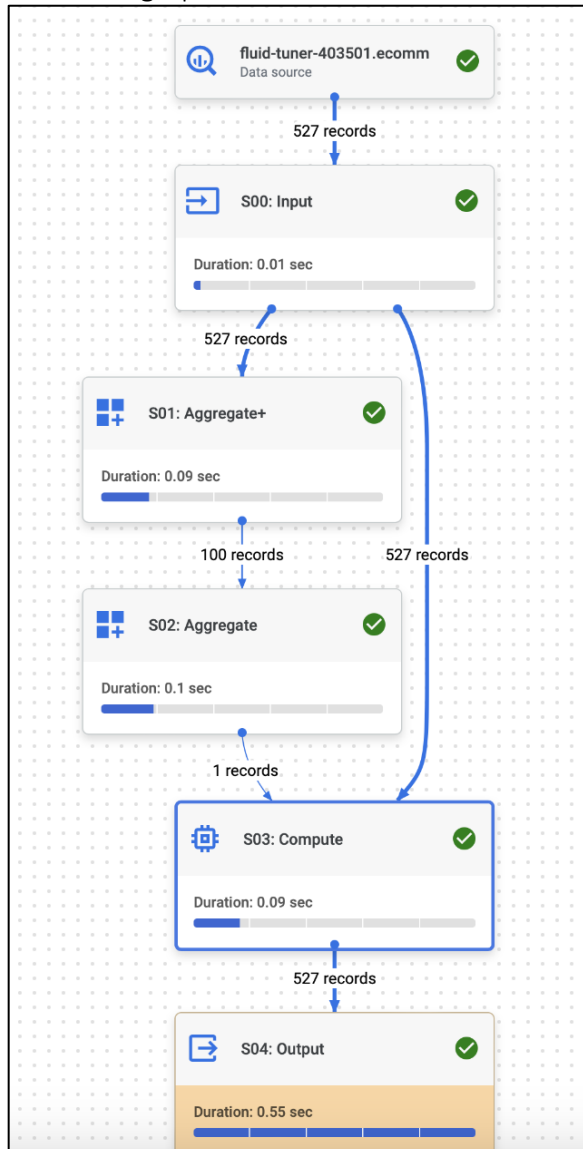
sales_data_compressed

QUERY SHARE COPY SNAPSHOT DELETE EXPORT REFRESH

This is a partitioned table. [Learn more](#) DISMISS

Row	order_id	order_date	sku	color	size	unit_price	quantity	revenue
79	200	2022-08-15 01:32:00 UTC	799	Dark Blue	XL	266	1	
80	208	2022-08-15 09:39:00 UTC	799	Dark Blue	XL	266	1	
81	203	2022-08-15 17:51:00 UTC	799	Dark Blue	XL	266	1	
82	207	2022-08-15 09:05:00 UTC	799	Dark Blue	XL	266	1	
83	201	2022-08-15 12:38:00 UTC	799	Dark Blue	XL	266	1	
84	205	2022-08-15 02:40:00 UTC	708	Dark Blue	3XL	277	1	
85	90	2022-09-02 13:03:00 UTC	799	Dark Blue	M	288	1	
86	91	2022-09-02 13:42:00 UTC	799	Dark Blue	XL	288	1	
87	91	2022-09-02 13:42:00 UTC	708	Dark Blue	2XL	298	1	
88	88	2022-09-02 11:26:00 UTC	708	Dark Blue	2XL	298	1	
89	89	2022-09-02 11:31:00 UTC	708	Dark Blue	2XL	298	1	
90	50	2022-06-27 10:08:00 UTC	799	Dark Blue	M	288	1	

Execution graph



4. Using columnar database we can also achieve parallel processing

When analyzing sales data for an e-commerce platform, multiple teams might run concurrent queries, such as sales by region, top-selling products, ensuring timely responses and optimal resource utilization.

5. In an e-commerce scenario, the sales data may come from multiple sources, such as logs, social media, and third-party applications. A columnar database has schema flexibility, and it will accommodate such data types and help businesses to make useful insights.

Salient features:

1. Execution graph: While working with columnar database, I came across execution graph which was quite attractive. It showcased how the query executed in parallel, how performance was optimized. This information would be of huge importance while working on large data and queries.

2. Columnar databases facilitate effective real-time analytics, playing a crucial role in ensuring the maintenance of ideal inventory levels, averting stock shortages, and reducing instances of excess inventory.

3. Columnar databases demonstrate superior performance in analytical queries, particularly in tasks such as aggregations, filtering, and the selection of specific columns. This proficiency stems from their vertical storage of data, enabling improved compression and the capability to selectively read only the required columns during the execution of queries.

4. Columnar databases can compress data well because values within a column are often similar. This means they need less storage space and can read and write data faster. Compression is important for large datasets because it lowers storage costs and makes the system work more efficiently overall.

5. Columnar databases usually have a set way of organizing data, but some can be flexible by allowing nested structures or semi-structured data. This flexibility is useful when handling different types of data and adapting to changing needs, which is common in modern e-commerce platforms.