



RETAIL REVENUE PREDICTION

DATA ANALYSIS USING THE AWS SERVICES

Presented By:

Himanshi Saini

Create a Bucket

General configuration

AWS Region

Asia Pacific (Mumbai) ap-south-1

Bucket type [Info](#)

- ☒ **General purpose**
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

Bucket name [Info](#)

Retail-sales

Object Ownership [Info](#)

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object

- ☒ **ACLs disabled (recommended)**
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

☒ Block *all* public access

Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

- ☒ **Block public access to buckets and objects granted through *new* access control lists (ACLs)**
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs fo
- ☒ **Block public access to buckets and objects granted through *any* access control lists (ACLs)**
S3 will ignore all ACLs that grant public access to buckets and objects.
- ☒ **Block public access to buckets and objects granted through *new* public bucket or access point policies**
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existin
- ☒ **Block public and cross-account access to buckets and objects through *any* public bucket or access point policies**
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

We have created our bucket

Bucket Versioning

- ☒ Disable
- ☐ Enable

Default encryption [Info](#)

Server-side encryption is automatically applied to new objects stored in this bucket.

Encryption type [Info](#)

- ☒ Server-side encryption with Amazon S3 managed keys (SSE-S3)
- ☐ Server-side encryption with AWS Key Management Service keys (SSE-KMS)
- ☐ Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)
Secure your objects with two separate layers of encryption. For details on pricing, see [DSSE-KMS pricing](#)

Bucket Key

Using an S3 Bucket Key

- ☐ Disable
- ☒ Enable

[Cancel](#)

Create bucket

General purpose buckets (2) [Info](#)

Copy ARN

Buckets are containers for data stored in S3.

Find buckets by name

	Name	AWS Region
<input type="radio"/>	honeybuck88976	Asia Pacific (Mumbai) ap-south-1
<input type="radio"/>	retailsalesbucket01	Asia Pacific (Mumbai) ap-south-1

We have prepared a bucket in which we can upload our data files (CSV, Excel, etc.). Whenever we need the files, we simply mention their location and use them.

Objects (4) [Info](#)

Copy

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get

Find objects by prefix

	Name	Type
<input type="checkbox"/>	<div><div></div><div>Retail Sales</div></div>	
<input type="checkbox"/>	<div><div></div><div>Cleaning_11Aug2025_1754892276139/</div></div>	Folder
<input type="checkbox"/>	<div><div></div><div>retail_store_inventory.csv</div></div>	csv
<input type="checkbox"/>	<div><div></div><div>retail_store_inventory.csv/</div></div>	Folder
<input type="checkbox"/>	<div><div></div><div>Retail-sales-new1_11Aug2025_1754907400502/</div></div>	Folder

Create Project in Glue Data Brew

The unique identifier for your DataBrew project (must follow naming rules)

Create project [Info](#)

Project details

Project name

Enter project name

The project name must contain 1-255 characters. Valid characters are alphanumeric (A-Z, a-z, 0-9), hyphen (-), period (.), and space.

Recipe details [Info](#)

Data cleaning steps in DataBrew are stored as a recipe. A recipe is connected to a project by default. An existing recipe with no associated project could also be applied to a project.

Attached recipe

Create new recipe ▼

Recipe name

Enter recipe name

The recipe name must contain 1-255 characters. Valid characters are alphanumeric (A-Z, a-z, 0-9), hyphen (-), period (.), and space.

☐ Import steps from recipe

Import recipe steps from an existing recipe into your project. The existing recipe that you chose will not be edited.

Explanation that cleaning/transform steps are stored as a recipe, which is linked to your project.

Option to build a brand-new recipe for this project.

The name for your recipe (must follow naming rules).

Choose a dataset option from where you want to select your dataset

Select a dataset

Select the dataset that you want to work on

My datasets

Your imported datasets

Sample files

Explore example files for your dataset

New dataset

Import new dataset

Select the IAM Role if already made, or create a new IAM Role

Permissions

Info

DataBrew needs permission to connect to data on your behalf. Use an IAM role with the **required policy** attached.

Role name

Choose the role that has access to connect to your data. Refresh to see the latest updates.

Choose an options

Like this, as shown below:

Q

Create new IAM role

Enter IAM role ARN

Choose existing IAM role

AWSGlueDataBrewServiceRole-Retail_IAM_role

arn:aws:iam::299189448554:role/service-role/AWSGlueDataBrewServiceRole-Retail_IAM_role

AWSGlueDataBrewServiceRole-Retail_sales_IAM_role

arn:aws:iam::299189448554:role/service-role/AWSGlueDataBrewServiceRole-Retail_sales_IAM_role

AWSGlueDataBrewServiceRole-himanshi_brew1

arn:aws:iam::299189448554:role/service-role/AWSGlueDataBrewServiceRole-himanshi_brew1

AWSGlueDataBrewServiceRole-himanshi_brew

arn:aws:iam::299189448554:role/service-role/AWSGlueDataBrewServiceRole-himanshi_brew

Final Click to create project

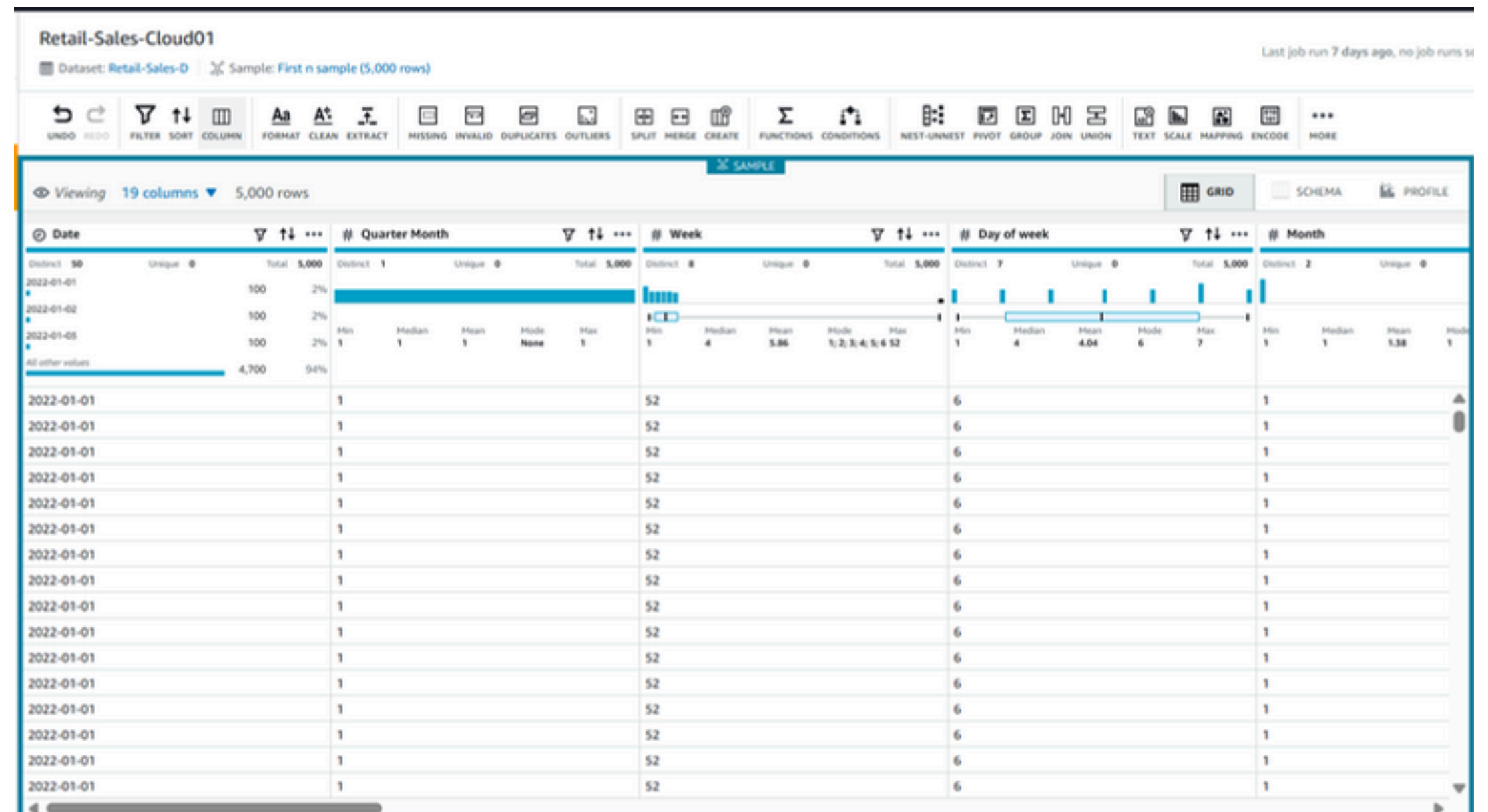
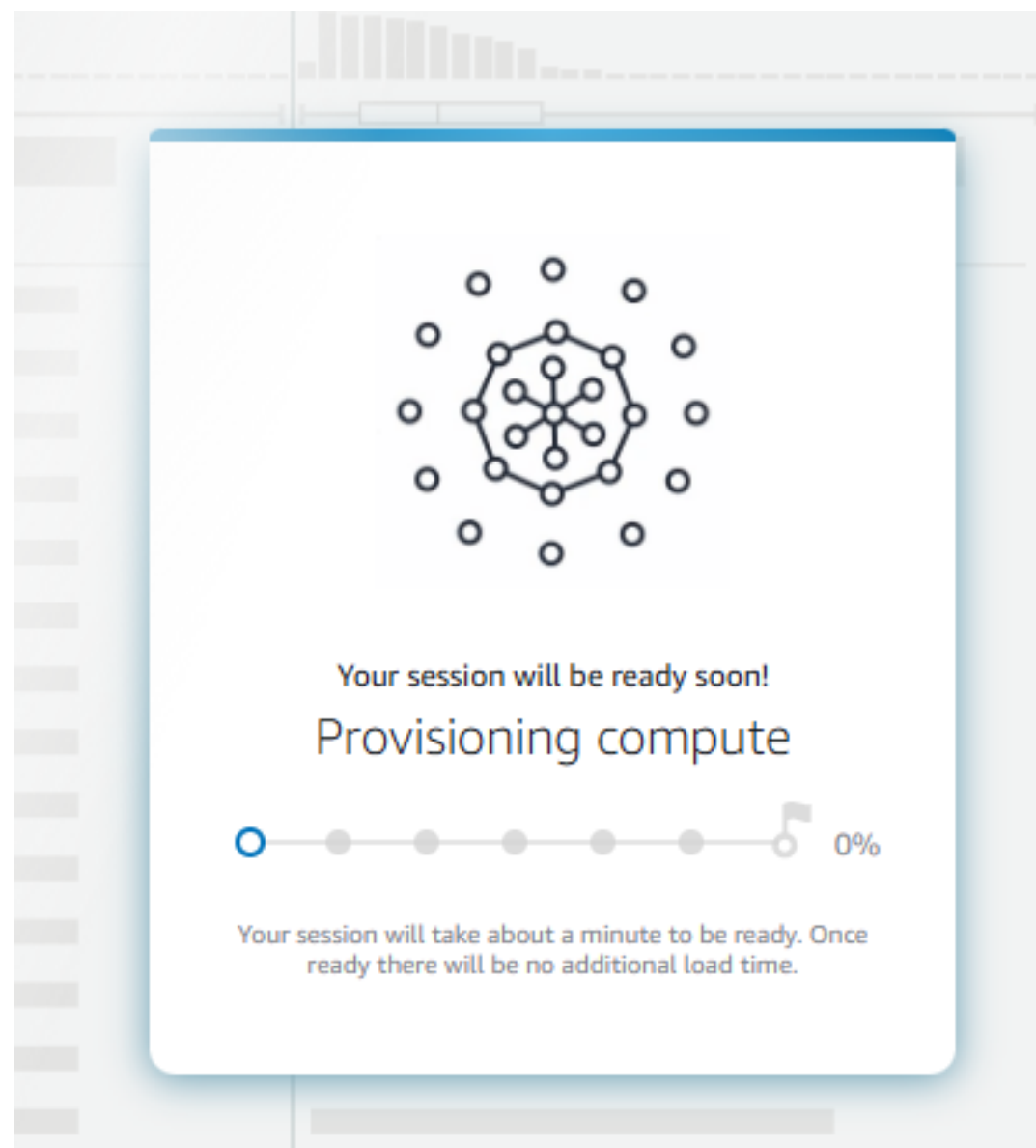
Cancel

Create project

Handling Data

Our Project is getting ready to be launched for Handling Inconsistencies and for data cleaning

This is the final page after loading, and here we will handle our data



We have performed several steps known as recipe to clean the data, preparing it for analysis and manipulation

<div><div></div><div>Recipe (9)</div><div></div></div>	
Retail-Sales-Cloud01-recipe Version 1.0	<div><div></div><div>Publish</div><div><div></div><div>More</div></div></div>
Applied steps (9) Clear all	<div><div></div><div></div></div>
1. Change format of Date to yyyy-mm-dd	
2. Create column Year using dateTime function YEAR	
3. Delete column Year	
4. Create column Month using dateTime function MONTH	
5. Change type of Date to Date	
6. Create column Date_extracted using dateTime function WEEK_DAY	
7. Rename Date_extracted to Day of week	
8. Create column Week using dateTime function WEEK_NUMBER	
9. Create column Quarter Month using dateTime function QUARTER	

These steps will not be applied to your entire dataset; they will only be implemented on your custom sample dataset. To apply these cleaning steps to the whole dataset, you need to create a new option called "Job." This option will enable you to apply all the cleaning steps to the entire dataset.

What is Job and how to Create Job

Name the “Job”


Create job[Info](#)


Job details

Job name
Identifier for the job

Applies your recipe’s cleaning/transformation steps to the full dataset for “RECIPE JOB” and”PROFILE JOB” for Analyzes your dataset to generate statistics, summaries, and data quality insights.

Job type
Type of job to run on the dataset

☒  **Create a recipe job**
Runs the transformations from the associated recipe on the population of the associated dataset.

☐  **Create a profile job**
Generates summary and statistics that give you the shape of your data.

Choose the dataset on which you want to run all these steps of cleaning and browse the file stores in the S3

Job input[Info](#)
The input dataset for the job and the recipe to be applied to it.

Run on

☒ **Dataset**
Run the job on an existing or new DataBrew dataset.

☐ **Project**
Run the job on a project with no associated job.

Choose dataset

Browse datasets

Connect new dataset

Select a recipe

Recipe version

Browse recipes

In AWS Glue DataBrew, a job function is the process that runs either data transformations (recipe job) or data profiling (profile job) on your dataset.

Job output settings[Info](#)
Running a job generates output files at specified file destinations.

Output 1

Output to
Output location

File type
Output format

Delimiter
CSV separator

Compression
Available types

S3 bucket owner's AWS account

☒ **Current AWS account**
299189448554

☐ **Another AWS account**

S3 location
Format is: s3://bucket/folder/

Browse

Choose the settings and options where you want your output to stored

Cancel

Create job

Create and run job

Output After Running Job

01

The processed output (from a recipe job) is stored in the Amazon S3 bucket you specify during job setup.

02

Then, you load the data into Athena for querying the database and conducting data analysis to understand it better.

Working on Athena for Querying the Data Base

1 .Launch the query editor

Get started

- ☒ Query your data with Trino SQL
Use Query editor to analyze data on S3, on premises, or on other clouds.
- ☐ Analyze your data using PySpark and Spark SQL
Use notebooks to build interactive Spark applications.

Launch query editor

2. Then we load the data from the S3 bucket or we can create db manually by option

Create a table from data source

- S3 bucket data
- AWS Glue Crawler [↗](#)

Create with SQL

- CREATE TABLE
- CREATE TABLE AS SELECT
- CREATE TABLE AS SELECT(ICEBERG)
- CREATE VIEW

3. Here we can see Tables created

▼ Tables (1)	< 1 >
⊕ retail_new_01	⋮
► Views (0)	< 1 >

3. Choose the db you want to work on

Database

retail_new_db ▲

Q

default

new_db


rea

retail_db

retail_new_db ✓

Query 1 : X | Query 2 : X | Query 3 : X | Query 4 : X | Query 5 : X | Query 6 : X | Query 7 : X | ✓ Retail_sales_queries : X

1

--  Understand the target variable (Units Sold)

2

3

4

-- What is the total and average sales per day, week, month, and quarter?

5

6

-- weekly total sales:

7

SELECT

8

"Week" AS week_num,

9

SUM("Units_Sold" * "Price") AS total_sales

10

FROM retail_new_01

11

GROUP BY "Week"

12

ORDER BY week_num;

13

14

-- weekly avg sales

15

-- WITH cte AS (

16

SELECT

17

"Week" AS week_num,

18

SUM("Units_Sold" * "Price") AS total_sales

19

FROM retail_new_01

20

GROUP BY "Week"

21

)

22

SELECT

ATHENA NOTEBOOK

This is the interface of the SQL Query Notebook where we run all queries to analyse the data



DATA ANALYSIS & QUERYING

Using Athena

WHAT IS THE TOTAL AND AVERAGE SALES PER DAY, WEEK, MONTH, AND QUARTER?

Weekly Total Revenue

```
SELECT
    "Week" AS week_num,
    SUM("Units_Sold" * "Price") AS revenue
FROM retail_new_01
GROUP BY "Week"
ORDER BY week_num;
```

Monthly Total Revenue

```
SELECT
    "month" AS Month_num,
    SUM("Units_Sold" * "Price") AS total_revenue
FROM retail_new_01
GROUP BY "month"
ORDER BY Month_num;
```

Weekly Avg Revenue

```
WITH cte AS (
    SELECT
        "Week" AS week_num,
        SUM("Units_Sold" * "Price") AS revenue
    FROM retail_new_01
    GROUP BY "Week"
)
SELECT
    week_num,
    AVG(total_sales) AS avg_revenue
FROM cte
GROUP BY week_num
ORDER BY week_num;
```

Monthly Avg Revenue

```
WITH cte AS (
    SELECT
        "month" AS Month_num,
        SUM("Units_Sold" * "Price") AS total_revenue
    FROM retail_new_01
    GROUP BY "month"
)
SELECT
    Month_num,
    AVG(revenue) AS avg_sales
FROM cte
GROUP BY Month_num
ORDER BY Month_num;
```

Quarter Sales

```
SELECT
    "quarter_month" AS quarter,
    SUM("Units_Sold" * "Price") AS revenue
FROM retail_new_01
GROUP BY "quarter_month"
ORDER BY quarter;
```

WHICH PRODUCTS GENERATE THE HIGHEST TOTAL REVENUE?

```
select product_id, sum(units_sold * price) as total_revenue
from retail_new_01
group by product_id
order by total_revenue desc limit 5;
```

WHICH PRODUCTS GENERATE THE HIGHEST TOTAL REVENUE?

```
SELECT
    DATE_FORMAT(date, '%Y-%m') AS year_month,
    SUM(units_sold * price) AS revenue
FROM
    retail_new_01
GROUP BY
    DATE_FORMAT(date, '%Y-%m')
ORDER BY
    DATE_FORMAT(date, '%Y-%m');
```

- Sales tend to be a little lower at the end of the year (November and December).
- Sales are a bit higher in the middle of the year, around June and July.

PRODUCT & STORE INSIGHTS

Which stores have the highest and lowest average daily revenue?

```
WITH daily_revenue AS (  
  SELECT  
    store_id,  
    product_id,  
    date,  
    SUM(Units_Sold * Price) AS total_revenue_per_day  
  FROM retail_new_01  
  GROUP BY store_id, product_id, date  
)  
SELECT  
  store_id,  
  product_id,  
  AVG(total_revenue_per_day) AS avg_daily_revenue  
FROM daily_revenue  
GROUP BY store_id, product_id  
ORDER BY avg_daily_revenue DESC;
```

Which product categories or individual products generate the most revenue?

```
select category , sum(units_sold * price) as total_revenue  
from retail_new_01  
group by category  
order by total_revenue desc ;
```

How does sales vary by region? Are some regions consistently outperforming others?

```
select region , sum(units_sold * price) as total_revenue  
from retail_new_01  
group by region  
order by total_revenue desc ;
```

WHAT'S THE LOW INVENTORY CONSTRAIN REVENUE?

Step 1: See average revenue at different inventory levels

```
SELECT
    inventory_level,
    AVG(Units_Sold * price) AS avg_revenue,
    COUNT(*) AS records_count
FROM retail_new_01
GROUP BY inventory_level
ORDER BY inventory_level;
```

How do units ordered compare to actual units sold (sales fulfillment gap)?

```
select units_ordered - units_sold as sales_fulfillment_gap
from retail_new_01
order by sales_fulfillment_gap desc ;
```

Step 2: Group inventory into buckets (if inventory has wide range)

```
SELECT
    CASE
        WHEN inventory_level BETWEEN 0 AND 10 THEN '0-10'
        WHEN inventory_level BETWEEN 11 AND 20 THEN '11-20'
        WHEN inventory_level BETWEEN 21 AND 50 THEN '21-50'
        ELSE '50+'
    END AS inventory_bucket,
    AVG(Units_Sold) AS avg_revenue,
    COUNT(*) AS records_count
FROM retail_new_01
GROUP BY
    CASE
        WHEN inventory_level BETWEEN 0 AND 10 THEN '0-10'
        WHEN inventory_level BETWEEN 11 AND 20 THEN '11-20'
        WHEN inventory_level BETWEEN 21 AND 50 THEN '21-50'
        ELSE '50+'
    END
ORDER BY inventory_bucket;
```

PRICING & PROMOTIONS

How do price changes?

```
SELECT
  CASE
    WHEN Price BETWEEN 0 AND 100 THEN 'Low'
    WHEN Price BETWEEN 101 AND 500 THEN 'Medium'
    ELSE 'High'
  END AS price_bin,
  AVG(units_sold) AS avg_units_sold,
  COUNT(*) AS records_count
FROM retail_new_01
GROUP BY
  CASE
    WHEN Price BETWEEN 0 AND 100 THEN 'Low'
    WHEN Price BETWEEN 101 AND 500 THEN 'Medium'
    ELSE 'High'
  END
ORDER BY price_bin;
```

Combined effect of Price and Discount on Revenue

```
SELECT
  CASE
    WHEN Price BETWEEN 0 AND 100 THEN 'Low'
    WHEN Price BETWEEN 101 AND 500 THEN 'Medium'
    ELSE 'High'
  END AS price_bin,
  CASE
    WHEN Discount BETWEEN 0 AND 10 THEN 'Low'
    WHEN Discount BETWEEN 11 AND 30 THEN 'Medium'
    ELSE 'High'
  END AS discount_bin,
  AVG(units_sold) AS avg_units_sold,
  COUNT(*) AS records_count
FROM retail_new_01
GROUP BY
  CASE
    WHEN Price BETWEEN 0 AND 100 THEN 'Low'
    WHEN Price BETWEEN 101 AND 500 THEN 'Medium'
    ELSE 'High'
  END,
  CASE
    WHEN Discount BETWEEN 0 AND 10 THEN 'Low'
    WHEN Discount BETWEEN 11 AND 30 THEN 'Medium'
    ELSE 'High'
  END
ORDER BY price_bin, discount_bin;
```

EXTERNAL FACTORS

How does weather condition influence revenue?

```
SELECT
    Weather_Condition,
    COUNT(*) AS num_days,
    SUM(Units_Sold*price) AS total_revenue,
    AVG(Units_Sold*price) AS avg_revenue
FROM retail_new_01
GROUP BY Weather_Condition
ORDER BY avg_revenue DESC;
```

Does holiday/promotion combined with weather or seasonality boost revenue more than any factor alone?

a) Holiday & Weather Interaction

```
SELECT
    holiday_promotion,
    weather_condition,
    COUNT(*) AS num_days,
    SUM(units_sold * price) AS total_revenue,
    AVG(units_sold * price ) AS avg_revenue
FROM retail_new_01
GROUP BY holiday_promotion, weather_condition
ORDER BY holiday_promotion, avg_revenue DESC;
```

b) Holiday & Seasonality Interaction

```
SELECT
    holiday_promotion,
    Seasonality,
    COUNT(*) AS num_days,
    SUM(units_sold) AS total_revenue,
    AVG(units_sold) AS avg_revenue
FROM retail_new_01
GROUP BY holiday_promotion, Seasonality
ORDER BY holiday_promotion, avg_revenue DESC;
```

DEMAND FORECAST & INVENTORY PLANNING

Inventory Level vs Demand Forecast & Actual sales

```
SELECT
    Date,
    store_id,
    Product_ID,
    Inventory_Level,
    Units_Sold,
    Demand_Forecast,
    (Inventory_Level - Units_Sold) AS inventory_remaining,
    (Inventory_Level - Demand_Forecast) AS inventory_vs_forecast
FROM retail_new_01
ORDER BY inventory_remaining ASC;
```

Interaction effects on sales

```
SELECT
    Category,
    Region,
    AVG(units_sold) AS avg_units_sold,
    COUNT(*) AS num_records
FROM retail_new_01
GROUP BY Category, Region
ORDER BY Category, Region;
```

Holiday × Discount

```
SELECT
    Holiday_Promotion,
    Discount,
    AVG(units_sold) AS avg_units_sold,
    COUNT(*) AS num_records
FROM retail_new_01
GROUP BY Holiday_Promotion, Discount
ORDER BY Holiday_Promotion, Discount;
```

END

01

After querying the data,
we can either store it in
Amazon Redshift, or we
can load the cleaned CSV
file and make a model in
an IPython notebook file

02

We can also create a
dashboard using
Amazon QuickSight.

03

But in our case we will
load the data in python
notebook and build the
ml model there and use
the power bi for
visualization



THANK YOU!

Contact : himanshisaini7802@gmail.com