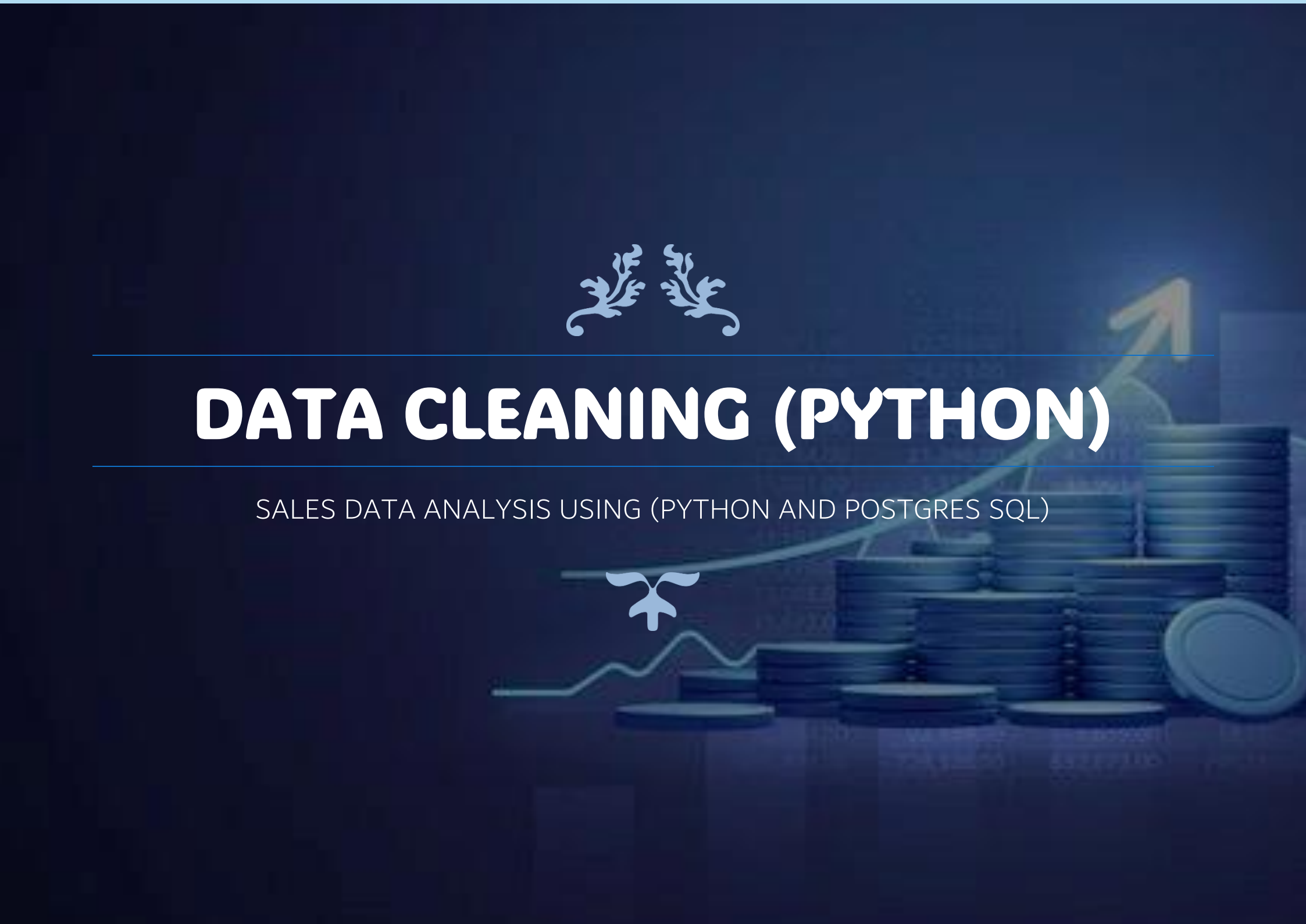
SALES DATA ANALYSIS

(Python + SQL)



DATA CLEANING (PYTHON)

SALES DATA ANALYSIS USING (PYTHON AND POSTGRES SQL)

Connect to Kaggle API and download file

```
import kaggle

!kaggle datasets download ankitbansal06/retail-orders -f orders.csv

Dataset URL: https://www.kaggle.com/datasets/ankitbansal06/retail-orders License(s): CC0-1.0

Downloading orders.csv.zip to C:\Users\kumar\LEARNING\Python + Postgres


  0%|                                        | 0.00/200k [00:00<?, ?B/s]
100%|##########################################| 200k/200k [00:00<00:00, 232kB/s]
```

Extract csv file from zipped download file

```
import zipfile
zip_ref = zipfile.ZipFile('orders.csv.zip') zip_ref.extractall() # extract file to dir
zip_ref.close() # close file
```

Import Libraries

```
import pandas as pd
```

READ Dataset

```
data=pd.read_csv('orders.csv',na_values=['Not Available','unknown'])
data.head(10)
```

	Order Id	Order Date	Ship Mode	Segment	Country
0	1	2023-03-01	Second Class	Consumer	United States
1	2	2023-08-15	Second Class	Consumer	United States
2	3	2023-01-10	Second Class	Corporate	United States
3	4	2022-06-18	Standard Class	Consumer	United States
4	5	2022-07-13	Standard Class	Consumer	United States
5	6	2022-03-13	NaN	Consumer	United States
6	7	2022-12-28	Standard Class	Consumer	United States
7	8	2022-01-25	Standard Class	Consumer	United States
8	9	2023-03-23	NaN	Consumer	United States

	City	State	Postal Code	Region	Category
0	Henderson	Kentucky	42420	South	Furniture
1	Henderson	Kentucky	42420	South	Furniture
2	Los Angeles	California	90036	West	Office Supplies
3	Fort Lauderdale	Florida	33311	South	Furniture
4	Fort Lauderdale	Florida	33311	South	Office Supplies
5	Los Angeles	California	90032	West	Technology
6	Los Angeles	California	90032	West	Office Supplies
7	Los Angeles	California	90032	West	Office Supplies
8	Los Angeles	California	90032	West	Office Supplies

	Sub Category	Product Id	cost price	List Price	Quantity
0	Bookcases	FUR-BO-10001798	240	260	2
1	Chairs	FUR-CH-10000454	600	730	3
2	Labels	OFF-LA-10000240	10	10	2
3	Tables	FUR-TA-10000577	780	960	5

4	Storage	OFF-ST-10000760	20	20	2
5	Furnishings	FUR-FU-10001487	50	50	7
6	Art	OFF-AR-10002833	10	10	4
7	Phones	TEC-PH-10002275	860	910	6
8	Binders	OFF-BI-10003910	20	20	3
9	Appliances	OFF-AP-10002892	90	110	5

	Discount Percent
0	2
1	3
2	5
3	2
4	5
5	3
6	3
7	5
8	2
9	3

```
data.dtypes
```

```
Order Id          int64
Order Date       object
Ship Mode        object
Segment          object
Country          object
City             object
State            object
Postal Code       int64
Region           object
Category         object
Sub Category     object
Product Id       object
cost price        int64
```

Formatting columns

```
data.columns=data.columns.str.lower().str.replace(' ','_')
```

Date Time format change

```
data['order_date']=pd.to_datetime(data['order_date'],format='%Y-%m-%d')
```

Column addition

```
data['profit']=data['list_price']-data['cost_price']
data['discount']=data['list_price']*data['discount_percent']*.01
data['sale_price']=data['list_price']-data['discount']
```

drop useless columns

```
data.drop(['list_price','cost_price','discount_percent'],axis='columns',inplace=True) data.head()
```

	order_id	order_date	ship_mode	segment	country	\
0	1	2023-03-01	Second Class	Consumer	United States	
1	2	2023-08-15	Second Class	Consumer	United States	
2	3	2023-01-10	Second Class	Corporate	United States	
	4	2022-06-18	Standard Class	Consumer	United States	

	city	state	postal_code	region	category	\
0	Henderson	Kentucky	42420	South	Furniture	

```
1         Henderson     Kentucky         42420   South         Furniture
2       Los Angeles   California         90036    West   Office Supplies
3   Fort Lauderdale      Florida         33311   South         Furniture
4   Fort Lauderdale      Florida         33311   South   Office Supplies
```

```
  sub_category       product_id   quantity   profit   discount   sale_price
0    Bookcases   FUR-BO-10001798          2       20        5.2        254.8
1       Chairs   FUR-CH-10000454          3      130       21.9        708.1
2       Labels   OFF-LA-10000240          2        0        0.5          9.5
3       Tables   FUR-TA-10000577          5      180       19.2        940.8
4      Storage   OFF-ST-10000760          2        0        1.0         19.0
```

CREATE TABLE IN POSTGRES (using psycopg2 libarary)

```
import psycopg2

# connect and create a new database

try:
    conn=psycopg2.connect(host='localhost',password='harsh',user='postgres')

    # auto commit

    conn.autocommit=True

    # cursor

    cur=conn.cursor()
    cur.execute('''CREATE DATABASE Orders;''')

    conn.close()
except psycopg2.errors as e:
    print(e)
```

```
    conn=psycopg2.connect(database='orders',host='localhost',password='harsh',user='postgres')
except psycopg2.errors as e:
    print(e)

# create cursor conn.autocommit=True
cur=conn.cursor()
```

CREATE TABLE Orders_table

```
query = """
CREATE TABLE Orders_table (
    order_id INT,
    order_date DATE,
    ship_mode VARCHAR,
    segment VARCHAR,
    country VARCHAR,
    city VARCHAR,
    state VARCHAR,
    postal_code INT,
    region VARCHAR,
    category VARCHAR,
    sub_category VARCHAR,
    product_id VARCHAR,
    quantity INT,
    profit INT,
    discount FLOAT,
    sale_price FLOAT
);
"""

try:
    cur.execute(query)
    print("Table Created")
```

```
except psycopg2.errors as e:
    print(e)
```

INSERT DATA In New TAble

```
insert_query = """
INSERT INTO Orders_table
    (order_id, order_date, ship_mode, segment, country, city, state, postal_code,
    region, category, sub_category, product_id, quantity, profit, discount, sale_price)
VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)
"""
try:
    for index, row in data.iterrows():
        # Execute the insert query for each row

        cur.execute(insert_query, row)
    print("Successfull insesrtion of data")
except psycopg2.errors as e:
    print(e)
cur.execute('''SELECT * from Orders_table''')

pd.DataFrame(cur.fetchall())
```

	0	1	2	3	4	\
0	1	2023-03-01	Second Class	Consumer	United States	
1	2	2023-08-15	Second Class	Consumer	United States	
2	3	2023-01-10	Second Class	Corporate	United States	
3	4	2022-06-18	Standard Class	Consumer	United States	
4	5	2022-07-13	Standard Class	Consumer	United States	
...	...	...	...	...	...	
9989	9990	2023-02-18	Second Class	Consumer	United States	
9990	9991	2023-03-17	Standard Class	Consumer	United States	
9991	9992	2022-08-07	Standard Class	Consumer	United States	
9992	9993	2022-11-19	Standard Class	Consumer	United States	
9993	9994	2022-07-17	Second Class	Consumer	United States	

```
                  5           6      7      8                 9            10  \
0          Henderson    Kentucky  42420  South         Furniture    Bookcases
1          Henderson    Kentucky  42420  South         Furniture       Chairs
2        Los Angeles  California  90036   West  Office Supplies       Labels
3     Fort Lauderdale     Florida  33311  South         Furniture       Tables
4     Fort Lauderdale     Florida  33311  South  Office Supplies      Storage
...               ...         ...    ...    ...               ...          ...
9989            Miami     Florida  33180  South         Furniture  Furnishings
9990       Costa Mesa  California  92627   West         Furniture  Furnishings
9991       Costa Mesa  California  92627   West        Technology       Phones
9992       Costa Mesa  California  92627   West  Office Supplies        Paper
9993      Westminster  California  92683   West  Office Supplies   Appliances


[9994 rows x 16 columns]
```

CLOSE CONNECTION

```
conn.close()
```

CONCLUSION

1. **Import dataset from Kaggle API**

   ```
   i. Unzip file and load dataset
   ii. Data Cleaning
   iii. Featuree Creation
   ```

2. **Connection to PostgreSQL**

   ```
   i. Creating database and table
   ii. Insertion of data into PostgreSQL
   ```

SQL QUERIES

SALES DATA ANALYSIS USING (PYTHON AND POSTGRES SQL)

FIND TOP 10 HIGHEST REVENUE GENERATING PRODUCTS

```
SELECT
        PRODUCT_ID, ROUND(SUM(SALE_PRICE)::NUMERIC,1) AS TOTAL_REVENUE
FROM ORDERS_TABLE
GROUP BY 1
ORDER BY SUM(SALE_PRICE) DESC
LIMIT 10;
```

product_id	total_revenue
TEC-CO-10004722	59514
OFF-BI-10003527	26525.3
TEC-MA-10002412	21734.4
FUR-CH-10002024	21096.2
OFF-BI-10001359	19090.2
OFF-BI-10000545	18249
TEC-CO-10001449	18151.2
TEC-MA-10001127	17906.4
OFF-BI-10004995	17354.8
OFF-SU-10000151	16325.8

FIND TOP 5 HIGHEST SELLING PRODUCTS IN EACH REGION

```
WITH CTE AS (
        SELECT
                REGION, PRODUCT_ID, ROUND(SUM(SALE_PRICE)::NUMERIC,1) AS TOTAL_SALES,
                ROW_NUMBER () OVER (PARTITION BY REGION ORDER BY SUM(SALE_PRICE)
DESC) AS RN
        FROM ORDERS_TABLE
        GROUP BY 1,2
)
SELECT
        REGION, PRODUCT_ID, TOTAL_SALES, RN AS RANK
FROM CTE
WHERE RN<=5
ORDER BY REGION, RN ASC;
```

region	product_id	total_sales	rank
Central	TEC-CO-10004722	16975	1
Central	TEC-MA-10000822	13770	2
Central	OFF-BI-10001120	11056.5	3
Central	OFF-BI-10000545	10132.7	4
Central	OFF-BI-10004995	8416.1	5
East	TEC-CO-10004722	29099	1
East	TEC-MA-10001047	13767	2
East	FUR-BO-10004834	11274.1	3
East	OFF-BI-10001359	8463.6	4
East	TEC-CO-10001449	8316	5
South	TEC-MA-10002412	21734.4	1
South	TEC-MA-10001127	11116.4	2
South	OFF-BI-10001359	8053.2	3
South	TEC-MA-10004125	7840	4
South	OFF-BI-10003527	7391.4	5
West	TEC-CO-10004722	13440	1
West	OFF-SU-10000151	12592.3	2
West	FUR-CH-10001215	9604	3
West	OFF-BI-10003527	7804.8	4
West	TEC-AC-10003832	7722.7	5

FIND TOP 5 HIGHEST SELLING PRODUCTS IN EACH REGION

```
WITH CTE AS (
        SELECT
                REGION, PRODUCT_ID, ROUND(SUM(SALE_PRICE)::NUMERIC,1) AS TOTAL_SALES,
                ROW_NUMBER () OVER (PARTITION BY REGION ORDER BY SUM(SALE_PRICE) DESC) AS RN
        FROM ORDERS_TABLE
        GROUP BY 1,2
)
SELECT
        REGION, PRODUCT_ID, TOTAL_SALES, RN AS RANK
FROM CTE
WHERE RN<=5
ORDER BY REGION, RN ASC;
```

month	sales_2022	sales_2023
1	94712.5	88632.6
2	90091	128124.2
3	80106	82512.3
4	95451.6	111568.6
5	79448.3	86447.9
6	94170.5	68976.5
7	78652.2	90563.8
8	104808	87733.6
9	79142.2	76658.6
10	118912.7	121061.5
11	84225.3	75432.8
12	95869.9	102556.1

FOR EACH CATEGORY WHICH MONTH HAD HIGHEST SALES

```
WITH CTE AS (
        SELECT CATEGORY,
                EXTRACT (YEAR FROM ORDER_DATE) AS YEAR,
                EXTRACT (MONTH FROM ORDER_DATE) AS MONTH,
                ROUND(SUM(SALE_PRICE)::NUMERIC,1) AS TOTAL_SALES
        FROM ORDERS_TABLE
        GROUP BY 1,2,3
        ORDER BY 1,2,3),
CTE2 AS (
        SELECT
                CATEGORY, YEAR, MONTH, TOTAL_SALES,
                 MAX(TOTAL_SALES) OVER (PARTITION BY CATEGORY) AS MAX_SALES
        FROM CTE
)
SELECT
        CATEGORY, YEAR, MONTH, TOTAL_SALES
FROM CTE2
WHERE TOTAL_SALES=MAX_SALES;
```

category	year	month	total_sales
Furniture	2022	10	42888.9
Office Supplies	2023	2	44118.5
Technology	2023	10	53000.1

WHICH SUB CATEGORY HAD HIGHEST GROWTH BY PROFIT IN 2023 COMPARE TO 2022

```
WITH CTE AS(
        SELECT SUB_CATEGORY, EXTRACT (YEAR FROM ORDER_DATE) AS YEAR, SUM(PROFIT) AS TOTAL_PROFIT
        FROM ORDERS_TABLE
        GROUP BY 1,2),
CTE2 AS(
        SELECT
                SUB_CATEGORY,
                SUM (CASE WHEN YEAR=2022 THEN TOTAL_PROFIT ELSE 0 END) AS TOTAL_PROFIT_2022,
                SUM (CASE WHEN YEAR=2023 THEN TOTAL_PROFIT ELSE 0 END) AS TOTAL_PROFIT_2023,
                SUM (CASE WHEN YEAR=2023 THEN TOTAL_PROFIT ELSE 0 END)-SUM (CASE WHEN YEAR=2022 THEN TOTAL_PROFIT ELSE 0 END) AS PROFIT_INCREASE
        FROM CTE
        GROUP BY SUB_CATEGORY)
SELECT *,
        CONCAT (ROUND (((PROFIT_INCREASE *100)/TOTAL_PROFIT_2022)::NUMERIC,2),' %') AS GROWTH_PCT
FROM CTE2
ORDER BY PROFIT_INCREASE DESC
LIMIT 1;
```

sub_category	total_profit_2022	total_profit_2023	profit_increase	growth_pct
Machines	9980	14500	4520	45.29%

THANK YOU

Link to Github