Rakamin Academy X KALBE Nutritionals

Data Warehouse Implementation

Kalbe Data Engineer VIX

Presented by
Fawwaz Nurmansyah

Fawwaz Nurmansyah

About Me:

A graduate Bachelor of Biology from Airlangga University who is interest in pursuing a career related to data after completing some bootcamp. Able to work well independently and as part of a team, even under tight deadlines and pressure.

Have less than a year experience in data science with understanding of data processing, data analysis, and machine learning. Proficient in extracting insights from complex datasets and transforming them into profitable recommendations for business.

Rakamin Academy

Experience

Final Project Data Science Bootcamp
Rakamin Academy

Doing Exploratory Data Analysis on an E-commerce Shipping Data from Kaggle, preprocessing data, and fitting a Machine Learning model to the dataset using Decision Tree, K-Nearest Neighbour (KNN), Adaboost, Extreme Gradient Boost (XGBoost), Random Forest Classifier, and CATBoost. The best fit models to predict the dataset using XGBoost with highest recall score of 98% and 97% recall cross-validation training and testing data.
Portofolio Link

Case Study

Rakamin Academy

Demonstrasi menjadwalkan run Bash Script dengan menggunakan crontab

rakamin.sh

```
#!/bin/bash
path=/hdfs/data/data1
name_of_directory=data1
dir=$path/$name_of_directory
filename_excel=daily_market_price.xlsx
source_dir=/local/data/market
if [ -d $dir ]; then
    echo "Direktori ${name_of_directory} ada!"
    cp $source_dir/$filename_excel $path
    echo "File berhasil dipindah!"
else
    echo "Direktori ${name_of_directory} tidak ada!"
    mkdir $dir
    exit 1
fi
```

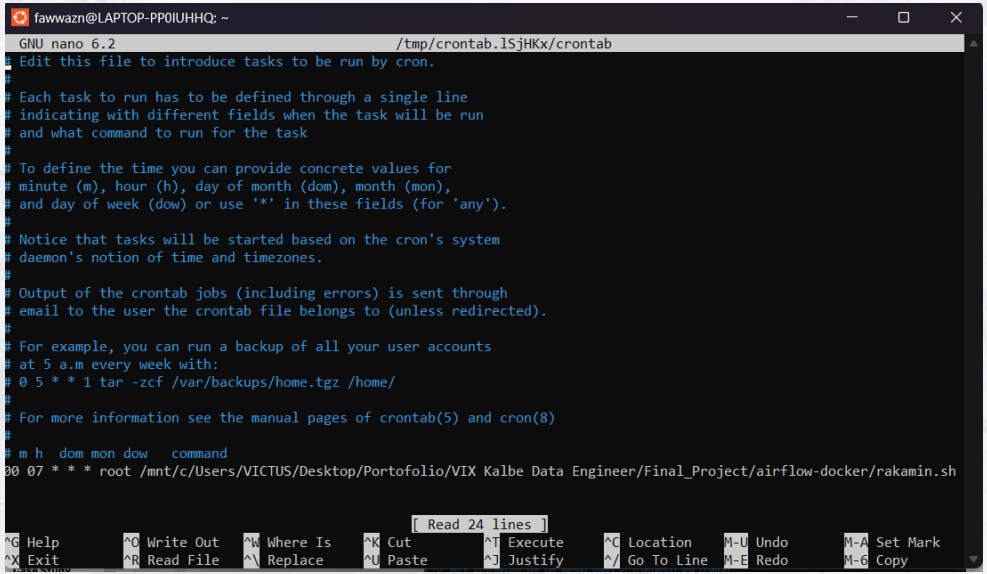
Membuat bash script untuk memeriksa direktori tersebut ada didalam path yang diberikan.

Bash script here

Case Study

Rakamin Academy

Demonstrasi menjadwalkan run Bash Script dengan menggunakan crontab



Membuat syntax crontab untuk menjalankan script bash yang dibuat pada jam 07:00 setiap hari.

Case Study

Rakamin Academy

Mengatur database di PostgreSQL ke Python lewat Jupyter Notebook

Import libraries

```
[1]
import psycopg2
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
```

Connecting to PostgreSQL

```
[2]
try:
    connection = psycopg2.connect(
        dbname='KALBE',
        user='postgres',
        password='**********',
        host='localhost',
        port='5432'
    )
    print('Successfully connected to the database!')
except psycopg2.Error as e:
    print(f"ERROR: {e}")
```

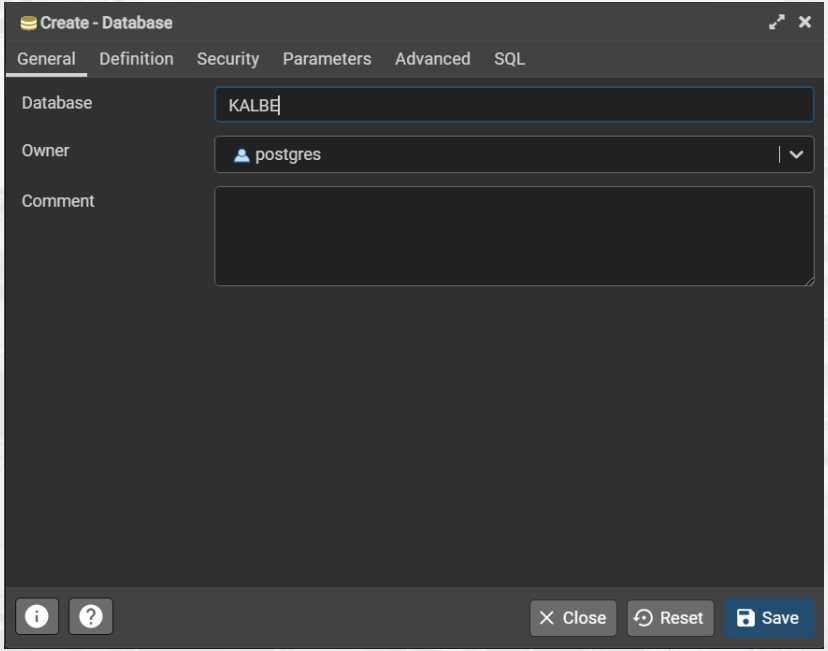
```
Successfully connected to the database!
```

Membuat koneksi ke database PostgreSQL dengan menggunakan library psycopg2 di Jupyter Notebook.

Case Study

Rakamin Academy

Mengatur database di PostgreSQL ke Python lewat Jupyter Notebook



Membuat database dengan nama ‘KALBE’ di PostgreSQL.

Case Study

Rakamin Academy

Mengatur database di PostgreSQL ke Python lewat Jupyter Notebook

Query | Query History

```
1  CREATE TABLE inventory (
2      Item_code VARCHAR (50),
3      Item_name VARCHAR (100),
4      Item_price INT,
5      Item_total INT
6  );
7
8  CREATE TABLE customer_orders (
9      order_no INT,
10     purchase_amount INT,
11     order_date DATE,
12     customer_id INT,
13     salesman_id INT
14 );
```

Membuat tabel inventory dan customer_orders dalam database ‘KALBE’ dan import file .csv ke dalam table tersebut.

Case Study

Rakamin Academy

Mengatur database di PostgreSQL ke Python lewat Jupyter Notebook

Query | Query History

```
1  SELECT * FROM public.customer_orders
2
```

Data Output | Messages | Notifications

	order_no integer	purchase_amount integer	order_date date	customer_id integer	salesman_id integer
1	10001	150	2022-10-05	2005	3002
2	10009	279	2022-09-10	2001	3005
3	10002	65	2022-10-05	2002	3001
4	10004	110	2022-08-17	2009	3003
5	10007	948	2022-09-10	2005	3002
6	10005	2400	2022-07-27	2007	3001

Query | Query History

```
1  SELECT * FROM public.inventory
2
```

Data Output | Messages | Notifications

	item_code character varying (50)	item_name character varying (100)	item_price integer	item_total integer
1	2341	Promag Tablet	3000	100
2	2342	Hydro Coco 250ML	7000	20
3	2343	Nutrive Benecol 100ML	20000	30
4	2344	Blackmores Vit C 500Mg	95000	45
5	2345	Entrasol Gold 370G	90000	120

Hasil import tabel inventory dan customer_order.

Case Study

Rakamin Academy

Mengatur database di PostgreSQL ke Python lewat Jupyter Notebook

```
query = 'SELECT * FROM public.inventory;'
df = pd.read_sql_query(query, connection)
df
```

[6] ✓ 0.0s

	item_code	item_name	item_price	item_total
0	2341	Promag Tablet	3000	100
1	2342	Hydro Coco 250ML	7000	20
2	2343	Nutrive Benecol 100ML	20000	30
3	2344	Blackmores Vit C 500Mg	95000	45
4	2345	Entrasol Gold 370G	90000	120

```
query = 'SELECT * FROM public.customer_orders;'
df = pd.read_sql_query(query, connection)
df
```

[7] ✓ 0.0s

	order_no	purchase_amount	order_date	customer_id	salesman_id
0	10001	150	2022-10-05	2005	3002
1	10009	279	2022-09-10	2001	3005
2	10002	65	2022-10-05	2002	3001
3	10004	110	2022-08-17	2009	3003
4	10007	948	2022-09-10	2005	3002
5	10005	2400	2022-07-27	2007	3001

Hasil import tabel inventory dan customer_order di Jupyter Notebook menggunakan library Pandas.

Case Study

Rakamin Academy

Mengatur database di PostgreSQL ke Python lewat Jupyter Notebook

```
query_item_total_high = 'SELECT item_name FROM public.inventory ORDER BY item_total DESC LIMIT 1;'
num_high_item = pd.read_sql_query(query_item_total_high, connection)
num_high_item
```

[11] ✓ 0.0s

	item_name
0	Entrasol Gold 370G

Menampilkan Item_name yang memiliki
value tertinggi di Item_total

Case Study

Mengatur database di PostgreSQL ke Python lewat Jupyter Notebook

```
num_high_item = pd.read_sql_query(query_item_total_high, connection)
new_item_price = 96900
query_update_item_price = f"UPDATE public.inventory SET item_price = {new_item_price} WHERE item_name = '{num_high_item['item_name'][0]}';"
with connection.cursor() as cursor:
    cursor.execute(query_update_item_price)
connection.commit()
print("Item_price updated successfully.")
```

[12] ✓ 0.0s Python

```
Item_price updated successfully.
```

```
query = 'SELECT * FROM public.inventory;'
df = pd.read_sql_query(query, connection)
df
```

[13] ✓ 0.0s Python

	item_code	item_name	item_price	item_total
0	2341	Promag Tablet	3000	100
1	2342	Hydro Coco 250ML	7000	20
2	2343	Nutrive Benecol 100ML	20000	30
3	2344	Blackmores Vit C 500Mg	95000	45
4	2345	Entrasol Gold 370G	96900	120

Menjalankan perintah UPDATE untuk mengganti Item_price pada row Item_name Entrasol Gold 370G

Case Study

Rakamin Academy

Mengatur database di PostgreSQL ke Python lewat Jupyter Notebook

```
new_item = ("Entrostop", 2343, 50, 19000)
query_insert_item = "INSERT INTO public.inventory (item_name, item_code, item_total, item_price) VALUES (%s, %s, %s, %s);"
with connection.cursor() as cursor:
    cursor.execute(query_insert_item, new_item)
connection.commit()
print("Successfully inserted new items.")
```

✓ 0.0s

```
Successfully inserted new items.
```

```
query = 'SELECT * FROM public.inventory;'
df = pd.read_sql_query(query, connection)
df
```

✓ 0.0s

	item_code	item_name	item_price	item_total
0	2341	Promag Tablet	3000	100
1	2342	Hydro Coco 250ML	7000	20
2	2343	Nutrive Benecol 100ML	20000	30
3	2344	Blackmores Vit C 500Mg	95000	45
4	2345	Entrasol Gold 370G	96900	120
5	2343	Entrostop	19000	50

Memasukkan Item_name baru dengan
Item_code 2343 ke dalam tabel inventory

Case Study

Rakamin Academy

Mengatur database di PostgreSQL ke Python lewat Jupyter Notebook

Delete the Item_name that has the lowest number of Item_total

```
try:
    query_delete_lowest_item = "DELETE FROM public.inventory WHERE item_total = (SELECT MIN(item_total) FROM public.inventory);"

    with connection.cursor() as cursor:
        cursor.execute(query_delete_lowest_item)

    connection.commit()
    print("Successfully deleted item with lowest Item_total value.")

except psycopg2.Error as e:
    connection.rollback()  # Rollback the transaction to clear the error state
    print(f"Error: {e}")
```

[10] ✓ 0.0s

```
Successfully deleted item with lowest Item_total value.
```

```
query = 'SELECT * FROM public.inventory;'
df = pd.read_sql_query(query, connection)
df
```

[11] ✓ 0.0s

	item_code	item_name	item_price	item_total
0	2341	Promag Tablet	3000	100
1	2343	Nutrive Benecol 100ML	20000	30
2	2344	Blackmores Vit C 500Mg	95000	45
3	2345	Entrasol Gold 370G	96900	120
4	2343	Entrostop	19000	50

Menghapus Item_name yang memiliki value Item_total paling rendah

Case Study

Rakamin Academy

Mengatur database di PostgreSQL ke Python lewat Jupyter Notebook

```
query_display = "SELECT * FROM customer_orders WHERE (purchase_amount < 100 OR (order_date < '2022-08-25' AND customer_id <= 2001));"
```

[19] ✓ 0.0s Python

```
dfdisplay = pd.read_sql_query(query_display, connection)
dfdisplay
```

[21] ✓ 0.0s Python

	order_no	purchase_amount	order_date	customer_id	salesman_id
0	10002	65	2022-10-05	2002	3001

Challenge:

Create a Query to display all customer orders where purchase amount is less than 100 or exclude those orders which order date is on or greater than 25 Aug 2022 and customer id is above 2001.

Case Study

Please explain what is wrong with this picture and give the best solution for this case.

Pada tabel berikut, kolom pada ddlPlant seharusnya tidak boleh diisi dengan nilai NULL. Akan tetapi, ddlPlant boleh diisi data yang duplikat pada kolom tersebut dan bernilai beda pada kolom lain karena secara logika, mungkin ada Plant di lokasi yang sama.

Case Study

Please explain what is wrong with this picture and give the best solution for this case.

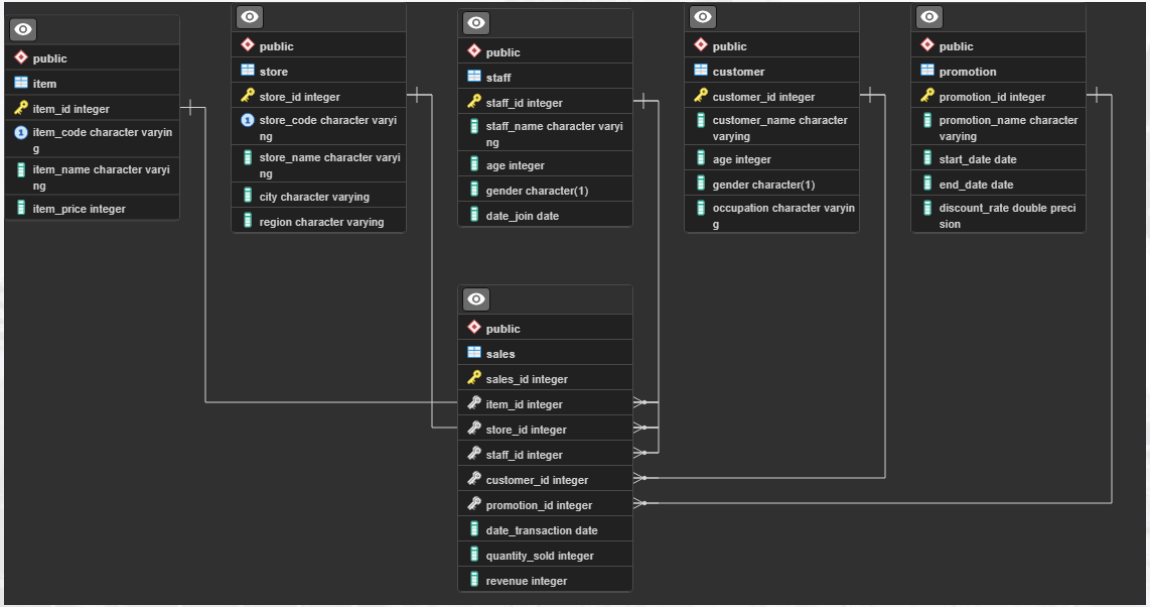
Dengan begitu, restriction kolom tersebut harus diganti dari NULL menjadi NOT NULL dengan query:

ALTER TABLE _FormN_168
ALTER COLUMN ddlPlant VARCHAR NOT NULL;

Case Study

Rakamin Academy

Create a simple star schema for KALBE database consist of 1 Fact and 5 Dimensions using Physical Data Model Theory.



SQL Query here
Table description here

Link Github

https://github.com/FawwazN/data-engineer-kalbe

Rakamin Academy

Video Presentation

https://drive.google.com/file/d/1udEwoA0gVV63lnY0fdYDNvCXvsqqCW1W/view?usp=sharing

Thank You