



# LEARNING PROGRESS REVIEW

# 4

OUR MINI PROJECT, LINUX &  
UNIX ADMINISTRATION, DOCKER

---

FAST TRACK DATA ENGINEER SCHOLARSHIP  
BY  
DIGITAL SKOLA



# GARAP RENDANG TEAM



- Suhendar Aditya
- Chandra Trio Pamungkas
- Aryo Putra
- Kurniaman Andreas Zega
- Ahmad Fayyadh

GARAP RENDANG

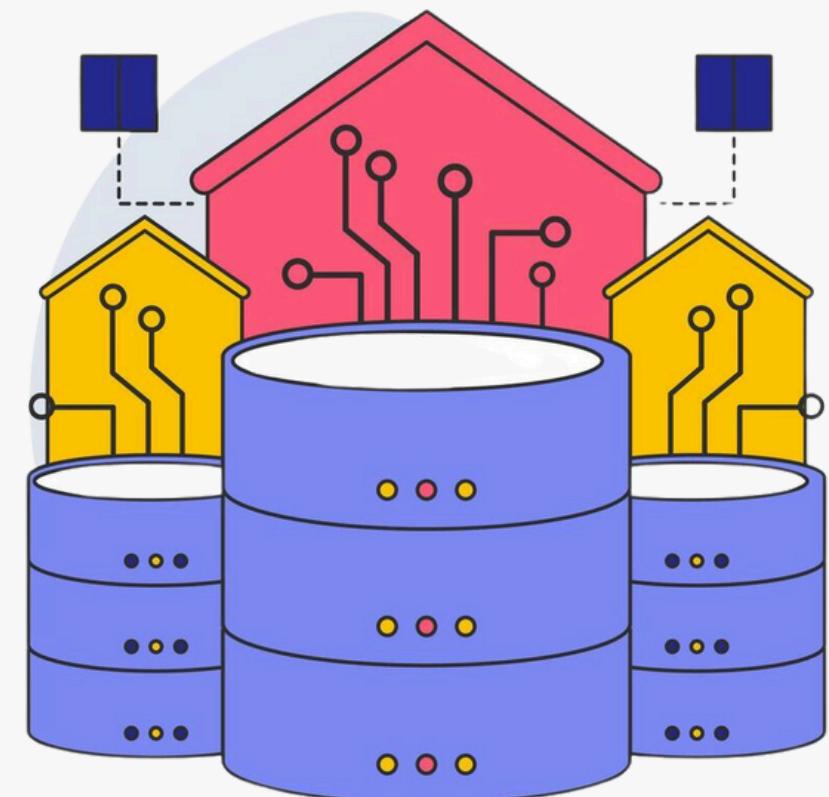


Grup Andalan Rakitan Pipeline  
Record Engineer Ngolah Data  
Anti Gagal



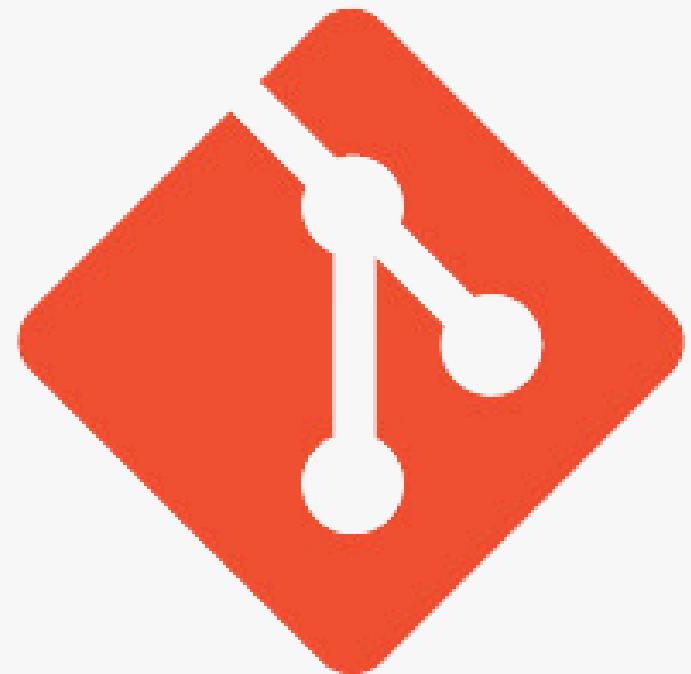
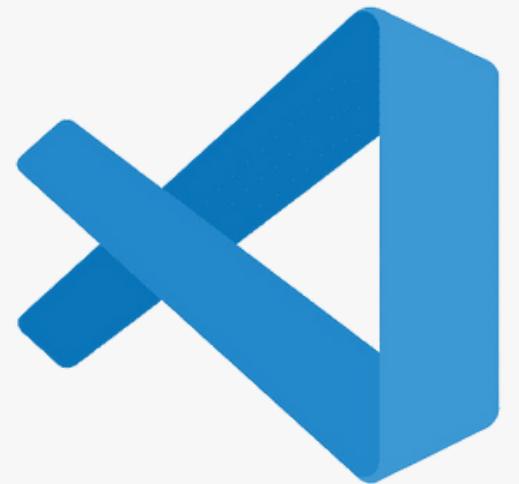
# MINI PROJECT

## DATA INGESTION FROM DATA SOURCE TO DATA WAREHOUSE

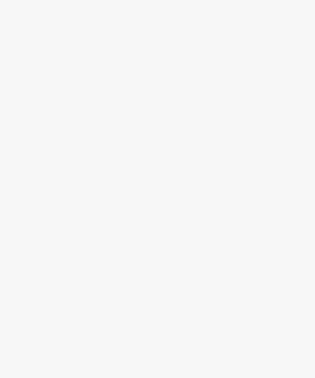




# Tools



Source: [pragmaticengineer.com](http://pragmaticengineer.com)



# Background Project

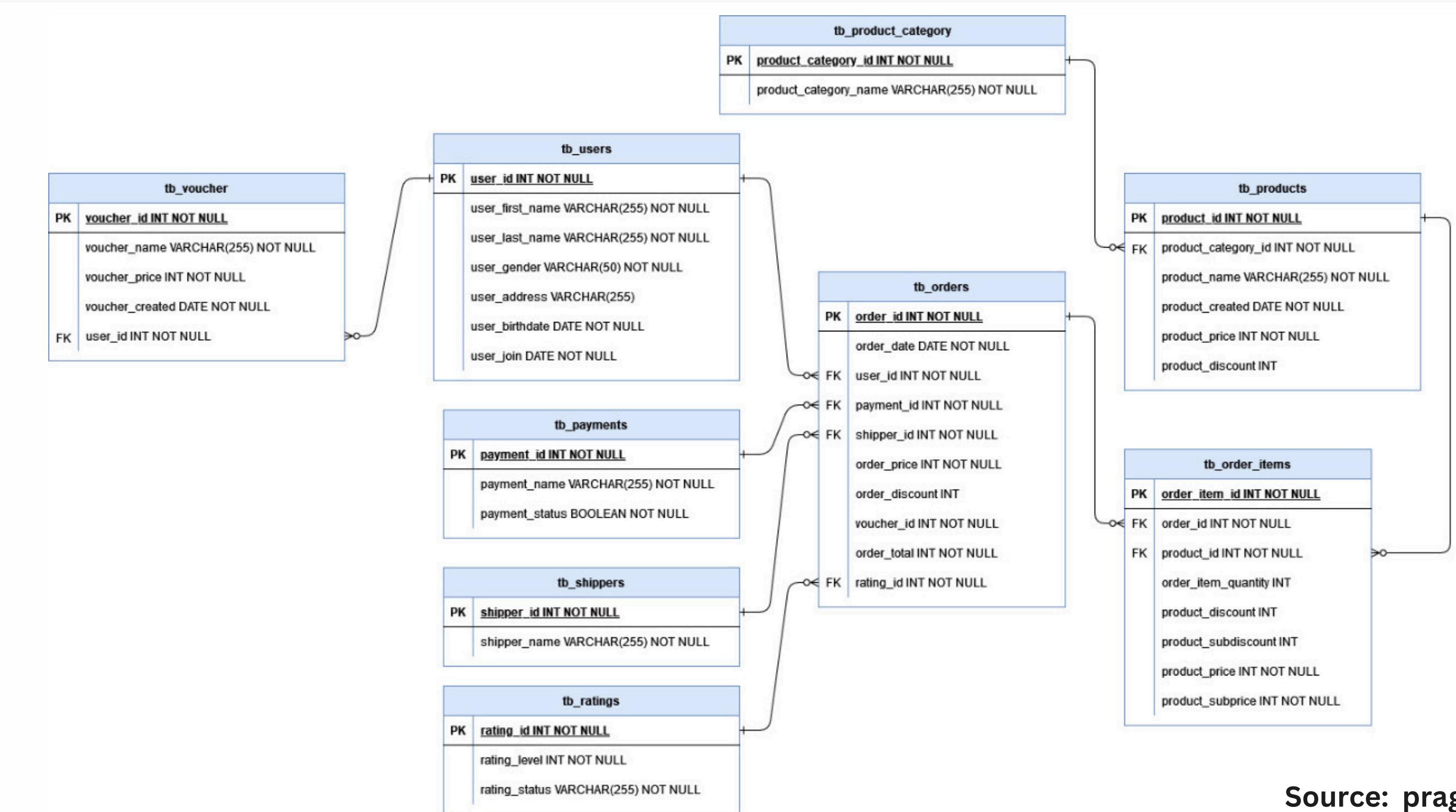


Tim data analyst membutuhkan tabel untuk membuat dashboard terkait detail order dari data source yang ada di database production marketplace. Anda sebagai data engineer di minta untuk membuat script data migrasi dari tabel source tersebut ke data warehouse, sehingga tim data analyst bisa menggunakan table dari data warehouse tanpa membebankan database production marketplace.

Source: [pragmaticengineer.com](http://pragmaticengineer.com)



# Schema



Source: [pragmaticengineer.com](http://pragmaticengineer.com)



# How we do it?



## 1. Install the requirements

The screenshot shows a code editor interface with the following details:

- EXPLORER** sidebar: FTDE-PROJECT1-GARAPRE... (expanded), query (expanded), dwh\_design.sql, query.sql, .gitignore, config.json, connection.py, main.py, requirements.txt.
- File list: requirements.txt, dwh\_design.sql, query.sql, connection.py, .gitignore, config.json.
- Content of requirements.txt:

```
1 psycopg2-binary
2 SQLAlchemy
3 sqlparse
4 pandas
```

Install menggunakan “pip/conda install -r requirements.txt”. jangan lupa untuk membuat virtual environment (venv) terlebih dahulu agar lebih mudah dan tertata dengan requirement mendatang di project lainnya.

Source: [pragmaticengineer.com](https://pragmaticengineer.com)



# How we do it?

## 2. Create connection.py

```
EXPLORER ... requirements.txt dwh_design.sql query.sql connection.py .gitignore config.json  
> OPEN EDITORS  
FTDE-P... [+] [-] ⌂ ⌂ connection.py  
query dwh_design.sql query.sql .gitignore config.json  
connection.py main.py requirements.txt  
connection.py  
1 import os  
2 import json  
3 import psycopg2  
4 from sqlalchemy import create_engine  
5  
6 # func config  
7 def config(connection_db):  
8     path = os.getcwd()  
9     with open(os.path.join(path, 'config.json')) as file:  
10         conf = json.load(file)[connection_db]  
11         return conf  
12  
13 # func get_conn  
14 def get_conn(conf, name_conn):  
15     try:  
16         conn = psycopg2.connect(  
17             host=conf['host'],  
18             database=conf['db'],  
19             user=conf['user'],  
20             password=conf['password'],  
21             port=conf['port'])  
22     )  
23     print(f"[INFO] succes connect postgres {name_conn}")  
24     engine=create_engine(  
25         "postgresql+psycopg2://{}:{}@{}:{}/{}".format(  
26             conf['user'],  
27             conf['password'],  
28             conf['host'],  
29             conf['port'],  
30             conf['db'])  
31     ))  
32     return conn, engine  
33 except Exception as e:  
34     print(f"[INFO] can't success connect to postgres {name_conn}")  
35     print(str(e))
```

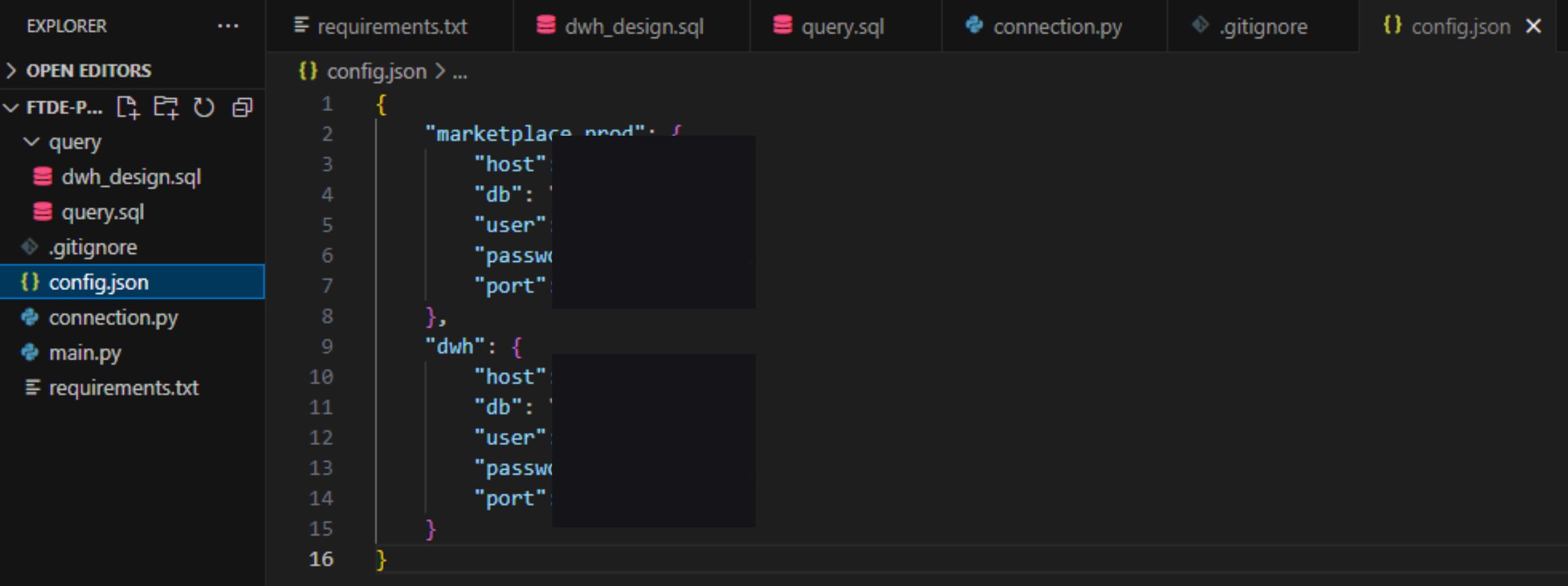
Membuat “connection .py” dengan detail user, password, host, port, dan db yang terdapat pada config.json.

Source: [pragmaticengineer.com](http://pragmaticengineer.com)



# How we do it?

## 3. Create config.json



```
1  {
2      "marketplace_prod": {
3          "host": "127.0.0.1",
4          "db": "marketplace",
5          "user": "root",
6          "password": "root",
7          "port": 3306
8      },
9      "dwh": {
10         "host": "127.0.0.1",
11         "db": "dwh",
12         "user": "root",
13         "password": "root",
14         "port": 3306
15     }
16 }
```

Berisi detail dari source dan destination, jangan lupa untuk memasukan file ini kedalam .gitignore agar tidak ter push kedalam repository github.

Source: [pragmaticengineer.com](http://pragmaticengineer.com)



# How we do it?

## 4. Create .gitignore

```
EXPLORER      ...
OPEN EDITORS
FTDE-P...
query
dwh_design.sql
query.sql
.gitignore
config.json
connection.py
main.py
requirements.txt

.gitignore
153
154 # Cython debug symbols
155 cython_debug/
156
157 # PyCharm
158 # JetBrains specific template is maintained in a separate JetBrains.gitignore that can
159 # be found at https://github.com/github/gitignore/blob/main/Global/JetBrains.gitignore
160 # and can be added to the global gitignore or merged into this file. For a more nuclear
161 # option (not recommended) you can uncomment the following to ignore the entire idea folder.
162 #.idea/
163 config.json
```

Lengkapnya dapat dicari dengan keyword “python gitignore”.

Source: [pragmaticengineer.com](http://pragmaticengineer.com)



# How we do it?



## 5. Create query folder and data warehouse design file

The screenshot shows a code editor interface with the following details:

- EXPLORER** sidebar:
  - OPEN EDITORS: FTDE-P...
  - query folder containing:
    - dwh\_design.sql (selected)
    - query.sql
    - .gitignore
    - config.json
    - connection.py
    - main.py
    - requirements.txt
- EDITOR TABS**: requirements.txt, dwh\_design.sql (active), query.sql, connection.py, .gitignore, config.json
- Code Content (dwh\_design.sql):**

```
1 DROP TABLE IF EXISTS dim_orders;
2 CREATE TABLE dim_orders (
3     order_id INT NOT NULL,
4     order_date DATE NOT NULL,
5     user_id INT NOT NULL,
6     payment_name VARCHAR(255),
7     shipper_name VARCHAR(255),
8     order_price INT,
9     order_discount INT,
10    voucher_name VARCHAR(255),
11    voucher_price INT,
12    order_total INT,
13    rating_status VARCHAR(255)
14 );
```

Membuat table pada data warehouse dengan detail column dari table lainnya pada source.

Source: [pragmaticengineer.com](http://pragmaticengineer.com)



# How we do it?

## 6. Create query file

The screenshot shows a code editor interface with the following details:

- EXPLORER** sidebar:
  - OPEN EDITORS**: FTDE-P...
  - query** folder:
    - dwh\_design.sql
    - query.sql** (selected)
  - .gitignore
  - config.json
  - connection.py
  - main.py
  - requirements.txt
- query.sql** content:

```
1 SELECT order_id,
2         order_date,
3         a.user_id,
4         b.payment_name,
5         c.shipper_name,
6         order_price,
7         order_discount,
8         d.voucher_name,
9         d.voucher_price,
10        order_total,
11        e.rating_status
12   FROM public.tb_orders a
13      LEFT JOIN public.tb_payments b ON a.payment_id = b.payment_id
14      LEFT JOIN public.tb_shippers c ON a.shipper_id = c.shipper_id
15      LEFT JOIN public.tb_vouchers d ON a.voucher_id = d.voucher_id
16      LEFT JOIN public.tb_ratings e ON a.rating_id = e.rating_id ;
```

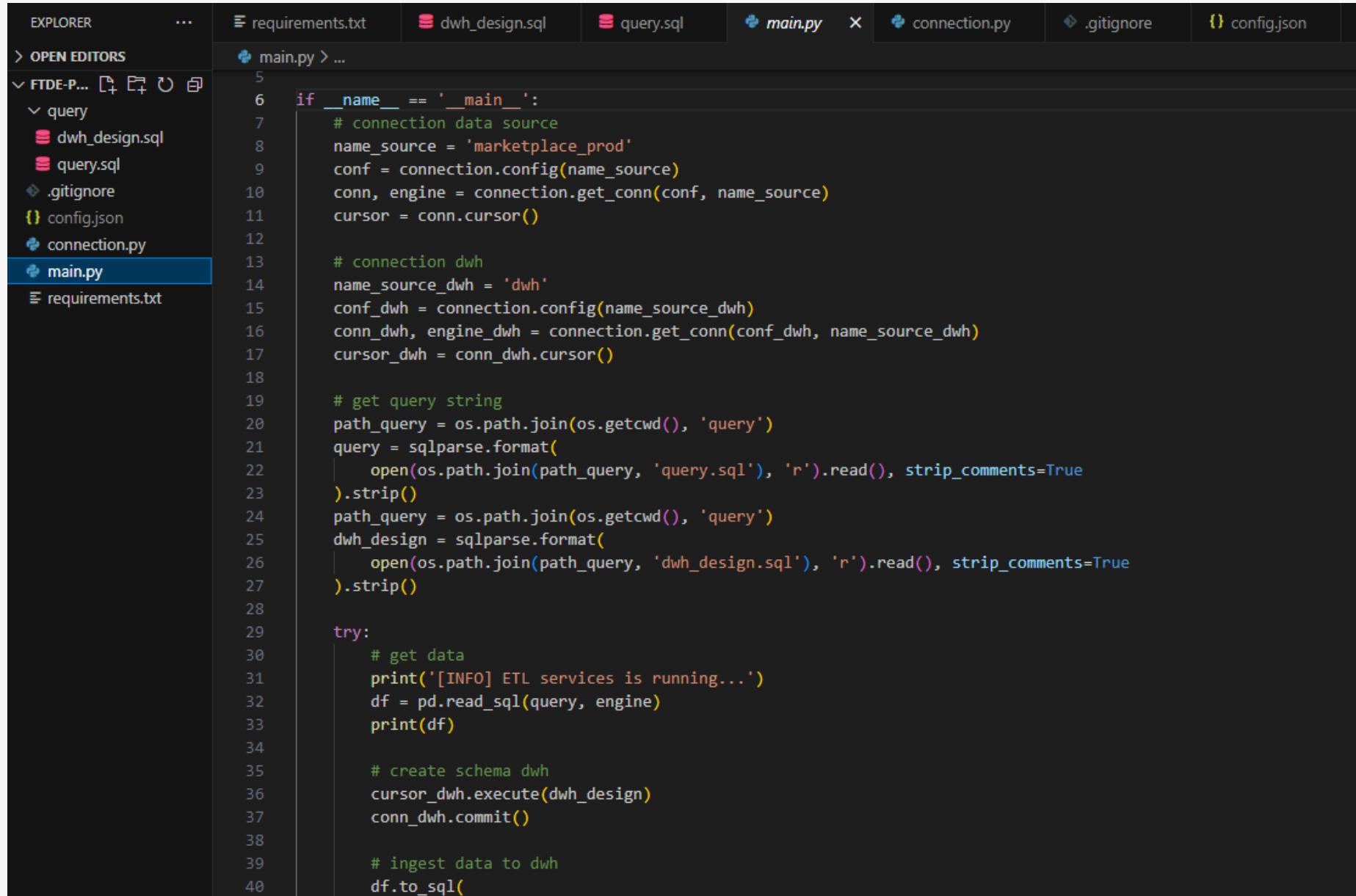
Membuat query untuk mengambil data dari beberapa table berbeda dengan ‘left join’ untuk diteruskan ke data warehouse.

Source: [pragmaticengineer.com](http://pragmaticengineer.com)



# How we do it?

## 7. Create main.py file



```
EXPLORER ... requirements.txt dwh_design.sql query.sql main.py connection.py .gitignore config.json  
OPEN EDITORS  
FTDE-P...  
query dwh_design.sql query.sql .gitignore config.json connection.py main.py requirements.txt  
5  
6 if __name__ == '__main__':  
7     # connection data source  
8     name_source = 'marketplace_prod'  
9     conf = connection.config(name_source)  
10    conn, engine = connection.get_conn(conf, name_source)  
11    cursor = conn.cursor()  
12  
13    # connection dwh  
14    name_source_dwh = 'dwh'  
15    conf_dwh = connection.config(name_source_dwh)  
16    conn_dwh, engine_dwh = connection.get_conn(conf_dwh, name_source_dwh)  
17    cursor_dwh = conn_dwh.cursor()  
18  
19    # get query string  
20    path_query = os.path.join(os.getcwd(), 'query')  
21    query = sqlparse.format(  
22        open(os.path.join(path_query, 'query.sql'), 'r').read(), strip_comments=True  
23    ).strip()  
24    path_query = os.path.join(os.getcwd(), 'query')  
25    dwh_design = sqlparse.format(  
26        open(os.path.join(path_query, 'dwh_design.sql'), 'r').read(), strip_comments=True  
27    ).strip()  
28  
29 try:  
30     # get data  
31     print('[INFO] ETL services is running...')  
32     df = pd.read_sql(query, engine)  
33     print(df)  
34  
35     # create schema dwh  
36     cursor_dwh.execute(dwh_design)  
37     conn_dwh.commit()  
38  
39     # ingest data to dwh  
40     df.to_sql(
```

Membuat query utama untuk menjalankan semuanya secara berurutan.

Source: [pragmaticengineer.com](http://pragmaticengineer.com)



# Results

From this

```
*<dwh> Script-9 ×  
▶ ┌ select * from public.dim_orders;  
▶ +  
dim_orders 1 ×  
⠄ select * from public.dim_orders | Enter a SQL expression to filter results (use Ctrl+Space)  
Grid ① 123 order_id ▾ ⏷ order_date ▾ 123 user_id ▾ ⏷ A-Z payment_name ▾ ⏷ A-Z shipper_name ▾ ⏷ 123 order_price ▾ ⏷ 123 order_discount ▾ ⏷ A-Z voucher_name ▾ ⏷ 123 voucher_price ▾ ⏷ 123 order_total ▾ ⏷ A-Z rating_status ▾  
Text
```

To this

```
*<dwh> Script-9 ×  
▶ ┌ select * from public.dim_orders;  
▶ +  
dim_orders 1 ×  
⠄ select * from public.dim_orders | Enter a SQL expression to filter results (use Ctrl+Space)  
Grid ① 123 order_id ▾ ⏷ order_date ▾ 123 user_id ▾ ⏷ A-Z payment_name ▾ ⏷ A-Z shipper_name ▾ ⏷ 123 order_price ▾ ⏷ 123 order_discount ▾ ⏷ A-Z voucher_name ▾ ⏷ 123 voucher_price ▾ ⏷ 123 order_total ▾ ⏷ A-Z rating_status ▾  
Text  
1 1,110,001 2022-01-20 100,101 - Debit JNE Express 250,000 15,000 New User 5,000 230,000 Medium Impact  
2 1,110,002 2022-01-29 100,102 - Debit JNE Express 620,000 40,000 New User 5,000 575,000 Medium Impact  
3 1,110,003 2022-02-13 100,103 - Credit JNE Express 6,000,000 1,000,000 New User 5,000 4,995,000 Very Low Impact  
4 1,110,004 2022-03-06 100,102 - Wallet JNE Express 3,150,000 45,000 [NULL] [NULL] 3,105,000 High Impact  
5 1,110,005 2022-04-28 100,105 - Debit Sicepat Express 4,000,000 1,000,000 New User 5,000 2,995,000 Very Low Impact  
6 1,110,006 2022-05-09 100,103 - Debit Sicepat Express 4,500,000 1,030,000 [NULL] [NULL] 3,470,000 Medium High Impact  
7 1,110,007 2022-05-21 100,106 - Debit JNE Express 870,000 25,000 [NULL] [NULL] 845,000 High Impact  
8 1,110,008 2022-06-02 100,108 - Credit Sicepat Express 2,000,000 0 New User 5,000 1,995,000 Medium High Impact  
9 1,110,009 2022-06-23 100,103 - Credit Lazada Express 2,000,000 0 [NULL] [NULL] 2,000,000 High Impact  
10 1,110,010 2022-07-01 100,102 - Credit Lazada Express 1,050,000 45,000 [NULL] [NULL] 1,005,000 Low Impact  
11 1,110,011 2022-07-21 100,110 - Wallet Sicepat Express 550,000 15,000 [NULL] [NULL] 535,000 High Impact  
12 1,110,012 2022-07-30 100,110 - Debit JNE Express 490,000 35,000 Body Soap Promo 10,000 445,000 Medium High Impact
```

Source: [pragmaticengineer.com](http://pragmaticengineer.com)



# LINUX

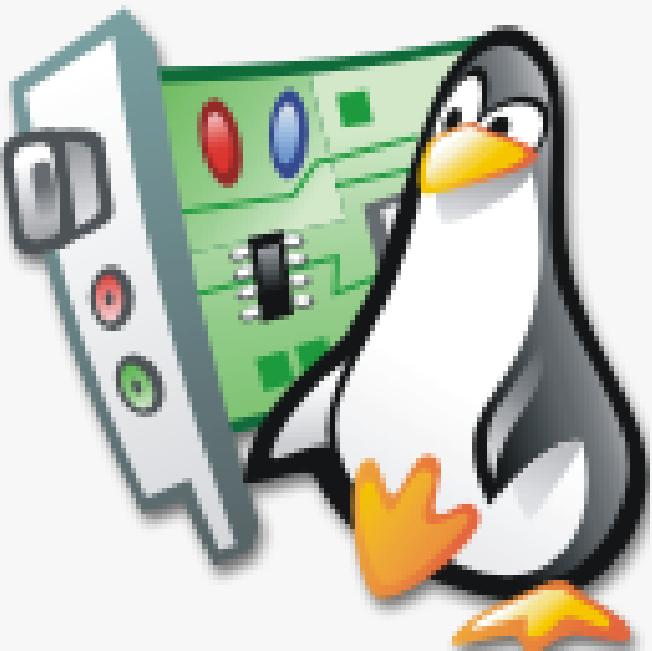
“Sistem operasi (OS) komputer yang bersifat open source dan gratis”

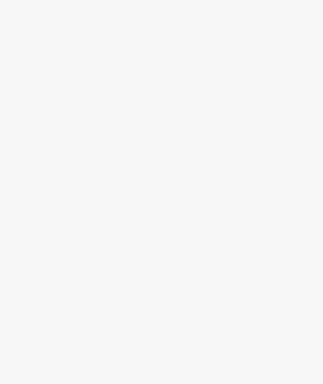




# ALASAN MEMPELAJARI LINUX

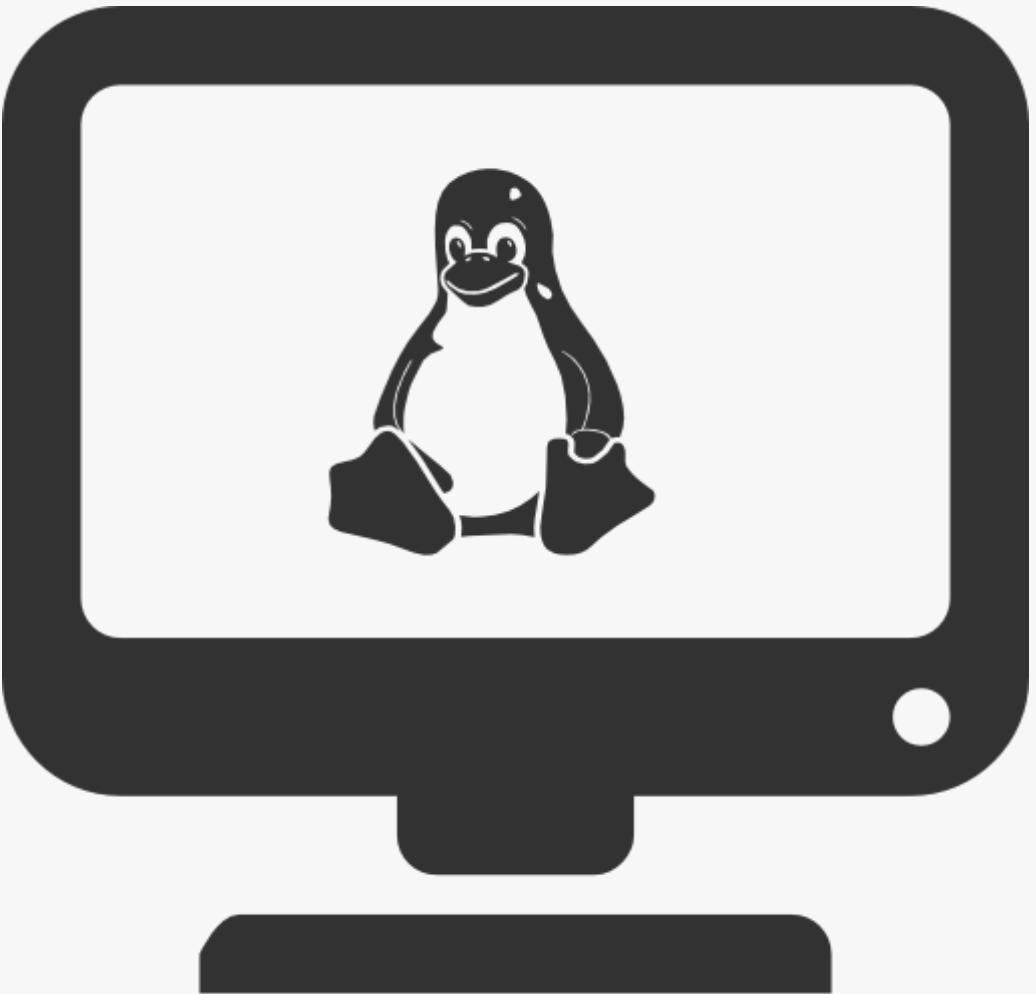
- **Open Source:** Bebas digunakan dan dimodifikasi.
- **Gratis:** Tidak memerlukan biaya lisensi.
- **Keamanan Tinggi:** Tingkat keamanan lebih baik dibandingkan sistem lainnya.
- **Dominasi Cloud:** 90% sistem cloud berbasis Linux.
- **Kompatibilitas Alat Data Engineering:** Sebagian besar alat untuk data engineering berjalan di Linux.





# TENTANG LINUX

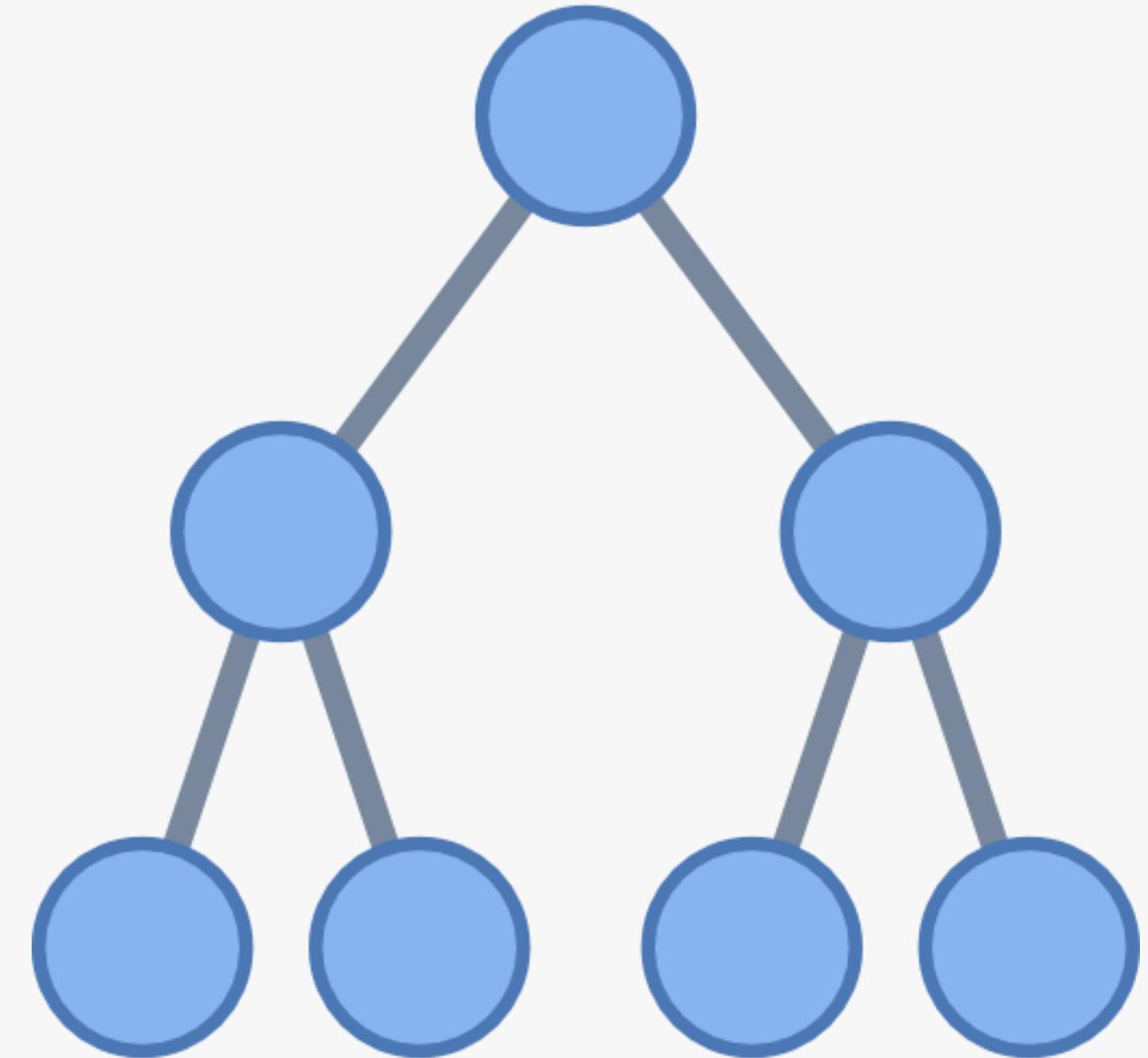
- **Sejarah:** diciptakan oleh Linus Torvalds, terinspirasi dari kernel Minix milik Andrew Tanenbaum.
- **Distro Linux:** sistem operasi berbasis kernel Linux, seperti Ubuntu, Fedora, dll., mencakup aplikasi seperti server web, database, dan desktop environment.





# STRUKTUR DIREKTORI LINUX

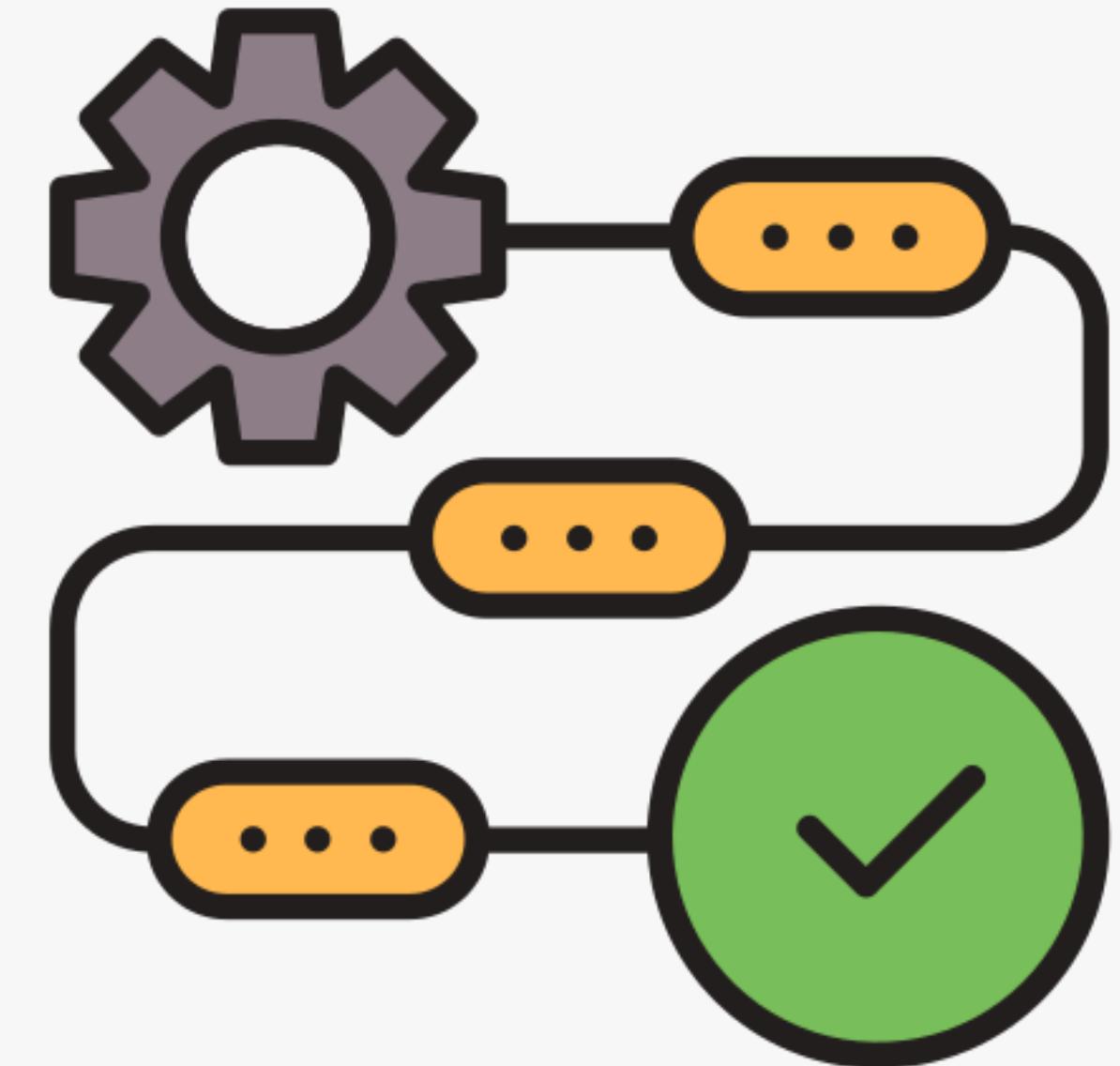
- **/ (Root)**: Direktori utama.
- **/bin**: File biner untuk perintah dasar shell.
- **/etc**: File konfigurasi sistem inti.
- **/home**: Direktori pribadi pengguna.
- **/var**: File runtime sistem, log, dan data lainnya.





# PROSES DI LINUX

- **Foreground Process:** Proses interaktif yang dijalankan oleh pengguna.
- **Background Process:** Proses non-interaktif yang berjalan otomatis.
- **Manajemen Proses:** Menggunakan perintah seperti ps dan top untuk memonitor penggunaan sumber daya.

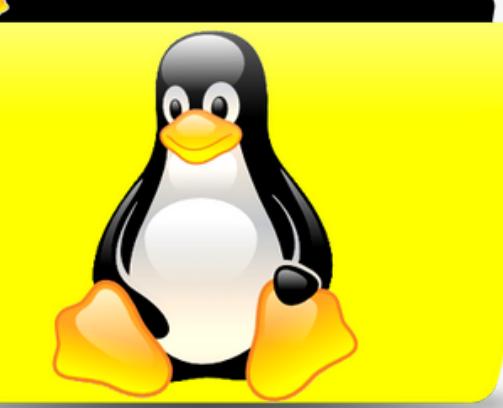




# CRONTAB

- **Cron Daemon:** Menjadwalkan tugas otomatis (cronjobs) pada waktu tertentu.
- **Perintah Crontab:**
  - crontab -e: Membuat atau mengubah cronjobs.
  - crontab -l: Melihat daftar cronjobs.
  - crontab -r: Menghapus cronjobs





LINUX



# KESIMPULAN

“ **Linux** merupakan sistem operasi yang fleksibel, aman, dan dominan di lingkungan cloud dan data engineering, sehingga penting untuk dipelajari oleh administrator sistem dan profesional teknologi.”





**Unix** dan **Linux** merupakan dua sistem operasi yang berbeda namun sering dibandingkan karena kemiripannya,

**Unix** merupakan sistem operasi komersial sedangkan **Linux** sistem operasi open source yang memiliki basis kernel Unix



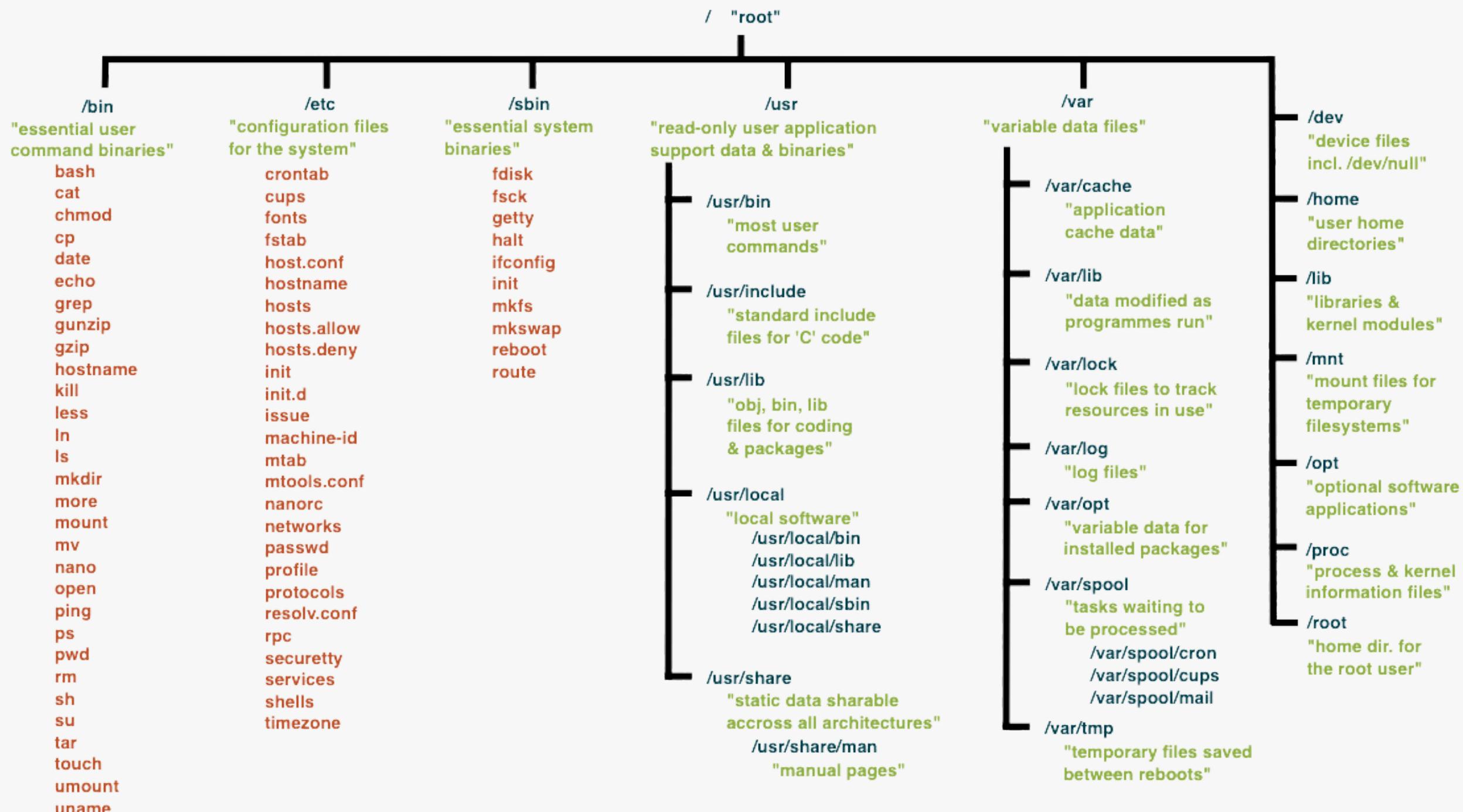
# UNIX SYSTEM ADMINISTRATOR

Bertugas mengelola sistem operasi unix, beberapa tugas nya antara lain :

- Mengatur lingkungan kerja (.env) yang menggunakan sistem operasi Unix multi-user
- Melakukan tugas manajemen sistem, seperti menyimpan dan melindungi data pengguna, serta menjaga keamanan
- Mengembangkan manual dan proses auto pada panduan teknis untuk perusahaan



## UNIX SYSTEM ADMINISTRATOR - FILESYSTEM HIERARCHY



(RCC-UCHICAGO, 2024)



# UNIX SYSTEM ADMINISTRATOR - USER MANAGE

Instruksi	Keterangan	Contoh
useradd	Menambahkan User	useradd [options] [User_name]
userdel	Menghapus entitas user	userdel [options] username
usermod	Memodifikasi data user	sudo usermod -c "[message]" test_user
passwd	Merubah password user	passwd [options] [username]
groupadd	Menambah sebuah group	groupadd [options] group_name



# UNIX SYSTEM ADMINISTRATOR - ENVIRONMENT

Instruksi	Keterangan	Contoh
env	Menjalankan program pada lingkungan tertentu	env [OPTION]... [-][NAME=VALUE]... [COMMAND [ARG]...]
printenv	Melihat environment (.env) yang ingin dilihat	printenv [name]
set	memasukan nilai ke dalam sebuah environment	set [options] [arguments]
unset	hapus environtment variable	unset var_name_here unset [options] var
export	memungkinkan memperbarui dan menyebarkan nilai variabel lingkungan ke sesi saat ini dan proses turunan yang dihasilkan	export [-f] [-n] [name[=value] ...] or export -p

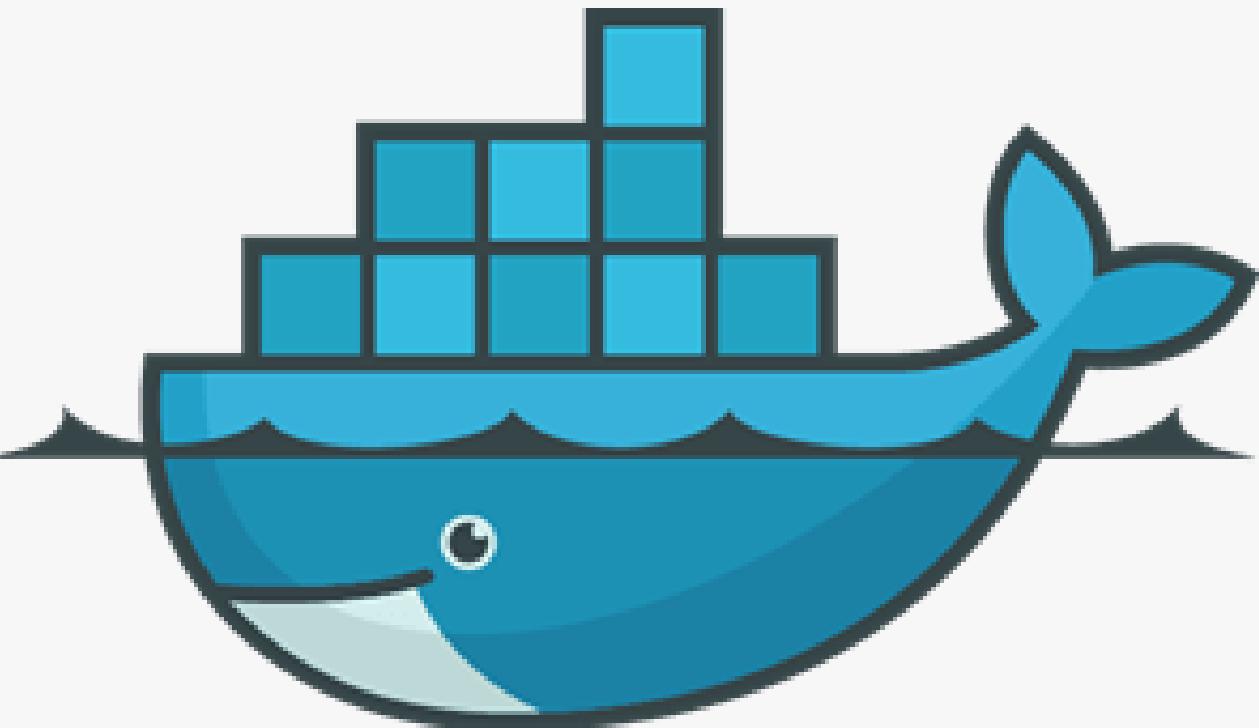


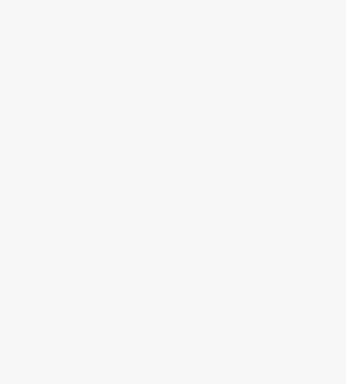
# KESIMPULAN

“ perintah **Unix** dapat menjadi alat yang ampuh untuk menavigasi sistem berkas, mengelola berkas, dan membangun sistem informasi”

# DOCKER

“Platform untuk mengembangkan, mengirim dan menjalankan aplikasi pada lingkungan yang terisolasi”





# ARSITEKTUR DOCKER

- **Docker Client:** Aplikasi yang digunakan untuk berinteraksi dan memberi perintah ke docker daemon.
- **Docker Daemon:** Aplikasi utama yang dilajangkan di atas operating system (OS) berfungsi mengelola image, container, volume, dan networks.
- **Docker Registry:** Tempat penyimpan docker images yang bisa disimpan secara privasi atau publik

Source: [pragmaticengineer.com](http://pragmaticengineer.com)



# KOMPONEN DOCKER

- **Images:** Sebuah template intruksi untuk membuat container
- **Containers:** Aplikasi yang dijalankan berdasarkan image yang digunakan pada lingkungan yang terisolasi.
- **Volumes:** Tempat penyimpanan yang dikelola oleh docker dan dapat digunakan oleh container
- **Networks:** Jaringan virtual yang dapat menghubungkan satu container dengan container lainnya.

Source: [pragmaticengineer.com](http://pragmaticengineer.com)



# DOCKERFILE

File yang berisi perintah-perintah yang dapat dijalankan untuk membuat image.

```
FROM python:3.8-slim-buster
WORKDIR /app
COPY requirements.txt requirements.txt
RUN pip install -r requirements.txt
COPY ..
CMD ["python", "app.py"]
```

Source: [pragmaticengineer.com](https://pragmaticengineer.com)



# DOCKER COMPOSE

Alat untuk mengatur dan menjalankan beberapa container sekaligus di docker berdasarkan apa yang telah didefinisikan pada file berformat YAML

**services:**

**web:**

**image: python:3.8-slim-buster**

**container\_name: flask\_app**

**working\_dir: /app**

**volumes:**

**- ./app**

**ports:**

**- "5000:5000"**

**command: >**

**sh -c "pip install -r requirements.txt && python app.py"**

Source: [pragmaticengineer.com](http://pragmaticengineer.com)



# THANK YOU

---

ANY FEEDBACKS?