## 🧩 ****Project Title:****

**"Product Price Intelligence System using SQLite"**

## ✅ ****Project Goal:****

To build a professional, efficient, and maintainable database system that:

* Collects product pricing data from multiple retailers.
* Stores it in a structured format (SQLite).
* Retrieves the best price per product.
* Can be visualized and eventually integrated into a price-scouting application.

## 🛠️ ****Step 1: Project Setup****

### 1. Create a new folder/project in PyCharm

Name it: product\_price\_tracker

### 2. Inside that project, create the following files:

pgsql

CopyEdit

product\_price\_tracker/

├── main.py

├── database.py

├── schema.sql

├── insert\_data.py

├── README.md

## 📄 ****Step 2: Create**** schema.sql

This file defines your database structure (tables).

sql

CopyEdit

-- schema.sql

CREATE TABLE IF NOT EXISTS products (

product\_id INTEGER PRIMARY KEY AUTOINCREMENT,

name TEXT NOT NULL,

category TEXT,

brand TEXT

);

CREATE TABLE IF NOT EXISTS retailers (

retailer\_id INTEGER PRIMARY KEY AUTOINCREMENT,

name TEXT NOT NULL,

website TEXT

);

CREATE TABLE IF NOT EXISTS prices (

price\_id INTEGER PRIMARY KEY AUTOINCREMENT,

product\_id INTEGER,

retailer\_id INTEGER,

price REAL,

currency TEXT,

date\_collected TEXT,

FOREIGN KEY (product\_id) REFERENCES products(product\_id),

FOREIGN KEY (retailer\_id) REFERENCES retailers(retailer\_id)

);

## 🧠 ****Step 3: Create**** database.py

This script connects to the database and sets it up using the schema.

python

CopyEdit

# database.py

import sqlite3

def create\_connection(db\_file="pricing\_db.db"):

return sqlite3.connect(db\_file)

def initialize\_database():

conn = create\_connection()

with open("schema.sql", "r") as schema\_file:

conn.executescript(schema\_file.read())

conn.commit()

conn.close()

## 📥 ****Step 4: Create**** insert\_data.py

This script inserts some initial test data.

python

CopyEdit

# insert\_data.py

from database import create\_connection

from datetime import date

def insert\_sample\_data():

conn = create\_connection()

cursor = conn.cursor()

# Insert products

products = [

('Wireless Mouse', 'Electronics', 'Logitech'),

('Noise Cancelling Headphones', 'Audio', 'Sony'),

('Portable Charger', 'Accessories', 'Anker')

]

cursor.executemany("INSERT INTO products (name, category, brand) VALUES (?, ?, ?)", products)

# Insert retailers

retailers = [

('Amazon', 'https://www.amazon.com'),

('Best Buy', 'https://www.bestbuy.com'),

('Walmart', 'https://www.walmart.com')

]

cursor.executemany("INSERT INTO retailers (name, website) VALUES (?, ?)", retailers)

# Insert prices

prices = [

(1, 1, 25.99, 'USD', str(date.today())),

(1, 2, 27.49, 'USD', str(date.today())),

(2, 1, 199.99, 'USD', str(date.today())),

(3, 3, 39.99, 'USD', str(date.today()))

]

cursor.executemany(

"INSERT INTO prices (product\_id, retailer\_id, price, currency, date\_collected) VALUES (?, ?, ?, ?, ?)",

prices

)

conn.commit()

conn.close()

## 🚀 ****Step 5: Create**** main.py

This script ties everything together and queries the best prices.

python

CopyEdit

# main.py

from database import initialize\_database

from insert\_data import insert\_sample\_data

import sqlite3

def get\_best\_prices():

conn = sqlite3.connect("pricing\_db.db")

cursor = conn.cursor()

query = """

SELECT p.name, MIN(pr.price) as best\_price

FROM prices pr

JOIN products p ON p.product\_id = pr.product\_id

GROUP BY p.name

"""

cursor.execute(query)

results = cursor.fetchall()

print("\nBest Prices:\n")

for row in results:

print(f"Product: {row[0]} | Best Price: ${row[1]:.2f}")

conn.close()

if \_\_name\_\_ == "\_\_main\_\_":

initialize\_database()

insert\_sample\_data()

get\_best\_prices()

## 📝 ****README.md (Professional Portfolio Touch)****

markdown

CopyEdit

# Product Price Intelligence System

This is a SQLite-based system that stores and retrieves product pricing data from multiple online retailers. It supports price comparisons and can be extended for integration with real-world web scraping tools or e-commerce APIs.

### Features

- Structured SQLite schema

- Insert and retrieve product data

- Price comparison report (best price per product)

- Modular code with clean separation

### Tools Used

- Python 3

- SQLite

- PyCharm IDE

### How to Run

1. Clone or download this project.

2. Open it in PyCharm.

3. Run `main.py`.

### Author

Deepa Ponnusamy

[GitHub Portfolio](https://github.com/python-sql09)

## ✅ Next Steps:

* Let me know when you’ve created the folder and files.
* I’ll help you test, improve, and then optionally expand this into:
  + JSON/web scraping input
  + Power BI/Excel export
  + Visual dashboards using Python or Tableau

Ready to build this like a pro? Let’s go step by step — just say **“ready for step 1”** and we’ll begin.