SET 1

Q1: Design a database schema for a **Library Management System** using AI assistance.

- Use AI tools to generate SQL commands for creating tables: `Books`, `Members`, and `Loans` and execute in VS code
- Insert at least 3 records in each table
- Write a query to display all books currently issued to members.

Q2. Using AI, generate a Python script to clean a **sales dataset** by handling missing values and normalizing the "transaction_amount" column using Min-Max scaling and standard scalar method. Use VS Code with Github copilot or cursor AI

- Generate dataset using ChatGPT that helps to complete the below two taks
- Handle missing values
- Normalize "transaction_amount"

==========

SET 2

Q1. Using AI, create a database schema for a **Student Attendance Management System**.

- Generate `CREATE TABLE` queries for `Students`, `Courses`, and `Attendance` with random data that can full fill below task
- Write a query to find students with attendance below 75%.

Q2. Generate a Python data preprocessing script using AI to:

- Generate dataset using ChatGPT that helps to complete the below two tasks
- Fill missing values with column mean.
- Encode categorical data using one-hot encoding.
- Save the cleaned dataset as `processed_students.csv`.

==========

SET 3

Q1. Generate an AI-assisted database schema for an **Online Food Delivery Application**. Use VS code with Github copilot

- Include tables: `Restaurants`, `Customers`, `Orders`, `Menu_Items` with some random data.
- Write SQL queries to list all customers who ordered items costing above ₹500.

Q2. Use AI to convert a Python dictionary manipulation program into equivalent **JavaScript code**.

- Verify correctness by comparing outputs of both scripts.

==========

SET 4

Q1. Design an **E-commerce Product Inventory** schema using AI-assisted SQL generation.

- Include tables: `Products`, `Suppliers`, and `Stock`.
- Write SQL query to insert random data in each table
- Write an SQL query to find products with stock quantity less than 10.

Q2. Generate an AI-assisted data cleaning script to:

- Generate dataset using ChatGPT that helps to complete the below two tasks
- Remove duplicates, handle NaN values, and standardize column names.
- Demonstrate the result using a sample CSV file

==========

SET 5

Q1. With the help of AI, create and populate a database schema for a **Flight Booking System**.

- Include tables: `Flights`, `Passengers`, `Bookings`. Insert random data that helps to complete below task
- Write a query to list all passengers booked for flights on a specific date.

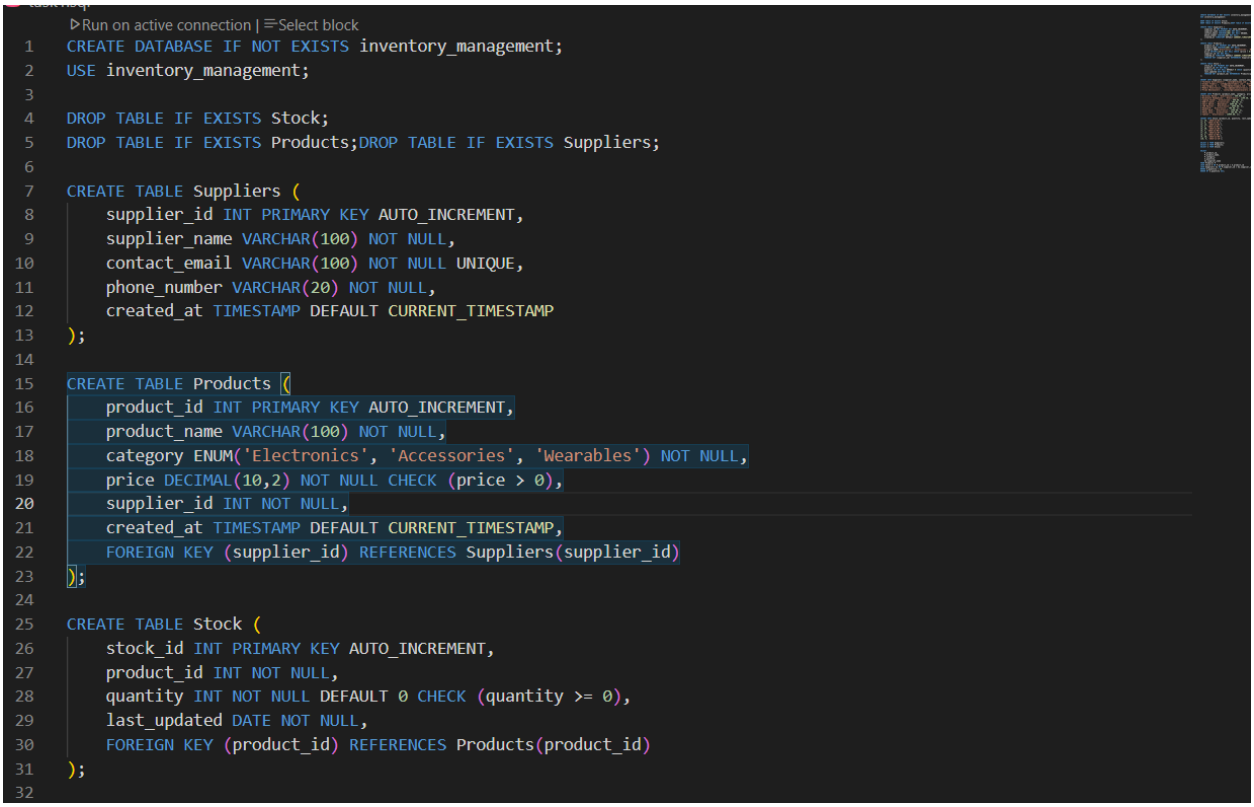
Q2. Use AI to convert a SQL query that retrieves employees with salary > 60,000 into equivalent **Python Pandas** code.

- Write SQL program to build table and insert data
- Ask chatGpt to generate data required data in .csv
- Write SQL query that retrieves employees with salary > 60,000 from SQL Table
- Write Python code that retrieves employees with salary > 60,000 from .csv file
- Execute both and compare results.

SET-4:

Question-1:

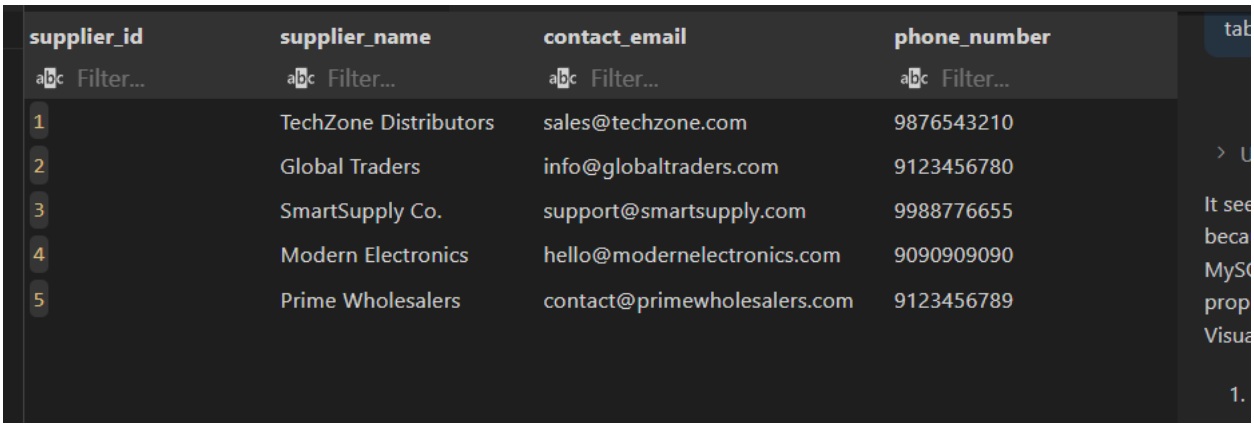
Code:



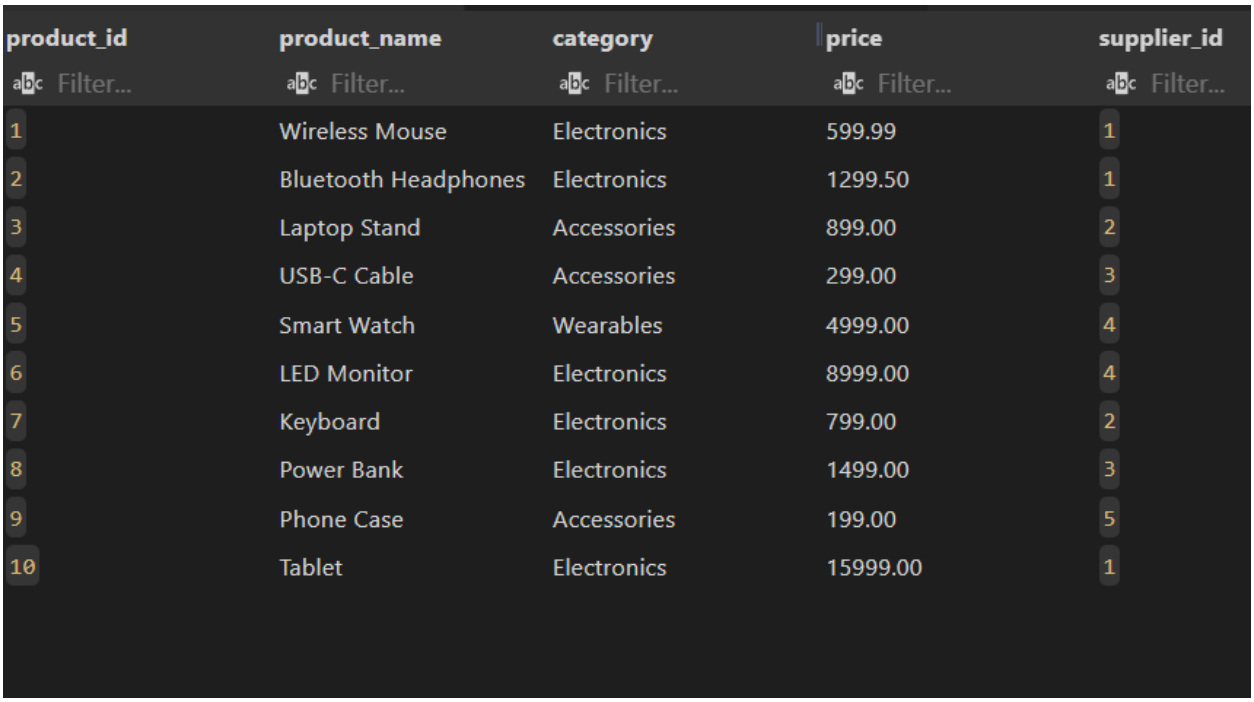
```sql
);

INSERT INTO Suppliers (supplier_name, contact_email, phone_number) VALUES
('TechZone Distributors', 'sales@techzone.com', '9876543210'),
('Global Traders', 'info@globaltraders.com', '9123456780'),
('SmartSupply Co.', 'support@smartsupply.com', '9988776655'),
('Modern Electronics', 'hello@modernelectronics.com', '9090909090'),
('Prime Wholesalers', 'contact@primewholesalers.com', '9123456789');

INSERT INTO Products (product_name, category, price, supplier_id) VALUES
('Wireless Mouse', 'Electronics', 599.99, 1),
('Bluetooth Headphones', 'Electronics', 1299.50, 1),
('Laptop Stand', 'Accessories', 899.00, 2),
('USB-C Cable', 'Accessories', 299.00, 3),
('Smart Watch', 'Wearables', 4999.00, 4),
('LED Monitor', 'Electronics', 8999.00, 4),
('Keyboard', 'Electronics', 799.00, 2),
('Power Bank', 'Electronics', 1499.00, 3),
('Phone Case', 'Accessories', 199.00, 5),
('Tablet', 'Electronics', 15999.00, 1);

INSERT INTO Stock (product_id, quantity, last_updated) VALUES
(1, 25, '2025-11-01'),
(2, 8, '2025-11-02'),
(3, 12, '2025-11-03'),
(4, 5, '2025-11-04'),
(5, 30, '2025-11-05'),
(6, 7, '2025-11-06'),
(7, 15, '2025-11-07'),
(8, 3, '2025-11-08'),
(9, 20, '2025-11-09'),
(10, 9, '2025-11-10');
```

```sql
SELECT * FROM Suppliers;
SELECT * FROM Products;
SELECT * FROM Stock;

SELECT
    P.product_id,
    P.product_name,
    P.category,
    S.quantity,
    Su.supplier_name
FROM Products P
JOIN Stock S ON P.product_id = S.product_id
JOIN Suppliers Su ON P.supplier_id = Su.supplier_id
WHERE S.quantity < 10
ORDER BY S.quantity ASC;
```

Output:



supplier_id	supplier_name	contact_email	phone_number
1	TechZone Distributors	sales@techzone.com	9876543210
2	Global Traders	info@globaltraders.com	9123456780
3	SmartSupply Co.	support@smartsupply.com	9988776655
4	Modern Electronics	hello@modernelectronics.com	9090909090
5	Prime Wholesalers	contact@primewholesalers.com	9123456789



product_id	product_name	category	price	supplier_id
1	Wireless Mouse	Electronics	599.99	1
2	Bluetooth Headphones	Electronics	1299.50	1
3	Laptop Stand	Accessories	899.00	2
4	USB-C Cable	Accessories	299.00	3
5	Smart Watch	Wearables	4999.00	4
6	LED Monitor	Electronics	8999.00	4
7	Keyboard	Electronics	799.00	2
8	Power Bank	Electronics	1499.00	3
9	Phone Case	Accessories	199.00	5
10	Tablet	Electronics	15999.00	1

stock_id	product_id	quantity	last_updated
Filter...	Filter...	Filter...	Filter...
1	1	25	2025-11-01
2	2	8	2025-11-02
3	3	12	2025-11-03
4	4	5	2025-11-04
5	5	30	2025-11-05
6	6	7	2025-11-06
7	7	15	2025-11-07
8	8	3	2025-11-08
9	9	20	2025-11-09
10	10	9	2025-11-10

product_id	product_name	category	quantity	supplier_name
Filter...	Filter...	Filter...	Filter...	Filter...
8	Power Bank	Electronics	3	SmartSupply Co.
4	USB-C Cable	Accessories	5	SmartSupply Co.
6	LED Monitor	Electronics	7	Modern Electronics
2	Bluetooth Headphones	Electronics	8	TechZone Distributors
10	Tablet	Electronics	9	TechZone Distributors

Question-2:

Code:

task-2.py > create_sample_dataset

```python
import pandas as pd
import numpy as np

# Generate sample dirty dataset
def create_sample_dataset():
    # Create sample data with duplicates and NaN values
    data = {
        'Customer ID': [1, 2, 2, 3, 4, 5, None, 5],
        'First_Name': ['John', 'Jane', 'Jane', 'Bob', None, 'Alice', 'Tom', 'Alice'],
        'Last name': ['Smith', 'Doe', 'Doe', 'Johnson', 'Brown', 'Wilson', 'Clark', 'Wilson'],
        'Age': [25, 30, 30, None, 45, 35, 28, 35],
        'SALARY': [50000, 60000, 60000, 75000, None, 65000, 55000, 65000]
    }

    df = pd.DataFrame(data)
    # Save the dirty dataset
    df.to_csv('sample_dirty_dataset.csv', index=False)
    return df

def clean_dataset(df):
    # 1. Standardize column names
    df.columns = df.columns.str.lower().str.replace(' ', '_').str.replace('-', '_')

    # 2. Remove duplicates
    print("\nNumber of duplicates:", df.duplicated().sum())
    df = df.drop_duplicates()

    # 3. Handle NaN values
    print("\nNaN values before cleaning:")
    print(df.isnull().sum())
```

```
    # Fill numeric columns with median
    numeric_columns = df.select_dtypes(include=[np.number]).columns
    for col in numeric_columns:
        df[col] = df[col].fillna(df[col].median())

    # Fill categorical columns with mode
    categorical_columns = df.select_dtypes(include=['object']).columns
    for col in categorical_columns:
        df[col] = df[col].fillna(df[col].mode()[0])

    print("\nNaN values after cleaning:")
    print(df.isnull().sum())

    return df

def main():
    # Create and load the sample dataset
    print("Creating sample dirty dataset...")
    original_df = create_sample_dataset()
    print("\nOriginal Dataset:")
    print(original_df)

    # Clean the dataset
    print("\nCleaning dataset...")
    cleaned_df = clean_dataset(original_df)

    # Save cleaned dataset
    cleaned_df.to_csv('cleaned_dataset.csv', index=False)
```

```
    print("\nCleaned Dataset:")
    print(cleaned_df)

    print("\nCleaned dataset has been saved as 'cleaned_dataset.csv'")

if __name__ == "__main__":
    main()
```

Output:

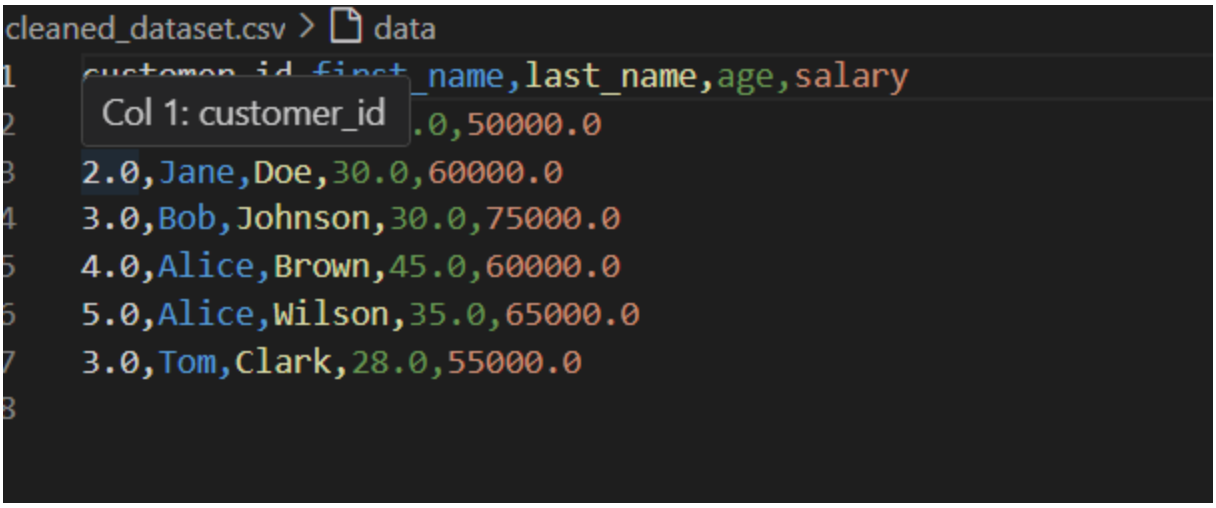
```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df[col] = df[col].fillna(df[col].mode()[0])

NaN values after cleaning:
customer_id    0
first_name     0
last_name      0
age            0
salary         0
dtype: int64

Cleaned Dataset:
   customer_id first_name last_name   age   salary
0          1.0       John     Smith  25.0  50000.0
1          2.0       Jane       Doe  30.0  60000.0
3          3.0        Bob   Johnson  30.0  75000.0
4          4.0      Alice     Brown  45.0  60000.0
5          5.0      Alice    Wilson  35.0  65000.0
6          3.0        Tom     Clark  28.0  55000.0

Cleaned dataset has been saved as 'cleaned_dataset.csv'
PS C:\Users\Namitha\OneDrive\Desktop\AIAC\Lab Test-4>
```

Original Data:

sample_dirty_dataset.csv > data

```
1   Customer ID,First_Name,Last name,Age,SALARY
2   1.0,John,Smith,25.0,50000.0
3   2.0,Jane,Doe,30.0,60000.0
4   2.0,Jane,Doe,30.0,60000.0
5   3.0,Bob,Johnson,,75000.0
6   4.0,,Brown,45.0,
7   5.0,Alice,Wilson,35.0,65000.0
8   ,Tom,Clark,28.0,55000.0
9   5.0,Alice,Wilson,35.0,65000.0
0
```

Cleaned data:


cleaned_dataset.csv > data
customer_id,first_name,last_name,age,salary
Col 1: customer_id
.0,50000.0
2.0,Jane,Doe,30.0,60000.0
3.0,Bob,Johnson,30.0,75000.0
4.0,Alice,Brown,45.0,60000.0
5.0,Alice,Wilson,35.0,65000.0
3.0,Tom,Clark,28.0,55000.0