CASE STUDY: Subscription Management & Renewal Prediction System

A SaaS company offers monthly subscription plans to customers.
They want to automate:

- Customer onboarding
- Renewal tracking
- Expiry reminders
- Revenue reporting
- Subscription health insights

Your task is to create a Python + MySQL + Date Analytics module for this system.

---

1. BUSINESS BACKGROUND

The SaaS product sells monthly subscriptions. Every subscription has:

- Start date
- Expiry date (start date + 30 days)
- Created timestamp
- Customer details
- Renewal status

The operations team needs:

- Lists of expiring subscriptions
- Renewal projections
- Overdue subscriptions
- Average subscription age
- Customer activity insights

Your job is to build a working analytics system that pulls data from a **database**, processes it in **Python**, and answers core business questions.

---

2. DATABASE DESIGN

Table: subscriptions

```
CREATE DATABASE subscription_app;
USE subscription_app;

CREATE TABLE subscriptions (
    sub_id INT PRIMARY KEY,
    customer_name VARCHAR(50),
    start_date DATE,
    expiry_date DATE,
    created_at DATETIME,
    plan_type VARCHAR(20)    -- Monthly, Quarterly, Yearly
);
```

Sample Data

```
INSERT INTO subscriptions VALUES
(1, 'Aisha Khan', '2024-12-15', '2025-01-15', '2024-12-15 10:30:00', 'Mor
(2, 'Rahul Sharma', '2025-01-05', '2025-02-05', '2025-01-05 09:45:00', 'M
(3, 'Imran Ali', '2025-02-10', '2025-03-10', '2025-02-10 14:12:00', 'Mont
(4, 'Meera Iyer', '2025-03-01', '2025-04-01', '2025-03-01 17:05:00', 'Mor
(5, 'Sanjay Patel', '2025-02-20', '2025-03-20', '2025-02-20 13:00:00', 'Q
```

---

3. PYTHON EXTRACT + TRANSFORM

Connect to Database and Pull Records

```
import pymysql
import pandas as pd
from datetime import date

conn = pymysql.connect(
    host="localhost",
```

```
    user="root",
    password="Password123!",
    database="subscription_app"
)

df = pd.read_sql("SELECT * FROM subscriptions", conn)

conn.close()

print(df)
```

Output contains Python `date` and `datetime` objects.

---

4. PYTHON DATE ANALYTICS

A. Days Left Before Expiry

```
df["days_left"] = (pd.to_datetime(df["expiry_date"]) - pd.Timestamp.today
```

B. Subscription Age (days since start)

```
df["age_days"] = (pd.Timestamp.today() - pd.to_datetime(df["start_date"])
```

C. Mark Status

```
def status(row):
    if row["days_left"] < 0:
        return "Expired"
    elif row["days_left"] <= 7:
        return "ExpiringSoon"
    else:
        return "Active"

df["status"] = df.apply(status, axis=1)
```

---

5. BUSINESS REPORTS

Report 1: Active, Expired, Expiring Soon Counts

```
print(df["status"].value_counts())
```

Report 2: Whose Subscription Expires This Week?

```
print(df[df["days_left"].between(0, 7)])
```

Report 3: Overdue Subscriptions (Expired but not Renewed)

```
print(df[df["days_left"] < 0])
```

Report 4: Revenue Projection

Monthly plan: assume 1000
Quarterly: 2700
Yearly: 10000

```
price_map = {
    "Monthly": 1000,
    "Quarterly": 2700,
    "Yearly": 10000
}

df["renewal_value"] = df["plan_type"].map(price_map)
```

Next 30-Day Revenue Forecast

```
forecast = df[df["days_left"] <= 30]["renewal_value"].sum()
print("Projected Revenue:", forecast)
```

---

6. DELIVERABLES (15 Tasks)

A. Database + SQL Tasks

1. Write SQL to fetch all subscriptions expiring within 10 days.
2. Write SQL to find customers who subscribed in the current month.
3. Fetch all yearly plans and sort by expiry_date.
4. Identify all subscriptions lasting more than 30 days (Quarterly/Yearly).
5. Write SQL to check data quality: expiry_date < start_date.

B. Python Extraction Tasks

6. Connect Python to MySQL and load the subscriptions table into a DataFrame.
7. Convert created_at into `"DD-MM-YYYY HH:MM"` format.
8. Add a column `days_overdue` for expired subscriptions.
9. Add a column `next_billing_date`.
10. Generate a list of customers whose billing is due today.

C. Analytics & Business Logic Tasks

11. Generate a report showing active vs expired vs expiring soon.
12. Create a renewal reminder list containing customers whose days_left <= 5.
13. Calculate customer lifetime (age_days).
14. Group by plan_type and calculate:

- total customers
- average age
- expected renewal revenue

15. Export the final DataFrame to `subscription_report.csv`.

---

7. FINAL OUTPUT EXPECTED

A complete working Python program that:

- Connects to MySQL
- Pulls subscription data
- Performs date transformations
- Generates renewal and expiry reports
- Exports dashboards or CSVs
- Answers the business questions
