Assignment 1 – Experiential Learning & Case Study

Student Name: Jayesh Vilas Kapade
Enrollment No.: ADT23SOCB0483
Course/Subject: DATA ENGINEERING
Date: 30/08/2025

---

1) Answer 1 – Research: Real-World Data Sources & Integration

1.1 What counts as a "real-world data source"

Real-world sources are systems that power day-to-day operations and/or external open datasets:

- **Operational databases:** MySQL, PostgreSQL, SQL Server, Oracle.
- **Cloud warehouses/lakes:** Azure Synapse/SQL, Snowflake, BigQuery, Amazon Redshift; lake storage (Azure Data Lake, Amazon S3, Google Cloud Storage).
- **Flat files & documents:** CSV, Excel, Parquet, JSON, XML.
- **APIs & streams:** REST/JSON APIs (e.g., public health, weather, finance), streaming platforms (Kafka/Event Hubs), webhooks.
- **Open data portals:** data.gov.in, World Bank, WHO, GHO, city/state portals.
- **Application exports:** ERP/POS/CRM logs, Google Analytics exports, Shopify exports, etc.

Selection criteria: business relevance, data freshness & latency needs, data quality, schema stability, access method/connector availability, cost, licensing/usage rights, security/privacy requirements.

1.2 Power BI & modern data platform integration patterns

- **Direct-to-Desktop ingestion:** Power BI Desktop connectors for files/DBs/APIs via *Get Data → Power Query*. Suitable for small/medium data.
- **Dataflows (Power BI Service):** Centralized, reusable ETL in the cloud; compute on Microsoft fabric; promotes single source of truth.
- **Gateway & Scheduled Refresh:** On-premises data → Power BI Service via gateway. Refresh frequency up to 8/day (Pro) or more (Premium/Fabric). Use **Incremental Refresh** for large facts.
- **Lakehouse/Warehouse patterns:** Store raw & curated data in a data lake/warehouse (e.g., *Azure Data Lake + Synapse/Snowflake/BigQuery*). Use ADF/Synapse Pipelines/Databricks for ELT; Power BI connects in import or DirectQuery mode.
- **Streaming/Real-time:** Push data to Power BI REST API, use Event Hubs/Kafka → Stream Analytics/Fabric Real-Time. Build live tiles.

Security & governance essentials:

- Use service principals/managed identities where possible; avoid personal credentials in production.
- Enforce data classification, dataset endorsements (Certified/Promoted), Data Loss Prevention policies.
- Apply row-level security (RLS) for store/region-based data access; validate with test users.
- Keep a lineage view (Power BI or Fabric) from source → dataflow → dataset → report.

1.3 Example: integrating a public/API or file source in Power BI

Files/Folder method (recommended for this assignment):

1. Place all CSVs in one folder.
2. *Get Data → Folder*; combine; inspect Power Query steps.
3. Define data types, trim/clean text, handle nulls, create reference queries for each table.
4. Create a separate Date table (mark as date table) or use provided dim_calendar.
5. Close & Apply → build star schema.

API method (conceptual):

```
let
  Source = Json.Document(Web.Contents("https://api.example.com/v1/sales",
[Query=[from="2025-01-01", to="2025-03-31"], Headers=[Authorization="Bearer <token>"]])),
  ToTable = Table.FromList(Source, Splitter.SplitByNothing(), null, null, ExtraValues.Error),
  Expand = Table.ExpandRecordColumn(ToTable, "Column1",
{"order_id","date","store_id","product_id","qty","price"})
in
  Expand
```

For authenticated APIs, store credentials securely and configure refresh in the Service.

1.4 Platforms quick compare (what to use when)

- **Power BI Dataflows/Fabric** → low-code ETL, centralized semantics, governance.
- **Azure Data Factory / Synapse Pipelines** → robust scheduled ingestion from 100+ sources; orchestration.
- **Databricks / Spark** → scalable transforms, ML; great for big data & notebooks.
- **Warehouses (Snowflake/BigQuery/Synapse)** → performant analytics, SQL governance, role-based security.
- **DirectQuery vs Import** → DirectQuery for near real-time/large datasets; Import for speed & DAX flexibility. Hybrid (= Composite) when needed.

Conclusion: Choose a source aligned to business need, land data in a governed store/dataflow, model a star schema, and visualize in Power BI with security, refresh and lineage in place.

---

2) Answer 2 – Case Study (80%): Retail Mini Project – Sales, Inventory & Promotions

Detailed Case Study: Healthcare Patient Visit Analysis:

1. Data Capture

In a modern hospital/clinic, multiple systems generate healthcare data every day:

- **Patient Registration System:** Captures patient details such as name, age, gender, contact, and unique patient ID.
- **Visit/OPD System:** Records each visit with date, time, department (e.g., Cardiology, Pediatrics), doctor, and reason for visit.
- **E-Prescription System:** Stores prescribed medicines, dosage, and duration for each visit.
- **Billing System:** Logs consultation charges, lab test costs, and payment mode (cash, UPI, card, insurance).

Example: On *01-08-2025*, Patient P001 (Age: 45, Male) visits the **Cardiology** department. The system captures:

- Visit details (date, department, doctor)
- Prescription (Atorvastatin, 10 mg × 30 days)
- Billing (₹800 paid via UPI)

---

2. Data Storage

All captured data is stored centrally for easy retrieval and analysis:

- **Database Options:** Relational database (PostgreSQL, MySQL) or a healthcare data warehouse (Snowflake, Azure Synapse, BigQuery).
- **Integration:** Patients, Visits, Prescriptions, and Billing tables are linked via **Patient_ID** and **Visit_ID**.

Example Table (Visits):

Visit_ID	Patient_ID	Date	Department	Doctor	Payment Mode	Bill Amount
V1001	P001	01-08-2025	Cardiology	Dr. Sharma	UPI	₹800

---

3. Data Processing

Before analysis, raw hospital data must be cleaned and transformed:

- **Remove Duplicates** → Ensure no duplicate visits are recorded.
- **Correct Errors** → Fix invalid patient ages (e.g., Age = 200).
- **Data Transformation** → Add derived columns such as:
    - *Total Bill = Consultation Fee + Lab Tests + Medicines*
    - *Age Group* (0–12, 13–18, 19–30, etc.)
- **Combine Data** → Link visits with prescriptions to see which medicines were prescribed most often.

Example: If Patient P002 buys 2 medicines (₹150 + ₹200) and consultation fee = ₹500,
→ **Total Bill = ₹850** is automatically calculated.

---

4. Data Analysis

Once processed, data can answer important healthcare questions:

- **Which departments get the most patients?**
  → Helps allocate doctors and staff (e.g., Pediatrics handles 300 patients/month).
- **Which medicines are prescribed most frequently?**
  → Helps pharmacy maintain stock and avoid shortages.
- **What is the patient demographic distribution?**
  → Understand age and gender trends (e.g., majority OPD patients are 31–45 years).
- **What is the revisit/readmission rate?**
  → Identify chronic patients who return within 30 days.

Example Insight: Analysis shows:

- Cardiology = 25% of visits
- Paracetamol = most prescribed drug
- 40% patients are in the 31–45 age group
- Peak visiting hours = 10 AM – 1 PM

---

5. Data Visualization (Power BI Dashboard)

Power BI (or Tableau) converts analysis into interactive dashboards:

- **Bar Chart → Visits by Department**
    - X-axis: Department
    - Y-axis: Number of Visits
- **Line Chart → Visits Over Time**
    - X-axis: Date/Month
    - Y-axis: Patient Visits
- **Pie Chart → Payment Mode Distribution**
    - % of Cash, UPI, Card, Insurance payments
- **KPI Cards → Quick Insights**
    - Total Patients
    - Total Revenue
    - Top Department
    - Top Prescribed Medicine

Example Dashboard Findings:

- Cardiology = 25% of visits
- UPI = 45% of payments
- Paracetamol prescribed most often
- Revenue peak in the first week of every month

---

6. Decision Making

The dashboard enables hospital administrators to take data-driven decisions:

- **Staff Allocation:** More doctors in Cardiology during rush hours (10 AM – 1 PM).
- **Pharmacy Stock:** Restock Paracetamol and Atorvastatin frequently to meet demand.
- **Patient Care:** Monitor chronic patients who revisit within 30 days.
- **Finance:** Encourage digital payments (UPI/Insurance) for faster processing.

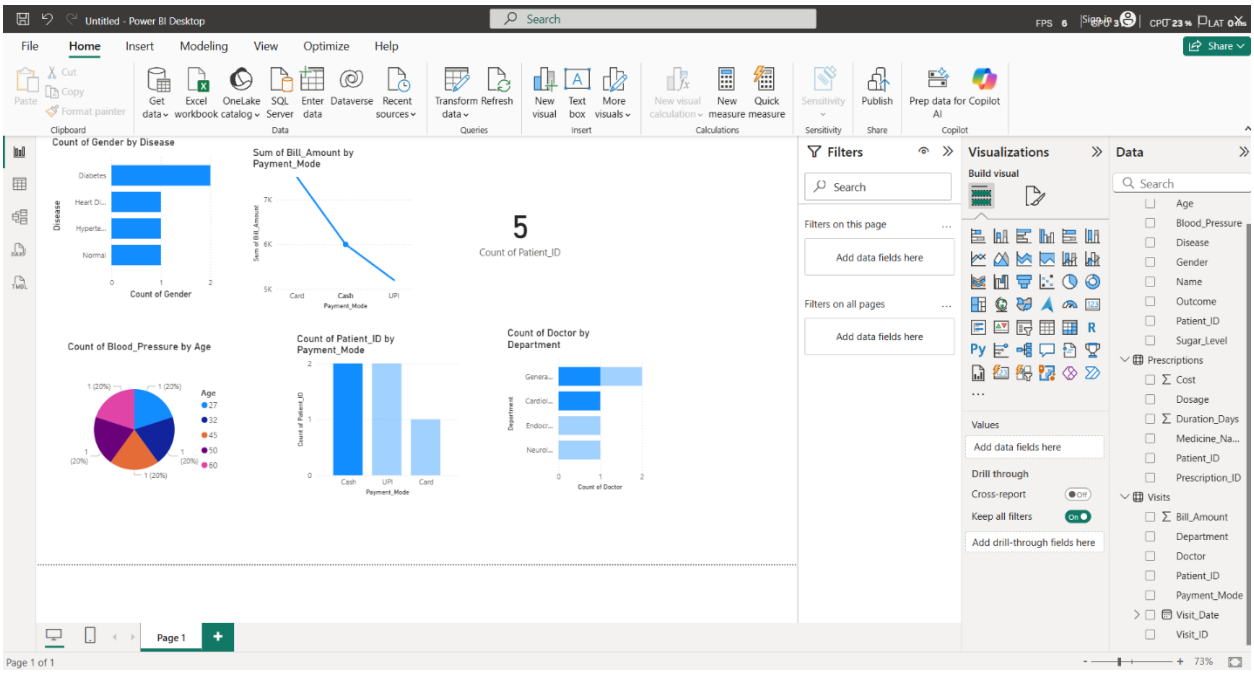
Example: After identifying Cardiology as busiest, the hospital assigns an extra doctor in the department, reducing average waiting time by 20%.

Conclusion

This mini project demonstrates the data lifecycle in healthcare:

- **Data Capture:** From registration, visits, prescriptions, and billing.
- **Data Storage:** Central database with linked tables.
- **Data Processing:** Clean, enrich, and combine datasets.
- **Data Analysis:** Identify department load, top medicines, patient demographics.
- **Data Visualization:** Dashboards for clear, actionable insights.
- **Decision Making:** Improve patient experience, optimize staff allocation, and streamline operations.

Thus, the hospital moves from **raw healthcare data → insights → better patient care & efficiency**, showcasing the power of data lifecycle management in healthcare.



Github Link:
https://github.com/JayeshKapade/Jayesh-kapade_ADT23SOCB0483_Assignment_1.git