Healthcare Admissions Database System

DMQL Course Project – End-to-End OLTP to OLAP Implementation

1. Project Overview-

This project implements a complete healthcare data management system to analyze patient appointment no-show behavior using the *Healthcare Dataset* from Kaggle. The system is designed as an end-to-end pipeline covering **data ingestion, transactional storage, analytical modeling, and visualization**.

The project is implemented in three phases:

- **Phase 1:** OLTP database design and data ingestion
- **Phase 2:** OLAP analytical layer using dbt and star schema modeling

2. Dataset Description

- **Source:** Kaggle – Healthcare Dataset
- **File:** `Healthcare_dataset.csv`
- **Rows:** ~10,000
- **Columns:** 15

Each record represents a patient appointment and includes demographic data, medical conditions, scheduling details, and a no-show indicator.

3. Technology Stack

- **Database:** PostgreSQL (Dockerized)
- **Administration:** pgAdmin (Dockerized)
- **Ingestion:** Python
- **Transformations & Analytics:** dbt-postgres
- **Visualization:** Streamlit
- **Orchestration:** Docker Compose

4. Phase 1 – OLTP Database Design & Ingestion

A normalized **OLTP schema (3NF)** was created in PostgreSQL to support transactional operations and maintain data integrity.

Key Features:

- Separate tables for patients, appointments, admissions, neighbourhoods, and medical conditions
- Primary and foreign key constraints
- Data validation and type enforcement

Data Ingestion:

- CSV data is loaded using a Python ingestion script
- Data is cleaned, transformed, and inserted into OLTP tables
- Row counts and integrity checks confirm successful ingestion

5. Phase 2 – OLAP Layer & dbt Transformations

dbt is used to transform OLTP data into an analytics-optimized **star schema**.

Star Schema Design:

- **Fact Table:** `fact_appointment`
- **Dimension Tables:**
  - `dim_patient`
  - `dim_date`
  - `dim_neighbourhood`
  - `dim_medical_condition`

dbt Capabilities:

- Modular SQL transformations
- Data quality tests
- Reproducible analytical models

This layer enables fast, reliable analytical querying.

6. Advanced SQL Analytics

Advanced SQL queries are used to analyze:

- No-show rates by age, gender, and neighbourhood

- Impact of SMS reminders on attendance
- Relationship between medical conditions and no-shows
- Temporal appointment trends

Queries leverage CTEs, window functions, and aggregations.

. Performance Tuning

Performance optimization techniques include:

- Indexing foreign keys and date columns
- Query execution plan analysis
- Star schema optimization

These improvements reduce query latency and improve dashboard performance.

8. Phase 3 – Application Layer (Streamlit)

A Streamlit dashboard is built on top of the analytical layer to provide:

- No-show KPIs
- Interactive filters
- Time-based and demographic insights

The dashboard enables non-technical users to explore healthcare appointment data.

9. Deployment & Orchestration

The entire system is containerized using Docker Compose, including:

- PostgreSQL
- pgAdmin
- Streamlit application

The platform can be started, stopped, or reset using a single command.

10. Conclusion

This project demonstrates a complete data lifecycle from raw healthcare data to analytical insights. It applies industry-standard practices in database design, data modeling, analytics engineering, and visualization, providing a scalable and production-ready healthcare analytics solution.