**Capstone Project:**

**End-to-End Data Modeling for a Retail Analytics Platform Objective**

Design and implement a scalable, high-performance data model for a retail analytics platform that enables businesses to track sales, customer behavior, inventory management, and supply chain performance. This project will require learners to bridge business and technical requirements, conduct JAD sessions, create conceptual/logical/physical models, integrate ETL processes, implement governance, and deploy using Git and Terraform.

**✅ 1. Requirement Gathering & Business Understanding**

**🎯 Simulated JAD Session (Joint Application Design)**

* **Stakeholders** (Simulated):
  + Retail Manager (inventory and sales)
  + Marketing Analyst (promotions, loyalty)
  + Supply Chain Officer (warehouses, suppliers)
  + Data Scientist (analytics needs)

**📝 Functional Requirements**

* Track inventory in real-time
* Monitor sales across channels
* Generate sales forecasts
* Enable personalized customer promotions
* Supply chain visibility from suppliers to store

**⚙️ Non-Functional Requirements**

* Scalable to millions of transactions/day
* Real-time and batch processing
* Secure (PII/GDPR-compliant)
* High availability & fault tolerance
* Auditability and traceability

**🧩 Key Business Entities & Workflows**

* **Entities**: Customers, Products, Orders, Stores, Suppliers, Warehouses, Promotions, Transactions
* **Workflows**:
  + Inventory movement from suppliers → warehouse → store
  + Sales transactions → customer profiles → feedback loop for personalization

**✅ 2. Conceptual, Logical, and Physical Data Models**

**📘 Conceptual Model**

* High-level ER Diagram (Entities + Relationships):
  + Customers ↔ Orders ↔ Products
  + Products ↔ Suppliers → Warehouses → Stores
  + Stores → Transactions
  + Promotions → Customers/Orders

**📗 Logical Model**

* Define attributes, keys:
  + **Customer**(ID, Name, Email, LoyaltyStatus)
  + **Product**(ID, Name, Category, Price, SupplierID)
  + **Order**(ID, CustomerID, StoreID, Timestamp, TotalAmount)
  + **Transaction**(ID, OrderID, PaymentMethod, DiscountApplied)
  + **Warehouse**(ID, Location, Capacity)
  + **Inventory**(ProductID, StoreID, Quantity, ReorderLevel)
  + **Promotion**(ID, Type, StartDate, EndDate, TargetGroup)
* **Normalization**: 3NF for OLTP; denormalized star schema for OLAP
* **Pipelines**:
  + Batch: daily sales and inventory
  + Real-time: streaming customer transactions (Kafka, Kinesis)

**📕 Physical Model**

* **Schema Design**:
  + Star schema for sales analysis: FactSales, DimCustomer, DimProduct, DimStore, DimDate
* **Indexing**:
  + On ProductID, OrderID, CustomerID
* **Partitioning**:
  + By Date or Region for fact tables
* **Constraints**:
  + Foreign keys, check constraints (e.g., non-negative inventory)

**✅ 3. ETL Process & Data Integration**

**🔄 ETL Architecture**

* Tools: Apache Airflow or DBT for orchestration
* **Sources**:
  + POS systems (Sales)
  + E-commerce platform (Orders)
  + CRM (Customer data)
  + Supplier systems (Inventory shipments)
  + Social Media/Reviews (External trends)

**ETL Steps:**

1. **Extract**: API pulls, database connectors, file ingestion
2. **Transform**:
   * Join, clean, deduplicate, enrich
   * Generate derived columns (e.g., RFM scores, trend flags)
3. **Load**:
   * Data warehouse (e.g., Snowflake, Redshift, BigQuery)

**✅ 4. Data Governance & Quality**

**📏 Data Quality Rules**

* Null checks, schema validation
* Duplicate detection (based on unique constraints or fuzzy matching)
* Valid email, date, and numeric formats

**🔐 Governance Policies**

* Access roles (admin, analyst, viewer)
* Masking/encryption for PII fields (e.g., Email, Name)
* GDPR support: opt-out handling, data deletion flags

**📂 Metadata Management**

* Use a catalog tool like **Apache Atlas** or **DataHub**
* Maintain schema versions, ETL lineage, data dictionaries

**✅ 5. Deployment & Version Control**

**🛠 Git Strategy**

* Repos:
  + /models/ – SQL/DDL files
  + /etl/ – Airflow DAGs/DBT models
  + /terraform/ – infra scripts
* Branching: Feature → Dev → Main with PR reviews

**☁️ Terraform Infrastructure**

* Create cloud resources (example: AWS)
  + RDS/Redshift or Snowflake instance
  + S3 for raw data storage
  + IAM roles for access

hcl

CopyEdit

resource "aws\_redshift\_cluster" "analytics" {

cluster\_identifier = "retail-analytics"

node\_type = "dc2.large"

number\_of\_nodes = 2

...

}

* Rollback: maintain versioned DDL files and use Flyway or Liquibase for migrations

**✅ 6. Testing, Validation, and Documentation**

**🔍 Testing**

* **Data Validation**:
  + Compare raw vs transformed counts
  + Check aggregation accuracy
* **Performance Testing**:
  + Query response time on FactSales with 100M rows
  + Load testing on batch pipelines

**📚 Documentation**

* ER diagrams (Lucidchart, dbdiagram.io)
* Data Dictionary (Excel or embedded in DB)
* Architecture diagram (Draw.io)
* README in GitHub

**🎁 Final Output**

You should end with:

* Fully versioned GitHub repo (models, ETL, infra)
* Deployed cloud data warehouse via Terraform
* Documented data model and architecture
* Sample dashboards (e.g., Power BI or Looker) to showcase insights like:
  + Sales by product/store/time
  + Customer segmentation
  + Inventory turnover rate
  + Promotion ROI