Prepared by:

* Bilal Fathy Hamad
* Omar Adel Mohammed
* Youssef Ahmed Alywa
* Mohammed Elsayed

**Team Roles:**

* Omar Adel: SQL Database and Data Collection
* Youssef Ahmed: SQL Scripts
* Bilal Fathy: Data Warehouse and ETL
* Mohammed Elsayed: ML Model**Week 1: SQL Database and SQL Scripts**

Overview:

This database system manages customer interactions, transactions, and product information for a retail business. The system consists of four main tables: Customer, Interactions, Transactions, and Products, which are interconnected to track customer behavior, sales, and inventory.

Table Structures:

1. Customer Table

* customer\_id (Primary Key)
* first\_name
* last\_name
* gender
* email
* phone\_number

2. Interactions Table

* interaction\_id (Primary Key)
* interaction\_date
* interaction\_type
* interaction\_score
* customer\_id (Foreign Key referencing Customer table)

3. Transactions Table

* + transaction\_id (Primary Key)
  + transaction\_date
  + amount
  + payment\_method
  + customer\_id (Foreign Key referencing Customer table)
  + product\_id (Foreign Key referencing Products table)

4. Products Table

* + product\_id (Primary Key)
  + manufacturer
  + modele
  + price

Key Functions:

1. Product Analysis Functions

* + numberOfPhone: Returns the count of products for a specified manufacturer
  + biggerThan: Returns the count of products priced above a specified amount
  + smallerThan: Returns the count of products priced below a specified amount

2. Transaction Analysis Capabilities

* + Track total transactions by customer
  + Monitor transactions by product
  + Analyze payment method distribution
  + Filter transactions by amount
  + Calculate total amount by customer for specific payment methods
  + View transactions within date ranges
  + Identify top customers by transaction volume
  + Generate monthly transaction reports
  + Calculate monthly revenue totals

3. Customer Interaction Analysis Features

* + Identify customers with high average interaction scores (above 8)
  + Filter interactions by customer gender
  + Identify customers with no recorded interactions
  + Count and categorize interactions per customer
  + Filter interactions by type (e.g., phone interactions)

Common Query Examples:

1. Customer Analysis

* + Find customers with high engagement (interaction scores > 8)
  + Identify inactive customers (no interactions)
  + Track interaction frequency by customer
  + Monitor customer interaction types and scores

2. Transaction Analysis

* + View top 5 customers by transaction volume
  + Identify top 15 products by sales frequency
  + Analyze payment method distribution
  + Track monthly transaction trends
  + Monitor high-value transactions
  + Generate date-range based reports

3. Product Analysis

* + Count products by manufacturer
  + Analyze product pricing distribution
  + Track product sales performance

Relationships:

* + Each customer can have multiple interactions and transactions
  + Each transaction is associated with one customer and one product
  + Interactions are linked to specific customers
  + Products are linked to transactions

**Week 2: Data Warehouse and ETL Script**

Data Warehouse ETL Process Documentation

Overview:

This documentation describes the Extract, Transform, Load (ETL) process for a customer data warehouse implemented in Python. The process handles data from multiple dimension tables and a fact table, performing data cleaning and integration to create a unified dataset for analysis.

System Components:

1. Database Structure

* + Customer\_Dimension: Contains customer demographic information
  + Product\_Dimension: Stores product details
  + Time\_Dimension: Manages temporal data
  + Sales\_Fact: Contains transactional sales data

2. Table Schemas

Customer\_Dimension:

* + Customer\_ID (Primary Key)
  + First\_Name
  + Last\_Name
  + Gender
  + Email
  + Phone\_Number

Product\_Dimension:

* + Product\_ID (Primary Key)
  + Manufacturer
  + Model
  + Price

Time\_Dimension:

* + Date\_ID (Primary Key)
  + Transaction\_Date
  + Years
  + Months
  + Day
  + Quarter

Sales\_Fact:

* + Sale\_ID (Primary Key)
  + Amount
  + Payment\_Method
  + Customer\_ID (Foreign Key)
  + Product\_ID (Foreign Key)
  + Date\_ID (Foreign Key)

ETL Process Steps:

1. Database Connection

* + Utilizes SQLAlchemy engine for database connectivity
  + Connects to a local SQL Server instance
  + Accesses the customerDWH database
  + Uses trusted connection for authentication

2. Data Extraction

* + Extracts complete datasets from all dimension and fact tables
  + Uses Pandas DataFrame for data handling
  + Performs SELECT operations on each table
  + Stores data in separate DataFrames for processing

3. Data Cleaning Process

Implements a standardized cleaning function that:

* Identifies numeric and non-numeric columns automatically
* Handles missing values in numeric columns using median imputation
* Fills missing values in non-numeric columns using mode
* Removes duplicate records
* Applied to all dimension and fact tables

4. Data Integration

Merges all cleaned DataFrames using left joins

Join sequence:

* Sales\_Fact with Customer\_Dimension on Customer\_ID
* Result with Product\_Dimension on Product\_ID
* Final merge with Time\_Dimension on Date\_ID

5. Data Validation

Performs quality checks on the final dataset:

* Reviews sample data (first few records)
* Generates descriptive statistics
* Checks for any remaining null values

Technical Requirements:

* + Python environment with Pandas library
  + SQLAlchemy for database connectivity
  + ODBC Driver 17 for SQL Server
  + Local SQL Server instance
  + Appropriate database permissions

Output:

The process produces a comprehensive dataset that combines:

* + Customer information
  + Product details
  + Temporal data
  + Sales transactions

This integrated dataset enables:

* + Customer behavior analysis
  + Product performance tracking
  + Temporal sales analysis
  + Cross-dimensional reporting capabilities

**Week 3: Machine Learning Model**

This script is designed to demonstrate the process of building, training, and evaluating a machine learning model using the Python programming language and the scikit-learn library. Below is a breakdown of the script's functionality:

1. Importing Libraries: The script begins by importing the necessary libraries, including pandas for data manipulation, scikit-learn for machine learning functionalities, and specific modules for data preprocessing and model evaluation.

2. Loading Data: The script loads three Excel files ('Sales.xlsx', 'Products.xlsx', 'Time.xlsx') containing sales, product, and time data, respectively, into separate DataFrames.

3. Data Preprocessing:

* Merge Operations: Two merge operations are performed to combine the sales data with product and time dimensions separately.
* Feature Engineering: New features 'Month' and 'Quarter' are derived from the 'Transaction\_Date' and 'Quarterr' columns, respectively.

4. Feature Selection: The features ('Price', 'Month', 'Quarter') and the target variable ('Amount') are selected for model training.

5. Train-Test Split: The data is split into training and testing sets using the `train\_test\_split` function from scikit-learn.

6. Data Cleaning:

* + Imputation: Missing values in the training set are imputed with the median value using the `fillna` method.
  + Handling Missing Values: Rows with missing values are dropped from both the training and testing sets to ensure data cleanliness and alignment.

7. Model Training:

* + Random Forest Regressor: A Random Forest Regressor model with 100 estimators is trained on the cleaned training data.

8. Prediction: The trained model is used to make predictions on the testing data ('X\_test').

9. Model Evaluation:

* + Mean Absolute Error (MAE): The Mean Absolute Error between the predicted and actual values is calculated and printed.
  + Root Mean Squared Error (RMSE): The Root Mean Squared Error, a measure of the model's accuracy, is calculated and displayed.