# SQL-and-Different-Shapes-of-Data-Term\_Project\_1

# Data Engineering Project

## Introduction

This project delivers a comprehensive data engineering solution for analyzing Adidas sales data, featuring a robust MySQL database schema, efficient ETL processes, and a Python-based data import mechanism. The focus is on adhering to best practices in naming, packaging, versioning, documentation, and testing to ensure maintainability and scalability.

## Project Composition

# Dataset:

`Addidas\_US\_Sales.csv`: The raw dataset of Adidas sales.

# Structure Breakdown:

`sql\_code\_Addidas\_Sales.sql`: SQL script for creating tables, ETL procedures for transforming and creating views as data marts

Python script `data.py` for transforming, cleaning and loading data from csv into the Operational layer tables.

# Database Schema:

Includes tables:

Retailer: Stores retailer information. (Retailer\_Entry\_ID:PK , Retailer\_ID: Unique identifier for each retailer, Retailer\_Name)

Product: Contains details about products.( Product\_ID: PK, Product\_Name, Price\_per\_Unit: Unit price of the product)

Location: Holds location-related data.( Location\_ID: PK, Region: Region where the sale took place, State: State where the sale occurred, City: City of the transaction)

SalesTransaction: Captures individual sales transactions.( Transaction\_ID:PK, Retailer\_Entry\_ID: FK referencing Retailer, Product\_ID: FKreferencing Product, Location\_ID: FKreferencing Location, Invoice\_Date: Date of the transaction, Units\_Sold: Number of units sold, Total\_Sales: Total sales amount, Operating\_Profit: Profit from the operation, Operating\_Margin: Margin from the operation, Sales\_Method: Method of the sale (e.g., online, in-store))

## Documentation:

`README.md`: Provides comprehensive documentation and setup instructions. The file you are reading right now.

## Instructions for Reproduction

### 1. Database Setup

Execute ` sql\_code\_Addidas\_Sales .sql` to create the necessary tables, ETL procedures and views as data marts in MySQL Workbench.

### 2. Python Data Load

1. Ensure you have Python installed on your system.

2. Install the required Python packages using: `pip install -r required\_packages.txt`

3. Run `data.py` to transform dataset and load data in MYSQL Workbench.

### 3. Reproduce Project

Clone the repository using git clone <https://github.com/zeep-code/DE1.git>

Follow Database setup as stated above and load data in using Python Data Load instructions

You need to have:

Database Server such as MYSQL Server connection

MYSQL Workbench or Command-Line Client

Python Installation & Required Libraries (pandas, SQLAlchemy, mysql-connector-python)

## Notes

- Make sure to configure your MySQL connection parameters in the scripts.

- The project assumes you have the necessary permissions to create databases, tables, and procedures.

- Adjust file paths and database credentials based on your local setup.

## Best Practices

Naming Conventions: Consistent and descriptive naming across all SQL entities and Python functions.

Documentation: Detailed comments within scripts and comprehensive README for guidance.

Packaging: Organized directory structure separating SQL scripts, Python source code, and tests.

Error Handling: Robust error checking in Python scripts and SQL procedures.

Performance Optimization: Indexing in database tables and efficient queries in ETL processes.

Scalability Consideration: Design allowing for easy expansion and modification of database schema and ETL processes.