Repository: <https://github.com/dnguyen-1/AdventureWorks507>

**Introduction**

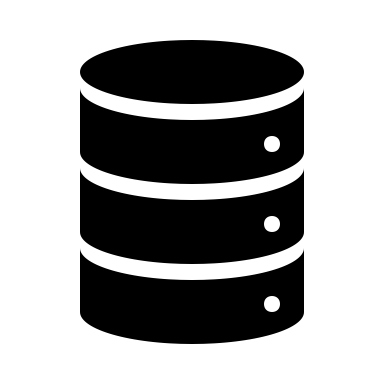
|  |  |
| --- | --- |
| Purpose | Produce an extract, transform, and load automated pipeline that enables a product to the marketing team to analyze sales |
| Scope | System will cover employee transactions of and details of the order and product, including order date, due date, ship date |
| Out of Scope | Customer demographic data, ie. Name, address and other attributes identifying the customer |
| Audience | Business intelligence developers, operational analyst, production leaders, and sales leaders |
| Contributors: | Gabriel  Jorge  Duy Nguyen |

**System Overview**

|  |  |
| --- | --- |
| Description | Data sources from customer transactions created by employees |
| Key Features | Key features include tables on:   * **Customers** Holds customer data such as names, addresses, and contact information. * **SalesOrderDetail** Represents line items for sales orders (product, quantity, price, etc.). * **SalesTerritory** Defines geographical or market segments (e.g., regions) to facilitate sales analytics. * **PurchaseOrderHeader** Stores high-level information about each purchase order (e.g., vendor, order date, status). * **PurchaseOrderDetail** Contains line-item details for each purchase order (e.g., product, quantity, cost). * **ShipMethod** Enumerates shipping/carrier methods for orders, enabling logistics and cost analysis. |

**Architecture and Data Design**

Customer



AdventureWorks.sql

**Report Design**

Security, Performance, and Limitations

|  |  |
| --- | --- |
| References | [I used generative AI to debug and information from the Microsoft repository, we should probably provide a reference] |
| Security | [@jorge can you sribe something on the security] |
| Future improvements and Current limitations | 1. Scalability:    * Partition large tables or implement data archiving for older records. 2. Cloud Integration:    * Migrate the MySQL instances to AWS RDS, Azure Database for MySQL, or GCP Cloud SQL. 3. CI/CD:    * Automate schema changes and data loading via a continuous integration pipeline (e.g., GitHub Actions). 4. Data Governance & Quality:    * Implement validation rules and data quality checks within the pipeline. |