# Acme Manufacturing Cloud Migration: Database Design

**Document Version:** 1.0
**Date:** October 26, 2024 - 13 December 2024

This document details the database architecture, schema, and migration strategy for the Acme Manufacturing cloud migration project. Both relational (RDS) and NoSQL (DynamoDB) databases will be used.

**I. Database Selection Rationale:**

- **Relational Database (RDS - MySQL):** Used for structured data requiring ACID properties (Atomicity, Consistency, Isolation, Durability), such as transactional data, customer information, and product details. MySQL was chosen due to [Explain reason for choosing MySQL, e.g., existing expertise, ease of use, scalability].
- **NoSQL Database (DynamoDB):** Used for semi-structured and unstructured data requiring high scalability and flexibility, such as product reviews, user activity logs, and sensor data. DynamoDB was selected due to [Explain reason for choosing DynamoDB, e.g., scalability requirements, flexibility in handling various data types, cost-effectiveness for variable workloads].

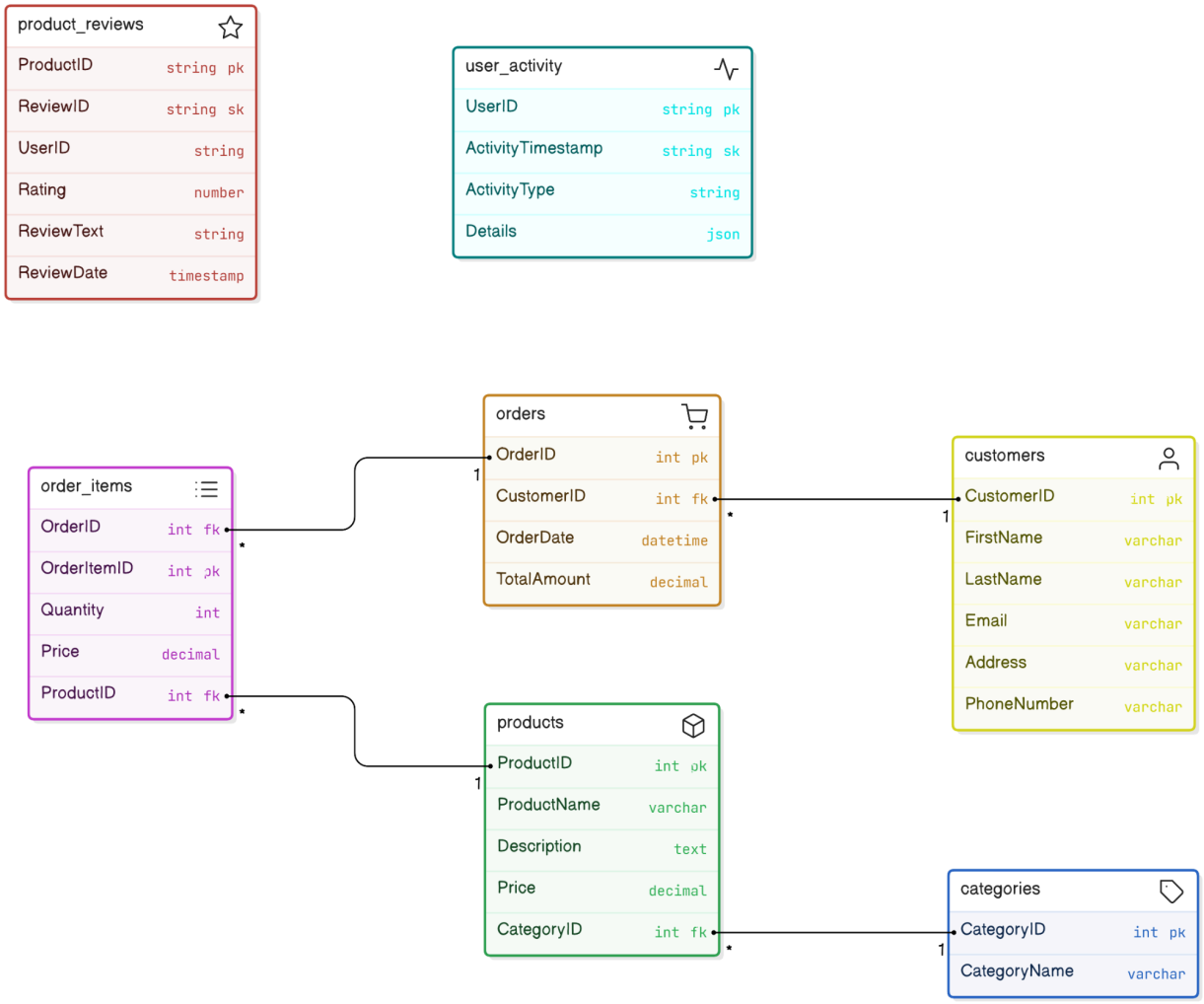
**II. Relational Database (RDS - MySQL):**

**A. Database Schema:**

The relational database (RDS MySQL) will consist of [Number] tables. Example tables and their attributes might include:

- **Customers:** CustomerID (INT, primary key), FirstName (VARCHAR), LastName (VARCHAR), Email (VARCHAR), Address (VARCHAR), PhoneNumber (VARCHAR).
- **Products:** ProductID (INT, primary key), ProductName (VARCHAR), Description (TEXT), Price (DECIMAL), CategoryID (INT, foreign key referencing Categories table).
- **Categories:** CategoryID (INT, primary key), CategoryName (VARCHAR).
- **Orders:** OrderID (INT, primary key), CustomerID (INT, foreign key referencing Customers table), OrderDate (DATETIME), TotalAmount (DECIMAL).
- **OrderItems:** OrderItemID (INT, primary key), OrderID (INT, foreign key referencing Orders table), ProductID (INT, foreign key referencing Products table), Quantity (INT), Price (DECIMAL).

Acme Manufacturing Cloud Migration Database Design




**B. Migration Strategy:**

The migration strategy for the MySQL database will involve using AWS Database Migration Service (DMS). The following steps will be followed:

1. **Assessment:** A thorough assessment of the existing MySQL database will be performed, identifying potential issues (e.g., large tables, data inconsistencies).
2. **Replication:** A replication instance will be set up using DMS to replicate data from the on-premises database to the RDS MySQL instance.
3. **Cutover:** After replication is complete, the application will be switched over to the RDS MySQL instance.
4. **Validation:** Thorough data validation will be performed to ensure data integrity after the migration.

**III. NoSQL Database (DynamoDB):**

**A. Table Design:**

DynamoDB will be used for specific data sets that require high scalability and flexibility. Example tables and attributes might include:

- **ProductReviews:** ProductID (STRING, partition key), ReviewID (STRING, sort key), UserID (STRING), Rating (NUMBER), ReviewText (STRING), ReviewDate (TIMESTAMP).
- **UserActivity:** UserID (STRING, partition key), ActivityTimestamp (STRING, sort key), ActivityType (STRING), Details (JSON).

**(Include a complete schema diagram here showing all tables, attributes, data types, and keys.)**

**B. Migration Strategy:**

The migration strategy for DynamoDB will involve exporting data from the existing data sources and importing it into the new DynamoDB tables. Tools such as the AWS CLI or AWS SDKs will be used. The following steps will be followed:

1. **Data Extraction:** Extract the data from the existing data sources.
2. **Data Transformation:** Transform the data to be compatible with DynamoDB's data model.
3. **Data Loading:** Load the data into the DynamoDB tables using the AWS CLI or an AWS SDK.
4. **Data Validation:** Verify data integrity and consistency after the migration.

**IV. Data Validation:**

Data validation will be performed after both the RDS and DynamoDB migrations are complete to ensure data integrity and consistency. This will involve comparing the data in the source and target databases and identifying any discrepancies. Automated scripts will be used to assist in this process.

**V. Migration Tools:**

- AWS Database Migration Service (DMS) for RDS.
- AWS CLI or AWS SDKs for DynamoDB.

This document provides a high-level overview of the database design and migration strategy. More detailed design specifications and migration plans will be maintained and updated regularly throughout the project. This design prioritizes scalability, performance, and data integrity. Regular reviews and updates of the database schema are crucial to adapt to changing requirements.