Database Design Document

Group 2: Ahmed Tamer Alqaoucy, Mohamed Farag, Aidan Urbina

# **Project Overview**

This project models a retail store management system. It incorporates four types of databases those being SQLite, MongoDB, Neo4j, and Redis. We expanded upon the idea that was given inside of the GitHub instructions.

# **Database design**

**Relational DB: SQLite**

* The purpose was to store structured data such as customers, products, and suppliers

**Document DB: MongoDB**

* The purpose was to store order data in the form of nested documents

**Graph DB: Neo4j**

* The purpose was to store payment and sales information

**Cache DB: Redis**

* The purpose was to store product cashing and popularity

# **Database Models**

**SQLite:**

* Customer:
  + Id (int, PK)
  + Name (text)
  + Email (text)
* Product:
  + Id (int, pk)
  + Name (text)
  + Price (float)
  + Stock\_quantity (int)
* Supplier:
  + Id (int, pk)
  + Name (text)
  + Contact\_info (text)

**MongoDB:**

{

"\_id": <ObjectId>,

"customer\_id": <int>,

"order\_date": "<YYYY-MM-DD>",

"status": "<string>",

"items": [

{

"product\_id": <int>,

"quantity": <int>,

"price": <float>

}

],

"total\_amount": <float>

}

**Neo4j:**

* Nodes: customer, order, supplier
* Relationships:
  + (customer)-[:PLACED]->(Order)
  + (order)-[:SUPPLIED\_BY]->(supplier)
  + (customer)-[:PAID]->(supplier)

**Redis:**

* (Products:{id}) JSON string of product information
* (Products\_views:{id}) Integer counter
* (popular\_products) sorted set of product views

# **Database Operations**

**SQLite Queries:**

* list\_products – Queries all products from the Product table.
* add\_product – Inserts a new product record into the Product table.
* list\_suppliers – Retrieves all supplier records.
* add\_supplier – Inserts a new supplier record.
* list\_customers – Retrieves all customer records.
* add\_customer – Inserts a new customer record.

**MongoDB Queries:**

* get\_orders – Finds orders by customer or fetches all orders from MongoDB Orders collection.
* Create\_order – Creates a new order

**Redis Queries:**

* get\_product – Checks Redis for a cached product by ID; if not found, loads from SQLite and caches it.
* popular\_products – Retrieves a sorted list of product IDs by Redis view counters.

**Neo4j Queries:**

* update\_product – Updates product metrics in Neo4j
* create\_order – Logs a customer placing an order and its relationship to a supplier.
* make\_payment – Records a payment transaction between a customer and a supplier.
* get\_payment – Queries payment relationships between nodes.
* sales\_analytics – Aggregates sales by time range using graph relationships.
* product\_history – Shows product sales relationships in graph history.