Basics of database systems

**Project – Order Database**

Lappeenranta-Lahti University of Technology LUT

Software Engineering

Spring 2023

Teemu Hämäläinen, 0612653

Table of contents

TABLE OF CONTENTS.............................................................................................................1

1 DEFINITION................................................................................................................2

2 MODELING.................................................................................................................3

2.1 Concept model...............................................................................................3

2.1 Relational model............................................................................................4

3 DATABASE IMPLEMENTATION...................................................................................5

4 DISCUSSION...............................................................................................................6

# Definition

**Order database**

The order database is a simple and small-scale database used by a small company called Bmazon. Bmazon sells products from different categories all over the world, and they need a database to handle it. The database stores information about the orders, what products they have, information about the product (such as price, category) and the amount of stock that each category of products has. The database also stores information about the shippers and suppliers of products, alongside of customer info.

The database utilizes a number of IDs, which allows for quick look-ups, additions and changes. At times these IDs are automatically incremented.

The database should be able to display quickly and easily different information, such as customers, status of orders, number of products in stock and the suppliers of products.

The following database queries are to be implemented:

1. Retrieve all customers and their info
2. Retrieve all orders that have not been filled yet along with the customer’s name and email address.
3. Retrieve the name and email of all suppliers who supply product with ProductID=x
4. Retrieve all orders made by the customer with the name of x
5. Retrieve the total number of products in stock for each category

# modeling

## Concept model

Figure 1 shows the ER model of the database. In total there are 7 entities, each with varying amounts of attributes. In total there are 5 relationships. There is a M:N relationship between Supplier and Product, meaning that a product can have multiple suppliers, and a single supplier can supply multiple different products).

The customer entity is separate from everything else but is linked with the **CustomerID** primary key. The Address data is located in Order details instead of customers, since the address of the order and the customer aren’t necessarily the same. Customers also have a composite attribute CustName, which consists of CustFirst and CustLast.

The model is subject to change when transforming to a relational model (most notably the implementation of customers-table), but the relationships and overall structure should remain the same. There will be a linking table between suppliers and products, due to them having a N:M relationship.

Diagram

Description automatically generated

**Figure 1:** ER model

## Relational model

Figure 2 shows the relational model that is based on the previous ER model. The N:M relationship between the suppliers and products resulted in an linking table productSupply. Customers name was transformed into a single attribute CustName, which was originally a composite attribute with first and last names of the customer.

Diagram

Description automatically generated

**Figure 2:** Relational modelfrom the ER model

# Database implementation

During implementation, the following constraints are implemented:

* **shippers**
  + Primary key ShipperID
  + All data must exist (NOT NULL)
* **orders**
  + Primary key OrderID
  + OrderID cannot be null (NOT NULL)
* **order details**
  + Foreign key **OrderID** references to order
    - ON DELETE CASCADE
  + Foreign key **ProductID** references to products
    - ON DELETE SET NULL
  + Foreign key **CustomerID** references to customer
    - ON DELETE SET NULL
  + All data must exist (NOT NULL)
* **customers**
  + Primary key CustomerID
  + CustEmail and CustomerID cannot be null (NOT NULL)
  + Other fields not mandatory (CustName, CustPhone)
* **products**
  + Primary key ProductID
  + Foreign key **CategoryID** references to categories
  + InStock and UnitPrice must exist (NOT NULL)
* **suppliers**
  + Primary key SupplierID
  + All data must exist (NOT NULL)
* **productSupply**
  + Primary key (Product id, SupplierID)
    - CONSTRAINT product\_supply\_pk
  + Foreign key **ProductID** references to products
    - ON DELETE CASCADE
    - ON UPDATE CASCADE
  + Foreign key **SupplierID** references to suppliers
    - ON DELETE CASCADE
    - ON UPDATE CASCADE
* **Categories**
  + Primary key CategoryID
  + CategoryName must exist (NOT NULL)

Some indices are also added in addition to the integrity constraints listed above, mainly to speed up certain actions like searching for specific shippers, joining the orders table with other tables based on the OrderID and joining the products table with other tables based on the CategoryID.

# discussion

The database could still be improved, for example by listing all of the customers orders in the customers table (for example as CustOrders). Some more indices could also be implemented/added to speed up certain queries. The ID system could also be adjusted, so that the products for example are IDs starting from 100000, shippers are 200000, customers 300000 etc.