**Unit 1 Assignment**

Laurence T. Burden

Purdue Global University

IN402: Modeling and Predictive Analysis

Steward Huang

November 11, 2023

The database created for this assignment is designed to meet the needs of an e-commerce store that sells t-shirts. The tables and attributes for this database has been built to the third normal form (3NF). This normalization process is important to mitigate performance, integrity, and maintainability issues (Teorey, et. al., 2006).

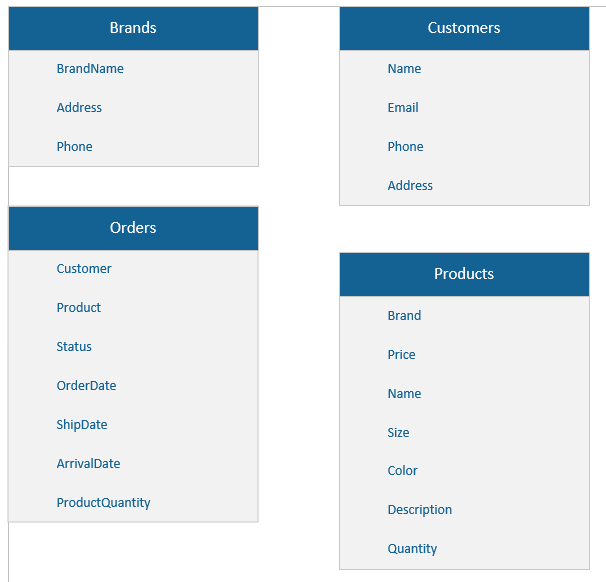
A non-normalized version of this database would contain 20 values stored in one large table. These columns for this table would include customer names, brand names, product information, order details, and more. A vast majority of rows in this hypothetical table would contain redundant data. Other rows may have columns that could be considered tables themselves.

“A table is in first normal form (1NF) if each domain contains simple values” (Kung & Tung, 2006). This definition can be distilled to mean that no column can contain tables as values. The process of bringing this database to 1NF is to create a minimum of three separate tables. These tables would include Customers, Orders, and Products.

A database is in second normal form if it meets the definition of first normal form and if non-key attributes are dependent on the entire key (Kung & Tung, 2006). This process is completed by moving certain repeating information out to a new table that is not dependent on the primary column. This can be done for this database by moving the brands into their own table. This allows the brands table to only hold information pertinent to the brand company. The product table would no longer need to hold the address and phone number for each brand. Instead, this data would be in the brands table and would be connected to a product by having a brand column in the products table.

**Figure 1**

*ER Diagram of T-Shirt Sales Database in Second Normal Form*



The third normal form (3NF) is met by first meeting the definition of the 2NF and also by ensuring that all non-key attributes do not depend on other non-key attributes (Kung & Tung, 2006). This form contains both primary keys and foreign keys. The concept of cardinality is brought into the design, too. The referenced T-Shirt sales database can be converted to the 3NF by constructing at least nine tables.

The tables for a 3NF version of the T-Shirt sales database include Statuses, Orders, OrderLines, Customers, ProductVariants, ShirtSizes, ShirtColors, Products, and Brands. Each of these tables contain a primary key and most use a foreign key to provide the relationships between entities.

The customers table contains a primary key that is a unique ID. This attribute is an integer type, because it does not need to be anything other than a whole number. The next four attributes are FirstName, LastName, Email, and Phone. These values will never exceed 50 characters, and for that reason a varchar is used with a limit of 50. The final attribute in this table is the Address. This is also an attribute that will need alphanumeric characters, but it may be longer than 50 characters. For this reason, a limit of 255 is used.

The Statuses table uses an integer to define its ID attribute. Both the ShortTitle and Description will be varchars, but the first can be written in 50 or less characters. Whereas the description will need to be longer and can contain up to 200 characters.

The Orders table uses an integer for its primary key attribute. This table has a foreign key that represents the CustomerID that made the order. This creates the many-to-one relationship between orders and customers. Another foreign key is the StatusID. This creates the many-to-one relationship between orders and statuses. The final three attributes are the OrderDate, ShipDate, and ArrivalDate. These are all represented with the datetime data type.

The OrderLines table uses two foreign keys to create its primary key. The OrderID and VariantID, both integers, tied to a line within an order will be unique when combined. The final attribute is the quantity of each item variant within an order. This will always be a whole number and is therefore represented with an integer data type.

Two tables are created to prevent incorrect size and colors being entered into the system. Both use integers for their ID attributes and 50-character varchars for representing the size and ColorName attributes respectively. These are then used as foreign keys in the ProductVariants table.

The ProductVariants table continues to use an integer for its ID field. The ProductID is brought in as a foreign key to represent the many-to-one relationship between variants and products. Finally, the quantity of stock on hand by the company is tracked with an integer.

The products table brings in the BrandID as a foreign key to represent the many-to-one relationship between products and brands. Both name and description are created with 255 character varchar. Finally, the price is represented by a float to allow for the tracking of fractional dollars.

**References**

Toby J. Teorey, Sam S. Lightstone, Tom Nadeau, & H.V. Jagadish. (2006). Database Modeling and Design : Logical Design: Vol. 4th ed. Morgan Kaufmann.

Hsiang-Jui Kung, & Hui-Lien Tung. (2006). An Alternative Approach to Teaching Database Normalization: A Simple Algorithm and an Interactive e-Learning Tool. Journal of Information Systems Education, 17(3), 315–325.

**Appendix A**

SQL Server Management Studio Screenshots

