**SQL-Online Retail Store**

**Student ID: 22082366**

**Name: Stanley Moses Gundapu**

**Abstract:**

This report documents the creation of a database for an online retail store, covering the schema design, data representation, and ethical considerations. The database consists of multiple tables, including Customers, Orders, Products, Order Items, and Categories, each serving a specific purpose in organising and managing store data. We discuss the significance of normalising data to maintain integrity, reduce redundancy, and enhance efficiency in storage and retrieval. Additionally, the report touches upon ethical considerations regarding data privacy and security in the context of online retail.

**Introduction:**

In the realm of e-commerce, efficient management of customer data, orders, products, and categories is crucial for the smooth operation of an online retail store. In this report, we delve into the process of generating a database for an online retail store using SQLite. We explore the rationale behind the database schema design, discuss the types of data represented in the database, and provide examples of queries to demonstrate its functionality.

**Database Generation:**

Generated data for an online retail store. I generated the data using SQLite and Python. The following are the tables in the database.

**Customers:**

Contains information about the customers, including their name, address, and contact information.

**Orders:**

Contains information about each order, including the order ID, customer ID (foreign key), order date, and total amount.

**Products:**

Contains information about the products available in the store, including the product ID, name, description, price, and quantity in stock.

**Order Items:**

This represents the relationship between orders and products, containing the quantity of each product ordered in a particular order.

**Categories:**

This section contains information about the categories of products available, including the category ID and name.

**Types of Data Represented:**

Nominal Data:

* Customer names, email addresses, product names, and category names are categorical data without any inherent order or ranking.
* These are stored as text (VARCHAR) data types in the database.

Interval Data:

* Order dates represent interval data.
* Numeric values are values where the difference between any two values is meaningful and consistent, but there is no true zero point.

Ratio Data:

* Prices, stock quantities, and total amounts represent ratio data.
* Numerical values with a true zero point, where the ratio between any two values is meaningful and consistent.
* Stored as ‘INTEGER’ or ‘REAL’ (floating-point) data types in the database.

**Database Schema:**

A white paper with text and numbers

Description automatically generated with medium confidence

A table of numbers and text

Description automatically generated with medium confidence

**Report Justification:**

* By separating related data into different tables, we can apply constraints and rules specific to each entity, ensuring that the data remains consistent and accurate.
* The Orders table has a foreign key customer that references the customer in the Customers table. This ensures that every order is associated with a valid customer.
* The Order Items table has foreign key order ID and product ID that reference the order ID and product ID in the Orders and Products tables, respectively. This ensures that every order item is associated with valid orders and products.
* By storing data in separate tables and establishing relationships between them, we can avoid storing redundant information.
* Reducing redundancy helps conserve storage space and improves data consistency by eliminating the risk of inconsistencies that can arise from duplicate data.
* For example, instead of storing customer information (e.g., name, address, phone number) in every order record, we store it once on the Customers table and reference it using foreign keys in the Orders table.
* Normalising the data by separating it into multiple tables reduces data duplication, leading to smaller table sizes and more efficient storage utilisation.
* Smaller table sizes and reduced redundancy also result in faster query performance and improved data retrieval times.
* For example, when querying order information, we only need to join the Orders table with the Customers table to retrieve customer details, rather than duplicating customer information in every order record.
* Ethically, it's important to consider data privacy and security when designing databases for online retail stores. I made sure to include only relevant information in the database, such as customer contact details and order history, while avoiding storing sensitive information like credit card numbers. I also implemented proper data encryption and access controls to protect customer data from unauthorised access.

**Example queries :**

1. Select all customers :

A screenshot of a computer

Description automatically generated

1. Select orders placed by a specific customer :

A screenshot of a computer

Description automatically generated

1. Join orders with customers to get customer details for each order:

A screenshot of a computer

Description automatically generated

1. Select products in a specific category:

A screenshot of a computer

Description automatically generated

1. Calculate total revenue:

A screenshot of a computer

Description automatically generated

**Conclusion:**

In conclusion, the database generation process for the online retail store exemplifies the importance of thoughtful schema design and data management techniques. By leveraging relational database principles such as normalisation and foreign key constraints, we ensure data integrity, minimise redundancy, and optimise performance.