Spring 2023 - CS 673: Scalable Databases Assignment #1

Select the problem / Define the problem.

A database is necessary for a restaurant to manage its orders, menus, and staff.

Create Tables with constraints

Design a Schema based on tables and explain the schema.

Create primary keys and foreign keys.

Schema Design: The database schema will have a number of tables that store details about various entities and the connections between them.

Table for customers: This table will store information about customers, including their names and phone numbers.

Menu Items: The menu items table will be used to record details about menu items, including their item id, name, and price.

Order table: Data about customer orders, including the order id, the customer id, the date of the order, and the total amount, will be kept in this table.

Order items table: This table will keep track of the order id, item id, and quantity of the things that consumers have ordered.

Employee table: This table will contain details on employees, including their names, job titles, and dates of hire.

Graphical user interface, text, application, email

Description automatically generatedGraphical user interface, text, application

Description automatically generated

Create Procedures.

A new item is added to the menu items table using the add menu item procedure. It takes three input parameters: item id, name, and price, which are used to set the values of the appropriate columns in the menu items table.

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Create functions.

Get order total is a MySQL function that takes an order id as input and returns a decimal number with a precision of 5 and a scale of 2  that represents the order's total cost. To keep the estimated total, the function first declares a variable total of type decimal(5,2). The sum of quantity times price for each item in the order with the specified order id is then used to determine the final total. The order items table and the menu items table are joined using the item id column to perform this calculation.

The function returns the calculated total using the RETURN statement after being stored in the total variable using the SELECT... INTO statement.

Graphical user interface, text, application

Description automatically generatedGraphical user interface, text, application

Description automatically generated

Create Views

This SQL command generates an order details view that presents client order details in a certain format. Orders, customers, order items, and menu items data from four tables are joined in a SELECT query to define the view. order id, customer name, date ordered, total amount, item name, quantity, and price columns are specified in the SELECT statement as the fields to be included in the view. The four tables being combined produce these columns. The INNER JOIN keyword is used to indicate the join condition, while the FROM clause specifies the tables to join. The customer id column in the orders table is connected to the customer id column in the customer’s table, the order id column in the orders table is connected to the order id column in the order items table, and the item id column in the order items table is connected to the item id column in the menu items table as part of the join condition in this case.

For each order in the system, the resulting view will provide the customer name and order data.

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Create Index

The "customers" table's "name" column will now have an index called "customer name idx" thanks to the first statement. The "name" field of the "customers" table will enable faster row searches thanks to this index.

The second statement adds an index with the name "order date idx" to the "orders" table's "date ordered" column. With the aid of this index, it will be quicker to search for rows in the "orders" table using the "date ordered" field.

Graphical user interface, text, application

Description automatically generated

Use of the following Clauses: Example: order by, between, group by, having, order by, AND, OR, with

Use Aggregate Functions

The names of menu items and their cumulative sales that are greater than ten are chosen by this SQL query. It arranges the outcomes according to total sales in descending order. It accomplishes this by connecting the menu items and order items tables based on their respective item IDs, grouping the outcomes based on the names of menu items, and summing the price and quantity for each group. The HAVING condition eliminates groups whose total sales are 10 or less.

Graphical user interface, text, application

Description automatically generated

This SQL statement builds a derived table containing all orders placed after September 4, 2022, using a common table expression (CTE) with the name "recent orders." The "recent orders" CTE is then joined with the "customers," "order items," and "menu items" fields to extract each customer's name and the sum of all orders placed after September 4, 2022. The price of each menu item is multiplied by the quantity requested, and the resulting numbers are added to determine the total amount. The statement then uses the GROUP BY clause to group the data by customer name.

Graphical user interface, text, application, email

Description automatically generated

This SQL statement returns all orders with a date ordered that is between September 3 and September 4, 2022. The results are filtered using the BETWEEN operator by supplying a range of values for the date ordered column. For the rows that satisfy the required criteria, the result set will contain all columns from the orders table.Graphical user interface, text, application, email

Description automatically generated

Use of nested queries, Scalar Subquery.

The name and prices of menu items with prices higher than the average are chosen using this SQL query. The average cost of every menu item in the menu items table is first determined by a subquery in the query. The menu items whose prices are higher than the average are then chosen by name and price from the same table.

Graphical user interface, text, application

Description automatically generated

Conclusion

The restaurant will be able to handle its menus, orders, and staff effectively. The performance will be enhanced and complex searches will be made simpler with the usage of procedures, functions, views, and indexes. The restaurant will be able to understand the company and make data-driven decisions using various clauses, aggregate functions, and nested inquiries.