Cognizant GenC Case Study

**Library Management System**

**Candidate ID:** 29854750

**Name:** B Jamin Enock

**Section 2: SQL Queries**

Basic Queries:

**CREATE DATABASE library;**

The command "CREATE DATABASE library;" sets up a digital storage space to organize and manage information for a library, facilitating efficient data handling and retrieval.

**USE library;**

The command "USE library;" selects the "library" database for subsequent operations, allowing access and modification of its stored data.

**CREATE TABLE books(book\_id INT,title VARCHAR(100),author VARCHAR(50),genre VARCHAR(50),price DECIMAL(10,2),stock INT,PRIMARY KEY(book\_id));**

The command "CREATE TABLE books" defines a structured format within the "library" database to store book details such as ID, title, author, genre, price, and stock, with "book\_id" as the primary key for uniquely identifying each book.

**CREATE TABLE borrowing\_id(borrowing\_id INT,book\_id INT,borrower\_id INT,quantity\_borrowed INT,borrowing\_date DATE,FOREIGN KEY(book\_id) REFERENCES books(book\_id),FOREIGN KEY(borrower\_id) REFERENCES borrowers(borrower\_id));**

The command "CREATE TABLE borrowing\_id" sets up a table in the database to manage borrowing records, including details such as borrowing ID, book ID, borrower ID, quantity borrowed, and borrowing date. It establishes foreign key constraints linking "book\_id" to the "books" table and "borrower\_id" to a hypothetical "borrowers" table for relational integrity.

**INSERT INTO books(book\_id,title,author,genre,price,stock) VALUES**

**(0,'The Silence of the Choir','Mohamed Mbougar Sarr','novel',200.20,10),**

**(1,'In Tongues','Thomas Grattan','fiction',300.40,4), (2,'Change','John Lambert','autobio',500,3),**

**(3,'Woman of Interest','Tracy O’Neill','nonfiction',100.30,5), (4,'Other Rivers','Peter Hessler','education',600.70,15);**

The command `INSERT INTO books` adds multiple rows of book data into the "books" table, including book ID, title, author, genre, price, and stock for each entry.

**SELECT \* FROM books;**

A screenshot of a computer

Description automatically generated

The query `SELECT \* FROM books;` retrieves all rows and columns from the "books" table, displaying details such as book ID, title, author, genre, price, and stock for each book stored in the database.

**INSERT INTO borrowers(borrower\_id,name,contact) VALUES**

**(0,'B Jamin Enock','8310652529'),**

**(1,'Jesin','9353545979'),**

**(2,'Suraj S','806574598'),**

**(3,'Rose','7568934527'),**

**(4,'Barnabas','9448303317');**

The SQL command `INSERT INTO borrowers` adds new records into the "borrowers" table, specifying each borrower's ID, name, and contact information. This operation is essential for maintaining a database of library users, allowing efficient management and retrieval of borrower details for transactions and communication purposes.

**SELECT \* FROM borrowers;**

A black and white screen with white text

Description automatically generated

The query `SELECT \* FROM borrowers;` retrieves all rows and columns from the "borrowers" table, displaying the borrower ID, name, and contact information for each borrower stored in the database.

**INSERT INTO borrowing(borrowing\_id,book\_id,borrower\_id,quantity\_borrowed,borrowing\_date) VALUES**

**(0,3,2,2,'2024-07-11'),**

**(1,2,4,3,'2024-07-12'),**

**(2,3,2,2,'2024-07-02'),**

**(3,4,1,0,'2024-07-04'),**

**(4,1,2,3,'2024-06-21');**

The SQL command `INSERT INTO borrowing` is used to add records into the "borrowing" table, specifying details such as borrowing ID, book ID, borrower ID, quantity borrowed, and borrowing date for each transaction. This operation allows the database to track and manage borrowing activities, facilitating efficient monitoring of book loans and associated borrower information.

**SELECT \* FROM borrowing;**

A screen shot of a computer

Description automatically generated

The query `SELECT \* FROM borrowing;` retrieves all rows and columns from the "borrowing" table, displaying details such as borrowing ID, book ID, borrower ID, quantity borrowed, and borrowing date for each borrowing transaction recorded in the database.

Questions:

1. Write a query to find the total quantity of each book borrowed.

**SELECT book\_id,SUM(quantity\_borrowed) FROM borrowing GROUP BY book\_id ORDER BY book\_id ASC;**

**A black screen with white text

Description automatically generated**

1. Write a query to find the book title and total quantity borrowed for each book.

**SELECT b.title,SUM(br.quantity\_borrowed) FROM books b INNER JOIN borrowing br WHERE b.book\_id = br.book\_id GROUP BY b.book\_id ORDER BY b.book\_id ASC;**

**A black screen with white text

Description automatically generated**

1. Write a query to find the titles of books that have never been borrowed.

**SELECT title FROM books WHERE book\_id NOT IN (SELECT book\_id FROM borrowing);**

**A black screen with white text

Description automatically generated**

1. Write a query to find the books that have been borrowed more than 10 times.

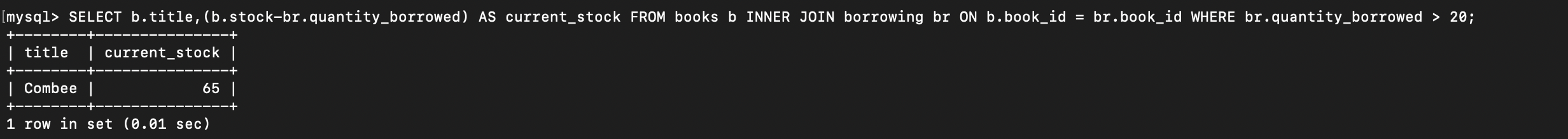
**SELECT \* FROM books WHERE book\_id IN (SELECT book\_id FROM borrowing WHERE quantity\_borrowed > 10);**

A screen shot of a black screen

Description automatically generated

1. Write a query to find the book titles and their current stock levels for books that have been borrowed more than 20 times.

**SELECT b.title,(b.stock-br.quantity\_borrowed) AS current\_stock FROM books b INNER JOIN borrowing br ON b.book\_id = br.book\_id WHERE br.quantity\_borrowed > 20;**

****