Insert Queries

Insert a new client:

```
INSERT INTO Client (client_id, client_name, contact_info, membership_type, account_status)
VALUES (1, ‘John Doe’, ‘johndoe@gmail.com’, ‘Regular’, ‘Active’);
```

Inserts Client listed into database.

Insert a new book:

```
INSERT INTO Book (book_id, title, author, isbn, publication_year, genre, avaliability_status,
popularity)
VALUES (111, 'The Great Gatsby', 'F. Scott Fitzgerald', '1234567890123', 1925, 'Classic',
'Available', 10);
```

Inserts Book listed into database. Even though we have the Great Gatsby already, the ID and ISBN is different, allowing it to be inserted.

Insert a new loan:

```
INSERT INTO Loan (loan_id, client_id, item_type, media_id, book_id, magazine_id,
borrow_date, due_date, return_date, fees_accrued);
VALUES (101, 1, ‘Book’, NULL, 101, NULL, CURRENT_DATE, DATE_ADD(CURRENT_DATE,
INTERVAL 14 DAY), NULL, 0.00);
```

Inserts Loan listed into database, attached to Client 1, Alice Johnson.

Insert a reservation:

```
INSERT INTO Reservation (reservation_id, client_id, item_type, media_id, book_id,
magazine_id, reservation_date, status, place_in_line);
VALUES (111, 1, ‘Book’, NULL, 1, NULL, CURRENT_DATE, ‘In line’, 6);
```

Inserts Reservation listed into database, attached to Client 1, Alice Johnson.The reservation is for book_id = 1, The Great Gatsby.

Update Queries

Updating a Client:

```
UPDATE Client
SET         contact_info = ‘johndoe1@gmail.com’
            membership_type = ‘Senior’
WHERE   client_id = 1;
```

Updates Client 1 ‘s contact information to be ‘johndoe1@gmail.com’ , and the membership type to be ‘Senior’.

Updating a Book:

```
UPDATE Book
SET         availability_status = ‘Reserved’
WHERE   book_id = 1;
```

Updates Book (id = 1, The Great Gatsby) to have a status of Reserved.

Search Queries

Search for books by title:

```
SELECT * FROM Book
WHERE title LIKE ‘%Gatsby%’;w
```

```
1|The Great Gatsby|F. Scott Fitzgerald|9780743273565|1925|Fiction|Available|8
```

Search for magazine by issue number:

```
SELECT * FROM Magazine
WHERE issue_number = 1;
```

```
1|National Geographic|2023-09|2023-09-01|Available|6
```

Search for media by type and genre:

```
SELECT * FROM Media
WHERE media_type = ‘Video’ AND genre = ‘Science;
```

```
4|Cosmos|Carl Sagan|Video|9780345331359|1980|Science|Available|6
```

Search for a client’s active loans:

```
SELECT *
FROM Loan
JOIN Client ON Loan.client_id = Client.client_id
WHERE Client.client_id = 1 AND Loan.return_date IS NULL;
```

```
1|1|Book||1||2024-04-01|2024-04-15||0.00|1|Alice
Johnson|alice@example.com|2023-05-01|Regular|Active
```

Notification Queries

Find loans due today:

```
SELECT client_id, loan_id
FROM Loan
WHERE due_date = CURRENT_DATE AND return_date IS NULL;
```

```
INSERT INTO Notification (notification_id, client_id, message)
VALUES (2001, 1, ‘Your loan is due today. Please return it by the end of the day to avoid late
fees.’);
```

Return null set

Find reservations where item became available:

```
SELECT Reservation.client_id, Reservation.reservation_id
FROM Reservation
JOIN Book ON Reservation.book_id = Book.book_id
WHERE Reservation.status = ‘In line’ AND Book.availiability_status = ‘Available’
ORDER BY place_in_line ASC;
```

```
INSERT INTO Notification (notification_id, client_id, message)
VALUES (2001, 1, ‘Your reservation is ready to pick up.’);
```

Return null set

Report Queries

Report for 10 most popular books:

```
SELECT book_id, title, author, popularity
FROM Book
```

```
ORDER BY popularity ASC
LIMIT 10;
```

```
4|The Catcher in the Rye|J.D.Salinger|7
1|The Great Gatsby|F. Scott Fitzgerald|8
5|Sapiens|Yuval Noah Hariri|8
3|To Kill a Mockingbird|Harper Lee|9
2|1984|George Orwell|10
```

Report for clients with outstanding fees:

```
SELECT Client.name, SUM(Loan.fees_accrued) AS total_fees
FROM Loan
JOIN Client ON Loan.client_id = Client.client_id
WHERE Loan.return_date IS NOT NULL
GROUP BY Client.client_id
HAVING total_fees > 0
ORDER BY total_fees DESC;
```

```
Carol White|0.30
```

Report for average borrowed items per membership type:

```
SELECT c.membership_type, COUNT(l.loan_id) / COUNT(DISTINCT c.client_id) AS
avg_loans_per_client
FROM Client as c
LEFT JOIN Loan as l on l.client_id = c.client_id
GROUP BY c.membership_type;
```

```
Regular | 1
```
