Shireen Taha
#500581584

Lab 4

Command Prompt - sqlite3 library.db

```
C:\Users\shireen>sqlite3 library.db
SQLite version 3.22.0 2018-01-22 18:45:57
Enter ".help" for usage hints.
sqlite> .dump
PRAGMA foreign_keys=OFF;
BEGIN TRANSACTION;
CREATE TABLE branch(
branchID INTEGER PRIMARY KEY not null,
brName varchar(255) not null,
address varchar(255) not null
);
INSERT INTO branch VALUES(1,'Toronto Public Library','350 Victoria Street, Toronto, ON');
CREATE TABLE Employee(
empID INTEGER PRIMARY KEY AUTOINCREMENT,
empName VARCHAR(255) NOT NULL,
Position VARCHAR(255) NOT NULL,
Salary REAL NOT NULL,
branchID INT NOT NULL,
CONSTRAINT ck_employee UNIQUE (empName),
CONSTRAINT fk_branch FOREIGN KEY (branchID) REFERENCES branch (branchID)
);
INSERT INTO Employee VALUES(1,'Mazen Bitar','Branch Manager',60000.0,1);
INSERT INTO Employee VALUES(2,'Shireen Taha','Database Manager',55000.0,1);
INSERT INTO Employee VALUES(3,'Farzana Saad','Information Specialist',68000.000000000000001,1);
INSERT INTO Employee VALUES(4,'Annie Yang','Network Administrator',60000.0,1);
INSERT INTO Employee VALUES(5,'Menan Parameswaran','Technical Services Manager',70000.0,1);
INSERT INTO Employee VALUES(6,'Fatima Hasan','Library Technician',48000.000000000000001,1);
INSERT INTO Employee VALUES(7,'Suruchi Arora','Computer Specialist',50000.0,1);
INSERT INTO Employee VALUES(8,'Yanani Saciharan','Administrative Assistant',43000.000000000000001,1);
INSERT INTO Employee VALUES(9,'Stephanie Treacy','Library Assistant',43000.000000000000001,1);
INSERT INTO Employee VALUES(10,'Emilia Cruze','Library Manager',69000.0,1);
CREATE TABLE Devices(
devID INTEGER PRIMARY KEY AUTOINCREMENT,
devName VARCHAR(255) NOT NULL,
Type VARCHAR(255) NOT NULL,
Status VARCHAR(255) NOT NULL,
empID INT NOT NULL,
cusID INT NOT NULL,
CONSTRAINT fk_employee FOREIGN KEY (empID) REFERENCES Employee (empID),
CONSTRAINT fk_customer FOREIGN KEY (cusID) REFERENCES Customer (cusID)
);
INSERT INTO Devices VALUES(1,'Laptop','Dell','In Use',5,6);
INSERT INTO Devices VALUES(2,'Laptop','Acer','In Use',5,1);
INSERT INTO Devices VALUES(3,'Tablet','Apple','Available',5,0);
INSERT INTO Devices VALUES(4,'Tablet','Windows','In Use',5,2);
INSERT INTO Devices VALUES(5,'Desktop Computer','Dell','In Use',5,3);
INSERT INTO Devices VALUES(6,'Desktop Computer','Windows','Available',5,0);
INSERT INTO Devices VALUES(7,'Printer','HP','In Use',5,4);
INSERT INTO Devices VALUES(8,'Printer','Dell','Available',5,0);
INSERT INTO Devices VALUES(9,'TV','Samsung','In Use',5,5);
INSERT INTO Devices VALUES(10,'TV','LG','Available',5,0);
INSERT INTO Devices VALUES(12,'Laptop','Dell','Returned',6,6);
INSERT INTO Devices VALUES(13,'Laptop','Acer','Returned',6,1);
INSERT INTO Devices VALUES(14,'Tablet','Windows','Returned',6,2);
INSERT INTO Devices VALUES(15,'Desktop Computer','Dell','Returned',6,3);
INSERT INTO Devices VALUES(16,'Printer','HP','Returned',6,4);
CREATE TABLE Books(
bookID INTEGER PRIMARY KEY AUTOINCREMENT,
Title VARCHAR(255) NOT NULL,
Author VARCHAR(255) NOT NULL,
Publisher VARCHAR(255) NOT NULL,
Genre VARCHAR(255) NOT NULL,
bkStatus VARCHAR(255) NOT NULL,
ISBN VARCHAR(255) NOT NULL,
empID INT NOT NULL,
cusID INT NOT NULL, Rating INT,
```

Shireen Taha
#500581584

Command Prompt - sqlite3 library.db

```
CREATE TABLE Books(
bookID INTEGER PRIMARY KEY AUTOINCREMENT,
Title VARCHAR(255) NOT NULL,
Author VARCHAR(255) NOT NULL,
Publisher VARCHAR(255) NOT NULL,
Genre VARCHAR(255) NOT NULL,
bkStatus VARCHAR(255) NOT NULL,
ISBN VARCHAR(255) NOT NULL,
empID INT NOT NULL,
cusID INT NOT NULL, Rating INT,
CONSTRAINT ck_books UNIQUE (ISBN),
CONSTRAINT fk_employee FOREIGN KEY (empID) REFERENCES Employee (empID),
CONSTRAINT fk_customer FOREIGN KEY (cusID) REFERENCES Customer (cusID)
);
INSERT INTO Books VALUES(3,'The Ministry of Utmost Happiness','Arundhati Roy','Alfred Knopf','Interpersonal Relations','Returned','9781524733155',6,3,1);
INSERT INTO Books VALUES(4,'Killers of the Flower Moon','David Grann','Doubleday','Crimes','Borrowed','9780385534246',5,1,2);
INSERT INTO Books VALUES(5,'The Hunger Games','Suzanne Collins','Scholastic Press','Science Fiction','Available','9780439023481',0,0,3);
INSERT INTO Books VALUES(6,'The Fault in Our Stars','John Green','Dutton Books','Romance','Borrowed','9780525478812',5,2,4);
INSERT INTO Books VALUES(7,'Divergent','Veronica Roth','Katherine Tegen Books','Science Fiction','Returned','9780062024039',6,3,5);
INSERT INTO Books VALUES(8,'City of Bones','Cassandra Clare','Margaret McElderry Books','Science Fiction','Borrowed','9781416914280',5,4,3);
INSERT INTO Books VALUES(9,'City of Glass','Cassandra Clare','Margaret McElderry Books','Science Fiction','Borrowed','9781416914303',5,4,3);
INSERT INTO Books VALUES(10,'Lady Midnight','Cassandra Clare','Margaret McElderry Books','Paranormal','Borrowed','9781442468351',5,4,4);
INSERT INTO Books VALUES(11,'The Perks of Being a Wallflower','Stephen Chbosky','MTV Books','Romance','Available','9780671027346',0,0,5);
INSERT INTO Books VALUES(12,'The Maze Runner','James Dashner','Delacorte Press','Dystopian','Borrowed','9780385737944',5,3,4);
INSERT INTO Books VALUES(13,'Thirteen Reasons Why','Jay Asher','Razorbill','Mystery','Borrowed','9781595147882',5,6,5);
INSERT INTO Books VALUES(14,'The Lightning Thief','Rick Riordan','Disney Hyperion Books','Fantasy','Borrowed','9780786838653',5,7,4);
INSERT INTO Books VALUES(15,'Uglies','Scott Westerfeld','Simon Pulse','Science Fiction','Borrowed','9780689865381',5,8,5);
INSERT INTO Books VALUES(16,'Graceling','Kristin Cashore','Harcourt','Romance','Borrowed','9780152063962',5,9,3);
INSERT INTO Books VALUES(17,'The Vampire Academy','Richelle Mead','Razorbill','Paranormal','Returned','9781595141743',6,3,4);
INSERT INTO Books VALUES(18,'Throne of Glass','Sarah Maas','Bloomsbury','Fantasy','Returned','9781599906959',6,3,4);
INSERT INTO Books VALUES(20,'Delirium','Lauren Oliver','HarperCollins','Dystopia','Returned','9780061726835',6,9,3);
INSERT INTO Books VALUES(22,'The Selection','Kiera Cass','HarperTeen','Romance','Returned','9780062059932',6,10,4);
INSERT INTO Books VALUES(23,'The Golden Compass','Philip Phullman','Alfred Knopf','Adventure','Returned','9780679879244',6,8,3);
INSERT INTO Books VALUES(24,'Speak','Laurie Anderson','Puffin','Contemporary','Returned','9780141310886',6,7,4);
INSERT INTO Books VALUES(25,'Matched','Ally Condie','Puffin','Science Fiction','Returned','9780525423645',6,6,3);
INSERT INTO Books VALUES(26,'If I Stay','Gayle Forman','Puffin','Romance','Returned','9780525421030',6,5,3);
INSERT INTO Books VALUES(27,'Looking for Alaska','John Green','Puffin','Contemporary','Available','9780142402511',0,0,4);
CREATE TABLE EmployeeRegistersCustomer(
empID INT NOT NULL,
cusID INT NOT NULL,
CONSTRAINT fk_employee FOREIGN KEY (empID) REFERENCES Employee (empID),
CONSTRAINT fk_customer FOREIGN KEY (cusID) REFERENCES Customer (cusID)
);
INSERT INTO EmployeeRegistersCustomer VALUES(2,1);
INSERT INTO EmployeeRegistersCustomer VALUES(2,2);
INSERT INTO EmployeeRegistersCustomer VALUES(2,3);
INSERT INTO EmployeeRegistersCustomer VALUES(2,4);
INSERT INTO EmployeeRegistersCustomer VALUES(2,5);
INSERT INTO EmployeeRegistersCustomer VALUES(2,6);
INSERT INTO EmployeeRegistersCustomer VALUES(2,7);
INSERT INTO EmployeeRegistersCustomer VALUES(2,8);
INSERT INTO EmployeeRegistersCustomer VALUES(2,9);
INSERT INTO EmployeeRegistersCustomer VALUES(2,10);
CREATE TABLE IssueStatus(
issueID INTEGER PRIMARY KEY AUTOINCREMENT,
bookISBN VARCHAR(255) NOT NULL,
bookTitle VARCHAR(255) NOT NULL,
customerID INT NOT NULL,
issueDate VARCHAR(255) NOT NULL,
deviceName VARCHAR(255) NOT NULL,
deviceType VARCHAR(255) NOT NULL,
CONSTRAINT ck_issueStatus UNIQUE (bookISBN)
);
INSERT INTO IssueStatus VALUES(1,'97803855342466','Killers of the Flower Moon',1,'March','Not Applicable','Not Applicable');
INSERT INTO IssueStatus VALUES(2,'9780525478812','The Fault in Our Stars',2,'March','Not Applicable','Not Applicable');
INSERT INTO IssueStatus VALUES(3,'9781416914280','City of Bones',4,'March','Not Applicable','Not Applicable');
```

Shireen Taha
#500581584

Command Prompt - sqlite3 library.db

```
INSERT INTO IssueStatus VALUES(1,'97803855342466','Killers of the Flower Moon',1,'March','Not Applicable','Not Applicable');
INSERT INTO IssueStatus VALUES(2,'9780525478812','The Fault in Our Stars',2,'March','Not Applicable','Not Applicable');
INSERT INTO IssueStatus VALUES(3,'9781416914280','City of Bones',4,'March','Not Applicable','Not Applicable');
INSERT INTO IssueStatus VALUES(4,'9781416914303','City of Glass',4,'March','Not Applicable','Not Applicable');
INSERT INTO IssueStatus VALUES(5,'9781442468351','Lady Midnight',4,'March','Not Applicable','Not Applicable');
INSERT INTO IssueStatus VALUES(6,'9780385737944','The Maze Runner',3,'April','Not Applicable','Not Applicable');
INSERT INTO IssueStatus VALUES(7,'9781595147882','Thirteen Reasons Why',6,'April','Not Applicable','Not Applicable');
INSERT INTO IssueStatus VALUES(8,'9780786838653','The Lightning Thief',7,'April','Not Applicable','Not Applicable');
INSERT INTO IssueStatus VALUES(9,'9780689865381','Uglies',8,'April','Not Applicable','Not Applicable');
INSERT INTO IssueStatus VALUES(10,'9780152063962','Graceling',9,'April','Not Applicable','Not Applicable');
INSERT INTO IssueStatus VALUES(11,'Not Applicable','Not Applicable',6,'March','Laptop','Dell');
INSERT INTO IssueStatus VALUES(12,'NA','Not Applicable',1,'April','Laptop','Acer');
CREATE TABLE EmployeeUpdatesIssueStatus(
empID INT NOT NULL,
issueID INT NOT NULL,
CONSTRAINT fk_employee FOREIGN KEY (empID) REFERENCES Employee (empID),
CONSTRAINT fk_issueStatus FOREIGN KEY (issueID) REFERENCES IssueStatus (issueID)
);
INSERT INTO EmployeeUpdatesIssueStatus VALUES(5,1);
INSERT INTO EmployeeUpdatesIssueStatus VALUES(5,2);
INSERT INTO EmployeeUpdatesIssueStatus VALUES(5,3);
INSERT INTO EmployeeUpdatesIssueStatus VALUES(5,4);
INSERT INTO EmployeeUpdatesIssueStatus VALUES(5,5);
INSERT INTO EmployeeUpdatesIssueStatus VALUES(5,6);
INSERT INTO EmployeeUpdatesIssueStatus VALUES(5,7);
INSERT INTO EmployeeUpdatesIssueStatus VALUES(5,8);
INSERT INTO EmployeeUpdatesIssueStatus VALUES(5,9);
INSERT INTO EmployeeUpdatesIssueStatus VALUES(5,10);
INSERT INTO EmployeeUpdatesIssueStatus VALUES(5,11);
INSERT INTO EmployeeUpdatesIssueStatus VALUES(5,12);
CREATE TABLE ReturnStatus(
returnID INTEGER PRIMARY KEY AUTOINCREMENT,
bkISBN VARCHAR(255) NOT NULL,
bkTitle VARCHAR(255) NOT NULL,
custID INT NOT NULL,
returnDate VARCHAR(255) NOT NULL,
devicName VARCHAR(255) NOT NULL,
devicType VARCHAR(255) NOT NULL,
CONSTRAINT ck_returnStatus UNIQUE (bkISBN)
);
INSERT INTO ReturnStatus VALUES(2,'9781524733155','The Ministry of Utmost Happiness',3,'April','Not Applicable','Not Applicable');
INSERT INTO ReturnStatus VALUES(3,'9780062024039','Divergent',3,'April','Not Applicable','Not Applicable');
INSERT INTO ReturnStatus VALUES(4,'9780061726835','Delirium',9,'April','Not Applicable','Not Applicable');
INSERT INTO ReturnStatus VALUES(5,'9780141310886','Speak',7,'April','Not Applicable','Not Applicable');
INSERT INTO ReturnStatus VALUES(6,'9780525423645','Matched',6,'April','Not Applicable','Not Applicable');
INSERT INTO ReturnStatus VALUES(7,'9780525421030','If I Stay',5,'April','Not Applicable','Not Applicable');
INSERT INTO ReturnStatus VALUES(8,'9780679879244','The Golden Compass',8,'April','Not Applicable','Not Applicable');
INSERT INTO ReturnStatus VALUES(9,'9780062059932','The Selection',10,'April','Not Applicable','Not Applicable');
INSERT INTO ReturnStatus VALUES(10,'Not Applicable','Not Applicable',3,'April','Desktop Computer','Dell');
INSERT INTO ReturnStatus VALUES(11,'NA','Not Applicable',4,'April','Printer','HP');
CREATE TABLE EmployeeUpdatesReturnStatus(
empID INT NOT NULL,
returnID INT NOT NULL,
CONSTRAINT fk_employee FOREIGN KEY (empID) REFERENCES Employee (empID),
CONSTRAINT fk_returnStatus FOREIGN KEY (returnID) REFERENCES ReturnStatus (returnID)
);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,2);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,3);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,4);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,5);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,6);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,7);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,8);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,9);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,10);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,11);
```

Shireen Taha
#500581584

```
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,2);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,3);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,4);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,5);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,6);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,7);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,8);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,9);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,10);
INSERT INTO EmployeeUpdatesReturnStatus VALUES(6,11);
CREATE TABLE IF NOT EXISTS "Customer"(
cusID INTEGER PRIMARY KEY AUTOINCREMENT,
cusName VARCHAR(255) NOT NULL,
cusAddress VARCHAR(255) NOT NULL,
RegistrationDate VARCHAR(255) NOT NULL,
branchID INT NOT NULL,
fineFee REAL,
CONSTRAINT ck_customer UNIQUE (cusName),
CONSTRAINT fk_branch FOREIGN KEY (branchID) REFERENCES branch (branchID)
);
INSERT INTO Customer VALUES(1,'Peter Randall','500 Kingston Rd, Toronto, ON','January 1, 2018',1,0.0);
INSERT INTO Customer VALUES(2,'Lailah Bradley','315 St Germain Ave, Toronto, ON','January 2, 2018',1,1.0);
INSERT INTO Customer VALUES(3,'Sam Valencia','48 St Clair Ave, Toronto, ON','January 3, 2018',1,2.0);
INSERT INTO Customer VALUES(4,'Terrance Sawyer','234 Willow Ave, Toronto, ON','January 4, 2018',1,3.0);
INSERT INTO Customer VALUES(5,'Veronica Page','26 Goodwood Park, Toronto, ON','January 5, 2018',1,4.0);
INSERT INTO Customer VALUES(6,'Alexandra Udinov','94 Queen St E, Toronto, ON','January 6, 2018',1,5.0);
INSERT INTO Customer VALUES(7,'Owen Matthews','24 Waverly Rd, Toronto, ON','January 7, 2018',1,3.5);
INSERT INTO Customer VALUES(8,'Michael Bishop','55 Berkeley St, Toronto, ON','January 8, 2018',1,4.5);
INSERT INTO Customer VALUES(9,'Nikita Meyers','70 Broadview Ave, Toronto, ON','January 9, 2018',1,6.0);
INSERT INTO Customer VALUES(10,'Ryan Fletcher','65 Don Mills Rd, Toronto, ON','January 10, 2018',1,9.0);
DELETE FROM sqlite_sequence;
INSERT INTO sqlite_sequence VALUES('Employee',10);
INSERT INTO sqlite_sequence VALUES('Devices',16);
INSERT INTO sqlite_sequence VALUES('Books',27);
INSERT INTO sqlite_sequence VALUES('Customer',10);
INSERT INTO sqlite_sequence VALUES('IssueStatus',12);
INSERT INTO sqlite_sequence VALUES('ReturnStatus',11);
COMMIT;
sqlite>
```

The queries listed in Lab 1 were modified and are focused on the management aspect of the library instead of customer side. Also, some of the tables were modified such that attributes (columns) were added.

Query 1: What are the highest rated books?

```
SELECT Title FROM Books
WHERE Rating = (SELECT max(Rating) FROM Books);
```

As per the SQL statements above, the title of the book with the highest rating from the table Books, is returned.

Query 2: What author do customers most commonly read?

```
SELECT Author FROM Books
GROUP BY Author
HAVING COUNT(*) = (SELECT MAX (Count)
FROM (SELECT COUNT(*) as Count FROM Books GROUP BY Author)
tmp);
```

As per the SQL statements above, from the table Books, the GROUP BY clause is used to look at the column Author, and find which author is borrowed the most by customers. Therefore HAVING is used to select the common value in the Author column of the Books table.

Query 3: What books have been borrowed in March?

```
SELECT bookTitle FROM IssueStatus
WHERE issueDate = 'March' AND bookISBN NOT LIKE 'Not Applicable';
```

As per the SQL statements above, the titles of the books are returned that have been borrowed in March. Therefore, the WHERE clause is used to check that the issueDate is March in the IssueStatus table, and that the value of bookISBN is not 'Not Applicable', since this value is only stored in the table when the devices, and not books, are issued.

Query 4: What devices have been borrowed in March?

```
SELECT deviceName FROM IssueStatus
WHERE issueDate = 'March' AND deviceName NOT LIKE 'Not Applicable';
```

As per the SQL statements above, the names of the devices are returned that have been borrowed in March. Therefore, the WHERE clause is used to check that the issueDate is March in the IssueStatus table, and that the value of deviceName is not 'Not Applicable', since this value is only stored in the table when the books, and not devices, are issued.

Query 5: What devices do customers most commonly use?

```
SELECT devName FROM Devices
GROUP BY devName
HAVING COUNT(*) = (SELECT MAX (Count)
FROM (SELECT COUNT(*) as Count FROM Devices GROUP BY devName)
tmp);
```

As per the SQL statements above, from the table Devices, the GROUP BY clause is used to look at the column devName, and find what devices are borrowed the most by customers. Therefore HAVING is used, to select the common value in the devName column of the Devices table.

Query 6: Which employee has the highest salary?

```
SELECT empName FROM Employee
WHERE Salary = (SELECT max(Salary) FROM Employee);
```

As per the SQL statements above, from the Employee table, the name of the employee with the highest salary is selected to be returned, Therefore, the WHERE clause is used to select the max value in the Salary column, which is then returned.

Shireen Taha
#500581584

Query 7: Which genre is the most commonly read genre?

```
SELECT Genre FROM Books
GROUP BY Genre
HAVING COUNT(*) = (SELECT MAX (Count)
FROM (SELECT COUNT(*) as Count FROM Books GROUP BY Genre)
tmp) AND bkStatus NOT LIKE 'Available';
```

As per the SQL statements above, from the table Books, the GROUP BY clause is used to look at the column Genre, and find what genre is most commonly read by customers. Therefore HAVING is used, to select the common value in the Genre column of the Books table.

Query 8: Which customer has the highest fine?

```
SELECT cusName FROM Customer
WHERE fineFee = (SELECT max(fineFee) FROM Customer);
```

As per the SQL statements above, from the Customer table, the name of the customer with the highest fine is selected to be returned, Therefore, the WHERE clause is used to select the max value in the fineFee column, which is then returned.

Query 9: Which customer borrows the same author's books often?

```
SELECT Customer.cusName FROM Customer INNER JOIN Books ON Customer.cusID =
Books.cusID
GROUP BY Author
HAVING COUNT(*) = (SELECT MAX (Count)
FROM (SELECT COUNT(*) as Count FROM Books GROUP BY Author)
tmp) AND (SELECT DISTINCT cusID FROM Books);
```

This query is similar to some of the queries mentioned above, except the difference is, that since the query is asking for the customer name, the tables Customer and Books have to be inner joined in order to find the common author read by customers in the Books table, and then to return the name of the customer from the Customer table, who reads that author.

Query 10: Which customer reads the most?

```
SELECT Customer.cusName FROM Customer INNER JOIN Books ON Customer.cusID =
Books.cusID
GROUP BY Books.cusID
HAVING COUNT(*) = (SELECT MAX (Count)
FROM (SELECT COUNT(*) as Count FROM Books GROUP BY cusID)
```

tmp) AND bkStatus NOT LIKE ‘Available’;
In this case, the tables Customer and Books are inner joined to find the name of the customer who borrows more than the rest of the customers. Therefore, Group By clause is used on the cusID column in the Books table to find the common value when the bkStatus does not have a value of ‘Available’.