DataBase Programming - Assignment: Creating & Manipulating a Database

Written by Nicholas D. Mensah

Western Governors University

Study.com

Computer Science 204: Database Programming

8/12/2022

Abstract

One paragraph, not indented. Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan.

Keywords: First line indented, lorem, ipsum, dolor

DataBase Programming - Assignment: Creating & Manipulating a Database

The Entity Relationship Diagram assists in determining the relationship to each table. From this table's alzyasis. Each table has primary keys which are BorrowId, ClientId, BookId and AuthorId. The borrower table contains foreign keys of clientId and bookId which allows it to have a many to many relation with the client table. The client table has a one to one relationship with the borrower table. The connection the book has results from a many to one relation from the borrower to the book table. The author table is a dependant to the book table due to the author and book author connection

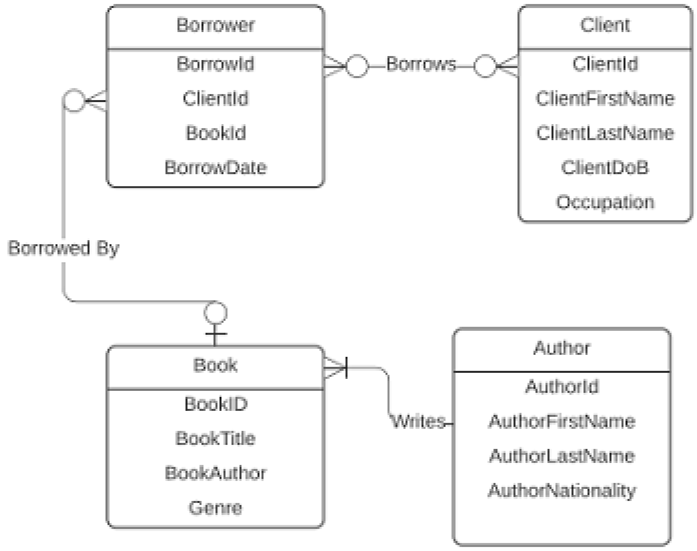
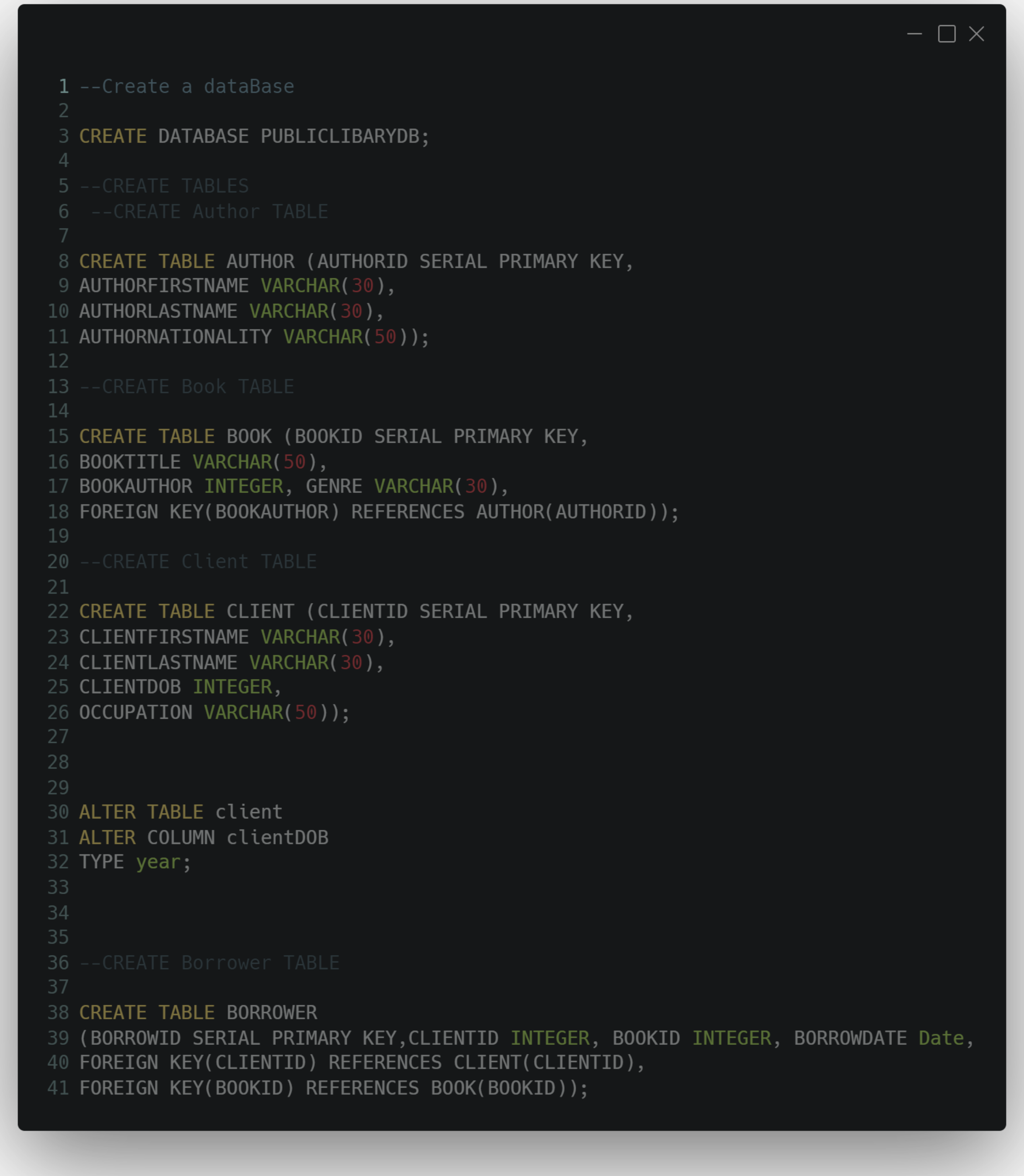


Figure 1: ERD for Library Database

CREATING DATABASE & UNIQUENESS



```sql
--Create a dataBase

CREATE DATABASE PUBLICLIBARYDB;

--CREATE TABLES
 --CREATE Author TABLE

CREATE TABLE AUTHOR (AUTHORID SERIAL PRIMARY KEY,
AUTHORFIRSTNAME VARCHAR(30),
AUTHORLASTNAME VARCHAR(30),
AUTHORNATIONALITY VARCHAR(50));

--CREATE Book TABLE

CREATE TABLE BOOK (BOOKID SERIAL PRIMARY KEY,
BOOKTITLE VARCHAR(50),
BOOKAUTHOR INTEGER, GENRE VARCHAR(30),
FOREIGN KEY(BOOKAUTHOR) REFERENCES AUTHOR(AUTHORID));

--CREATE Client TABLE

CREATE TABLE CLIENT (CLIENTID SERIAL PRIMARY KEY,
CLIENTFIRSTNAME VARCHAR(30),
CLIENTLASTNAME VARCHAR(30),
CLIENTDOB INTEGER,
OCCUPATION VARCHAR(50));



ALTER TABLE client
ALTER COLUMN clientDOB
TYPE year;



--CREATE Borrower TABLE

CREATE TABLE BORROWER
(BORROWID SERIAL PRIMARY KEY,CLIENTID INTEGER, BOOKID INTEGER, BORROWDATE Date,
FOREIGN KEY(CLIENTID) REFERENCES CLIENT(CLIENTID),
FOREIGN KEY(BOOKID) REFERENCES BOOK(BOOKID));
```

We by creating a database in the server named PUBLIC LIBRARY by using the the key word of :

CREATE DATABASE public Library DB;

Next we will add CONSTRAINTs and check for uniqueness based on the ERD. we will create TABLE USING THE :

CREATE TABLE table_name

Then we will set up the Columns or fields with the field name followed by the Type For example integer, VARCHAR(max), and or Dates. The varchar will be the variety of characters the values a value can have most of the time it has a maximum value. We usually use 30 or 50 being we are using shorter characters including the spaces.

After, tables were established I INSERT VALUES INTO the table using the :

INSERT INTO table_name(field, field, field, fild…..)

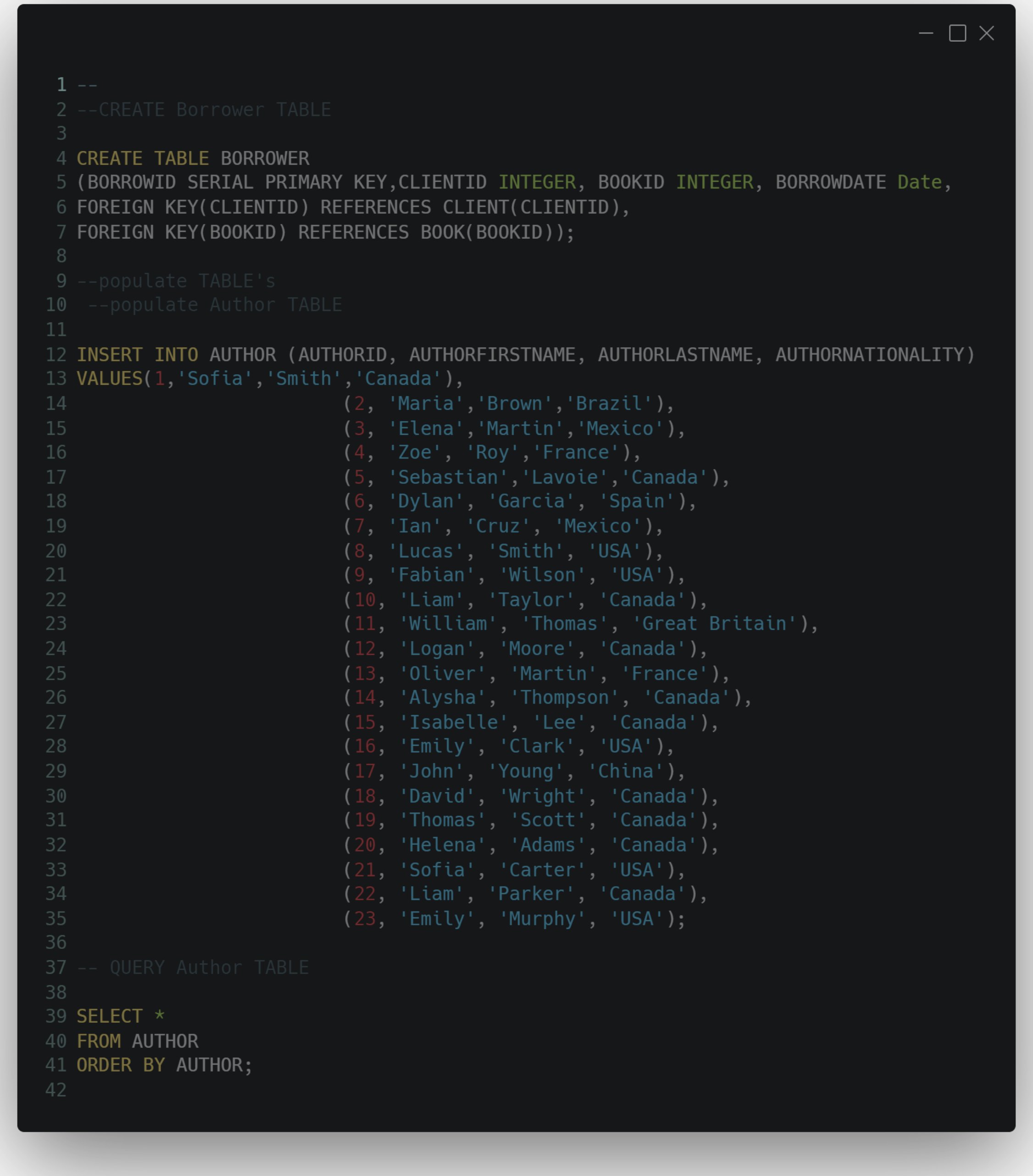
Now, we prepare the values to insert into the rows to produce a record based on the fields that have tested for uniqueness and established connection based on PK, FK, NULL, and other constraints:

VALUES(1,'Sofia','Smith','Canada'),

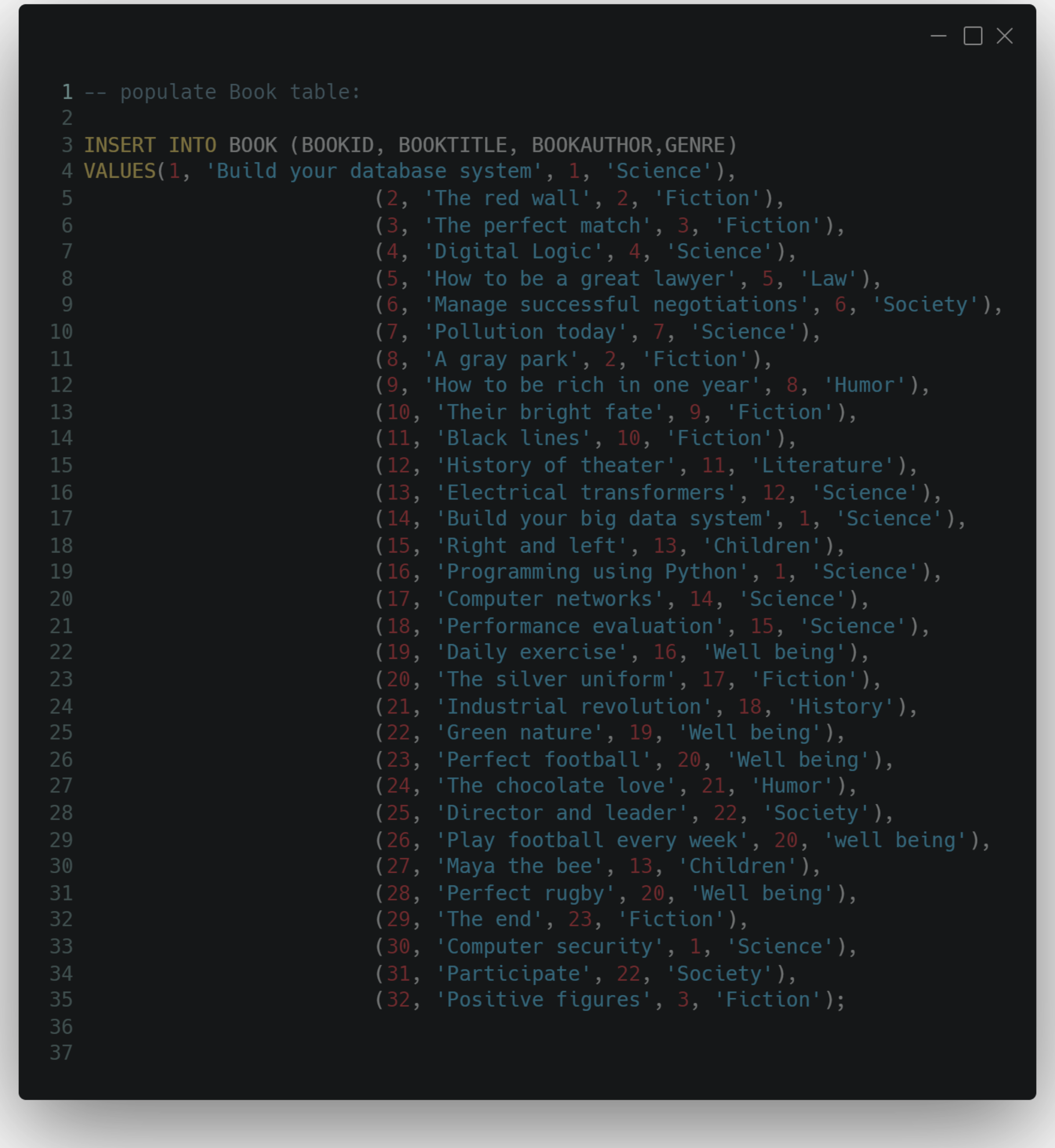
(2, 'Maria','Brown','Brazil'),

(3, 'Elena','Martin','Mexico'),

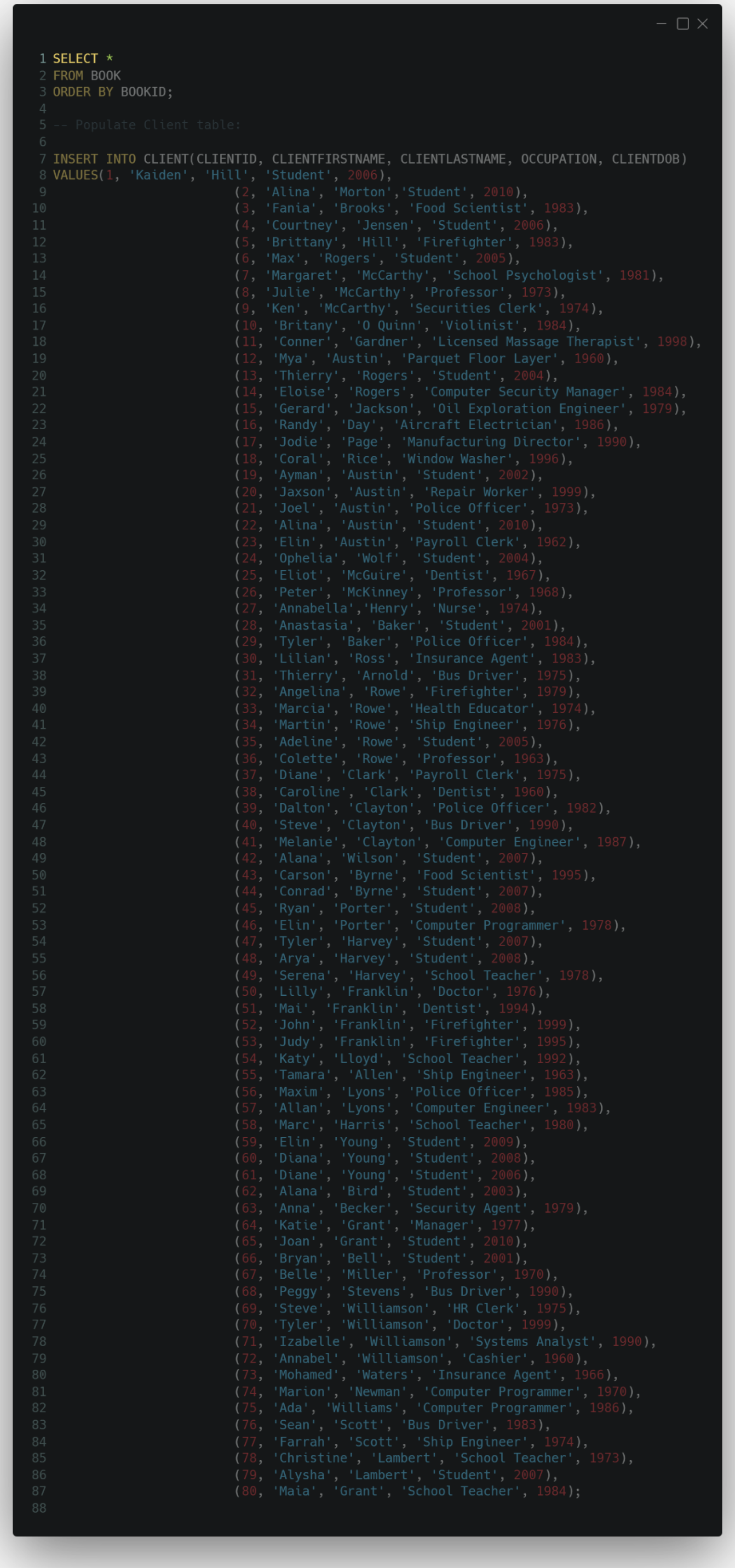
After, each insertion of the tables values a simple query verifies the records and fields



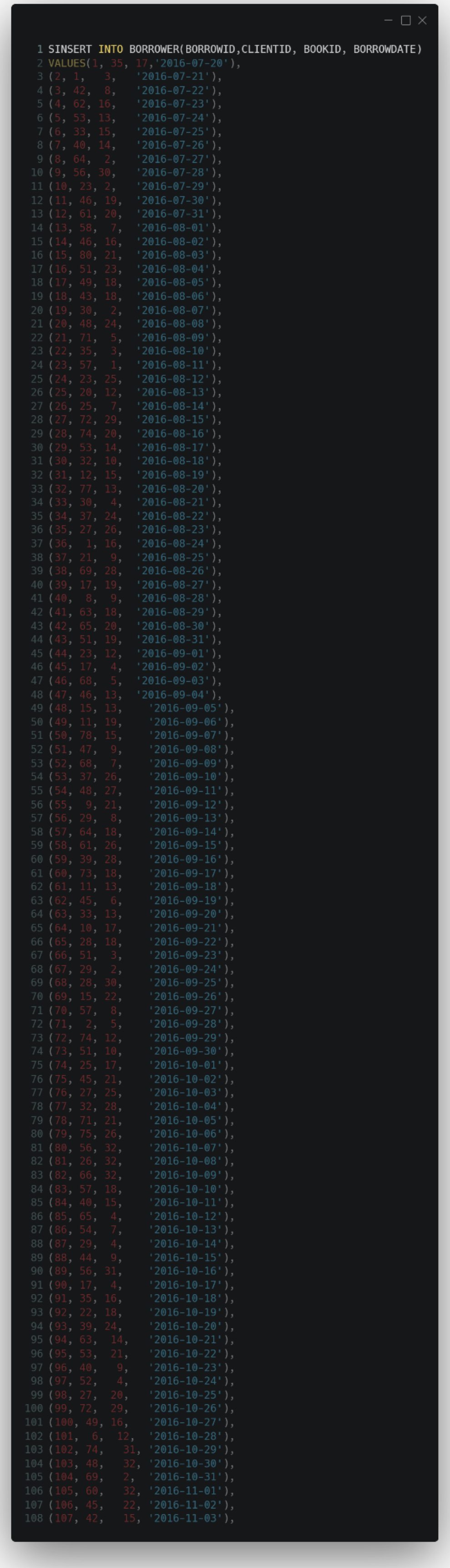
```sql
--
--CREATE Borrower TABLE

CREATE TABLE BORROWER
(BORROWID SERIAL PRIMARY KEY,CLIENTID INTEGER, BOOKID INTEGER, BORROWDATE Date,
FOREIGN KEY(CLIENTID) REFERENCES CLIENT(CLIENTID),
FOREIGN KEY(BOOKID) REFERENCES BOOK(BOOKID));

--populate TABLE's
 --populate Author TABLE

INSERT INTO AUTHOR (AUTHORID, AUTHORFIRSTNAME, AUTHORLASTNAME, AUTHORNATIONALITY)
VALUES(1,'Sofia','Smith','Canada'),
                        (2, 'Maria','Brown','Brazil'),
                        (3, 'Elena','Martin','Mexico'),
                        (4, 'Zoe', 'Roy','France'),
                        (5, 'Sebastian','Lavoie','Canada'),
                        (6, 'Dylan', 'Garcia', 'Spain'),
                        (7, 'Ian', 'Cruz', 'Mexico'),
                        (8, 'Lucas', 'Smith', 'USA'),
                        (9, 'Fabian', 'Wilson', 'USA'),
                        (10, 'Liam', 'Taylor', 'Canada'),
                        (11, 'William', 'Thomas', 'Great Britain'),
                        (12, 'Logan', 'Moore', 'Canada'),
                        (13, 'Oliver', 'Martin', 'France'),
                        (14, 'Alysha', 'Thompson', 'Canada'),
                        (15, 'Isabelle', 'Lee', 'Canada'),
                        (16, 'Emily', 'Clark', 'USA'),
                        (17, 'John', 'Young', 'China'),
                        (18, 'David', 'Wright', 'Canada'),
                        (19, 'Thomas', 'Scott', 'Canada'),
                        (20, 'Helena', 'Adams', 'Canada'),
                        (21, 'Sofia', 'Carter', 'USA'),
                        (22, 'Liam', 'Parker', 'Canada'),
                        (23, 'Emily', 'Murphy', 'USA');

-- QUERY Author TABLE

SELECT *
FROM AUTHOR
ORDER BY AUTHOR;

```



```sql
-- populate Book table:

INSERT INTO BOOK (BOOKID, BOOKTITLE, BOOKAUTHOR,GENRE)
VALUES(1, 'Build your database system', 1, 'Science'),
                        (2, 'The red wall', 2, 'Fiction'),
                        (3, 'The perfect match', 3, 'Fiction'),
                        (4, 'Digital Logic', 4, 'Science'),
                        (5, 'How to be a great lawyer', 5, 'Law'),
                        (6, 'Manage successful negotiations', 6, 'Society'),
                        (7, 'Pollution today', 7, 'Science'),
                        (8, 'A gray park', 2, 'Fiction'),
                        (9, 'How to be rich in one year', 8, 'Humor'),
                        (10, 'Their bright fate', 9, 'Fiction'),
                        (11, 'Black lines', 10, 'Fiction'),
                        (12, 'History of theater', 11, 'Literature'),
                        (13, 'Electrical transformers', 12, 'Science'),
                        (14, 'Build your big data system', 1, 'Science'),
                        (15, 'Right and left', 13, 'Children'),
                        (16, 'Programming using Python', 1, 'Science'),
                        (17, 'Computer networks', 14, 'Science'),
                        (18, 'Performance evaluation', 15, 'Science'),
                        (19, 'Daily exercise', 16, 'Well being'),
                        (20, 'The silver uniform', 17, 'Fiction'),
                        (21, 'Industrial revolution', 18, 'History'),
                        (22, 'Green nature', 19, 'Well being'),
                        (23, 'Perfect football', 20, 'Well being'),
                        (24, 'The chocolate love', 21, 'Humor'),
                        (25, 'Director and leader', 22, 'Society'),
                        (26, 'Play football every week', 20, 'well being'),
                        (27, 'Maya the bee', 13, 'Children'),
                        (28, 'Perfect rugby', 20, 'Well being'),
                        (29, 'The end', 23, 'Fiction'),
                        (30, 'Computer security', 1, 'Science'),
                        (31, 'Participate', 22, 'Society'),
                        (32, 'Positive figures', 3, 'Fiction');


```

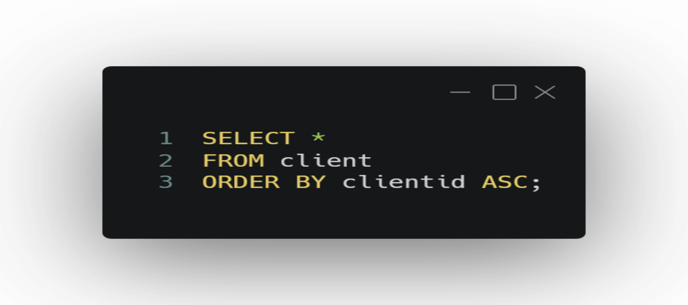


```sql
SELECT *
FROM BOOK
ORDER BY BOOKID;

-- Populate Client table:

INSERT INTO CLIENT(CLIENTID, CLIENTFIRSTNAME, CLIENTLASTNAME, OCCUPATION, CLIENTDOB)
VALUES(1, 'Kaiden', 'Hill', 'Student', 2006),
                        (2, 'Alina', 'Morton','Student', 2010),
                        (3, 'Fania', 'Brooks', 'Food Scientist', 1983),
                        (4, 'Courtney', 'Jensen', 'Student', 2006),
                        (5, 'Brittany', 'Hill', 'Firefighter', 1983),
                        (6, 'Max', 'Rogers', 'Student', 2005),
                        (7, 'Margaret', 'McCarthy', 'School Psychologist', 1981),
                        (8, 'Julie', 'McCarthy', 'Professor', 1973),
                        (9, 'Ken', 'McCarthy', 'Securities Clerk', 1974),
                        (10, 'Britany', 'O Quinn', 'Violinist', 1984),
                        (11, 'Conner', 'Gardner', 'Licensed Massage Therapist', 1998),
                        (12, 'Mya', 'Austin', 'Parquet Floor Layer', 1960),
                        (13, 'Thierry', 'Rogers', 'Student', 2004),
                        (14, 'Eloise', 'Rogers', 'Computer Security Manager', 1984),
                        (15, 'Gerard', 'Jackson', 'Oil Exploration Engineer', 1979),
                        (16, 'Randy', 'Day', 'Aircraft Electrician', 1986),
                        (17, 'Jodie', 'Page', 'Manufacturing Director', 1990),
                        (18, 'Coral', 'Rice', 'Window Washer', 1996),
                        (19, 'Ayman', 'Austin', 'Student', 2002),
                        (20, 'Jaxson', 'Austin', 'Repair Worker', 1999),
                        (21, 'Joel', 'Austin', 'Police Officer', 1973),
                        (22, 'Alina', 'Austin', 'Student', 2010),
                        (23, 'Elin', 'Austin', 'Payroll Clerk', 1962),
                        (24, 'Ophelia', 'Wolf', 'Student', 2004),
                        (25, 'Eliot', 'McGuire', 'Dentist', 1967),
                        (26, 'Peter', 'McKinney', 'Professor', 1968),
                        (27, 'Annabella','Henry', 'Nurse', 1974),
                        (28, 'Anastasia', 'Baker', 'Student', 2001),
                        (29, 'Tyler', 'Baker', 'Police Officer', 1984),
                        (30, 'Lilian', 'Ross', 'Insurance Agent', 1983),
                        (31, 'Thierry', 'Arnold', 'Bus Driver', 1975),
                        (32, 'Angelina', 'Rowe', 'Firefighter', 1979),
                        (33, 'Marcia', 'Rowe', 'Health Educator', 1974),
                        (34, 'Martin', 'Rowe', 'Ship Engineer', 1976),
                        (35, 'Adeline', 'Rowe', 'Student', 2005),
                        (36, 'Colette', 'Rowe', 'Professor', 1963),
                        (37, 'Diane', 'Clark', 'Payroll Clerk', 1975),
                        (38, 'Caroline', 'Clark', 'Dentist', 1960),
                        (39, 'Dalton', 'Clayton', 'Police Officer', 1982),
                        (40, 'Steve', 'Clayton', 'Bus Driver', 1990),
                        (41, 'Melanie', 'Clayton', 'Computer Engineer', 1987),
                        (42, 'Alana', 'Wilson', 'Student', 2007),
                        (43, 'Carson', 'Byrne', 'Food Scientist', 1995),
                        (44, 'Conrad', 'Byrne', 'Student', 2007),
                        (45, 'Ryan', 'Porter', 'Student', 2008),
                        (46, 'Elin', 'Porter', 'Computer Programmer', 1978),
                        (47, 'Tyler', 'Harvey', 'Student', 2007),
                        (48, 'Arya', 'Harvey', 'Student', 2008),
                        (49, 'Serena', 'Harvey', 'School Teacher', 1978),
                        (50, 'Lilly', 'Franklin', 'Doctor', 1976),
                        (51, 'Mai', 'Franklin', 'Dentist', 1994),
                        (52, 'John', 'Franklin', 'Firefighter', 1999),
                        (53, 'Judy', 'Franklin', 'Firefighter', 1995),
                        (54, 'Katy', 'Lloyd', 'School Teacher', 1992),
                        (55, 'Tamara', 'Allen', 'Ship Engineer', 1963),
                        (56, 'Maxim', 'Lyons', 'Police Officer', 1985),
                        (57, 'Allan', 'Lyons', 'Computer Engineer', 1983),
                        (58, 'Marc', 'Harris', 'School Teacher', 1980),
                        (59, 'Elin', 'Young', 'Student', 2009),
                        (60, 'Diana', 'Young', 'Student', 2008),
                        (61, 'Diane', 'Young', 'Student', 2006),
                        (62, 'Alana', 'Bird', 'Student', 2003),
                        (63, 'Anna', 'Becker', 'Security Agent', 1979),
                        (64, 'Katie', 'Grant', 'Manager', 1977),
                        (65, 'Joan', 'Grant', 'Student', 2010),
                        (66, 'Bryan', 'Bell', 'Student', 2001),
                        (67, 'Belle', 'Miller', 'Professor', 1970),
                        (68, 'Peggy', 'Stevens', 'Bus Driver', 1990),
                        (69, 'Steve', 'Williamson', 'HR Clerk', 1975),
                        (70, 'Tyler', 'Williamson', 'Doctor', 1999),
                        (71, 'Izabelle', 'Williamson', 'Systems Analyst', 1990),
                        (72, 'Annabel', 'Williamson', 'Cashier', 1960),
                        (73, 'Mohamed', 'Waters', 'Insurance Agent', 1966),
                        (74, 'Marion', 'Newman', 'Computer Programmer', 1970),
                        (75, 'Ada', 'Williams', 'Computer Programmer', 1986),
                        (76, 'Sean', 'Scott', 'Bus Driver', 1983),
                        (77, 'Farrah', 'Scott', 'Ship Engineer', 1974),
                        (78, 'Christine', 'Lambert', 'School Teacher', 1973),
                        (79, 'Alysha', 'Lambert', 'Student', 2007),
                        (80, 'Maia', 'Grant', 'School Teacher', 1984);
```



```
SINSERT INTO BORROWER(BORROWID,CLIENTID, BOOKID, BORROWDATE)
VALUES(1, 35, 17,'2016-07-20'),
(2, 1,   3,   '2016-07-21'),
(3, 42,  8,   '2016-07-22'),
(4, 62, 16,   '2016-07-23'),
(5, 53, 13,   '2016-07-24'),
(6, 33, 15,   '2016-07-25'),
(7, 40, 14,   '2016-07-26'),
(8, 64,  2,   '2016-07-27'),
(9, 56, 30,   '2016-07-28'),
(10, 23, 2,   '2016-07-29'),
(11, 46, 19,  '2016-07-30'),
(12, 61, 20,  '2016-07-31'),
(13, 58,  7,  '2016-08-01'),
(14, 46, 16,  '2016-08-02'),
(15, 80, 21,  '2016-08-03'),
(16, 51, 23,  '2016-08-04'),
(17, 49, 18,  '2016-08-05'),
(18, 43, 18,  '2016-08-06'),
(19, 30,  2,  '2016-08-07'),
(20, 48, 24,  '2016-08-08'),
(21, 71,  5,  '2016-08-09'),
(22, 35,  3,  '2016-08-10'),
(23, 57,  1,  '2016-08-11'),
(24, 23, 25,  '2016-08-12'),
(25, 20, 12,  '2016-08-13'),
(26, 25,  7,  '2016-08-14'),
(27, 72, 29,  '2016-08-15'),
(28, 74, 20,  '2016-08-16'),
(29, 53, 14,  '2016-08-17'),
(30, 32, 10,  '2016-08-18'),
(31, 12, 15,  '2016-08-19'),
(32, 77, 13,  '2016-08-20'),
(33, 30,  4,  '2016-08-21'),
(34, 37, 24,  '2016-08-22'),
(35, 27, 26,  '2016-08-23'),
(36,  1, 16,  '2016-08-24'),
(37, 21,  9,  '2016-08-25'),
(38, 69, 28,  '2016-08-26'),
(39, 17, 19,  '2016-08-27'),
(40,  8,  9,  '2016-08-28'),
(41, 63, 18,  '2016-08-29'),
(42, 65, 20,  '2016-08-30'),
(43, 51, 19,  '2016-08-31'),
(44, 23, 12,  '2016-09-01'),
(45, 17,  4,  '2016-09-02'),
(46, 68,  5,  '2016-09-03'),
(47, 46, 13,  '2016-09-04'),
(48, 15, 13,    '2016-09-05'),
(49, 11, 19,    '2016-09-06'),
(50, 78, 15,    '2016-09-07'),
(51, 47,  9,    '2016-09-08'),
(52, 68,  7,    '2016-09-09'),
(53, 37, 26,    '2016-09-10'),
(54, 48, 27,    '2016-09-11'),
(55,  9, 21,    '2016-09-12'),
(56, 29,  8,    '2016-09-13'),
(57, 64, 18,    '2016-09-14'),
(58, 61, 26,    '2016-09-15'),
(59, 39, 28,    '2016-09-16'),
(60, 73, 18,    '2016-09-17'),
(61, 11, 13,    '2016-09-18'),
(62, 45,  6,    '2016-09-19'),
(63, 33, 13,    '2016-09-20'),
(64, 10, 17,    '2016-09-21'),
(65, 28, 18,    '2016-09-22'),
(66, 51,  3,    '2016-09-23'),
(67, 29,  2,    '2016-09-24'),
(68, 28, 30,    '2016-09-25'),
(69, 15, 22,    '2016-09-26'),
(70, 57,  8,    '2016-09-27'),
(71,  2,  5,    '2016-09-28'),
(72, 74, 12,    '2016-09-29'),
(73, 51, 10,    '2016-09-30'),
(74, 25, 17,    '2016-10-01'),
(75, 45, 21,    '2016-10-02'),
(76, 27, 25,    '2016-10-03'),
(77, 32, 28,    '2016-10-04'),
(78, 71, 21,    '2016-10-05'),
(79, 75, 26,    '2016-10-06'),
(80, 56, 32,    '2016-10-07'),
(81, 26, 32,    '2016-10-08'),
(82, 66, 32,    '2016-10-09'),
(83, 57, 18,    '2016-10-10'),
(84, 40, 15,    '2016-10-11'),
(85, 65,  4,    '2016-10-12'),
(86, 54,  7,    '2016-10-13'),
(87, 29,  4,    '2016-10-14'),
(88, 44,  9,    '2016-10-15'),
(89, 56, 31,    '2016-10-16'),
(90, 17,  4,    '2016-10-17'),
(91, 35, 16,    '2016-10-18'),
(92, 22, 18,    '2016-10-19'),
(93, 39, 24,    '2016-10-20'),
(94, 63,  14,   '2016-10-21'),
(95, 53,  21,   '2016-10-22'),
(96, 40,   9,   '2016-10-23'),
(97, 52,   4,   '2016-10-24'),
(98, 27,  20,   '2016-10-25'),
(99, 72,  29,   '2016-10-26'),
(100, 49, 16,   '2016-10-27'),
(101,  6,  12,  '2016-10-28'),
(102, 74,   31, '2016-10-29'),
(103, 48,   32, '2016-10-30'),
(104, 69,   2,  '2016-10-31'),
(105, 60,   32, '2016-11-01'),
(106, 45,   22, '2016-11-02'),
(107, 42,   15, '2016-11-03'),
```

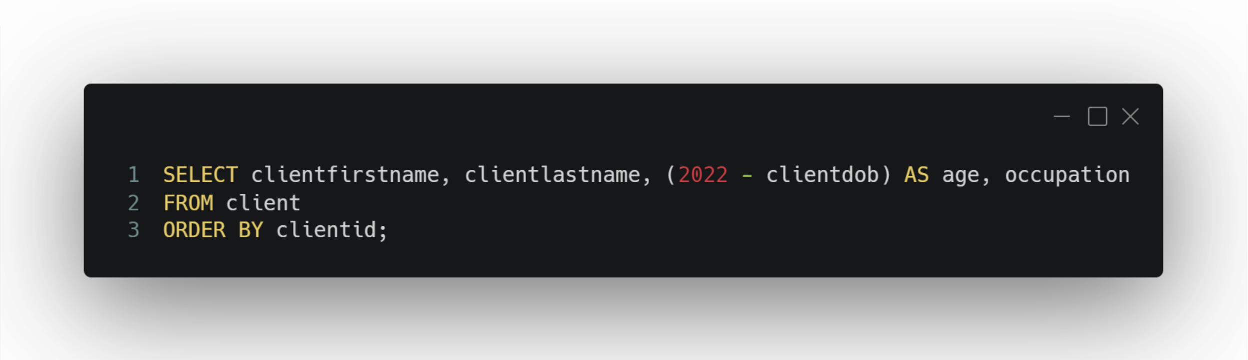
1) Display(\d) all contents from the client TABLE/



```
SELECT *
FROM client
ORDER BY clientid ASC;
```

The first query wanted us to return all of the fields which are represented by an asterisk, From the client Table. The query was organized by the primary clientId in ascending order. The result-set returns 80 records.

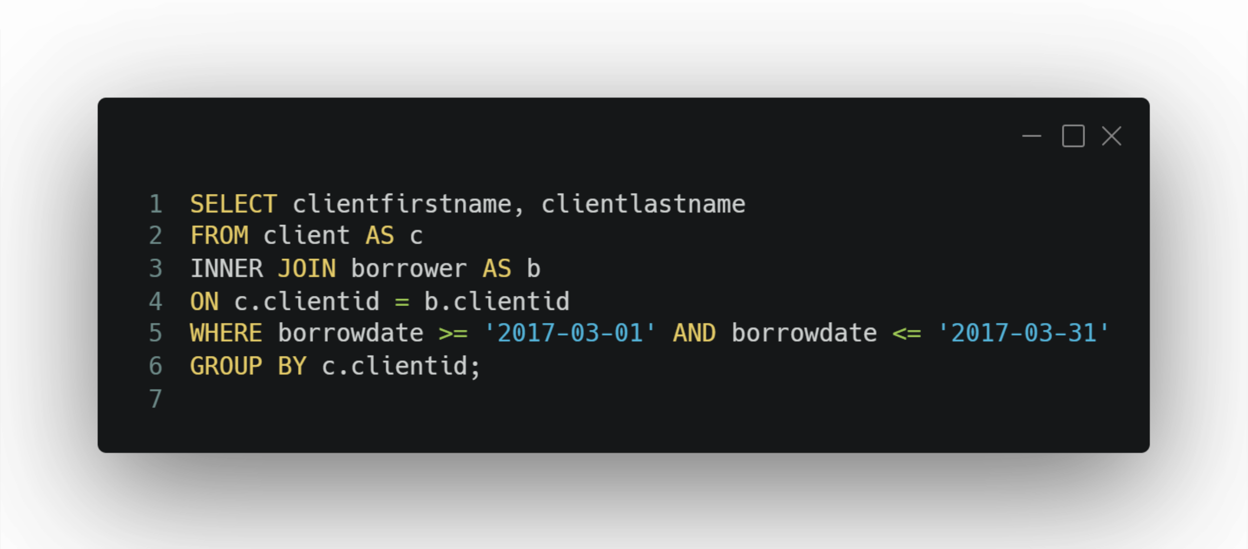
2) -- First names, last names, ages and occupations of all clients/



```
SELECT clientfirstname, clientlastname, (2022 - clientdob) AS age, occupation
FROM client
ORDER BY clientid;
```

Next, the second query SELECT's the fields of the client first name, the clients last name and the age ALIASed (AS) to a field(column) from subtracting the current year from the integer type from the clientDob. The fields are obtained from the client Table Organized by the clientId in ascending order. ASC was omitted because the default when ordering fields are by ascending order. The result set returns 80 records.

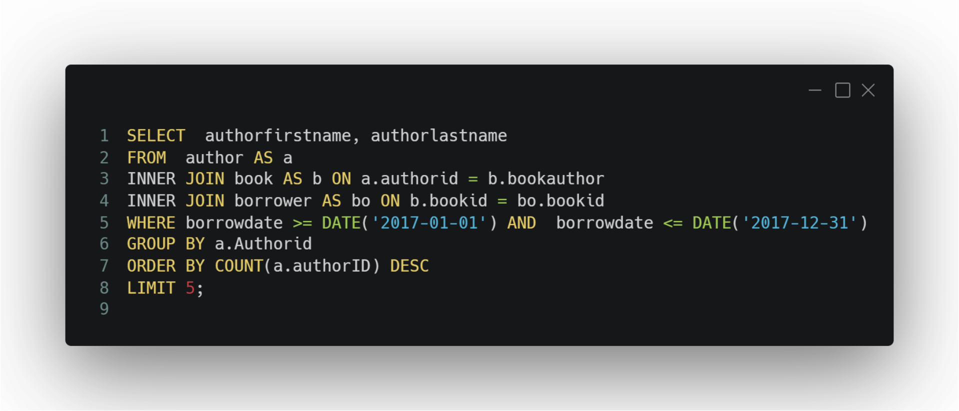
3) -- First and last names of clients that borrowed books in March 2018/



```sql
SELECT clientfirstname, clientlastname
FROM client AS c
INNER JOIN borrower AS b
ON c.clientid = b.clientid
WHERE borrowdate >= '2017-03-01' AND borrowdate <= '2017-03-31'
GROUP BY c.clientid;

```

Query three returns fields of first name and last name of the client that borrowed books in March of 2018 FROM the client TABLE. The client will be aliased with a c in order to keep it simple when joining client TABLE with the borrower TABLE. THE borrower TABLE will be ALIASed with a b. The INNER JOIN returns data to the results set only if all the data will be available. To elaborate no null values. We query with the WHERE clause so based on the condition the barrow date remains with the time period of March of 2018. This result-set returns 25 records.

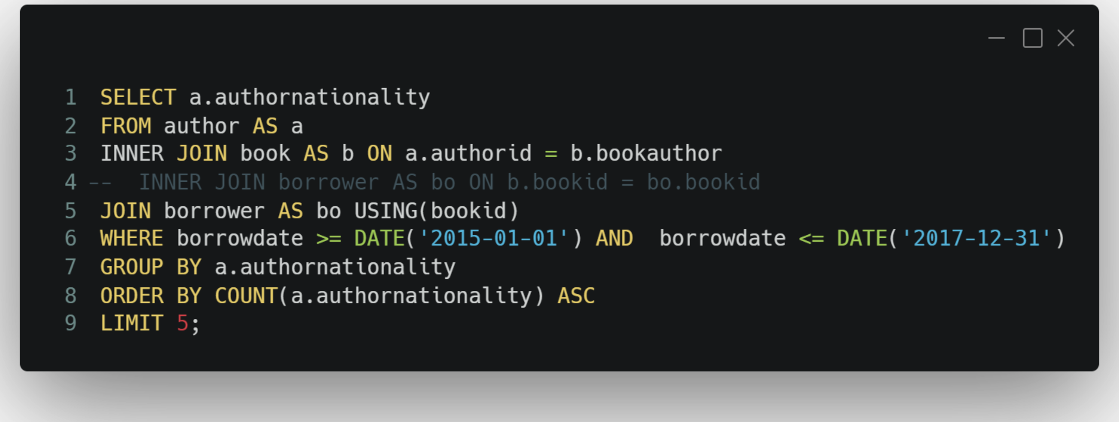
4) -- First and last names of the top 5 authors clients borrowed in 2017



```sql
SELECT  authorfirstname, authorlastname
FROM  author AS a
INNER JOIN book AS b ON a.authorid = b.bookauthor
INNER JOIN borrower AS bo ON b.bookid = bo.bookid
WHERE borrowdate >= DATE('2017-01-01') AND  borrowdate <= DATE('2017-12-31')
GROUP BY a.Authorid
ORDER BY COUNT(a.authorID) DESC
LIMIT 5;
```

The authors' first name and last name fields will be returned, based on the condition of three tables that will be joined based on primary keys from each table. The author table will first be joined with the book table USING(authorID and book author) of an inner join to avoid returning null values. Next the borrower table will be joined to meet the condition of books borrowed in the year 2017. We group by author Id to keep track of how many rows contain the same value. The field will be ordered by author ID in descending order. To restrict the records of the rowset by 5 I used the LIMIT CLAUSE. The result set will be returned in descending order DESC in order to obtain the top 5 results,

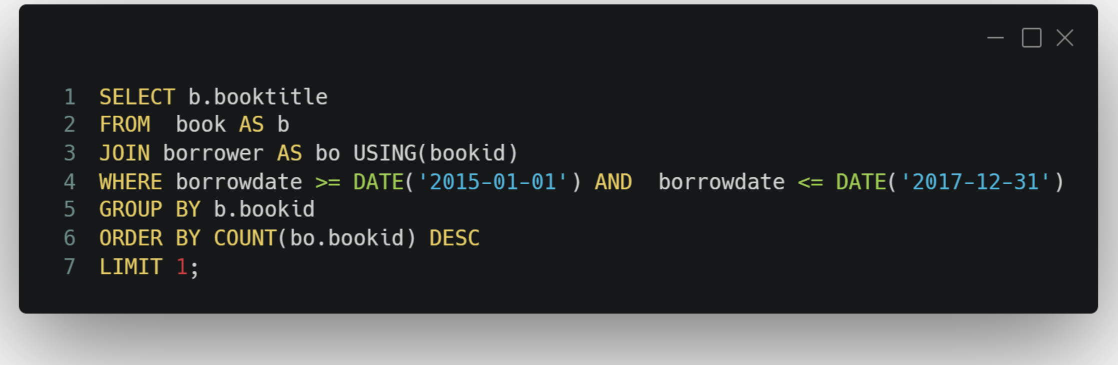
5) -- Nationalities of the least 5 authors that clients borrowed during the years 2015-2017



```sql
SELECT a.authornationality
FROM author AS a
INNER JOIN book AS b ON a.authorid = b.bookauthor
--  INNER JOIN borrower AS bo ON b.bookid = bo.bookid
JOIN borrower AS bo USING(bookid)
WHERE borrowdate >= DATE('2015-01-01') AND  borrowdate <= DATE('2017-12-31')
GROUP BY a.authornationality
ORDER BY COUNT(a.authornationality) ASC
LIMIT 5;
```

Next, getting the nationalities of the least favorite authors clients borrowed from during the period of 2015 and 2017 requires querying from the book, borrower and author tables. We use an outer join to gain access to the book author books by joining with the author id and book author. I wanted to showcase the USING() by joining a table by the book id. We group by to obtain the occurrences of borrowed books. Then the records are ordered by nationality in ascending order restricted to five records. The records contain Spain, China, Great Britain, Brazil and France.

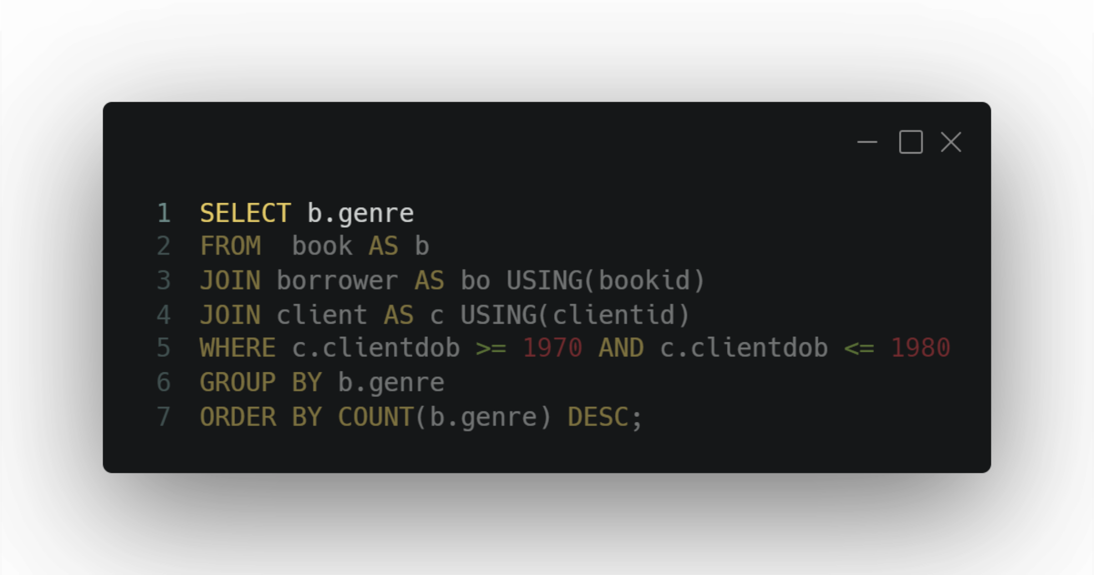
6) -- The book that was most borrowed during the years 2015-2017



```
SELECT b.booktitle
FROM  book AS b
JOIN borrower AS bo USING(bookid)
WHERE borrowdate >= DATE('2015-01-01') AND  borrowdate <= DATE('2017-12-31')
GROUP BY b.bookid
ORDER BY COUNT(bo.bookid) DESC
LIMIT 1;
```

The most popular book based on the rowset was *Electrical Transformer* between the period of 2015 and 2017. The result set will return the book title field by joining the book table by utilizing the primary key of bookid but the foreign key of the borrower table. The condition will be that the barrow date resides between 2015 and 2017 by using the AND clause which joins conditions. I showcased mySQLs DATE() function to extract the date but ultimately just to experiment with query functions. We keep count with the group by clause and get a restricted order bookid. We restrict by 1 but could also utilized the aggregate function of MAX();

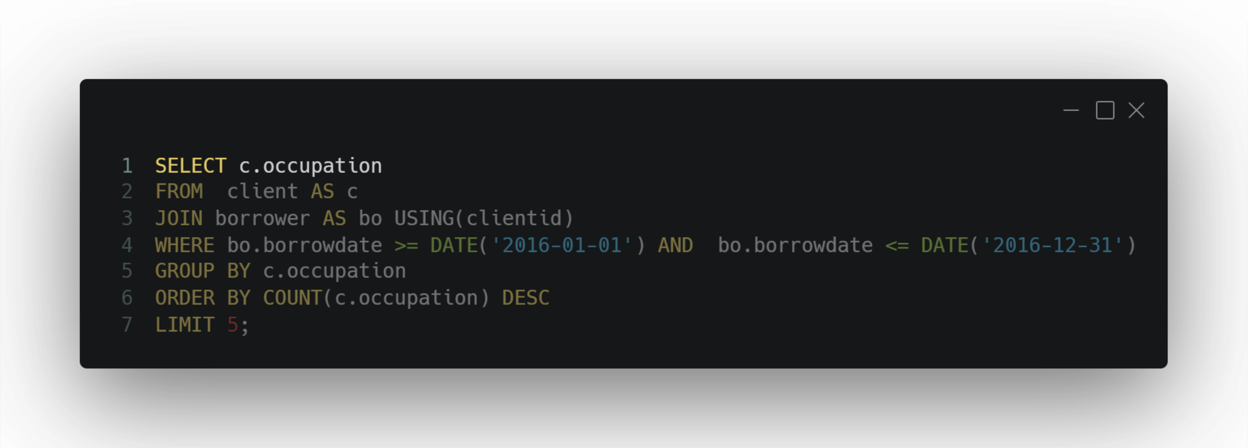
7) - Top borrowed genres for client born in years 1970-1980



```sql
SELECT b.genre
FROM  book AS b
JOIN borrower AS bo USING(bookid)
JOIN client AS c USING(clientid)
WHERE c.clientdob >= 1970 AND c.clientdob <= 1980
GROUP BY b.genre
ORDER BY COUNT(b.genre) DESC;
```

Based on the schema the genre can be retrieved from the book TABLE by joining the book and borrower table with the PK of bookid.since we want to know the result based on clients demographic of ages between 1970 and 1980 we will utilize the client table joining with the clientid. We keep count of the genre by grouping by genre. The result set returns 10 records ordered by the count in descending DESC order.

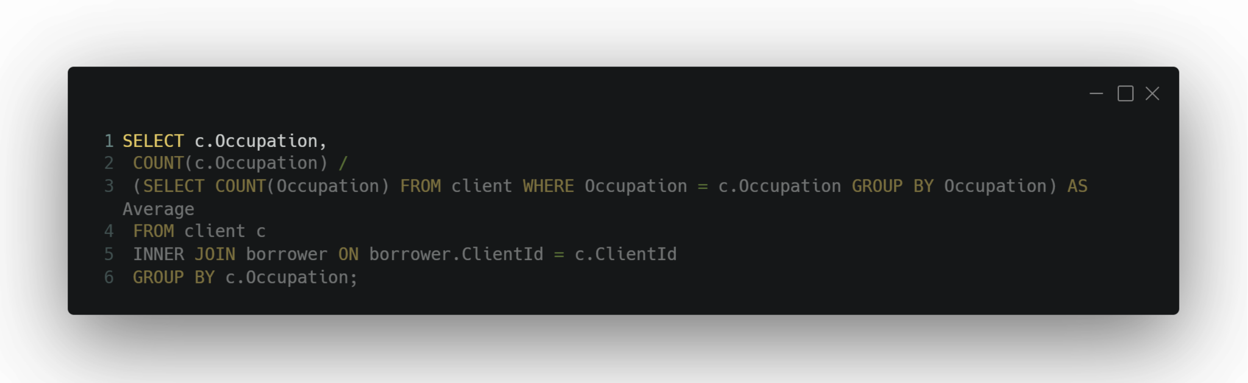
8) -- Top 5 occupations that borrowed the most in 2016



```sql
SELECT c.occupation
FROM  client AS c
JOIN borrower AS bo USING(clientid)
WHERE bo.borrowdate >= DATE('2016-01-01') AND  bo.borrowdate <= DATE('2016-12-31')
GROUP BY c.occupation
ORDER BY COUNT(c.occupation) DESC
LIMIT 5;
```

Next we query the public library database to determine which occupation read the most during the period of 2016. This query will be restricted to five by using the limit clause of and then setting it to 5. Occupation field will be located in the clients table which can be joined to the borrower table using the PK of client id.we join condition to the beginning of 2016 and the last date of 2016 using the AND clause. The row set contained Students, Police Officers, School teachers, Bus Drivers and computer programmers. To keep count we utilized grouping by the occupation, which then also ordered the occupation in descending order to obtain highest readers.

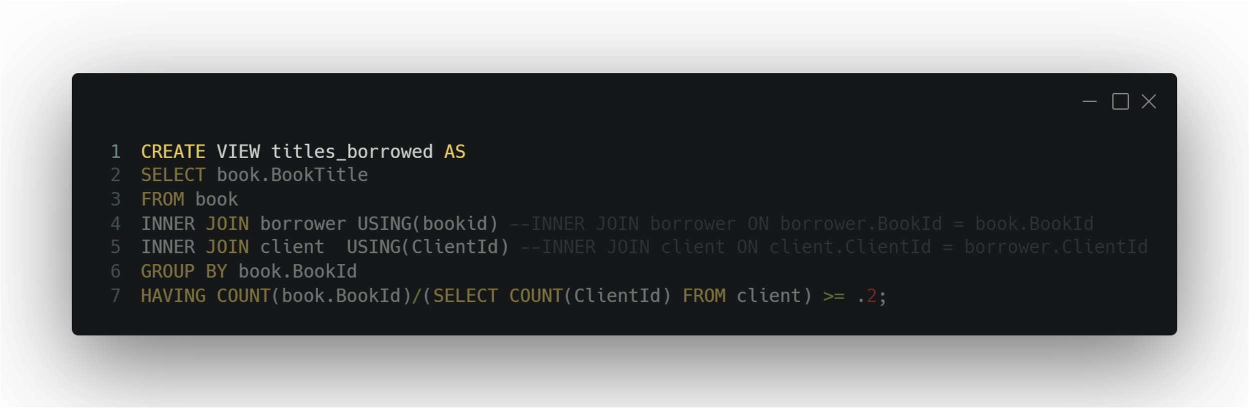
9) -- Average number of borrowed books by job title



```
SELECT c.Occupation,
 COUNT(c.Occupation) /
 (SELECT COUNT(Occupation) FROM client WHERE Occupation = c.Occupation GROUP BY Occupation) AS
Average
 FROM client c
 INNER JOIN borrower ON borrower.ClientId = c.ClientId
 GROUP BY c.Occupation;
```

I wanted to showcase the beauty of SQL by just doing simple math to obtain the average by dividing the part over the aggregate total. But another way would have been to utilize the AVG() aggregate function and keep count of the book id . client and borrower are joined by the clientid. The rowset will return 32 records in no specified order.

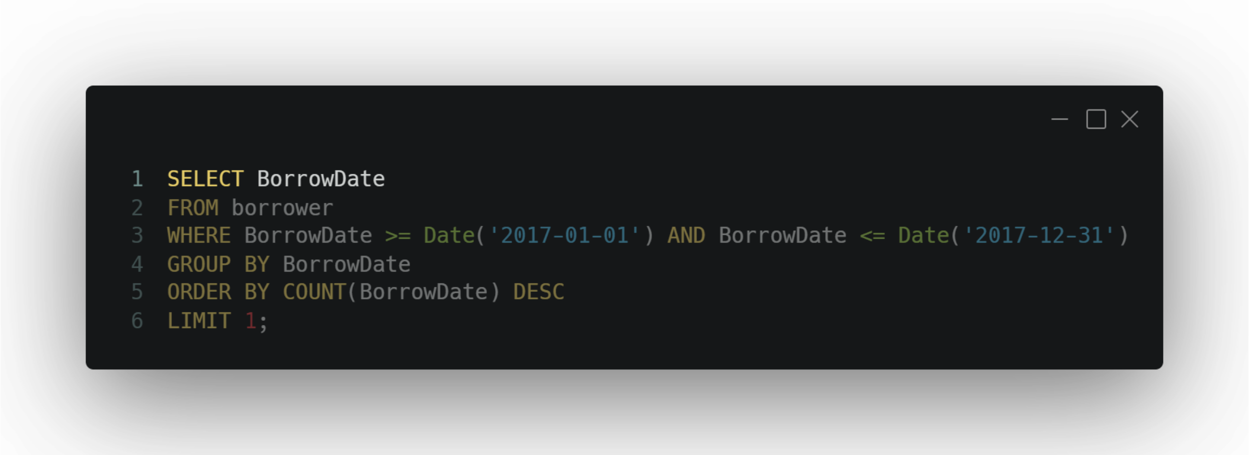
10) -- Create a VIEW and display the titles that were borrowed by at least 20% of clients



```
CREATE VIEW titles_borrowed AS
SELECT book.BookTitle
FROM book
INNER JOIN borrower USING(bookid) --INNER JOIN borrower ON borrower.BookId = book.BookId
INNER JOIN client  USING(ClientId) --INNER JOIN client ON client.ClientId = borrower.ClientId
GROUP BY book.BookId
HAVING COUNT(book.BookId)/(SELECT COUNT(ClientId) FROM client) >= .2;
```

Next we create a view so the user could see a subset of titles borrowed by 20 percent of clients. The view will be returned as a book title using a nested query. Book and borrower are joined using the bookid PK. we will keep cout by grouping by the book id. Then the HAVING clause will group the records that are at least 20 percent. Now we can select * wildcard of the title_barrowed table.

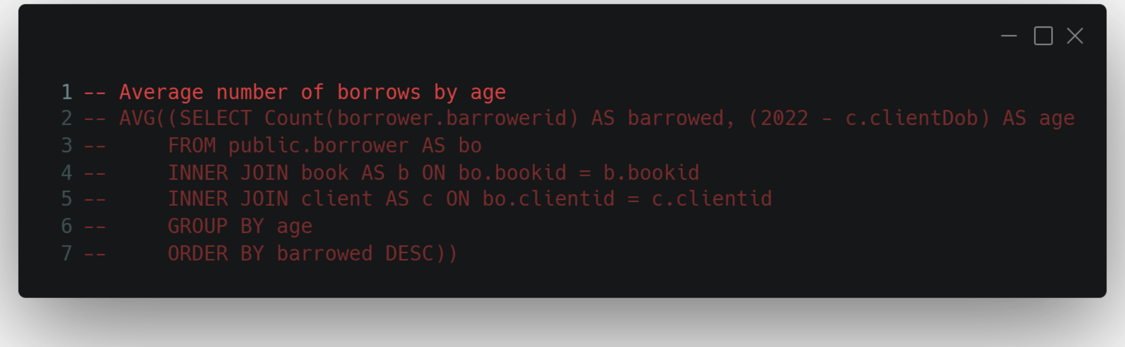
11) -- The top month of borrows in 2017



```
SELECT BorrowDate
FROM borrower
WHERE BorrowDate >= Date('2017-01-01') AND BorrowDate <= Date('2017-12-31')
GROUP BY BorrowDate
ORDER BY COUNT(BorrowDate) DESC
LIMIT 1;
```

In order to obtain the top months of borrower during the period of 2017. I queried the borrower table for the field of the borrowDate. We do specify the condition using the AND clause to obtain the rowset between the first and the last of the month.thethe count was monitored by the barrowdate ordered in descending order.

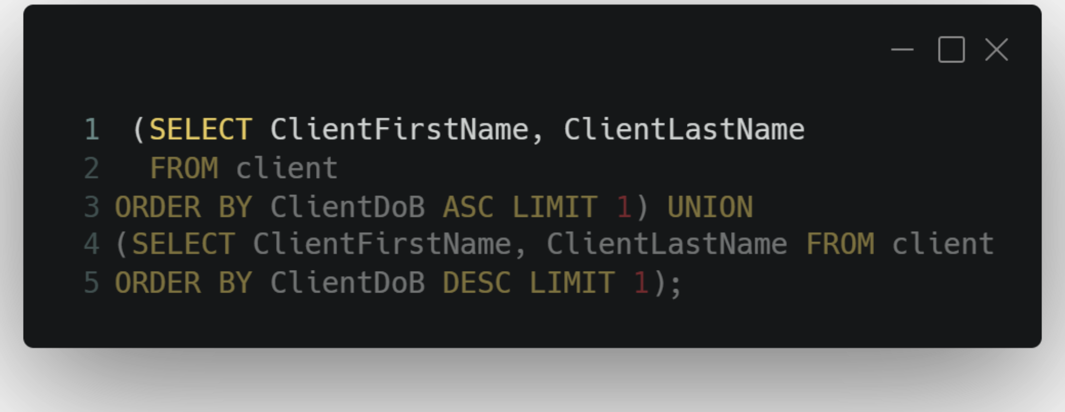
12) -- Average number of borrows by age



```
-- Average number of borrows by age
-- AVG((SELECT Count(borrower.barrowerid) AS barrowed, (2022 - c.clientDob) AS age
--     FROM public.borrower AS bo
--     INNER JOIN book AS b ON bo.bookid = b.bookid
--     INNER JOIN client AS c ON bo.clientid = c.clientid
--     GROUP BY age
--     ORDER BY barrowed DESC))
```

I attempted to go to stack overflow for this one without a positive outcome. I failed this one. I attempted to query using a subquery to keep track of the count which I believed would resulted in a count which I could later get an average for

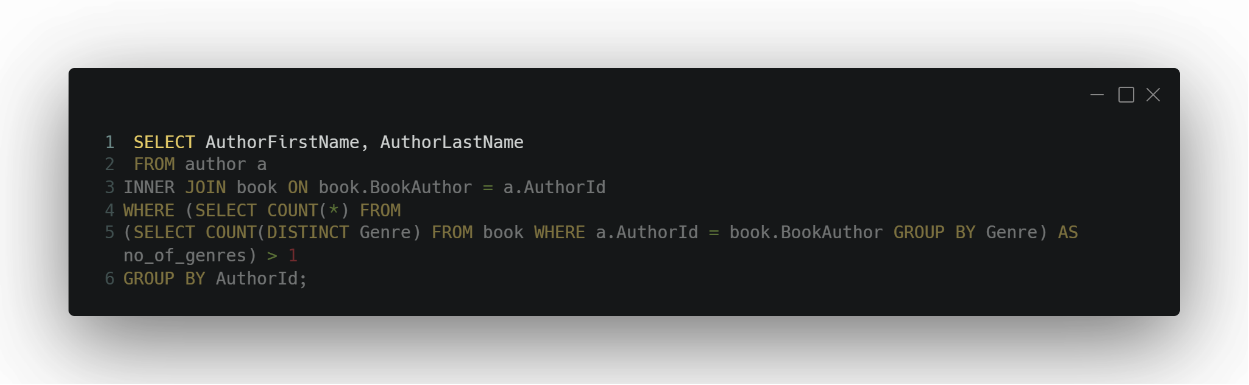
13) -- The oldest and the youngest clients of the library



```
 (SELECT ClientFirstName, ClientLastName
  FROM client
ORDER BY ClientDoB ASC LIMIT 1) UNION
(SELECT ClientFirstName, ClientLastName FROM client
ORDER BY ClientDoB DESC LIMIT 1);
```

A union of the two statements resulted in the contributions to the results of the youngest and oldest patrons of the library. I queried the oldest individual by listing the date of birth in descending order and the youngest in reverse or ascending order. Both were restricted by one.

14) -- First and last names of authors that wrote books in more than one genre



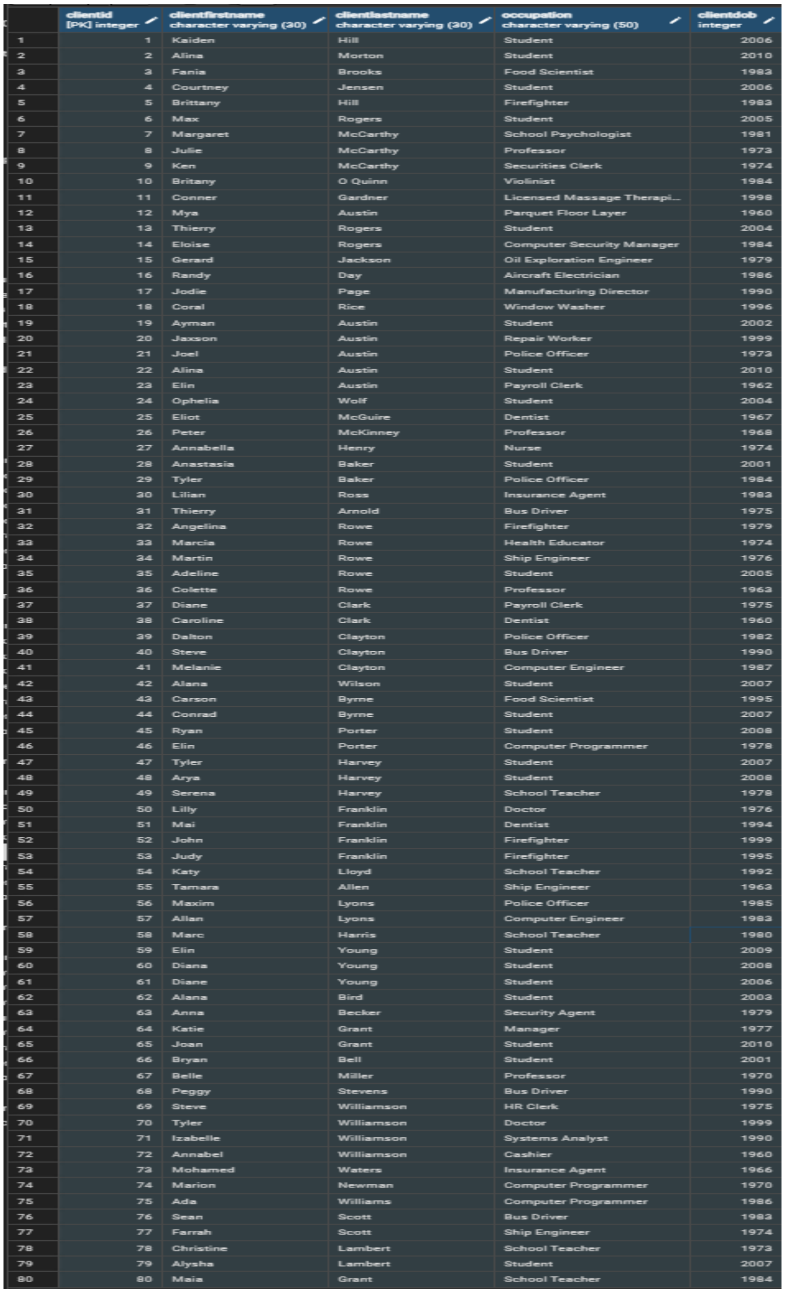
```sql
 SELECT AuthorFirstName, AuthorLastName
 FROM author a
INNER JOIN book ON book.BookAuthor = a.AuthorId
WHERE (SELECT COUNT(*) FROM
(SELECT COUNT(DISTINCT Genre) FROM book WHERE a.AuthorId = book.BookAuthor GROUP BY Genre) AS
no_of_genres) > 1
GROUP BY AuthorId;
```

Finally, the first and last name of author that wrote in more than one book genre was obtained by joining the authorid and th book author

Results

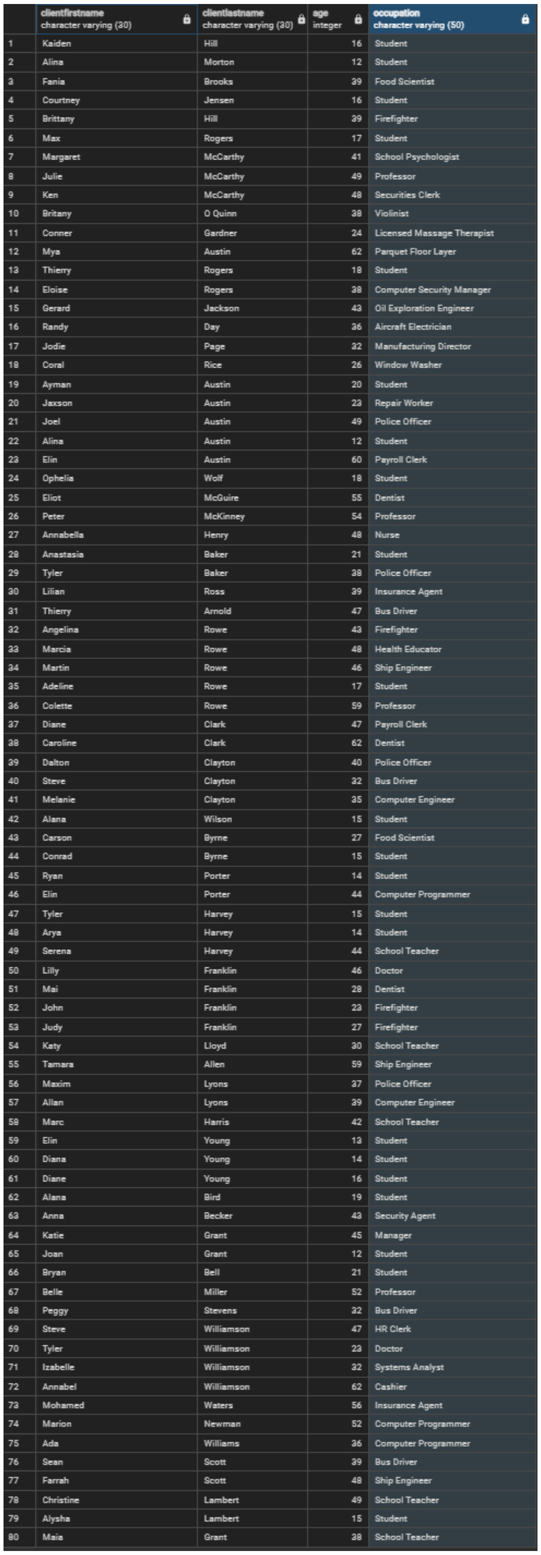
The results are shown using carbon and were tested using PGADMIN for postgres. The results are labeled based on the result set of each query.

Query Result-Set 1



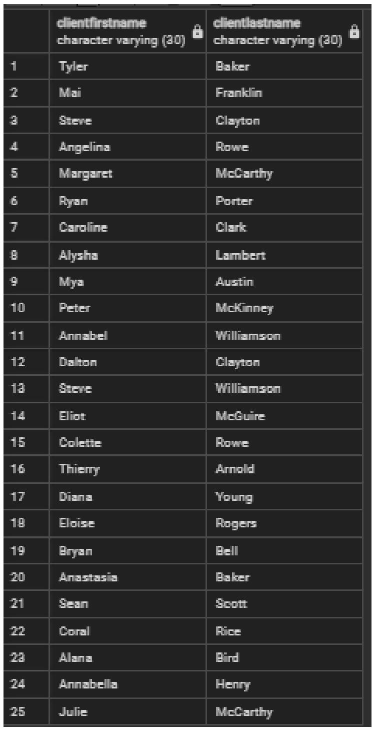
	clientid [PK] integer	clientfirstname character varying (30)	clientlastname character varying (30)	occupation character varying (50)	clientdob integer
1	1	Kaiden	Hill	Student	2006
2	2	Alina	Morton	Student	2010
3	3	Fania	Brooks	Food Scientist	1983
4	4	Courtney	Jensen	Student	2006
5	5	Brittany	Hill	Firefighter	1983
6	6	Max	Rogers	Student	2005
7	7	Margaret	McCarthy	School Psychologist	1981
8	8	Julie	McCarthy	Professor	1973
9	9	Ken	McCarthy	Securities Clerk	1974
10	10	Britany	O Quinn	Violinist	1984
11	11	Conner	Gardner	Licensed Massage Therapi...	1998
12	12	Mya	Austin	Parquet Floor Layer	1960
13	13	Thierry	Rogers	Student	2004
14	14	Eloise	Rogers	Computer Security Manager	1984
15	15	Gerard	Jackson	Oil Exploration Engineer	1979
16	16	Randy	Day	Aircraft Electrician	1986
17	17	Jodie	Page	Manufacturing Director	1990
18	18	Coral	Rice	Window Washer	1996
19	19	Ayman	Austin	Student	2002
20	20	Jaxson	Austin	Repair Worker	1999
21	21	Joel	Austin	Police Officer	1973
22	22	Alina	Austin	Student	2010
23	23	Elin	Austin	Payroll Clerk	1962
24	24	Ophelia	Wolf	Student	2004
25	25	Eliot	McGuire	Dentist	1967
26	26	Peter	McKinney	Professor	1968
27	27	Annabella	Henry	Nurse	1974
28	28	Anastasia	Baker	Student	2001
29	29	Tyler	Baker	Police Officer	1984
30	30	Lilian	Ross	Insurance Agent	1983
31	31	Thierry	Arnold	Bus Driver	1975
32	32	Angelina	Rowe	Firefighter	1979
33	33	Marcia	Rowe	Health Educator	1974
34	34	Martin	Rowe	Ship Engineer	1976
35	35	Adeline	Rowe	Student	2005
36	36	Colette	Rowe	Professor	1963
37	37	Diane	Clark	Payroll Clerk	1975
38	38	Caroline	Clark	Dentist	1960
39	39	Dalton	Clayton	Police Officer	1982
40	40	Steve	Clayton	Bus Driver	1990
41	41	Melanie	Clayton	Computer Engineer	1987
42	42	Alana	Wilson	Student	2007
43	43	Carson	Byrne	Food Scientist	1995
44	44	Conrad	Byrne	Student	2007
45	45	Ryan	Porter	Student	2008
46	46	Elin	Porter	Computer Programmer	1978
47	47	Tyler	Harvey	Student	2007
48	48	Arya	Harvey	Student	2008
49	49	Serena	Harvey	School Teacher	1978
50	50	Lilly	Franklin	Doctor	1976
51	51	Mai	Franklin	Dentist	1994
52	52	John	Franklin	Firefighter	1999
53	53	Judy	Franklin	Firefighter	1995
54	54	Katy	Lloyd	School Teacher	1992
55	55	Tamara	Allen	Ship Engineer	1963
56	56	Maxim	Lyons	Police Officer	1985
57	57	Allan	Lyons	Computer Engineer	1983
58	58	Marc	Harris	School Teacher	1980
59	59	Elin	Young	Student	2009
60	60	Diana	Young	Student	2008
61	61	Diane	Young	Student	2006
62	62	Alana	Bird	Student	2003
63	63	Anna	Becker	Security Agent	1979
64	64	Katie	Grant	Manager	1977
65	65	Joan	Grant	Student	2010
66	66	Bryan	Bell	Student	2001
67	67	Belle	Miller	Professor	1970
68	68	Peggy	Stevens	Bus Driver	1990
69	69	Steve	Williamson	HR Clerk	1975
70	70	Tyler	Williamson	Doctor	1999
71	71	Izabelle	Williamson	Systems Analyst	1990
72	72	Annabel	Williamson	Cashier	1960
73	73	Mohamed	Waters	Insurance Agent	1966
74	74	Marion	Newman	Computer Programmer	1970
75	75	Ada	Williams	Computer Programmer	1986
76	76	Sean	Scott	Bus Driver	1983
77	77	Farrah	Scott	Ship Engineer	1974
78	78	Christine	Lambert	School Teacher	1973
79	79	Alysha	Lambert	Student	2007
80	80	Maia	Grant	School Teacher	1984

Query Result-Set 2



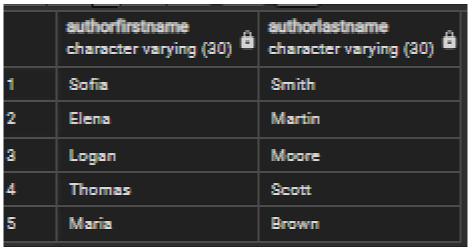
	clientfirstname character varying (30)	clientlastname character varying (30)	age integer	occupation character varying (50)
1	Kaiden	Hill	16	Student
2	Alina	Morton	12	Student
3	Fania	Brooks	39	Food Scientist
4	Courtney	Jensen	16	Student
5	Brittany	Hill	39	Firefighter
6	Max	Rogers	17	Student
7	Margaret	McCarthy	41	School Psychologist
8	Julie	McCarthy	49	Professor
9	Ken	McCarthy	48	Securities Clerk
10	Britany	O Quinn	38	Violinist
11	Conner	Gardner	24	Licensed Massage Therapist
12	Mya	Austin	62	Parquet Floor Layer
13	Thierry	Rogers	18	Student
14	Eloise	Rogers	38	Computer Security Manager
15	Gerard	Jackson	43	Oil Exploration Engineer
16	Randy	Day	36	Aircraft Electrician
17	Jodie	Page	32	Manufacturing Director
18	Coral	Rice	26	Window Washer
19	Ayman	Austin	20	Student
20	Jaxson	Austin	23	Repair Worker
21	Joel	Austin	49	Police Officer
22	Alina	Austin	12	Student
23	Elin	Austin	60	Payroll Clerk
24	Ophelia	Wolf	18	Student
25	Eliot	McGuire	55	Dentist
26	Peter	McKinney	54	Professor
27	Annabella	Henry	48	Nurse
28	Anastasia	Baker	21	Student
29	Tyler	Baker	38	Police Officer
30	Lilian	Ross	39	Insurance Agent
31	Thierry	Arnold	47	Bus Driver
32	Angelina	Rowe	43	Firefighter
33	Marcia	Rowe	48	Health Educator
34	Martin	Rowe	46	Ship Engineer
35	Adeline	Rowe	17	Student
36	Colette	Rowe	59	Professor
37	Diane	Clark	47	Payroll Clerk
38	Caroline	Clark	62	Dentist
39	Dalton	Clayton	40	Police Officer
40	Steve	Clayton	32	Bus Driver
41	Melanie	Clayton	35	Computer Engineer
42	Alana	Wilson	15	Student
43	Carson	Byrne	27	Food Scientist
44	Conrad	Byrne	15	Student
45	Ryan	Porter	14	Student
46	Elin	Porter	44	Computer Programmer
47	Tyler	Harvey	15	Student
48	Arya	Harvey	14	Student
49	Serena	Harvey	44	School Teacher
50	Lilly	Franklin	46	Doctor
51	Mai	Franklin	28	Dentist
52	John	Franklin	23	Firefighter
53	Judy	Franklin	27	Firefighter
54	Katy	Lloyd	30	School Teacher
55	Tamara	Allen	59	Ship Engineer
56	Maxim	Lyons	37	Police Officer
57	Allan	Lyons	39	Computer Engineer
58	Marc	Harris	42	School Teacher
59	Elin	Young	13	Student
60	Diana	Young	14	Student
61	Diane	Young	16	Student
62	Alana	Bird	19	Student
63	Anna	Becker	43	Security Agent
64	Katie	Grant	45	Manager
65	Joan	Grant	12	Student
66	Bryan	Bell	21	Student
67	Belle	Miller	52	Professor
68	Peggy	Stevens	32	Bus Driver
69	Steve	Williamson	47	HR Clerk
70	Tyler	Williamson	23	Doctor
71	Izabelle	Williamson	32	Systems Analyst
72	Annabel	Williamson	62	Cashier
73	Mohamed	Waters	56	Insurance Agent
74	Marion	Newman	52	Computer Programmer
75	Ada	Williams	36	Computer Programmer
76	Sean	Scott	39	Bus Driver
77	Farrah	Scott	48	Ship Engineer
78	Christine	Lambert	49	School Teacher
79	Alysha	Lambert	15	Student
80	Maia	Grant	38	School Teacher

Query Result-Set 3



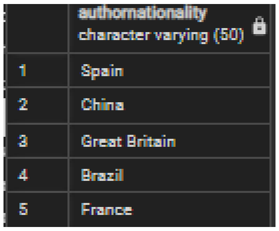
	clientfirstname character varying (30)	clientlastname character varying (30)
1	Tyler	Baker
2	Mai	Franklin
3	Steve	Clayton
4	Angelina	Rowe
5	Margaret	McCarthy
6	Ryan	Porter
7	Caroline	Clark
8	Alysha	Lambert
9	Mya	Austin
10	Peter	McKinney
11	Annabel	Williamson
12	Dalton	Clayton
13	Steve	Williamson
14	Eliot	McGuire
15	Colette	Rowe
16	Thierry	Arnold
17	Diana	Young
18	Eloise	Rogers
19	Bryan	Bell
20	Anastasia	Baker
21	Sean	Scott
22	Coral	Rice
23	Alana	Bird
24	Annabella	Henry
25	Julie	McCarthy

Query Result-Set 4



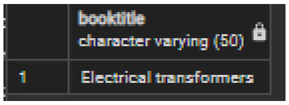
	authorfirstname character varying (30)	authorlastname character varying (30)
1	Sofia	Smith
2	Elena	Martin
3	Logan	Moore
4	Thomas	Scott
5	Maria	Brown

Query Result-Set 5



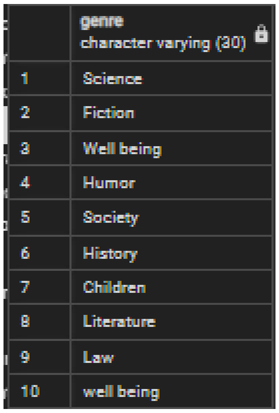
	authornationality character varying (50)
1	Spain
2	China
3	Great Britain
4	Brazil
5	France

Query Result-Set 6



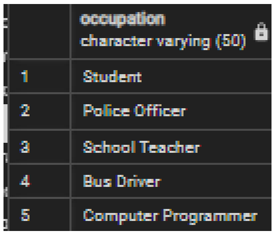
	booktitle character varying (50)
1	Electrical transformers

Query Result-Set 7



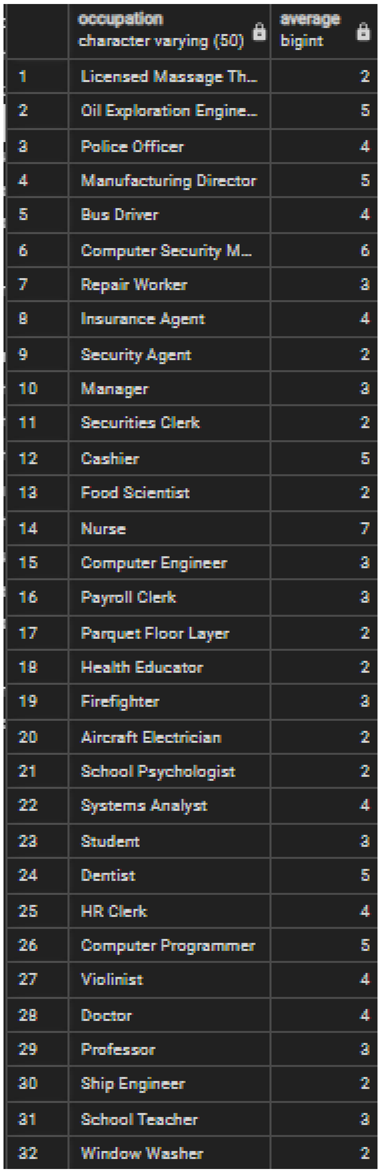
	genre character varying (30)
1	Science
2	Fiction
3	Well being
4	Humor
5	Society
6	History
7	Children
8	Literature
9	Law
10	well being

Query Result-Set 8



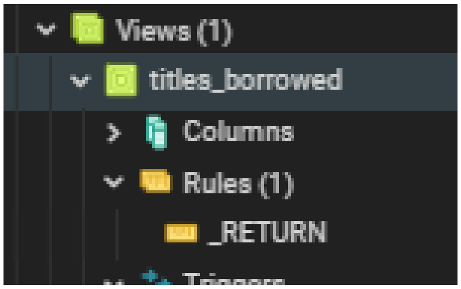
	occupation character varying (50)
1	Student
2	Police Officer
3	School Teacher
4	Bus Driver
5	Computer Programmer

Query Result-Set 9

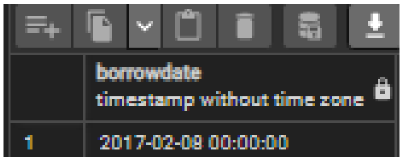


	occupation character varying (50)	average bigint
1	Licensed Massage Th...	2
2	Oil Exploration Engine...	5
3	Police Officer	4
4	Manufacturing Director	5
5	Bus Driver	4
6	Computer Security M...	6
7	Repair Worker	3
8	Insurance Agent	4
9	Security Agent	2
10	Manager	3
11	Securities Clerk	2
12	Cashier	5
13	Food Scientist	2
14	Nurse	7
15	Computer Engineer	3
16	Payroll Clerk	3
17	Parquet Floor Layer	2
18	Health Educator	2
19	Firefighter	3
20	Aircraft Electrician	2
21	School Psychologist	2
22	Systems Analyst	4
23	Student	3
24	Dentist	5
25	HR Clerk	4
26	Computer Programmer	5
27	Violinist	4
28	Doctor	4
29	Professor	3
30	Ship Engineer	2
31	School Teacher	3
32	Window Washer	2

Query Result-Set 10



Query Result-Set 11

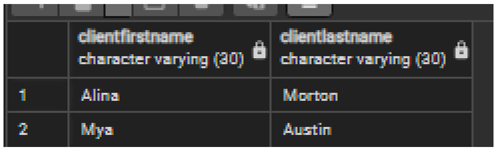


	borrowdate timestamp without time zone
1	2017-02-08 00:00:00

Query Result-Set 13

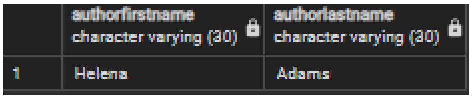
No result did not understand question

Query Result-Set 14



	clientfirstname character varying (30)	clientlastname character varying (30)
1	Alina	Morton
2	Mya	Austin

Query Result-Set 15



	authorfirstname character varying (30)	authorlastname character varying (30)
1	Helena	Adams

Discussion

Using queries has given me much more insight in creating the manipulations of the database which contains tables. Postgres was the database management system that I utilized and it was the one I was most familiar with. SQL is a beautiful language which can query databases more efficiently. The Flaw is that postgres is only in English and it has low reading speed. I love postgres because I recognize it for scalability and it being compliant with sql.

References

Database Table: Design & Conventions. (2016, July 19). Retrieved from https://study.com/academy/lesson/database-table-design-conventions.html.

Advanced SQL Query Syntax. (2018, July 31). Retrieved from https://study.com/academy/lesson/advanced-sql-query-syntax.html.

SQL Views: Definition & Example. (2020, March 14). Retrieved from https://study.com/academy/lesson/sql-views-definition-example.html.

Advanced SQL Subqueries: Use & Examples. (2021, November 16). Retrieved from https://study.com/academy/lesson/advanced-sql-subqueries-use-examples.html.

SQL: CREATE Index. (2016, August 7). Retrieved from https://study.com/academy/lesson/sql-create-index.html.

Duckett, J. (2018). *PHP & MySQL: Server-side web development*. Wiley.