Name: Darren Lewis
Date:5/15/2023
Computer Science 204

Create Author Table and Set Primary Key to AuthorID

```
CREATE TABLE Author (
  AuthorID INT PRIMARY KEY,
  AuthorFirstName VARCHAR(50),
  AuthorLastName VARCHAR(50),
  AuthorNationality VARCHAR(50)
);


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');
```

Create Book Table and Set Primary Key to BookID

```
CREATE TABLE Book (
  BookID INT,
  BookTitle VARCHAR(100),
  BookAuthor INT,
```

```
  Genre VARCHAR(50),
  PRIMARY KEY (BookID),
  FOREIGN KEY (BookAuthor) REFERENCES Author(AuthorID)
);

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');
```

Create Client Table and Set Primary Key to ClientID

```
CREATE TABLE Client (
  ClientID INT,
  ClientFirstName VARCHAR(100),
  ClientLastName VARCHAR(100),
```

```
  ClientDoB INT,
  Occupation VARCHAR(100),
  PRIMARY KEY (ClientID)
);

INSERT INTO Client (ClientID, ClientFirstName, ClientLastName, ClientDoB, Occupation)
VALUES
  (1, 'Kaiden', 'Hill', 2006, 'Student'),
  (2, 'Alina', 'Morton', 2010, 'Student'),
  (3, 'Fania', 'Brooks', 1983, 'Food Scientist'),
  (4, 'Courtney', 'Jensen', 2006, 'Student'),
  (5, 'Brittany', 'Hill', 1983, 'Firefighter'),
  (6, 'Max', 'Rogers', 2005, 'Student'),
  (7, 'Margaret', 'McCarthy', 1981, 'School Psychologist'),
  (8, 'Julie', 'McCarthy', 1973, 'Professor'),
  (9, 'Ken', 'McCarthy', 1974, 'Securities Clerk'),
  (10, 'Britany', 'O''Quinn', 1984, 'Violinist'),
  (11, 'Conner', 'Gardner', 1998, 'Licensed Massage Therapist'),
  (12, 'Mya', 'Austin', 1960, 'Parquet Floor Layer'),
  (13, 'Thierry', 'Rogers', 2004, 'Student'),
  (14, 'Eloise', 'Rogers', 1984, 'Computer Security Manager'),
  (15, 'Gerard', 'Jackson', 1979, 'Oil Exploration Engineer'),
  (16, 'Randy', 'Day', 1986, 'Aircraft Electrician'),
  (17, 'Jodie', 'Page', 1990, 'Manufacturing Director'),
  (18, 'Coral', 'Rice', 1996, 'Window Washer'),
  (19, 'Ayman', 'Austin', 2002, 'Student'),
  (20, 'Jaxson', 'Austin', 1999, 'Repair Worker'),
  (21, 'Joel', 'Austin', 1973, 'Police Officer'),
  (22, 'Alina', 'Austin', 2010, 'Student'),
  (23, 'Elin', 'Austin', 1962, 'Payroll Clerk'),
  (24, 'Ophelia', 'Wolf', 2004, 'Student'),
  (25, 'Eliot', 'McGuire', 1967, 'Dentist'),
  (26, 'Peter', 'McKinney', 1968, 'Professor'),
  (27, 'Annabella', 'Henry', 1974, 'Nurse'),
  (28, 'Anastasia', 'Baker', 2001, 'Student'),
  (29, 'Tyler', 'Baker', 1984, 'Police Officer'),
  (30, 'Lilian', 'Ross', 1983, 'Insurance Agent'),
  (31, 'Thierry', 'Arnold', 1975, 'Bus Driver'),
  (32, 'Angelina', 'Rowe', 1979, 'Firefighter'),
  (33, 'Marcia', 'Rowe', 1974, 'Health Educator'),
  (34, 'Martin', 'Rowe', 1976, 'Ship Engineer'),
  (35, 'Adeline', 'Rowe', 2005, 'Student'),
  (36, 'Colette', 'Rowe', 1963, 'Professor'),
  (37, 'Diane', 'Clark', 1975, 'Payroll Clerk'),
  (38, 'Caroline', 'Clark', 1960, 'Dentist'),
  (39, 'Dalton', 'Clayton', 1982, 'Police Officer'),
  (40, 'Steve', 'Clayton', 1990, 'Bus Driver'),
  (41, 'Melanie', 'Clayton', 1987, 'Computer Engineer'),
  (42, 'Alana', 'Wilson', 2007, 'Student'),
```

```
(43, 'Carson', 'Byrne', 1995, 'Food Scientist'),
(44, 'Conrad', 'Byrne', 2007, 'Student'),
(45, 'Ryan', 'Porter', 2008, 'Student'),
(46, 'Elin', 'Porter', 1978, 'Computer Programmer'),
(47, 'Tyler', 'Harvey', 2007, 'Student'),
(48, 'Arya', 'Harvey', 2008, 'Student'),
(49, 'Serena', 'Harvey', 1978, 'School Teacher'),
(50, 'Lilly', 'Franklin', 1976, 'Doctor'),
(51, 'Mai', 'Franklin', 1994, 'Dentist'),
(52, 'John', 'Franklin', 1999, 'Firefighter'),
(53, 'Judy', 'Franklin', 1995, 'Firefighter'),
(54, 'Katy', 'Lloyd', 1992, 'School Teacher'),
(55, 'Tamara', 'Allen', 1963, 'Ship Engineer'),
(56, 'Maxim', 'Lyons', 1985, 'Police Officer'),
(57, 'Allan', 'Lyons', 1983, 'Computer Engineer'),
(58, 'Marc', 'Harris', 1980, 'School Teacher'),
(59, 'Elin', 'Young', 2009, 'Student'),
(60, 'Diana', 'Young', 2008, 'Student'),
(61, 'Diane', 'Young', 2006, 'Student'),
(62, 'Alana', 'Bird', 2003, 'Student'),
(63, 'Anna', 'Becker', 1979, 'Security Agent'),
(64, 'Katie', 'Grant', 1977, 'Manager'),
(65, 'Joan', 'Grant', 2010, 'Student'),
(66, 'Bryan', 'Bell', 2001, 'Student'),
(67, 'Belle', 'Miller', 1970, 'Professor'),
(68, 'Peggy', 'Stevens', 1990, 'Bus Driver'),
(69, 'Steve', 'Williamson', 1975, 'HR Clerk'),
(70, 'Tyler', 'Williamson', 1999, 'Doctor'),
(71, 'Izabelle', 'Williamson', 1990, 'Systems Analyst'),
(72, 'Annabel', 'Williamson', 1960, 'Cashier'),
(73, 'Mohamed', 'Waters', 1966, 'Insurance Agent'),
(74, 'Marion', 'Newman', 1970, 'Computer Programmer'),
(75, 'Ada', 'Williams', 1986, 'Computer Programmer'),
(76, 'Sean', 'Scott', 1983, 'Bus Driver'),
(77, 'Farrah', 'Scott', 1974, 'Ship Engineer'),
(78, 'Christine', 'Lambert', 1973, 'School Teacher'),
(79, 'Alysha', 'Lambert', 2007, 'Student'),
(80, 'Maia', 'Grant', 1984, 'School Teacher');
```

Create Borrower Table and Set Primary Key to ClientID

```
CREATE TABLE Borrower (
  BorrowID INT PRIMARY KEY,
  ClientID INT,
  BookID INT,
```

```
  BorrowDate DATE,
  FOREIGN KEY (ClientID) REFERENCES Client(ClientID),
  FOREIGN KEY (BookID) REFERENCES Book(BookID)
);

INSERT INTO Borrower (BorrowID, ClientID, BookID, BorrowDate) VALUES
(1, 35, 17, '2016-07-20'),
(2, 1, 3, '2017-04-19'),
(3, 42, 8, '2016-10-03'),
(4, 62, 16, '2016-04-05'),
(5, 53, 13, '2017-01-17'),
(6, 33, 15, '2015-11-26'),
(7, 40, 14, '2015-01-21'),
(8, 64, 2, '2017-09-10'),
(9, 56, 30, '2017-08-02'),
(10, 23, 2, '2018-06-28'),
(11, 46, 19, '2015-11-18'),
(12, 61, 20, '2015-11-24'),
(13, 58, 7, '2017-06-17'),
(14, 46, 16, '2017-02-12'),
(15, 80, 21, '2018-03-18'),
(16, 51, 23, '2015-09-01'),
(17, 49, 18, '2015-07-28'),
(18, 43, 18, '2015-11-04'),
(19, 30, 2, '2018-08-10'),
(20, 48, 24, '2015-05-13'),
(21, 71, 5, '2016-09-05'),
(22, 35, 3, '2016-07-03'),
(23, 57, 1, '2015-03-17'),
(24, 23, 25, '2017-08-16'),
(25, 20, 12, '2018-07-24'),
(26, 25, 7, '2015-01-31'),
(27, 72, 29, '2016-04-10'),
(28, 74, 20, '2017-07-31'),
(29, 53, 14, '2016-02-20'),
(30, 32, 10, '2017-07-24'),
(31, 12, 15, '2018-04-25'),
(32, 77, 13, '2017-06-09'),
(33, 30, 4, '2017-10-24'),
(34, 37, 24, '2016-01-14'),
(35, 27, 26, '2017-06-05'),
(36, 1, 16, '2018-05-06'),
(37, 21, 9, '2016-03-19'),
(38, 69, 28, '2017-03-29'),
(39, 17, 19, '2017-03-14'),
(40, 8, 9, '2016-04-22'),
(41, 63, 18, '2015-01-25'),
(42, 65, 20, '2016-10-10'),
(43, 51, 19, '2015-07-28'),
```

(44, 23, 12, '2017-01-25'),
(45, 17, 4, '2017-04-18'),
(46, 68, 5, '2016-09-06'),
(47, 46, 13, '2017-09-30'),
(48, 15, 13, '2017-07-05'),
(49, 11, 19, '2017-12-14'),
(50, 78, 15, '2017-01-26'),
(51, 47, 9, '2015-03-03'),
(52, 68, 7, '2016-05-26'),
(53, 37, 26, '2017-02-06'),
(54, 48, 27, '2015-12-30'),
(55, 9, 21, '2017-10-21'),
(56, 29, 8, '2018-04-01'),
(57, 64, 18, '2017-08-29'),
(58, 61, 26, '2018-02-21'),
(59, 39, 28, '2016-07-26'),
(60, 73, 18, '2018-08-22'),
(61, 11, 13, '2018-01-17'),
(62, 45, 6, '2016-07-20'),
(63, 33, 13, '2018-03-18'),
(64, 10, 17, '2016-06-06'),
(65, 28, 18, '2017-02-17'),
(66, 51, 3, '2016-12-09'),
(67, 29, 2, '2015-09-18'),
(68, 28, 30, '2017-09-14'),
(69, 74, 20, '2015-12-12'),
(70, 15, 22, '2015-01-14'),
(71, 57, 8, '2017-08-20'),
(72, 2, 5, '2015-01-18'),
(73, 74, 12, '2018-04-14'),
(74, 51, 10, '2016-02-25'),
(75, 25, 17, '2015-02-24'),
(76, 45, 21, '2017-02-10'),
(77, 27, 25, '2016-08-03'),
(78, 32, 28, '2016-06-15'),
(79, 71, 21, '2017-05-21'),
(80, 75, 26, '2016-05-03'),
(81, 56, 32, '2015-12-23'),
(82, 26, 32, '2015-05-16'),
(83, 66, 32, '2015-05-30'),
(84, 57, 18, '2017-09-15'),
(85, 40, 15, '2016-09-02'),
(86, 65, 4, '2017-08-17'),
(87, 54, 7, '2015-12-19'),
(88, 29, 4, '2017-07-22'),
(89, 44, 9, '2017-12-31'),
(90, 56, 31, '2015-06-13'),
(91, 17, 4, '2015-04-01'),
(92, 35, 16, '2018-07-19'),

(93, 22, 18, '2017-06-22'),
(94, 39, 24, '2015-05-29'),
(95, 63, 14, '2018-01-20'),
(96, 53, 21, '2016-07-31'),
(97, 40, 9, '2016-07-10'),
(98, 52, 4, '2017-04-05'),
(99, 27, 20, '2016-09-04'),
(100, 72, 29, '2015-12-06'),
(101, 49, 16, '2017-12-19'),
(102, 6, 12, '2016-12-04'),
(103, 74, 31, '2016-07-27'),
(104, 48, 32, '2016-06-29'),
(105, 69, 2, '2016-12-27'),
(106, 60, 32, '2017-10-29'),
(107, 45, 22, '2017-06-12'),
(108, 42, 15, '2017-05-14'),
(109, 79, 8, '2016-10-13'),
(110, 70, 18, '2016-12-04'),
(111, 34, 8, '2016-03-06'),
(112, 43, 8, '2015-12-19'),
(113, 42, 32, '2016-04-20'),
(114, 67, 5, '2017-03-06'),
(115, 80, 25, '2015-06-23'),
(116, 54, 11, '2017-05-03'),
(117, 34, 28, '2017-08-30'),
(118, 65, 20, '2017-08-26'),
(119, 61, 19, '2018-01-05'),
(120, 38, 12, '2018-01-17'),
(121, 51, 4, '2016-05-13'),
(122, 7, 16, '2016-03-17'),
(123, 46, 16, '2016-11-25'),
(124, 75, 30, '2018-08-12'),
(125, 72, 32, '2018-08-12'),
(126, 51, 4, '2018-02-28'),
(127, 46, 4, '2016-09-14'),
(128, 40, 26, '2016-07-01'),
(129, 59, 9, '2017-01-09'),
(130, 47, 32, '2017-10-21'),
(131, 34, 13, '2015-07-14'),
(132, 29, 28, '2017-06-26'),
(133, 35, 23, '2016-11-10'),
(134, 41, 26, '2015-06-19'),
(135, 77, 13, '2016-11-20'),
(136, 68, 31, '2017-01-01'),
(137, 37, 26, '2017-08-09'),
(138, 77, 10, '2016-06-25'),
(139, 21, 2, '2016-11-19'),
(140, 57, 14, '2016-01-09'),
(141, 35, 20, '2017-06-14'),

(142, 27, 13, '2016-07-29'),
(143, 68, 22, '2015-05-04'),
(144, 51, 32, '2015-04-14'),
(145, 26, 19, '2017-01-28'),
(146, 54, 18, '2018-04-11'),
(147, 56, 8, '2016-08-09'),
(148, 53, 10, '2017-02-26'),
(149, 63, 15, '2016-07-23'),
(150, 71, 2, '2016-10-02'),
(151, 47, 5, '2017-04-17'),
(152, 31, 5, '2017-05-16'),
(153, 31, 4, '2016-05-20'),
(154, 25, 15, '2015-11-21'),
(155, 78, 16, '2016-01-05'),
(156, 64, 26, '2016-11-28'),
(157, 24, 4, '2016-07-09'),
(158, 64, 23, '2016-02-07'),
(159, 50, 14, '2017-09-13'),
(160, 19, 16, '2017-05-06'),
(161, 38, 9, '2016-02-28'),
(162, 46, 9, '2016-04-18'),
(163, 78, 9, '2015-10-03'),
(164, 27, 13, '2015-06-22'),
(165, 20, 9, '2018-07-24'),
(166, 72, 2, '2017-09-08');

1. Display all contents of the Clients table. select all from client

```
SELECT *
FROM client;
```

1	**Kaiden**	**Hill**	**2006**	**Student**
2	**Alina**	**Morton**	**2010**	**Student**
3	**Fania**	**Brooks**	**1983**	**Food Scientist**
4	**Courtney**	**Jensen**	**2006**	**Student**
5	**Brittany**	**Hill**	**1983**	**Firefighter**
6	**Max**	**Rogers**	**2005**	**Student**
7	**Margaret**	**McCarthy**	**1981**	**School Psychologist**
8	**Julie**	**McCarthy**	**1973**	**Professor**
9	**Ken**	**McCarthy**	**1974**	**Securities Clerk**

10	Britany	O'Quinn	1984	Violinist
11	Conner	Gardner	1998	Licensed Massage Therapist
12	Mya	Austin	1960	Parquet Floor Layer
13	Thierry	Rogers	2004	Student
14	Eloise	Rogers	1984	Computer Security Manager
15	Gerard	Jackson	1979	Oil Exploration Engineer
16	Randy	Day	1986	Aircraft Electrician
17	Jodie	Page	1990	Manufacturing Director
18	Coral	Rice	1996	Window Washer
19	Ayman	Austin	2002	Student
20	Jaxson	Austin	1999	Repair Worker
21	Joel	Austin	1973	Police Officer
22	Alina	Austin	2010	Student
23	Elin	Austin	1962	Payroll Clerk
24	Ophelia	Wolf	2004	Student
25	Eliot	McGuire	1967	Dentist
26	Peter	McKinney	1968	Professor
27	Annabella	Henry	1974	Nurse
28	Anastasia	Baker	2001	Student
29	Tyler	Baker	1984	Police Officer
30	Lilian	Ross	1983	Insurance Agent
31	Thierry	Arnold	1975	Bus Driver
32	Angelina	Rowe	1979	Firefighter
33	Marcia	Rowe	1974	Health Educator
34	Martin	Rowe	1976	Ship Engineer
35	Adeline	Rowe	2005	Student

36	Colette	Rowe	1963	Professor
37	Diane	Clark	1975	Payroll Clerk
38	Caroline	Clark	1960	Dentist
39	Dalton	Clayton	1982	Police Officer
40	Steve	Clayton	1990	Bus Driver
41	Melanie	Clayton	1987	Computer Engineer
42	Alana	Wilson	2007	Student
43	Carson	Byrne	1995	Food Scientist
44	Conrad	Byrne	2007	Student
45	Ryan	Porter	2008	Student
46	Elin	Porter	1978	Computer Programmer
47	Tyler	Harvey	2007	Student
48	Arya	Harvey	2008	Student
49	Serena	Harvey	1978	School Teacher
50	Lilly	Franklin	1976	Doctor
51	Mai	Franklin	1994	Dentist
52	John	Franklin	1999	Firefighter
53	Judy	Franklin	1995	Firefighter
54	Katy	Lloyd	1992	School Teacher
55	Tamara	Allen	1963	Ship Engineer
56	Maxim	Lyons	1985	Police Officer
57	Allan	Lyons	1983	Computer Engineer
58	Marc	Harris	1980	School Teacher
59	Elin	Young	2009	Student
60	Diana	Young	2008	Student
61	Diane	Young	2006	Student

62	**Alana**	**Bird**	**2003**	**Student**
63	**Anna**	**Becker**	**1979**	**Security Agent**
64	**Katie**	**Grant**	**1977**	**Manager**
65	**Joan**	**Grant**	**2010**	**Student**
66	**Bryan**	**Bell**	**2001**	**Student**
67	**Belle**	**Miller**	**1970**	**Professor**
68	**Peggy**	**Stevens**	**1990**	**Bus Driver**
69	**Steve**	**Williamson**	**1975**	**HR Clerk**
70	**Tyler**	**Williamson**	**1999**	**Doctor**
71	**Izabelle**	**Williamson**	**1990**	**Systems Analyst**
72	**Annabel**	**Williamson**	**1960**	**Cashier**
73	**Mohamed**	**Waters**	**1966**	**Insurance Agent**
74	**Marion**	**Newman**	**1970**	**Computer Programmer**
75	**Ada**	**Williams**	**1986**	**Computer Programmer**
76	**Sean**	**Scott**	**1983**	**Bus Driver**
77	**Farrah**	**Scott**	**1974**	**Ship Engineer**
78	**Christine**	**Lambert**	**1973**	**School Teacher**
79	**Alysha**	**Lambert**	**2007**	**Student**
80	**Maia**	**Grant**	**1984**	**School Teacher**

2. First names, last names, ages and occupations of all clients. Select only first and last columns from client

```
SELECT clientfirstname, clientlastname, FLOOR(DATEDIFF(CURDATE(), clientdob) / 365) AS age,
occupation
FROM client;
```

Kaiden	**Hill**	**Student**
Alina	**Morton**	**Student**
Fania	**Brooks**	**Food Scientist**
Courtney	**Jensen**	**Student**
Brittany	**Hill**	**Firefighter**
Max	**Rogers**	**Student**
Margaret	**McCarthy**	**School Psychologist**
Julie	**McCarthy**	**Professor**
Ken	**McCarthy**	**Securities Clerk**
Britany	**O'Quinn**	**Violinist**
Conner	**Gardner**	**Licensed Massage Therapist**
Mya	**Austin**	**Parquet Floor Layer**
Thierry	**Rogers**	**Student**
Eloise	**Rogers**	**Computer Security Manager**
Gerard	**Jackson**	**Oil Exploration Engineer**
Randy	**Day**	**Aircraft Electrician**
Jodie	**Page**	**Manufacturing Director**
Coral	**Rice**	**Window Washer**
Ayman	**Austin**	**Student**
Jaxson	**Austin**	**Repair Worker**
Joel	**Austin**	**Police Officer**
Alina	**Austin**	**Student**
Elin	**Austin**	**Payroll Clerk**
Ophelia	**Wolf**	**Student**
Eliot	**McGuire**	**Dentist**
Peter	**McKinney**	**Professor**

Annabella	**Henry**	**Nurse**
Anastasia	**Baker**	**Student**
Tyler	**Baker**	**Police Officer**
Lilian	**Ross**	**Insurance Agent**
Thierry	**Arnold**	**Bus Driver**
Angelina	**Rowe**	**Firefighter**
Marcia	**Rowe**	**Health Educator**
Martin	**Rowe**	**Ship Engineer**
Adeline	**Rowe**	**Student**
Colette	**Rowe**	**Professor**
Diane	**Clark**	**Payroll Clerk**
Caroline	**Clark**	**Dentist**
Dalton	**Clayton**	**Police Officer**
Steve	**Clayton**	**Bus Driver**
Melanie	**Clayton**	**Computer Engineer**
Alana	**Wilson**	**Student**
Carson	**Byrne**	**Food Scientist**
Conrad	**Byrne**	**Student**
Ryan	**Porter**	**Student**
Elin	**Porter**	**Computer Programmer**
Tyler	**Harvey**	**Student**
Arya	**Harvey**	**Student**
Serena	**Harvey**	**School Teacher**
Lilly	**Franklin**	**Doctor**
Mai	**Franklin**	**Dentist**
John	**Franklin**	**Firefighter**

Judy	**Franklin**	**Firefighter**
Katy	**Lloyd**	**School Teacher**
Tamara	**Allen**	**Ship Engineer**
Maxim	**Lyons**	**Police Officer**
Allan	**Lyons**	**Computer Engineer**
Marc	**Harris**	**School Teacher**
Elin	**Young**	**Student**
Diana	**Young**	**Student**
Diane	**Young**	**Student**
Alana	**Bird**	**Student**
Anna	**Becker**	**Security Agent**
Katie	**Grant**	**Manager**
Joan	**Grant**	**Student**
Bryan	**Bell**	**Student**
Belle	**Miller**	**Professor**
Peggy	**Stevens**	**Bus Driver**
Steve	**Williamson**	**HR Clerk**
Tyler	**Williamson**	**Doctor**
Izabelle	**Williamson**	**Systems Analyst**
Annabel	**Williamson**	**Cashier**
Mohamed	**Waters**	**Insurance Agent**
Marion	**Newman**	**Computer Programmer**
Ada	**Williams**	**Computer Programmer**
Sean	**Scott**	**Bus Driver**
Farrah	**Scott**	**Ship Engineer**
Christine	**Lambert**	**School Teacher**

Alysha	**Lambert**	**Student**
Maia	**Grant**	**School Teacher**

3. First and last names of clients that borrowed books in March 2018. Join book and client

```
SELECT clientFirstName, clientLastName
FROM client
INNER JOIN borrower ON client.clientid = borrower.clientid
INNER JOIN book ON borrower.bookid = book.bookid
WHERE borrower.borrowdate >= '2018-03-01'
  AND borrower.borrowdate < '2018-04-01'
```

Maia	**Grant**
Marcia	**Rowe**
Alysha	**Lambert**
Tyler	**Baker**
Katy	**Lloyd**
Angelina	**Rowe**
Gerard	**Jackson**
Carson	**Byrne**

4. First and last names of the top 5 authors clients borrowed in 2017. Join book and author, borrower and book, and borrower and client.

```
SELECT author.authorfirstname, author.authorlastname
FROM author
INNER JOIN book ON author.authorid = book.authorid
INNER JOIN borrower ON book.bookid = borrower.bookid
INNER JOIN client ON borrower.clientid = client.clientid
WHERE YEAR(borrower.borrowdate) = 2017
GROUP BY author.authorid, author.authorfirstname, author.authorlastname
ORDER BY COUNT(*) DESC
LIMIT 5;
```

Elena	**Martin**
Logan	**Moore**
Sofia	**Smith**
Maria	**Brown**
Zoe	**Roy**

5. Nationalities of the least 5 authors that clients borrowed during the years 2015-2017. Join book and author, borrower and book.

```
SELECT author.authornationality
FROM author
INNER JOIN book ON author.authorid = book.authorid
INNER JOIN borrower ON book.bookid = borrower.bookid
WHERE YEAR(borrower.borrowdate) BETWEEN 2015 AND 2017
GROUP BY author.authornationality
ORDER BY COUNT(*) ASC
LIMIT 5;
```

Spain
Great Britain
China
Brazil
France

6. The book that was most borrowed during the years 2015-2017. Join borrower and book, where statement with years.

```
SELECT book.booktitle, COUNT(*) AS borrow_count
FROM book
INNER JOIN borrower ON book.bookid = borrower.bookid
WHERE YEAR(borrower.borrowdate) BETWEEN 2015 AND 2017
GROUP BY book.booktitle
ORDER BY borrow_count DESC
LIMIT 1;
```

7. Top borrowed genres for client born in years 1970-1980. Created a count on the genre returns to total the amounts.

```
SELECT book.genre, COUNT(*) AS borrow_count
FROM client
INNER JOIN borrower ON client.clientid = borrower.clientid
INNER JOIN book ON borrower.bookid = book.bookid
WHERE client.clientdob BETWEEN 1970 AND 1980
GROUP BY book.genre
ORDER BY borrow_count DESC;
```

Science	**24**
Fiction	**16**
Well being	**15**
Humor	**5**
Society	**4**
Children	**3**
History	**3**
Literature	**3**
Law	**3**

8. Top 5 occupations that borrowed the most in 2016. Join borrower and client, used count again.

```
SELECT client.occupation, COUNT(*) AS borrow_count
FROM client
INNER JOIN borrower ON client.clientid = borrower.clientid
WHERE YEAR(borrower.borrowdate) = 2016
GROUP BY client.occupation
ORDER BY borrow_count DESC
LIMIT 5;
```

Student	**32**
Bus Driver	**8**
Dentist	**6**
Computer Programmer	**6**
Police Officer	**5**

9. Average number of borrowed books by job title. Used average on top of count for this.

```
SELECT client.occupation, AVG(borrow_count) AS average_borrowed_books
FROM (
    SELECT borrower.clientid, COUNT(*) AS borrow_count
    FROM borrower
    INNER JOIN client ON borrower.clientid = client.clientid
    GROUP BY borrower.clientid
) AS subquery
INNER JOIN client ON subquery.clientid = client.clientid
GROUP BY client.occupation;
```

Student	**4.4211**
Firefighter	**3.2500**
Health Educator	**2.0000**
Bus Driver	**4.0000**
Manager	**3.0000**
Police Officer	**4.5000**
Payroll Clerk	**3.0000**
Computer Programmer	**5.6667**
School Teacher	**3.6000**
Dentist	**5.6667**
Food Scientist	**5.0000**

Insurance Agent	**4.0000**
Systems Analyst	**4.0000**
Computer Engineer	**3.0000**
Repair Worker	**3.0000**
Cashier	**5.0000**
Parquet Floor Layer	**2.0000**
Ship Engineer	**4.0000**
Nurse	**7.0000**
HR Clerk	**4.0000**
Manufacturing Director	**5.0000**
Professor	**3.5000**
Security Agent	**2.0000**
Oil Exploration Engineer	**5.0000**
Licensed Massage Therapist	**2.0000**
Securities Clerk	**2.0000**
Violinist	**4.0000**
Doctor	**4.0000**
School Psychologist	**2.0000**
Computer Security Manager	**6.0000**
Aircraft Electrician	**2.0000**
Window Washer	**2.0000**

10. Create a VIEW and display the titles that were borrowed by at least 20% of clients. Created the view first using same concepts as previous and then called the view with an asterisk on popular_books.

```
CREATE VIEW popular_books AS
SELECT book.booktitle, COUNT(DISTINCT borrower.clientid) AS borrower_count
FROM book
```

```
INNER JOIN borrower ON book.bookid = borrower.bookid
GROUP BY book.booktitle
HAVING COUNT(DISTINCT borrower.clientid) >= 0.2 * (SELECT COUNT(DISTINCT clientid) FROM
borrower);

SELECT * FROM popular_books;
```

Electrical transformers 17

11. The top month of borrows in 2017. Took the month out of the date, had to set the field to date for this to work.

```
SELECT EXTRACT(MONTH FROM borrowdate) AS borrow_month, COUNT(*) AS borrow_count
FROM borrower
WHERE EXTRACT(YEAR FROM borrowdate) = 2017
GROUP BY borrow_month
ORDER BY borrow_count DESC
LIMIT 1;
```

Aug - 10 borrows
8 10

12. Average number of borrows by age. Subtracted birth year from the current year to create an age. After that I used the average function as per previous examples.

```
SELECT client_age AS age, AVG(borrow_count) AS average_borrows
FROM (
   SELECT borrower.clientid, COUNT(*) AS borrow_count,
        YEAR(CURDATE()) - client.clientdob AS client_age
   FROM borrower
   INNER JOIN client ON borrower.clientid = client.clientid
   WHERE client.clientdob IS NOT NULL
   GROUP BY borrower.clientid
) AS subquery
GROUP BY client_age
ORDER BY client_age;
```

13 2.3333

15 6.0000

16	**5.0000**
17	**5.5000**
18	**4.5000**
19	**3.0000**
20	**5.0000**
21	**2.0000**
22	**4.5000**
24	**3.6667**
25	**2.0000**
27	**2.0000**
28	**4.5000**
29	**10.0000**
31	**3.0000**
33	**5.5000**
36	**2.0000**
37	**3.0000**
38	**4.0000**
39	**5.5000**
40	**3.7500**
41	**3.0000**
42	**2.0000**
43	**1.0000**
44	**4.3333**
45	**5.5000**
46	**3.0000**
47	**3.5000**

48 2.6667

49 3.2500

50 3.6667

53 4.5000

55 4.0000

56 3.0000

57 1.0000

60 5.0000

61 3.0000

63 3.6667

13. The oldest and the youngest clients of the library. A very simple code for min and max. I created age using the current year and date of birth as I did previously.

```
SELECT YEAR(CURDATE()) - MIN(client.clientdob) AS oldest_age, YEAR(CURDATE()) -
MAX(client.clientdob) AS youngest_age
FROM client
```

63 13

14. First and last names of authors that wrote books in more than one genre. This one returned no rows but also no errors, I don't believe these parameters apply to anyone in the author table.

```
SELECT author.authorfirstname, author.authorlastname
FROM author
INNER JOIN book ON author.authorid = book.authorid
GROUP BY author.authorfirstname, author.authorlastname
HAVING COUNT(DISTINCT book.genre) > 1;
);
```

This returned 0 rows, When checking the table it appears this might be correct, I dont think there are any authors that have books in multiple genres in the author table.