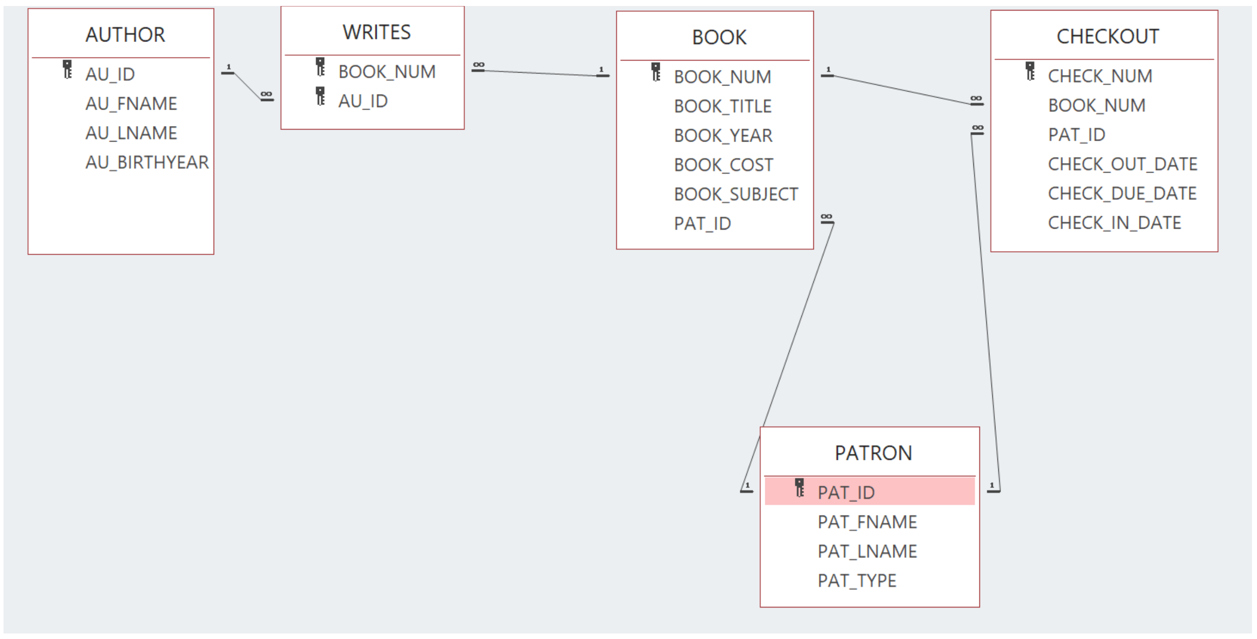
Jenna Bennett
Julia Anderson
Kelsey Ridener
Noah Smith
Miles Blomgren
Lucas Weaver

Team Name: Team

Final Group Project

Part I:

1. 

56. Write a query that displays the book title, cost and year of publication for every book in the system. Sort the results by book title.

SELECT BOOK_TITLE, BOOK_COST, BOOK_YEAR
FROM BOOK
ORDER BY BOOK_TITLE

57. Write a query that displays the first and last name of every patron, sorted by last name and then the first name. Ensure the sort is case insensitive (Figure P7.57). (50 rows)

SELECT PAT_FNAME, PAT_LNAME
FROM PATRON
ORDER BY UPPER (PAT_LNAME), UPPER(PAT_FNAME)

58. Write a query to display the checkout number, checkout date, and due date for every book that has been checked out sorted by checkout number (Figure P7.58). (68 rows)

```
SELECT CHECKOUT.CHECK_NUM, CHECKOUT.CHECK_OUT_DATE,
CHECKOUT.CHECK_DUE_DATE
FROM CHECKOUT
ORDER BY CHECKOUT.CHECK_NUM;
```

59. **Write a query to display the book number, book title, and subject for every book sorted by book number (Figure P7.59). (20 rows)**

```
SELECT BOOK_NUM, BOOK_TITLE AS TITLE, BOOK_SUBJECT AS [Subject of Book]
FROM BOOK
ORDER BY BOOK_NUM;
```

60. **Write a query to display the different years in which books have been published. Include each year only once and sort the results by year (Figure P7.60).**

```
SELECT DISTINCT BOOK_YEAR
FROM BOOK
ORDER BY BOOK_YEAR;
```

61. **Write a query to display the different subjects on which FACT has books. Include each subject only once and sort the results by subject (Figure P7.61).**

```
SELECT DISTINCT BOOK_SUBJECT
FROM BOOK
ORDER BY BOOK_SUBJECT;
```

62. **Write a query to display the book number, title, and cost of each book sorted by book number (Figure P7.62).**

```
SELECT BOOK_NUM, BOOK_TITLE, BOOK_COST, AS “Replacement Cost”
FROM BOOK
ORDER BY BOOK_NUM;
```

63. **Write a query to display the checkout number, book number, patron ID, checkout date, and due date for every checkout that has ever occurred in the system. Sort the results by checkout date in descending order and then by checkout number in ascending order (Figure P7.63). (68 rows)**

```
SELECT CHECKOUT.CHECK_NUM, CHECKOUT.BOOK_NUM, CHECKOUT.PAT_ID,
CHECKOUT.CHECK_OUT_DATE, CHECKOUT.CHECK_DUE_DATE
FROM CHECKOUT
ORDER BY CHECKOUT.CHECK_OUT_DATE DESC, CHECKOUT.CHECK_NUM ASC;
```

64. **Write a query to display the book title, year, and subject for every book. Sort the results by book subject in ascending order, year in descending order, and then title in ascending order (Figure P7.64). (20 rows)**

```
SELECT BOOK_TITLE, BOOK_YEAR, BOOK_SUBJECT
FROM BOOK
ORDER BY BOOK_SUBHECT, BOOK_YEAR DESC, BOOK_TITLE;
```

65. **Write a query to display the book number, title, and cost for all books that cost $59.95 sorted by book number (Figure P7.65).**

```
SELECT BOOK_NUM, BOOK_TITLE, BOOK_COST
FROM BOOK WHERE (((BOOK_COST)=59.95))
ORDER BY BOOK_NUM;
```

66. **Write a query to display the book number, title, and replacement cost for all books in the “Database” subject sorted by book number (Figure P7.66).**

```
SELECT BOOK_NUM, BOOK_TITLE, BOOK_COST
FROM BOOK WHERE (((BOOK_SUBJECT)=”Database”))
ORDER BY BOOK_NUM;
```

67. **Write a query to display the checkout number, book number, and checkout date of all books checked out before April 5, 2017 sorted by checkout number (Figure P7.67).**

```
SELECT CHECK_NUM, BOOK_NUM, CHECK_OUT_DATE
FROM CHECKOUT WHERE (((CHECK_OUT_DATE)<#4/5/2017#))
ORDER BY CHECK_NUM;
```

68. **Write a query to display the book number, title, and year of all books published after 2015 and on the “Programming” subject sorted by book number (Figure P7.68).**

```
SELECT BOOK_NUM,BOOK_TITLE, BOOK_YEAR
FROM BOOK WHERE BOOK_YEAR > 2015 AND BOOK_SUBJECT = ‘Programming”
ORDER BY BOOK NUM;
```

69. **Write a query to display the book number, title, subject, and cost for all books that are on the subjects of “Middleware” or “Cloud,” and that cost more than $70 sorted by book number (Figure P7.69).**

```
SELECT BOOK_NUM, BOOK_TITLE, BOOK_SUBJECT, BOOK_COST
FROM BOOK WHERE (BOOK_SUBJECT =’Middleware’ OR BOOK_SUBJECT = ‘Cloud’) AND
BOOK_COST> 70
ORDER BY BOOK_NUM;
```

70. **Write a query to display the author ID, first name, last name, and year of birth for all authors born in the decade of the 1980s sorted by author ID (Figure P7.70).**

```
SELECT AU_ID, AU_FNAME,AU_LNAME,AU_BIRTHYEAR
FROM AUTHOR WHERE AU_BIRTHYEAR BETWEEN 1980 AND 1989
ORDER BY AU_ID;
```

71. **Write a query to display the book number, title, and subject for all books that contain the word “Database” in the title, regardless of how it is capitalized. Sort the results by book number (Figure P7.71).**

```
SELECT BOOK_NUM, BOOK_TITLE, BOOK_SUBJECT
FROM BOOK_TITLE LIKE “*database*”
ORDER BY BOOK_NUM;
```

72. **Write a query to display the patron ID, first and last name of all patrons who are students, sorted by patron ID (Figure P7.72). (44 rows)**

```
SELECT PAT_ID, PAT_FNAME, PAT_LNAME
FROM PATRON WHERE PAT_TYPE=”Student”
ORDER BY PAT_ID;
```

73. **Write a query to display the patron ID, first and last name, and patron type for all patrons whose last name begins with the letter “C,” sorted by patron ID (Figure P7.73).**

```
SELECT PAT_ID, PAT_FNAME, PAT_LNAME
FROM PATRON
WHERE Upper(PAT_TYPE)= ‘STUDENT’;
```

74. **Write a query to display the author ID, first and last name of all authors whose year of birth is unknown. Sort the results by author ID (Figure P7.74).**

```
SELECT AU_ID, AU_FNAME, AU_LNAME
FROM AUTHOR WHERE AU_BIRTHYEAR IS NULL
ORDER BY AU_ID;
```

75. **Write a query to display the author ID, first and last name of all authors whose year of birth is known. Ensure the results are sorted by author ID (Figure P7.75).**

```
SELECT AU_ID, AU_FNAME, AU_LNAME
FROM AUTHOR WHERE AU_BIRTHYEAR IS NOT NULL
ORDER BY AU_ID;
```

76. **Write a query to display the checkout number, book number, patron ID, checkout date, and due date for all checkouts that have not yet been returned. Sort the results by book number (Figure P7.76).**

```
SELECT CHECK_NUM, BOOK_NUM, PAT_ID,CHECK_OUT_DATE,CHECK_DUE_DATE
FROM CHECKOUT
WHERE CHECK_IN_DATE is NULL
ORDER BY BOOK-NUM;
```

77. **Write a query to display the author ID, first name, last name, and year of birth for all authors. Sort the results in descending order by year of birth, and then in ascending order by last name (Figure P7.77). (Note: Some DBMS sort NULLs as being large and some DBMS sort NULLs as being small.)**

```
SELECT AU_ID, AU_FNAME, AU_LNAME, AU_BIRTHYEAR
FROM AUTHOR
ORDER BY AU_BIRTHYEAR DESC, AU_LNAME;
```

78. **Write a query to display the number of books in the FACT system (Figure P7.78).**

```
SELECT COUNT(BOOK_NUM) AS “Number of Books”
FROM BOOK;
```

79. **Write a query to display the number of different book subjects in the FACT system (Figure P7.79).**

```
SELECT COUNT([B].BOOK_SUBJECT) AS [Number of Subjects]
FROM (SELECT DISTINCT BOOK_SUBJECT FROM BOOK AS [B];
```

80. **Write a query to display the number of books that are available (not currently checked out) (Figure P7.80).**

```
SELECT COUNT(BOOK_NUM) AS [Available Books]
FROM BOOK WHERE PAT_ID IS NULL;
```

81. **Write a query to display the highest book cost in the system (Figure P7.81).**

```
SELECT MAX(BOOK_COST) AS [Most Expensive]
FROM BOOK;
```

82. **Write a query to display the lowest book cost in the system (Figure P7.82).**

SELECT MIN(BOOK_COST) AS [Least Expensive]
FROM BOOK;

83. **Write a query to display the number of different patrons who have ever checked out a book (Figure P7.83).**

SELECT COUNT(P.[PAT.ID]) AS [DIFFERENT PATRONS] FROM (SELECT DISTINCT PAT_ID FROM CHECKOUT) AS [P];

84. **Write a query to display the subject and the number of books in each subject. Sort the results by the number of books in descending order and then by subject name in ascending order (Figure P7.84).**

SELECT BOOK_SUBJECT, COUNT(BOOK_NUM) AS [Book In Subject]
FROM BOOK
GROUP BY BOOK_SUBJECT
ORDER BY COUNT(BOOK_NUM) DESC, BOOK_SUBJECT;

85. **Write a query to display the author ID and the number of books written by that author. Sort the results in descending order by number of books, then in ascending order by author ID (Figure P7.85).**

SELECT AU_ID, COUNT(BOOK_NUM) AS [Books Written]
FROM WRITES
GROUP BY AU_ID
ORDER BY COUNT(BOOK_NUM) DESC, AU_ID ASC;

86. **Write a query to display the total value of all books in the library (Figure P7.86).**

SELECT SUM (BOOK_COST AS [Library Value]
FROM BOOK;

87. **Write a query to display the patron ID, book number, and days kept for each checkout. "Days Kept" is the difference from the date on which the book is returned to the date it was checked out. Sort the results by days kept in descending order, then by patron ID, and then by book number (Figure P7.87). (68 rows)**

SELECT PAT_ID, PAT_FNAME&" "&PAT_LNAME AS [Patron Name], PAT_TYPE
FROM PATRON
ORDER BY PAT_ID;

88. **Write a query to display the patron ID, patron full name, and patron type for each patron sorted by patron ID (Figure P7.88). (50 rows)**

SELECT BOOK.BOOK_NUM, [BOOK_TITLE] &"("&[BOOK_YEAR]&")" AS BOOK, BOOK.BOOK_SUBJECT
FROM BOOK;

89. **Write a query to display the book number, title with year, and subject for each book. Sort the results by the book number (Figure P7.89). (20 rows)**

SELECT BOOK_NUM, CONCAT(BOOK_TITLE, ' (', BOOK_YEAR, ')') AS BOOK, BOOK_SUBJECT
FROM BOOK
ORDER BY BOOK_NUM

90. **Write a query to display the author last name, author first name, and book number for each book written by that author. Sort the results by author last name, first name, and then book number (Figure P7.90). (25 rows)**

```
SELECT AU_LNAME, AU_FNAME, BOOK_NUM
FROM AUTHOR JOIN WRITES ON AUTHOR.AU_ID = WRITES.AU_ID
ORDER BY AU_LNAME, AU_FNAME, BOOK_NUM
```

91. **Write a query to display the author ID, book number, title, and subject for each book. Sort the results by book number and then author ID (Figure P7.91). (25 rows)**

```
SELECT AU_ID, BOOK.BOOK_NUM, BOOK_TITLE, BOOK_SUBJECT
FROM BOOK JOIN WRITES ON BOOK.BOOK_NUM = WRITES.BOOK_NUM
ORDER BY BOOK.BOOK_NUM, AU_ID
```

92. **Write a query to display the author last name, first name, book title, and replacement cost for each book. Sort the results by book number and then author ID (Figure P7.92). (25 rows)**

```
SELECT AU_LNAME, AU_FNAME, BOOK_TITLE, BOOK_COST
FROM AUTHOR JOIN WRITES ON AUTHOR.AU_ID = WRITES.AU_ID JOIN BOOK ON
WRITES.BOOK_NUM = BOOK.BOOK_NUM
ORDER BY BOOK.BOOK_NUM, AUTHOR.AU_ID
```

93. **Write a query to display the patron ID, book number, patron first name and last name, and book title for all currently checked out books. (Remember to use the redundant relationship described in the assignment instructions for current checkouts.) Sort the output by patron last name and book title (Figure P7.93).**

```
SELECT PATRON.PAT_ID, BOOK_NUM, PAT_FNAME, PAT_LNAME, BOOK_TITLE
FROM PATRON JOIN BOOK ON PATRON.PAT_ID = BOOK.PAT_ID
ORDER BY PAT_LNAME, BOOK_TITLE
```

94. **Write a query to display the patron ID, full name (first and last), and patron type for all patrons. Sort the results by patron type and then by last name and first name. Ensure that all sorting is case insensitive (Figure P7.94). (50 rows)**

```
SELECT PAT_ID, CONCAT(PAT_FNAME, ' ', PAT_LNAME) AS NAME, PAT_TYPE
FROM PATRON
ORDER BY Upper(PAT_TYPE), Upper(PAT_LNAME), PAT_FNAME;
```

95. **Write a query to display the book number and the number of times each book has been checked out. Do not include books that have never been checked out. Sort the results by the number of times checked out in descending order and then by book number in descending order (Figure P7.95).**

```
SELECT BOOK_NUM, Count(*) AS "Times Checked Out"
FROM CHECKOUT
GROUP BY BOOK_NUM
ORDER BY COUNT(*) DESC, BOOK_NUM DESC
```

96. **Write a query to display the author ID, first and last name, book number, and book title of all books in the subject “Cloud.” Sort the results by book title and then by author last name (Figure P7.96).**

```
SELECT AUTHOR.AU_ID, AU_FNAME, AU_LNAME, BOOK.BOOK_NUM, BOOK_TITLE
```

```
FROM AUTHOR JOIN WRITES ON AUTHOR.AU_ID = WRITES.AU_ID JOIN BOOK ON
WRITES.BOOK_NUM = BOOK.BOOK_NUM
WHERE BOOK_SUBJECT = 'Cloud'
ORDER BY BOOK_TITLE, AU_LNAME
```

97. **Write a query to display the book number, title, author last name, author first name, patron ID, last name, and patron type for all books currently checked out to a patron. Sort the results by book title (Figure P7.97).**

```
SELECT BOOK_NUM, BOOK_TITLE, PATRON.PAT_ID, PAT_LNAME, PAT_TYPE
FROM BOOK JOIN PATRON ON BOOK.PAT_ID = PATRON.PAT_ID
ORDER BY BOOK_TITLE
```

98. **Write a query to display the book number, title, and number of times each book has been checked out. Include books that have never been checked out. Sort the results in descending order by the number of times checked out and then by title (Figure P7.98).**

```
SELECT BOOK.BOOK_NUM, BOOK_TITLE, Count(CHECK_NUM) AS "Times Checked Out"
FROM BOOK LEFT JOIN CHECKOUT ON BOOK.BOOK_NUM = CHECKOUT.BOOK_NUM
GROUP BY BOOK.BOOK_NUM, BOOK_TITLE
ORDER BY COUNT(CHECK_NUM) DESC, BOOK_TITLE
```

99. **Write a query to display the book number, title, and number of times each book has been checked out. Limit the results to books that have been checked out more than five times. Sort the results in descending order by the number of times checked out and then by title (Figure P7.99).**

```
SELECT BOOK.BOOK_NUM, BOOK_TITLE, Count(CHECK_NUM) AS "Times Checked Out"
FROM BOOK JOIN CHECKOUT ON BOOK.BOOK_NUM = CHECKOUT.BOOK_NUM
GROUP BY BOOK.BOOK_NUM, BOOK_TITLE
HAVING Count(CHECK_NUM) > 5
ORDER BY COUNT(CHECK_NUM) DESC, BOOK_TITLE
```

100. **Write a query to display the author ID, author last name, book title, checkout date, and patron last name for all the books written by authors with the last name “Bruer” that have ever been checked out by patrons with the last name “Miles.” Sort the results by check out date (Figure P7.100).**

```
SELECT AUTHOR.AU_ID, AU_LNAME, BOOK_TITLE, CHECK_OUT_DATE, PAT_LNAME
FROM AUTHOR JOIN WRITES ON AUTHOR.AU_ID = WRITES.AU_ID JOIN BOOK ON
WRITES.BOOK_NUM = BOOK.BOOK_NUM
JOIN CHECKOUT ON BOOK.BOOK_NUM = CHECKOUT.BOOK_NUM JOIN PATRON ON
PATRON.PAT_ID = CHECKOUT.PAT_ID
WHERE PAT_LNAME = 'Miles' AND AU_LNAME = 'Bruer'
ORDER BY CHECK_OUT_DATE
```

101. **Write a query to display the patron ID, first and last name of all patrons who have never checked out any book. Sort the result by patron last name and then first name (Figure P7.101).**

```
SELECT PATRON.PAT_ID, PAT_FNAME, PAT_LNAME
FROM PATRON LEFT JOIN CHECKOUT ON PATRON.PAT_ID = CHECKOUT.PAT_ID
WHERE CHECK_NUM IS NULL
```

ORDER BY PAT_LNAME, PAT_FNAME

102. Write a query to display the patron ID, last name, number of times that patron has ever checked out a book, and the number of different books the patron has ever checked out. For example, if a given patron has checked out the same book twice, that would count as two checkouts but only one book. Limit the results to only patrons who have made at least three checkouts. Sort the results in descending order by number of books, then in descending order by number of checkouts, and then in ascending order by patron ID (Figure P7.102).

SELECT PATRON>PAT_ID, PAT_LNAME, Count (CHECK_NUM) AS "NUM CHECKOUTS", Count (DISTINCT BOOK_NUM) AS "NUM DIFFERENT BOOKS"
FROM CHECKOUT JOIN PATRON ON CHECKOUT>PAT_ID = PATRON>PAT_ID
GROUP BY PATRON.PAT_ID, PAT_LNAME
HAVING Count (CHECK_NUM) > 2
ORDER BY COUNT (DISTINCT BOOK_NUM) DESC, COUNT (CHECK_NUM) DESC, PATRON>PAT_ID

103. Write a query to display the average number of days a book is kept during a checkout (Figure P7.103).

SELECT Round(Avg(datediff(CHECK_IN_DATE, CHECK_OUT_DATE)), 2)
AS "Average Days Kept"
FROM CHECKOUT

104. Write a query to display the patron ID and the average number of days that patron keeps books during a checkout. Limit the results to only patrons who have at least three checkouts. Sort the results in descending order by the average days the book is kept (Figure P7.104).

SELECT PAT_ID, Round(Avg(datediff(CHECK_IN_DATE, CHECK_OUT_DATE)), 2) AS "Average Days Kept"
FROM CHECKOUT
GROUP BY PAT_ID
HAVING Count (CHECK_NUM) > 2
ORDER BY Round(Avg(datediff(CHECK_IN_DATE, CHECK_OUT_DATE)), 2) DESC

105. Write a query to display the book number, title, and cost of books that have the lowest cost of any books in the system. Sort the results by book number (Figure P7.105).

SELECT BOOK_NUM, BOOK_TITLE, BOOK_COST
FROM BOOK
WHERE BOOK_COST = (SELECT Min (BOOK_COST) FROM BOOK)
ORDER BY BOOK_NUM

106. Write a query to display the author ID, first and last name for all authors who have never written a book with the subject Programming. Sort the results by author last name (Figure P7.106).

SELECT AU_ID, AU_FNAME, AU_LNAME
FROM AUTHOR
WHERE AU_ID NOT IN (SELECT AU_ID FROM BOOK JOIN WRITES ON BOOK.BOOK_NUM = WRITES.BOOK_NUM WHERE BOOK_SUBJECT = 'Programming')

ORDER BY AU_LNAME

107. **Write a query to display the book number, title, subject, average cost of books within that subject, and the difference between each book's cost and the average cost of books in that subject. Sort the results by book title (Figure P7.107).**

```
SELECT BOOK_NUM, BOOK_TITLE, BOOK.BOOK_SUBJECT, Round(AVGCOST, 2) AS
"Average Subject Cost", BOOK_COST - Round(AVGCOST, 2) AS DIFFERENCE
FROM BOOK JOIN (SELECT BOOK_SUBJECT, Avg(BOOK_COST) AS AVGCOST
                FROM BOOK BOOK2
                GROUP BY BOOK_SUBJECT) AS SUBAVGS ON BOOK.BOOK_SUBJECT =
SUBAVGS.BOOK_SUBJECT
ORDER BY BOOK_TITLE
```

108. **Write a query to display the book number, title, subject, author last name, and the number of books written by that author. Limit the results to books in the Cloud subject. Sort the results by book title and then author last name (Figure P7.108).**

```
SELECT BOOK.BOOK_NUM, BOOK_TITLE, BOOK_SUBJECT, AU_LNAME, NUMBOOKS AS
"Num Books by Author"
FROM BOOK JOIN WRITES ON BOOK.BOOK_NUM = WRITES.BOOK_NUM JOIN
                (SELECT AUTHOR.AU_ID, AU_LNAME, Count(*) AS NUMBOOKS
                 FROM AUTHOR JOIN WRITES ON AUTHOR.AU_ID = WRITES.AU_ID
                GROUP BY AUTHOR.AU_ID, AU_LNAME) AS AUTHBOOKS ON
WRITES.AU_ID = AUTHBOOKS.AU_ID
WHERE BOOK_SUBJECT = 'Cloud'
ORDER BY BOOK_TITLE, AU_LNAME
```

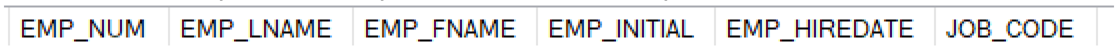
109. **Write a query to display the lowest average cost of books within a subject and the highest average cost of books within a subject (Figure P7.109).**

```
SELECT Min(AVGCOST) AS “Lowest Avg Cost”, Max(AVGCOST) AS “Highest Avg Cost”
FROM (SECLECT BOOK_SUBJECT, Round(Avg(BOOK_COST), 2) AS AVGCOST
FROM BOOK
GROUP BY BOOK_SUBJECT AS SUBAVGS
```

Part II:

1. **Write the SQL code that will create only the table structure for a table named EMP_1. This table will be a subset of the EMPLOYEE table. The basic EMP_1 table structure is summarized in the following table. Use EMP_NUM as the primary key. Note that the JOB_CODE is the FK to JOB so be certain to enforce referential integrity. Your code should also prevent null entries in EMP_LNAME and EMP_FNAME.**

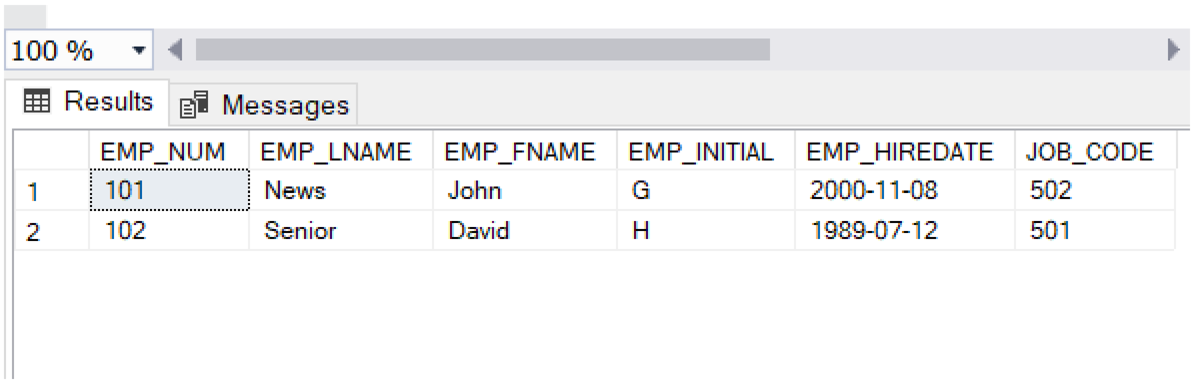
```
CREATE TABLE EMP_1 (
EMP_NUM          CHAR(3)          PRIMARY KEY,
EMP_LNAME        VARCHAR(15)      NOT NULL,
EMP_FNAME        VARCHAR(15)      NOT NULL,
EMP_INITIAL      CHAR(1),
EMP_HIREDATE     DATE,
JOB_CODE         CHAR(3),
FOREIGN KEY (JOB_CODE) REFERENCES JOB);
```



EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_INITIAL	EMP_HIREDATE	JOB_CODE

2. **Having created the table structure in Problem 1, write the SQL code to enter the first two rows for the table shown in Figure P8.2. Each row should be inserted individually, without using a subquery. Insert the rows in the order that they are listed in the figure.**

```
INSERT INTO EMP_1 VALUES (101, ‘News’, ‘John’, ‘G’, ’08-Nov-00’, 502);
INSERT INTO EMP_1 VALUES (102, ‘Senior’, ‘David’, ‘H’, ’12-Jul-89’, 501);
```



100 %

Results | Messages

	EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_INITIAL	EMP_HIREDATE	JOB_CODE
1	101	News	John	G	2000-11-08	502
2	102	Senior	David	H	1989-07-12	501

4. **Write the SQL code that will save the changes made to the EMP_1 table (if supported by your DBMS).**

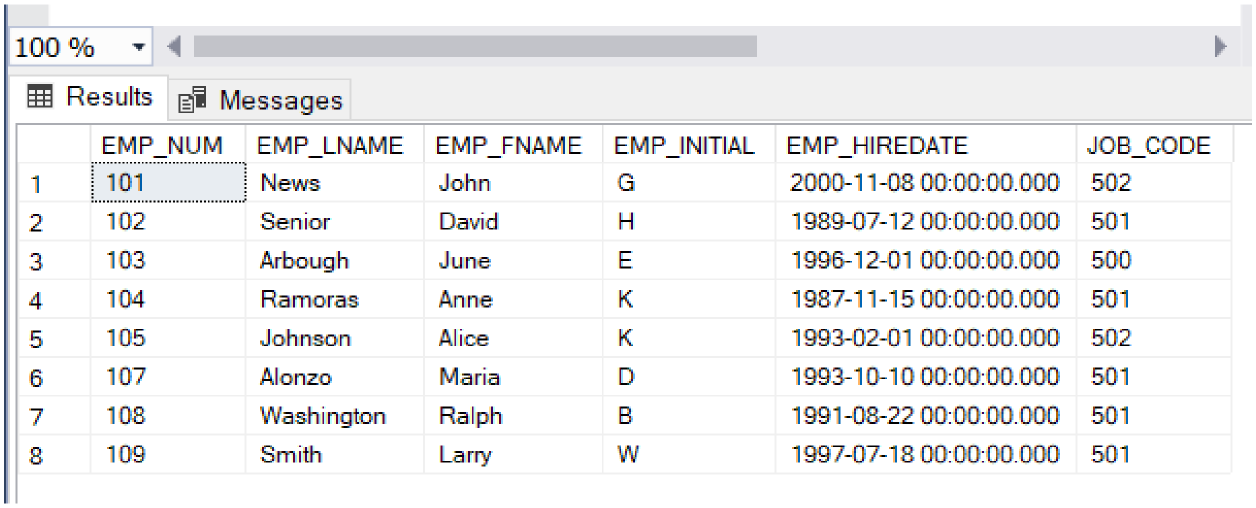
```
COMMIT;
```

5. **Write the SQL code to change the job code to 501 for the person whose employee number (EMP_NUM) is 107.**

```
UPDATE EMP_1
SET     JOB_CODE = ‘501’
WHERE          EMP_NUM = ‘107’;
```

6. **Write the SQL code to delete the row for William Smithfield, who was hired on June 22, 2004, and whose job code is 500. (Hint: Use logical operators to include all of the information given in this problem. Remember, if you are using MySQL, you will have to first disable “safe mode.”)**

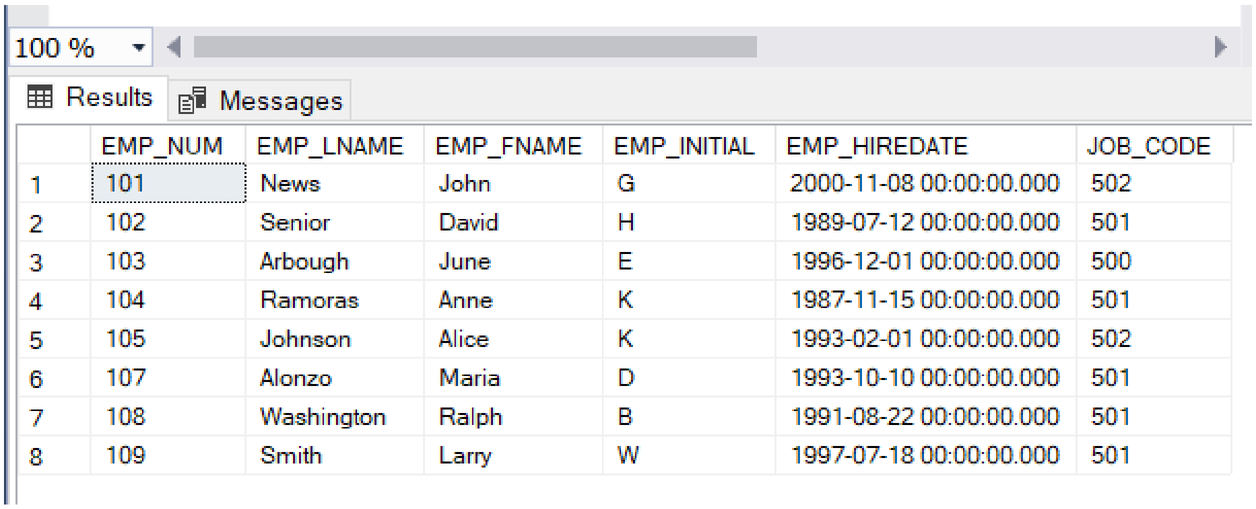
```
DELETE FROM EMP_1
WHERE          EMP_LNAME = ’Smithfield’,
AND            EMP_FNAME = ‘William’,
AND            EMP_HIREDATE = ‘22-Jun-04’,
AND            JOB_CODE = ‘500’;
```



	EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_INITIAL	EMP_HIREDATE	JOB_CODE
1	101	News	John	G	2000-11-08 00:00:00.000	502
2	102	Senior	David	H	1989-07-12 00:00:00.000	501
3	103	Arbough	June	E	1996-12-01 00:00:00.000	500
4	104	Ramoras	Anne	K	1987-11-15 00:00:00.000	501
5	105	Johnson	Alice	K	1993-02-01 00:00:00.000	502
6	107	Alonzo	Maria	D	1993-10-10 00:00:00.000	501
7	108	Washington	Ralph	B	1991-08-22 00:00:00.000	501
8	109	Smith	Larry	W	1997-07-18 00:00:00.000	501

7. **Write the SQL code to create a copy of EMP_1, including all of its data, and naming the copy EMP_2.**

```
CREATE TABLE EMP_2 AS SELECT * FROM EMP_1
SELECT * FROM EMP_2;
```

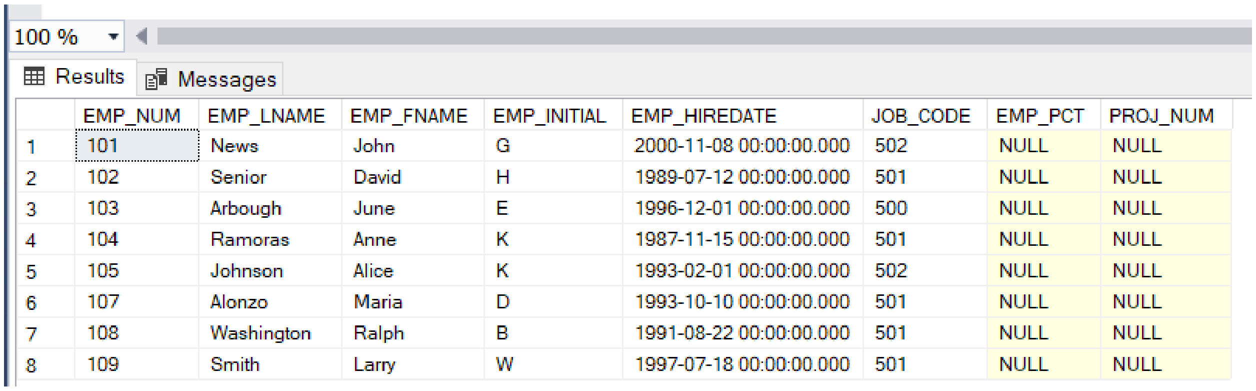

100 %

Results | Messages

	EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_INITIAL	EMP_HIREDATE	JOB_CODE
1	101	News	John	G	2000-11-08 00:00:00.000	502
2	102	Senior	David	H	1989-07-12 00:00:00.000	501
3	103	Arbough	June	E	1996-12-01 00:00:00.000	500
4	104	Ramoras	Anne	K	1987-11-15 00:00:00.000	501
5	105	Johnson	Alice	K	1993-02-01 00:00:00.000	502
6	107	Alonzo	Maria	D	1993-10-10 00:00:00.000	501
7	108	Washington	Ralph	B	1991-08-22 00:00:00.000	501
8	109	Smith	Larry	W	1997-07-18 00:00:00.000	501

8. **Using the EMP_2 table, write the SQL code that will add the attributes EMP_PCT and PROJ_NUM to EMP_2. The EMP_PCT is the bonus percentage to be paid to each employee. The new attribute characteristics are: EMP_PCT NUMBER(4,2) PROJ_NUM CHAR(3) Note: If your SQL implementation requires it, you may use DECIMAL(4,2) or NUMERIC(4,2) rather than NUMBER(4,2).**

```
ALTER TABLE EMP_2
ADD (EMP_PCT       NUMBER (4,2)),
ADD (PROJ_NUM      CHAR(3));
```

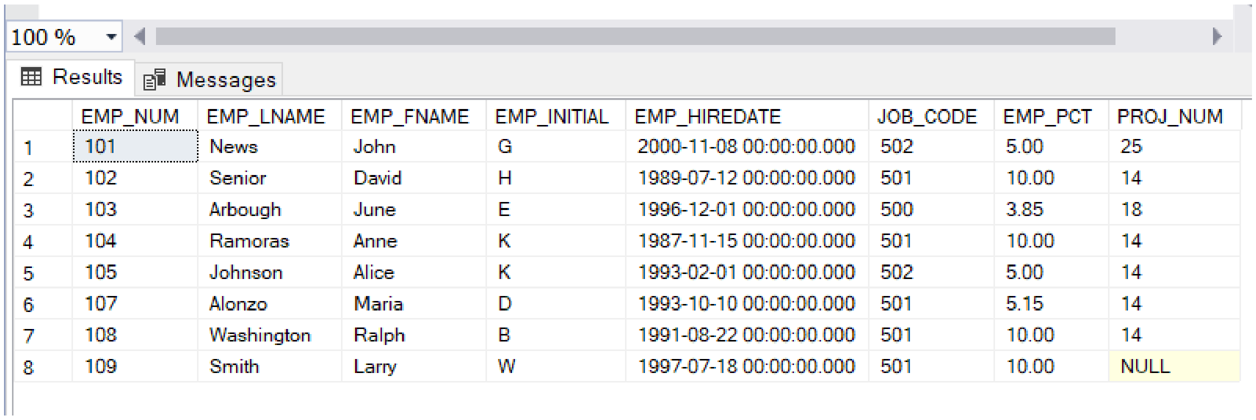

100 %

Results | Messages

	EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_INITIAL	EMP_HIREDATE	JOB_CODE	EMP_PCT	PROJ_NUM
1	101	News	John	G	2000-11-08 00:00:00.000	502	NULL	NULL
2	102	Senior	David	H	1989-07-12 00:00:00.000	501	NULL	NULL
3	103	Arbough	June	E	1996-12-01 00:00:00.000	500	NULL	NULL
4	104	Ramoras	Anne	K	1987-11-15 00:00:00.000	501	NULL	NULL
5	105	Johnson	Alice	K	1993-02-01 00:00:00.000	502	NULL	NULL
6	107	Alonzo	Maria	D	1993-10-10 00:00:00.000	501	NULL	NULL
7	108	Washington	Ralph	B	1991-08-22 00:00:00.000	501	NULL	NULL
8	109	Smith	Larry	W	1997-07-18 00:00:00.000	501	NULL	NULL

9. **Using the EMP_2 table, write the SQL code to change the EMP_PCT value to 3.85 for the person whose employee number (EMP_NUM) is 103.**

```
UPDATE      EMP_2
SET         EMP_PCT=3.85
WHERE       EMP_NUM= ‘103’;
```


100 %

Results Messages

	EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_INITIAL	EMP_HIREDATE	JOB_CODE	EMP_PCT	PROJ_NUM
1	101	News	John	G	2000-11-08 00:00:00.000	502	5.00	25
2	102	Senior	David	H	1989-07-12 00:00:00.000	501	10.00	14
3	103	Arbough	June	E	1996-12-01 00:00:00.000	500	3.85	18
4	104	Ramoras	Anne	K	1987-11-15 00:00:00.000	501	10.00	14
5	105	Johnson	Alice	K	1993-02-01 00:00:00.000	502	5.00	14
6	107	Alonzo	Maria	D	1993-10-10 00:00:00.000	501	5.15	14
7	108	Washington	Ralph	B	1991-08-22 00:00:00.000	501	10.00	14
8	109	Smith	Larry	W	1997-07-18 00:00:00.000	501	10.00	NULL