Prac #7

SQL Practice

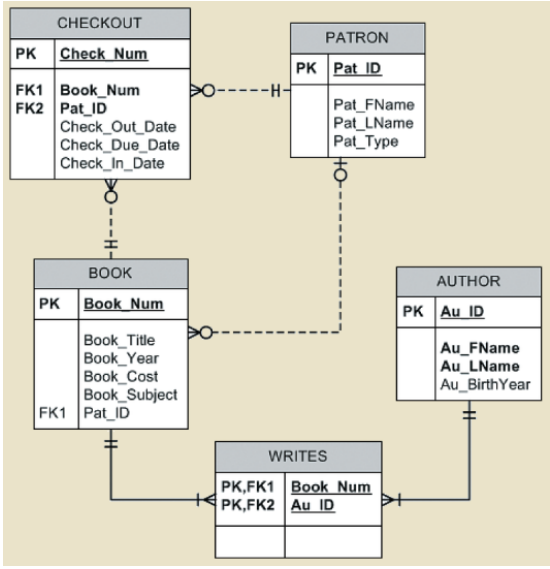
1. Using SQL Commands on MySQL Workbench for Creating a Database Model

2. Writing/Executing SQL queries in MySQL Workbench to extract information from a database

Through Prac #6, you developed a simple relational database for a small e-book library (as shown in the ERD provided) using MySQL Workbench.

While you created the database schema and tables and filled the table using import facility in Prac #6, you can also use SQL queries in to create tables and insert all necessary data to the table created.

In the first section of this prac (Task 1 and Task 2), you will learn how to use MySQL Workbench facilities to create SQL queries and run (execute) them mainly for the purpose of data definition (creating table structures) or basic data manipulation (inserting data records to the existing tables etc.).



In the later section of this prac (Task 3), you are going to practice basic SQL queries to extract useful information using MySQL Workbench. You will use the library database you created in previous section of this prac, and execute a number of SQL queries to extract information from the database.

- **Learning outcomes and objectives**
  Student will be able to:
  - use basic SQL for data definition (to create tables) and for data manipulation (to add, modify, delete, and retrieve data)
  - compose various SQL queries in particular using SELECT command by adding restrictions to the search criteria,
  - use special SQL commands to restrict or adjust the way of displaying output of the query

- **Pre-requisites**
  You are assumed basic knowledge about SQL commands to create a database structure and manipulate the contents of the data within it. This topic was covered in this week's lecture (Lecture 06). The following sections of the Coronel-Morris textbook, which explains SQL to create a table manually and to do data manipulation, are also required reading.
  - 12th Edition: Chapter 7.1 – 7.5
  - 13th Edition: Chapter 8.1 - 8.4

- **Task Overview**

  [Task 1] is to create a library database using SQL commands on MySQL Workbench. The required steps are summarised here:

  - Step 1: Create an initial (empty) database schema having no tables contained.
  - Step 2: Write and Run SQL queries (via SQL query editor provided by MySQL Workbench) to create each table structure
  - Step 3: Write and Run SQL queries to insert data to each table
  - Step 4: Apply reverse engineering process to create the ERD for the database model created (You can do this step after Step 2)

  [Task 2] is to apply various SQL commands to manipulate the structure of existing tables to add, modify, and remove columns and constraints. You will also need to apply SQL commands to do data manipulation like insert, update, and delete rows of data. For this practice, you will create a simple database using the SQL command scripts provided.

  [Task 3] is to apply basic SQL commands to compose queries to extract information from a given database.

Note

Solutions for [Task 1] and [Task 2] are already provided thus they do not require any submission for marking. You can use this only for your own learning and practice. Also, the database you created through [Task 1] will be directly used to complete [Task 3].

You are only required to submit all queries for [Task 3] for marking.

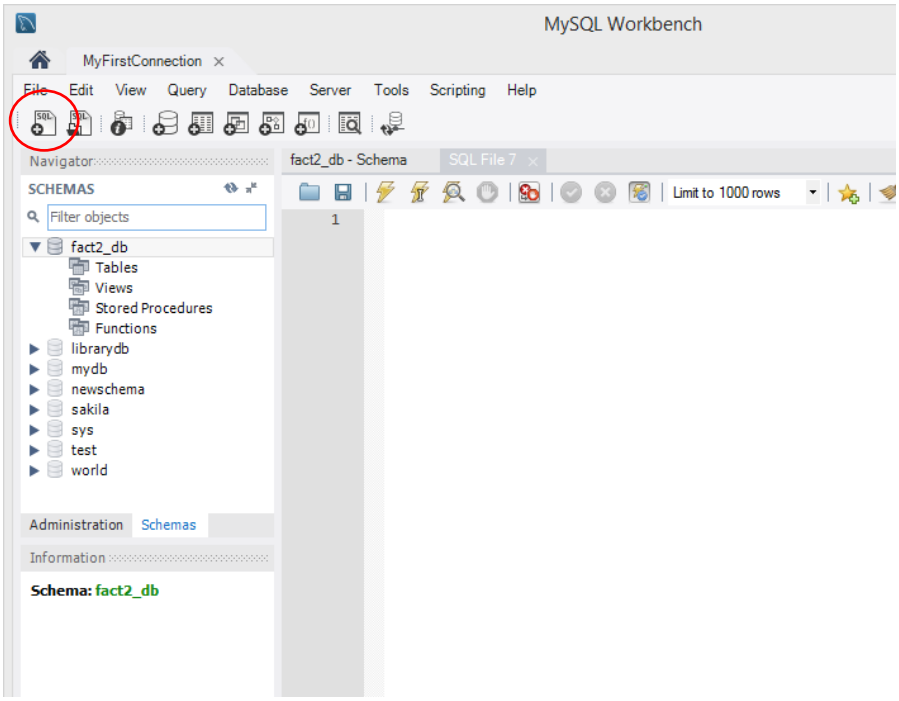
[Task 1]
Create a database using SQL commands on MySQL Workbench

We can use SQL to create database and table structures and to perform basic data management chores (insert, update, and delete). Follow the process as instructed below to create the library database using SQL commands.

1. Start the new connection on MySQL Workbench
2. On the SCHEMAS Navigator panel, put the mouse over the list of schemas and right-click on the mouse, then select “Create Schema ….” menu.
3. Rename the schema into your own new library database title (e.g. ‘fact2_db’), and click “Apply”. This will process to create a new (empty) database.
4. Check through SCHEMAS Navigator panel to find that the current list of schemas includes your new database now. (If needed, refresh the list by clicking the refresh button)
5. Double click on the new database (fact2_db) and you will find that the database contains currently no components like tables

Now, it is time to create table structures. Each table structure will be created using one SQL query each.

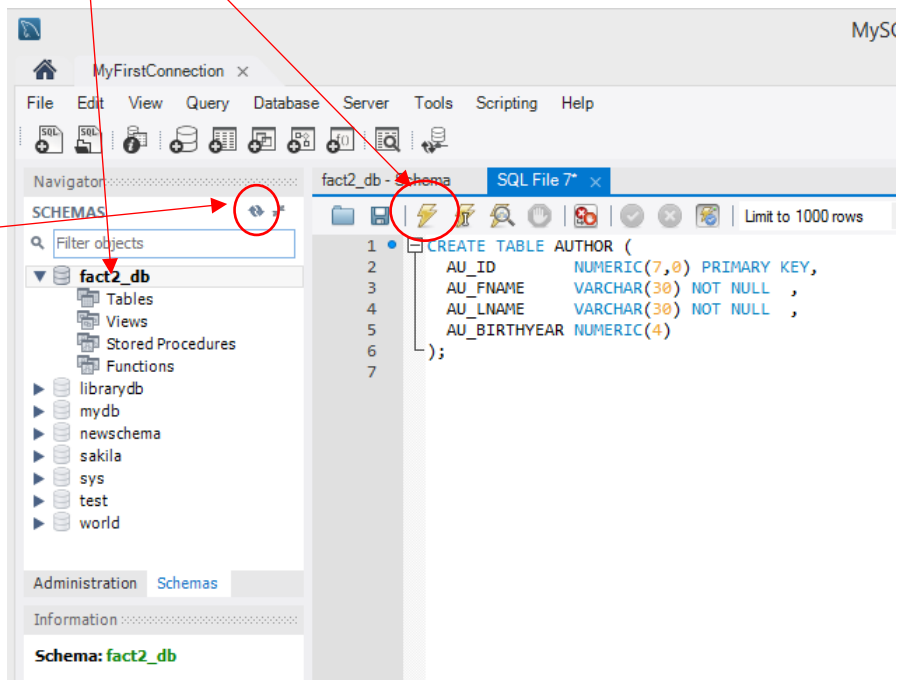
6. Open a new SQL query editor by clicking the SQL+ icon on the overhead menu bar.



7. Write the following SQL query to create the author table. After writing the query, click '**execute**' button to execute the SQL code. (Note: you need to make sure to select the database (fact2_db) by **double-clicking** on the database name before executing the query)

```
CREATE TABLE AUTHOR (
 AU_ID       NUMERIC(7,0) PRIMARY KEY,
 AU_FNAME    VARCHAR(30) NOT NULL ,
 AU_LNAME    VARCHAR(30) NOT NULL ,
 AU_BIRTHYEAR NUMERIC(4)
);
```

After running this query, you will find (through the navigator panel) that the fact2_db contains one new table called author. (**refresh** the list if needed)



(You can close the current query editor for your convenience, but it is optional to save the individual query or not. At this stage, you do not need to save, but you will need to save the query if needed for other practical task)

8. Create the second table patron by running the following SQL query.

```
CREATE TABLE PATRON (
 PAT_ID     NUMERIC(10,0) PRIMARY KEY,
 PAT_FNAME   VARCHAR(20) NOT NULL   ,
 PAT_LNAME   VARCHAR(20) NOT NULL   ,
 PAT_TYPE    VARCHAR(10) NOT NULL
);
```

9. You can apply the same way to create the other three tables by running each SQL query. However, you run a series of queries by running once. For this, you need to add BEGIN; command at the start and add COMMIT; at the end. The whole code to create three tables (book, patron, writes) at one time is shown below.

```
BEGIN;

CREATE TABLE PATRON (
 PAT_ID     NUMERIC(10,0) PRIMARY KEY,
 PAT_FNAME   VARCHAR(20) NOT NULL   ,
 PAT_LNAME   VARCHAR(20) NOT NULL   ,
 PAT_TYPE    VARCHAR(10) NOT NULL
);

CREATE TABLE BOOK (
 BOOK_NUM   NUMERIC(10,0) PRIMARY KEY,
 BOOK_TITLE VARCHAR(120) NOT NULL  ,
 BOOK_YEAR  NUMERIC(4)            ,
 BOOK_COST  NUMERIC(8,2)          ,
 BOOK_SUBJECT VARCHAR(120)        ,
 PAT_ID    NUMERIC(10),
 FOREIGN KEY(PAT_ID) REFERENCES PATRON(PAT_ID)
);

CREATE TABLE CHECKOUT (
 CHECK_NUM        NUMERIC(15) PRIMARY KEY,
 BOOK_NUM         NUMERIC(10),
 PAT_ID          NUMERIC(10),
 CHECK_OUT_DATE    DATE,
 CHECK_DUE_DATE    DATE,
 CHECK_IN_DATE     DATE,
 FOREIGN KEY (BOOK_NUM) REFERENCES BOOK(BOOK_NUM),
 FOREIGN KEY (PAT_ID) REFERENCES PATRON(PAT_ID)
);

CREATE TABLE WRITES (
 BOOK_NUM  NUMERIC(10),
 AU_ID    NUMERIC(7),
 CONSTRAINT WRITES_BOOK_AU_PK PRIMARY KEY (BOOK_NUM, AU_ID),
 CONSTRAINT WRITES_BOOK_NUM_FK FOREIGN KEY(BOOK_NUM) REFERENCES
BOOK(BOOK_NUM),
 CONSTRAINT WRITES_AU_ID_FK FOREIGN KEY(AU_ID) REFERENCES AUTHOR(AU_ID)
);

COMMIT;
```

Now your database (fact2_db) currently has table structures with no actual record (data). At this stage, you may import table data directly from external files as learned in the previous prac (Prac#6). However, for this task, let's try to use SQL code to insert each data into the existing table one by one (or by series of SQL codes as a whole).

10. Create a new query to insert the first record of author table.

    INSERT INTO AUTHOR VALUES (185, 'Benson', 'Reeves', 1990);

11. To insert all other records, you need to create and run multiple number of SQL queries. For your convenience, MySQL codes for creating this database are provided. Please download the file 'fact_mysql.txt' and use the code appropriately for your usage. (Hint: you can run the multiple queries at one time using 'begin;' and 'commit;' command)

    (Please note that the first AUTHOR row was already inserted, thus you will have to remove the first INSERT command when you run the SQL codes provided (fact_mysql.txt))

12. Fully check (through SCHEMAS navigator) that all tables and data records are correctly imported.

13. Save the database as a self-contained file (fact2_dump.sql) so that you can open the database anywhere when needed in the future. (Refer to previous pracs (Prac #5 or Prac #6) to learn how to back up your database).

[Task 2]

Manipulate an existing database using SQL commands on MySQL Workbench

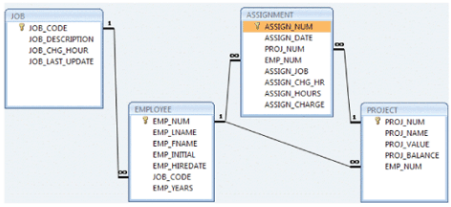
Source: Coronel-Morris textbook (13th edition) **Chapter 8 Problems Q1 ~ Q15**

1. Create a database named 'ConstructCo_db' on MySQL Workbench. Use the SQL command scripts provided (ConstructCo_MySQL.txt).

2. Apply 'Reverse Engineering' process to create an ERD for this database. Save the ERD as a file named 'ConstructCo.mwb'.

If you created the database successfully, you will see that the structure and contents of the ConstructCo database are as shown in the following figure.

Relational diagram



Database name: Ch08_ConstructCo

Table name: EMPLOYEE

EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_INITIAL	EMP_HIREDATE	JOB_CODE	EMP_YEARS
101	News	John	G	08-Nov-00	502	12
102	Senior	David	H	12-Jul-89	501	23
103	Arbough	June	E	01-Dec-96	500	18
104	Ramoras	Anne	K	15-Nov-87	501	25
105	Johnson	Alice	K	01-Feb-93	502	19
106	Smithfield	William		22-Jun-04	500	8
107	Alonzo	Maria	D	10-Oct-93	500	19
108	Washington	Ralph	B	22-Aug-91	501	21
109	Smith	Larry	W	18-Jul-97	501	15
110	Olenko	Gerald	A	11-Dec-95	505	17
111	Wabash	Geoff	B	04-Apr-91	506	21
112	Smithson	Darlene	M	23-Oct-94	507	18
113	Joenbrood	Delbert	K	15-Nov-96	508	16
114	Jones	Annelise		20-Aug-93	508	19
115	Bawangi	Travis	B	25-Jan-92	501	20
116	Pratt	Gerald	L	05-Mar-97	510	15
117	Williamson	Angie	H	19-Jun-96	509	16
118	Frommer	James	J	04-Jan-05	510	7

Table name: JOB

JOB_CODE	JOB_DESCRIPTION	JOB_CHG_HOUR	JOB_LAST_UPDATE
500	Programmer	35.75	20-Nov-17
501	Systems Analyst	96.75	20-Nov-17
502	Database Designer	125.00	24-Mar-18
503	Electrical Engineer	84.50	20-Nov-17
504	Mechanical Engineer	67.90	20-Nov-17
505	Civil Engineer	55.78	20-Nov-17
506	Clerical Support	26.87	20-Nov-17
507	DSS Analyst	45.95	20-Nov-17
508	Applications Designer	48.10	24-Mar-18
509	Bio Technician	34.55	20-Nov-17
510	General Support	18.36	20-Nov-17

Table name: ASSIGNMENT

ASSIGN_NUM	ASSIGN_DATE	PROJ_NUM	EMP_NUM	ASSIGN_JOB	ASSIGN_CHG_HR	ASSIGN_HOURS	ASSIGN_CHARGE
1001	22-Mar-18	18	103	503	84.50	3.5	295.75
1002	22-Mar-18	22	117	509	34.55	4.2	145.11
1003	22-Mar-18	18	117	509	34.55	2.0	69.10
1004	22-Mar-18	18	103	503	84.50	5.9	498.55
1005	22-Mar-18	25	108	501	96.75	2.2	212.85
1006	22-Mar-18	22	104	501	96.75	4.2	406.35
1007	22-Mar-18	25	113	508	50.75	3.8	192.85
1008	22-Mar-18	18	103	503	84.50	0.9	76.05
1009	23-Mar-18	15	115	501	96.75	5.6	541.80
1010	23-Mar-18	15	117	509	34.55	2.4	82.92
1011	23-Mar-18	25	105	502	105.00	4.3	451.50
1012	23-Mar-18	18	108	501	96.75	3.4	328.95
1013	23-Mar-18	25	115	501	96.75	2.0	193.50
1014	23-Mar-18	22	104	501	96.75	2.8	270.90
1015	23-Mar-18	15	103	503	84.50	6.1	515.45
1016	23-Mar-18	22	105	502	105.00	4.7	493.50
1017	23-Mar-18	18	117	509	34.55	3.8	131.29
1018	23-Mar-18	25	117	509	34.55	2.2	76.01
1019	24-Mar-18	25	104	501	110.50	4.9	541.45
1020	24-Mar-18	15	101	502	125.00	3.1	387.50
1021	24-Mar-18	22	108	501	110.50	2.7	298.35
1022	24-Mar-18	22	115	501	110.50	4.9	541.45
1023	24-Mar-18	22	105	502	125.00	3.5	437.50
1024	24-Mar-18	15	103	503	84.50	3.3	278.85
1025	24-Mar-18	18	117	509	34.55	4.2	145.11

Table name: PROJECT

PROJ_NUM	PROJ_NAME	PROJ_VALUE	PROJ_BALANCE	EMP_NUM
15	Evergreen	1453500.00	1002350.00	103
18	Amber Wave	3500500.00	2110346.00	108
22	Rolling Tide	805000.00	500345.20	102
25	Starflight	2650500.00	2309880.00	107

The **ConstructCo** database stores data for a consulting company that tracks all charges to projects. The charges are based on the hours each employee works on each project.

Note that the ASSIGNMENT table stores the JOB_CHG_HOUR values as an attribute (ASSIGN_CHG_HR) to maintain historical accuracy of the data. The JOB_CHG_HOUR values are likely to change over time. In fact, a JOB_CHG_HOUR change will be reflected in the ASSIGNMENT table. And, naturally, the employee primary job assignment might change, so the ASSIGN_JOB is also stored. Because those attributes are required to maintain the historical accuracy of the data, they are *not* redundant.

Given the structure and contents of the ConstructCo database, write and run SQL commands for the following problems (1)~(15).

Note that this task is not to be submitted. Solutions are already provided for your practice.

(1) Write the SQL code that will create the table structure for a table named EMP_1. This table is a subset of the EMPLOYEE table. The basic EMP_1 table structure is summarized in the table below. 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        VARCHAR(3)  PRIMARY KEY,
EMP_LNAME      VARCHAR(15) NOT NULL,
EMP_FNAME      VARCHAR(15) NOT NULL,
EMP_INITIAL    VARCHAR(1),
EMP_HIREDATE   DATE,
JOB_CODE       VARCHAR(3),
FOREIGN KEY (JOB_CODE) REFERENCES JOB(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 as below. Each row should be inserted individually, without using a subquery. Insert the rows in the order that they are listed in the table as shown.

[The contents of the EMP_1 table]

EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_INITIAL	EMP_HIREDATE	JOB_CODE
101	News	John	G	08-Nov-00	502
102	Senior	David	H	12-Jul-89	501
103	Arbough	June	E	01-Dec-96	500
104	Ramoras	Anne	K	15-Nov-87	501
105	Johnson	Alice	K	01-Feb-93	502
106	Smithfield	William		22-Jun-04	500
107	Alonzo	Maria	D	10-Oct-93	500
108	Washington	Ralph	B	22-Aug-91	501
109	Smith	Larry	W	18-Jul-97	501

```
INSERT INTO EMP_1 VALUES ('101', 'News', 'John', 'G', '2000-11-08', '502');
INSERT INTO EMP_1 VALUES ('102', 'Senior', 'David', 'H', '1989-07-12', '501');
```

(3) Using the EMPLOYEE table that already exists, use a subquery to insert the remaining rows from the EMPLOYEE table into the EMP_1 table. Remember, your subquery should only retrieve the columns needed for the EMP_1 table and only the employees shown in the figure.

```
INSERT INTO EMP_1 SELECT EMP_NUM, EMP_LNAME, EMP_FNAME, EMP_INITIAL,
EMP_HIREDATE, JOB_CODE FROM EMPLOYEE WHERE EMP_NUM BETWEEN 103 AND 109;
```

(4) Write the SQL code that will save the changes made to the EMP_1 table.

```
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;
```

Note:

Once you run this SQL code, you may get one error message from Workbench regarding 'safe update mode'. You can set up this under Edit>Preferences>SQL Editor>unclick "Safe Updates". Please be cautious that this setting requires restart of your server and reconnection thus you will need to save your database firstly in particular if you use Workbench in a lab computer.

(6) Write the SQL code to delete the row for the person named William Smithfield, who was hired on June 22, 2004, and whose job code classification is 500. (*Hint*: Use logical operators to include all of the information given in this problem.

```
DELETE FROM EMP_1
WHERE EMP_LNAME = 'Smithfield'
AND EMP_FNAME = 'William'
AND EMP_HIREDATE = '2004-06-22'
AND JOB_CODE = 500;
```

(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;
```

(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_PCTNUMBER(4,2)
PROJ_NUMCHAR(3)

(*Note*: If your SQL implementation allows it, you may use DECIMAL(4,2) or NUMERIC(4,2) rather than NUMBER(4,2). Use DECIMAL(4,2) for MySQL Workbench.)

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

(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;
```

(10) Using the EMP_2 table, write a single SQL command to change the EMP_PCT values to 5.00 for the people with employee numbers 101, 105, and 107.

```
UPDATE EMP_2
SET EMP_PCT = 5
WHERE EMP_NUM IN (101, 105, 107);
```

(11) Using the EMP_2 table, write a single SQL command to change the EMP_PCT values to 10.00 for all employees who do not currently have a value for EMP_PCT.

```
UPDATE EMP_2
SET EMP_PCT = 10
WHERE EMP_PCT IS NULL;
```

(12) Using the EMP_2 table, write a single SQL command to add 0.15 to the EMP_PCT of the employee whose name is Maria D. Alonzo. (Use the employee name in your command to determine the correct employee.)

```
UPDATE EMP_2
SET EMP_PCT = EMP_PCT + .15
WHERE EMP_FNAME = 'Maria'
AND EMP_LNAME = 'Alonzo'
AND EMP_INITIAL = 'D';
```

(13) Using a single command sequence with the EMP_2 table, write the SQL code that will change the project number (PROJ_NUM) to 18 for all employees whose job classification (JOB_CODE) is 500.

```
UPDATE EMP_2
SET PROJ_NUM = 18
WHERE JOB_CODE = 500;
```

(14) Using a single command sequence with the EMP_2 table, write the SQL code that will change the project number (PROJ_NUM) to 25 for all employees whose job classification (JOB_CODE) is 502 or higher.

```
UPDATE EMP_2
SET PROJ_NUM = 25
WHERE JOB_CODE >= 502;
```

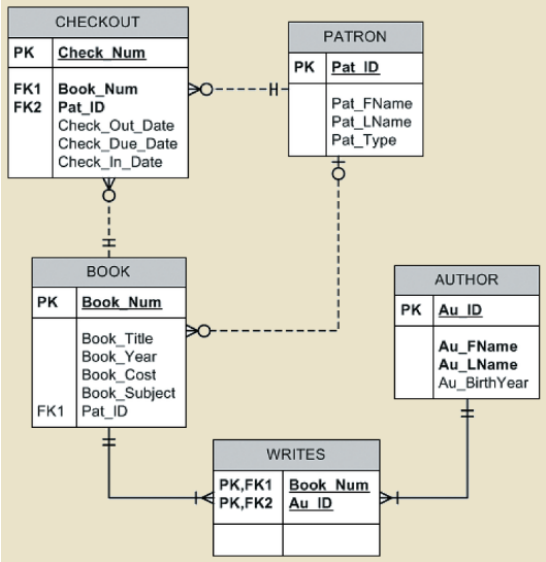
(15) Write the SQL code that will change the PROJ_NUM to 14 for employees who were hired before January 1, 1994, and whose job code is at least 501.

```
UPDATE EMP_2
SET PROJ_NUM = 14
WHERE EMP_HIREDATE < '1994-01-01' AND JOB_CODE >= 501;
```

[Task 3]

Extract information from the database using SQL commands on MySQL Workbench

Open the library database you created in [Task 1] of this prac on MySQL Workbench. The conceptual model of the database is presented in the ERD as shown here.



You will have to refer to this ERD when you compose SQL queries for this task, to understand further details about this database including table structure, columns included in each table, PKs and FKs, relationships between tables etc.

You are given a number of exercises to practice to write/run SQL queries. Solutions for some exercises are already provided. For each of these exercises, you will need to write and save an SQL query (though some exercises already show solutions, you are always recommended to write and run the code yourself).

Do not forget to save each query (or copy/paste the query to a WORD document) as soon as after writing/running it. For example, save the query as 'Q1.sql' for the first question and keep it for your own record. For the submission of this prac, you are required to submit one WORD or Text file containing all query texts you completed as Q1~Q25 to be marked off for this prac activity.

Exercises:

For each of these exercises, a figure of the correct output is provided. If the output of the query is very large, only the first several rows of the output are shown. The head section of each query result must look exactly same as the output provided.

For your own back-up purpose, save a query for each question as the name of "Q1.sql", "Q2.sql", … To submit this prac, you are required to copy-and-paste all queries to one WORD or Text document and submit it to be marked off. Please notate clearly each query number (e.g. Q1, Q2, …).

Note that some questions are provided with the accompanied solution to help your learning.

Note:
As a final submission of this practical activity, you are required to submit one MS WORD document showing all SQL queries (in text) you completed for [Task 3] exercises as Q1~Q25 to be marked off for Prac 7.

1. Write a query that displays the first and last name of every patron. (See the figure below for first part of the output. The actual result will have 50 rows)

PAT_FNAME	PAT_LNAME
Vera	Alvarado
Holly	Anthony
Cedric	Baldwin
Cory	Barry
Nadine	Blair
Erika	Bowen
Gerald	Burke
Ollie	Cantrell
robert	carter
Keith	Cooley

Answer provided:

```
SELECT PAT_FNAME, PAT_LNAME
FROM PATRON
ORDER BY PAT_LNAME;
```

Note:
The 'ORDER BY' keyword ensures to sort the result records in ascending order (by default) to one or more columns specified.

2. Write a query to display the checkout number, check out date, and due date for every book that has been checked out sorted by checkout number. (See the figure below for first part of the output. The actual result will have 68 rows)

CHECK_NUM	CHECK_OUT_DATE	CHECK_DUE_DATE
91001	3/31/2017	4/14/2017
91002	3/31/2017	4/7/2017
91003	3/31/2017	4/14/2017
91004	3/31/2017	4/14/2017
91005	3/31/2017	4/7/2017
91006	4/5/2017	4/12/2017
91007	4/5/2017	4/12/2017
91008	4/5/2017	4/12/2017
91009	4/5/2017	4/19/2017
91010	4/5/2017	4/19/2017
91011	4/5/2017	4/12/2017

Note:
As you use MySQL Workbench to run this query, the output of date values (CHECK_DUE_DATE) will be presented in a form of 'yyyy-mm-dd'. Simply ignore the difference of your output and the example output provided here.

3. Write a query to display the book number, book title, and subject for every book sorted by book number (See the figure below for the output. The actual result will have 20 rows)

BOOK_NUM	TITLE	Subject of Book
5235	Beginner's Guide to JAVA	Programming
5236	Database in the Cloud	Cloud
5237	Mastering the database environment	Database
5238	Conceptual Programming	Programming
5239	J++ in Mobile Apps	Programming
5240	iOS Programming	Programming
5241	JAVA First Steps	Programming
5242	C# in Middleware Deployment	Middleware
5243	DATABASES in Theory	Database
5244	Cloud-based Mobile Applications	Cloud
5245	The Golden Road to Platform independence	Middleware

Answer provided:

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

Note above the use of backticks (`) for a column name that contains a space. A backtick is created by pressing the tilde (~) key **without** shift.

4. Write a query to display the book number, title, and cost of each book sorted by book number (See the figure below for the output. The actual result will have 20 rows).

BOOK_NUM	BOOK_TITLE	Replacement Cost
5235	Beginner's Guide to JAVA	59.95
5236	Database in the Cloud	79.95
5237	Mastering the database environment	89.95
5238	Conceptual Programming	59.95
5239	J++ in Mobile Apps	49.95
5240	iOS Programming	79.95
5241	JAVA First Steps	49.95
5242	C# in Middleware Deployment	59.95
5243	DATABASES in Theory	129.95
5244	Cloud-based Mobile Applications	69.95
5245	The Golden Road to Platform independence	119.95
5246	Capture the Cloud	69.95
5247	Shining Through the Cloud: Sun Programming	109.95
5248	What You Always Wanted to Know About Database, But Were Afraid to Ask	49.95

5. Write a query to display the different years in which books have been published in. Include each year only once and sort the results by year (See the figure below for the output).

BOOK_YEAR
2014
2015
2016
2017

Answer provided:

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

6. Write a query to display the different subjects on which this library has books. Include each subject only once and sort the results by subject (See the figure below for the output).

BOOK_SUBJECT
Cloud
Database
Middleware
Programming

7. 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 (See the figure below for first part of the output. The actual result will have 68 rows)

CHECK_NUM	BOOK_NUM	PAT_ID	CHECK_OUT_DATE	CHECK_DUE_DATE
91067	5252	1229	5/24/2017	5/31/2017
91068	5238	1229	5/24/2017	5/31/2017
91066	5242	1228	5/19/2017	5/26/2017
91064	5236	1183	5/17/2017	5/31/2017
91065	5244	1210	5/17/2017	5/24/2017
91060	5235	1209	5/15/2017	5/22/2017
91061	5246	1172	5/15/2017	5/22/2017
91062	5254	1223	5/15/2017	5/22/2017
91063	5243	1223	5/15/2017	5/22/2017
91056	5254	1224	5/10/2017	5/17/2017

Answer provided:

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

8. 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 (See the figure below for the output. The actual result will have 20 rows).

BOOK_TITLE	BOOK_YEAR	BOOK_SUBJECT
Capture the Cloud	2016	Cloud
Starlight Applications	2016	Cloud
Cloud-based Mobile Applications	2015	Cloud
Database in the Cloud	2014	Cloud
Beyond the Database Veil	2016	Database
What You Always Wanted to Know About Database, But Were Afraid to Ask	2016	Database
DATABASES in Theory	2015	Database
Mastering the database environment	2015	Database
Reengineering the Middle Tier	2016	Middleware
The Golden Road to Platform independence	2016	Middleware

9. Write a query to display the book number, title, and cost for all books that cost $59.95 sorted by book number (See the figure for the output).

BOOK_NUM	BOOK_TITLE	BOOK_COST
5235	Beginner's Guide to JAVA	59.95
5238	Conceptual Programming	59.95
5242	C# in Middleware Deployment	59.95
5251	Thoughts on Revitalizing Ruby	59.95

Answer provided:

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

10. Write a query to display the book number, title, and replacement cost for all books in the "Database" subject sorted by book number (See the figure below for the output).

BOOK_NUM	BOOK_TITLE	BOOK_COST
5237	Mastering the database environment	89.95
5243	DATABASES in Theory	129.95
5248	What You Always Wanted to Know About Database, But Were Afraid to Ask	49.95
5252	Beyond the Database Veil	69.95

11. 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 (See the figure for the output).

CHECK_NUM	BOOK_NUM	CHECK_OUT_DATE
91001	5235	3/31/2017
91002	5238	3/31/2017
91003	5240	3/31/2017
91004	5237	3/31/2017
91005	5236	3/31/2017

Answer provided:

```
SELECT CHECK_NUM, BOOK_NUM, CHECK_OUT_DATE
FROM checkout
WHERE CHECK_OUT_DATE < '2017-04-05'
ORDER BY CHECK_NUM;
```

12. 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 (See the figure below for the output).

BOOK_NUM	BOOK_TITLE	BOOK_YEAR
5247	Shining Through the Cloud: Sun Programming	2016
5251	Thoughts on Revitalizing Ruby	2016
5253	Virtual Programming for Virtual Environments	2016
5254	Coding Style for Maintenance	2017

13. 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 (See the figure below for the output).

BOOK_NUM	BOOK_TITLE	BOOK_SUBJECT	BOOK_COST
5236	Database in the Cloud	Cloud	79.95
5245	The Golden Road to Platform independence	Middleware	119.95
5250	Reengineering the Middle Tier	Middleware	89.95

Answer provided:

```
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;
```

14. 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 (See the figure below for the output).

AU_ID	AU_FNAME	AU_LNAME	AU_BIRTHYEAR
218	Rachel	Beatney	1983
383	Neal	Walsh	1980
394	Robert	Lake	1982
438	Perry	Pearson	1986
460	Connie	Paulsen	1983
581	Manish	Aggerwal	1984
603	Julia	Palca	1988

15. 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 (See the figure below for the output).

BOOK_NUM	BOOK_TITLE	BOOK_SUBJECT
5236	Database in the Cloud	Cloud
5237	Mastering the database environment	Database
5243	DATABASES in Theory	Database
5248	What You Always Wanted to Know About Database, But Were Afraid to Ask	Database
5252	Beyond the Database Veil	Database

Answer provided:

```
SELECT BOOK_NUM, BOOK_TITLE, BOOK_SUBJECT
FROM BOOK
WHERE Upper(BOOK_TITLE) LIKE '%DATABASE%'
ORDER BY BOOK_NUM;
```

Note:
Unlike many other DBMS, MySQL ignores case-sensitive in 'Like' operation, thus actually it does not require to use 'Upper' function under MySQL environment.

16. Write a query to display the patron ID, first and last name of all patrons who are students, sorted by patron ID (See the figure below for first part of the output. The actual result will have 44 rows)

PAT_ID	PAT_FNAME	PAT_LNAME
1166	Vera	Alvarado
1171	Peggy	Marsh
1172	Tony	Miles
1174	Betsy	Malone
1180	Nadine	Blair
1181	Allen	Horne
1182	Jamal	Melendez
1184	Jimmie	Love
1185	Sandra	Yang
1200	Lorenzo	Torres

17. 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 (See the figure below for the output).

PAT_ID	PAT_FNAME	PAT_LNAME	PAT_TYPE
1160	robert	carter	Faculty
1208	Ollie	Cantrell	Student
1210	Keith	Cooley	STUdent

18. 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 (See the figure below for the output).

AU_ID	AU_FNAME	AU_LNAME
229	Carmine	Salvadore
262	Xia	Chiang
559	Rachel	McGill

Answer provided:

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

19. 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 (See the figure below for the output).

AU_ID	AU_FNAME	AU_LNAME
185	Benson	Reeves
218	Rachel	Beatney
251	Hugo	Bruer
273	Reba	Durante
284	Trina	Tankersly
383	Neal	Walsh
394	Robert	Lake
438	Perry	Pearson
460	Connie	Paulsen
581	Manish	Aggerwal
592	Lawrence	Sheel
603	Julia	Palca

20. Write a query to display the checkout number, book number, patron ID, check out date, and due date for all checkouts that have not yet been returned. Sort the results by book number (See the figure below for the output).

CHECK_NUM	BOOK_NUM	PAT_ID	CHECK_OUT_DATE	CHECK_DUE_DATE
91068	5238	1229	5/24/2017	5/31/2017
91053	5240	1212	5/9/2017	5/16/2017
91066	5242	1228	5/19/2017	5/26/2017
91061	5246	1172	5/15/2017	5/22/2017
91059	5249	1207	5/10/2017	5/17/2017
91067	5252	1229	5/24/2017	5/31/2017

21. 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 (See the figure below for the output). (Note that some DBMS sort NULLs as being large and some DBMS sort NULLs as being small.)

AU_ID	AU_FNAME	AU_LNAME	AU_BIRTHYEAR
185	Benson	Reeves	1990
603	Julia	Palca	1988
438	Perry	Pearson	1986
581	Manish	Aggerwal	1984
218	Rachel	Beatney	1983
460	Connie	Paulsen	1983
394	Robert	Lake	1982
383	Neal	Walsh	1980
592	Lawrence	Sheel	1976
251	Hugo	Bruer	1972
273	Reba	Durante	1969
284	Trina	Tankersly	1961
262	Xia	Chiang	
559	Rachel	McGill	
229	Carmine	Salvadore	

22. 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. (See the figure below for the output. The actual result will have 68 rows)

PATRON	BOOK	Days Kept
1160	5240	9
1160	5240	9
1165	5235	9
1183	5236	8
1184	5240	8
1185	5240	8
1202	5236	8
1203	5235	8
1204	5236	8
1207	5242	8
1209	5235	8
1219	5248	8
1222	5240	8
1226	5244	8
1165	5252	7
1185	5254	7

Answer provided:

```
SELECT PAT_ID AS PATRON, BOOK_NUM AS
BOOK, datediff(CHECK_IN_DATE,
CHECK_OUT_DATE) AS 'Days Kept'
FROM CHECKOUT
ORDER BY datediff(CHECK_IN_DATE, CHECK_OUT_DATE) DESC, PAT_ID,
BOOK_NUM;
```

23. Write a query to display the patron ID, patron full name, and patron type for each patron, sorted by patron ID (See the figure below for the output. The actual result will have 50 rows)

PAT_ID	Patron Name	PAT_TYPE
1160	robert carter	Faculty
1161	Kelsey Koch	Faculty
1165	Cedric Baldwin	Faculty
1166	Vera Alvarado	Student
1167	Alan Martin	FACULTY
1170	Cory Barry	faculty
1171	Peggy Marsh	STUDENT

Answer provided:

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

Note: CONCAT is a special function compatible with MySQL which can be used to concatenate multiple columns data into one. If you use any other DBMS, you need to refer to the manual of the DBMS for further information about the specific functions they provide for concatenating two columns data into one.

24. Write a query to display the book number, title with year, and subject for each book. Sort the results by the book number (See the figure below for the output. The actual result will have 20 rows)

BOOK_NUM	BOOK	BOOK_SUBJECT
5235	Beginner's Guide to JAVA (2014)	Programming
5236	Database in the Cloud (2014)	Cloud
5237	Mastering the database environment (2015)	Database
5238	Conceptual Programming (2015)	Programming
5239	J++ in Mobile Apps (2015)	Programming
5240	iOS Programming (2015)	Programming
5241	JAVA First Steps (2015)	Programming
5242	C# in Middleware Deployment (2015)	Middleware
5243	DATABASES in Theory (2015)	Database

25. 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. (See the figure below for the output. The actual result will have 50 rows)

PAT_ID	NAME	PAT_TYPE
1165	Cedric Baldwin	Faculty
1170	Cory Barry	faculty
1160	robert carter	Faculty
1183	Helena Hughes	Faculty
1161	Kelsey Koch	Faculty
1167	Alan Martin	FACULTY
1166	Vera Alvarado	Student
1202	Holly Anthony	Student
1180	Nadine Blair	STUDENT

This is the end of Prac 7.
You are required to submit via LearnJCU one WORD document containing all SQL queries you composed for [Task 3] Exercises.

Marking Criteria:

You will be given 0 to 3 marks depending on your completeness/correctness of your work.

- 0 or 0.5 – not attempted or mostly wrong or not reasonable logic used in most queries
- 1, 1.5 or 2 – attempted with good efforts but not fully correct. 40%~70% queries are fully correct
- 2.5 or 3 – fully correct and reasonable or very minor errors caused by simple mistake, missing, unnecessary complication etc. 80%~100% queries are fully correct

Please note that the individual feedback for each answer you submit will not be provided but once all markings are done the sample solution will be provided so that you can refer to them for your self-review.