Name: Koki Yamanaka
Student ID : T00681865
course title : COMP_3610
DATE: 29th January 2023

Assignment 1

Working with Oracle 12c SQL and DB concepts

Due Date: 29th January 2023 @11:59 PM

Max marks : 60 **Weight: 5%**

Part 1 Maintaining and redesigning an existing database

Create the following schema for maintaining the books issued to the patrons.

books (book_id int, title varchar(20), author_last_name varchar(20), author_first_name varchar(20), rating char(1))

patrons (patron_id int, last_name varchar(20), first_name varchar(20), street_address varchar(30), city varchar(10), zip char(7))

transactions (transaction_id int, patron_id int, book_id int, transaction_date date, transaction_type char(1))

Book_Id, Patron_id and Transaction_id are primary keys. Patron_id and book_id in transactions table are foreign keys.
Possible values for the transaction_type are 1 = checking out , 2 = returning, 3 = placing a hold.
Possible values for the rating are 1 = text book, 2 = reference book, 3 = others

Use the existing database to do the following:

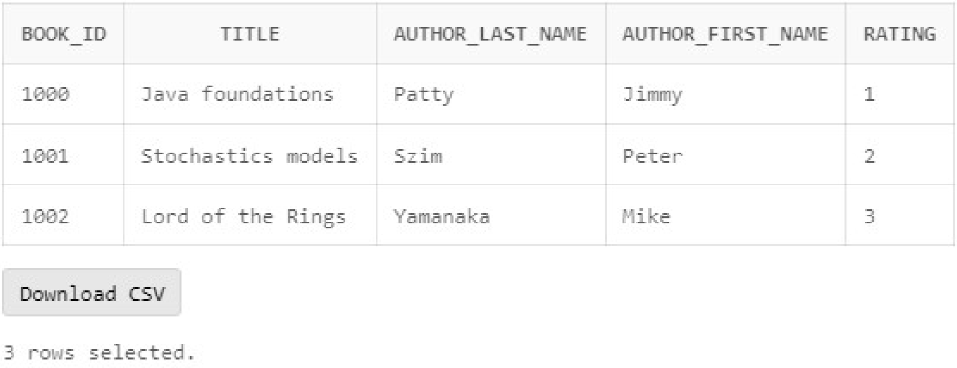
1. Adding values in the primary key: (4 marks)
   CREATE SEQUENCE is a SQL statement to create a **sequence**, which can be used to generate unique integers. The CURRVAL pseudocolumn, returns the current value of the sequence, and the NEXTVAL pseudocolumn, increments the sequence and returns the new value. You can use sequences to automatically generate primary key values.

```
CREATE SEQUENCE customers_seq
 START WITH     1000
 INCREMENT BY   1
```

   Use customer_seq to insert data as primary key as:
   Insert into books values(books_seq.nextval, ' ', ' ', ……)

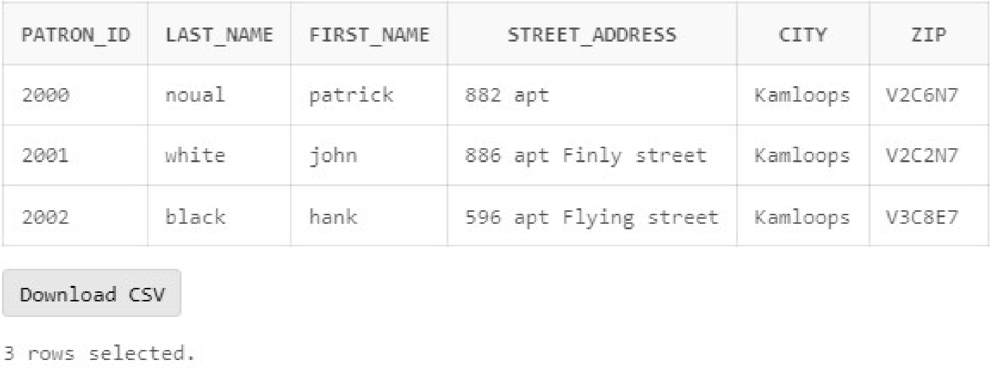
```
-- generate primary keys
-- generate for books
CREATE SEQUENCE sequence_1
start with 1000
increment by 1;
-- generate for patrons
CREATE SEQUENCE sequence_patrons
start with 2000
increment by +1;
-- generate for transaction
CREATE SEQUENCE sequence_transaction
start with 3000
increment by +1;

-- (1) add primary keys (and insert data to table)
-- add 3 books
INSERT INTO books (book_id,title,author_last_name, author_first_name, rating)
VALUES (sequence_1.nextval,'Java foundations','Patty','Jimmy','1');
INSERT INTO books (book_id,title,author_last_name, author_first_name, rating)
VALUES (sequence_1.nextval,'Stochastics models','Szim','Peter','2');
INSERT INTO books (book_id,title,author_last_name, author_first_name, rating)
VALUES (sequence_1.nextval,'Lord of the Rings','Yamanaka','Mike','3');

 -- add 3 patrons
INSERT INTO patrons (patron_id ,last_name ,first_name, street_address , city ,
zip)
VALUES (sequence_patrons.nextval,'noual','patrick','882 apt'
,'Kamloops','V2C6N7');
INSERT INTO patrons (patron_id ,last_name ,first_name, street_address , city ,
zip)
VALUES (sequence_patrons.nextval,'white','john','886 apt Finly street'
,'Kamloops','V2C2N7');
INSERT INTO patrons (patron_id ,last_name ,first_name, street_address , city ,
zip)
VALUES (sequence_patrons.nextval,'black','hank','596 apt Flying street'
,'Kamloops','V3C8E7');

-- add 3 transactions
INSERT INTO transactions (transaction_id , patron_id , book_id  ,
transaction_date  , transaction_type)
VALUES (sequence_transaction.nextval,2000,1000, to_date('09-01-22','MM-DD-
YY') ,'1');
INSERT INTO transactions (transaction_id , patron_id , book_id  ,
transaction_date  , transaction_type)
VALUES (sequence_transaction.nextval,2001,1001, to_date('09-03-22','MM-DD-
YY') ,'2');
INSERT INTO transactions (transaction_id , patron_id , book_id  ,
transaction_date  , transaction_type)
```

```
VALUES (sequence_transaction.nextval,2002,1002, to_date('09-05-22','MM-DD-YY') ,'3');
```



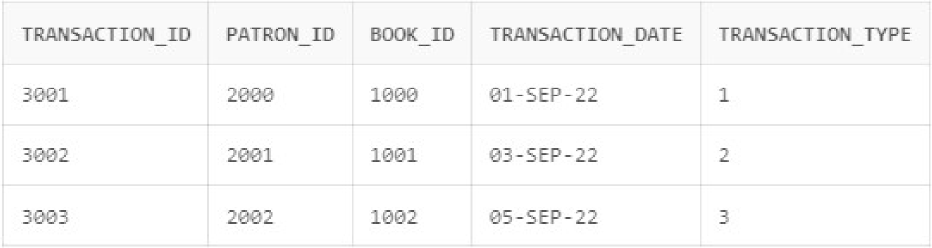
BOOK_ID	TITLE	AUTHOR_LAST_NAME	AUTHOR_FIRST_NAME	RATING
1000	Java foundations	Patty	Jimmy	1
1001	Stochastics models	Szim	Peter	2
1002	Lord of the Rings	Yamanaka	Mike	3

```
3 rows selected.
```



PATRON_ID	LAST_NAME	FIRST_NAME	STREET_ADDRESS	CITY	ZIP
2000	noual	patrick	882 apt	Kamloops	V2C6N7
2001	white	john	886 apt Finly street	Kamloops	V2C2N7
2002	black	hank	596 apt Flying street	Kamloops	V3C8E7

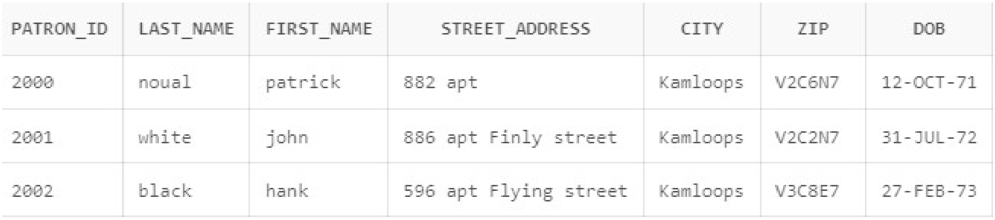
```
3 rows selected.
```



TRANSACTION_ID	PATRON_ID	BOOK_ID	TRANSACTION_DATE	TRANSACTION_TYPE
3001	2000	1000	01-SEP-22	1
3002	2001	1001	03-SEP-22	2
3003	2002	1002	05-SEP-22	3

2. **Adding new columns:** (3 marks)
   a. Write an SQL statement to add a new column DOB to the PATRONS table. Print the query and the results.

```
ALTER TABLE patrons
ADD DOB DATE;
```

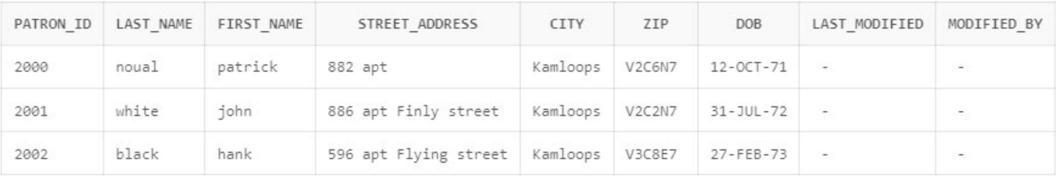


PATRON_ID	LAST_NAME	FIRST_NAME	STREET_ADDRESS	CITY	ZIP	DOB
2000	noual	patrick	882 apt	Kamloops	V2C6N7	12-OCT-71
2001	white	john	886 apt Finly street	Kamloops	V2C2N7	31-JUL-72
2002	black	hank	596 apt Flying street	Kamloops	V3C8E7	27-FEB-73

b. Write an SQL statement to add LAST_MODIFIED and MODIFIED_BY columns to the PATRONS table. The LAST_MODIFIED column will have the server date and time and the MODIFIED_BY will have the USER name.

```
ALTER TABLE PATRONS
ADD LAST_MODIFIED DATE;
```

```
ALTER TABLE PATRONS
ADD MODIFIED_BY VARCHAR(10);
```



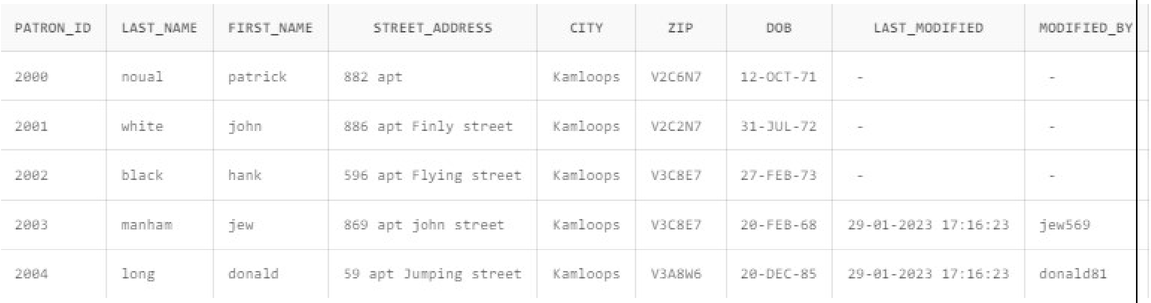
PATRON_ID	LAST_NAME	FIRST_NAME	STREET_ADDRESS	CITY	ZIP	DOB	LAST_MODIFIED	MODIFIED_BY
2000	noual	patrick	882 apt	Kamloops	V2C6N7	12-OCT-71	-	-
2001	white	john	886 apt Finly street	Kamloops	V2C2N7	31-JUL-72	-	-
2002	black	hank	596 apt Flying street	Kamloops	V3C8E7	27-FEB-73	-	-

3. **Add patrons (3 marks)**

a. **Add 2 patrons with their *DOB, LAST_MODIFIED, and MODIFIED_BY*.**

```
INSERT INTO patrons (patron_id ,last_name ,first_name, street_address ,
city , zip,DOB, LAST_MODIFIED, MODIFIED_BY)
VALUES (sequence_patrons.nextval,'manham','jew','869 apt john street'
,'Kamloops','V3C8E7',to_date('02-20-68','MM-DD-YY'),TO_CHAR (Sysdate,
'DD-MM-YYYY HH24:MI:SS'),'jew569');
```

```
INSERT INTO patrons (patron_id ,last_name ,first_name, street_address ,
city , zip,DOB, LAST_MODIFIED, MODIFIED_BY)
VALUES (sequence_patrons.nextval,'long','donald','59 apt Jumping street'
,'Kamloops','V3A8W6',to_date('12-20-85','MM-DD-YY'),TO_CHAR (Sysdate,
'DD-MM-YYYY HH24:MI:SS'),'donald81');
```

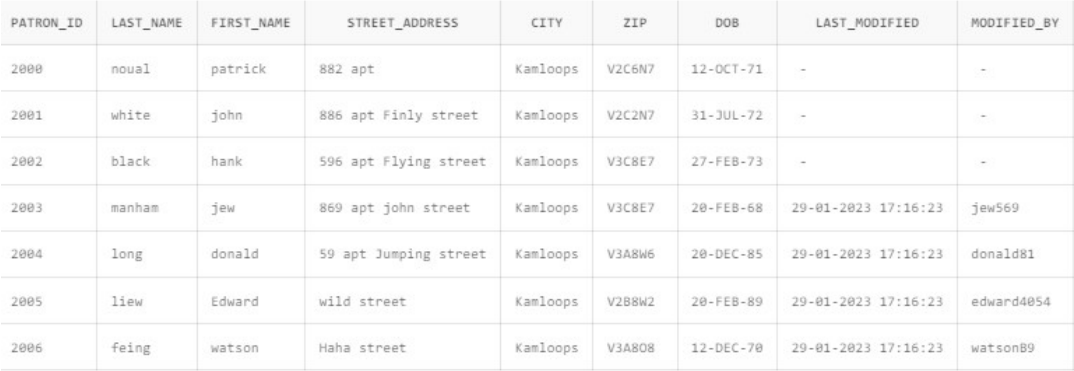


PATRON_ID	LAST_NAME	FIRST_NAME	STREET_ADDRESS	CITY	ZIP	DOB	LAST_MODIFIED	MODIFIED_BY
2000	noual	patrick	882 apt	Kamloops	V2C6N7	12-OCT-71	-	-
2001	white	john	886 apt Finly street	Kamloops	V2C2N7	31-JUL-72	-	-
2002	black	hank	596 apt Flying street	Kamloops	V3C8E7	27-FEB-73	-	-
2003	manham	jew	869 apt john street	Kamloops	V3C8E7	20-FEB-68	29-01-2023 17:16:23	jew569
2004	long	donald	59 apt Jumping street	Kamloops	V3A8W6	20-DEC-85	29-01-2023 17:16:23	donald81

b. Write an SQL statement to add two new patrons to the table. Use the TO_DATE function for their DOBs. Use the existing sequence for the patron id. Print the query and the results.

```
INSERT INTO patrons (patron_id ,last_name ,first_name, street_address , city ,
zip,DOB, LAST_MODIFIED, MODIFIED_BY)
```

```
VALUES (sequence_patrons.nextval,'liew','Edward','wild street'
,'Kamloops','V2B8W2',to_date('02-20-89','MM-DD-YY'),TO_CHAR (Sysdate, 'DD-
MM-YYYY HH24:MI:SS'),'edward4054');

INSERT INTO patrons (patron_id ,last_name ,first_name, street_address , city ,
zip,DOB, LAST_MODIFIED, MODIFIED_BY)
VALUES (sequence_patrons.nextval,'feing','watson','Haha street'
,'Kamloops','V3A8O8',to_date('12-12-70','MM-DD-YY'),TO_CHAR (Sysdate, 'DD-
MM-YYYY HH24:MI:SS'),'watsonB9');
```



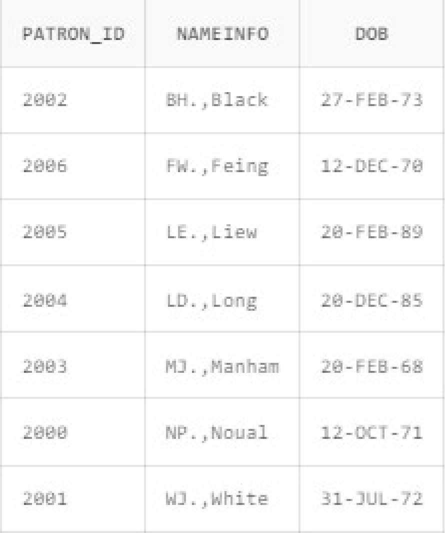
PATRON_ID	LAST_NAME	FIRST_NAME	STREET_ADDRESS	CITY	ZIP	DOB	LAST_MODIFIED	MODIFIED_BY
2000	noual	patrick	882 apt	Kamloops	V2C6N7	12-OCT-71	-	-
2001	white	john	886 apt Finly street	Kamloops	V2C2N7	31-JUL-72	-	-
2002	black	hank	596 apt Flying street	Kamloops	V3C8E7	27-FEB-73	-	-
2003	manham	jew	869 apt john street	Kamloops	V3C8E7	20-FEB-68	29-01-2023 17:16:23	jew569
2004	long	donald	59 apt Jumping street	Kamloops	V3A8W6	20-DEC-85	29-01-2023 17:16:23	donald81
2005	liew	Edward	wild street	Kamloops	V2B8W2	20-FEB-89	29-01-2023 17:16:23	edward4054
2006	feing	watson	Haha street	Kamloops	V3A8O8	12-DEC-70	29-01-2023 17:16:23	watsonB9

Part 2 SQL queries (35 marks)

Use the modified database from PART 1.

1. Write and SQL query to list the patrons (patron_id, last_name, first_name, DOB) whose data were added/modified today (use SYSDATE as today's date). The first name and last name should have the following format: ***Upper case Initial, period, comma, and Last name in a mixed case***. Use the ISO standard for DOB (YYYY-MM-DD). Sort the results by last name. Print the query and the results.

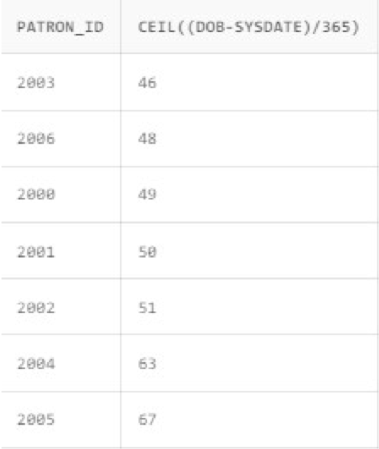
```
SELECT PATRON_ID, substr(UPPER(last_name),0,1) || '' ||
substr(UPPER(FIRST_name),0,1)  || '.,' ||  substr(UPPER(last_name),0,1)  || '' ||
substr((last_name),2) AS nameInfo, DOB
FROM patrons ORDER BY last_name;
```



PATRON_ID	NAMEINFO	DOB
2002	BH.,Black	27-FEB-73
2006	FW.,Feing	12-DEC-70
2005	LE.,Liew	20-FEB-89
2004	LD.,Long	20-DEC-85
2003	MJ.,Manham	20-FEB-68
2000	NP.,Noual	12-OCT-71
2001	WJ.,White	31-JUL-72

2. Write SQL to list all patrons with their ids and age in years (rounded up). Age is calculated based on the server date/time. Sort the results by age in an ascending order
.

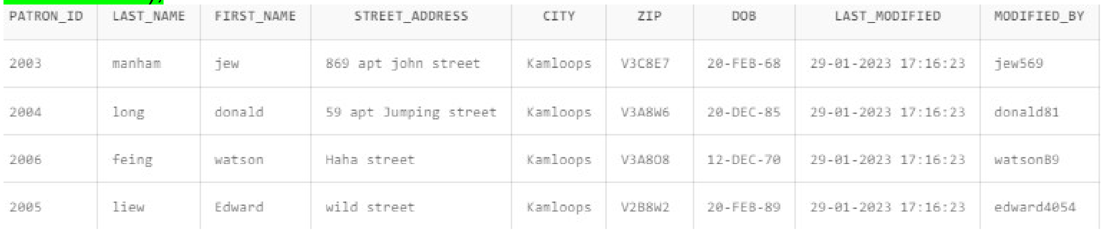
SELECT PATRON_ID, CEIL((DOB-sysdate)/365) FROM patrons
ORDER BY CEIL((DOB-sysdate)/365) ASC;



PATRON_ID	CEIL((DOB-SYSDATE)/365)
2003	46
2006	48
2000	49
2001	50
2002	51
2004	63
2005	67

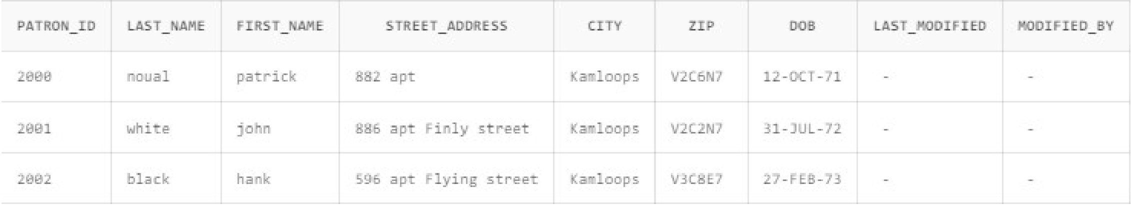
3. Write SQL to list all patrons who had any transactions ever. Do not include the patron if they do not have transactions.

-- no transactions patrons
SELECT * FROM patrons WHERE PATRON_ID NOT IN (SELECT PATRON_ID FROM transactions);



PATRON_ID	LAST_NAME	FIRST_NAME	STREET_ADDRESS	CITY	ZIP	DOB	LAST_MODIFIED	MODIFIED_BY
2003	manham	jew	869 apt john street	Kamloops	V3C8E7	20-FEB-68	29-01-2023 17:16:23	jew569
2004	long	donald	59 apt Jumping street	Kamloops	V3A8W6	20-DEC-85	29-01-2023 17:16:23	donald81
2006	feing	watson	Haha street	Kamloops	V3A8O8	12-DEC-70	29-01-2023 17:16:23	watsonB9
2005	liew	Edward	wild street	Kamloops	V2B8W2	20-FEB-89	29-01-2023 17:16:23	edward4054

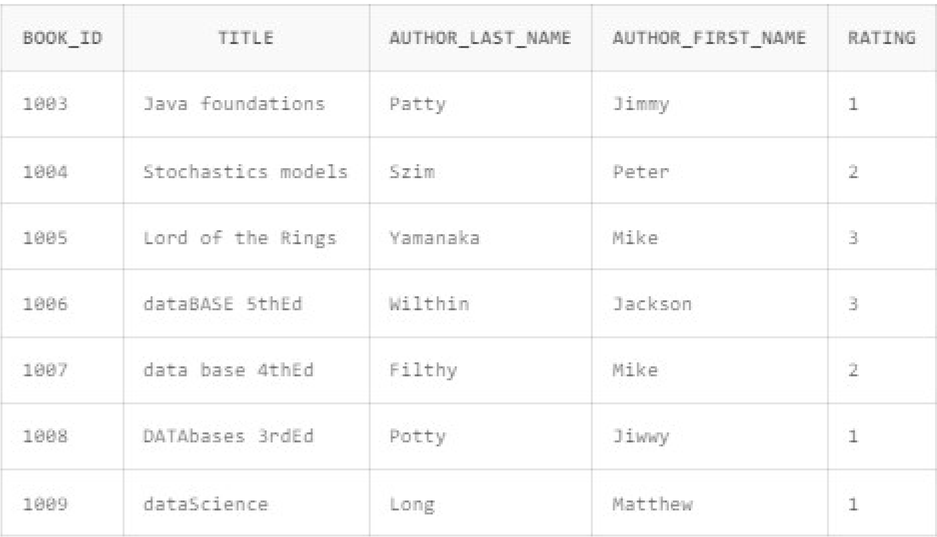
- - has transactions patrons
SELECT * FROM patrons WHERE PATRON_ID IN (SELECT PATRON_ID FROM transactions);



PATRON_ID	LAST_NAME	FIRST_NAME	STREET_ADDRESS	CITY	ZIP	DOB	LAST_MODIFIED	MODIFIED_BY
2000	noual	patrick	882 apt	Kamloops	V2C6N7	12-OCT-71	-	-
2001	white	john	886 apt Finly street	Kamloops	V2C2N7	31-JUL-72	-	-
2002	black	hank	596 apt Flying street	Kamloops	V3C8E7	27-FEB-73	-	-

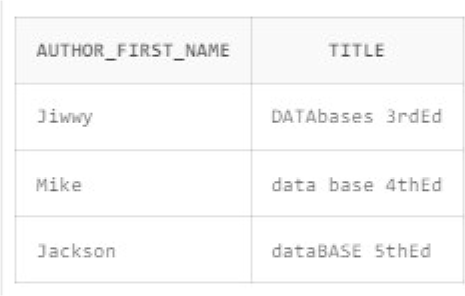
4. Write SQL to list the books with the word “database” or “data base” or “databases” in their title. Make your search case insensitive. Print the author’s last name (mixed case) and the title. Sort the results by the rating.

current records in book table



BOOK_ID	TITLE	AUTHOR_LAST_NAME	AUTHOR_FIRST_NAME	RATING
1003	Java foundations	Patty	Jimmy	1
1004	Stochastics models	Szim	Peter	2
1005	Lord of the Rings	Yamanaka	Mike	3
1006	dataBASE 5thEd	Wilthin	Jackson	3
1007	data base 4thEd	Filthy	Mike	2
1008	DATAbases 3rdEd	Potty	Jiwwy	1
1009	dataScience	Long	Matthew	1

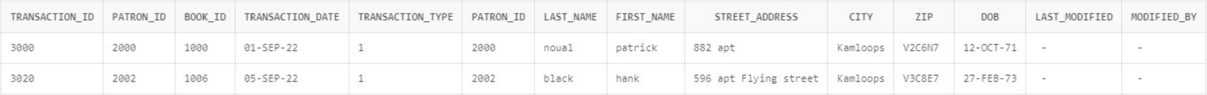
```
SELECT AUTHOR_FIRST_NAME ,TITLE  FROM books where UPPER(TITLE) LIKE
UPPER('%database%')
   OR UPPER(TITLE) LIKE UPPER('%data base%')
   OR UPPER(TITLE) LIKE UPPER('%databases%')
   ORDER BY RATING ASC;
```



AUTHOR_FIRST_NAME	TITLE
Jiwwy	DATAbases 3rdEd
Mike	data base 4thEd
Jackson	dataBASE 5thEd

5. Write a join command to display all the patrons with at least one book issued to them.

```
SELECT  *  FROM transactions
INNER JOIN patrons ON patrons.PATRON_ID=transactions.PATRON_ID WHERE
TRANSACTION_TYPE = '1';
```

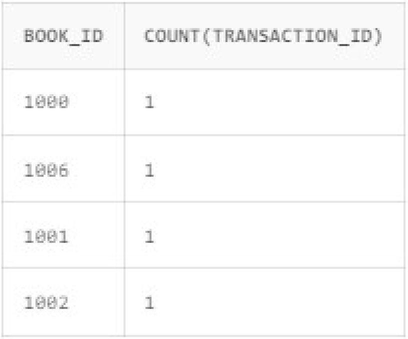


TRANSACTION_ID	PATRON_ID	BOOK_ID	TRANSACTION_DATE	TRANSACTION_TYPE	PATRON_ID	LAST_NAME	FIRST_NAME	STREET_ADDRESS	CITY	ZIP	DOB	LAST_MODIFIED	MODIFIED_BY
3000	2000	1000	01-SEP-22	1	2000	noual	patrick	882 apt	Kamloops	V2C6N7	12-OCT-71	-	-
3020	2002	1006	05-SEP-22	1	2002	black	hank	596 apt Flying street	Kamloops	V3C8E7	27-FEB-73	-	-

6. Write SQL to list all the books (book ids and first 10 characters of the title) and their total number of transactions in 2020. Include all books even if they have no transactions (count should be 0). Sort the results by the most “popular” book (the book with the most transactions).

```
SELECT book_id, count(TRANSACTION_ID) from transactions
```

group by book_id;



BOOK_ID	COUNT(TRANSACTION_ID)
1000	1
1006	1
1001	1
1002	1

Part 3 Research - 15 marks

Using online documentation, find out about data types used in MS Access 2010, Oracle 12g, and MySQL 5.0. Study two data types modeling issues in these three DBMS:

1. Compare and contrast the methods for creating an **automatic unique identifiers** in each DBMS. Put your answer in a short point/paragraph/table format. List references (at least one source).

   - unique identifiers used to distinguish 1 column to another.

   -

MS Access 2010	Oracle 12g	MySQL 5.0
- Syntax to create is short and easier. - Using GUID can cause database server to have poor performance. - Procedures for generation can't be grouped into packages. - Identifiers are less complex, which reduces safety for security purposes.	- Examples are SYS_GUID () - It takes more time to generate a unique key as Oracle doesn't interpret RAW datatype directly. - It guarantees value uniqueness as values are largely based on random generation. - Numbers generated are random rather than sequential, which reduces contention for hot blocks. - Can't perform arithmetic operations on value	- Examples are UUID function, returns a universal unique identifier. - Values are generated uniquely based on space and time. - Values are unique (value that occurs once in data) across tables, databases, and servers. - Values can be generated offline, which increases security. - It makes less complexity in the logic structure. (i.e., We can insert primary keys into 2 tables simultaneously) - It makes debugging more difficult as default format is complex.

Sources: https://www.javatpoint.com/mysql-uuid , link2

2. Compare and contrast the **storage of date, time, and time zone.** Specify the data types for storing date, time, and time zone in each DBMS. How dates are stored internally in each DBMS? What is the maximum range for dates? Prepare your answer in a table

format. List references (at least one source).

MS Access 2010	Oracle 12g	MySQL 5.0
- Stores Date/Time datatype as a double-precision. The integer part represents date, while the decimal part represents time. - Internally, storing and calculating date or time are done in double-precision format. - The range for date is 0001-01-01 through 9999-12-31	- DATE datatype stores year, month, day, hour, minute and second - We use timestamp datatype when storing time interval between 2 events. - Time zones are created through performing arithmetic. - Internally, Oracles stores dates in numeric format. - The range has no limits, but must in the format of 'YYYY-MM-DD'	- DATE datatype store only year, month, and day. So, DATATIME datatypes must be used for preserving time. - Internally, dates are stored as 2 integers. The first integer is the number of dates before/after base date. While 2 integer stores the number of clock ticks after midnight. - The range for dates is ‘1000-01-01’ to ‘9999-12-31’ and must in the format ‘YYYY-MM-DD’

Sources: https://learn.microsoft.com/en-us/sql/t-sql/functions/date-and-time-data-types-and-functions-transact-sql?view=sql-server-ver16

Submission of the Assignment

Submit the assignment having two files:

1. A pdf file containing
   i. Command written below the question
   ii. Screen shots of the results generated from query.
   iii. Your document should have a title page with your name, student number, course title, and date and saved as pdf
2. Make a text file having all the commands ONLY. Save it as a .SQL file separately