



Database Management Sample Assignment | [www.expertsmind.com](http://www.expertsmind.com)

## Coursework 2 - Further SQL and PLSQL

---

This is an **INDIVIDUAL** coursework.

**Before you start this coursework you should have completed the SQL exercises on the Bus/Drivers/Depots Database. If you need to revise this topic, see Text book Chapters 5 & 6.**

**Important note** - check that you have not, in previous modules, created tables with the same names as in this exercise. If any of these table names already exist in your area within Oracle then you must either delete them (using 'drop table xyz cascade constraints') or use different names for tables in the coursework database. There may also be problems if constraint primary key and foreign key names are the same so it is advisable to delete all tables created in previous modules. After creating and loading tables check that all data is present.

Use Oracle SQL\*PLUS to answer the following queries. It is recommended that you either use Notepad or similar (not Word) to initially create the queries.

### The Children's Library Database

This scenario relates to a database which is to be set up for a library which lends books to children:

A Children's Library keeps information on books held, borrowers who borrow these books and the loans of these books, which the borrowers make. In addition information is held about the authors and publishers of these books.

The information is to be held in a Relational database and implemented using Oracle. Two files are available on OasisPlus for you to create the relations and to load the

relations with sample data. The tables in the database are shown below. The identifier attribute(s) are underlined, foreign keys are asterisked \*.

Table	Attributes
Borrower	<u>borId</u> , borName, borAddress, borMaxbooks
BookTitle	<u>isbn</u> , btName, datePublished, pubId*, ageLower, ageUpper, value
BookCopy	<u>bcd</u> , isbn*, dateAcquired, dateDestroyed
Loan	<u>borId*</u> , <u>bcd*</u> , <u>dateOut</u> , dateDue, dateBack
Publisher	<u>pubId</u> , pubName, pubAddress
Author	<u>authorId</u> , authorName
Authorship	<u>authorId*</u> , <u>isbn*</u>

Note that the BookCopy table holds information on the physical books stored in the library whereas the BookTitle table holds information on a particular publication of a book. For example, there are two copies of 'Winnie the Pooh', with bcd of 101 and 102 with an ISBN (International Standard Book Number) of 10: 0786843411. The BookTitle table hold information about the recommended age range for each book (ageLower and ageUpper). A book may have a number of authors and this is indicated in the Authorship table.

The attribute borMaxbooks indicates the maximum number of books that a borrower can borrow at a time. Also, a book, which is still out on loan, will have a blank dateBack field in the loan table.

As copies of books become old, damaged and dirty the books are removed from the library and destroyed. Destroyed book copies have a date to indicate this otherwise the date is null.

It is essential that you draw an Entity-Relationship diagram mapped to a relational database, showing the entities and relationships involved. Assume that attributes with the same name are based on the same domain. Do not hand in this diagram – it is not assessed.

## Coursework 2 - Further SQL

Use join conditions to answer the queries in this questions 1 and 2

Q1 Find borrowers (by name) who have ever borrowed a book written by Phillip Pullman.

Q2 Find borrowers (by name) who have currently on loan a book written by Phillip Pullman

Q3 Give the ISBN and title of each book and the number of copies currently on loan listing most popular first.

Q4 Display the names of borrowers who have never borrowed books published by the publisher Mammoth.

Q5 Give the titles of books which have the same value as 'Northern Lights'. Do not include Northern Lights in your output.



Q6 List all book titles and the total value of the book copies for a particular title which are currently on loan where the total value is greater than or equal to £20.

Q7 List the borrowers (by number and name, once only), who have currently on loan a book which has the same title as a book that Jenny Wren has borrowed. Do not include Jenny Wren in the output.

Q8 Give the name of any borrower who has ever borrowed all the book copies in the library. It is possible that no-one has borrowed all the copies. (Hint: this is an example of relational algebra divide).

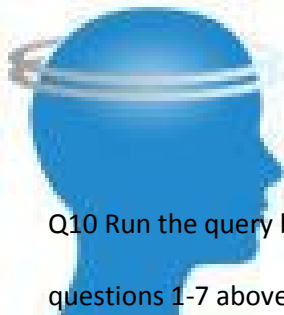
Continued/



Q9 Run the Create View statement and the query below and show the results. Give the English meaning of the query (as in questions 1-7 above):

```
Create view borrowedBooks
As      Select l.borId, bc.isbn, bc.bcid
        From Loan l, bookCopy bc
        Where l.bcid = bc.bcid;
```

```
Select b.borName
From Borrower b
Where not exists
  (select *
   from bookTitle bt, Publisher p
   where bt.pubId=p.pubId
   and pubName='Puffin'
   and not exists
     (select *
      from borrowedBooks bb
      where bb.borId = b.borId
      and bb.isbn = bt.isbn));
```



*Experts Mind*

Live Experts 24x7



Q10 Run the query below and show the results. Give the English meaning of the query (as in questions 1-7 above):

```
Select distinct b.borName
From  Borrower b, Loan l, Book_copy bc,
      Author a1, Author a2,
      Authorship ash1, Authorship ash2
Where b.borId = l.borId
And l.bcid = bc.bcid
And bc.isbn = ash1.isbn
And bc.isbn = ash2.isbn
And ash1.authorId = a1.authorId
And ash2.authorId = a2.authorId
```

And a1.authorName = 'Janet Ahlberg'

And a2.authorName = 'Allan Ahlberg';

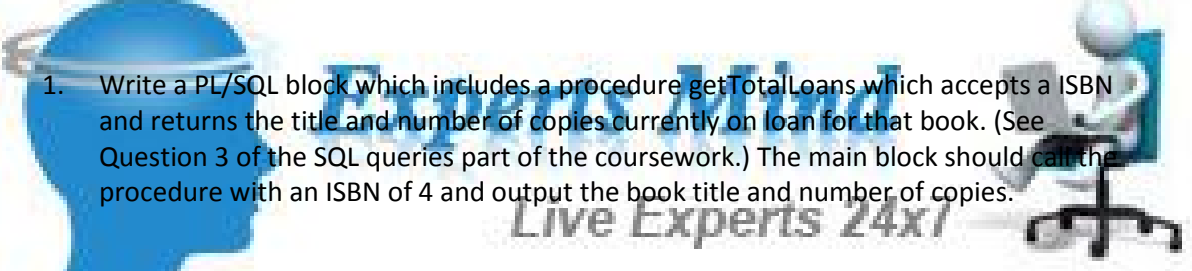
## Coursework 2 PL/SQL - Triggers and stored procedures

**Don't forget to type the command 'set serveroutput on' before executing the code in this section. If you are forced out of SQL\*PLUS don't forget to retype this command.**

Again it is recommended that you either use Notepad or similar to initially create the code.

**Before you start this coursework you should have completed the PL/SQL exercises on the Bus/Drivers/Depots Database. If you need to revise this topic, see text book Chapter 6.**

I have put some PLSQL code examples (from the text book) on OasisPlus to use as templates to help you with this part of the coursework.

- 
1. Write a PL/SQL block which includes a procedure `getTotalLoans` which accepts a ISBN and returns the title and number of copies currently on loan for that book. (See Question 3 of the SQL queries part of the coursework.) The main block should call the procedure with an ISBN of 4 and output the book title and number of copies.
  2. Write a PL/SQL block, which displays for all book titles held in the library (not those destroyed). You should include the ISBN, the book title, publisher name and the number of copies of each held (whether on loan or not). Use a cursor to do this question.
  3. Write a stored function called `getPublisher`. This function takes as input an ISBN for a book and returns the publisher name.

Call the function from within an SQL statement to display the publisher for book title 'The Twits'

4. Create a trigger '`checkRecommendedAge`' to enforce the (harsh) constraint that children are only allowed to borrow books deemed suitable for their age. The trigger fires whenever there is a new loan and outputs an error message whenever an attempt is made to borrow a book where the borrower's actual age is not within the lower to upper age range for the book. Ignore the fact that some of the data already in the database may violate the constraint.

Show what happens when you try to insert following records into the Loan table using the following statements:

```
Insert into Loan values (001,144,'29-aug-2008','18-sep-2008','');
```

```
Insert into Loan values (001,101,'29-aug-2008','18-sep-2008','');
```

```
Insert into Loan values (001,120,'29-aug-2008','18-sep-2008','');
```

For testing purposes the following commands (which set the database back to its original state) might be useful:

```
delete from Loan where borId=001 and bcId=144 and dateOut='29-aug-2008';
```

```
delete from Loan where borId=001 and bcId=101 and dateOut='29-aug-2008';
```

```
delete from Loan where borId=001 and bcId=120 and dateOut='29-aug-2008';
```



**Note well: read following page before handing in your coursework**

**Your submission should consist of an Oracle SQL print-out of the SQL code which is your solution to the problems above.** Note that marks will be awarded for SQL and PLSQL code which is easy to read and hence easy to debug. Each answer should consist of the following:

- For SQL coding, indication of the question a particular solution refers to, written as a comment e.g. /\* Question 1 List the borrowers.....\*/
- For PLSQL coding the question number is required only
- The SQL and PLSQL code in its entirety, formatted with indentations so it is easy to read (see code in text book). Please use a reasonable size of font.

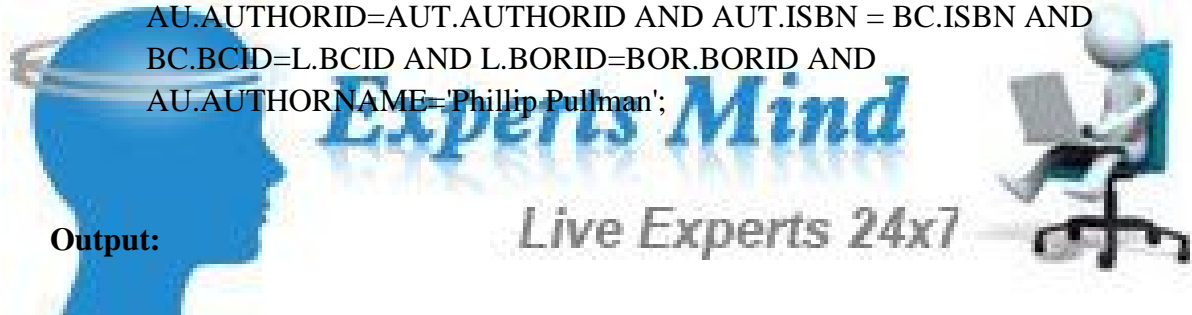
- SQL\*Plus Screen print to verify that you have executed the code
- The resulting table created from the SQL execution or results from your PLSQL execution. You should copy and paste the results from SQL\*Plus. If the output is very long, hand in the first page only. Note that the correct solutions generate small tables. Do not print/hand in unnecessary paper- save trees!
- No cds or similar to be handed in

Q1 Find borrowers (by name) who have ever borrowed a book written by Phillip Pullman.

**Query:**

```
SELECT BOR.BORNAME FROM AUTHOR AU, AUTHORSHIP
AUT,BOOKCOPY BC,LOAN L,BORROWER BOR WHERE
AU.AUTHORID=AUT.AUTHORID AND AUT.ISBN = BC.ISBN AND
BC.BCID=L.BCID AND L.BORID=BOR.BORID AND
AU.AUTHORNAME='Phillip Pullman';
```

**Output:**



BORNAME
Keith Kettle
Jay Patel

Q2 Find borrowers (by name) who have currently on loan a book written by Phillip Pullman

**Query:**

```
SELECT BOR.BORNAME FROM AUTHOR AU, AUTHORSHIP AUT,BOOKCOPY BC,LOAN
L,BORROWER BOR WHERE AU.AUTHORID=AUT.AUTHORID AND AUT.ISBN =
```



BC.ISBN AND BC.BCID=L.BCID AND L.BORID=BOR.BORID AND

AU.AUTHORNAME='Phillip Pullman' AND L.DATEBACK IS NULL;

**Output:**

BORNAME
Keith Kettle



Q3 Give the ISBN and title of each book and the number of copies currently on loan listing  
most popular first

**Query:**

```
SELECT BT.ISBN,BT.BTNAME,COUNT(*) FROM BOOKTITLE  
BT,BOOKCOPY BC,LOAN L WHERE BT.ISBN=BC.ISBN AND  
BC.BCID=L.BCID AND L.DATEBACK IS NULL GROUP BY  
BT.ISBN,BT.BTNAME ORDER BY COUNT(*) DESC
```

**Output:**



ISBN	BTNAME	COUNT(*)
5	Matilda	3
6	Burglar Bill	3
1	Winnie the Pooh	2
4	The Silver Sword	2
7	The Twits	2
11	Making Up	1
12	Northern Lights	1
14	Harry Potter and the Deathly	1
13	Harry Potter and the Phil	1
9	Hiding Out	1
2	What Katy Did	1



Q4 Display the names of borrowers who have never borrowed books published by the publisher Mammoth.

**Query:**

```
SELECT BOR.BORNAME FROM BOOKTITLE BT,LOAN L,BOOKCOPY  
BC,BORROWER BOR WHERE BT.ISBN=BC.ISBN AND  
BC.BCID=L.BCID AND L.BORID=BOR.BORID AND BT.PUBID NOT IN  
(SELECT PUBID FROM PUBLISHER WHERE PUBNAME<> 'Mammoth')
```

**Output:**




Q5 Give the titles of books which have the same value as 'Northern Lights'. Do not include Northern Lights in your output.

**Query:**

```
SELECT BTNAME FROM BOOKTITLE WHERE VALUE IN (SELECT VALUE FROM  
BOOKTITLE WHERE BTNAME ='Northern Lights') AND BTNAME <> 'Northern Lights';
```

**Output:**



BTNAME
Toms Midnight Garden
Harry Potter and the Deathly
Harry Potter and the Phil
Making Up
Swallows and Amazons
What Katy Did

Q6 List all book titles and the total value of the book copies for a particular title which are currently on loan where the total value is greater than or equal to £20.

**Query:**

```
SELECT BT.BTNAME,SUM(BT.VALUE) FROM BOOKTITLE BT,BOOKCOPY BC,LOAN L
WHERE BT.ISBN=BC.ISBN AND BC.BCID=L.BCID AND L.DATEBACK IS NULL GROUP BY
BT.BTNAME HAVING SUM(BT.VALUE)>=20
```



**Output:**

BTNAME	SUM(BT.VALUE)
The Silver Sword	22
Matilda	24
The Twits	28

Q7 List the borrowers (by number and name, once only), who have currently on loan a book which has the same title as a book that Jenny Wren has borrowed. Do not include Jenny Wren in the output.

**Query:**

```
SELECT DISTINCT BOR.BORID,BOR.BORNAME FROM BORROWER BOR,LOAN
L,BOOKTITLE BT WHERE BOR.BORID=L.BORID AND BT.BTNAME IN (SELECT
BT.BTNAME FROM BORROWER BBR,LOAN L,BOOKCOPY BC,BOOKTITLE BT WHERE
BBR.BORID=L.BORID AND L.BCID=BC.BCID AND BT.ISBN=BC.ISBN AND
BBR.BORNAME = 'Jenny Wren') AND BOR.BORNAME <> 'Jenny Wren'
```



*Experts Mind*  
*Live Experts 24x7*



**Output:**

BORID	BORNAME
15	Polly Peck
1	Jack Jones
12	Billy Black
14	Keith Kettle
2	Betty Smith
9	Jay Patel

Q8 Give the name of any borrower who has ever borrowed all the book copies in the library.

It is possible that no-one has borrowed all the copies. (Hint: this is an example of relational algebra divide).

**Query:**

```
SELECT BOR.BORNAME, COUNT(*) FROM BORROWER BOR, LOAN L, BOOKCOPY BC
WHERE BOR.BORID=L.BORID AND L.BCID=BC.BCID AND L.BCID IN (SELECT BCID
FROM BOOKCOPY) GROUP BY BOR.BORNAME HAVING COUNT(*) = (SELECT
COUNT(*) FROM BOOKCOPY)
```



Q9 Run the Create View statement and the query below and show the results. Give the

English meaning of the query (as in questions 1-7 above):

```
Create view borrowedBooks
As Select l.borId, bc.isbn, bc.bcid
   From Loan l, bookCopy bc
   Where l.bcid = bc.bcid;
```

```
Select b.borName
From Borrower b
Where not exists
  (select *
   from bookTitle bt, Publisher p
   where bt.pubId=p.pubId
   and pubName='Puffin'
   and not exists
     (select *
      from borrowedBooks bb
      where bb.borId = b.borId
```

```
and bb.isbn = bt.isbn));
```

**Meaning of the Query:**

List the borrowers (name), who did not borrow the books till now as well as the borrowed book publisher name should not be Puffin





Q10 Run the query below and show the results. Give the English meaning of the query (as in questions 1-7 above):

```
Select distinct b.borName
From Borrower b, Loan l, Bookcopy bc,
Author a1, Author a2,
Authorship ash1, Authorship ash2
Where b.borId = l.borId
And l.bcId = bc.bcId
And bc.isbn = ash1.isbn
And bc.isbn = ash2.isbn
And ash1.authorId = a1.authorId
And ash2.authorId = a2.authorId
And a1.authorName = 'Janet Ahlberg'
And a2.authorName = 'Allan Ahlberg';
```



**Meaning of the Query:**

List the borrowers (name, once only), who have borrowed a book which has the author name may be 'Janet Ahlberg' or book authorname be 'Allan Ahlberg'

5. Write a PL/SQL block which includes a procedure `getTotalLoans` which accepts a ISBN and returns the title and number of copies currently on loan for that book. (See Question 3 of the SQL queries part of the coursework.) The main block should call the procedure with an ISBN of 4 and output the book title and number of copies.

**Query:**

```
create or replace procedure "GETTOTALLOANS"
```

```
(tisbn IN BOOKTITLE.ISBN%type ,
```

```
ttitle OUT BOOKTITLE.BTNAME%type,
```

```
ncopies OUT NUMBER)
```

```
is
```

```
begin
```

```
    SELECT BT.BTNAME INTO ttitle FROM BOOKTITLE BT, LOAN L, BOOKCOPY BC
```

```
    WHERE BT.ISBN=BC.ISBN AND BC.BCID=L.BCID AND BC.ISBN=tisbn;
```

```
    SELECT COUNT(*) INTO ncopies FROM BOOKCOPY BC,LOAN L WHERE
```

```
    BC.BCID=L.BCID AND BC.ISBN=tisbn GROUP BY BC.BCID;
```

```
end;
```



6. Write a PL/SQL block, which displays for all book titles held in the library (not those destroyed). You should include the ISBN, the book title, publisher name and the number of copies of each held (whether on loan or not). Use a cursor to do this question.

**Query:**

```
DECLARE
```

```
TISBN BOOKTITLE.ISBN%TYPE;
```

```
TTITLE BOOKTITLE.BTNAME%TYPE;
```

```
PNAME PUBLISHER.PUBNAME%TYPE;
```

```
TCOPY NUMBER;
```

```
CURSOR cbooks IS
```

```
SELECT BT.ISBN,BT.BTNAME,P.PUBNAME,COUNT(*) FROM BOOKTITLE  
BT,BOOKCOPY BC,LOAN L,Publisher P WHERE BT.ISBN=BC.ISBN AND BC.BCID=L.BCID  
AND P.PUBID=BT.PUBID AND L.DATEBACK IS NOT NULL GROUP BY  
BT.ISBN,BT.BTNAME,P.PUBNAME;
```

```
BEGIN
```

```
OPEN cbooks;
```

```
LOOP
```

```
FETCH cbooks INTO TISBN ,TTITLE,PNAME , TCOPY;
```

```
DBMS_OUTPUT.put_line(TISBN) ;
```

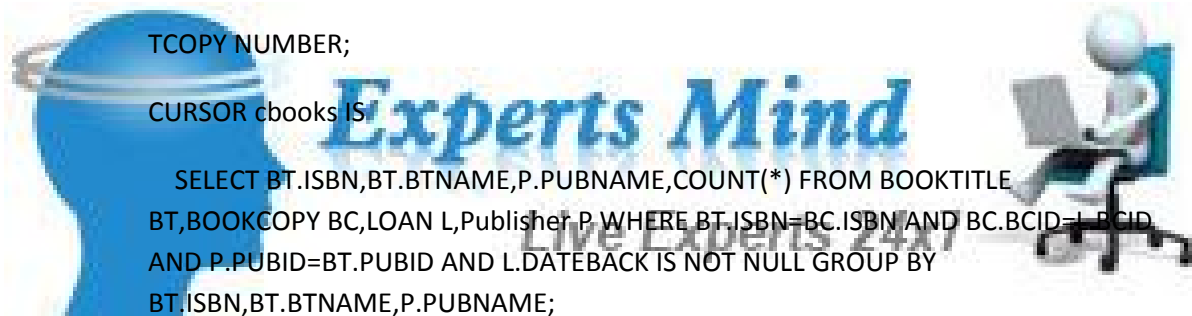
```
DBMS_OUTPUT.put_line(TTITLE);
```

```
DBMS_OUTPUT.put_line(TCOPY );
```

```
END LOOP;
```

```
CLOSE cbooks;
```

```
END;
```



3. Write a stored function called getPublisher. This function takes as input an ISBN for a book and returns the publisher name.

Call the function from within an SQL statement to display the publisher for book title 'The Twits'

**Query:**

```
create or replace function "GETPUBLISHER"
```

```
(tisbn in VARCHAR2 default '5')
```

```
return VARCHAR2
```

```
is
```

```
pname varchar2(20);
```

```
begin
```

```
    SELECT P.PUBNAME INTO pname FROM PUBLISHER P,BOOKTITLE BT WHERE
```

```
    BT.PUBID=P.PUBID AND BT.ISBN=tisbn;
```

```
    RETURN pname;
```

```
end;
```

**Output:**



```
select getPublisher(7) from dual;
```

**Results** Explain Describe Saved SQL History

GETPUBLISHER(7)

Puffin

1 rows returned in 0.00 seconds

[CSV Export](#)



# Experts Mind

Live Experts 24x7



4. Create a trigger 'checkRecommendedAge' to enforce the (harsh) constraint that children are only allowed to borrow books deemed suitable for their age. The trigger fires whenever there is a new loan and outputs an error message whenever an attempt is made to borrow a book where the borrower's actual age is not within the lower to upper age range for the book. Ignore the fact that some of the data already in the database may violate the constraint.

**Query:**

```
create or replace trigger "CHECKRECOMMENDEDAGE"
```

```
BEFORE
```

```
delete or insert on "LOAN"
```

```
for each row
```

```
DECLARE
```

```
    bage number;
```

```
    blow number;
```

```
    bhigh number;
```

```
BEGIN
```

```
    SELECT BORAGE INTO bage FROM BORROWER WHERE BORID=:NEW.BORID;
```

```
    SELECT BT.AGELOWER INTO blow  FROM BOOKTITLE BT,BOOKCOPY BC WHERE  
BT.ISBN=BC.ISBN AND BC.BCID=:NEW.BCID;
```

```
    SELECT BT.AGEUPPER INTO bhigh FROM BOOKTITLE BT,BOOKCOPY BC WHERE  
BT.ISBN=BC.ISBN AND BC.BCID=:NEW.BCID;
```



IF bage>=blow AND bage<=bhigh THEN

:NEW.BORID:=:NEW.BORID;

:NEW.BCID:=:NEW.BCID;

:NEW.DATEOUT:=:NEW.DATEOUT;

:NEW.DATEDUE:=:NEW.DATEDUE;

:NEW.DATEBACK:=:NEW.DATEBACK;

END IF;

EXCEPTION

WHEN OTHERS THEN

raise\_application\_error(-20004, 'Borrow age is not within the low and high limit');

end;

