CENG 352 Database Management Systems
Spring 2014-2015 – Programming Assignment
Due Date: Mar. 27, 2015

1 Objectives

This assignment aims to help you get familiar with Oracle database, constraints, triggers, stored procedures, PL-SQL, cursors and PHP. First you are required to create the general structure of the tables. Next, you are required to create procedures that allow retrieving, manipulating records on these tables. Finally, you are required to design a basic Internet application using PHP. In what follows, we explain each of these stages in detail.

2 Directives

Part 1: Creating Tables

For this part of the assignment, we will give you the structure of the tables. You are responsible to create tables, add constraints, triggers, etc. to these created tables according to given specifications.

Connection: Successfully connect the DBMS using **Sql Developer** with your Oracle account information. To connect to the DBMS, you need a username, password, database and IP address of the host where Oracle resides. A few details about parameters are given below;

- Username: Your student ID. Ex. e1887454
- Password: We will send to your mail address.
- Hostname: 144.122.71.31
- Port: 8085
- SID: XE

You will find more details about connecting to Oracle in recitation slides.

Table Structure:
Create table using the names and data types given below;

Users: userid INTEGER, email VARCHAR2(30), name VARCHAR2(30), surname VARCHAR2(30)
BookInfo: isbn VARCHAR2(13), title VARCHAR2(100), pubYear CHAR(4) , quantity INTEGER
BorrowedBy: userid INTEGER, isbn VARCHAR2 (13), issueDate DATE, dayReturned DATE

Create appropriate database structures according to the specifications given below:

Specifications:

1. Email should end with "edu.tr". You should create appropriate constraint/constraints to allow only such emails.

2. Name and surname cannot be empty.
3. Title of the book should be unique within table.
4. If we want to delete a user, BorrowedBy information about that user should also be deleted.
5. Don't allow to delete a book record if there exists a user who borrowed one of that book.
6. When a new record is inserted on BorrowedBy, issueDate defaults to current date. Below is an example;

```
insert into borrowedBy(userid, ISBN)    values(1,'9780306406157');
```

USERID	ISBN	ISSUEDATE	DAYRETURNED
1	9780306406157	03-MAR-15	(null)

7. ISBN length has changed from 10 digits to 13 digits since 2007. We don't want users to insert 10-digit ISBN. If a database user wants to insert a 10 digit ISBN record to BookInfo table, the ISBN should be converted to 13-digit form and inserted with length 13.
   Conversion formula from 10 digit ISBN to 13 digit is as follows;
   a) remove 1 digit from the end.
   b) add 978 to the front
   c) add check bit to the end.
      Formally, the check digit calculation is as follows:

$$x_{13} = (10 - (x_1 + 3x_2 + x_3 + 3x_4 + \cdots + x_{11} + 3x_{12}) \bmod 10) \bmod 10.$$

   where x_{13} denotes the check bit(and also right most bit) of 13-digit ISBN (x_1 x_2 x_3 ... x_{12} x_{13}). In the formula, each digit from left to right, is *alternately* multiplied by **1** or **3**. Then we calculate sum of those products. At the end, modulo 10 of the sum is subtracted from 10.
   For example, the ISBN-13 check digit of 978-0-306-40615-CHK is calculated as follows:

= (10 – (9×1 + 7×3 + 8×1 + 0×3 + 3×1 + 0×3 + 6×1 + 4×3 + 0×1 + 6×3 + 1×1 + 5×3) mod 10) mod 10
= (10 - (9 + 21 + 8 + 0 + 3 + 0 + 6 + 12 + 0 + 18 + 1 + 15) mod 10) mod 10
= (10 – (93) mod 10) mod 10
=(10-3) mod 10
= 7

   You should create appropriate trigger on BookInfo table for converting 10-digit ISBNs. Please note that while inserting 13-digit we don't need a conversion.

   Below are some examples:

```
insert into bookInfo(ISBN, title , pubYear , quantity ) values('0306406151','t1','2005',1);
```

ISBN	TITLE	PUBYEAR	QUANTITY
9780306406157	t1	2005	1

```
insert into bookInfo(ISBN, title , pubYear , quantity ) values('9781449324452','t2','2006',2);
```

ISBN	TITLE	PUBYEAR	QUANTITY
9780306406157	t1	2005	1
9781449324452	t2	2006	2

insert into bookInfo(ISBN, title , pubYear , quantity) values('**0596514107**','t3','2007',3);

ISBN	TITLE	PUBYEAR	QUANTITY
9780306406157	t1	2005	1
9781449324452	t2	2006	2
9780596514105	t3	2007	3

Table population: Insert the following records into the newly created relations.

Users			
userid	**email**	**name**	**surname**
1	yourID@metu.edu.tr	YourName	YourSurname

BookInfo			
ISBN	Title	pubYear	Quantity
9781449324452	OraclePL/SQL Programming	2014	2
9781430263005	Expert Oracle Database Architecture	2015	1
9780071494458	Oracle Database 11g PL/SQL Programming	2008	3

Implementation: You should create appropriate database structures on Oracle db. The file you are required to upload is defined under "**What to submit?**" title of this document.

Part 2: Creating Stored Procedures

At this part of the assignment, we will give you an oracle *package specification* with empty package body. You are responsible to fill the procedures at the package body.

At the recitation, we will provide examples which you may like to use for this part. A brief explanation for using Sql Developer, PL-SQL language and compiling packages will also be provided at recitation.

Definitions of Stored Procedures:

1. We assume that userids serve as logins and emails serve as passwords. Check user_id and email of a user. Result is 'T' if user id and email values match a row at users table, otherwise result is 'F'.

```
PROCEDURE login(
    p_user_id IN INTEGER,
    p_email   IN VARCHAR2,
    p_RESULT OUT CHAR);
```

2. If user wants to borrow a book and number of the books is greater than zero, then decrease the quantity of the book by one and insert a record to borrowedBy (Please

note that dayReturned field should be null). If successful, return result as 'T', otherwise result is 'F'.

```
PROCEDURE borrow_book(
    p_user_id IN INTEGER,
    p_ISBN    IN VARCHAR2,
    p_RESULT OUT CHAR);
```

3. While returning a book, if the book hasn't been returned, increase the quantity of the book by one and update the return date as current date.

```
PROCEDURE return_book(
    p_user_id   IN INTEGER,
    p_ISBN      IN VARCHAR2,
    p_issuedate IN DATE);
```

4. List ISBN,title, publication year and issue date of books that a user borrowed but were **not** returned back.

```
PROCEDURE books_borrowed(
    p_user_id IN INTEGER ,
    p_books OUT SYS_REFCURSOR);
```

5. List ISBN,title, publication year, issue date and return date of books that a user returned.

```
PROCEDURE books_returned(
    p_user_id IN INTEGER ,
    p_books OUT SYS_REFCURSOR);
```

6. List book information that starts with the text(p_title) in its title column. Search should be case insensitive.

```
PROCEDURE search_books(
    p_title IN VARCHAR2,
    p_books OUT SYS_REFCURSOR);
```

7. List book information that are published after p_year1 and before p_year2.

```
PROCEDURE search_books(
    p_year1 IN VARCHAR2,
    p_year2 IN VARCHAR2,
    p_books  OUT SYS_REFCURSOR)
```

Implementation: For this part of the homework, we will provide you an oracle package called *ceng352* in two sql files (part2_spec.sql, part2_body.sql). First sql file includes package spec. and the second file includes the package body. You should implement the pl-sql procedures defined above in the package body.

Part 3: A simple Web based application using PHP

For this part of the assignment, you are required to design an internet application on top of the simple database you created above.

For adding, updating, deleting or retrieving rows at your web pages, you are only allowed to use stored procedures you created at part2. You are **not** allowed to access the tables directly.
The website should include the following pages:

1. *Index page:* All users start at a common login page, where each user is expected to enter his/her userid and email(as password). If the login operation fails, an appropriate error message should be given. In addition, give an error if one or both of the input fields left blank and "login" button is clicked (i.e., use a simple Java Script code to check). Use the procedure **ceng352.login** to check user & password.
2. *List Books page*: On the top of this page, list books that were borrowed but not returned back. Next to each of these books, display a link, namely "return", so that the user can return the book back. On the bottom, list books that were returned. On the bottom also provide a link for "Search books" page. Use the procedure **ceng352.books_borrowed ceng352.books_returned** and **ceng352.return_book**
3. *Search books page*: Provide a text field where user enters a text to search in book titles in the library. Also provide 2 other text fields where user enters two 4-digit years. Finally provide "search" buttons to allow users to search for the given conditions. Next to each of results, display a link, namely "borrow", so that the user can borrow the book. Use **ceng352.search_books and ceng352.borrow_book** procedures.
4. *Logging out:* When user clicks "logout" button, don't allow unsigned user to borrow or return books.

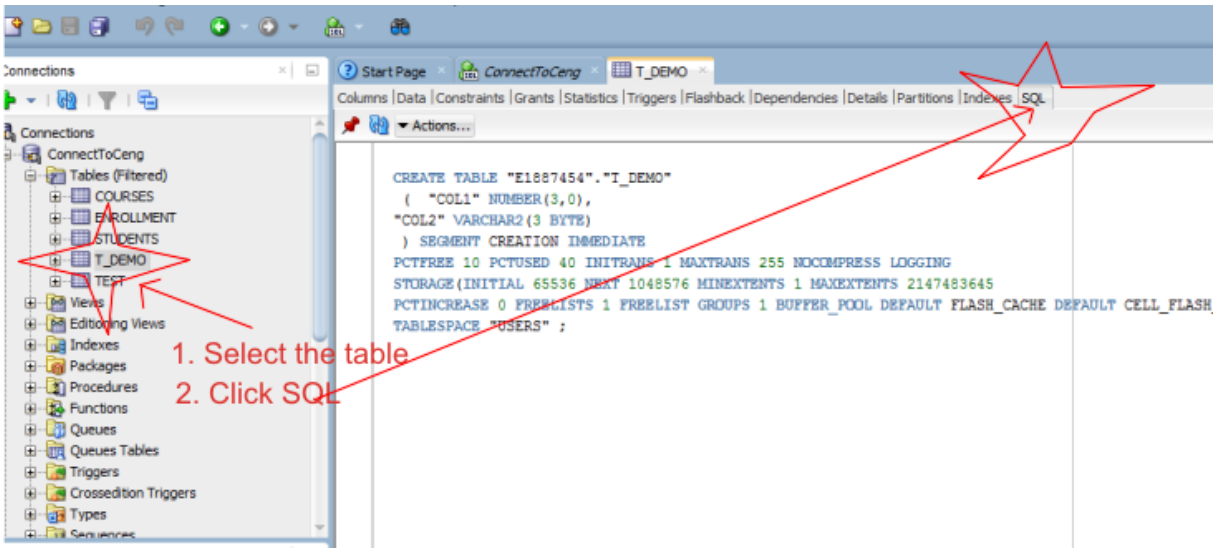
You may need to keep track of the current user id through pages.

Implementation: You should use PHP to implement the application logic (except the Java Script codes used to check blank form fields at the client side). Your application should connect to your Oracle database prepared in the first part of this assignment.
You are free to develop your PHP application at your local machine (install any packages that can be necessary) and finally test under your public html directory, as it should work there! (if the public_html directory does not exist, create it at your home directory and set its r-w-x permissions appropriately). (After testing, you must remove your files from your public html folder to prevent unauthorized users who may want to have a glance in your solution!)
You will copy and submit this folder as described below, and we will copy it under our own public_html and execute there. To allow this, please use relative links in your web pages (i.e., if you include links to an absolute page http://www.ceng.metu.edu.tr/~e123456/ index.php, it would not work under my public html folder. If this happens, you won't get any grade from this part of the assignment).

3 What to submit?

You will submit a WinRAR file including tree folders, part1 and part2 and part3. Folders must be exactly named as *surname_name_part1* and *surname_name_part2 and surname_name_part3*. The RAR file must be named as *surname_name.* Each folder should include the following:

- Part1 folder: In this folder, provide the source code in a single sql file(part1.sql). We will basically open your file, run your SQL statements in this file to create the structures. Grading will be based on the appropriate structures (triggers, constraints) at your tables.
  HINT: You can copy the required sql statements from sql developer. Select the table from the tree on the left. Then Click SQL. Copy the generated script for 3 tables and combine them into a single sql file called part1.sql.



- Part2 folder: You should include a single file, part2_body.sql. This file includes the **body** of oracle package ceng352. Grading will be based on the **execution** of the procedures in this package.

- Part3 folder: This folder should include all your PHP files. We will copy them to our public_html and then execute there. Grading will be based on the **execution** of your code.