Computer Science Department  
Michigan State University  
CSE480 Database Systems, Spring 2003  
Lab Week #9: PL/SQL

YOUR NAME:

OBJECTIVES:
This lab has the following two objectives related to the project:

- PHP code for web query interfaces  
  (this part is a continuation of lab 8).
- Introduction to PL/SQL.

(30 minutes)

PHP Codes for Web Query Interfaces:

- Two sample php files (multi.php and proc.php) are provided in the lab. They are commented so that you read and understand them. Ask TA if you have any questions about the codes. These codes will help you in project 2.

To use these codes, you first replace the Oracle username/password in the code provided, with your own.

- **Multi.php**: It displays an interface for query and processing of various course descriptions from tables. In the top textbox in the interface, next to the **Query button**, you can enter a part of the course number such as **CSE**, then click **Query button**. Only course numbers containing CSE will be returned in the table below the textbox. Run the program cgi-bin/multi.php and see the results.

Now you answer the following:

1. Display all the courses in the table courseDescription by using the **Query button**.
2. Why do we need the variable $queryCno in the multi.php?
3. Why do we initialize $queryCno with " 1=1 "?
4. Which php file is called by clicking the **Query button**? Which parameter is being passed and what values are being passed for this parameter.
After the courses are displayed as a result of clicking the **Query button**, you can choose some of these courses for further details, by checking the buttons to the right of the course numbers and then clicking the **Proc button** at the bottom. Now you answer the following questions.

1. Which type of input is used in the php code to implement multi-choice function? How is the result of the multi-choice stored?

2. Which php file is called by clicking **Proc button**?

3. What is the name of the parameter that is being passed and what values are being passed?

This is the end of part I of your lab. Now you will work on PL/SQL of Oracle

(60 minutes)

PL/SQL is Oracle's Procedural Language extension to SQL. This provides more efficient implementation and allows using stored procedures and packages. In this lab you will only use a stored procedure that is already written for you. You will then modify them to implement another procedure which is some what similar.

Run PL/SQL directly in SQL PLUS. To see the results, you need to run the following statement first:

```sql
SET SERVEROUTPUT ON
```

1. The following PL/SQL implements a cursor and displays the first 3 highest students' grades in course '00001' by using the built-in procedure `put_line` of the package `dbms_output`:

A text file, `lab.txt`, is provided for you to cut and paste the following codes

```sql
DECLARE
    CURSOR c1 is
        SELECT sname, grade FROM Student s, Taken t
            where s.sid=t.sid and SeqId='00001'
        ORDER BY grade DESC;
    my_sname VARCHAR2(40);
    my_grade NUMBER(2,1);
BEGIN
    OPEN c1;
    FOR i IN 1..3 LOOP
```
FETCH c1 INTO my_sname, my_grade;
EXIT WHEN c1%NOTFOUND; /* in case the number requested */
   /* is more than the total */
   /* number of enrolled students */
   /* display the result */
dbms_output.put_line('Name: ' || my_sname || ' ' || 'Grade: ' || my_grade);
END LOOP;
CLOSE c1;
END;
/

(a) Run the above program in sqlPlus.
(b) Modify the above to output the average of the top three grades.

2. Following stored procedure implements the same function as above except that the seqid can be input from the user. To output the result, we need the following temporary table.

create table my_tmp(
   sname varchar2(40),
   grade number(2,1)
);

The PL/SQL stored proc:

create or replace procedure firstThree(my_seqid in varchar2) as
CURSOR c1 is
   SELECT sname, grade FROM Student s, Taken t
   where s.sid=t.sid and SeqId=my_seqid
   ORDER BY grade DESC;
   my_sname VARCHAR2(40);
   my_grade NUMBER(2,1);
BEGIN
   /* clear the my_tmp table */
   delete from my_tmp;
   commit;

   OPEN c1;
   FOR i IN 1..3 LOOP

FETCH c1 INTO my_sname, my_grade;
EXIT WHEN c1%NOTFOUND;  /* in case the number requested */
    /* is more than the total */
    /* number of enrolled students */
/* display the result */
dbms_output.put_line('Name: ' || my_sname || ' ' || 'Grade: ' || my_grade);
/* put into temporary table */
insert into my_tmp values(my_sname, my_grade);
commit;
END LOOP;
CLOSE c1;
END;
/
(a) Run the above stored proc in sqlPlus as follows:

begin
    firstThree('00001');
end;
/

The result can be seen in table my_tmp

(b) Modify the above stored procedure to output the average of the top three grades.

3. The following implements the same function as above in php. A php
   named as "call_sp.php" calls the stored procedure in Oracle. Note
   that the php provides a host bind variable :my_seqid as the input
   parameter.

#!/opt/bin/php

<?php
putenv("ORACLE_HOME=/opt/oracle/9.2.0");
putenv("ORACLE_SID=cse480");
putenv("TWO_TASK=cse480");

$connection = OCILogon ("cse480", "cse480");
if ($connection == false){
    echo OCIError($connection)."<BR>";
}
exit;
}

// the statement to call the stored procedure
$query = "begin
    firstThree(:my_seqid);
end;";

$cursor = OCIParse ($connection, $query);
if ($cursor == false){
    echo OCIError($cursor)."<BR>";
    exit;
}

// after parsing, bind the variable with Oracle variable.
OCIBindByName($cursor, ":my_seqid", &$my_seqid, 5);
$my_seqid = '00001';

$result = OCIExecute ($cursor);
if ($result == false){
    echo OCIError($cursor)."<BR>";
    exit;
}

OCICcommit ($connection);
OCILogoff ($connection);
?>

(a) Run the above program.
(b) Modify the stored procedure and the above php code to output
the average of the top three grades.
   You need to modify the header of firstThree to
create or replace procedure firstThree(my_seqid IN varchar2, my_average
OUT number)
to include an output parameter my_average which outputs the
average grade. Note that keyword OUT is used instead of key-\nword IN. You also need to add the following statement into the
above php code to bind the output parameter my_average to a
new php variable $myAverage.
OCIBindByName($cursor, ":my_average", &$myAverage, -1);
 echo the result in php.