CSC 352 / 452: Database Programming

assignment #5 (60 Points)

**CSC 352/452-501: Due on Sunday, 8/16/2015 at 11:59PM**

**CSC 352/452-510: Due on Monday, 8/17/2015 at 11:59PM**

Unless prior arrangements are made, homework turned in late but within 24 hours of the due time will be graded at 75% credit, homework turned in between 24 and 48 hours will be graded at 50% credit, and homework turned in later than 48 hours will not be accepted.

**Please note that only TEXT files will be accepted. All other file types (e.g., DOC, DOCX, RTF, PDF, JPG, or ZIP) will be rejected.**

* Please read the assignment carefully. Using different tables (names, columns, or data types) or procedure headers (names, parameters, or data types) will receive a grade of zero.
* Please review your assignment file before submitting it to make sure you have the correct one. It is your responsibility to ensure that you upload the correct assignment file.

**1) (0 Point)**

The HR\_USERS table stores information about user accounts.

HR\_USERS(**USER\_ID**,

PASSWORD,

LAST\_SUCCESSFUL\_LOGIN\_TIME,

LAST\_FAILED\_LOGIN\_TIME,

FAILED\_PASSWORD\_ATTEMPT\_COUNT,

IS\_LOCKED\_OUT,

LAST\_LOCKED\_OUT\_TIME,

LAST\_PASSWORD\_CHANGED\_TIME);

USER\_ID: Primary Key

IS\_LOCKED\_OUT: 'Y' – the user account is locked out

'N' – the user account is not locked out

FAILED\_PASSWORD\_ATTEMPT\_COUNT: the number of consecutive failed password attempts

Create and populate the HR\_USERS table as described below.

CREATE TABLE hr\_users

( USER\_ID NUMBER PRIMARY KEY,

PASSWORD VARCHAR2(30) NOT NULL,

LAST\_SUCCESSFUL\_LOGIN\_TIME DATE,

LAST\_FAILED\_LOGIN\_TIME DATE,

FAILED\_PASSWORD\_ATTEMPT\_COUNT NUMBER,

IS\_LOCKED\_OUT CHAR NOT NULL

CONSTRAINT ck\_hr\_users CHECK (IS\_LOCKED\_OUT IN ('Y','N')),

LAST\_LOCKED\_OUT\_TIME DATE,

LAST\_PASSWORD\_CHANGED\_TIME DATE);

/

INSERT INTO hr\_users

VALUES(82001, '99CpsBTKpN',TO\_DATE('02-APR-2015 14:05:08', 'DD-MON-YYYY HH24:MI:SS'),

NULL, 0, 'N', NULL, TO\_DATE('02-DEC-2014 13:05:08', 'DD-MON-YYYY HH24:MI:SS'));

INSERT INTO hr\_users

VALUES(82002, 'ZWNWnQJT90', TO\_DATE('02-DEC-2014 16:15:01', 'DD-MON-YYYY HH24:MI:SS'),

NULL, 0, 'N', NULL, TO\_DATE('02-NOV-2014 11:11:18', 'DD-MON-YYYY HH24:MI:SS'));

INSERT INTO hr\_users

VALUES(82003, 'gc88Wmvpx8', TO\_DATE('01-APR-2015 19:15:08', 'DD-MON-YYYY HH24:MI:SS'),

TO\_DATE('05-MAY-2015 21:45:18', 'DD-MON-YYYY HH24:MI:SS'), 1, 'N', NULL, TO\_DATE('30-JAN-2015 23:01:01', 'DD-MON-YYYY HH24:MI:SS'));

INSERT INTO hr\_users

VALUES(82004, 'KcxweSYg555', TO\_DATE('03-JAN-2015 14:12:33', 'DD-MON-YYYY HH24:MI:SS'),

TO\_DATE('06-MAY-2015 09:12:22', 'DD-MON-YYYY HH24:MI:SS'), 5, 'Y', TO\_DATE('06-MAY-2015 09:12:22', 'DD-MON-YYYY HH24:MI:SS'), NULL);

INSERT INTO hr\_users

VALUES(82005, 'CDYe44BBXd1', TO\_DATE('22-MAR-2015 05:22:18', 'DD-MON-YYYY HH24:MI:SS'),

NULL, 0, 'N', NULL, NULL);

INSERT INTO hr\_users

VALUES(82006, 'vhSDHMDg666', TO\_DATE('07-FEB-2015 04:00:08', 'DD-MON-YYYY HH24:MI:SS'),

NULL, 0, 'N', NULL, TO\_DATE('01-FEB-2015 04:35:01', 'DD-MON-YYYY HH24:MI:SS'));

COMMIT;

**2) (30 Points)**

Based on the HR\_USERS table created in (1), create a procedure for validating user login.

The procedure header is

CREATE OR REPLACE PROCEDURE user\_login

(

p\_user\_id NUMBER,

p\_password VARCHAR2,

p\_ret\_code OUT NUMBER

)

(You cannot change the procedure header. You will get a zero point if a different procedure header is used. Submitting more than one procedure will receive 0 points. No DBMS\_OUTPUT statement is needed in the procedure.)

The logical steps are as follows;

1. The value of p\_user\_id is not in the USER\_ID column of the HR\_USERS table. You update the p\_ret\_code parameter:

* p\_ret\_code <== 0

1. The value of p\_user\_id is in the USER\_ID column of the HR\_USERS table, but the corresponding user account is locked out (IS\_LOCKED\_OUT = 'Y'). You update the p\_ret\_code parameter:

* p\_ret\_code <== -10

(When a user account is locked out, only an administrator can manually unlock it.)

1. The value of p\_user\_id is in the USER\_ID column of the HR\_USERS table and the value of p\_password matches the corresponding password in the table. You update the corresponding row in the HR\_USERS table and the p\_ret\_code parameter:

* LAST\_SUCCESSFUL\_LOGIN\_TIME <== SYSDATE
* FAILED\_PASSWORD\_ATTEMPT\_COUNT <== 0
* LAST\_FAILED\_LOGIN\_TIME <== NULL
* p\_ret\_code <== 1

1. The value of p\_user\_id is in the USER\_ID column of the HR\_USERS table, but the value of p\_password does not match the corresponding password in the table. You update the corresponding row in the HR\_USERS table and the p\_ret\_code parameter:

* FAILED\_PASSWORD\_ATTEMPT\_COUNT <==

FAILED\_PASSWORD\_ATTEMPT\_COUNT + 1

* LAST\_FAILED\_LOGIN\_TIME <== SYSDATE
* Case 1: FAILED\_PASSWORD\_ATTEMPT\_COUNT != 5
* p\_ret\_code <== -1
* Case 2: FAILED\_PASSWORD\_ATTEMPT\_COUNT = 5
* IS\_LOCKED\_OUT <== 'Y'
* LAST\_LOCKED\_OUT\_TIME <== SYSDATE
* p\_ret\_code <== -10

You need to test your procedure with different parameters in a **PL/SQL block**.

**3) (30 Points)**

Based on the HR\_USERS table created in (1), create a procedure to change the password for a given user ID.

* A password is case-sensitive.
* A password must be between 10 and 30 characters in length.
* A password must include at least one uppercase alphabetic character (A-Z), one lowercase alphabetic character (a-z), and one numeric character (0-9).
* You need to update the **PASSWORD** and **LAST\_PASSWORD\_CHANGED\_TIME** columns if the password can be reset. (Oracle SYSDATE function returns the current date and time.)

The procedure header is

CREATE OR REPLACE PROCEDURE reset\_password

(

p\_user\_id NUMBER,

p\_current\_password VARCHAR2,

p\_new\_password VARCHAR2,

p\_ret\_code OUT NUMBER

)

(You cannot change the procedure header. You will get a zero point if a different procedure header is used. Submitting more than one procedure will receive 0 points. No DBMS\_OUTPUT statement is needed in the procedure.)

The logical steps are as follows;

1. The value of p\_user\_id is not in the USER\_ID column of the HR\_USERS table. The password cannot be reset. You update the p\_ret\_code parameter:

* p\_ret\_code <== -10

1. The current password is not correct. The password cannot be reset. You update the p\_ret\_code parameter:

* p\_ret\_code <== -20

1. The new password is the same as the current password. The password cannot be reset. You update the p\_ret\_code parameter:

* p\_ret\_code <== -30

1. The new password is too long or too short. The password cannot be reset. You update the p\_ret\_code parameter:

* p\_ret\_code <== -40

1. The new password does not include an uppercase alphabetic character. The password cannot be reset. You update the p\_ret\_code parameter:

* p\_ret\_code <== -50

1. The new password does not include a lowercase alphabetic character. The password cannot be reset. You update the p\_ret\_code parameter:

* p\_ret\_code <== -60

1. The new password does not include a numeric character. The password cannot be reset. You update the p\_ret\_code parameter:

* p\_ret\_code <== -70

1. The password can be reset. You update the corresponding row in the HR\_USERS table and the p\_ret\_code parameter:

* PASSWORD <== p\_new\_password
* LAST\_PASSWORD\_CHANGED\_TIME <== SYSDATE
* p\_ret\_code <== 1

Hint: To test a string for alphabetic and numeric characters, you could use the REGEXP\_LIKE function.

*(REGEXP\_LIKE is similar to the LIKE condition, except REGEXP\_LIKE performs regular expression matching instead of the simple pattern matching performed by LIKE. This condition evaluates strings using characters as defined by the input character set.)*

Examples:

1) SELECT COUNT(\*) INTO v\_n FROM DUAL WHERE REGEXP\_LIKE(v\_1, '[A-Z]');

v\_n = 0: v\_1 does not include an uppercase alphabetic character.

2) SELECT COUNT(\*) INTO v\_n FROM DUAL WHERE REGEXP\_LIKE(v\_1, '[a-z]');

v\_n = 0: v\_1 does not include a lowercase alphabetic character.

3) SELECT COUNT(\*) INTO v\_n FROM DUAL WHERE REGEXP\_LIKE(v\_1, '[0-9]');

v\_n = 0: v\_1 does not include a numeric character.

You need to test your procedure with different parameters in a **PL/SQL block**.

**Please submit a text file containing all the source codes to D2L before or on due date.**