**TERM – WINTER 2023 **

| **Course & Section Code:** | **DBS501** |
| --- | --- |
| **Course Name:** | **Procedure Using Oracles PL/SQL** |
| **Course Start & End Dates:** | Click or tap here to enter text. |
| **Q&A / Virtual Office Hour:** | **6:30 PM – 9:30 PM** |
| **Instructor Name & Email:** | **Ersan Cam: ersan.cam@senecacollege.ca** |

SENECA COLLEGE DBS 501 Course - Assigment#2

**General Instructions :**

You can use any Web based front end programming tool which can connect to our database in Lab or your own database

* Clean Coding, Interaction with Database (Connectivity, Login page etc ) , GUI features (Drop down menu , list box, Radio boxes etc if needed) and actual Presentation skills will be evaluated during presentation time.
* Expectation is that everyone in the team shares responsibility. If one person writes some stored procedure and the other one writes function , one or two can work on GUI same as in real life environment.
* **Presentation will be on last day.**
* Make power point presentation and do share with instructor in advance. Instructor will open presentation for you to present in front of class or you can plug your own laptop.
* If you want to use your own laptop , it is your responsibility to test connectivity of Projector otherwise do send all files and presentation to instructor and do DEMO from Instructor’s computer.

**Marking guidelines**

**Back-end programming (Up to 70 points)**

All the procedure ,function, triggers must be cleanly compiled with no errors. Tested on SQL Developer

You will be collecting points by doing successful Demo and test in front of Instructor and audience

**2-Front end GUI effectiveness (0 up to30 pnts)**

How user friendly is your GUI? Is it just basic & quickly written-basic but functional GUI or efficient and colorful/fancy GUI ?

Presentation skills (Eye contact with audience - Voice control - body gesture etc (Do some research on basic presentation skills )

**Try to create a HR application Menu like below or use your imagination. You are completely free to design any shape of form for the menus**

|  |  |  |
| --- | --- | --- |
| **Employee Main Menu** | **Jobs Main Menu** | **Departments Main Menu** |

**Please Note that all the table names here (EMLOYEES , JOBS, DEPARTMENTS) must refer to HR\_EMPLOYEES, HR\_JOBS, HR\_DEPARTMENTS naming convention as your Ryerson database has this HR\_ prefix**

**Employee Hiring / Terminating / Update Employee Info Screen**

**Task1-1**

Create a Procedure called Employee\_hire\_sp which accepts 9 parameters (p\_first\_name, p\_last\_name, p\_email, p\_salary, p\_hire\_date, p\_phone, p\_job\_id , p\_manager\_id, p\_department\_id )

Once these values come as parameter then procedure BEGIN END should have simple INSERT statement

BEGIN

**INSERT INTO** employees (EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL,

PHONE\_NUMBER, HIRE\_DATE,JOB\_ID, SALARY, MANAGER\_ID, DEPARTMENT\_ID )

**VALUES** (EMPLOYEES\_SEQ.NEXTVAL, p\_first\_name, p\_last\_name ,,,,,,,,,,,,)

COMMIT;

END;

**TEST:** You can test your procedure in SQL developer to make sure it works and insert new employee record to table

**Task1-2 - GUI work**

Create a Page (Form) for Employee Hiring activities UNDER Employee Main Menu

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Employee Hiring Form**   |  |  |  | | --- | --- | --- | | **First Name :** | **Last Name:** | **Email**: |  |  |  |  | | --- | --- | --- | | **Phone:** | HireDate: SYSDATE (as default) | **Salary** : |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | **JOB\_ID** | **JOB\_TITLE** | | FI\_ACCOUNT | ACCOUNTANT | | SA\_REP | Sales Representative | | SA\_MAN | Sales Manager | |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Manager :**   |  |  |  | | --- | --- | --- | | **Employee\_id** | **First\_Name** | **Last\_Name** | | 103 | Alexander | Hunold | | 108 | Nancy | Greenberg | |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | Department :   |  |  | | --- | --- | | **Department\_Id** | **DEPARTMENT \_NAME** | | 10 | Administration | | 20 | Marketing | |   Cancel  Hire |

In the form above you can see entry edit boxes such as phone or salary. They are simple edit boxes and when user enter these values they will be passed to **employee\_hire\_sp** procedure.

But for **Manager** box, or **Job** box or **Department** box , you need to make Dropdown lookup menu . When user click on Department box dropdown menu list would come from database showing only department ID and department Name (Select from Departments table). if End user picks Marketing department then department\_id 20 would be send to procedure as INPUT parameter. *Look for example down there for dropdown. Search title* ***HINT for Dropdown Menu item***

Once you collect all this information from front end and user click on at the bottom, black border button called Hiring then send them all as INPUT parameter to your procedure and make sure employee is hired by listing it at the bottom

Blue border box is for cancel option (optional)

**Task 1-3**

Under **Employee Main Menu** , create a link to list and Update all or any one the employee record.

Make a form to list all employees like excel grid format on the page and then let end user edit SALARY or PHONE or EMAIL information not any other info just any of these 3.

If any one of this info or all of these 3 info got change by end user then accept the new value and send to database as UPDATE COMMAND and replace old value with new one in employee table.

*Sample :*

*UPDATE EMPLOYEE SET SALARY= <new salary> WHERE employee\_id= ID from the form selected -highlighted record line*

It is up to you to create a procedure for this or simply use front end GUI capability to do list of all employee and edit in place on the form. Either way is OK for me.

**Note:**

Some web designers choose to have one single / same GUI doing all these different functionalities.

One for Entering new Employee information

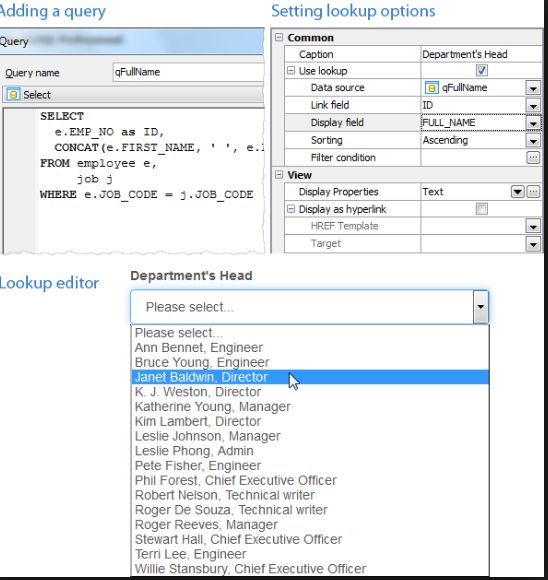
One for Updating Employee details (except Employee ID)

And also in the same page you should list all the current employee as list at the bottom part of page

It is Up to you how you want to design the page(s) . You can create several different GUI pages doing each action described above or have one page and do multiple tasks in it.

**HINT for Dropdown Menu item**

As example, here below you can see one Dropdown lookup table box



**Task 2: JOB related activities (Jobs menu)**

**Task 2-1** **Identify JOB description**

Under the JOBS MAIN menu create a link called **Identify JOB description**

In this link or (page or form) idea is End user will enter JOB \_ID let’s say SA\_REP and form will bring accurate **JOB\_Description** from **Jobs** table

There will be one Edit box to accept JOB\_ID from end user

To do that, you can either use FUNCTION or Front end SQL Select

A ) create a Function called **get\_job** (accept one INPUT parameter as **job\_id** varchar2) and RETURN VARCHAR2

Function will get JOB\_ID as INPUT and search this in WHERE condition from JOBS table and bring JOB\_TITLE info back to page (Form)

**Or**

B) you don’t need to create function, you can simply use FRONT END capability to write simple SQL Select statement and return JOB\_Description product to the database.

**Task 2-2 - Allow end user to change JOB\_DESCRIPTIPON**

In this page (form) you will bring and list all the JOB\_ID, JOB\_TITLE , MIN\_SALARY and MAX\_SALARY columns from database JOBS table to web page same as Excel Grid and then you can allow user to update (edit ) any values other than JOB\_ID.

-Let’s say USER picked FI\_ACCOUNT record and edit description as FIN- ACCOUNTANT or MIN\_SALARY 8500 then get these changes and update IN the database JOBS table permanently

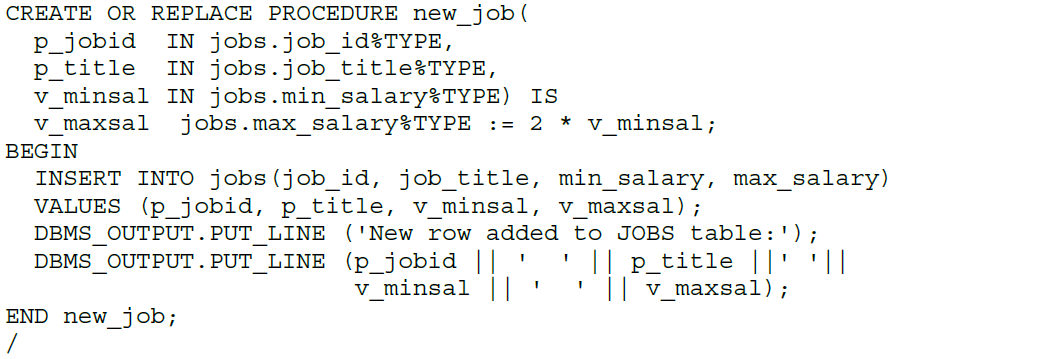
-To do that ,again you can create Function or Store procedure to do this UPDATE command. You can accept any edit box new value as parameter and call Function or procedure and ask them to do it.

-Or simply use front end edit/update capability on one record. UP TO YOU

-In each method please DO NOT FORGET TO ADD COMMIT; command after UPDATE command.

**Task 2-3 Create a New JOB**

Create this procedure in the database at first and test this procedure in SQL developer. Show your test work to us during the demo (I am sure you would have same procedure already as part of in class practice)



**Task 2-3 GUI work**

Build a GUI web page (Form) under JOBS MAIN Menu

This link or page (form) will allow 4 different edit boxes like below.

**Job\_id** (enter as AS\_MAN )

**Title** (enter as ASSISTANT MANAGER)

**Minimum Salary** (Enter as 3500)

**Maximum Salary** (Enter as 5500)

Collect all these info and let End user CLICK a button called **CREATE JOB** and this button will execute (call) **new\_job** procedure. Don’t forget to add COMMIT inside procedure to make sure records entered in database permanently. You can even return a message to end user stating “A new job has been created” --this last part is optional

**Task 3 – Creating a Trigger and also a Store Procedure to verify and check any Job’s minimum and Maximum Salary range if they are in acceptable limit**

For example there is this record in JOBS table

**Job\_id job\_title min\_salary max\_salary**

AC\_MGR Accounting Manager 8200 16000

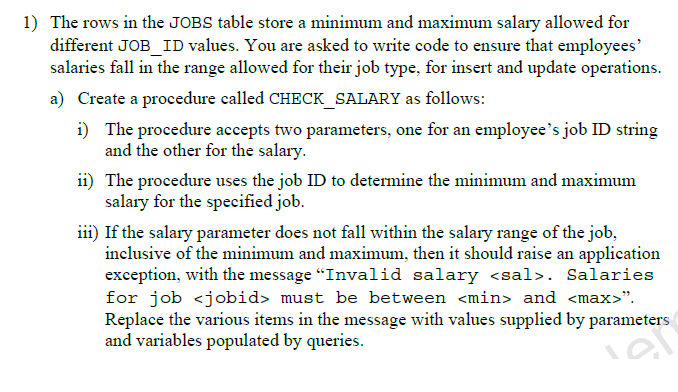
Let’s assume if we hire a new employee as AC\_MGR and assign salary as 5000 or 17000 system will give warning to us.

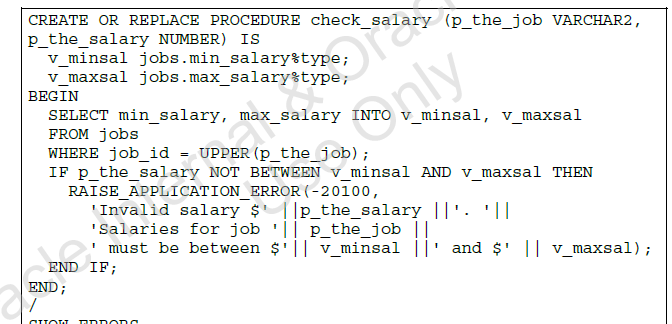
Or if there is existing employee working as AC\_MGR and salary is 9000 and we want to UPDATE employee salary to 17000 then again system should give as warning and not allow us /end user to change.

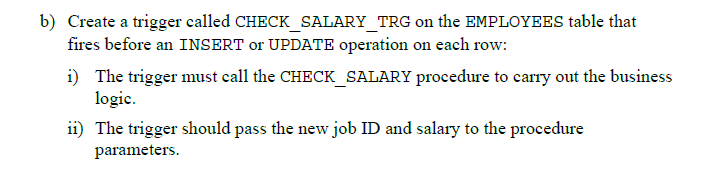
**How?**

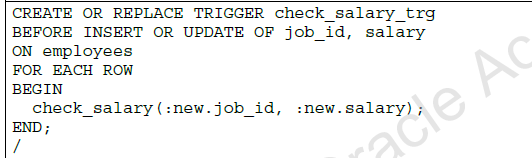
We will have Trigger attached to EMPLOYEES table and that trigger will be fired when someday request to INSERT new record to employee table or UPDATE salary on the same table (Two conditions). When trigger fired after these two actions then Trigger will call another Stored procedure to accept JOB\_ID and salary of employee AS INPUT and the check IF salary of new employee falls inside JOB\_ID min and max salary range. If yes accept it if not warning.

**HINTS :**









**Test Cases:**

**Test Case 1:** Test whether this trigger and Stored procedure control mechanism works or not

Try to Hire a new Employee from Employee Main menu Hiring page

First name: Elenor Last name: Beh department:30 salary :1000

Job\_ID: SA\_REP email: abc@abc manager\_id =145

Explain what happened and show this in your GUI and /or SQL developer

**Test case 2:**

UPDATE the salary of Employee ID 115 to $2000

You can use main Employee Menu Employee INFO update feature to do these test cases. If you can not then use SQAL developer to run simple UPDATE command.

Explain what happened and why? show us warning message on GUI and /or also in SQL developer

**Test case 3:**

CHANGE (UPDATE ) employee\_ID 115’s JOB\_ID to HR\_

Explain what happened and why? show us warning message on GUI and /or also in SQL developer

As example look for below commands for testing. But as Instructor I prefer you do these below activities thru GUI screens

