Coursework for Module 5FTC2090

Description:

In this coursework, you are required to write SQL queries, PL/SQL blocks, functions, and triggers to satisfy the needs of Questions 1-15.

Database Schema

The questions in this coursework should run on the HR database schema, so you must define all tables of the HR schema and insert sample data to fill in the HR database. You will be provided with two SQL scripts to do the previous steps. The first script – namely 'hr_create.sql'- is the file you need to run to create the HR database tables and define all attributes of these tables; the second script – namely 'hr_populate.sql'- is the file you need to run to insert a sample dataset into the HR database tables and define the needed constraints that should apply on theses tables and their attributes.

Submission Information

You must collect all the deliverables for all questions into one SQL script, named as 'SISUserID_FName_LNAME_CW_5FTC2090.sql'; where you need to replace the SISUserID with your university ID, FName by your first name, LNAME by your last name. Upload the submission file onto Canvas before the deadline.

Marking Scheme [50 marks]

Question Number	Marking Criteria	Mark
1	Correct syntax and semantics of the view	1
	Proof of correct compilation of the view	0.5
	Test of view by displaying view results	0.5
	Matching the results with the question's view requirements	1
2	Correct syntax of the view	1
	Proof of correct compilation of the view	0.5
	Test of view by displaying view results	0.5
	Matching the results with the question's view requirements	1
3	Correct syntax and semantics of the view	1

	Proof of correct compilation of the view	0.5
	Test of view by displaying view results	0.5
	Matching the results with the question’s view requirements	1
4	Correct syntax and semantics of the view	1
	Proof of correct compilation of the view	0.5
	Test of view by displaying view results	0.5
	Matching the results with the question’s view requirements	1
5	Correct syntax and semantics of the view	1
	Proof of correct compilation of the view	0.5
	Test of view by displaying the view results	0.5
	Matching the results with the question’s view requirements	1
6	Correct syntax and semantics of the PL/SQL block	1
	Proof of correct compilation of the PL/SQL block	1
	Correct output matching the needed data for display	0.5
7	Correct syntax and semantics of the function	1
	Proof of correct compilation of the function	1
	Correct use of the formula given in the function	0.5
8	Correct syntax and semantics of the SQL clause	1
	Proof of correct return output based on the called function	1
9	Correct syntax and semantics of the SQL clause	2
	Proof of returning the required results	1
10	Correct syntax and semantics of the function	2
	Correct return results	1
11	Correct syntax and semantics PL/SQL block	2
	Proof of calling the predefined function returning the number of departments	1
	Correct output results	1
12	Correct syntax and semantics of the trigger	2
	Proof of correct compilation of the trigger	1
	Correct test of the trigger action	1
13	Correct syntax and semantics of the trigger	2
	Proof of correct compilation of the trigger	1
	Correct test of the trigger action	1
14	Correct explanation and identification of the error cause	1
	Syntax and semantics of the block after handling the error case	3
	Test of the block after capturing the error and correct/descriptive error message to user	1
15	Correct explanation and identification of the error cause	1
	Syntax and semantics of the block after handling the error case	3
	Test of the block after capturing the error and correct/descriptive error message to user	1

QUESTION 1 - Creating simple view:

Create a simple VIEW called EMPJOB_V which displays all data in JOBS.

QUESTION 2 Creating view with alias:

Create a simple view called EMPDATE_V which displays all employees from EMPLOYEES.

Your view should display EMPLOYEE_ID, LAST_NAME, HIRE_DATE and JOB_ID order by hiring date.

Make sure you rename HIRE_DATE to STARTING_ON

QUESTON 3 VIEW WITH CONDITION

Create a view named DEPTNOHEAD_V which displays all departments which have no manager.

QUESTION 4 VIEW FROM JOINT TABLE

Create View DEPTCOUNT_V that lists the number of employees in each department.

The view should display the department id, the department name and the number of employees working for the department.

Order in alphabetic order of the department name.

QUESTION 5 READ ONLY VIEW

Create Read only view EMPREAD_V which displays all records from EMPLOYEES but do not allow user to add or update or delete records in EMPLOYEES.

QUESION 6 PLSQL BLOCK WITH CURSOR

Write a PL/SQL block which uses cursor to list all of employees and their hire date who have been recruited after year '2015'.

Your DBMS output should have the employees' last name, salary and hire date.

Question 7 CREATE PLSQL FUNCTION [simple function]

Write a PL/SQL function named 'increaseSalary' which takes original salary and percentage increment as argument and return the new salary.

Note : new salary = original salary * (1+percentageIncrement)

Question 8 CALLING THE FUNCTION WITH SQL CLAUSE

Call the function 'increaseSalary()' defined in question 7 to find the new salary of the item.

Let the original salary = 2000 and increment = 20%

Hint: call the function in SQL clause

Question 9 CALLING THE FUNCTION WITH SQL CLAUSE

Write a query that displays the original salary and increased salary of all employees who have been hired before 2017, and their salary is less than 3000.

Your query should display last name, hiring date, original salary as oldSalary and increased salary as newSalary

HINT:: call increaseSalary() in the SQL Clause

Question 10 CREATE PLSQL FUNCTION [simple function]

Write a simple PL/SQL function 'getDepartmentsNum' that returns total number of departments.

Question 11 CALLING THE PL/SQL FUNCTION AS NORMAL FUNCTION

Write a PLSQL block that calls the function ' getDepartmentsNum ()' as a normal function.

Your PL/SQL block should display number of departments in the DBMS Output

Note:

use DBMS_OUTPUT.put_line for display.

Call the function getDepartmentsNum () inside PL/SQL block not in SQL clause.

QUESTION 12 CREATING A SIMPLE TRIGGER

Write a trigger named 'displaySalaryChange' that displays old salary, new salary, and salary difference in DBMS Output whenever a salary of some employee is updated.

QUESTION 13 CONDITIONAL TRIGGER [type 1]

Write a trigger called stopSoaringSalary that overwrite the change in employee’s increase in salary if new salary is increased by more than 20%

Your trigger must display 'Business ERROR !! you cannot increase employees’ salary by more than 20% !!' and alter the change.

QUESTION 14 PLSQL EXCEPTION [type 1]

```
DECLARE
  JobRecord  JOBS%ROWTYPE;
BEGIN
  SELECT * INTO JobRecord FROM JOBS WHERE JOBS.JOB_TITLE='Sales Agent';
  dbms_output.put_line('Job Sales Title :' || JobRecord.JOB_Title);
END;
```

PLSQL block above throws error as:

```
Error report -
ORA-01403: no data found
ORA-06512: at line 4
01403. 00000 -  "no data found"
*Cause:    No data was found from the objects.
*Action:   There was no data from the objects which may be due to end of fetch.
```

Identify the type of the error and rewrite the PL/SQL block such that the error is trapped.

Display appropriate error message for the error.

QUESTION 15 PLSQL EXCEPTION [type 2]

```
DECLARE
  JobRecord  JOBS%ROWTYPE;
BEGIN
  SELECT * INTO JobRecord FROM JOBS WHERE JOBS.JOB_TITLE like 'Sales%';
  dbms_output.put_line('Job Sales Title :' || JobRecord.JOB_Title);
END;
```

PLSQL block above throws error as:

```
Error report -
ORA-01422: exact fetch returns more than requested number of rows
ORA-06512: at line 4
01422. 00000 -  "exact fetch returns more than requested number of rows"
*Cause:    The number specified in exact fetch is less than the rows returned.
*Action:   Rewrite the query or change number of rows requested
```

Identify the type of the error and rewrite the PL/SQL block such that the error is trapped.

Display appropriate error message for the error.