1. Write a PL/SQL block to calculate total salary with commision of an employee whose ID is 7756.
2. Write a PL/SQL program to arrange the number of two variable in such a way that the small number will store in num\_small variable and large number will store in num\_large variable.
3. Write a PL/SQL program to check whether a number is even or odd.
4. Write a PL/SQL program to check whether a date falls on weekend i.e. SATURDAY or SUNDAY.
5. Write a PL/SQL procedure to calculate the incentive achieved according to the specific sale limit.

For sale over 50,000 : 5%

For sale over 40,000 : 3%

For sale over 30,000 : 2%

For sale over 20,000 : 1%

For sale below 20,000 : 0%

1. Write a PL/SQL program to count the number of employees in a particular department and check whether this department have any vacancies or not. Total occupancy of any department is 5.
2. Write a PL/SQL program to display the description(highest and lowest salary) against a grade( using if then and CASE both).
3. Write a PL/SQL program to check whether a given character is letter or digit.
4. Write a PL/SQL program to convert a temperature in scale Fahrenheit to Celsius and vice versa.
5. Write a PL/SQL program to display which day is a specific date.
6. Write a program in PL/SQL to update the salary of a specifc employee by 8% if the salary exceeds the mid range of the salary against this job and update up to mid range if the salary is less than the mid range of the salary, and display a suitable message.
7. Write a program in PL/SQL to print 1st n numbers.
8. Write a program in PL/SQL to print 1st n numbers with a difference of 3 and starting from 1.
9. Write a program in PL/SQL to print the prime numbers between 1 to 50.
10. Write a program in PL/SQL to check whether a number is prime or not using goto statement with for loop.
11. Write a program in PL/SQL to find the number of rows effected on any DML operation, by the use of SQL%ROWCOUNT attributes of an implicit cursor.
12. Write a program in PL/SQL to show the uses of SQL%FOUND to determine if a DELETE statement affected any rows.
13. Write a program in PL/SQL to show the uses of SQL%NOTFOUND to determine if a UPDATE statement affected any rows.
14. Write a program in PL/SQL to display a table based detail information for the employee of ID 149 from the employees table.
15. Write a program in PL/SQL to display a cursor based detail information of employees from employees table.
16. Write a program in PL/SQL to retriev the records from the employees table and display them using cursors.
17. Write a program in PL/SQL to declare a record datatype with same datatype of tables using %TYPE attribute.
18. Write a program in PL/SQL to create an implicit cursor with for loop to retrieve emp details along with department details of all clerks having salary less than 1000.
19. Write a program in PL/SQL to create an explicit cursor with for loop to display product details with price information based on user supplied product id.
20. Create a PL/SQL block to increase salary of employees in the department 50 using WHERE CURRENT OF clause.
21. Write a program in PL/SQL to FETCH details of purchases and total amount of purchases by each customer with the uses of nested cursor.
22. Write a program in PL/SQL to print a list of managers and the name of the departments.
23. Write a PL/SQL block to display the name of the department and their costliest employee. Displays department name and costliest employee.
24. Write a PL/SQL block to display the last name, first name and overpaid amount by using parameters.
25. Write a PL/SQL block to display the number of employees by month. Print number of employees by month.
26. Write a function incr\_salary, which takes two parameters, employee no and the percentage by which the salary needs to be raised and returns the new salary
27. Write a procedure Get\_On\_Hand that has one in and one out parameter. This procedure should take in equipment number and send the quantity on hand for this equipment through the out parameter.
28. Write a procedure called emp\_dept which has one IN and one OUT parameter. This procedure should take in employee no and send the name of the department he is working for by getting it from the department table.
29. Write a trigger, which checks the quantity on hand in the equipment table for every update and displays an error message if the quantity on hand falls below 2.
30. Write a trigger, which fires before the update of employee table for each row and checks if the new salary is below 20000, if it is then it raises an error.
31. Write a trigger on the department table which fires after deleting a row and stores the information of this department into a new table ‘xdept’ which has the same structure as that of the department and an additional field called deldate which stores the date the department is deleted.
32. Write a package which has one procedure and one function in it. The procedure should take an employees no and return his name and department number. The function is to take employees no and a percentage by which his salary is to be raised and raise his salary accordingly and return the new salary.
33. Create a procedure to find out the NO of the dept manager for an employee. In other words, write a procedure that takes an employee’s NO as the input variable, and return the NO of his or her dept manager using the same variable. Return ‘-1’ if no dept manager can be found.
34. The employees’ salary is to be classified as following:

Class A: <= 30000

Class B: > 30000 and <= 60000

Class C: > 60000

Create a function to return the class of a salary for an employee, given his or her NO.

1. Assuming each department is granted only 5000 dollar for purchasing PCs, create a function to check if a new row can be added to the equipment table. In other words, create a function that takes a *dept\_no*, *eqp\_qty\_on\_hand* and *eqp\_value* as the parameters, and returns a 0 if the total value (after adding the row) does not exceed the allowance, and return –1 if it does.
2. Create a trigger to ensure that no department manager or employee starts on Saturday and Sunday.
3. Create a trigger to limit 5 assignments to one employee.
4. Create two tables, LT\_ASGN\_HISTORY and ASGN\_HISTORY, both with the following columns:

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Datatype | Size | Comments |
| ***work\_empno*** | CHAR | 9 | **Primary Key.** Employee social security number. Also **Composite Foreign Key** link to *assignment* table |
| ***work\_pro\_number*** | NUMBER | 2 | **Primary Key.** Project number. Also **Composite Foreign Key** link to *assignment* table. |
| *work\_hours\_completed* | NUMBER | (5,1) | Number of hours an employee has completed on a project. |
| *date\_last\_updated* | DATE |  | **Primary Key**. Date when this row is changed. |

Create a trigger that does the following:

When the *work\_hours\_planned* becomes 0, (not NULL, assuming that a *work\_hours\_planned* with NULL values means it is not assigned yet, while *work\_hours\_planned* = 0 means the assignment has been completed),

* 1. If the total work hours exceed 100, insert a new record to LT\_ASGN\_HISTORY (long term assignment history).
  2. Else insert a new record to ASGN\_HISTORY.

1. Create a procedure that calculate the total work hours planned for an employee given his or her no.