

## DBMS (PL/SQL) Test -8

1. Write PL/SQL blocks to retrieve the first\_name from the employees table.

The screenshot shows the Oracle SQL Developer interface. At the top, there's a toolbar with 'Language' set to 'SQL' and 'Rows' set to '10'. Below the toolbar, the SQL editor contains the following PL/SQL code:

```
2  Solution:
3  DECLARE
4      FIRST_NAME  employee11.FIRST_NAME%TYPE;
5  BEGIN
6      select FIRST_NAME into FIRST_NAME from employee11;
7      DBMS_OUTPUT.PUT_LINE('First Name = ' || FIRST_NAME);
8  END;
```

Below the editor, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, showing the output: 'First Name = lakshman' and 'Statement processed.'

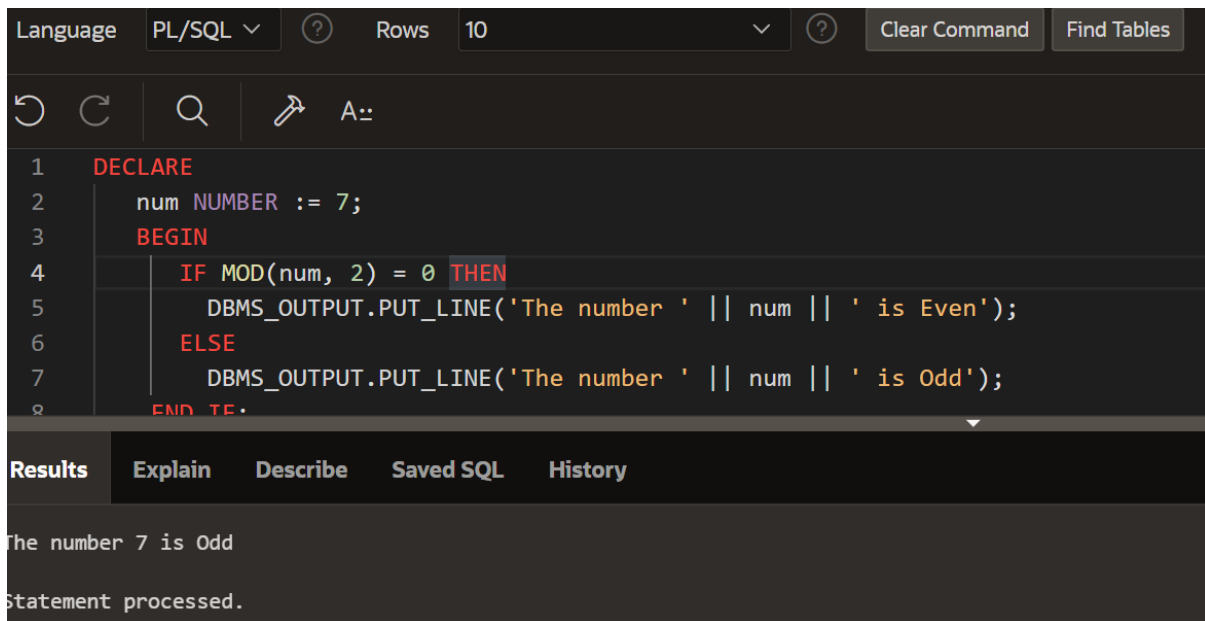
2. Write a PL/SQL block to display Hello World.

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there's a navigation bar with 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. Below the navigation bar, there's a search bar and a 'Schema' dropdown set to 'IN\_B861\_SQL\_S03\_ADMIN'. The 'SQL Commands' tab is active, showing the following PL/SQL code:

```
1  declare
2      r varchar2(50);
3  begin
4      r:='Hello World!';
5      DBMS_output.put_line('This is output ' || r);
6  end;
```

Below the editor, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, showing the output: 'This is output Hello World!' and 'Statement processed.'

3. Write a PL/SQL program to check whether a number is even or odd.

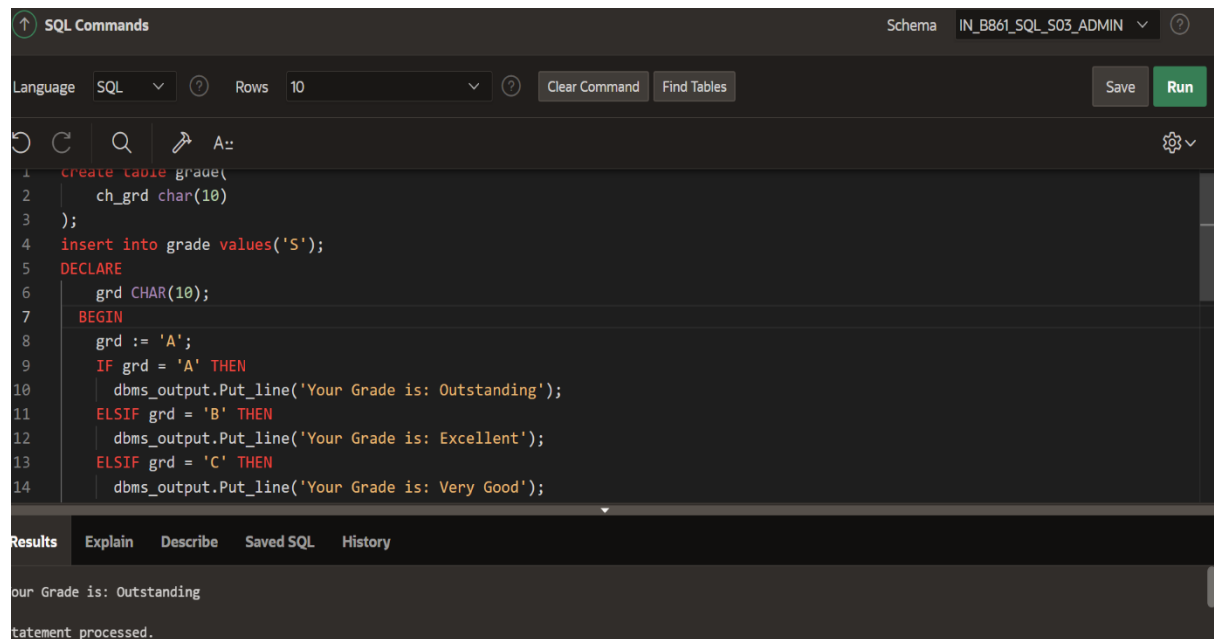


The screenshot shows a PL/SQL IDE interface. At the top, there's a toolbar with 'Language' set to 'PL/SQL', 'Rows' set to '10', and buttons for 'Clear Command' and 'Find Tables'. Below the toolbar is a code editor with the following PL/SQL code:

```
1 DECLARE
2     num NUMBER := 7;
3 BEGIN
4     IF MOD(num, 2) = 0 THEN
5         DBMS_OUTPUT.PUT_LINE('The number ' || num || ' is Even');
6     ELSE
7         DBMS_OUTPUT.PUT_LINE('The number ' || num || ' is Odd');
8     END IF;
```

Below the code editor, there's a 'Results' tab selected, showing the output: 'The number 7 is Odd' and 'Statement processed.'

4. Write a PL/SQL program to display the description against a grade.



The screenshot shows a SQL IDE interface. At the top, there's a toolbar with 'Language' set to 'SQL', 'Rows' set to '10', and buttons for 'Clear Command', 'Find Tables', 'Save', and 'Run'. Below the toolbar is a code editor with the following SQL and PL/SQL code:

```
1 create table grade(
2     ch_grd char(10)
3 );
4 insert into grade values('S');
5 DECLARE
6     grd CHAR(10);
7 BEGIN
8     grd := 'A';
9     IF grd = 'A' THEN
10         dbms_output.put_line('Your Grade is: Outstanding');
11     ELSIF grd = 'B' THEN
12         dbms_output.put_line('Your Grade is: Excellent');
13     ELSIF grd = 'C' THEN
14         dbms_output.put_line('Your Grade is: Very Good');
```

Below the code editor, there's a 'Results' tab selected, showing the output: 'Your Grade is: Outstanding' and 'Statement processed.'

5. Write a PL/SQL program to convert a temperature in scale Fahrenheit to Celsius and vice versa.

The screenshot shows the 'SQL Commands' window in Oracle APEX. The language is set to SQL, and the schema is 'IN\_B861\_SQL\_S03\_ADMIN'. The PL/SQL block is as follows:

```

1 DECLARE
2     temp1      NUMBER := 20;
3     t_scale    CHAR := 'C';
4     new_temp   NUMBER;
5     new_scale  CHAR;
6 BEGIN
7     IF t_scale != 'C'
8     AND
9     t_scale != 'F' THEN
10        dbms_output.put_line ('The scale you input is not a valid scale');
11        new_temp := 0;
12        new_scale := 'C';
13    ELSE
14        IF t_scale = 'C' THEN

```

The results pane shows the output of the execution:

```

The new temperature in scale F is: 68
statement processed.

```

6. Write a PL/SQL block to concatenate the first name and last name of each employee in the employees table and display the full name.

The screenshot shows the 'SQL Commands' window in Oracle APEX. The language is set to SQL, and the schema is 'IN\_B861\_SQL\_S03\_ADMIN'. The PL/SQL block is as follows:

```

1 DECLARE
2     v_full_name varchar(50);
3 BEGIN
4     for employees in (select first_name, last_name FROM employees) LOOP
5         v_full_name := employees.first_name || ' ' || employees.last_name;
6         DBMS_output.put_line('FULL NAME: ' || v_full_name);
7     END LOOP;
8 END;

```

The results pane shows the output of the execution:

```

FULL NAME: lakshman gumma1
FULL NAME: lakshman gumma2
FULL NAME: lakshman gumma3

```

7. Write a PL/SQL block that replaces all occurrences of the substring 'SA\_MAN' with 'Sales Manager' in the job titles of employees in the employees table. Display the updated job titles.

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The 'SQL Commands' tab is active, showing a PL/SQL program. The program declares a variable `v_job_id` of type `employees.job_id%TYPE`, loops through all employees, and updates the job ID to 'Sales Manager' for those with the job title 'SA\_MAN'. The interface also shows a 'Schema' dropdown set to 'IN\_B861\_SQL\_S03\_ADMIN' and a 'Run' button.

```

1 DECLARE
2   v_job_id employees.job_id%TYPE;
3 BEGIN
4   FOR emp IN (SELECT job_id FROM employees) LOOP
5     IF v_job_id = 'SA_MAN' THEN
6       v_job_id := 'Sales Manager';
7     END IF;
8     DBMS_OUTPUT.PUT_LINE('Updated Job ID: ' || v_job_id);
9   END LOOP;
10 END;
11

```

Below the code editor, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. A message states: 'Enter SQL statement or PL/SQL command and click Run to see the results.'

8. Write a PL/SQL program to display the employee IDs, names, job titles, hire dates, and salaries of all employees.

The screenshot shows the Oracle APEX SQL Workshop interface with a PL/SQL program designed to fetch and display employee data. The program declares variables for employee ID, first name, last name, job title, hire date, and salary. It then uses a cursor named `emp_cursor` to select all columns from the `employees` table. The interface shows the 'History' tab selected, and a 'Find' search bar is visible at the bottom.

```

1 DECLARE
2   -- Declare variables to hold the employee data
3   v_employee_id  employees.employee_id%TYPE;
4   v_first_name   employees.first_name%TYPE;
5   v_last_name    employees.last_name%TYPE;
6   v_job_title    jobs.job_title%TYPE;
7   v_hire_date    employees.hire_date%TYPE;
8   v_salary       employees.salary%TYPE;
9
10  -- Cursor to fetch the employee data
11  CURSOR emp_cursor IS
12    SELECT e.employee_id, e.first_name, e.last_name, j.job_title, e.hire_date, e.salary
13    FROM employees e
14    JOIN iobs i ON e.iob id = i.iob id

```

9. Write a PL/SQL program to display the names of all countries.

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The 'SQL Commands' tab is active, showing a PL/SQL program. The program declares a variable `v_country_name` of type `country.country_name`, creates a cursor `c_countries` to select `country_name` from the `country` table, and then loops through the cursor to output each country name. The interface includes a 'Run' button and a 'Find' search bar.

```
1 DECLARE
2   v_country_name country.country_name%TYPE;
3   CURSOR c_countries IS SELECT country_name FROM country;
4 BEGIN
5   OPEN c_countries;
6   FETCH c_countries INTO v_country_name;
7   WHILE c_countries%FOUND LOOP
8     DBMS_OUTPUT.PUT_LINE(v_country_name);
9     FETCH c_countries INTO v_country_name;
10  END LOOP;
11  CLOSE c_countries;
12 END;
```

10. Write a PL/SQL program to display the job titles of all employees. Return a heading of job title.

The screenshot shows the Oracle APEX SQL Workshop interface with a new PL/SQL program. The program declares variables `v_location_id` and `v_city` of type `locations.location_id` and `locations.city` respectively. It creates a cursor `c_locations` to select `location_id` and `city` from the `locations` table. The program then outputs a heading 'Location ID | City' followed by a separator line, and then loops through the cursor to output each location ID and city. The interface includes a 'Run' button and a 'Find' search bar.

```
1 DECLARE
2   v_location_id locations.location_id%TYPE;
3   v_city locations.city%TYPE;
4   CURSOR c_locations IS SELECT location_id, city FROM locations;
5 BEGIN
6   DBMS_OUTPUT.PUT_LINE('Location ID | City');
7   DBMS_OUTPUT.PUT_LINE('-----');
8   OPEN c_locations;
9   FETCH c_locations INTO v_location_id, v_city;
10  WHILE c_locations%FOUND LOOP
11    DBMS_OUTPUT.PUT_LINE(v_location_id || ' | ' || v_city);
12    FETCH c_locations INTO v_location_id, v_city;
13  END LOOP;
14  CLOSE c_locations;
```