GENERAL

pl sql interview questions and answers for 2 experience

Published: September 3, 2026 | Index ID: #-

PL/SQL Interview Questions and Answers for 2 Year Experienced Developers

Oracle's PL/SQL (Procedural Language/Structured Query Language) remains one of the most widely used database languages in enterprise environments. Whether you're applying for a junior developer role, a database analyst position, or a mid level PL/SQL engineer, interviewers will expect you to demonstrate a solid grasp of core concepts, practical coding skills, and an awareness of performance considerations.

Below is a curated set of interview questions tailored for candidates with roughly two years of experience. Each question is followed by a concise, yet thorough answer that highlights both the theoretical foundation and the practical application. The content is organized into thematic sections, making it easy for you to study and for interviewers to evaluate your depth of knowledge.

1. Core PL/SQL Concepts

An anonymous block consists of three optional sections: DECLARE (where you declare variables, cursors, and types), BEGIN (the executable section), and EXCEPTION (error handling). The block ends with END;. Example:

```
DECLARE
  v_count NUMBER;
BEGIN
  SELECT COUNT(*) INTO v_count FROM employees;
  DBMS_OUTPUT.PUT_LINE("Count: " || v_count);
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE("No data found.");
END;
```

What is a package in PL/SQL?

A package is a schema object that groups related procedures, functions, variables, cursors, and types. It has two parts: a specification (public interface) and a body (implementation). Packages promote modularity, encapsulation, and can improve performance by reusing compiled code.

Procedures perform actions and may return multiple values via OUT parameters; they are ideal for operations that modify data. Functions return a single value and can be used in SQL statements. If you need to return a value to the caller, use a function; otherwise, a procedure is appropriate.

2. Cursors and Exception Handling

```
CURSOR emp_cur IS SELECT employee_id, salary FROM employees WHERE department_id = 10;
BEGIN
  OPEN emp_cur;
  LOOP
```

```
    FETCH emp_cur INTO v_id, v_salary;
    EXIT WHEN emp_cur%NOTFOUND;
    -- Process each row
  END LOOP;
  CLOSE emp_cur;
END;
```

Explain the significance of the %FOUND, %NOTFOUND, %NOTFOUND, and %ROWCOUNT attributes.

These attributes provide status information about the cursor after a fetch operation:

- %FOUND – True if the last fetch returned a row.
- %NOTFOUND – True if the last fetch did not return a row.
- %ROWCOUNT – Number of rows processed so far.
- %ISOPEN – Indicates whether the cursor is open.

Implicit cursors are automatically created by Oracle for each SQL statement that returns a single row. They are referenced by the SQL and SQL%ROWCOUNT attributes. Example:

```
SELECT COUNT(*) INTO v_cnt FROM employees;
DBMS_OUTPUT.PUT_LINE(SQL%ROWCOUNT); -- returns 1
```

Describe a typical exception handling strategy in PL/SQL.

Start with a generic WHEN OTHERS clause to catch unforeseen errors. For known error conditions, use named exceptions or predefined ones (e.g., NO_DATA_FOUND, TOO_MANY_ROWS). Example:

```
BEGIN
  -- risky operation
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE("No record found.");
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE("Error: " || SQLERRM);
END;
```

3. Packages, Procedures, and Functions

```
CREATE OR REPLACE PACKAGE emp_pkg IS
      PROCEDURE update_salary(p_emp_id IN NUMBER, p_new_salary IN NUMBER);
      FUNCTION get_bonus(p_emp_id IN NUMBER) RETURN NUMBER;
    END emp_pkg;
```

Then create the body with implementations:

```
CREATE OR REPLACE PACKAGE BODY emp_pkg IS
      PROCEDURE update_salary(p_emp_id IN NUMBER, p_new_salary IN NUMBER) IS
      BEGIN
```

```
    UPDATE employees SET salary = p_new_salary WHERE employee_id = p_emp_id;
  END update_salary;

  FUNCTION get_bonus(p_emp_id IN NUMBER) RETURN NUMBER IS
    v_bonus NUMBER;
  BEGIN
    SELECT salary * 0.1 INTO v_bonus FROM employees WHERE employee_id = p_emp_id;
    RETURN v_bonus;
  END get_bonus;
END emp_pkg;
```

What is a package body that is not specified?

If you create a package body without a corresponding specification, Oracle treats it as an anonymous body, and the procedures/functions are private. This is useful for encapsulating internal logic that should not be exposed publicly.

IN – Input only; the caller cannot modify the value. OUT – Output only; the procedure populates the variable. IN OUT – Both input and output; the caller supplies an initial value and receives a modified value.

Use the EXEC command or the anonymous block syntax. Example:

```
EXEC emp_pkg.update_salary(101, 75000);
```

4. Triggers and Functions

BEFORE triggers fire before the triggering statement executes, allowing you to validate or modify data. AFTER triggers fire after the statement, useful for logging or updating related tables.

A row-level trigger (FOR EACH ROW) fires once for each row affected by the DML statement. A statement-level trigger fires once per statement regardless of the number of rows affected.

Use a package variable or a global temporary table to track whether the trigger has already executed. Alternatively, use the PRAGMA AUTONOMOUS_TRANSACTION to isolate the trigger's work.

5. Performance Tuning and Optimization

It returns a detailed stack trace of the error, which is invaluable for debugging complex PL/SQL code. Example usage:

```
DBMS_OUTPUT.PUT_LINE(DBMS_UTILITY.FORMAT_ERROR_BACKTRACE);
```

How do you identify and resolve a "TOO_MANY_ROWS" exception?

Ensure the query returns a single row, perhaps by adding a ROWNUM = 1 condition or using aggregate functions. Use SELECT INTO only when you are certain of a single result.

It creates a transaction that is independent of the main transaction. Useful for logging or auditing within triggers or procedures where you want changes to persist even if the main transaction rolls back.

6. SQL vs. PL/SQL

Bind variables allow you to reuse the same SQL statement with different values without reparsing. They improve performance and reduce memory usage.

SELECT INTO fetches a row into PL/SQL variables, while INSERT SELECT inserts the result set directly into a table.

7. Data Types, Variables, and Collections

Baca Juga (Artikel Terkait)

- **advanced engineering electromagnetics solution manual edition hardcover by balanis constantine a published by wiley**
  URL: http://www.denniswildberger.com/advanced-engineering-electromagnetics-solution-manual-edition-hardcover-by-balanis-constantine-a-published-by-wiley.pdf
- **advanced engineering electromagnetics solutions manual**
  URL: http://www.denniswildberger.com/advanced-engineering-electromagnetics-solutions-manual.pdf
- **advanced engineering electromagnetics wiley 1989 grading**
  URL: http://www.denniswildberger.com/advanced-engineering-electromagnetics-wiley-1989-grading.pdf
- **advanced engineering fluid mechanics by biswas**
  URL: http://www.denniswildberger.com/advanced-engineering-fluid-mechanics-by-biswas.pdf

Tags: #interview #questions #answers #experience