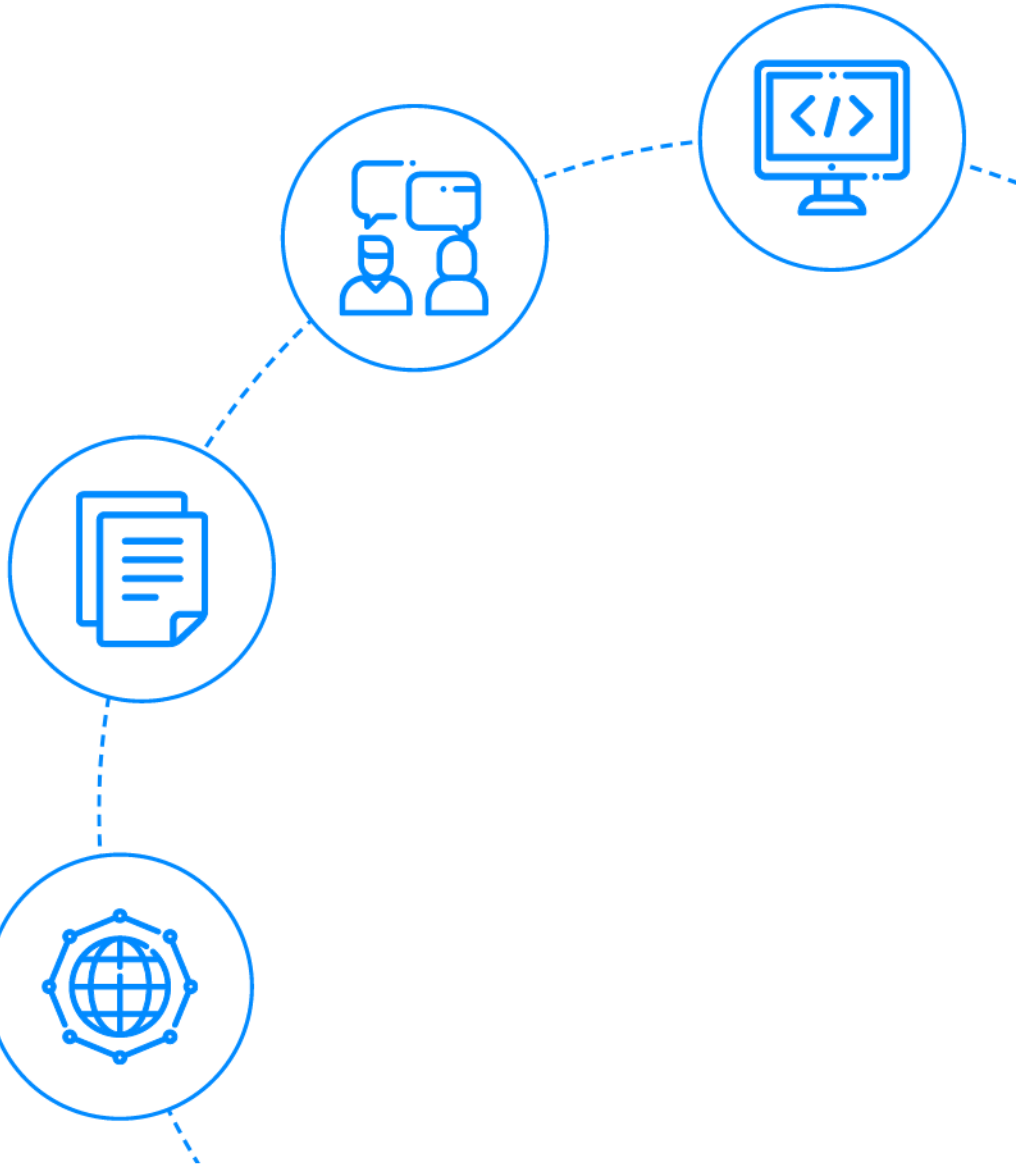
InterviewBit

PL SQL Interview Questions



To view the live version of the page, click here.

Contents

PL/SQL Basic Interview Questions

PL/SQL Intermediate Interview Questions

PL/SQL Intermediate Interview Questions (.....Continued)

19. What are the virtual tables available during the execution of the database trigger?
20. Differentiate between the cursors declared in procedures and the cursors declared in the package specifications.

PL/SQL Advanced Interview Questions

21. What are COMMIT, ROLLBACK and SAVEPOINT statements in PL/SQL?
22. How can you debug your PL/SQL code?
23. What is the difference between a mutating table and a constraining table?
24. In what cursor attributes the outcomes of DML statement execution are saved?
25. Is it possible to declare column which has the number data type and its scale larger than the precision? For example defining columns like: column name NUMBER (10,100), column name NUMBER (10,-84)

PL/SQL Programs

26. Write a PL/SQL program using WHILE loop for calculating the average of the numbers entered by user. Stop the entry of numbers whenever the user enters the number 0.
27. Write a PL/SQL procedure for selecting some records from the database using some parameters as filters.
28. Write a PL/SQL code to count the number of Sundays between the two inputted dates.
29. Write PL/SQL code block to increment the employee's salary by 1000 whose employee_id is 102 from the given table below.
30. Write a PL/SQL code to find whether a given string is palindrome or not.
31. Write PL/SQL program to convert each digit of a given number into its corresponding word format.
32. Write PL/SQL program to find the sum of digits of a number.

PL/SQL Conclusion

Let's get Started

PL/SQL stands for Procedural Language extensions to SQL (Structured Query Language). It was created by Oracle in order to overcome the disadvantages of SQL for easier building and handling of critical applications in a comprehensive manner. Following are the disadvantages of SQL:

- There is no provision of decision-making, looping, and branching in SQL.
- Since the SQL statements get passed to the Oracle engine all at the same time, the speed of execution decreases due to the nature of increased traffic.
- There is no feature of error checking while manipulating the data.

PL/SQL was introduced to overcome the above disadvantages by retaining the power of SQL and combining it with the procedural statements. It is developed as a block-structured language and the statements of the block are passed to the oracle engine which helps to increase the speed of processing due to the decrease in traffic.

PL/SQL Basic Interview Questions

1. What are the features of PL/SQL?

Following are the features of PL/SQL:

- PL/SQL provides the feature of decision making, looping, and branching by making use of its procedural nature.
- Multiple queries can be processed in one block by making use of a single command using PL/SQL.
- The PL/SQL code can be reused by applications as they can be grouped and stored in databases as PL/SQL units like functions, procedures, packages, triggers, and types.
- PL/SQL supports exception handling by making use of an exception handling block.
- Along with exception handling, PL/SQL also supports error checking and validation of data before data manipulation.
- Applications developed using PL/SQL are portable across computer hardware or operating system where there is an Oracle engine.

2. What do you understand by PL/SQL table?

- PL/SQL tables are nothing but objects of type tables that are modeled as database tables. They are a way to provide arrays that are nothing but temporary tables in memory for faster processing.
- These tables are useful for moving bulk data thereby simplifying the process.

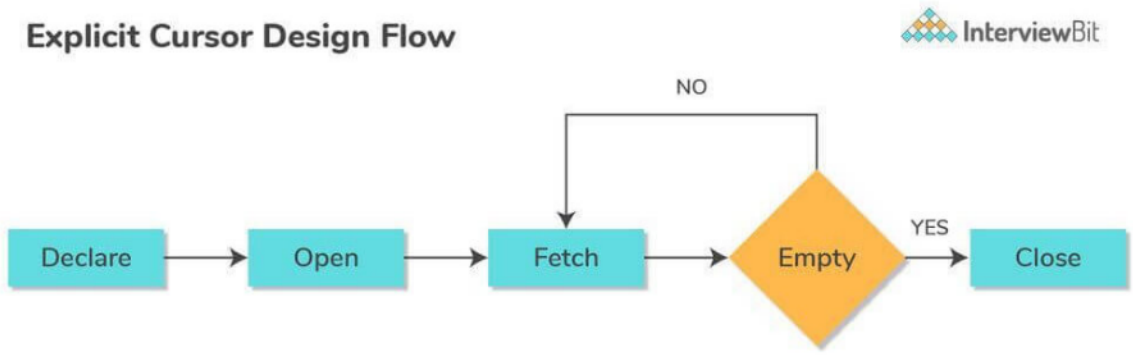
3. Explain the basic structure followed in PL/SQL?

- The basic structure of PL/SQL follows the BLOCK structure. Each PL/SQL code comprises SQL and PL/SQL statement that constitutes a PL/SQL block.
- Each PL/SQL block consists of 3 sections:
    - The optional Declaration Section
    - The mandatory Execution Section
    - The optional Exception handling Section

```
[DECLARE]
--declaration statements (optional)
BEGIN
--execution statements
[EXCEPTION]
--exception handling statements (optional)
END;
```

4. What is a PL/SQL cursor?

- A PL/SQL cursor is nothing but a pointer to an area of memory having SQL statements and the information of statement processing. This memory area is called a context area. This special area makes use of a special feature called cursor for the purpose of retrieving and processing more than one row.
- In short, the cursor selects multiple rows from the database and these selected rows are individually processed within a program.
- There are two types of cursors:
    - **Implicit Cursor**:
        - Oracle automatically creates a cursor while running any of the commands - SELECT INTO, INSERT, DELETE or UPDATE implicitly.
        - The execution cycle of these cursors is internally handled by Oracle and returns the information and status of the cursor by making use of the cursor attributes- ROWCOUNT, ISOPEN, FOUND, NOTFOUND.
    - **Explicit Cursor:**
        - This cursor is a SELECT statement that was declared explicitly in the declaration block.
        - The programmer has to control the execution cycle of these cursors starting from OPEN to FETCH and close.
        - The execution cycle while executing the SQL statement is defined by Oracle along with associating a cursor with it.
- **Explicit Cursor Execution Cycle:**
    - Due to the flexibility of defining our own execution cycle, explicit cursors are used in many instances. The following diagram represents the execution flow of an explicit cursor:



- **Cursor Declaration:**
  - The first step to use an explicit cursor is its declaration.
  - Declaration can be done in a package or a block.
  - **Syntax:** `CURSOR cursor_name IS query;` where cursor_name is the name of the cursor, the query is the query to fetch data from any table.
- **Open Cursor:**
  - Before the process of fetching rows from cursor, the cursor has to be opened.
  - **Syntax** to open a cursor: `OPEN cursor_name;`
  - When the cursor is opened, the query and the bind variables are parsed by Oracle and the SQL statements are executed.
  - The execution plan is determined by Oracle and the result set is determined after associating the cursor parameters and host variables and post these, the cursor is set to point at the first row of the result set.
- **Fetch from cursor:**
  - FETCH statement is used to place the content of the current row into variables.
  - **Syntax:** `FETCH cursor_name INTO variable_list;`
  - In order to get all the rows of a result set, each row needs to be fetched.
- **Close Cursor:**
  - Once all the rows are fetched, the cursor needs to be closed using the CLOSE statement.
  - **Syntax**: `CLOSE cursor_name;`
  - The instructions tell Oracle to release the memory allocated to the cursor.
    - Cursors declared in procedures or anonymous blocks are by default closed post their execution.
    - Cursors declared in packages need to be closed explicitly as the scope is global.
    - Closing a cursor that is not opened will result in INVALID_CURSOR exception.

5. What is the use of WHERE CURRENT OF in cursors?

- We use this clause while referencing the current row from an explicit cursor. This clause allows applying updates and deletion of the row currently under consideration without explicitly referencing the row ID.
- **Syntax:**

```
UPDATE table_name SET field=new_value WHERE CURRENT OF cursor_name
```

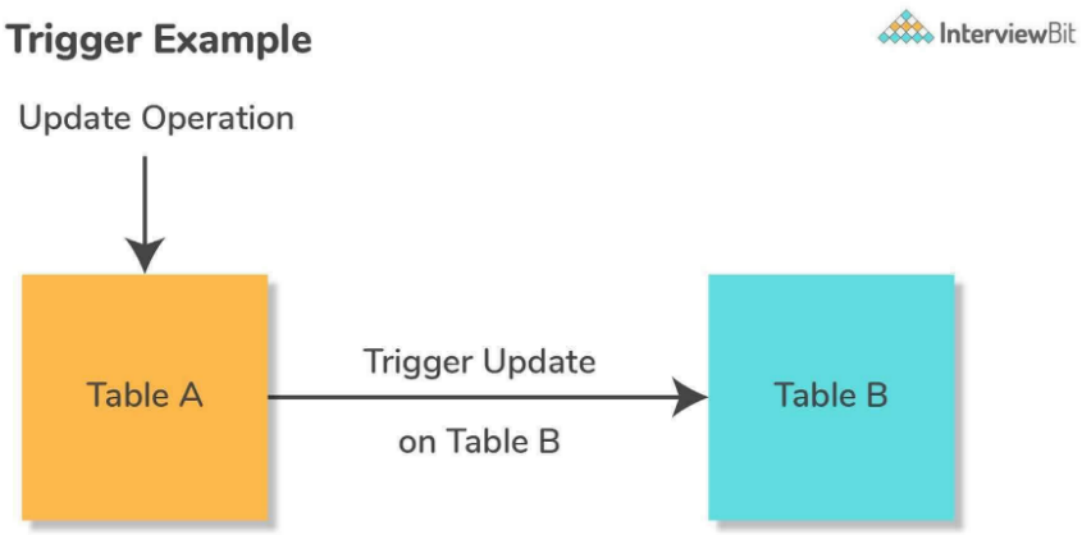
6. How can a name be assigned to an unnamed PL/SQL Exception Block?

- This can be done by using Pragma called **EXCEPTION_INIT**.
- This gives the flexibility to the programmer to instruct the compiler to provide custom error messages based on the business logic by overriding the pre-defined messages during the compilation time.
- **Syntax:**

```
DECLARE
  exception_name EXCEPTION;
  PRAGMA EXCEPTION_INIT (exception_name, error_code);
BEGIN
// PL/SQL Logic
EXCEPTION
 WHEN exception_name THEN
    // Steps to handle exception
END;
```

7. What is a Trigger? Name some instances when “Triggers” can be used.

- As the name indicates, ‘Trigger’ means to ‘activate’ something. In the case of PL/SQL, a trigger is a stored procedure that specifies what action has to be taken by the database when an event related to the database is performed.



- **Syntax:**

```
TRIGGER trigger_name
 trigger_event
 [ restrictions ]
BEGIN
 actions_of_trigger;
END;
```

In the above syntax, if the `trigger_name` the trigger is in the enabled state, the `trigger_event` causes the database to fire `actions_of_trigger` if the `restrictions` are TRUE or unavailable.

- They are mainly used in the following scenarios:
  - In order to maintain complex integrity constraints.
  - For the purpose of auditing any table information.
  - Whenever changes are done to a table, if we need to signal other actions upon completion of the change, then we use triggers.
  - In order to enforce complex rules of business.
  - It can also be used to prevent invalid transactions.
- You can refer https://docs.oracle.com/database/121/TDDDG/tdddg_triggers.htm for more information regarding triggers.

8. When does a DECLARE block become mandatory?

- This statement is used by anonymous blocks of PL/SQL such as non-stored and stand-alone procedures. When they are being used, the statement should come first in the stand-alone file.

9. How do you write comments in a PL/SQL code?

- Comments are those sentences that have no effect on the functionality and are used for the purpose of enhancing the readability of the code. They are of two types:
  - **Single Line Comment:** This can be created by using the symbol `--` and writing what we want to mention as a comment next to it.
  - **Multi-Line comment:** These are the comments that can be specified over multiple lines and the syntax goes like `/* comment information */`
- **Example:**

```
SET SERVEROUTPUT ON;
DECLARE

-- Hi There! I am a single line comment.
var_name varchar2(40) := 'I love PL/SQL' ;
BEGIN
 /*
 Hi! I am a multi line
 comment. I span across
 multiple lines
 */
dbms_output.put_line(var_name);
END;
/
Output:
I love PL/SQL
```

10. What is the purpose of WHEN clause in the trigger?

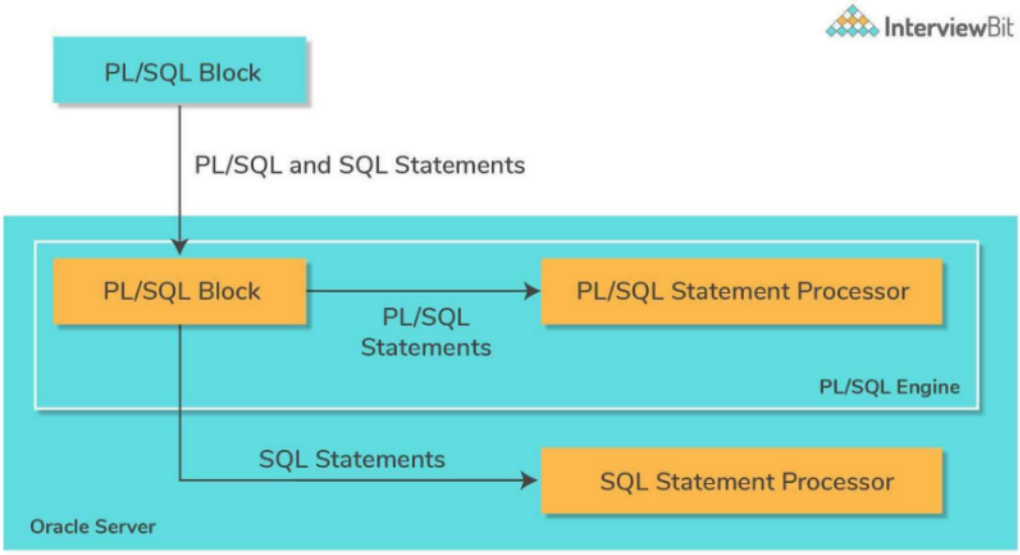
- WHEN clause specifies for what condition the trigger has to be triggered.

PL/SQL Intermediate Interview Questions

11. Can you explain the PL/SQL execution architecture?

The PL/SQL engine does the process of compilation and execution of the PL/SQL blocks and programs and can only work if it is installed on an Oracle server or any application tool that supports Oracle such as Oracle Forms.

- PL/SQL is one of the parts of Oracle RDBMS, and it is important to know that most of the Oracle applications are developed using the client-server architecture. The Oracle database forms the server-side and requests to the database form a part of the client-side.
- So based on the above fact and the fact that PL/SQL is not a standalone programming language, we must realize that the PL/SQL engine can reside in either the client environment or the server environment. This makes it easy to move PL/SQL modules and sub-programs between server-side and client-side applications.
- Based on the architecture shown below, we can understand that PL/SQL engine plays an important role in the process and execute the PL/SQL statements and whenever it encounters the SQL statements, they are sent to the SQL Statement Processor.



- **Case 1: PL/SQL engine is on the server:** In this case, the whole PL/SQL block gets passed to the PL/SQL engine present on the Oracle server which is then processed and the response is sent.
- **Case 2: PL/SQL engine is on the client:** Here the engine lies within the Oracle Developer tools and the processing of the PL/SQL statements is done on the client-side.
    - In case, there are any SQL statements in the PL/SQL block, then they are sent to the Oracle server for SQL processing.
    - When there are no SQL statements, then the whole block processing occurs at the client-side.

12. Why is SYSDATE and USER keywords used?

- **SYSDATE:**
    - This keyword returns the current time and date on the local database server.
    - The syntax is SYSDATE.
    - In order to extract part of the date, we use the TO_CHAR function on SYSDATE and specify the format we need.
    - Usage:
        - `SELECT SYSDATE FROM dual;`
        - `SELECT id, TO_CHAR(SYSDATE, 'yyyy/mm/dd') from InterviewBitEmployeeTable where customer_id < 200;`
- **USER:**
    - This keyword returns the user id of the current session.
    - Usage:
        - `SELECT USER FROM dual;`

13. Differentiate between implicit cursor and explicit cursor.

Implicit Cursor	Explicit Cursor
An implicit cursor is used when a query returns a single row value.	When a subquery returns more than one row, an explicit cursor is used. These rows are called Active Set.
This is used for all DML operations like DECLARE, OPEN, FETCH, CLOSE.	This is used to process Multirow SELECT Statements.
NO_DATA_FOUND Exception is handled here.	NO_DATA_FOUND cannot be handled here.

14. Differentiate between SQL and PL/SQL.

SQL	PL/SQL
SQL is a natural language meant for the interactive processing of data in the database.	PL/SQL is a procedural extension of SQL.
Decision-making and looping are not allowed in SQL.	PL/SQL supports all features of procedural language such as conditional and looping statements.
All SQL statements are executed at a time by the database server which is why it becomes a time-consuming process.	PL/SQL statements are executed one block at a time thereby reducing the network traffic.
There is no error handling mechanism in SQL.	This supports an error handling mechanism.

15. What is the importance of %TYPE and %ROWTYPE data types in PL/SQL?

- **%TYPE:** This declaration is used for the purpose of anchoring by providing the data type of any variable, column, or constant. It is useful during the declaration of a variable that has the same data type as that of its table **column**.
    - Consider the example of declaring a variable named `ib_employeeid` which has the data type and its size same as that of the column `employeeid` in table `ib_employee` .
      The syntax would be : `ib_employeeid ib_employee.employeeid%TYPE;`
- **%ROWTYPE:** This is used for declaring a variable that has the same data type and size as that of a row in the table. The row of a table is called a record and its fields would have the same data types and names as the columns defined in the table.
    - For example: In order to declare a record named `ib_emprecord` for storing an entire row in a table called `ib_employee` , the syntax is:

      ```
      ib_emprecord ib_employee%ROWTYPE;
      ```

16. What are the various functions available for manipulating the character data?

- The functions that are used for manipulating the character data are called String Functions.
  - **LEFT:** This function returns the specified number of characters from the left part of a string.
    - Syntax: LEFT(string_value, numberOfCharacters).
    - For example, LEFT('InterviewBit', 9) will return 'Interview'.
  - **RIGHT:** This function returns the defined number of characters from the right part of a string.
    - Syntax: RIGHT(string_value, numberOfCharacters)
    - For example, RIGHT('InterviewBit',3) would return 'Bit'.
  - **SUBSTRING:** This function would select the data from a specified start position through the number of characters defined from any part of the string.
    - Syntax: SUBSTRING(string_value, start_position, numberOfCharacters)
    - For example, SUBSTRING('InterviewBit',2,4) would return 'terv'.
  - **LTRIM:** This function would trim all the white spaces on the left part of the string.
    - Syntax: LTRIM(string_value)
    - For example, LTRIM(' InterviewBit') will return 'InterviewBit'.
  - **RTRIM:** This function would trim all the white spaces on the right part of the string.
    - Syntax: RTRIM(string_value)
    - For example, RTRIM('InterviewBit ') will return 'InterviewBit'.
  - **UPPER:** This function is used for converting all the characters to the upper case in a string.
    - Syntax: UPPER(string_variable)
    - For example, UPPER('interviewBit') would return 'INTERVIEWBIT'.
  - **LOWER:** This function is used for converting all the characters of a string to lowercase.
    - Syntax: LOWER(string_variable)
    - For example, LOWER('INterviewBit') would return 'interviewbit'.

17. What is the difference between ROLLBACK and ROLLBACK TO statements in PL/SQL?

- **ROLLBACK command** is used for rolling back all the changes from the beginning of the transaction.
- **ROLLBACK TO command** is used for undoing the transaction only till a SAVEPOINT. The transactions cannot be rolled back before the SAVEPOINT and hence the transaction remains active even before the command is specified.

18. What is the use of SYS.ALL_DEPENDENCIES?

- SYS.ALL_DEPENDENCIES is used for describing all the dependencies between procedures, packages, triggers, functions that are accessible to the current user. It returns the columns like name, dependency_type, type, referenced_owner etc.

19. What are the virtual tables available during the execution of the database trigger?

- The THEN and NOW tables are the virtual tables that are available during the database trigger execution. The table columns are referred to as THEN.column and NOW.column respectively.
- Only the NOW.column is available for insert-related triggers.
- Only the THEN.column values are available for the DELETE-related triggers.
- Both the virtual table columns are available for UPDATE triggers.

20. Differentiate between the cursors declared in procedures and the cursors declared in the package specifications.

- The cursors that are declared in the procedures will have the local scope and hence they cannot be used by other procedures.
- The cursors that are declared in package specifications are treated with global scope and hence they can be used and accessed by other procedures.

PL/SQL Advanced Interview Questions

21. What are COMMIT, ROLLBACK and SAVEPOINT statements in PL/SQL?

- These are the three transaction specifications that are available in PL/SQL.
- **COMMIT:** Whenever any DML operations are performed, the data gets manipulated only in the database buffer and not the actual database. In order to save these DML transactions to the database, there is a need to COMMIT these transactions.
    - COMMIT transaction action does saving of all the outstanding changes since the last commit and the below steps take place:
        - The release of affected rows.
        - The transaction is marked as complete.
        - The details of the transaction would be stored in the data dictionary.
    - **Syntax:** COMMIT;
- **ROLLBACK:** In order to undo or erase the changes that were done in the current transaction, the changes need to be rolled back. ROLLBACK statement erases all the changes since the last COMMIT.
    - **Syntax:** ROLLBACK;
- **SAVEPOINT:** This statement gives the name and defines a point in the current transaction process where any changes occurring before that SAVEPOINT would be preserved whereas all the changes after that point would be released.
    - **Syntax:** SAVEPOINT <savepoint_name>;

22. How can you debug your PL/SQL code?

- We can use DBMS_OUTPUT and DBMS_DEBUG statements for debugging our code:
    - DBMS_OUTPUT prints the output to the standard console.
    - DBMS_DEBUG prints the output to the log file.

23. What is the difference between a mutating table and a constraining table?

- A table that is being modified by the usage of the DML statement currently is known as a mutating table. It can also be a table that has triggers defined on it.
- A table used for reading for the purpose of referential integrity constraint is called a constraining table.

24. In what cursor attributes the outcomes of DML statement execution are saved?

- The outcomes of the execution of the DML statement is saved in the following 4 cursor attributes:
  - SQL%FOUND: This returns TRUE if at least one row has been processed.
  - SQL%NOTFOUND: This returns TRUE if no rows were processed.
  - SQL%ISOPEN: This checks whether the cursor is open or not and returns TRUE if open.
  - SQL%ROWCOUNT: This returns the number of rows processed by the DML statement.

25. Is it possible to declare column which has the number data type and its scale larger than the precision? For example defining columns like: column name NUMBER (10,100), column name NUMBER (10,-84)

- Yes, these type of declarations are possible.
- Number (9, 12) indicates that there are 12 digits after decimal point. But since the maximum precision is 9, the rest are 0 padded like 0.000999999999.
- Number (9, -12) indicates there are 21 digits before the decimal point and out of that there are 9 possible digits and the rest are 0 padded like 999999999000000000000.0

PL/SQL Programs

26. Write a PL/SQL program using WHILE loop for calculating the average of the numbers entered by user. Stop the entry of numbers whenever the user enters the number 0.

```
DECLARE
   n NUMBER;
   average NUMBER :=0 ;
   sum NUMBER :=0 ;
   count NUMBER :=0 ;
BEGIN
   -- Take input from user
   n := &input_number;
   WHILE(n<>0)
       LOOP
           -- Increment count to find total elements
           count := count+1;
           -- Sum of elements entered
           sum := sum+n;
           -- Take input from user
           n := &input_number;
       END LOOP;
   -- Average calculation
   average := sum/count;
   DBMS_OUTPUT.PUT_LINE('Average of entered numbers is '||average);
END;
```

27. Write a PL/SQL procedure for selecting some records from the database using some parameters as filters.

- Consider that we are fetching details of employees from ib_employee table where salary is a parameter for filter.

```
CREATE PROCEDURE get_employee_details @salary nvarchar(30)
AS
BEGIN
   SELECT * FROM ib_employee WHERE salary = @salary;
END;
```

28. Write a PL/SQL code to count the number of Sundays between the two inputted dates.

```
--declare 2 dates of type Date
DECLARE
   start_date Date;
   end_date Date;
   sundays_count Number:=0;
BEGIN
   -- input 2 dates
   start_date:='&input_start_date';
   end_date:='&input_end_date';
   /*
   Returns the date of the first day after the mentioned date
   and matching the day specified in second parameter.
   */
   start_date:=NEXT_DAY(start_date-1, 'SUNDAY');
   --check the condition of dates by using while loop.
   while(start_date<=end_date)
   LOOP
       sundays_count:=sundays_count+1;
       start_date:=start_date+7;
   END LOOP;

   -- print the count of sundays
   dbms_output.put_line('Total number of Sundays between the two dates:'||sundays_count
END;
/
```

Input:
start_date = '01-SEP-19'
end_date = '29-SEP-19'

Output:
Total number of Sundays between the two dates: 5

29. Write PL/SQL code block to increment the employee's salary by 1000 whose employee_id is 102 from the given table below.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL_ID
100	ABC	DEF	abef
101	GHI	JKL	ghkl
102	MNO	PQR	mnqr
103	STU	VWX	stwx

```
DECLARE
 employee_salary  NUMBER(8,2);

 PROCEDURE update_salary (
   emp        NUMBER,
   salary IN OUT NUMBER
 ) IS
 BEGIN
   salary := salary + 1000;
 END;

BEGIN
 SELECT salary INTO employee_salary
 FROM ib_employee
 WHERE employee_id = 102;

 DBMS_OUTPUT.PUT_LINE
  ('Before update_salary procedure, salary is: ' || employee_salary);

 update_salary (100, employee_salary);

 DBMS_OUTPUT.PUT_LINE
  ('After update_salary procedure, salary is: ' || employee_salary);
END;
/
```

Result:

```
Before update_salary procedure, salary is: 17000
After update_salary procedure, salary is: 18000
```

30. Write a PL/SQL code to find whether a given string is palindrome or not.

```
DECLARE
-- Declared variables string, letter, reverse_string where string is the original strir
string VARCHAR2(10) := 'abccba';
letter VARCHAR2(20);
reverse_string VARCHAR2(10);
BEGIN
FOR i IN REVERSE 1..LENGTH(string) LOOP
 letter := SUBSTR(string, i, 1);
 -- concatenate letter to reverse_string variable
 reverse_string := reverse_string ||''||letter;
END LOOP;
IF reverse_string = string THEN
       dbms_output.Put_line(reverse_string||''||' is palindrome');
ELSE
       dbms_output.Put_line(reverse_string ||'' ||' is not palindrome');
END IF;
END;
```

31. Write PL/SQL program to convert each digit of a given number into its corresponding word format.

```
DECLARE
-- declare necessary variables
   -- num represents the given number
   -- number_to_word represents the word format of the number
   -- str, len and digit are the intermediate variables used for program execution
num   INTEGER;
number_to_word VARCHAR2(100);
digit_str   VARCHAR2(100);
len   INTEGER;
digit   INTEGER;
BEGIN
   num := 123456;
   len := LENGTH(num);
   dbms_output.PUT_LINE('Input: ' ||num);
   -- Iterate through the number one by one
   FOR i IN 1..len LOOP
       digit := SUBSTR(num, i, 1);
       -- Using DECODE, get the str representation of the digit
       SELECT Decode(digit, 0, 'Zero ',
                      1, 'One ',
                      2, 'Two ',
                      3, 'Three ',
                      4, 'Four ',
                      5, 'Five ',
                      6, 'Six ',
                      7, 'Seven ',
                      8, 'Eight ',
                      9, 'Nine ')
       INTO digit_str
       FROM dual;
       -- Append the str representation of digit to final result.
       number_to_word := number_to_word || digit_str;
   END LOOP;
   dbms_output.PUT_LINE('Output: ' ||number_to_word);
END;
```

Input: 12345
Output: One Two Three Four Five

32. Write PL/SQL program to find the sum of digits of a number.

```
DECLARE
--Declare variables num, sum_of_digits and remainder of datatype Integer
num  INTEGER;
sum_of_digits INTEGER;
remainder  INTEGER;
BEGIN
num := 123456;
sum_of_digits := 0;
-- Find the sum of digits until original number doesnt become null
WHILE num <> 0 LOOP
 remainder := MOD(num, 10);
 sum_of_digits := sum_of_digits + remainder;
 num := TRUNC(num / 10);
END LOOP;
dbms_output.PUT_LINE('Sum of digits is '|| sum_of_digits);
END;
```

Input: 9874
Output: 28

PL/SQL Conclusion

33. PL SQL Interview

- PL/SQL is a programming extension of SQL developed by Oracle which combines the power of SQL in the field of data manipulation and the power of procedural language for faster and effective processing of data thereby resulting in the creation of powerful queries.
- PL/SQL enhances the security, increases the platform portability, and makes it more robust by means of instructing the compiler 'what to do' and 'how to do' using SQL and procedural form respectively.
- Finally, it gives more power over the database to the programmers due to the feature of decision making, filtering, and looping abilities thereby making it a more convenient and reliable means for the programmers to work on it.

References:
https://oracle-base.com/articles/misc/introduction-to-plsql

Recommended Tutorials:

SQL Server Interview Questions
MySQL Interview Questions
MongoDB Interview Questions
DBMS Interview Questions

Links to More Interview Questions

C Interview Questions	Php Interview Questions	C Sharp Interview Questions
Web Api Interview Questions	Hibernate Interview Questions	Node Js Interview Questions
Cpp Interview Questions	Oops Interview Questions	Devops Interview Questions
Machine Learning Interview Questions	Docker Interview Questions	Mysql Interview Questions
Css Interview Questions	Laravel Interview Questions	Asp Net Interview Questions
Django Interview Questions	Dot Net Interview Questions	Kubernetes Interview Questions
Operating System Interview Questions	React Native Interview Questions	Aws Interview Questions
Git Interview Questions	Java 8 Interview Questions	Mongodb Interview Questions
Dbms Interview Questions	Spring Boot Interview Questions	Power Bi Interview Questions
Pl Sql Interview Questions	Tableau Interview Questions	Linux Interview Questions
Ansible Interview Questions	Java Interview Questions	Jenkins Interview Questions