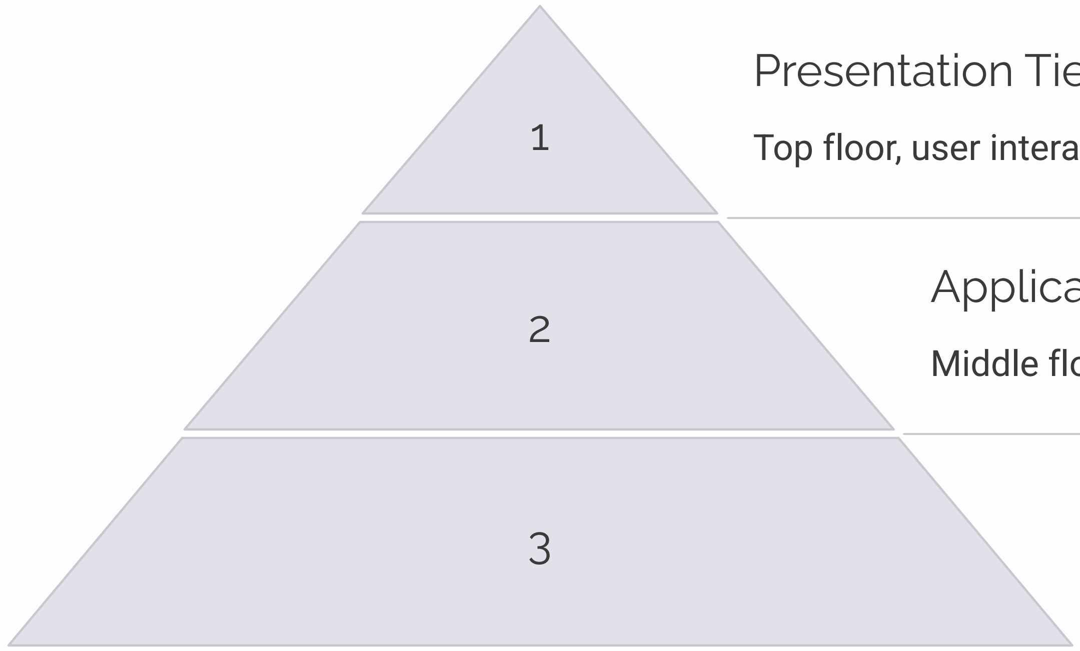
Understanding PL/SQL Architecture

Alright, welcome back. In this lesson we're going to understand the PLSQL architecture. Let's imagine PLSQL architecture as a multi-storey building. Each floor represents a different tier or layer of the system. This analogy will help us understand how PLSQL fits into the larger database ecosystem. We have the ground floor, data management tier. This is where your Oracle database lives. It's the foundation of everything, storing all your data and tables and other structures. Think of it as the basement of our building, where all the important resources are kept. So we have the middle floors, application processing tier.

por Mayko Silva

PL/SQL Architecture Tiers



Presentation Tier

Top floor, user interaction

Application Processing Tier

Middle floors, PL/SQL programs

Data Management Tier

Ground floor, Oracle database

Here we will find the PL SQL programs that processes data and implement business logic. This is where the magic happens, where data is transformed, calculations are made, and decisions are taken based on your coded instructions. So we have the presentation tier or top floor. This is where users interact with the system, perhaps through a web interface or mobile app. It's like the penthouse suite of our beauty, where the results of all the work done on the lower floors are presented in a user-friendly manner. Now, let's take a look at the PL SQL Engine. At the heart of this building is the PL SQL engine like an elevator that moves between floors processing your PL SQL code and interacting with the database.

PL/SQL Engine Workflow

1

Code Reception

Engine receives PL/SQL code

2

Parsing

Checks code for syntax errors

3

Execution

Executes code, moving between tiers

4

Result Return

Returns result to appropriate tier

Here's how it works. We have code reception. This is where the engine receives your PL SQL code. Then we have parsing. It checks the code for syntax errors much like a security check in your building. Then we have execution. If the code passes the syntax check the engine executes it moving between the tier as necessary. Then we have result return. Finally it returns the result for the appropriate tier. Now let's take a look at the power of PLSQL blocks. We can see it as our rooms in our building. PL SQL programs are built using blocks.

PL/SQL Block Structure

Declaration Section

Where you define variables, like labeling items in a room.

```
DECLARE
  v_name VARCHAR2(50);
  v_age NUMBER;
```

Execution Section

Where the action happens, like performing tasks in the room.

```
BEGIN
  -- Your code here
END;
```

Exception Handling Section

Your safety net, handling unexpected situations (optional but recommended).

```
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    -- Handle exception
  WHEN OTHERS THEN
    -- Handle other errors
```

Think of these blocks as rooms in your building. Each room or block can have 3 sections. We have the declaration section where you define variables and it's like labeling items in a room. For example, we have this piece of code showing us how we can declare variables, ok? Then we have the execution section. This is where the action happens like performing tasks in the room it always starts with begin okay and ends with the end for example okay it starts here and ends here alright then we have exception handling section This is your safe net handling unexpected situations.

PL/SQL Compilation Process

1

Syntax Checking

First the ingredients, your code, are checked for quality. The PLSQL compiler checks for any syntax error, like missing semicolons or misspelled words.

2

Semantic Checking

Then the recipe is reviewed to ensure it makes sense. The compiler checks if you're using variables correctly, if your SQL statements are valid and if you have the right permissions to access database objects.

3

Code Generation

Finally the meal is prepared. The compiler generates either interpreted code, p-code or native code.

It's optional but highly recommended. For example we have here an exception ok when no data found when no data found then print this message when other errors then an error occurred ok And now we're going to see how PLSQL works behind the scenes. The buildings operation. When you run a PLSQL program it goes through a compilation process. This is like preparing a meal in our building's kitchen.

PL/SQL Code Types

P-code

It's like a set of cooking instructions that need to be interpreted each time the meal is prepared. It's more flexible but might be slightly slower.

Native Code

This is like pre-cooking the meal. It's faster to serve but less flexible if you need to make change.

Ok, And then we have the next phase, semantic checking. Then the recipe is reviewed to ensure it makes sense. The compiler checks if you're using variables correctly, if your SQL statements are valid and if you have the right permissions to access database objects. Then we have the code generation phase. Finally the meal is prepared. The compiler generates either interpreted code, p-code or native code. P-code it's like a set of cooking instructions that need to be interpreted each time the meal is prepared. It's more flexible but might be slightly slower. Then we have also native code.

This is like pre-cooking the meal. It's faster to serve but less flexible if you need to make change. The choice between P code and native code is determined by the PL SQL underscore code underscore type parameter. Alright, we have this parameter. Ok, and finally let's talk about efficiency benefits of PLSQL.

Efficiency Benefits of PL/SQL

Reduced Network Traffic

PLSQL can handle multiple statements, reducing back and forth communication between the application and the database.

Improved Performance

PLSQL is tightly integrated with the SQL engine, allowing for optimized execution of SQL statements within PL SQL blocks.

Portability

PL SQL code can be easily moved between different Oracle environments from on-premise databases to cloud solutions.

Scalability

PLSQL's modular structure allows for easy maintenance and expansion of your database application as they grow.

PLSQL architectures offer several advantages. We have reduced network traffic because PLSQL can handle multiple statements. It reduces the back and forth communication between the application and the database. We have improved performance because PLSQL is tightly integrated with the SQL engine, allowing for optimized execution of SQL statements within PL SQL blocks.

Portability. PL SQL code can be easily moved between different Oracle environments from on-premise databases to cloud solutions. And we have also scalability because the PLSQL's modular structure allows for easy maintenance and expansion of your database application as they grow. Understanding this architecture is crucial for writing efficient, scalable database application. As you continue to explore PLSQL you will discover how this robust architecture supports even the most complex database operations. Alright? I see you on the next lesson!