Procedures And Functions ====>
==========

* Procedures and Functions are subprograms which can be created and saved in the database as database objects.

* Any stored procedure or function created, remains useless unless it is called.

* Therefore, after creating a stored procedure or function it is necessary to make a call for that stored procedure or function from another PL/SQL block in order to serve the purpose for which it has been created.

* Whenever a block of code for stored procedure or function is written it is then automatically compiled by the oracle engine.

* Once it is compiled, it is then stored by the oracle engine in the database as a database object.

* Now , whenever the calling of procedure or function is done ,the oracle engine loads the compiled code into a memory due to which execution becomes faster.

Procedures V/s Functions ===>
==========

Procedure ==>
==========
- A procedure cannot return a value.
- Cannot be called from SELECT statement.
- Use OUT parameter to return the value.
- Return data type will not be specified atthe time of creation.

Function ==>
==========
- A function MUST return a value.
- Can be called from SELECT statement.
- Use RETURN to return the value, but can use OUT also.
- Return data type is mandatory at the time of creation.

Syntax Of Creating A Procedure ===>
==========

```
CREATE OR REPLACE PROCEDURE <procedure_name>
    (<variable_name>IN / OUT / IN OUT <datatype>,
    <variable_name>IN / OUT / IN OUT <datatype>,...) IS/AS
    variable/constant declaration;
BEGIN
    -- PL/SQL subprogram body;
EXCEPTION
    -- Exception Handling block ;
END <procedure_name>;
```

Explanation :-

* CREATE or REPLACE PROCEDURE is a keyword used for specifying the name of the procedure to be created.

* procedure_name is for procedure's name and variable_name is the variable name for variable used in the stored procedure.

* BEGIN, EXCEPTION and END are keywords used to indicate different sections of the procedure being created.

* IN/OUT/IN OUT are parameter modes.

- `IN`:- An IN parameter lets you pass a value to the subprogram. It is a read-only parameter. Inside the subprogram, an IN parameter acts like a constant. It cannot be assigned a value. You can pass a constant, literal, initialized variable, or expression as an IN parameter. I It is the default mode of parameter passing.

- `OUT` :- An OUT parameter returns a value to the calling program. Inside the subprogram, an OUT parameter acts like a variable. You can change its value and reference the value after assigning it. The actual parameter must be variable.

- `IN OUT` :- An IN OUT parameter passes an initial value to a subprogram and returns an updated value to the caller. It can be assigned a value and the value can be read. The actual parameter corresponding to an IN OUT formal parameter must be a variable, not a constant or an expression. Formal parameter must be assigned a value. Actual parameter is passed by value.

Executing A Procedure ==>

* A standalone procedure can be called in two ways –
- Using the EXECUTE keyword
- Calling the name of the procedure from a PL/SQL block

* Defining :-

```
  1  create or replace procedure GREETINGS as
  2  begin
  3     dbms_output.put_line('Good Morning');
  4  end GREETINGS;
  5  /
```

* Calling :-

```
  1 begin
  2     GREETINGS;
  3  end;
  4  /
```

```
Good Morning
```