# LO9.2 Create anonymous and stored procedures

## PL/SQL blocks

PL/SQL blocks can be anonymous or named.

Anonymous blocks – are not given names and are not stored in the database.

Named blocks – are stored in the database, are given a name, and can be referenced by that name later on.

NOTE: Up until this point, we have been using PL/SQL blocks that are anonymous.

There are 3 types of named blocks:

1. Stored Procedures
2. Functions
3. Triggers

## Stored Procedures

* A named PL/SQL block which performs one or more tasks
* Similar to a procedure in other programming languages.
* It takes in a number of parameters but does not return a value.
* Its similar to anonymous PL/SQL block, but it is named for repeated usage

We can pass parameters to procedures in three ways called types:  
1) IN-parameters – the parameter can be referenced by the value and cannot be overwritten

2) OUT-parameters – the parameter cannot be referenced and can be overwritten.

3) IN OUT-parameters – the parameter can be both referenced and overwritten

\*note: If a parameter is not explicitly given a type, by default it is an IN type.

### Syntax for a stored procedure

*CREATE [OR REPLACE] PROCEDURE proc\_name [(list of parameters)]*

*IS*

*Declaration section*

*BEGIN*

*Execution section*

*EXCEPTION*

*Exception section*

*END;*

\*The syntax within the brackets [ ] indicate they are optional.

The syntax above means:

CREATE – creates the procedure if no other procedure with the same name exists

CREATE OR REPLACE – create new (as above) or replace an existing procedure with the current code

IS - marks the beginning of the body of the procedure and is similar to DECLARE in anonymous PL/SQL Blocks.

So if we have a procedure that we are passing in variables, it might look like:

*CREATE PROCEDURE myProcedure*

*(name\_in IN varchar2, name\_out OUT varchar2, name\_in\_out IN OUT varchar3)*

*IS*

### Executing a Stored Procedure

There are two ways to execute a procedure.

1) From the SQL script window in SQL Developer:

*EXECUTE [or EXEC] procedure\_name;*

2) Within another procedure – simply use the procedure name

*procedure\_name;*