* [PL/SQL - Home](https://www.tutorialspoint.com/plsql/index.htm)
* [PL/SQL - Overview](https://www.tutorialspoint.com/plsql/plsql_overview.htm)
* [PL/SQL - Environment](https://www.tutorialspoint.com/plsql/plsql_environment_setup.htm)
* [PL/SQL - Basic Syntax](https://www.tutorialspoint.com/plsql/plsql_basic_syntax.htm)
* [PL/SQL - Data Types](https://www.tutorialspoint.com/plsql/plsql_data_types.htm)
* [PL/SQL - Variables](https://www.tutorialspoint.com/plsql/plsql_variable_types.htm)
* [PL/SQL - Constants and Literals](https://www.tutorialspoint.com/plsql/plsql_constants.htm)
* [PL/SQL - Operators](https://www.tutorialspoint.com/plsql/plsql_operators.htm)
* [PL/SQL - Conditions](https://www.tutorialspoint.com/plsql/plsql_conditional_control.htm)
* [PL/SQL - Loops](https://www.tutorialspoint.com/plsql/plsql_loops.htm)
* [PL/SQL - Strings](https://www.tutorialspoint.com/plsql/plsql_strings.htm)
* [PL/SQL - Arrays](https://www.tutorialspoint.com/plsql/plsql_arrays.htm)
* [PL/SQL - Procedures](https://www.tutorialspoint.com/plsql/plsql_procedures.htm)
* [PL/SQL - Functions](https://www.tutorialspoint.com/plsql/plsql_functions.htm)
* [PL/SQL - Cursors](https://www.tutorialspoint.com/plsql/plsql_cursors.htm)
* [PL/SQL - Records](https://www.tutorialspoint.com/plsql/plsql_records.htm)
* [PL/SQL - Exceptions](https://www.tutorialspoint.com/plsql/plsql_exceptions.htm)
* [PL/SQL - Triggers](https://www.tutorialspoint.com/plsql/plsql_triggers.htm)
* [PL/SQL - Packages](https://www.tutorialspoint.com/plsql/plsql_packages.htm)
* [PL/SQL - Collections](https://www.tutorialspoint.com/plsql/plsql_collections.htm)
* [PL/SQL - Transactions](https://www.tutorialspoint.com/plsql/plsql_transactions.htm)
* [PL/SQL - Date & Time](https://www.tutorialspoint.com/plsql/plsql_date_time.htm)
* [PL/SQL - DBMS Output](https://www.tutorialspoint.com/plsql/plsql_dbms_output.htm)
* [PL/SQL - Object Oriented](https://www.tutorialspoint.com/plsql/plsql_object_oriented.htm)

**What are PL/SQL functions in Oracle Database?**

In Oracle Database we can define a

PL/**SQL function as a self-contained sub-program** that is meant to do **some specific well defined task**.

Functions are **named PL/SQL block** which means they can be stored into the database as a **database object** and **can be reused**.

That is also the reason why some books refer **to PL/SQL functions as stored functions.**

**Types of PL/SQL functions in Oracle Database**

There are two types of PL/SQL functions in Oracle Database, these are

1. **Pass-by-Value Functions and**
2. **Pass-by-Reference functions**

In Oracle Database both types of functions should have to return some values and these values should be a valid SQL or PL/SQL datatype.

**Syntax of PL/SQL Functions in Oracle Database**

CREATE [OR REPLACE] FUNCTION function\_name

(Parameter 1, Parameter 2…)

RETURN datatype

IS

Declare variable, constant etc.

BEGIN

Executable Statements

Return (Return Value);

END;

I have discussed the PL/SQL function’s syntax line by line in the video tutorial on my [YouTube channel](https://www.youtube.com/rebellionrider) on the same topic. I suggest you to go and check out that [tutorial](https://youtu.be/6OJIrPx61mU) once.

**Function Execution Method**

Depending on your creativity and programming skills, a PL/SQL function can be called by multiple ways.

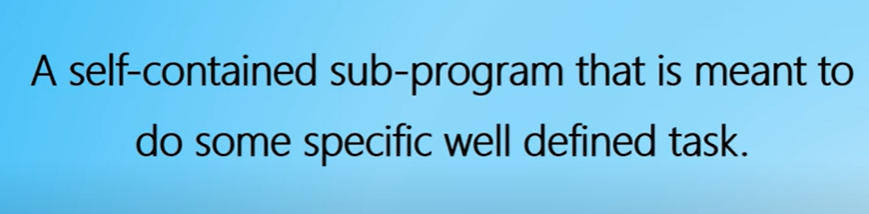
Here are some general ways of calling a PL/SQL function in Oracle Database

1. You can use [SQL\*Plus utility](http://www.rebellionrider.com/how-to-connect-with-oracle-database-using-sqlplus-utility/) of the Oracle Database to invoke a PL/SQL function that can be called from PL/SQL as procedural statement.
2. An anonymous [PL/SQL block](http://www.rebellionrider.com/block-types-in-pl-sql-by-rebellionrider-manish-sharma/) can also be used to call a function.
3. You can even call a **function directly into a SELECT or DML statement.**

Stay tuned we will discuss each of these execution methods of PL/SQL functions in the next tutorial.

**Restrictions on calling a function**

1. A function that returns SQL datatype can be used inside SQL statement and a PL/SQL function that returns PL/SQL datatype only works inside PL/SQL blocks. An exception to this rule is that, you cannot call a function that contains a DML operation inside a SQL query. However you can call a function that performs a DML operation inside INSERT, UPDATE and DELETE.
2. **A function called from an UPDATE or DELETE statement on a table cannot query (SELECT) or perform transaction (DMLs) on the same table.**
3. **A function called from SQL expressions cannot contain the TCL (COMMIT or ROLLBACK) command or the DDL (CREATE or ALTER) command**



In order to keep this tutorial simple and easy to understand, we will create a very easy function which will **calculate the area of a circle**. I guess that will serve the purpose and help you in learning how to create PL/SQL functions in Oracle Database.

As discussed in the previous tutorial that the function body is divided into two parts

* First is the **header** of the PL/SQL function and
* Second is the **execution part** of the PL/SQL function

So let’s start with header of our function.

**Step 1. Create the Header of a PL/SQL Function.**

The header consists of the signature of the function or the declaration of the PL/SQL function.

--Function Header

CREATE OR REPLACE FUNCTION circle\_area (radius NUMBER)

RETURN NUMBER IS

**Step 2. Declare Variables or the Constant.**

If your program requires you to declare any variable or constant or anything then you can do it right after creating the header, that too without using the DECLARE keyword.

--Declare a constant and a variable

pi CONSTANT NUMBER(7,2) := 3.141;

area NUMBER(7,2);

**Step 3. Create the Execution Part of the PL/SQL function.**

Once you have created the **header of your function** and **declared all your necessary variables** as well as constants then you are all set to create the **execution part of your PL/SQL function**. Here in the execution section of a PL/SQL function, you write all your execution statements. This part also defines the working of your function.

BEGIN

--Area of Circle pi\*r\*r;

area := pi \* (radius \* radius);

RETURN area;

END;

/

**Quick Info:  
To calculate the square of the circle’s radius in the area of circle, you can also use the inbuilt function of POWER (p, q). This function takes two numeric input and returns one numeric value which will be the answer to the arithmetic expression of p raise to q.**

Now let’s join all the above chunks of codes together into a single named unit.

**PL/SQL function for calculating “Area of the Circle”.**

--Function Header

CREATE OR REPLACE FUNCTION circle\_area (radius NUMBER)

RETURN NUMBER IS

--Declare a constant and a variable

pi CONSTANT NUMBER(7,2) := 3.141;

area NUMBER(7,2);

BEGIN

--Area of Circle pi\*r\*r;

area := pi \* (radius \* radius);

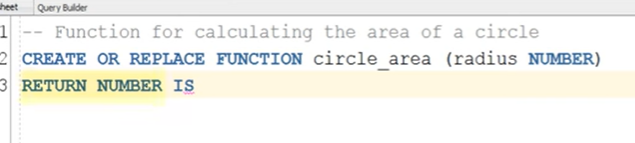
RETURN area;

END;

A successful compilation will create a named PL/SQL block which is your PL/SQL function with the name circle\_area.

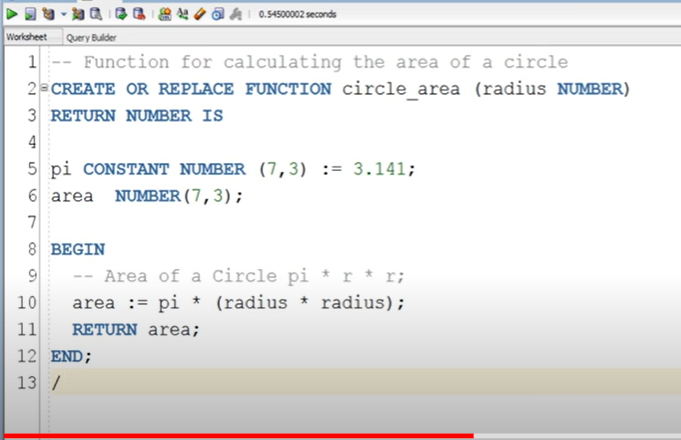
As PL/SQL functions are named PL/SQL block thus they are **permanently saved in your database** which you can use anytime.

In order to see your PL/SQL Function in action you have to call it through your program. Your program can be an **anonymous PL/SQL block, or a named PL/SQL Block or even using a SELECT statement**.



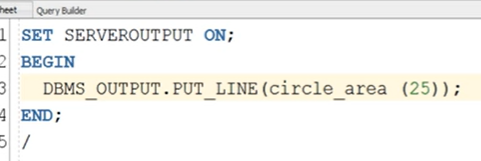
Radius is single parameter with data type Number

Return – we need to mention datatype

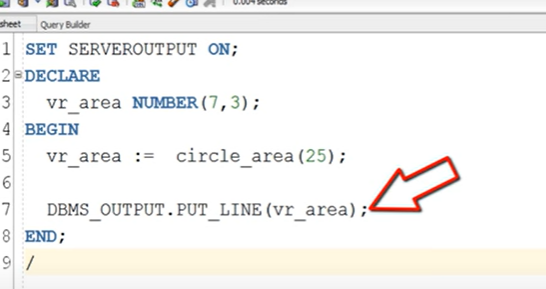


We can call function

1. In an anonymous block



Or



# **What Are PL/SQL Stored Procedures In Oracle Database**

Similar to [PL/SQL Functions](http://www.rebellionrider.com/how-to-create-pl-sql-function-in-oracle-database/) a stored Procedure is a ***self-contained*subprogram** that is meant to do **some specific tasks**.

Also similar to functions, procedures are named **PL/SQL blocks** **thus** they can be reused because they are stored into the **database as a database object**. But *unlike PL/SQL functions a* ***stored procedure does not return any value*.**

**Syntax of PL/SQL Stored Procedures**

CREATE [OR REPLACE] PROCEDURE pro\_name (Parameter – List)

IS [AUTHID DEFINER | CURRENT\_USER]

Declare statements

BEGIN

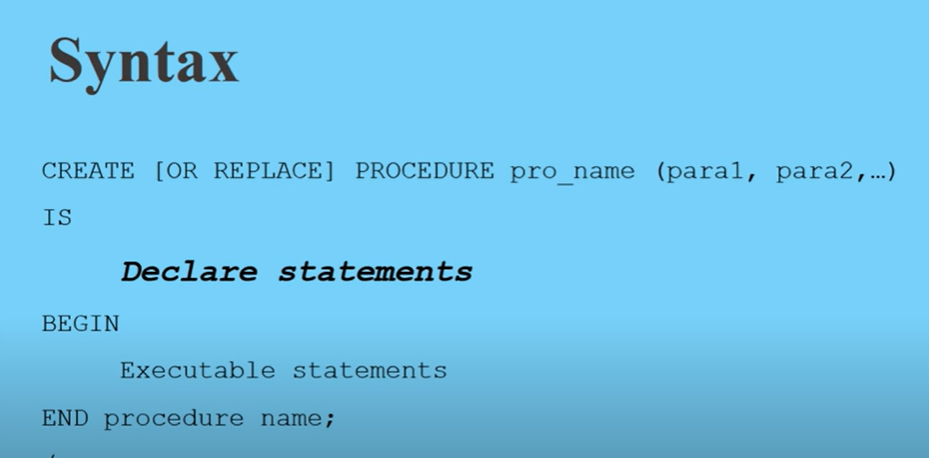
Executable statements

END procedure name;

/

\

Is – indicates ending of stored procedure



The above Syntax of PL/SQL stored procedure is pretty much similar to the syntax of PL/SQL Functions that we saw in the last PL/SQL tutorial.  Except for two things:

1. **There is no Return clause.**

A core difference between a PL/SQL Function and a stored procedure is that unlike Functions a stored procedure does not return any value.

1. AUTHID Clause.

The AUTHID clause is used for s**etting the authority model for the PL/SQL Procedures.** This clause has two flags.

1. DEFINER and
2. CURRENT\_USER

As this clause is o**ptional** thus in case if you do not use AUTHID clause then Oracle Engine will set the authority (AUTHID) to the **DEFINER by default for you**. Now, you must be wondering what these DEFINER and CURRENT\_USER rights are?

**DEFINER right:** Definer right is the default right assigned to the procedure by oracle engine. This right means anyone with Execution Privilege on the procedure acts as if they are the owner of the schema in which the privilege is created.

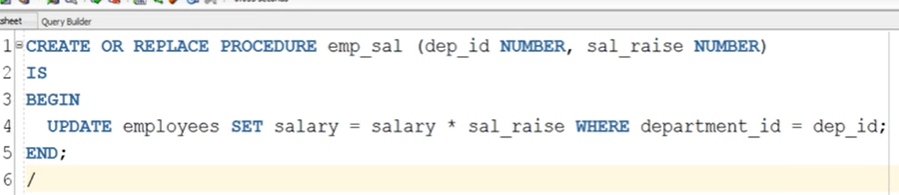
**CURRENT\_USER right:**Setting the authority level of a stored procedure to the current\_user right overrides the default right which is definer and change it to the invoker rights.

Invoker right authority means that you call the procedure to act on your local data and it requires that you replicate data objects in any participating schema.

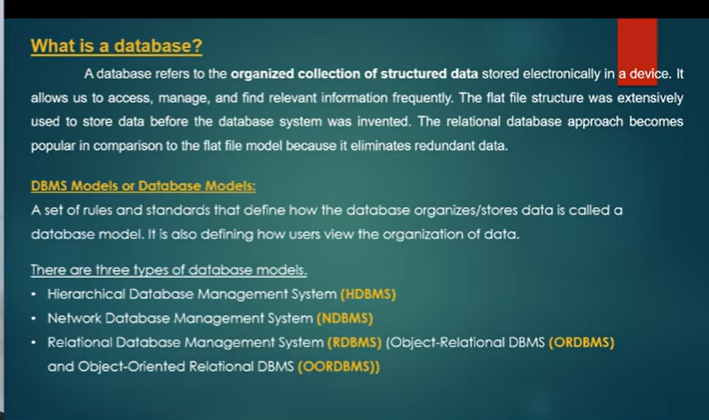
**Some Extra Points About Stored Procedure**

* You can define a procedure with or without [formal parameters](http://www.rebellionrider.com/actual-parameters-versus-formal-parameters/).
* A parameter can be either ***pass-by-value* or *pass-by-reference***.
* A procedure will be a **pass-by-value procedure** when you don’t specify the parameter mode because it uses the default IN mode.

That’s it on Introduction to PL/SQL Stored Procedures. I guess you will also enjoy reading these below mentioned blogs. All the blogs are written by taking Job Interview and Oracle Database Certification in Mind, do make sure to check them out.

* Differences between PL/SQL Function and PL/SQL Stored Procedures?
* What are Parameter Modes in PL/SQL Functions and Procedures?
* 
* [What are Formal and Actual Parameters?](http://www.rebellionrider.com/actual-parameters-versus-formal-parameters/)

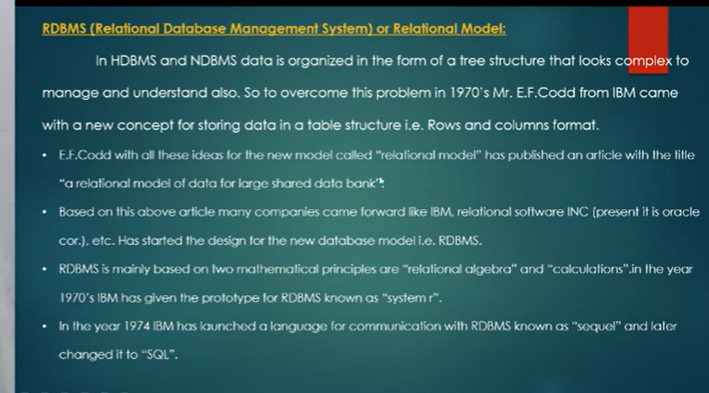
Always remember: You can help others in learning and support me and my channel as well as this blog by sharing it with your friends on your social media.



Previous file are flat without any structure – hard to retrieve data

Hirarchial: one parent – many childs

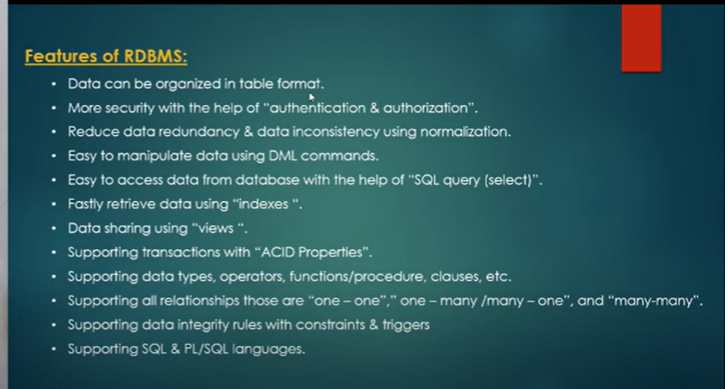
Network: many to many

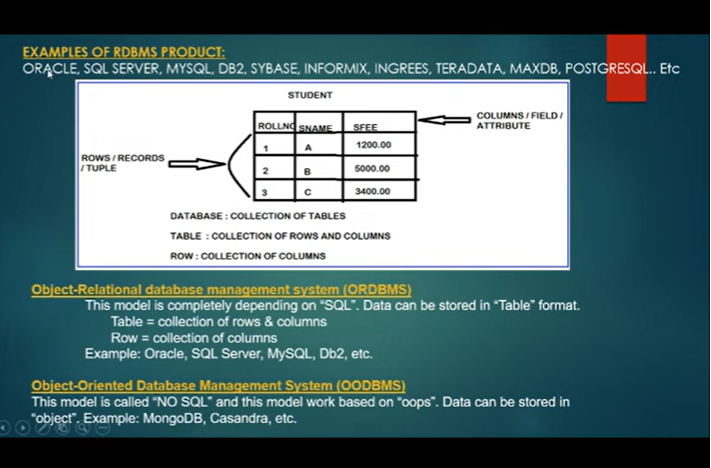


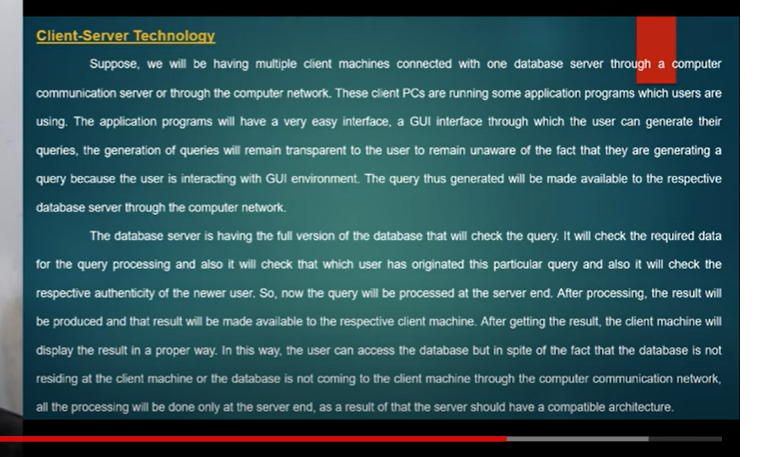
By using tables

And relational algebra

1stdeveloped sql





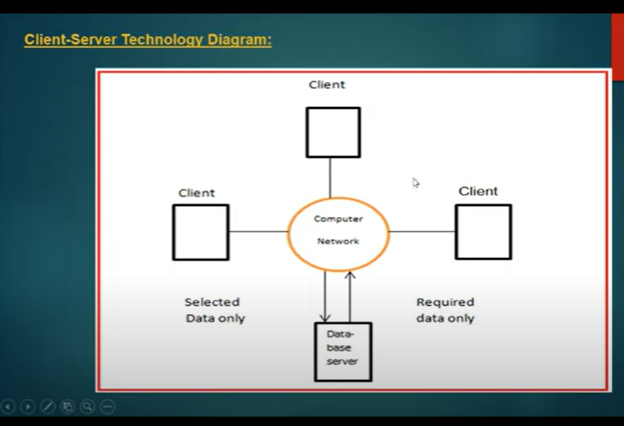


Rdbms works based on client-server tecch

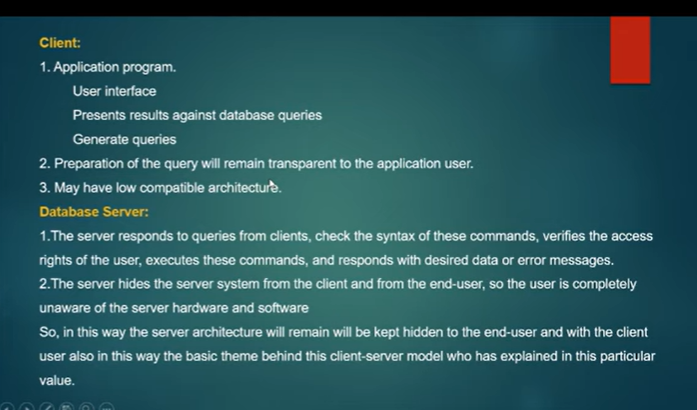
Server maintains data

It send data to clients to network

Client slide: will query



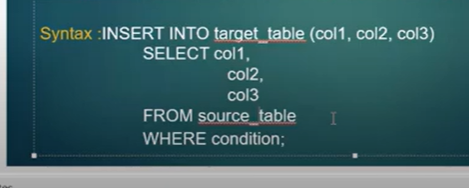
Client side will be having application programming

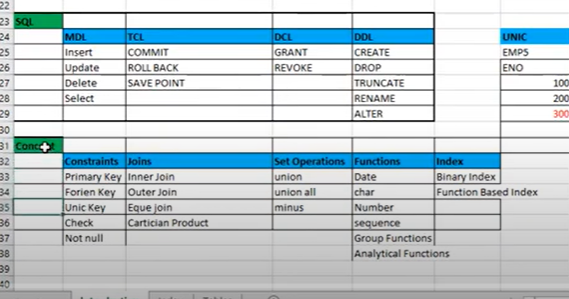


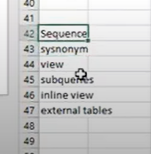
# Oracle and it's Versions-ORACLE 19C-SQL & PL/SQL



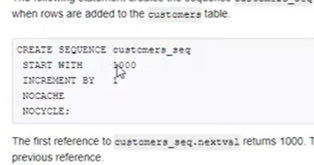
Insert:







Docs.oracle.com – standard



Plsql:

Procedural query language using sql

Use of plsql:

Procedure – called program

Why to develop database programs ?

Who is taking your data from your database server: plsql programmes, they are already stored

Plsql object: any click on webpage: response comes from back end

Plsql collection of program statements + sql queries

Extension to sql

We write sql query to get data(if everyday we want data , dailyexecute, how client write query – programme)

These programmes linked with webpages

Some kind of actions which we able to perform insql?

At a time in sql one query executed

Each query hits database

If num of hits increasing database performance decreases

Queries not saved in database

We need to re write queries again and again

* + If I save that query in database , I can execute whenever I want( I can save query as pl sal object)

In sql we don’t have programme constructs like loop, condition, branching statements all these available in plsql

Plsql qutomate business activites

Database server: collection of programmes and data

Plsql”

Collection of user defined objects

Acc to user requirement objects will be developed

Advantages of PLSQL:

Business activities automated

Executing multiple queries in d form of programs

Reduces number of hits to dbms- improve dbms performance

Improves network performance- because of less network usage

Enhance ability(making changes to old procedures), modularity- sub tasks(to decrease complexity ) and reusability(to get function there is function)

PLSQL-OBJECTS:  
2 categories :

1, anonymous blocks/prorams

2, functions/functions/triggers/packages/types

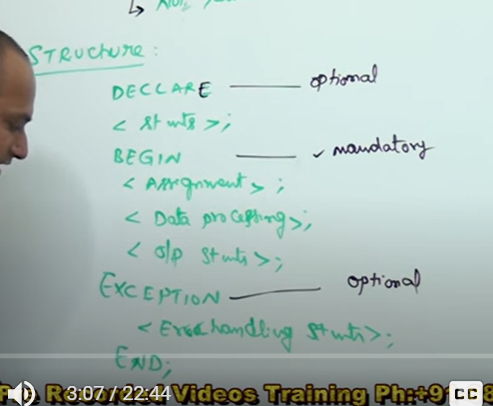
Programs divided into- static(accept server input) and dynamic programs

Program: collection of programming statements and queries to perform set of tasks or single task

Nature of program: program not saved in database

They are temporary

Structure:



Declare Block:

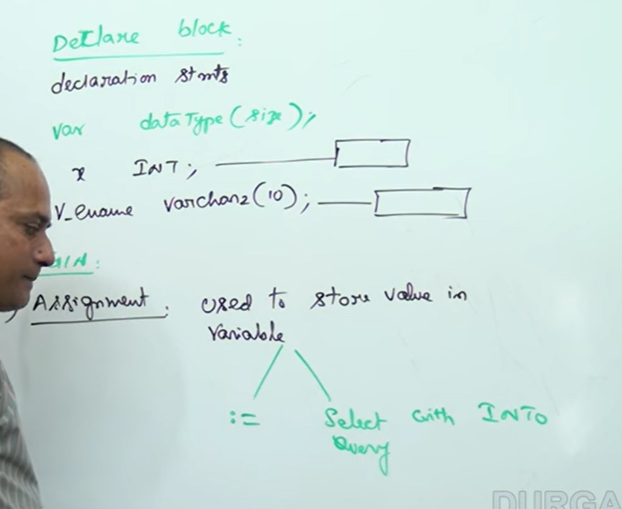
Declaration statement

For example when we want to calculate 2 number

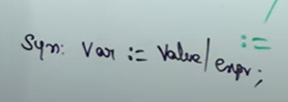
We need to save the values

Display result

Values should be saved under variables – memory space also req and data type



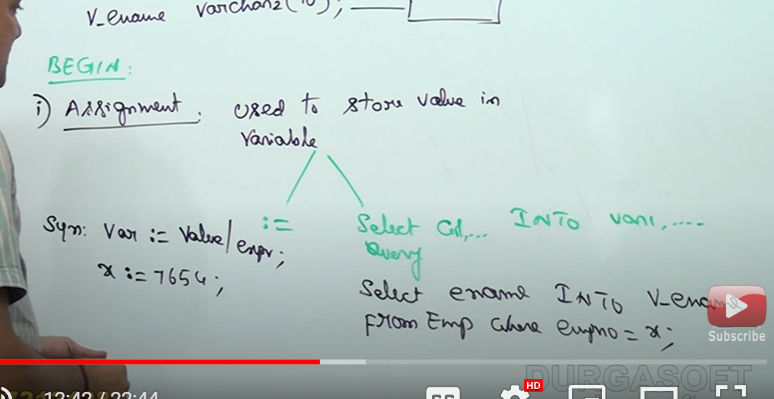
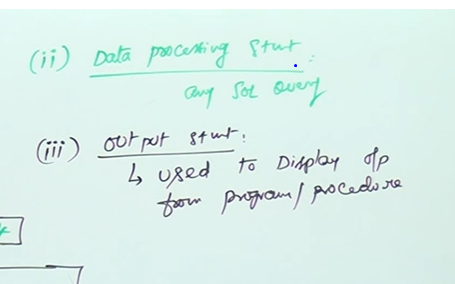
Syntax for assignment operator

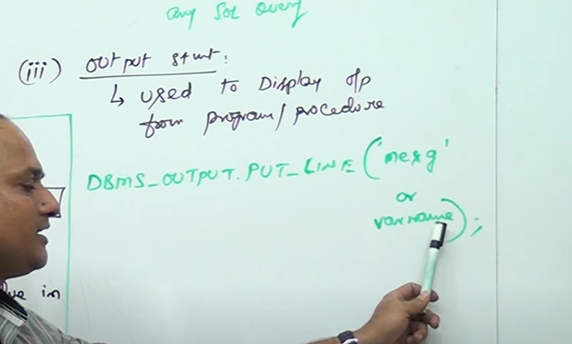


we can write value

or func calling statement

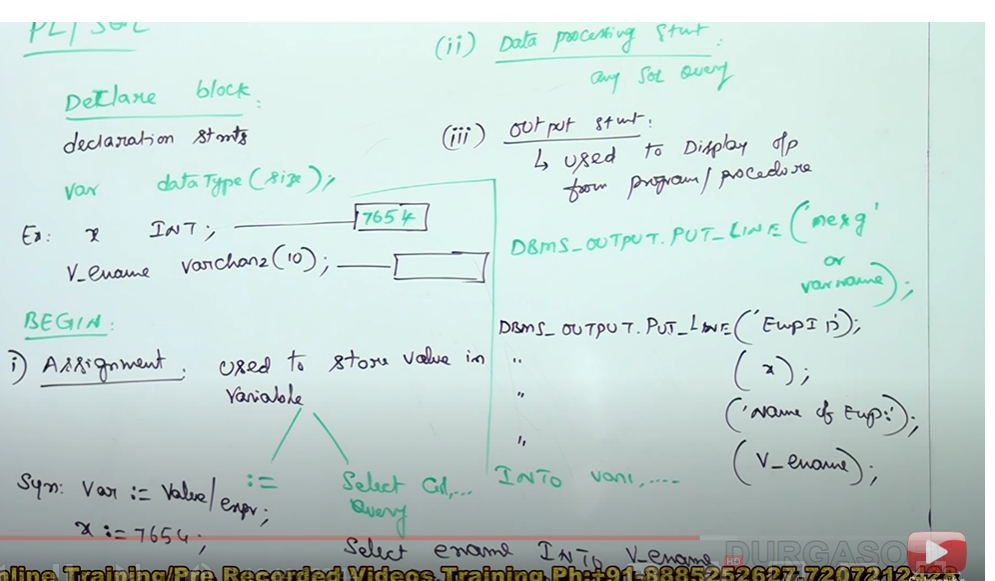
procedure calling statement

* + Select query taking a value from table and assigned to variable / we must save that value into variable
  + 
  + Data processing statement- means any sql query we are using
  + 
  + To display output
  + Oracle is providing pre-defined function



Normal mesg: single code

Variable name : without any codes



Final block :

Exceptional handling : error handling statement

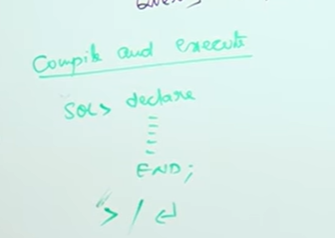
Instead of tech error info; to display user friendly info to client- ex: invalid emplee ID or duplicate Id

Last :

END

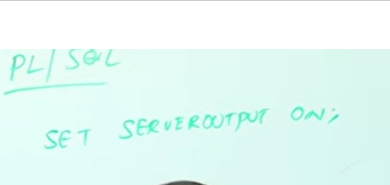
To end the program

Once I finish program how to execute?



Last line – enter slash and enter

Then program will be compiled and executed

* We get mesg “PLSQl code successfully completed “
* We wont get output, we get only mesg
* Any plsql programe : doent display output
* For that we must use
* 
* It enable to display output

SEQUENCES and Synonyms:

**Sequence:**

Sequential number should start with num

We can keep upper limit

Constant increment value

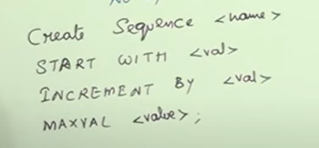
No null values

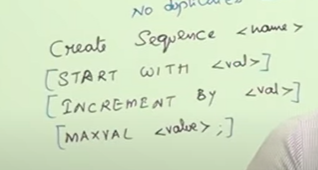
No duplicate value

Primary key column, which is number type can be created to generate sequential integers

We can use sequences to generate primary key values if that primary key is integer type

Syntax for creating sequence:





Everything optional

In such case default value= 1

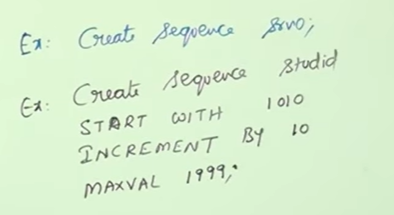
Default increment value =1

Without table we can create any number of seq, sequences are independent object

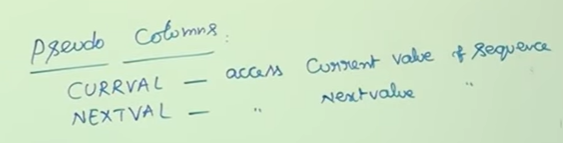
Where as views and cursor are depends on table

Seq: can be created by 2 methods

1. With default values
2. With user defined values

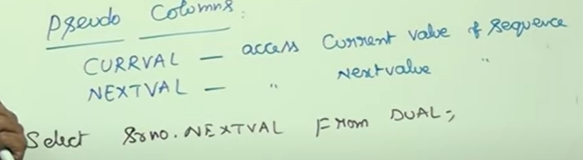


We have 2 psedo columns along with that

1. Curval
2. Nextval
3. 

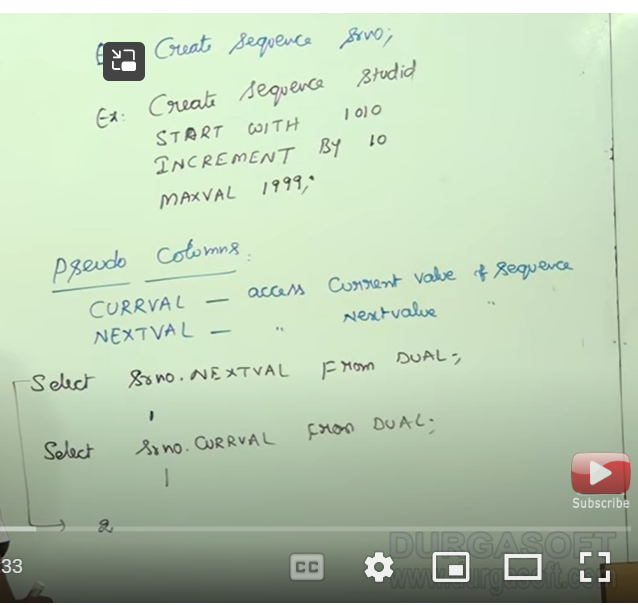
Initially to start sequence – it will start with next value

Then only it can take any number of times the curval and generates next val

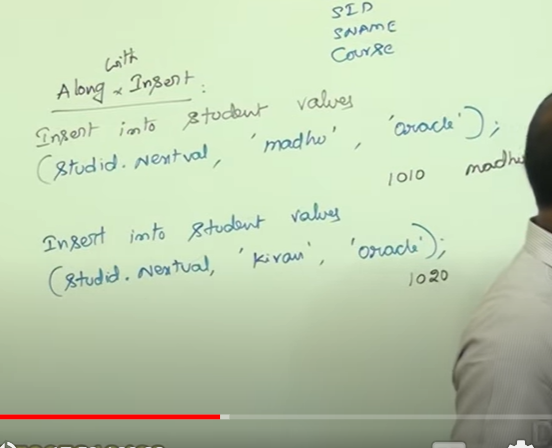


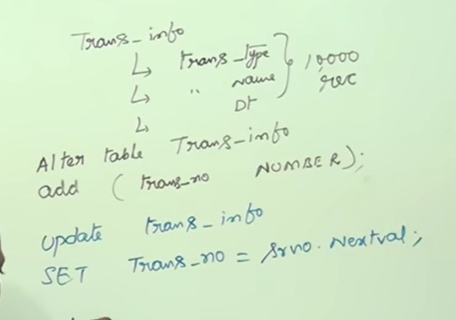
Srno: is seq number name

Dual we mention as we don’t require table for creating seq num



How can I insert record along with seq





System table which maintains sequence:



Delete seq:



Synonym:

Database object

Table name are diff to remember

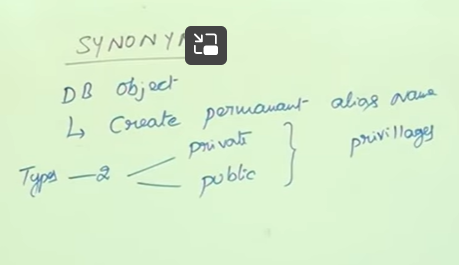
They create short names for

(Valid for temporary – upto execution of query

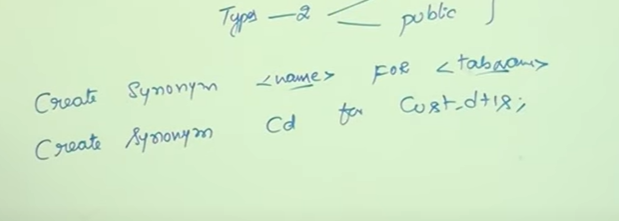
We create during join )

To create a permanent alias name

We can use any transaction by using alias name

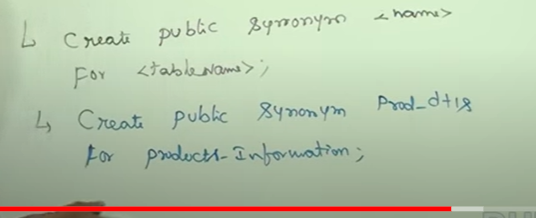
 to create view and synonym we need to get permission from DBA  
private synonym: useful /accessible for that particular uses

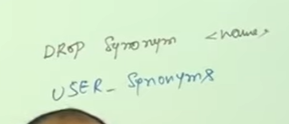
Public: any user can access – created by DBA  
how to create pvt synonym:



By default it is pvt syn

How to create Public?





# PL/SQL Type Compatibility Part - 1