# PL/SQL

DECLARE

<declarations section> -> Name varchar2(20) := ‘ALI’; OR <exception\_name> **EXCEPTION**

BEGIN

<executable command(s)> -> dbms\_output.put\_line(‘NAME:’ || Name);

EXCEPTION

<exception handling> -> WHEN <Exception Name> THEN <action>

END;

* NO\_DATA\_FOUND: Raised when a SELECT INTO statement doesn't return any rows.
* TOO\_MANY\_ROWS: Raised when a SELECT INTO statement returns more than one row.
* ZERO\_DIVIDE: Raised when attempting to divide by zero.
* VALUE\_ERROR: Raised when a conversion or arithmetic operation fails.
* INVALID\_CURSOR: Raised when attempting operations on a closed or invalid cursor.
* LOGIN\_DENIED: Raised when a login to the database fails.
* STORAGE\_ERROR: Raised when there's insufficient memory or storage.
* PROGRAM\_ERROR: Raised for unexpected program errors.
* TIMEOUT\_ON\_RESOURCE: Raised when an operation times out.
* OTHERS: A catch-all exception that can be used to catch any unhandled exception.

To raise exception, we call RAISE <exception\_name>

## Differences in PL/SQL

SELECT DEPARTMENT\_ID

**INTO DEPTID**

FROM EMPLOYEES

WHERE EMPLOYEE\_ID = 10;

IF (<condition>) THEN

<action>

ELSIF (<condition>) THEN

<action>

ELSE

<action>

END IF;

CASE <variable>

WHEN x1 THEN

<action>

WHEN x2 THEN

<action>

ELSE

<action>

END CASE;

FOR x IN 1..<number> LOOP

<action>

END LOOP;

FOR x IN (SELECT EMPLOYEE\_ID FROM EMPLOYEES WHERE DEPARTMENT\_ID = 90) LOOP

<action>

END LOOP;

CREATE OR REPLACE FUNCTION FUNC (VAR\_NAME IN <type>, …)

RETURN <type>

IS

<declarations>

BEGIN

<action>

END;

/

SELECT FUNC(80) FROM DUAL;

CREATE OR REPLACE TYPE EMP\_OBJ\_TYPE AS OBJECT (

EMPLOYEE\_ID NUMBER(6,0),

FIRST\_NAME VARCHAR(30),

LAST\_NAME VARCHAR(30),

DEPARTMENT\_ID NUMBER(4,0)

);

CREATE TYPE EMP\_TBL\_TYPE as TABLE OF EMP\_OBJ\_TYPE;

CREATE OR REPLACE FUNCTION GETALL

RETURN EMP\_TBL\_TYPE

IS

EMPLOYEE\_ID NUMBER(6,0);

FIRST\_NAME VARCHAR(30);

LAST\_NAME VARCHAR(30);

DEPARTMENT\_ID NUMBER(4,0);

**EMP\_DETAILS EMP\_TBL\_TYPE := EMP\_TBL\_TYPE();**

BEGIN

**EMP\_DETAILS.EXTEND();**

SELECT EMPLOYEE\_ID,FIRST\_NAME, LAST\_NAME,DEPARTMENT\_ID INTO

EMPLOYEE\_ID,FIRST\_NAME,LAST\_NAME,DEPARTMENT\_ID FROM EMPLOYEES where EMPLOYEE\_ID=100;

**EMP\_DETAILS(1) := EMP\_OBJ\_TYPE(EMPLOYEE\_ID,FIRST\_NAME,LAST\_NAME,DEPARTMENT\_ID);**

RETURN EMP\_DETAILS;

END;

/

CREATE OR REPLACE FUNCTION GETALL1

RETURN EMP\_TBL\_TYPE

IS

EMPLOYEE\_ID NUMBER(6,0);

FIRST\_NAME VARCHAR(30);

LAST\_NAME VARCHAR(30);

DEPARTMENT\_ID NUMBER(4,0);

EMP\_DETAILS EMP\_TBL\_TYPE := EMP\_TBL\_TYPE();

BEGIN

EMP\_DETAILS.EXTEND();

SELECT EMP\_OBJ\_TYPE( EMPLOYEE\_ID,FIRST\_NAME, LAST\_NAME,DEPARTMENT\_ID) **BULK COLLECT**

INTO EMP\_DETAILS FROM EMPLOYEES;

RETURN EMP\_DETAILS;

END;

/

CREATE OR REPLACE PROCEDURE PROC\_NAME (<params (like func)>)

IS

<declarations>

BEGIN

<action>

END;

/

EXEC PROC\_NAME(<args>)

DECLARE

CURSOR Cursor\_EMP IS SELECT \* FROM employees ORDER BY salary DESC;

row\_emp Cursor\_EMP%ROWTYPE;

BEGIN

OPEN Cursor\_EMP;

LOOP

FETCH Cursor\_EMP INTO row\_emp;

EXIT WHEN Cursor\_EMP%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE( 'EMPLOYEE id: ' ||row\_emp.EMPLOYEE\_ID || ' EMPLOYEE NAME: ' || row\_emp.FIRST\_NAME || ' EMPLOYEE CONTACT: ' || row\_emp.PHONE\_NUMBER || '.');

END LOOP;

CLOSE Cursor\_EMP;

END;

/

CREATE OR REPLACE TRIGGER TRIGGER\_NAME

{BEFORE|AFTER|INSTEAD OF}

{INSERT [OR] | UPDATE [OR] | DELETE}

[OF col\_name]

ON table\_name

[FOR EACH ROW]

WHEN (<condition>)

BEGIN

<actions>

END;

SET TRANSACTION ISOLATION LEVEL [READ COMMITTED | READ UNCOMMITED | SERIALIZABLE | REPEATABLE READ];

SET TRANSACTION READ [WRITE | ONLY];

SET TRANSACTION NAME ‘NAME’;

SET TRANSACTION [DEFERRABLE | NOT DEFERRABLE];

SAVEPOINT SPNAME;

ROLLBACK TO SAVEPOINT SPNAME;

CREATE INDEX <index\_name> ON <table\_name> (column1, column2);

DROP INDEX <index\_name>;

SEE MONGODB LAB