```
-- Anonymer PL/SQL-Block

DECLARE
    -- Variablen und Datentypen
    v_num1 NUMBER(2);
    v_num2 NUMBER(8,2);
    v_name VARCHAR2(20) := 'KING LOUIS';
    v_time DATE := sysdate;
    v_bool BOOLEAN := TRUE;

    -- Referenzierung (wann immer möglich)
    v_empno   emp.empno%TYPE;
    v_job     emp.job%TYP;
    v_emp_rec emp%ROWTYPE;

    -- EXPLIZITER Cursor
    CURSOR c_food IS SELECT * FROM food;
    CURSOR c_emp  IS SELECT * FROM emp WHERE abt = '05';

    -- User Defined Exceptions
    -- Im Gegemsatz zu vordefinierten und anonymen Exceptions
    e_fatal_error EXCEPTION;


BEGIN
    dbms_output.put_line('Go, Charlie, go!');


    ------------------------------------------------------
    ---IMPLIZIERTER Cursor:-------------------------------
    --   o Wird für alle SQL-Statements definiert       --
    --   o Bricht bei SELECT das Programm ab, wenn      --
    --       - Keine Daten gefunden wurden              --
    --       - Mehr wie ein Datensatz gefunden wurde.   --
    ------------------------------------------------------
    ------------------------------------------------------

    SELECT empno, job
      INTO v_empno, v_job
      FROM emp
      WHERE ename = v_name;

    -- Entscheidungen
    IF v_job = 'MANAGER' THEN
        -- Werte aktualisieren
        UPDATE emp
          SET   sal = sal * 1,1
          WHERE empno = v_empno;

    ELSIF v_job = 'WORKER' THEN
        UPDATE emp
          SET   sal = sal * 1,05
          WHERE empno = v_empno;

    ELSE
        -- Eine Exception werfen
        RAISE e_fatal_error;

    END IF;
```

```
------------------------------------------------
---Auslesen eines Satzes mit EXPLIZITEM Cursor----
-- c_food%FOUND                                --
-- c_food%NOTFOUND                             --
-- c_food%ROWCOUNT                             --
-- c_food%ISOPEN                               --
------------------------------------------------
------------------------------------------------

OPEN c_food;

FETCH c_food INTO v_name, v_time;
dbms_output.put_line(v_name || '  MHD: ' to_char(v_time, 'dd.mm.yyyy'));
  -- ...

CLOSE c_food;



----------------------------------------
---EINSCHUB: NÜTZLICHE SPRACHELEMENTE----
--   || Concatinierung                 --
--   to_char(...)                      --
--   length(...)                       --
--   substr(str, pos)  mit pos >= 1    --
----------------------------------------
----------------------------------------


-----------------------------------------------
---SPASS MIT SCHLEIFEN-----------------------
--   o Loop Boundary / Schleifenkop         --
--       - Bestimmung der Loop-Art          --
--       - Definition der Abbruchbedingung  --
--       - Sowie das Statement END LOOP     --
--   o Loop Body                            --
-----------------------------------------------
-----------------------------------------------

-- Basic Loop
LOOP
    v_num1 := v_num1 + 1;
    EXIT WHEN v_num1 = 3;
END LOOP;


-- Numeric For Loop
FOR v_num IN 1 .. 5 LOOP

    -- Ausführen von Prozeduren
    CALL p_proc1;
    EXECUTE p_proc2;

    -- Ausführen einer Funktion
    v_bool := f_func1(...);

END LOOP;


-- Cursor For Loop (Statt OPEN, FETCH, CLOSE)
FOR v_emp_rec IN c_emp LOOP
```

```
        dbms_output.put_line(v_emp_rec.name);
    END LOOP;


    -- Gilt in PL/SQL als unschön
    FOR v_emp_rec IN SELECT * FROM emp WHERE name = 'Heinz'
        -- ...
    END LOOP;


EXCEPTION
    WHEN fatal_error THEN
        dbms_output('Was ist denn jetzt schon wieder?');

    -- Immer am Schluss:
    WHEN OTHERS THEN
        dbms_output('Irgendwas ist immer!');

    ---------------------------------------------
    ---------------------------------------------
    -- Kein Rücksprung in den Block möglich.  --
    -- Block wird verlassen                   --
    -- Darum: Geschachtelte Blöcke anwenden,  --
    --        um Exception abzufangen!        --
    -- Bei ungefangener Exception: ABBRUCH    --
    ---------------------------------------------
    ---------------------------------------------
END;


-----------------------------------------------
-- STORED OBEJCTS------------------------------
-- Stored Procedures                          --
--  o Ein Datenbankobjekt, wie Tabelle auch.  --
--  o Enthält:                                --
--     - Quelltext                            --
--     - Procedure im parsed format           --
--     - Meta-Code Compilat                   --
--     - Fehlermeldungen bei Syntaxfehlern    --
--     - Status: valid / invalid              --
--  o CREATE PROCEDURE  (ggf. Fehlermeldung)  --
--  o REPLACE PROCEDURE (ggf. Fehlermeldung)  --
--  o CREATE OR REPLACE PROCEDURE             --
-----------------------------------------------
-----------------------------------------------

CREATE OR REPLACE PROCEDURE p_proc1
IS          -- oder AS. Eins von beiden MUSS da sein!
BEGIN
EXCEPTION
END;

-- GRANT EXECUTE ON p_proc1 FOR user24;


CREATE OR REPLACE PROCEDURE p_proc2
(
  in_deptid IN NUMBER
  io_misc IN OUT VARCHAR2
  out_found OUT BOOLEAN
)
```

```
IS
    v_str VARCHAR2(20);

BEGIN
EXCEPTION
END;


CREATE OR REPLACE FUNCTION f_func1
(...)
RETURN BOOLEAN -- VARCAHR2, NUMBER, ...
IS
BEGIN

    RETURN TRUE;

EXCEPTION
END;
```