MP02B BASES DE DATOS

ESQUEMA RESUMIDO DE LOS RESUMENES PAL EXAMEN

VISTAS

CREATE VIEW *nombrevista* AS SELECT *columnas* FROM *tabla* WHERE *condicion*;

USUARIOS

CREATE USER *usuario* IDENTIFIED BY *contraseña*;
DROP USER *usuario*;
GRANT/REVOKE create session TO/FROM *usuario*;
GRANT/REVOKE all privileges TO/FROM *usuario*;
GRANT/REVOKE select/insert/delete/update ON *nombretabla* TO/FROM *usuario* [WITH GRANT OPTION];
GRANT/REVOKE *1soloPermiso* ANY TABLE TO/FROM *usuario*;

Para otorgar permisos sobre columnas especificas de una tabla, primero hay que crear una vista con esas columnas, y luego otorgar permisos sobre esa vista.

BLOQUES ANÓNIMOS

```
DECLARE
   variables
BEGIN
   [-----] <--Esto lo uso para indicar bloques generales de código
END;
```

PROCEDIMIENTOS

```
CREATE OR REPLACE PROCEDURE nombreprocedimiento[(parametros)] AS
   variables
BEGIN
   [----]
EXCEPTION
   [----]
END;
```

-Para llamar a un procedimiento se usa CALL *nombreprocedimiento*();
-Los parametros se pasan (*nombreparametro* IN/OUT/IN OUT *tipoparametro*)

FUNCIONES

```
CREATE OR REPLACE FUNCTION nombrefuncion[(parametros)] RETURN tiporetorno AS
   variables
BEGIN
   [----]
EXCEPTION
   [----]
END;
```

TIPOS DE DATOS MAS USADOS

NUMBER-CHAR-VARCHAR-VARCHAR2-BOOLEAN-DATE-%TYPE-%ROWTYPE

DECLARACION VARIABLES

nombrevariable [CONSTANT] *tipodato* [NOT NULL] [:=*valor*];

ASIGNACION VARIABLES

nombrevariable := valor;
SELECT *consultaSQL* INTO *nombrevariable* [FROM DUAL];
nombrevariable :=&texto;

Sentencias de asignación concretas que se han usado en las PACS

SELECT EXTRACT(YEAR FROM(SYSDATE)) INTO *nombrevariable* FROM DUAL;
SELECT FLOOR(MONTHS_BETWEEN(SYSDATE,*fecha*)/12) INTO *edad* FROM DUAL;

SENTENCIAS A CONOCER

SET SERVER OUTPUT ON
DBMS_OUTPUT.PUT_LINE(*texto*);

SENTENCIAS DE CONTROL Y BUCLES

IF … THEN
  [----]
ELSIF … THEN
  [----]
ELSE
  [----]
END;
	CASE *variable*
	  WHEN … THEN
	  [----]
	  WHEN … THEN
	  [----]
	  ELSE
	  [----]
	END;
		WHILE … LOOP
		  [----]
		END;

LOOP
  [----]
  EXIT WHEN …;
END LOOP;
	LOOP
	  [----]
	  IF … THEN
	  EXIT;
	  END IF;
	END LOOP;
		FOR *indice* IN [REVERSE]
		valorinicial … *valorfinal*
		LOOP
		  [----]
		END LOOP;

OJO! Que todos los … indican condicionales menos los del bucle for, que en la sentencia se usa literalmente.

CURSORES 1:

```
DECLARE
  CURSOR nombrecursor IS sentenciaSQL select;
BEGIN
  FOR variableQuevaserunRegistro IN nombrecursor LOOP
    [----]
  END LOOP;
END;
```

CURSORES 2

```
DECLARE
   CURSOR nombrecursor is sentenciaSQL select;
BEGIN
   OPEN nombrecursor;
      LOOP
         FETCH nombrecursor INTO variablesquesecorrespondanconloscamposdelselect;
            EXIT WHEN nombrecursor%NOTFOUND;
               [----]
       END LOOP;
   CLOSE nombrecursor;
END;
```

ATRIBUTOS DE LOS CURSORES

%ISOPEN - %NOTFOUND - %FOUND - %ROWCOUNT

EXCEPCIONES

```
DECLARE
   nombreexcepcion EXCEPTION;
BEGIN
   RAISE nombreexcepcion;
EXCEPTION
   WHEN nombreexcepcion THEN
      [----]
END;
```

+En las excepciones predefinidas, no hace falta declararlas ni lanzarlas.
+Principales excepcines predefinidas que podemos necesitar:
-NO_DATA_FOUND -TOO_MANY_ROWS – INVALID_NUMBER – OTHERS
+Si no queremos declarar una excepcion y no la vamos a manejar, tambien pordemos usar
-RAISE_APPLICATION_ERROR(-20000,'*texto a mostrar*');

DISPARADORES

```
CREATE OR REPLACE TRIGGEER nombredisparador
    AFTER/BEFORE INSERT OR DELETE OR [UPDATE [OF nombrecolumnas]]
       ON nombretablas [FOR EACH ROW] [WHEN sentenciacondicional]
DECLARE
   variables
BEGIN
   [----]
END;
```

+Se usa IF *INSERTING/DELETING/UPDATING THEN* cuando queremos saber el tipo de operación que ha activado el disparador.
+Se usa :OLD.*columna* :NEW.*columna* para acceder a los valores antiguos y nuevos del registro.

BBDDOR

CREACION DE UN TIPO DE OBJETO (CLASES)

```
CREATE OR REPLACE TYPE nombreclase AS OBJECT(
   atributosvarios tipoatributos,
   MEMBER FUNCTION nombremetodo[(parametros)] [RETURN tiporetorno],
   ORDER MEMBER FUNCTION compareTo(e IN nombreclase) RETURN NUMBER
);
```

CREACION DEL CUERPO

```
CREATE OR REPLACE TYPE BODY nombreclase AS

   MEMBER FUNCTION nombremetodo[(parametros)] RETURN tiporetorno IS
   variables;
   BEGIN
      [----]
      [RETURN retorno;]
   END;

   ORDER MEMBER FUNCTION compareTo(e IN nombreclase) RETURN NUMBER IS
   variables;
   BEGIN
      [----]
      RETURN retorno;
   END;
END;
```

INSTANCIAR UN OBJETO (CONSTRUCTORES)

```
DECLARE
   nombreobjeto nombreclase;
BEGIN
   nombreobjeto := NEW nombreclase(valoratributo1,valoratributo2,....valoratributoN);
   INSERT INTO tablaquesea VALUES(campo1,...campoN, nombreclase(nombreobjeto));
   INSERT INTO tablaquesea VALUES(campoN, NEW nombreclase(valoratributos));
END;
```

ARRAYS

CREACION DEL TIPOARRAY

```
CREATE TYPE clasearray AS VARARRAY(tamaño) OF tipodato;
```

USO EN TABLA

```
CREATE TABLE nombretabla(
   atributosvarios tipoatributos,
   campomultivaluado clasearray
);
```

INSERCION DE DATOS

```
INSERT INTO nombretabla VALUES(atributosvarios,
                               clasearray(tipodato1(valor1),
                                          tipodatoN(valorN)
                                         )
                               );
```

TABLAS ANIDADAS

CREACION DEL TIPOTABLE

```
CREATE TYPE clasetabla AS TABLE OF tipodato;
```

USO EN TABLA

```
CREATE TABLE nombretabla(
   atributosvarios tipoatributos,
   campomultivaluado clasetabla,
   NESTED TABLE campomultivaluado STORE AS alias
);
```

INSERCION DE DATOS

```
INSERT INTO nombretabla VALUES(atributos varios,
                               clasetabla(tipodato1(valor1),
                                          tipodatoN(valorN)
                                          )
                               );
```