

# SQL El lenguaje de Consulta Estructurado

Operaciones de Manipulación,  
Vistas, Metadatos

# Operaciones de Manipulación

- SQL proporciona operaciones para manipular la información contenida en las tablas:
  - **INSERT**: permite insertar nuevas filas en las tablas.
  - **UPDATE**: modifica la información de filas existentes
  - **DELETE**: borra filas de las tablas
- También permite definir **transacciones**, que son conjuntos de operaciones de manipulación que se tratan atómicamente, como si fueran una única operación. Las operaciones pertenecientes a una transacción se ejecutan todas, o bien, no se ejecuta ninguna, es decir, se deja al sistema en el estado previo a la ejecución de la primera operación de la transacción.

## Operación de inserción

- Para añadir una fila a una tabla se utiliza  
`INSERT INTO tabla VALUES fila_a_insertar`
- Para añadir un conjunto de filas a una tabla se utiliza  
`INSERT INTO tabla Subconsulta`
- Si el orden de las columnas en *fila\_a\_insertar* o en el resultado de Subconsulta es diferente al orden de las columnas de tabla, hay que indicar el orden en el que vienen los datos  
`tabla(col1, col2, col3, ..., coln)` indica que la primera columna de *fila\_a\_insertar* o Subconsulta se insertará en col1 de la tabla, la segunda columna en col2, ...

## Operación de inserción

- Matricular al alumno con dni 12127891 en el grupo A de la asignatura con código 112 del curso 16/17

```
INSERT INTO matricular VALUES ('12127891',112,'A','16/17',null)
```

- Matricular a todos los alumnos en el grupo B de la asignatura con código 111 del curso 16/17

```
INSERT INTO Matricular (Asignatura, Grupo, Curso, Alumno)  
SELECT 111, 'B','16/17', dni  
FROM Alumnos ;
```

Los atributos no indicados se asignan al valor nulo o al valor por defecto definido para ellos si existe.

## Operación de modificación

- Para modificar datos de las filas se utiliza

**UPDATE** *tabla*

**SET** *atributo1 = expr1, atributo2 = expr2,... atributon = exprn*

**WHERE** *condición*

- Sólo las filas que satisfacen la condición serán modificadas.
- La parte WHERE es opcional. Si no existe, se modifican todas las filas.
- Podemos cambiar más de un atributo de una misma fila. Para cada uno de ellos hay que indicar el nuevo valor que tomará.
- Se pueden utilizar subconsultas tanto en la parte WHERE como en la parte SET.

## Operación de modificación

- Dar aprobado general a los alumnos del grupo B de la asignatura con código 112 del curso 15/16

```
UPDATE Matricular  
SET calificacion = 'AP'  
WHERE grupo = 'B' and asignatura = 112 and curso ='15/16';
```

- Asignar 6 créditos a todas las asignaturas que tengan grupo B

```
UPDATE asignaturas  
SET credits = 6  
WHERE codigo IN ( SELECT asignatura  
                  FROM matricular  
                  WHERE grupo = 'B'  
                );
```

## Operación de modificación

- Asignar el despacho 2-2-43-A y el teléfono 2755 a todos los profesores del departamento de código 3.

```
UPDATE profesores  
SET telefono = '2755', despacho = '2-2-43-A'  
WHERE departamento = 3;
```

- Cambiar el número de créditos de cada asignatura por el número de matrículas que se han realizado de esa asignatura.

```
UPDATE asignaturas asig  
SET credits = ( SELECT COUNT(*)  
                FROM matricular  
                WHERE asignatura = asig.codigo ) ;
```

## Operación de eliminación

- Para eliminar filas de una tabla se utiliza

**DELETE FROM** *tabla*

**WHERE** *condición*

- Sólo las filas que satisfacen la condición serán borradas.
- La parte WHERE es opcional. Si no existe, se borran todas las filas.
- Se pueden utilizar subconsultas en la parte WHERE.



## Operación de modificación

- Eliminar todas las matrículas de la asignatura con código 112.

```
DELETE FROM matricular  
WHERE asignatura = 112;
```

- Borrar las asignaturas que no tienen a nadie matriculado.

```
DELETE FROM asignaturas asig  
WHERE NOT EXISTS (  
    SELECT *  
    FROM matricular  
    WHERE asignatura = asig.codigo  
);
```

## Transacciones

- En Oracle una transacción se puede definir mediante un conjunto de sentencias dentro de un bloque PL/SQL (lenguaje de programación incrustado en Oracle)

BEGIN

sentencias

END

- Las sentencias del bloque deben terminar con
  - **COMMIT**. Se confirman los cambios y se guardan en la base de datos.
  - **ROLLBACK**. Se deshacen los cambios y se vuelve al estado de la base de datos previo.

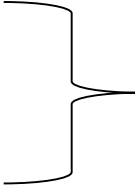
## Transacciones. Ejemplo

```
DECLARE
  importe NUMBER;  ctaOrigen VARCHAR2(23);  ctaDestino VARCHAR2(23);
BEGIN
  importe := 100;
  ctaOrigen := '2530 10 2000 1234567890';
  ctaDestino := '2532 10 2010 0987654321';

  UPDATE CUENTAS SET SALDO = SALDO - importe WHERE CUENTA = ctaOrigen;
  UPDATE CUENTAS SET SALDO = SALDO + importe WHERE CUENTA = ctaDestino;
  INSERT INTO MOVIMIENTOS VALUES (ctaOrigen, ctaDestino, importe*(-1), SYSDATE);
  INSERT INTO MOVIMIENTOS VALUES (ctaDestino, ctaOrigen, importe, SYSDATE);

  COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    dbms_output.put_line('Error en la transaccion:' || SQLERRM);
    dbms_output.put_line('Se deshacen las modificaciones');
    ROLLBACK;
END;
```

## Elementos del nivel externo

- En el nivel externo se determinan las parcelas de información del esquema conceptual a las que los usuarios de la base de datos tienen acceso.
  - CREATE
  - ALTER
  - DROP

**VIEW** permiten crear, modificar y borrar vistas para los usuarios.
- También hay que indicar qué operaciones pueden realizar los usuarios sobre los elementos del nivel conceptual. Es decir hay que definir permisos.
  - **GRANT**, concede permisos a los usuarios
  - **REVOKE**, revoca permisos de los usuarios.

# Vistas

- Una vista se puede considerar una tabla virtual que se basa en una o más tablas o vistas

**CREATE VIEW** *nombreVista (atributo1, atributo2, ..., atributon)*  
**AS** *subconsulta\_de\_definición.*

- La subconsulta\_de\_definición determina las columnas que formarán parte de la vista.
  - Si ninguna de dichas columnas es el resultado de aplicar funciones u operaciones aritméticas, entonces no tendremos que especificar nombres de atributos. Se utilizan los mismos que en las tablas en las que se basa.
- Se puede utilizar como una tabla más.
- La vista siempre muestra la información actualizada.
  - Cambios en las tablas base se reflejan inmediatamente en la vista.

## Vistas

- Crear una vista que muestre el código junto al año de entrada en la universidad de los alumnos cuyo primer apellido comienza por la letra M.

```
CREATE VIEW Alumnos_con_Experiencia(Codigo, Cuando) AS  
(SELECT Dni, TO_CHAR(Fecha_Prim_Matricula,'yyyy')  
FROM Alumnos  
WHERE Apellido1 LIKE 'M%' );
```

- Usar la vista Alumnos\_con\_Experiencia para mostrar la información de los alumnos con más de 2 años de antigüedad en la universidad.

```
SELECT *  
FROM Alumnos_con_Experiencia  
WHERE to_number(to_char(sysdate,'yyyy')) - to_number(Cuando) > 2;
```

CODIGO	CUANDO
23456456	2014
12312342	2013
25423411	2014
62512427	2014
38260925	2014
47345210	2014
36982815	2013
28294639	2013
29707760	2014
12127891	2013
31333317	2014

## Vistas. Opciones

```
CREATE [OR REPLACE] [FORCE] VIEW nombreVista (atr1, atr2, ..., atrn)  
AS subconsulta_de_definición  
[WITH READ ONLY]  
[WITH CHECK OPTION]
```

- OR REPLACE: permite sobrescribir la vista si ya existe.
- FORCE: crea la vista aunque las tablas base no existan o no se tengan permisos sobre ellas.
- WITH READ ONLY: impide que se pueda modificar la vista.
- WITH CHECK OPTION: impide modificar las filas de la vista si el resultado no coincide con alguna de las filas que devolvería la subconsulta.

## Actualización de vistas

- Podemos realizar operaciones de actualización del contenido de una vista. Dichos cambios serán propagados a las tablas base correspondientes si es posible.
- Para que una vista sea actualizable debe cumplir:
  - Sus columnas deben hacer referencia a columnas de alguna tabla base (no pueden ser el resultado de alguna expresión)
  - La subconsulta no puede tener:
    - Operaciones de conjunto
    - DISTINCT
    - Funciones de agregación
    - GROUP BY
    - ORDER BY.
    - Reuniones



## Actualización de vistas

- Crear una vista que muestre el la información de las asignaturas con código menor de 130.

```
CREATE VIEW asignaturas_menor_130  
AS select * from asignaturas where codigo < 130;
```

- Insertar, usando la vista, la asignatura Programación de Videojuegos, con código 1 y el resto de los valores a null o al valor definido por defecto.

```
INSERT INTO asignaturas_menor_130(codigo,nombre)  
VALUES (1, 'Programación de Videojuegos');  
COMMIT;
```

- Mostrar el contenido de la vista

```
SELECT * FROM asignaturas_menor_130;
```

CODIGO	NOMBRE	...
1	Programación de Videojuegos	...
101	Programación Orientada a Objetos	...
110	Sistemas Operativos	...
111	Estadística	...
112	Bases de Datos	...
113	Calculo Numerico	...
114	Teoria de Automatas	...
115	Administración de Bases de Datos	...
116	Logica Computacional	...
122	Estructura de Computadores	...
123	Computación Altas Prestaciones	...

## Actualización de vistas

- Insertar la asignatura 'Aprendizaje Computacional' con código 300

```
INSERT INTO asignaturas_menor_130(codigo,nombre)  
VALUES (300, 'Aprendizaje Computacional');  
COMMIT;
```

- Mostramos la vista

```
SELECT * FROM asignaturas_menor_130;
```

- ¡¡Pero la nueva asignatura se ha insertado!!

```
SELECT * FROM asignaturas;
```

CODIGO	NOMBRE	...
1	Programación de Videojuegos	...
101	Programación Orientada a Objetos	...
110	Sistemas Operativos	...
111	Estadística	...
112	Bases de Datos	...
113	Calculo Numerico	...
114	Teoria de Automatas	...
115	Administración de Bases de Datos	...
116	Logica Computacional	...
122	Estructura de Computadores	...
123	Computación Altas Prestaciones	...

CODIGO	NOMBRE	...
200	Teoria de la señal	...
140	Prácticas en empresa	...
113	Calculo Numerico	...
114	Teoria de Automatas	...
115	Administración de Bases de Datos	...
116	Logica Computacional	...
112	Bases de Datos	...
110	Sistemas Operativos	...
122	Estructura de Computadores	...
133	Ingeniería Web	...
134	Dispositivos Electronicos	...
144	Modelos Computacionales	...
123	Computación Altas Prestaciones	...
111	Estadística	...
101	Programación Orientada a Objetos	...
201	Matemática Discreta	...
1	Programación de Videojuegos	...
300	Aprendizaje Computacional	...

## Actualización de vistas

- Para evitarlo utilizamos la opción WITH CHECK OPTION

```
CREATE OR REPLACE VIEW asignaturas_menor_130  
AS  
select * from asignaturas where codigo < 130  
WITH CHECK OPTION;
```

- Ahora al insertar una fila que la subconsulta no devolvería (código mayor o igual a 130) el SGBD devuelve un error.

Informe de error -

Error SQL: ORA-01402: violación de la cláusula WHERE en la vista WITH CHECK OPTION  
01402. 00000 - "view WITH CHECK OPTION where-clause violation"

- Por último, borramos las filas introducidas en el ejemplo mediante la vista.

```
DELETE FROM asignaturas_menor_130 where codigo =1 or codigo = 300;  
COMMIT;
```

## Permisos

- Tipos de permisos:
  - SELECT, permiso de lectura
  - INSERT, permiso de inserción de filas nuevas
  - UPDATE, permiso de modificación de filas
  - DELETE, permiso de eliminación de filas.
- Conceder permisos sobre una tabla/vista a un usuario:  
**GRANT** *permisos* ON *tabla* **TO** *usuario* [*WITH GRANT OPTION*];
  - La opción WITH GRANT OPTION permite al usuario receptor del permiso concederlo a otros usuarios.
- Quitar permisos sobre una tabla/vista a un usuario:  
**REVOKE** *permisos* ON *tabla* **FROM** *usuario*

## Permisos

- Dar permisos de lectura e inserción al usuario ENCISO sobre la vista ALUMNOS\_CON\_EXPERIENCIA

```
GRANT SELECT, INSERT ON Alumnos_con_Experiencia  
TO enciso ;
```

- Quitar el permiso de lectura de la tabla matricular a todos los usuarios

```
REVOKE SELECT ON matricular  
TO public;
```

# Metadatos

- Los **metadatos** son datos que describen otros datos. En el ámbito de las bases de datos, los metadatos son datos almacenados en tablas del sistema que describen la estructura de los elementos de la base de datos (tablas, columnas, restricciones, vistas,...)
- El conjunto de tablas que contienen metadatos se denomina **diccionario de datos** o catálogo del sistema.
- En Oracle las tablas del diccionario de datos sólo son leídas y modificadas por el SGBD.
- Sin embargo, existen una serie de vistas públicas que los usuarios pueden utilizar para consultar información del diccionario de datos.

## Vistas del diccionario de datos

- El nombre de las vistas del diccionario de datos comienza por alguno de los siguientes prefijos:
  - **ALL**, vista que describe elementos a los que el usuario tiene acceso.
  - **DBA**, vista que describe elementos de la base de datos completa. Sólo los administradores tienen permiso para utilizar estas vistas.
  - **USER**, vista que describe elementos creados por el usuario.
- Algunas de las vistas de usuario son
  - USER\_TABLES, USER\_TAB\_COLUMNS
  - USER\_CONSTRAINTS, USER\_CONS\_COLUMNS
  - USER\_VIEWS
- Todas tienen sus vistas “hermanas” para administradores (dba) y acceso extendido (all)

# Metadatos sobre tablas

- **ALL\_TABLES** describe las tablas a las que tiene acceso el usuario

Algunos atributos interesantes son

- TABLE\_NAME, nombre la tabla
- TABLESPACE\_NAME, espacio de tablas en el que se creó la tabla
- OWNER, el creador de la tabla

- **USER\_TABLES** describe las tablas creadas por el usuario

Tiene los mismos atributos que ALL\_TABLES excepto OWNER.

- ¿Cómo se obtendría el nombre de todas las tablas a las que tiene acceso el usuario actual y que no ha creado él?
- ¿Cómo podemos borrar todas las tablas que hemos creado?



# Metadatos sobre tablas

- **USER\_TAB\_COLUMNS** describe las columnas de las tablas del usuario

Atributos:

- TABLE\_NAME, nombre de la tabla o vista que contiene la columna
- COLUMN\_NAME , nombre de la columna
- DATA\_TYPE, tipo de datos de la columna
- DATA\_LENGTH, longitud del tipo de datos en bytes
- DATA\_PRECISION, DATA\_SCALE: precisión y longitud del tipo NUMBER
- NULLABLE, obligatoriedad del atributo
- COLUMN\_ID, orden del atributo al crearse la tabla.
- ...

## Metadatos sobre tablas

- Simular el comando DESC para la tabla profesores.

```
SELECT column_name "Nombre", decode(nullable,'N','NOT NULL',' ') "Nulo",  
       data_type || decode(data_type,'DATE',' ','NUMBER','(' || data_precision || ')','(' || data_length || ')') "Tipo"  
FROM user_tab_columns  
WHERE upper(table_name) = 'PROFESORES';
```

DESC profesores;

Nombre	Nulo	Tipo
-----	-----	-----
ID	NOT NULL	VARCHAR2 (20)
NOMBRE	NOT NULL	VARCHAR2 (20)
APELLIDO1	NOT NULL	VARCHAR2 (20)
APELLIDO2		VARCHAR2 (20)
DEPARTAMENTO	NOT NULL	NUMBER (3)
TELEFONO		VARCHAR2 (4)
EMAIL		VARCHAR2 (100)
DESPACHO		VARCHAR2 (10)
FECHA_NACIMIENTO		DATE
ANTIGUEDAD		DATE
DIRECTOR_TESIS		VARCHAR2 (20)

Nombre	Nulo	Tipo
ID	NOT NULL	VARCHAR2 (20)
NOMBRE	NOT NULL	VARCHAR2 (20)
APELLIDO1	NOT NULL	VARCHAR2 (20)
APELLIDO2		VARCHAR2 (20)
DEPARTAMENTO	NOT NULL	NUMBER (3)
TELEFONO		VARCHAR2 (4)
EMAIL		VARCHAR2 (100)
DESPACHO		VARCHAR2 (10)
FECHA_NACIMIENTO		DATE
ANTIGUEDAD		DATE
DIRECTOR_TESIS		VARCHAR2 (20)

## Metadatos sobre restricciones

- **USER\_CONSTRAINTS** describe las restricciones creadas por el usuario

Atributos:

- CONSTRAINT\_NAME Nombre (único) de la restricción
- TABLE\_NAME, tabla que contiene la restricción
- CONSTRAINT\_TYPE, tipo de restricción
- SEARCH\_CONDITION, predicado si restricción de tipo 'C' (Check)
- R\_CONSTRAINT\_NAME, restricción que define la clave destino si restricción de tipo 'R' (clave foránea)
- ...

## Metadatos sobre restricciones

- Extraer las parejas de tablas tales que la primera componente tiene una clave foránea hacia la segunda.

```
SELECT r1.table_name, r2.table_name  
FROM user_constraints r1, user_constraints r2  
WHERE r1.r_constraint_name = r2.constraint_name;
```

TABLE_NAME	TABLE_NAME_1
CORREQUISITOS	ASIGNATURAS
CORREQUISITOS	ASIGNATURAS
IMPARTIR	ASIGNATURAS
MATRICULAR	ASIGNATURAS
PRERREQUISITOS	ASIGNATURAS
PRERREQUISITOS	ASIGNATURAS
MATRICULAR	ALUMNOS
ASIGNATURAS	DEPARTAMENTOS
PROFESORES	DEPARTAMENTOS
IMPARTIR	PROFESORES
INVESTIGADORES	PROFESORES
PROFESORES	PROFESORES
ASIGNATURAS	MATERIAS
MUNICIPIO	PROVINCIA
ALUMNOS	MUNICIPIO

# Metadatos sobre restricciones

- **USER\_CONS\_COLUMNS** describe columnas que aparecen en las restricciones creadas por el usuario

## Atributos

- CONSTRAINT\_NAME, nombre de la restricción en la que aparece la columna
- TABLE\_NAME, nombre de la tabla en la que aparece la columna
- COLUMN\_NAME, nombre de la columna que aparece en la restricción
- POSITION, posición de la columna en la definición de la restricción

## Metadatos sobre restricciones

- Mostrar los atributos que forman parte de la clave primaria de la tabla Matricular.

```
SELECT column_name
FROM user_cons_columns RCOL, user_constraints R
WHERE RCOL.constraint_name = R.constraint_name AND
      R.constraint_type = 'P' AND upper(R.table_name) = 'MATRICULAR';
```

COLUMN_NAME
ALUMNO
ASIGNATURA
GRUPO
CURSO

- Determinar a qué atributos referencian las claves foráneas definidas en la tabla Matricular. Mostrar Atr\_origen, Atr\_destino, Tabla\_destino.

```
SELECT RCOL1.column_name, RCOL2.column_name, RCOL2.table_name
FROM user_cons_columns RCOL1, user_constraints R1, user_constraints R2, user_cons_columns RCOL2
WHERE RCOL1.constraint_name = R1.constraint_name AND
      R1.r_constraint_name = R2.constraint_name AND
      R2.constraint_name = RCOL2.constraint_name and
      R1.constraint_type = 'R' AND R1.table_name = 'MATRICULAR';
```

COLUMN_NAME	COLUMN_NAME_1	TABLE_NAME
ASIGNATURA	CODIGO	ASIGNATURAS
ALUMNO	DNI	ALUMNOS