

# UT4 - SQL (DDL en Oracle) - Resumen para estudiar

Idea clave: DDL define reglas/estructura; DML prueba si se cumplen (inserts que pasan y que fallan).

## 1) SQL en 30 segundos

- DML  
datos: **SELECT, INSERT, UPDATE, DELETE**
- DDL  
estructura/objetos: **CREATE, ALTER, DROP, TRUNCATE, RENAME**
- DCL/TCL  
permisos + transacciones: **GRANT, REVOKE, COMMIT, ROLLBACK**

## 2) Conceptos (Oracle)

- **Usuario:** cuenta con permisos.
- **Esquema:** objetos de un usuario (tablas, vistas, índices...).
- **Objetos:** TABLE, VIEW, INDEX, SYNONYM.

## 3) CREATE TABLE (plantillas)

```
CREATE TABLE TABLA (  
    COL1 TIPO,  
    COL2 TIPO  
);
```

```
CREATE TABLE DEPT (  
    DEPTNO  NUMBER(2)      CONSTRAINT DEPT_PK PRIMARY KEY,  
    DNOMBRE VARCHAR2(30)  CONSTRAINT DEPT_DNOMBRE_NN NOT NULL,  
    LOC     VARCHAR2(30)  DEFAULT 'MADRID',  
    ACTIVO  CHAR(1)  DEFAULT 'S'  CONSTRAINT DEPT_ACTIVO_CK CHECK (ACTIVO  
IN ('S', 'N'))  
);
```

## 4) Constraints (lo mas importante)

- **NOT NULL (NN)**: obliga a valor.
- **UNIQUE (UQ)**: no repetidos (suele permitir NULL).
- **PRIMARY KEY (PK)**: identifica fila = UQ + NN (solo 1 PK por tabla).
- **FOREIGN KEY (FK)**: relacion entre tablas (padre debe tener PK/UQ).
- **CHECK (CK)**: valida regla (rangos, IN...).
- **DEFAULT**: valor si no se indica en el INSERT.

**Columna vs Tabla**: columna (1 col) / tabla (compuestas, al final).

**FK y ON DELETE**: **CASCADE** borra hijos; **SET NULL** pone FK a NULL (no puede ser NOT NULL).

## 5) ALTER TABLE (operaciones tipicas)

```
ALTER TABLE T ADD (NUEVA_COL VARCHAR2(50));
ALTER TABLE T MODIFY (NUEVA_COL VARCHAR2(120));
ALTER TABLE T RENAME COLUMN VIEJA TO NUEVA;
ALTER TABLE T DROP COLUMN NUEVA_COL;
```

```
ALTER TABLE T ADD CONSTRAINT T_EMAIL_UQ UNIQUE (EMAIL);
ALTER TABLE T ADD CONSTRAINT T_SAL_CK CHECK (SALARIO >= 0);
ALTER TABLE T DROP CONSTRAINT T_EMAIL_UQ;
ALTER TABLE T DISABLE CONSTRAINT T_SAL_CK;
ALTER TABLE T ENABLE CONSTRAINT T_SAL_CK;
```

Error tipico: **MODIFY ... NOT NULL** falla si ya hay filas con NULL.

## 6) DROP vs TRUNCATE vs RENAME

- **DROP TABLE**: elimina la tabla (estructura). Tip: **CASCADE CONSTRAINTS**.
- **TRUNCATE TABLE**: elimina TODAS las filas, mantiene estructura (rapido).
- **RENAME**: renombra un objeto.

## 7) Indices / Vistas / Sinonimos

**Indice** (acelera lecturas; encarece escrituras)

```
CREATE INDEX EMP_APELLIDO_IX ON EMP (APELLIDO);
DROP INDEX EMP_APELLIDO_IX;
```

**Vista** (simplifica JOINS)

```
CREATE OR REPLACE VIEW V_EMP_DEPT AS
SELECT e.EMPNO, e.APELLIDO, d.DNOMBRE
FROM EMP e
```

```
JOIN DEPT d ON d.DEPTNO = e.DEPTNO;
```

### **Sinonimo** (alias a un objeto)

```
CREATE SYNONYM EMP_ALIAS FOR EMP;  
DROP SYNONYM EMP_ALIAS;
```

## **8) Errores tipicos (como pensar)**

- **FK**: insertas hijo con padre inexistente - crea padre primero o corrige clave.
- **NOT NULL**: faltan datos obligatorios.
- **PK/UNIQUE**: valor repetido - cambia dato o revisa clave.
- **CHECK**: regla de negocio no se cumple - revisa rango/IN.

## **9) Checklist para examen**

- Crear tabla con constraints nombradas.
- Distinguir PK vs UQ vs NN.
- Crear FK correcta y entender ON DELETE.
- ALTER (add/modify/drop/rename + add/drop constraint).
- Explicar DROP vs TRUNCATE.
- Crear INDEX y justificar.
- Crear VIEW.
- Crear SYNONYM (si procede).