



Bases de Datos

SQL Básico

Structured Query Language



SQL

- Definición (LDD).
- Consulta, actualización (LMD).
- Vistas.
- Índices (en SQL2 ya no hay).
- Inclusión en lenguajes (C, JAVA,...)



Nomenclatura

SOL	Mod. Relacional
Tabla	Relación
Fila	Tupla
Columna	Atributo



Crear tablas en SQL

```
CREATE TABLE DEPARTAMENTO  
( NOMBRED VARCHAR(15) NOT NULL,  
  NUMEROD   INT      NOT NULL,  
  NSS_JEFE   CHAR(9)  NOT NULL  
    DEFAULT "888665555",  
  FECHA_INIC_JEFE DATE,  
  PRIMARY KEY(NUMEROD),  
  UNIQUE(NOMBRED),  
  FOREIGN KEY(NSS_JEFE) REFERENCES  
  EMPLEADO(NSS));
```



Crear tablas en SQL

```
CREATE TABLE DEPARTAMENTO  
( NOMBRED VARCHAR(15) NOT NULL,  
  NUMEROD      INT      NOT NULL,  
  NSS_JEFE      CHAR(9)  NOT NULL  
    DEFAULT "888665555",  
  FECHA_INIC_JEFE DATE,  
  CONSTRAINT CLPDEPTO PRIMARY  
  KEY(NUMEROD),  
  CONSTRAINT CLSDE UNIQUE(NOMBRED),  
  CONSTRAINT CLEGTESDEP FOREIGN KEY  
  (NSS_JEFE) REFERENCES EMPLEADO(NSS)  
  ON DELETE SET DEFAULT ON UPDATE  
  CASCADE);
```

Tipos de datos

- Enteros:
 - **INTEGER / INT**
- Reales:
 - **FLOAT**
 - **REAL**
 - **DECIMAL(i,j) / DEC(i,j)**
 - **NATIONAL CHARACTER VARYING(n)**

Tipos de datos

- Longitud fija:
CHAR(n) CHARACTER(n)
- Longitud variable:
 - **CHARACTER VARYING(n)**
 - **VARCHAR(n)**

Tipos de datos

- Longitud fija:
 - **BIT(n)** Por defecto **n=1**.
- Longitud variable:
 - **BIT VARYING(n)**
- Ejemplos:
 - SiNo BIT
 - UnFlag BIT(100)

Tipos de datos

- **DATE**
 - Generalmente YYYY-MM-DD.
- **TIME / TIME(i)**
 - Normalmente HH:MM:SS.

Otros

CREATE DOMAIN TIPO_NSS AS CHAR(9);

- **DEFAULT**

- Si no se especifica es NULL.

- **NOT NULL** Restricción.

Restricciones de tabla

- En **CREATE TABLE**.
- Tras la descripción de atributos (columnas).

Claves

- **PRIMARY KEY:** Clave primaria.
- **UNIQUE:** Clave candidata.
- **FOREIGN KEY:** Clave extranjera.

Integridad referencial, clave externa

- Opciones:
 - **SET NULL**
 - **CASCADE**
 - **SET DEFAULT**
- Calificar con:
 - **ON DELETE**
 - **ON UPDATE**



Integridad referencial

```
CREATE TABLE EMPLEADO ...  
CONSTRAINT CLESUPEREMP FOREIGN  
KEY (NSS_SUPERV) REFERENCES  
EMPLEADO(NSS)  
ON DELETE SET NULL  
ON UPDATE CASCADE;
```

Si se modifica, propagar el nuevo valor a todos los empleados que lo tienen como jefe

Al borrar un empleado poner NULL en todos los empleados que lo tengan como jefe

Integridad referencial

- **CASCADE** es adecuada para :
 - Vínculo (TRABAJA_EN).
 - Atributos multivaluados (LOCALIZACIONES_DEPT).
 - T. de entidad débiles (DEPENDIENTE).

Integridad referencial

- El nombre de una restricción sirve para poderla sustituir o desechar.
- La asignación de nombre es opcional.



Borrar tablas ,DROP

DROP TABLE t1 CASCADE;
DROP TABLE t1 RESTRICT;

- **RESTRICT:** Borrar sólo si no existen referencias a la tabla:
 - En clave externa de otra relación.
 - En una vista.
- **CASCADE:** Borrar tabla y todas las restricciones y vistas donde haya referencias a ésta.



Modificar columnas, ALTER TABLE

- **Añadir**

```
ALTER TABLE EMPRESA.EMPLEADO ADD  
PUESTO VARCHAR(12);
```



Modificar columnas, ALTER TABLE

○ **Borrar**

**ALTER TABLE EMPRESA.EMPLEADO DROP
DIRECCIÓN CASCADE;**

**ALTER TABLE EMPRESA.EMPLEADO DROP
DIRECCIÓN RESTRICT;**



Modificar columnas, ALTER TABLE

- **Modificar definición**

```
ALTER TABLE EMPRESA.DEPARTAMENTO  
ALTER NSS_JEFE DROP DEFAULT;  
ALTER TABLE EMPRESA.DEPARTAMENTO  
ALTER NSS_JEFE SET DEFAULT  
"333445555";
```



Modificar restricciones, ALTER TABLE

- **Borrar**

```
ALTER TABLE EMPRESA.EMPLEADO DROP  
CONSTRAINT CLESUPEREMP;
```



Modificar restricciones, ALTER TABLE

○ Añadir

```
ALTER TABLE EMPRESA.EMPLEADO ADD  
CONSTRAINT CLESUPEREMP  
FOREIGN KEY (NSS_SUPERV)  
REFERENCES EMPLEADO(NSS)  
ON DELETE SET NULL  
ON UPDATE CASCADE;
```



Estructura básica


```
SELECT columnas  
FROM tablas  
[WHERE condición]
```



Ejemplo1

- Fecha de nacimiento y dirección de John Smith.

```
SELECT FECHA_NCTO, DIRECCIÓN  
FROM EMPLEADO  
WHERE NOMBRE='John'  
AND APELLIDO='Smith';
```

- **SELECT** FECHA_NCTO, DIRECCIÓN
FROM EMPLEADO
WHERE NOMBRE='John' **AND**
APELLIDO='Smith';

- $\pi_{\text{FECHA_NCTO}, \text{DIRECCIÓN}}$
 $(\sigma_{\text{NOMBRE}='John' \text{ AND } \text{APELLIDO}='Smith'}(\text{EMPLEADO}))$



Tablas reunidas (JOIN en FROM)

- Nombre y dirección de los empleados del departamento de Investigación.

C1: SELECT NOMBRE, APELLIDO, DIRECCIÓN
FROM EMPLEADO, DEPARTAMENTO
WHERE NOMBRED='Investigación' **AND**
ND=NÚMEROD;

C1A: SELECT NOMBRE, APELLIDO, DIRECCIÓN
FROM (EMPLEADO **INNER JOIN** DEPARTAMENTO
ON ND=NÚMEROD)
WHERE NOMBRED='Investigación';



Tablas reunidas (JOIN en FROM)

○ Tipos

- INNER JOIN (o JOIN)
- NATURAL JOIN
- OUTER JOIN
- LEFT [OUTER] JOIN
- RIGHT [OUTER] JOIN
- FULL [OUTER] JOIN

○ NO se pueden definir seudónimos de tablas reunidas:

FROM (EMPLEADO ~~INNER JOIN~~ DEPARTAMENTO **ON**
ND=NÚMEROD) ~~AS ED ...~~

JOIN en FROM

C1B: SELECT NOMBRE, APELLIDO, DIRECCIÓN
FROM (EMPLEADO **NATURAL JOIN** (DEPARTAMENTO **AS**
DEPTO(NOMBRED, ND, NSSG,FECHAIG)))
WHERE NOMBRED='Investigación';

Se están renombrando
atributos

- Renombra DEPARTAMENTO.NÚMEROD por ND.
- Condición de reunión implícita:
EMPLEADO.ND = DEPTO.ND



Anidados

```
C2A:SELECT NÚMEROP, NÚMD, APELLIDO, DIRECCIÓN  
FROM (PROYECTO INNER JOIN DEPARTAMENTO  
      ON NÚMD=NÚMEROD)  
      INNER JOIN EMPLEADO ON NSS_JEFE=NSS  
WHERE LOCALIZACIÓNP='Stafford';
```



Left outer join

```
C8B: SELECT  E.APELLIDO AS NOMBRE_EMPLEADO,  
              S.APELLIDO AS NOMBRE_SUPERVISOR  
FROM        (EMPLEADO E LEFT OUTER JOIN  
              EMPLEADO S ON E.NSS_SUPERV=SS.NSS);
```



Ejemplo 2

- Nombre y dirección de los empleados del departamento de Investigación.

```
SELECT NOMBRE, APELLIDO,  
DIRECCIÓN  
FROM EMPLEADO Inner join  
DEPARTAMENTO on NÚMEROD=ND  
WHERE NOMBRED='Investigación'
```

Ejemplo 2

- **SELECT** NOMBRE, APELLIDO, DIRECCIÓN
FROM EMPLEADO Inner join
DEPARTAMENTO on NÚMEROD=ND
WHERE NOMBRED='Investigación'
- Secuencia: $\sigma - | \times | - \pi$
 $DI \leftarrow \sigma_{\text{NOMBRED}='Investigación'}(\text{DEPARTAMENTO})$
 $EM_DI \leftarrow DI | \times |_{\text{NUMEROD}=\text{ND}} \text{EMPLEADO}$
 $\text{RESUL} \leftarrow \pi_{\text{NOMBREP}, \text{APELLIDO}, \text{DIRECCIÓN}}(EM_DI)$



Ejemplo 3

- N° de proyecto, n° de depto. que lo controla, apellido, dirección y fecha de nacimiento del jefe del depto. de todos los proyectos realizados en Stafford.
- **SELECT** NÚMEROP, NÚMD, APELLIDO,
DIRECCIÓN, FECHA_NCTO
FROM (PROYECTO inner join
DEPARTAMENTO on NÚMD=NÚMEROD)
inner join EMPLEADO on NSS_JEFE=NSS
WHERE LOCALIZACIÓNP='Stafford';

Ejemplo 3

- **SELECT** NÚMEROP, NÚMD, APELLIDO,
DIRECCIÓN, FECHA_NCTO
FROM (PROYECTO inner join DEPARTAMENTO
on NÚMD=NÚMEROD)
inner join EMPLEADO on NSS_JEFE=NSS
WHERE LOCALIZACIÓNP='Stafford';
- $PR_ST \leftarrow \sigma_{LOCALIZACIÓNP='Stafford'}(PROYECTO)$
 $DP_CN \leftarrow PR_ST \bowtie_{NÚMD=NÚMEROD} DEPARTAMENTO$
 $JEFE_DP_PRY \leftarrow DP_CN \bowtie_{NSS_JEFE=NSS} EMPLEADO$
 $RESULTADO \leftarrow \pi_{NÚMEROP, NÚMD, APELLIDO, DIRECCIÓN, FECHA_NCTO}$
 $(JEFE_DP_PRY)$

Calificar atrib y Seudónimos de tablas

- Nombre, apellido de cada empleado junto al nombre y apellido de su supervisor.

```
SELECT E.NOMBRE, E.APELLIDO,  
        S.NOMBRE, S.APELLIDO  
FROM    EMPLEADO E, EMPLEADO S  
WHERE E.NSS_SUPERV=S.NSS;
```

EMPLEADO E	NOMBRE	INIC	APELLIDO	<u>NSS</u>	...	NSS_SUPERV	ND
EMPLEADO S	NOMBRE	INIC	APELLIDO	<u>NSS</u>	...	NSS_SUPERV	ND



Calificar atrib y Seudónimos de tablas

```
SELECT E.NOMBRE, E.APELLIDO, E.DIR  
FROM EMPLEADO E, DEPARTAMENTO D  
WHERE D.NOMBRE = 'Investigación' AND  
D.ND=E.ND;
```



Omisión de WHERE

- Equivale a **WHERE TRUE**
- NSS de todos los empleados:

```
SELECT NSS  
FROM EMPLEADO;
```



Producto cartesiano

- Sin **WHERE** y más de una relación en **FROM**
- Combinaciones posibles de NSS de empleados con nombres de departamento:

```
SELECT  NSS, NOMBRED  
        FROM    EMPLEADO , DEPARTAMENTO;
```

Select *

- Para obtener todos los atributos

- **SELECT ***

FROM EMPLEADO

WHERE ND=5;

- **SELECT EMPLEADO.***

FROM EMPLEADO

WHERE ND=5;



ALL

**SELECT ALL SALARIO
FROM EMPLEADO;**

- SQL no elimina automáticamente las tuplas repetidas:
 - Usando *funciones agregadas* nos puede interesar no eliminarlos.



DISTINCT

Si sólo nos interesan los salarios
distintos

```
SELECT DISTINCT SALARIO  
FROM EMPLEADO;
```



DISTINCT

Pueden aparecer salarios repetidor
pero no dos filas con igual salario
y apellido:

```
SELECT DISTINCT SALARIO,  
APELLIDO FROM EMPLEADO;
```



Operación de conjuntos

- **UNION** (\cup), **INTERSECT** (\cap) y **EXCEPT** o **MINUS**(resta).
- Algunos SGBD sólo implementan la unión.
- Por defecto las tuplas repetidas se eliminan del resultado. Si las queremos mantener **ALL**
- Compatibilidad unión



Operación de unión de conjuntos

Números de proyecto donde participa Smith como trabajador o como jefe del dpto. controlador:

```
( SELECT NUMEROP
  FROM    ( PROYECTO inner join DEPARTAMENTO) on
NÚMD=NÚMEROD) inner join EMPLEADO on
NSS_JEFE=NSS
WHERE APELLIDO= 'Smith')
UNION
( SELECT NUMEROP
  FROM    TRABAJA_EN inner join EMPLEADO on
NSSE=NSS
WHERE  APELLIDO='Smith' );
```



Operación de unión de conjuntos

Números de proyecto donde participa Smith como trabajador y como jefe del dpto. controlador:

```
( SELECT NUMEROP
  FROM    ( PROYECTO inner join DEPARTAMENTO) on
NÚMD=NÚMEROD) inner join EMPLEADO on
NSS_JEFE=NSS
WHERE APELLIDO= 'Smith')
INTERSECT
( SELECT NUMEROP
  FROM    TRABAJA_EN inner join EMPLEADO on
NSSE=NSS
WHERE  APELLIDO='Smith' );
```



Operación de unión de conjuntos

Números de proyecto donde participa Smith como trabajador y no como jefe del dpto. controlador:

```
( SELECT NUMEROP
  FROM    ( PROYECTO inner join DEPARTAMENTO) on
NÚMD=NÚMEROD) inner join EMPLEADO on
NSS_JEFE=NSS
WHERE APELLIDO= 'Smith')
MINUS
( SELECT NUMEROP
  FROM    TRABAJA_EN inner join EMPLEADO on
NSSE=NSS
WHERE  APELLIDO='Smith' );
```



Comparar subcadenas, LIKE

- Empleados que viven en Houston, estado de Texas:

```
SELECT NOMBRE, APELLIDO  
FROM EMPLEADO  
WHERE DIRECCIÓN LIKE '%Houston,  
TX%';
```

- **_** sustituye a un carácter arbitrario.
- **%** a un n° indeterminado de caracteres.



Comparar subcadenas, LIKE

- Empleados que nacieron en la década de 1950:

```
SELECT NOMBRE, APELLIDO  
FROM EMPLEADO  
WHERE FECHA_NCTO LIKE '195_';
```




IN

- NSS de los empleados que trabajan en los proyectos 1, 2 ó 3.

```
SELECT DISTINCT NSSE  
FROM TRABAJA_EN  
WHERE NÚMP IN (1,2,3);
```



BETWEEN

- NSS de los empleados del departamento 5 cuyo salario está entre 30.000 y 40.000 \$.

SELECT *

FROM EMPLEADO

WHERE ND=5 AND

**(SALARIO BETWEEN 30000 AND
40000);**



Operaciones

- Aritméticas
 - '+', '-', '*', '/'
- Concatenación
 - '|'



Operaciones aritméticas

- Nombre y salario de los empleados que trabajan en 'ProductoX' tras aumentarles el sueldo un 10% :

```
SELECT NOMBRE, APELLIDO,  
1.1*SALARIO  
  
      FROM      (EMPLEADO inner join  
TRABAJA_EN on NSS=NSSE)inner  
join PROYECTO on NP=NÚMEROP  
  
      WHERE NOMBREP='ProductoX';
```

Ordenación de tuplas (ORDER BY)

Empleados y proyectos donde trabajan ordenados por depto y en cada depto alfabéticamente por apellido y nombre

```
SELECT NOMBRED, APELLIDO, NOMBRE,  
NOMBREP FROM ((DEPARTAMENTO inner  
join EMPLEADO on NÚMEROD=ND ) inner  
join TRABAJA_EN on NSS=NSSE )inner join  
pPROYECTO on NP=NÚMEROP  
ORDER BY NOMBRED, APELLIDO, NOMBREP;
```



Ordenación de tuplas (ORDER BY)

- **DESC ASC**
 - Por defecto, el orden es ascendente.
- **ORDER BY NOMBRED DESC,
APELLIDO ASC, NOMBRE ASC**



Seudónimos de atributos

```
SELECT E.APELLIDO AS
NOMBRE_EMPLEADO,
S.APELLIDO AS
NOMBRE_SUPERVISOR
FROM EMPLEADO AS E,
EMPLEADO AS S
WHERE E.NSS_SUPERV = S.NSS;
```

Seudónimos de atributos

FROM EMPLEADO E

SQL1

FROM EMPLEADO **AS** E

SQL2



Funciones agregadas

- **COUNT** (cuenta)
- **SUM** (suma)
- **MAX** (máximo)
- **MIN** (mínimo)
- **AVG** (media)

Funciones agregadas

- Suma de salarios de los empleados, junto a salarios máximo, mínimo y medio:

**C19: SELECT SUM(SALARIO), MAX(SALARIO),
MIN(SALARIO), AVG(SALARIO)
FROM EMPLEADO;**

EMPLEADO		
NOMBRE	SALARIO	ND
John	nulo	5
Franklin	15.000	5
Ramesh	10.000	5
Joyce	10.000	5
Alicia	10.000	4
Jennifer	20.000	4
Ahmad	20.000	4
Jaime	20.000	1

C19:	SUM(SA- LARIO)	MAX(SA- LARIO)	MIN(SA- LARIO)	AVG(SAL- ARIO)
	105.000	20.000	10.000	15.000

Funciones agregadas

- Suma de salarios de los empleados, junto a salarios máximo, mínimo y medio para empleados del dpto. 'Investigación':

```
SELECT SUM(SALARIO), MAX(SALARIO),  
MIN(SALARIO), AVG(SALARIO)  
FROM EMPLEADO INNER JOIN DEPARTAMENTO ON  
ND=NÚMEROD WHERE NOMBRED='Investigación';
```

EMPLEADO

NOMBRE	SALARIO	ND
John	nulo	5
Franklin	15.000	5
Ramesh	10.000	5
Joyce	10.000	5
Alicia	10.000	4
Jennifer	20.000	4
Ahmad	20.000	4
Jaime	20.000	1

DEPARTAMENTO

NOMBRED	NÚMEROD
Investigación	5
Administración	4
Dirección	1

SUM(SA-LARIO)	MAX(SA-LARIO)	MIN(SA-LARIO)	AVG(SAL-ARIO)
35.000	15.000	10.000	11.666

35.000 / 3



Funciones agregadas

- Cuántos empleados hay en la EMPRESA y cuántos en el departamento 'Investigación':

C21: SELECT COUNT(*)
FROM EMPLEADO;

* se refiere a tuplas

C22: SELECT COUNT(*)
FROM EMPLEADO INNER JOIN DEPARTAMENTO
ON ND=NÚMEROD
WHERE NOMBRED='Investigación';

Funciones agregadas

- Cuántos salarios diferentes hay:

**C23: SELECT COUNT(DISTINCT SALARIO)
FROM EMPLEADO;**

no cuenta
el valor nulo

EMPLEADO

NOMBRE	SALARIO	ND
John	nulo	5
Franklin	15.000	nulo
Ramesh	10.000	5
Joyce	10.000	5
Alicia	10.000	4
Jennifer	20.000	4
Ahmad	20.000	4
Jaime	20.000	1

C23:

COUNT(DISTINCT(SALARIO))
3

C23':

COUNT(SALARIO)
7

C23'':

COUNT(*)
8

Funciones agregadas

**C23': SELECT COUNT(SALARIO)
FROM EMPLEADO;**

Cuenta las filas con
salario no nulo

**C23'': SELECT COUNT(*)
FROM EMPLEADO;**

Cuenta todas las
filas de la tabla

EMPLEADO

NOMBRE	SALARIO	ND
John	nulo	5
Franklin	15.000	nulo
Ramesh	10.000	5
Joyce	10.000	5
Alicia	10.000	4
Jennifer	20.000	4
Ahmad	20.000	4
Jaime	20.000	1

C23:

COUNT(DISTINCT(SALARIO))
3

C23':

COUNT(SALARIO)
7

C23'':

COUNT(*)
8



Funciones agregadas


- ¿Qué ocurre cuando todos los salarios valen null?
- ¿Qué ocurre cuando la tabla está vacía?

Atributos de agrupación: GROUP BY

- Obtener el número de departamento junto a su número de empleados y salario medio:

**C24: SELECT ND, COUNT(*), AVG(SALARIO)
FROM EMPLEADO
GROUP BY ND;**

EMPLEADO			
NOMBRE		SALARIO	ND
John		30.000	5
Franklin		40.000	5
Ramesh		38.000	5
Joyce	...	25.000	5
Alicia		25.000	4
Jennifer		43.000	4
Ahmad		25.000	4
Jaime		55.000	1



ND	COUNT(*)	AVG(SALARIO)
5	4	33250
4	3	31000
1	1	55000

Fig. 8.4 (a)

Atributos de agrupación: GROUP BY

- Obtener el número de departamento junto a su número de empleados y salario medio:

```
C24: SELECT ND, COUNT(*), AVG(SALARIO)  
FROM EMPLEADO  
GROUP BY ND;
```

- Los valores nulos forman su propio grupo.

EMPLEADO					
NOMBRE		SALARIO		ND	
Jon		null		null	
Juan		10		null	
Rosa		10		5	
Ana	...	10	...	5	

ND	COUNT(*)	AVG(SALARIO)
null	2	10
5	2	10



GROUP BY, restricciones

- En **GROUP BY** está el/los atributo/s de agrupación.
- Atributos de **SELECT** todos en **GROUP BY**.
- Atrib. de **GROUP BY** no obligatorio en **SELECT**.
- Se suelen poner en **SELECT** algunos/todos los de **GROUP BY**.



GROUP BY

- Obtener por cada proyecto su número y nombre junto al número de empleados que trabajan en él.

```
SELECT NÚMEROP, NOMBREP, COUNT(*)  
      FROM PROYECTO INNER JOIN TRABAJA_EN  
            ON NÚMEROP=NÚMP  
      GROUP BY NÚMEROP, NOMBREP;
```

- ¿Se podría eliminar NOMBREP del GROUP BY?

GROUP BY

PROYECTO
INNER JOIN
TRABAJA_EN:

Agrupación por varios
atributos

NOMBREP	NÚMEROP
ProductoX	1
ProductoX	1
ProductoY	2
ProductoY	2
ProductoY	2
ProductoZ	3
ProductoZ	3
Automatización	10
Automatización	10
Automatización	10
Reorganización	20
Reorganización	20
Reorganización	20
Nuevas prestaciones	30
Nuevas prestaciones	30
Nuevas prestaciones	30

NÚMP	HORAS
1	32.5
1	20.0
2	7.5
2	20.0
2	10.0
3	40.0
3	10.0
10	10.0
10	10.0
10	35.0
20	10.0
20	15.0
20	nulo
30	5.0
30	20.0
30	30.0

NÚMEROP	NOMBREP	COUNT(*)
1	ProductoX	2
2	ProductoY	3
3	ProductoZ	2
10	Automatización	3
20	Reorganización	3



CASE

```
SELECT NOMBRE,  
  CASE  
    WHEN ESTADOCIVIL='S' THEN 'SOLTERO/A'  
    WHEN ESTADOCIVIL='C' THEN 'CASADA/O'  
    WHEN ESTADOCIVIL='D' THEN 'DIVORCIADO/A'  
    ELSE 'VIUDA/O'  
  END, EDAD, FECHA_NACIMIENTO  
FROM PERSONAS;
```

- En los WHEN cualquier condición (AND, OR, ...)
- Ahorro espacio almacenamiento: S/C/D/V frente a Soltero/Casado ...



CASE

```
SELECT NOMBRE,  
        CASE ESTADO_CIVIL  
          WHEN 'S' THEN 'SOLTERO/A'  
          WHEN 'C' THEN 'CASADA/O'  
          WHEN 'D' THEN 'DIVORCIADO/A'  
          ELSE 'VIUDA/O'  
        END, EDAD, FECHA_NACIMIENTO  
FROM PERSONAS;
```

- En los WHEN un valor posible del atributo.



CASE

```
UPDATE EMPLEADO  
SET SUELDO = CASE DEPTO  
    WHEN 'VIDEO' THEN SUELDO=SUELDO*1.1  
    WHEN 'MÚSICA' THEN SUELDO=SUELDO*1.2  
    ELSE 0  
    END;
```

INSERT

A1: INSERT INTO EMPLEADO

VALUES ('Ricardo', 'C', 'Martínez',
'653298653', '1952-12-30', 'Olmo 98,
Cedros, MX', 'H', 37000, '987654321', 4)

- Mismo orden en el que se especificaron los atributos.



INSERT

A1A: INSERT INTO

EMPLEADO(NOMBRE, APELLIDO,
ND,NSS)

VALUES ('Ricardo', 'Martínez', 4,
'653298653')

- Así los atributos con valor NULL o DEFAULT se pueden omitir.
- Mismo orden en el que se especifican los atributos en **INSERT**.



INSERT

```
A2:INSERT INTO EMPLEADO  
(NOMBRE,APELLIDO,NSS,ND)  
VALUES ('Roberto, 'Huerta',  
'980760540', 2)
```



INSERT

A2A: INSERT INTO EMPLEADO
(NOMBRE, APELLIDO, ND)

VALUES ('Roberto, 'Huerta', 2)

- Rechazada por no proporcionar valor para NSS (clave primaria: NOT NULL).



INSERT (2)

A3A: CREATE TABLE INFO_DEPTOS (
NOMBRE_DEPTO VARCHAR(15),
NÚM_DE_EMPS INTEGER,
SAL_TOTAL INTEGER);

INSERT (2)

```
INSERT INTO INFO_DEPTOS  
(NOMBRE_DEPTO,  
  NÚM_DE_EMPS, SAL_TOTAL)  
SELECT NOMBRED, COUNT (*),  
        SUM(SALARIO)  
FROM    (DEPARTAMENTO JOIN  
EMPLEADO ON NÚMEROD=ND)  
GROUP BY NOMBRED;
```



INSERT (2)

- Inserta varias tuplas
 - el resultado de la consulta
- Utilidad: Tabla temporal donde realizar consultas.
 - Sus datos pueden *perder actualidad*.
 - Alternativa sin este problema: Vista.



DELETE

DELETE FROM EMPLEADO

WHERE APELLIDO='Brown';

- Una sola tabla.
- WHERE: selección de tuplas a eliminar.
- El borrado se puede propagar
 - RI referencial



DELETE

```
DELETE FROM EMPLEADO  
WHERE NSS='123456789';
```




DELETE

```
DELETE FROM EMPLEADO  
WHERE ND IN  
      (SELECT NÚMEROD  
        FROM DEPARTAMENTO  
        WHERE  
        NOMBRED='Investigación');
```



DELETE

DELETE FROM EMPLEADO;

- Sin WHERE se borran todas las tuplas.
- DROP eliminaría la definición de la tabla.



UPDATE

```
UPDATE PROYECTO  
    SET LOCALIZACIÓNP='Bellaire',  
NUMD=5    WHERE NÚMEROP=10;
```



UPDATE

```
UPDATE EMPLEADO  
SET SALARIO=SALARIO*1.1  
WHERE ND IN  
(SELECT NÚMEROD  
FROM DEPARTAMENTO  
WHERE NOMBRED= 'Investigación');
```