



Repaso SQL



El lenguaje SQL.

Lenguaje SQL:

- ✓ **manipulación de datos (consulta y actualización):**
 - **SELECT** (consulta)
 - **INSERT** (inserción de tuplas)
 - **DELETE** (borrado de tuplas)
 - **UPDATE** (modificación de tuplas)

- ✓ **definición de datos (definición del esquema)**

El lenguaje SQL.

SELECT [ALL | DISTINCT] $A_{1i}, A_{2j}, \dots A_{nk}$ | *
FROM $R_1, R_2, \dots R_n$
[WHERE *condición*]
[GROUP BY $B_1, B_2, \dots B_m$]
[HAVING *condición*]

El lenguaje SQL.

SELECT [ALL | DISTINCT] $A_{1i}, A_{2j}, \dots, A_{nk}$ | *
FROM R_1, R_2, \dots, R_n
[WHERE *condición*]

condición que
cumplen los datos que
se desea consultar

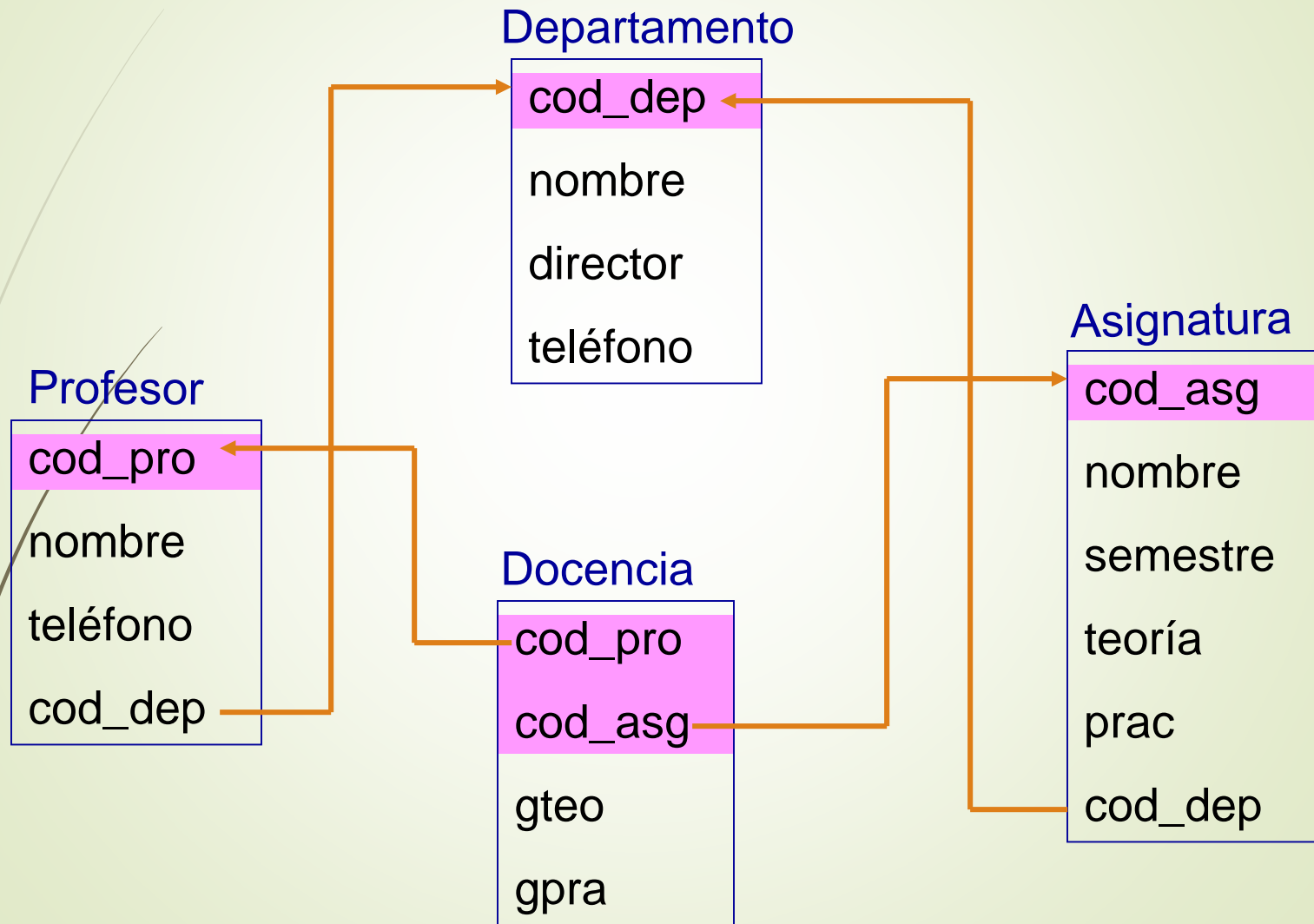
datos que se desea
consultar

relaciones que
intervienen en la
consulta

El lenguaje SQL.

atributos identificadores

atributos de referencia



El lenguaje SQL.

Esquema relacional

Departamento (cod_dep: tira(5), nombre: tira(40), director tira(30),
teléfono : entero)

Asignatura (cod_asg: tira(3), nombre: tira(40), semestre: tira(2),
teoría: real, prác: real, cod_dep: tira(5))

Profesor (cod_pro : tira(3), nombre : tira(40), teléfono: entero,
cod_dep: tira(5))

Docencia (cod_asg: tira(3), cod_pro: tira(3), gteo: entero, gpra: entero)

El lenguaje SQL.

BD relacional

Departamento

cod_dep	nombre	director	teléfono
DSIC	Sistemas Informáticos y Computación	V. Botti	3500
DISCA	Ingeniería de Sistemas, Computadores y Automática	A. Crespo	5700
MAT	Matemática Aplicada	P. Pérez	6600
FIS	Física Aplicada	J. Linares	5200
IDM	Idiomas	B. Montero	5300
EIO	Estadística e Investigación Operativa	L. Barceló	4900
OEM	Org. de Empresas, Economía Financ. y Contabilidad	M. Pérez	6800

Asignatura

cod_asg	nombre	semestre	teoría	prac	cod_dep
BDA	Bases de Datos	2B	3	3	DSIC
AD1	Algoritmos y Estructuras de Datos 1	1A	4	2	DSIC
FCO	Fundamentos de computadores	1A	4,5	4,5	DISCA
MAD	Matemática Discreta	1A	3	3	MAT
INT	Inglés Técnico	1B	3	3	IDM
FFI	Fundamentos Físicos de la Informática	1A	3	3	FIS
EC2	Estructuras de Computadores 2	2A	3	3	DISCA

Docencia

cod_asg	cod_pro	gteo	gpra
BDA	JCC	2	4
MAD	RFC	1	2
FCO	DGT	2	2
AD1	MAF	1	1
INT	CPG	1	0
EC2	JBD	2	0
BDA	MCG	1	3
AD1	JCC	1	1
FCO	JBD	2	2
AD1	MCG	1	1

Profesor

cod_pro	nombre	teléfono	cod_dep
JCC	Juan C. Casamayor Ródenas	7796	DSIC
RFC	Robert Fuster i Capilla	6789	MAT
JBD	José V. Benlloch Dualde	5760	DISCA
MAF	María Alpuente Frasnado	3560	DSIC
CPG	Cristina Pérez Guillot	7439	IDM
JTM	José M. Torralba Martínez	4590	OEM
IGP	Ignacio Gil Pechuán	3423	OEM
DGT	Daniel Gil Tomás	5679	DISCA
MCG	Matilde Celma Giménez	7756	DSIC

El lenguaje SQL.

Consultas sobre una relación.

```
SELECT [ALL | DISTINCT]  $A_1, A_2, \dots A_k$  | *  
FROM R  
[WHERE condición]
```

de las tuplas de R para las cuales la *condición* se evalúa a **cierto**, obtener el valor de los atributos $A_1, A_2, \dots A_k$



R DONDE *condición* [$A_1, A_2, \dots A_k$]

← Álgebra
Relacional

Cálculo
Relacional
de Tuplas →

$RX.A_1, RX.A_2, \dots RX.A_k$ | $R(RX) \wedge$ *condición*

El lenguaje SQL.

Obtener el código, el nombre y el departamento de adscripción de todas las asignaturas.

```
SELECT código, nombre, cod_dep  
FROM Asignatura
```

Obtener el código y el nombre de todas las asignaturas del DSIC.

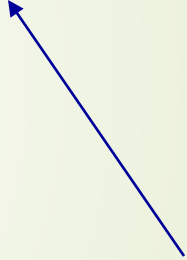
```
SELECT código, nombre  
FROM Asignatura  
WHERE cod_dep="DSIC"
```

El lenguaje SQL.

Obtener el código, el nombre y el total de créditos de todas las asignaturas del DSIC.

```
SELECT código, nombre, teoría + prac  
FROM Asignatura  
WHERE cod_dep='DSIC'
```

operadores aritméticos



El lenguaje SQL.

Obtener el código de los departamentos que tienen adscritas asignaturas.

```
SELECT cod_dep  
FROM Asignatura
```



DSIC
DSIC
DISCA
MAT
IDM
FIS
DISCA

```
SELECT DISTINCT cod_dep  
FROM Asignatura
```



DSIC
DISCA
MAT
IDM
FIS

se eliminan las filas duplicadas en la relación resultante de la consulta

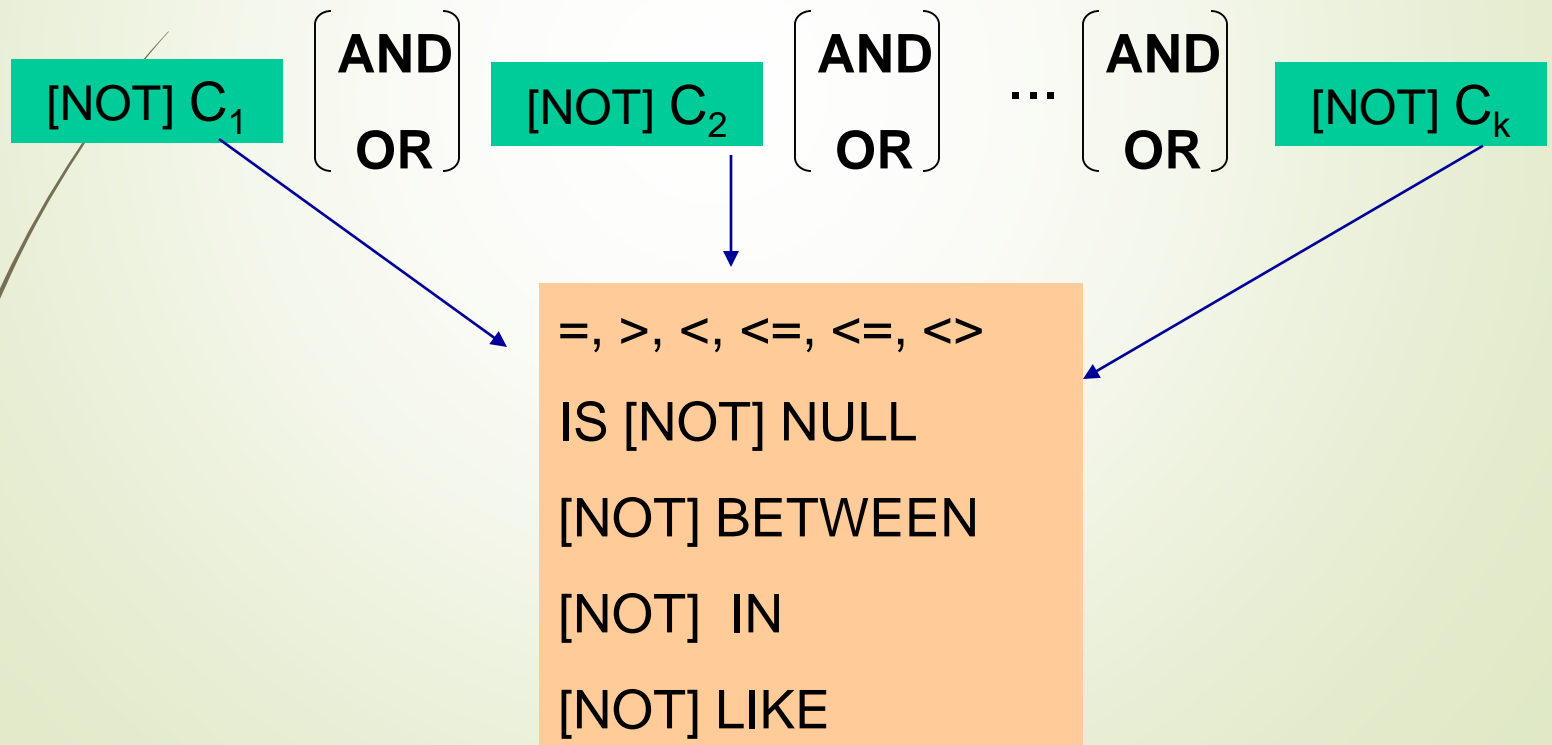
El lenguaje SQL.

Obtener todos los datos de todos los profesores del DISCA.

```
SELECT *  
FROM Profesor  
WHERE cod_dep='DISCA'
```

El lenguaje SQL.

SELECT [ALL | DISTINCT] A_1, A_2, \dots, A_k | *
FROM R
[WHERE *condición*]



El lenguaje SQL.

Obtener el código y el nombre de las asignaturas del DISCA de primer curso.

```
SELECT código, nombre  
FROM Asignatura  
WHERE cod_dep='DISCA'  
      AND  
      (semestre='1A' OR semestre ='1B')
```

```
SELECT código, nombre  
FROM Asignatura  
WHERE cod_dep='DISCA'  
      AND  
      semestre IN ('1A', '1B')
```

El lenguaje SQL.

Obtener el código de los profesores que tienen a su cargo entre 3 y 5 grupos de clase en una misma asignatura, indicando cual es ésta.

```
SELECT cod_pro, cod_asg  
FROM Docencia  
WHERE (gteo+gpri) BETWEEN 3 AND 5
```



```
SELECT cod_pro, cod_asg  
FROM Docencia  
WHERE (gteo+gpri) >=3 AND (gteo+gpri) <= 5
```

El lenguaje SQL.

Obtener el nombre de las asignaturas que tratan de Programación.

```
SELECT nombre  
FROM Asignatura  
WHERE nombre LIKE '%Programación%'
```

El predicado LIKE sirve para comparar una tira de caracteres con un patrón construido con ayuda de los comodines - y %

- : un carácter

% : una tira de 0 o mas caracteres

El lenguaje SQL.

Obtener el nombre de los departamentos que no tienen director.

```
SELECT nombre  
FROM Departamento  
WHERE director IS NULL
```

El predicado IS NULL sirve para comprobar si un atributo tiene valor NULO.

```
SELECT nombre  
FROM Departamento  
WHERE director = NULL
```

NULL no es una constante

El lenguaje SQL.

Condiciones

LÓGICA TRIVALUADA

Predicados: $<$, $>$, $=$, \geq , \leq , \neq , IN, BETWEEN, LIKE

Evaluación de condiciones:

Si en un argumento de un predicado un atributo tiene **VALOR NULO** el predicado se evalúa a **INDEFINIDO**, en caso contrario se evalúa de acuerdo a la semántica del predicado.

Predicado IS NULL:

IS NULL (A) se evalúa a **CIERTO** para una tupla si el atributo A tiene **VALOR NULO** en la tupla, en caso contrario se evalúa a **FALSO**.

El lenguaje SQL.

Consulta de valores de grupo (valores agregados):

SELECT [**ALL** | **DISTINCT**] *funciones agregadas*
FROM R
[WHERE *condición*]

valores globales para el conjunto de tuplas que cumplen la condición.

AVG
MAX
MIN
SUM
COUNT

COUNT (*)

(**[ALL** | **DISTINCT**] *expresión_escalar*)

El lenguaje SQL.

Consulta de valores de grupo (valores agregados):

- Las funciones agregadas no se pueden anidar.
- Para las funciones **sum** y **avg** los argumentos deben ser numéricos.
- **distinct** indica que los valores redundantes sean eliminados antes de que se realice el cálculo correspondiente.
- La función especial **count(*)**, en la que no está permitido incluir **distinct** ni **all**, da como resultado el cardinal del conjunto de filas de la selección.
- Los cálculos se realizan después de la selección y aplicar las condiciones.
- Los valores nulos son eliminados antes de realizar los cálculos (incl. **count**).
- Si el número de filas de la selección es 0, la función **count** devuelve el valor 0 y las otras funciones el valor nulo.

El lenguaje SQL.

Obtener el número total de créditos de las asignaturas de primer curso.

```
SELECT SUM (teoría) + SUM (prac)  
FROM Asignatura  
WHERE (semestre='1A' OR semestre ='1B')
```

```
SELECT SUM (teoría + prac)  
FROM Asignatura  
WHERE (semestre='1A' OR semestre ='1B')
```

El lenguaje SQL.

cod_asg	nombre	semestre	teoría	prac	cod_dep
BDA	Bases de Datos	2B	3	3	DSIC
AD1	Algoritmos y Estructuras de Datos 1	1A	4	2	DSIC
FCO	Fundamentos de computadores	1A	4,5	4,5	DISCA
MAD	Matemática Discreta	1A	3	3	MAT
INT	Inglés Técnico	1B	3	3	IDM
FFI	Fundamentos Físicos de la Informática	1A	3	3	FIS
EC2	Estructuras de Computadores 2	2A	3	3	DISCA

SELECT SUM (teoría) + SUM (prac)
FROM Asignatura
WHERE (semestre="1A" OR semestre ="1B")

El lenguaje SQL.

Obtener el número de profesores del DSIC.

```
SELECT COUNT (*)  
FROM Profesor  
WHERE cod_dep="DSIC"
```

Obtener el número máximo y mínimo de créditos de las asignaturas.

```
SELECT MAX (teoría + prac), MIN (teoría + prac)  
FROM Asignatura
```

El lenguaje SQL.

Obtener el código y el nombre de profesores del DSIC, así como el total de profesores que tiene adscritos.

```
SELECT cod_pro, nombre, COUNT(*)  
FROM Profesor  
WHERE cod_dep='DSIC'
```

cod_pro	nombre	teléfono	cod_dep
JCC	Juan C. Casamayor Ródenas	7796	DSIC
RFC	Robert Fuster i Capilla	6789	MAT
JBD	José V. Benlloch Dualde	5760	DISCA
MAF	María Alpuente Frasnado	3560	DSIC
CPG	Cristina Pérez Guillot	7439	IDM
JTM	José M. Torralba Martínez	4590	OEM
IGP	Ignacio Gil Pechuán	3423	OEM
DGT	Daniel Gil Tomás	5679	DISCA
MCG	Matilde Celma Giménez	7756	DSIC



cod_pro	nombre
JCC	Juan C. Casamayor Ródenas
MAF	María Alpuente Frasnado
MCG	Matilde Celma Giménez

¡No se pueden consultar simultáneamente datos de las tuplas individuales y datos globales del grupo de tuplas.!

En consultas no agrupadas, la selección sólo podrá incluir referencias a funciones agregadas o literales ya que las funciones van a devolver un único valor.

El lenguaje SQL.

Consultas sobre una relación.

SELECT [ALL | DISTINCT] A_1, A_2, \dots, A_k | *
FROM R
[WHERE *condición*]

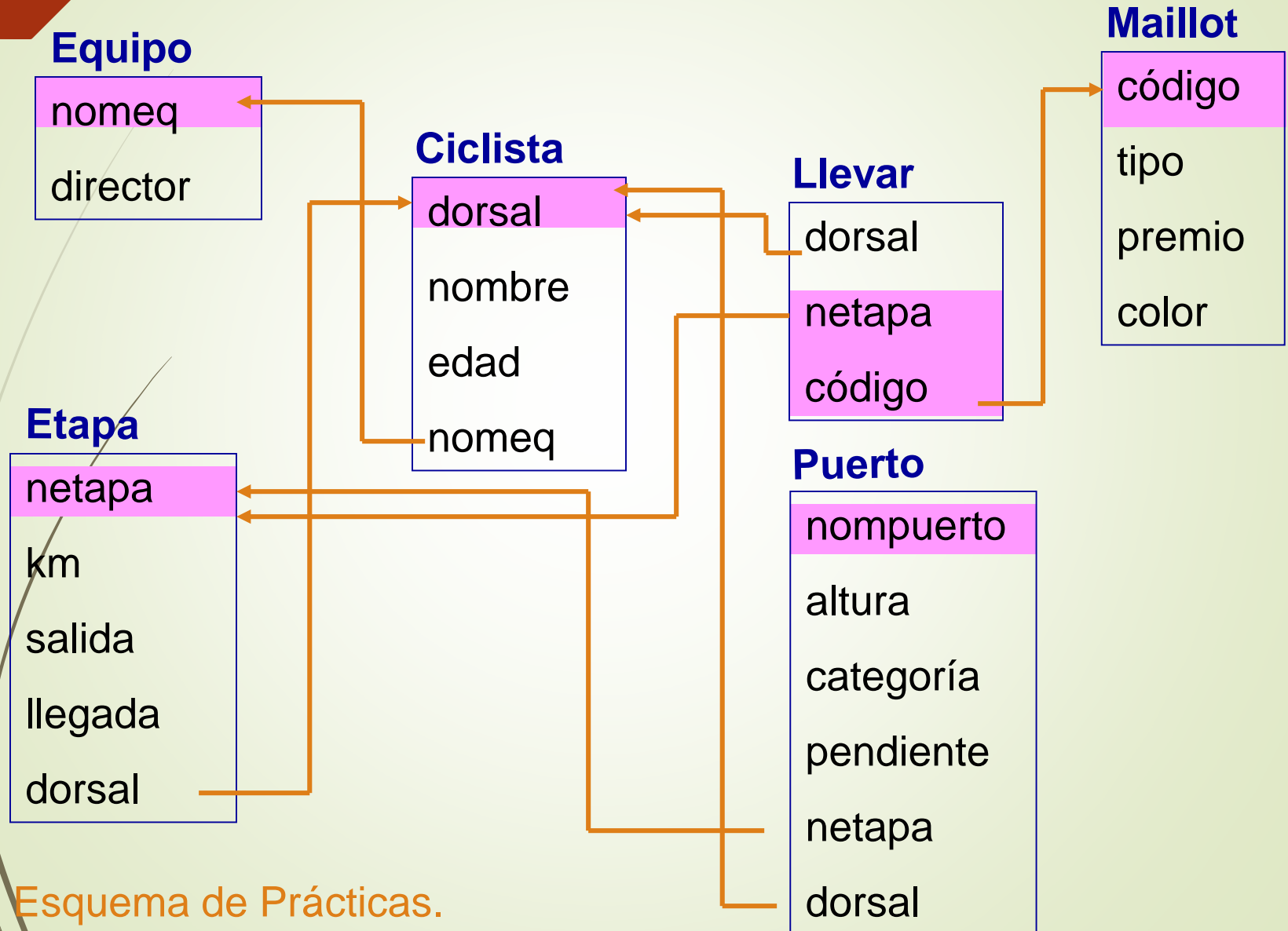
expresiones lógicas

- ✓ atributos
- ✓ expresiones
- ✓ funciones agregadas



Esquema de CICLISMO

Esquema CICLISMO



Esquema CICLISMO (Oracle)

```
CREATE TABLE equipo (  
    nomeq VARCHAR2(25) CONSTRAINT PK_equi PRIMARY KEY,  
    descripción VARCHAR2(100));
```

```
CREATE TABLE ciclista (  
    dorsal NUMBER(3) CONSTRAINT PK_cicli PRIMARY KEY,  
    nombre VARCHAR2(30) NOT NULL,  
    edad NUMBER(2) ,  
    nomeq VARCHAR(25) NOT NULL  
    CONSTRAINT FK_cicli_equi REFERENCES equipo (nomeq));
```



Esquema CICLISMO (Oracle)

```
CREATE TABLE etapa (  
  netapa NUMBER(2) CONSTRAINT PK_eta PRIMARY KEY,  
  km NUMBER(3) ,  
  salida VARCHAR(35),  
  llegada VARCHAR(35),  
  dorsal NUMBER(3) CONSTRAINT FK_etapa_cicli REFERENCES ciclista (dorsal));
```

```
CREATE TABLE puerto (  
  nompuerto VARCHAR2(35) CONSTRAINT PK_puerto PRIMARY KEY,  
  altura NUMBER(4),  
  categoria CHAR(1),  
  pendiente NUMBER(3,2),  
  netapa NUMBER(2) NOT NULL CONSTRAINT FK_puerto_eta REFERENCES  
    etapa(netapa),  
  dorsal NUMBER(3) CONSTRAINT FK_puerto_cicli REFERENCES ciclista (dorsal));
```

Esquema CICLISMO (Oracle)

```
CREATE TABLE maillot (  
  codigo CHAR(3) CONSTRAINT PK_mai PRIMARY KEY,  
  tipo VARCHAR2(30),  
  color VARCHAR2(20),  
  premio NUMBER(7) );
```

```
CREATE TABLE llevar (  
  dorsal NUMBER(3) NOT NULL CONSTRAINT  
    FK_llevar_cicli REFERENCES ciclista (dorsal),  
  netapa NUMBER(2) CONSTRAINT FK_llevar_etapa  
    REFERENCES etapa (netapa),  
  codigo CHAR(3) CONSTRAINT FK_llevar_mai  
    REFERENCES maillot (codigo),  
  CONSTRAINT PK_lle PRIMARY KEY (netapa, codigo));
```

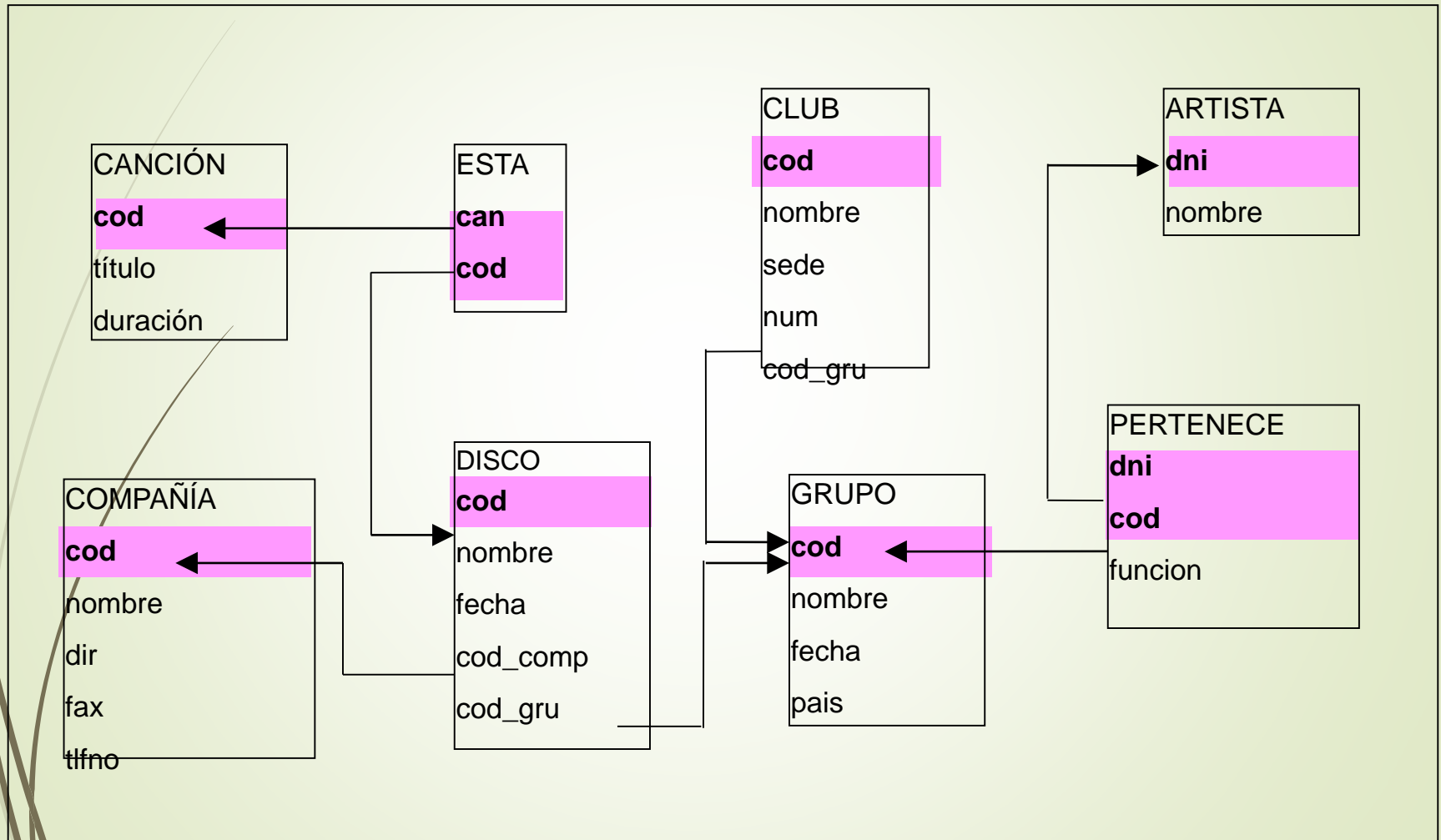


Esquema de MÚSICA

Esquema MUSICA

atributos identificadores

atributos de referencia





Esquema MUSICA (ORACLE)

```
CREATE TABLE artista (  
    dni VARCHAR2(10) CONSTRAINT PK_arti  
        PRIMARY KEY,  
    nombre VARCHAR2(30) NOT NULL);
```

```
CREATE TABLE grupo (  
    cod CHAR(3) CONSTRAINT PK_gru PRIMARY  
        KEY,  
    nombre VARCHAR2(30) NOT NULL,  
    fecha DATE,  
    pais VARCHAR(10) );
```

Esquema MUSICA (ORACLE)

```
CREATE TABLE club (  
  cod CHAR(3) CONSTRAINT PK_club PRIMARY KEY,  
  nombre VARCHAR2(30) NOT NULL,  
  sede VARCHAR2(30),  
  num NUMBER(6),  
  cod_gru CHAR(3) NOT NULL CONSTRAINT FK_club_grupo  
    REFERENCES grupo(cod));
```

```
CREATE TABLE companyia (  
  cod CHAR(3) CONSTRAINT PK_compa PRIMARY KEY,  
  nombre VARCHAR2(30) NOT NULL,  
  dir VARCHAR2(30),  
  fax VARCHAR2(15),  
  tfno VARCHAR2(15) );
```

Esquema MUSICA (ORACLE)

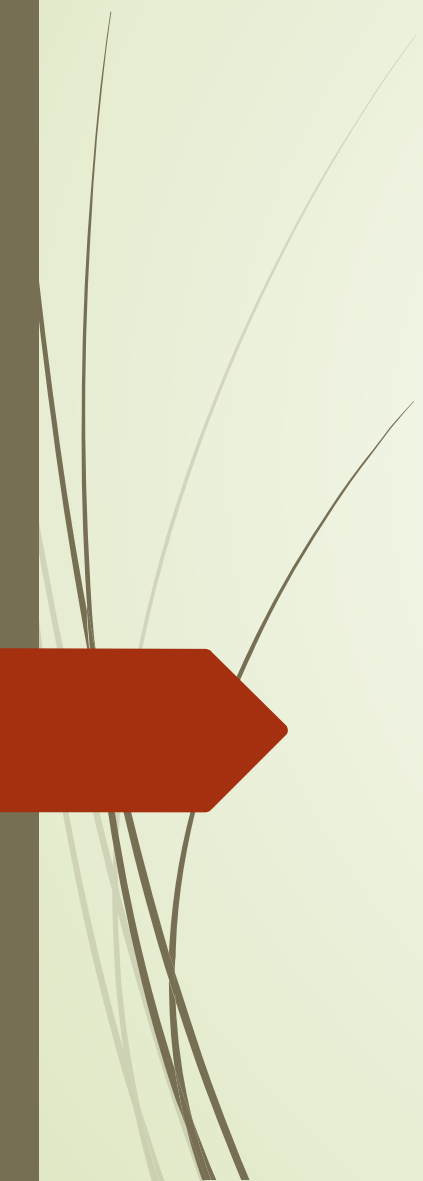
```
CREATE TABLE cancion (  
  cod NUMBER(3) CONSTRAINT PK_can PRIMARY  
    KEY,  
  titulo VARCHAR2(30) NOT NULL,  
  duracion NUMBER(2) );
```

```
CREATE TABLE disco (  
  cod CHAR(3) CONSTRAINT PK_dis PRIMARY KEY,  
  nombre VARCHAR2(30),  
  fecha DATE ,  
  cod_comp CHAR(3) NOT NULL CONSTRAINT  
    FK_disco_comp REFERENCES companyia (cod),  
  cod_gru CHAR(3) NOT NULL CONSTRAINT  
    FK_disco_grupo REFERENCES grupo(cod));
```

Esquema MUSICA (ORACLE)

```
CREATE TABLE esta (  
    can NUMBER(3) CONSTRAINT FK_esta_can  
        REFERENCES cancion (cod),  
    cod CHAR(3) CONSTRAINT FK_esta_disco  
        REFERENCES disco (cod),  
    CONSTRAINT PK_esta PRIMARY KEY (can, cod));
```

```
CREATE TABLE pertenece (  
    dni VARCHAR2(10) CONSTRAINT FK_perte_arti  
        REFERENCES artista (dni),  
    cod CHAR(3) CONSTRAINT FK_perte_grupo  
        REFERENCES grupo (cod),  
    funcion VARCHAR2(15),  
    CONSTRAINT PK_perte PRIMARY KEY (dni, cod));
```

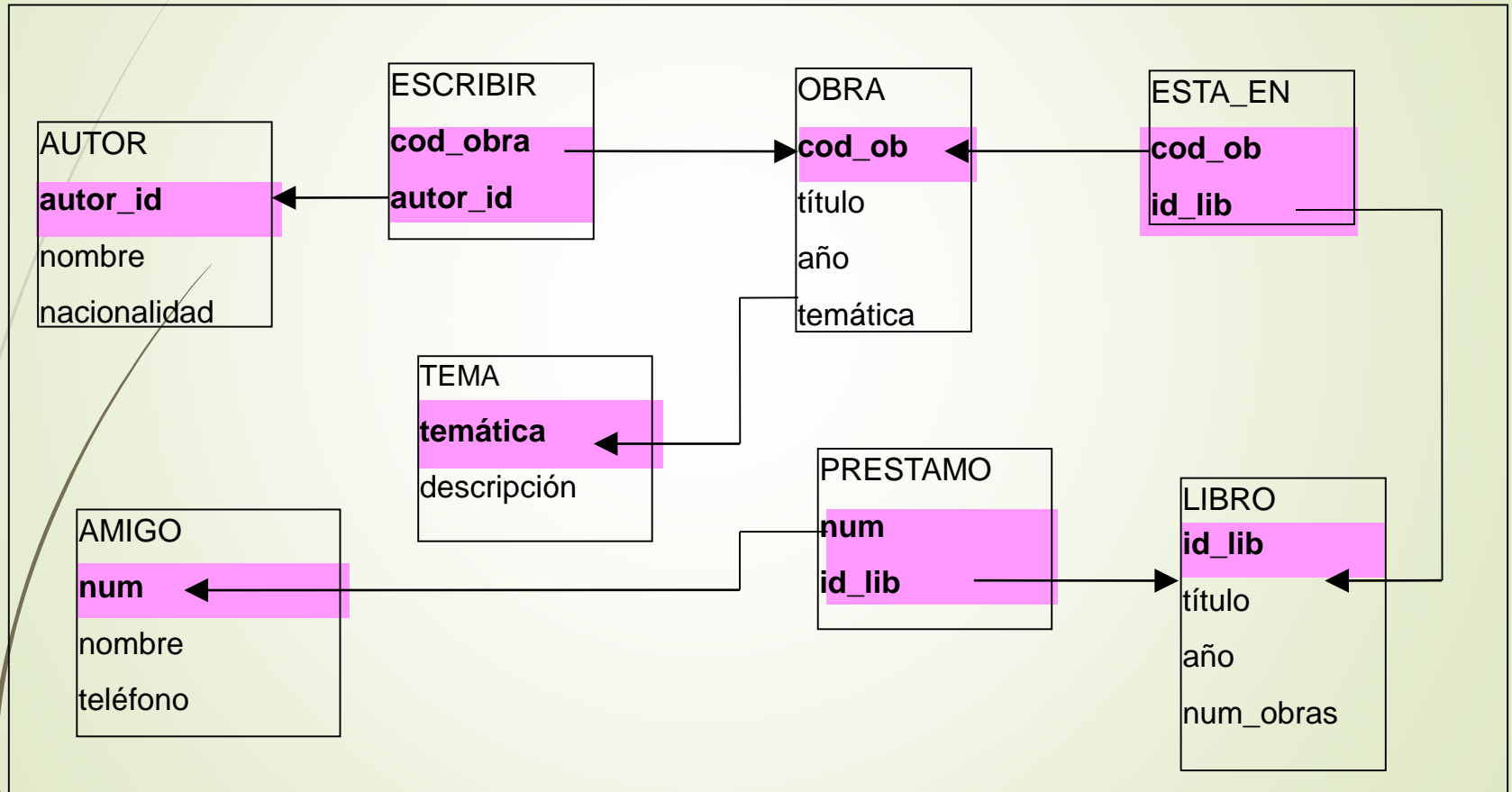


Esquema de BIBLIOTECA

Esquema BIBLIOTECA

atributos identificadores

atributos de referencia





Esquema BIBLIOTECA (ORACLE)

```
CREATE TABLE autor (  
    autor_id CHAR(4),  
    nombre VARCHAR2(35) NOT NULL,  
    nacionalidad VARCHAR2(20),  
    CONSTRAINT cp_autor PRIMARY KEY (autor_id)) ;
```

```
CREATE TABLE libro (  
    id_lib VARCHAR2(10),  
    titulo VARCHAR2(80),  
    año NUMBER(5,0),  
    num_obras NUMBER(5,0),  
    CONSTRAINT cp_lib PRIMARY KEY (id_lib)) ;
```




Esquema BIBLIOTECA (ORACLE)

```
CREATE TABLE tema (  
    tematica VARCHAR2(20),  
    descripcion VARCHAR2(50),  
    CONSTRAINT cp_tema PRIMARY KEY (tematica)) ;
```

```
CREATE TABLE obra (  
    cod_ob NUMBER(10,0),  
    titulo VARCHAR2(80) NOT NULL,  
    tematica VARCHAR2(20),  
    CONSTRAINT cp_obra PRIMARY KEY (cod_ob),  
    CONSTRAINT ca_obra_tema FOREIGN KEY  
        (tematica) REFERENCES tema(tematica)) ;
```


Esquema BIBLIOTECA (ORACLE)

```
CREATE TABLE amigo (  
    num NUMBER(5),  
    nombre VARCHAR2(60) NOT NULL,  
    telefono VARCHAR2(10),  
    CONSTRAINT cp_amigo PRIMARY KEY (num)) ;
```

```
CREATE TABLE prestamo (  
    num NUMBER(5),  
    id_lib VARCHAR2(10),  
    CONSTRAINT cp_pres PRIMARY KEY (num, id_lib),  
    CONSTRAINT ca_pres_obra FOREIGN KEY (num)  
        REFERENCES amigo(num),  
    CONSTRAINT ca_pres_libro FOREIGN KEY (id_lib)  
        REFERENCES libro(id_lib)) ;
```

Esquema BIBLIOTECA (ORACLE)

```
CREATE TABLE esta_en (  
  cod_ob NUMBER(10,0),  
  id_lib VARCHAR2(10),  
  CONSTRAINT cp_esta_en PRIMARY KEY (cod_ob, id_lib),  
  CONSTRAINT ca_estaen_obra FOREIGN KEY (COD_OB)  
    REFERENCES obra(cod_ob),  
  CONSTRAINT ca_estaen_libro FOREIGN KEY (ID_LIB) REFERENCES  
    libro(id_lib)) ;
```

```
CREATE TABLE escribir (  
  autor_id CHAR(4),  
  cod_ob NUMBER(10,0),  
  CONSTRAINT cp_escribir PRIMARY KEY (autor_id, cod_ob),  
  CONSTRAINT ca_esc_obra FOREIGN KEY (cod_ob) REFERENCES  
    obra(cod_ob),  
  CONSTRAINT ca_esc_autor FOREIGN KEY (autor_id) REFERENCES  
    autor(autor_id)) ;
```



Consultas sobre varias relaciones

El lenguaje SQL.

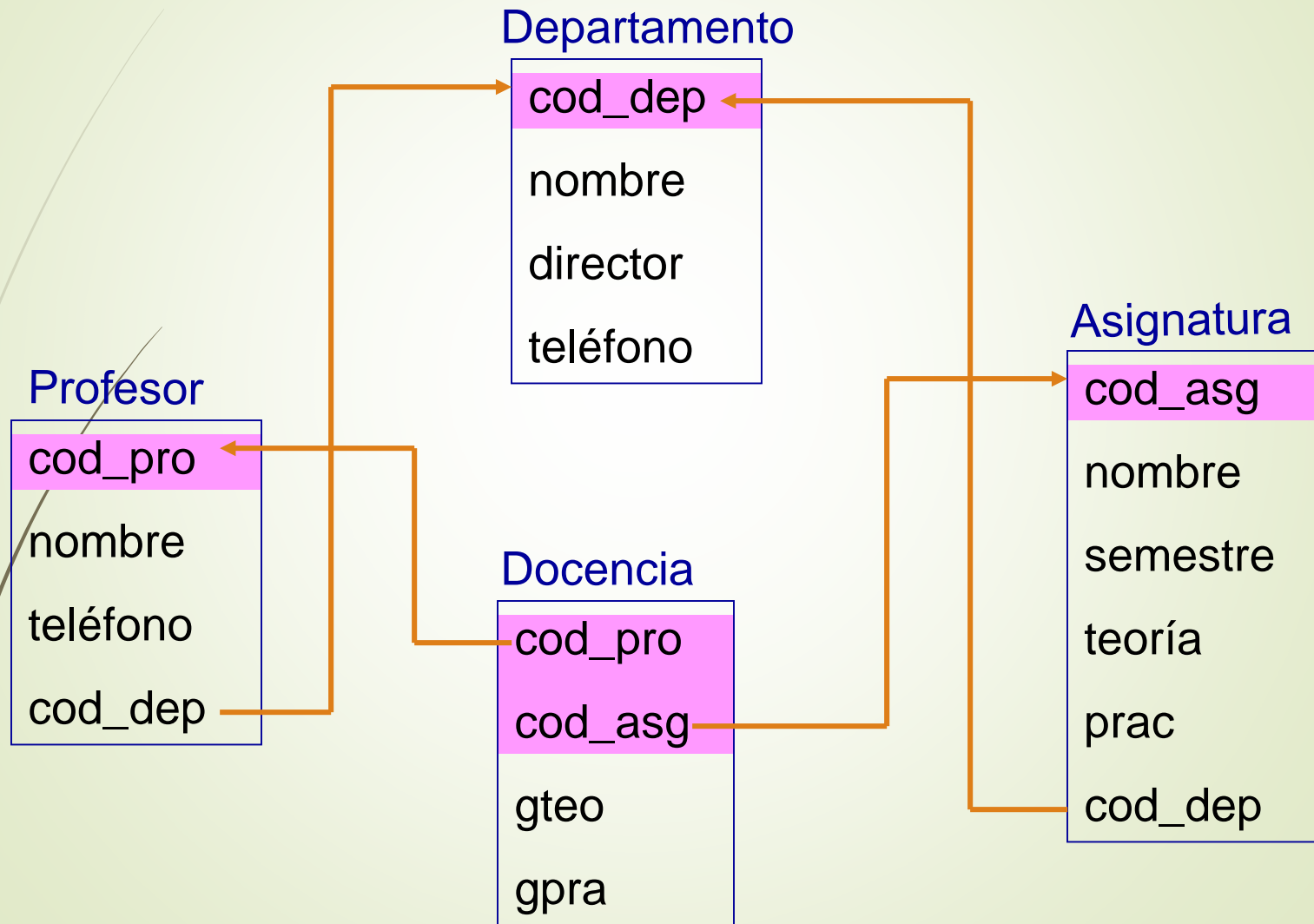
Consultas sobre varias relaciones.

```
SELECT [ALL | DISTINCT]  $A_{1i}, A_{2j}, \dots A_{kl}$  *  
FROM  $R_1, R_2, \dots R_n$   
[WHERE condición]  
[GROUP BY  $B_1, B_2, \dots B_m$ ]  
[HAVING condición]
```

El lenguaje SQL.

atributos identificadores

atributos de referencia



El lenguaje SQL.

Esquema relacional

Departamento (cod_dep: tira(5), nombre: tira(40), director tira(30),
teléfono : entero)

Asignatura (cod_asg: tira(3), nombre: tira(40), semestre: tira(2),
teoría: real, prác: real, cod_dep: tira(5))

Profesor (cod_pro : tira(3), nombre : tira(40), teléfono: entero,
cod_dep: tira(5))

Docencia (cod_asg: tira(3), cod_pro: tira(3), gteo: entero, gpra: entero)

El lenguaje SQL.

BD relacional

Departamento

cod_dep	nombre	director	teléfono
DSIC	Sistemas Informáticos y Computación	V. Botti	3500
DISCA	Ingeniería de Sistemas, Computadores y Automática	A. Crespo	5700
MAT	Matemática Aplicada	P. Pérez	6600
FIS	Física Aplicada	J. Linares	5200
IDM	Idiomas	B. Montero	5300
EIO	Estadística e Investigación Operativa	L. Barceló	4900
OEM	Org. de Empresas, Economía Financ. y Contabilidad	M. Pérez	6800

Asignatura

cod_asg	nombre	semestre	teoría	prac	cod_dep
BDA	Bases de Datos	2B	3	3	DSIC
AD1	Algoritmos y Estructuras de Datos 1	1A	4	2	DSIC
FCO	Fundamentos de computadores	1A	4,5	4,5	DISCA
MAD	Matemática Discreta	1A	3	3	MAT
INT	Inglés Técnico	1B	3	3	IDM
FFI	Fundamentos Físicos de la Informática	1A	3	3	FIS
EC2	Estructuras de Computadores 2	2A	3	3	DISCA

Docencia

cod_asg	cod_pro	gteo	gpra
BDA	JCC	2	4
MAD	RFC	1	2
FCO	DGT	2	2
AD1	MAF	1	1
INT	CPG	1	0
EC2	JBD	2	0
BDA	MCG	1	3
AD1	JCC	1	1
FCO	JBD	2	2
AD1	MCG	1	1

Profesor

cod_pro	nombre	teléfono	cod_dep
JCC	Juan C. Casamayor Ródenas	7796	DSIC
RFC	Robert Fuster i Capilla	6789	MAT
JBD	José V. Benlloch Dualde	5760	DISCA
MAF	María Alpuente Frasnado	3560	DSIC
CPG	Cristina Pérez Guillot	7439	IDM
JTM	José M. Torralba Martínez	4590	OEM
IGP	Ignacio Gil Pechuán	3423	OEM
DGT	Daniel Gil Tomás	5679	DISCA
MCG	Matilde Celma Giménez	7756	DSIC

El lenguaje SQL.

Consultas sobre varias relaciones.

```
SELECT [ALL | DISTINCT]  $A_{1i}, A_{2j}, \dots A_{nk}$  | *  
FROM  $R_1, R_2, \dots R_n$   
[WHERE condición]
```

de las tuplas de $R_1, R_2, \dots R_n$ para las cuales la *condición* se evalúa a **CIERTO**, obtener el valor de los atributos $A_{1i}, A_{2j}, \dots A_{nk}$

A_{1i} : denota el i ésimo atributo de la relación R_1

El lenguaje SQL.

Obtener de todos los profesores: su código, su nombre y el nombre de su departamento de adscripción.

```
SELECT cod_pro, nombre, cod_dep  
FROM Profesor
```

```
SELECT Profesor.cod_pro, Profesor.nombre,  
       Departamento.nombre  
FROM Profesor, Departamento  
WHERE Profesor.cod_dep = Departamento.cod_dep
```

De los pares de tuplas de *Profesor* y *Departamento* para los cuales la condición se evalúa a **CIERTO** obtener el valor de los atributos *cod_pro* y *nombre* de *Profesor* y *nombre* de *Departamento*.

El lenguaje SQL.

Profesor

cod_pro	nombre	teléfono	cod_dep
JCC	Juan C. Casamayor Ródenas	7796	DSIC
RFC	Robert Fuster i Capilla	6789	MAT
JBD	José V. Benlloch Dualde	5760	DISCA
MAF	María Alpuente Frasnado	3560	DSIC
CPG	Cristina Pérez Guillot	7439	IDM
JTM	José M. Torralba Martínez	4590	OEM
IGP	Ignacio Gil Pechuán	3423	OEM
DGT	Daniel Gil Tomás	5679	DISCA
MCG	Matilde Celma Giménez	7756	DSIC

Departamento

cod_dep	nombre	director	teléfono
DSIC	Sistemas Informáticos y Computación	V. Botti	3500
DISCA	Ingeniería de Sistemas, Computadores y Automática	A. Crespo	5700
MAT	Matemática Aplicada	P. Pérez	6600
FIS	Física Aplicada	J. Linares	5200
IDM	Idiomas	B. Montero	5300
EIO	Estadística e Investigación Operativa	L. Barceló	4900
OEM	Org. de Empresas, Economía Financ. y Contabilidad	M. Pérez	6800

las tuplas
marcadas de
Profesor y
Departamento
cumplen la
condición

Profesor.cod_pro	Profesor.nombre	Departamento.nombre
JCC	Juan C. Casamayor Ródenas	Sistemas Informáticos y Computación
RFC	Robert Fuster i Capilla	Matemática Aplicada
JBD	José V. Benlloch Dualde	Ingeniería de Sistemas Computadores y Automática
MAF	María Alpuente Frasnado	Sistemas Informáticos y Computación
CPG	Cristina Pérez Guillot	Idiomas
JTM	José M. Torralba Martínez	Organización de Empresas
IGP	Ignacio Gil Pechuán	Organización de Empresas
DGT	Daniel Gil Tomás	Ingeniería de Sistemas Computadores y Automática
MCG	Matilde Celma Giménez	Sistemas Informáticos y Computación

relación resultante

El lenguaje SQL.

Profesor

cod_pro	nombre	teléfono	cod_dep
JCC	Juan C. Casamayor Ródenas	7796	DSIC
RFC	Robert Fuster i Capilla	6789	MAT
JBD	José V. Benlloch Dualde	5760	DISCA
MAF	María Alpuente Frasnado	3560	DSIC
CPG	Cristina Pérez Guillot	7439	IDM
JTM	José M. Torralba Martínez	4590	OEM
IGP	Ignacio Gil Pechuán	3423	OEM
DGT	Daniel Gil Tomás	5679	DISCA
MCG	Matilde Celma Giménez	7756	DSIC

Departamento

cod_dep	nombre	director	teléfono
DSIC	Sistemas Informáticos y Computación	V. Botti	3500
DISCA	Ingeniería de Sistemas, Computadores y Automática	A. Crespo	5700
MAT	Matemática Aplicada	P. Pérez	6600
FIS	Física Aplicada	J. Linares	5200
IDM	Idiomas	B. Montero	5300
EIO	Estadística e Investigación Operativa	L. Barceló	4900
OEM	Org. de Empresas, Economía Financ. y Contabilidad	M. Pérez	6800

las tuplas
marcadas de
Profesor y
Departamento
no cumplen la
condición

Profesor.cod_pro	Profesor.nombre	Departamento.nombre
JCC	Juan C. Casamayor Ródenas	Sistemas Informáticos y Computación
RFC	Robert Fuster i Capilla	Matemática Aplicada
JBD	José V. Benlloch Dualde	Ingeniería de Sistemas, Computadores y Automática
MAF	María Alpuente Frasnado	Sistemas Informáticos y Computación
CPG	Cristina Pérez Guillot	Idiomas
JTM	José M. Torralba Martínez	Organización de Empresas
IGP	Ignacio Gil Pechuán	Organización de Empresas
DGT	Daniel Gil Tomás	Ingeniería de Sistemas, Computadores y Automática
MCG	Matilde Celma Giménez	Sistemas Informáticos y Computación

relación resultante

El lenguaje SQL.

Profesor

cod_pro	nombre	teléfono	cod_dep
JCC	Juan C. Casamayor Ródenas	7796	DSIC
RFC	Robert Fuster i Capilla	6789	MAT
JBD	José V. Benlloch Dualde	5760	DISCA
MAF	María Alpuente Frasnado	3560	DSIC
CPG	Cristina Pérez Guillot	7439	IDM
JTM	José M. Torralba Martínez	4590	OEM
IGP	Ignacio Gil Pechuán	3423	OEM
DGT	Daniel Gil Tomás	5679	DISCA
MCG	Matilde Celma Giménez	7756	DSIC

Departamento

cod_dep	nombre	director	teléfono
DSIC	Sistemas Informáticos y Computación	V. Botti	3500
DISCA	Ingeniería de Sistemas, Computadores y Automática	A. Crespo	5700
MAT	Matemática Aplicada	P. Pérez	6600
FIS	Física Aplicada	J. Linares	5200
IDM	Idiomas	B. Montero	5300
EIO	Estadística e Investigación Operativa	L. Barcelo	4900
OEM	Org. de Empresas, Economía Financ. y Contabilidad	M. Pérez	6800

las tuplas
marcadas de
Profesor y
Departamento
cumplen la
condición

Profesor.cod_pro	Profesor.nombre	Departamento.nombre
JCC	Juan C. Casamayor Ródenas	Sistemas Informáticos y Computación
RFC	Robert Fuster i Capilla	Matemática Aplicada
JBD	José V. Benlloch Dualde	Ingeniería de Sistemas Computadores y Automática
MAF	María Alpuente Frasnado	Sistemas Informáticos y Computación
CPG	Cristina Pérez Guillot	Idiomas
JTM	José M. Torralba Martínez	Organización de Empresas
IGP	Ignacio Gil Pechuán	Organización de Empresas
DGT	Daniel Gil Tomás	Ingeniería de Sistemas Computadores y Automática
MCG	Matilde Celma Giménez	Sistemas Informáticos y Computación

relación resultante

El lenguaje SQL.

Consultas sobre varias relaciones.

SELECT [ALL | DISTINCT] $A_{1i}, A_{2j}, \dots A_{nk}$ | *
FROM $R_1, R_2, \dots R_n$
[WHERE *condición*]



$R_1 \times R_2 \times \dots \times R_n$ DONDE *condición* [$A_{1i}, A_{2j}, \dots, A_{nk}$]

Álgebra
Relacional

El lenguaje SQL.

Consultas sobre varias relaciones.

SELECT [ALL | DISTINCT] $A_{1i}, A_{2j}, \dots A_{nk}$ | *
FROM $R_1, R_2, \dots R_n$
[WHERE *condición*]



Cálculo
Relacional
de Tuplas

$\{R_1X.A_{1i}, R_2X.A_{2j}, \dots, R_nX.A_{nk} | R_1(R_1X) \wedge R_2(R_2X) \wedge \dots \wedge R_n(R_nX) \wedge \textit{condición} \}$

Se asume que para cada relación que aparece en el FROM existe una variable-tupla declarada sobre el esquema de la relación.

$R_1X: R_1, R_2X: R_2, \dots, R_nX: R_n$

El lenguaje SQL.

```
SELECT cod_pro, Profesor.nombre, Departamento.nombre  
FROM Profesor, Departamento  
WHERE Profesor.cod_dep = Departamento.cod_dep
```

¡ la calificación de los atributos sólo es necesaria cuando puede existir ambigüedad!

El lenguaje SQL.

Para cada relación en el FROM se puede declarar una variable de recorrido específica: PX, DX

```
SELECT PX.cod_pro, PX.nombre, DX.nombre  
FROM Profesor PX, Departamento DX  
WHERE PX.cod_dep = DX.cod_dep
```



$\{PX.cod_pro, PX.nombre, DX.nombre \mid$
 $Profesor(PX) \wedge Departamento(DX) \wedge PX.cod_dep=DX.cod_dep\}$

Cálculo
Relacional
de Tuplas

El lenguaje SQL.

Obtener la docencia de los profesores del DSIC: nombre del profesor, nombre de la asignatura impartida y grupos impartidos.

```
SELECT PX.nombre, AX.nombre, DX.gteo, DX.gpra
FROM Profesor PX, Docencia DX, Asignatura AX
WHERE PX.cod_pro = DX.cod_pro
        AND
        AX.cod_asg = DX.cod_asg
        AND
        PX.cod_dep="DSIC"
```

El lenguaje SQL.

Asignatura

cod_asg	nombre	semestre	teoría	prac	cod_dep
BDA	Bases de Datos	2B	3	3	DSIC
AD1	Algoritmos y Estructuras de Datos 1	1A	4	2	DSIC
FCO	Fundamentos de computadores	1A	4,5	4,5	DISCA
MAD	Matemática Discreta	1A	3	3	MAT
INT	Inglés Técnico	1B	3	3	IDM
FFI	Fundamentos Físicos de la Informática	1A	3	3	FIS
EC2	Estructuras de Computadores 2	2A	3	3	DISCA

Profesor

cod_pro	nombre	teléfono	cod_dep
JCC	Juan C. Casamayor Ródenas	7796	DSIC
RFC	Robert Fuster i Capilla	6789	MAT
JBD	José V. Benlloch Dualde	5760	DISCA
MAF	María Alpuente Frasnado	3560	DSIC
CPG	Cristina Pérez Guillot	7439	IDM
JTM	José M. Torralba Martínez	4590	OEM
IGP	Ignacio Gil Pechuán	3423	OEM
DGT	Daniel Gil Tomás	5679	DISCA
MCG	Matilde Celma Giménez	7756	DSIC

cod_asg	cod_pro	gteo	gpri
BDA	JCC	2	4
MAD	RFC	1	2
FCO	DGT	2	2
AD1	MAF	1	1
INT	CPG	1	0
EC2	JBD	2	0
BDA	MCG	1	3
AD1	JCC	1	1
FCO	JBD	2	2
AD1	MCG	1	1

Docencia

las tuplas marcadas de Profesor, Asignatura y Docencia cumplen la condición

El lenguaje SQL.

Obtener el nombre de los profesores que imparten mas de una asignatura.

```
SELECT PX.nombre
FROM Profesor PX, Docencia D1X, Docencia D2X
WHERE PX.cod_pro = D1X.cod_pro
        AND
        PX.cod_pro = D2X.cod_pro
        AND
        D1X.cod_asg <> D2X.cod_asg
```

En este ejemplo el uso de las variables de recorrido (alias de relación) es imprescindible.

El lenguaje SQL.

Obtener el nombre de los profesores que imparten asignaturas que no son de su departamento.

```
SELECT PX.nombre
FROM Profesor PX, Asignatura AX, Docencia DX
WHERE PX.cod_pro = DX.cod_pro
        AND
        AX.cod_asg = DX.cod_asg
        AND
        PX.cod_dep <> AX.cod_dep
```

El lenguaje SQL.

Consultas sobre varias relaciones.

SELECT [ALL | DISTINCT] $A_{1i}, A_{2j}, \dots A_{nk} \mid ^*$
FROM $R_1, R_2, \dots R_n$
[WHERE *condición*]

condiciones de concatenación de
las relaciones $R_1, R_2, \dots R_n$

AND

condiciones de la consulta

la concatenación entre $R_1, R_2, \dots R_n$ se realiza
usualmente sobre los atributos de referencia de las
relaciones.



El lenguaje SQL.

Subconsultas.

En una consulta (SELECT externa) se utiliza una subconsulta (SELECT interna) para obtener datos que serán utilizados como argumentos de predicados de la consulta.

El lenguaje SQL.

Obtener el nombre de los profesores adscritos al mismo departamento que 'JCC'.

```
SELECT PX.nombre
FROM  Profesor PX
WHERE PX.cod_dep
      =
      (SELECT PY.cod_dep
       FROM Profesor PY
       WHERE PY.cod_pro='JCC')
```

subconsulta: devuelve un valor utilizado como operando del predicado de comparación =

El lenguaje SQL.

Obtener el nombre de los profesores adscritos al mismo departamento que 'JCC'.

```
SELECT PX.nombre  
FROM Profesor PX, Profesor PY  
WHERE PX.cod_dep =  
        PY.cod_dep  
        AND  
        PY.cod_pro='JCC'
```

¡en ocasiones las subconsultas no son imprescindibles aunque sirven para estructurar la consulta!

El lenguaje SQL.

Obtener el nombre de los profesores que imparten alguna de las asignaturas impartidas por el profesor de código 'JCC'.

```
SELECT PX.nombre
FROM Profesor PX, Docencia DX
WHERE PX.cod_pro = DX.cod_pro
        AND
        DX.cod_asg IN (SELECT DY.cod_asg
                        FROM Docencia DY
                        WHERE DY.cod_pro='JCC')
```

subconsulta: devuelve una lista de valores utilizada como argumento del predicado IN

El lenguaje SQL.

Obtener el nombre de los profesores que imparten alguna asignatura de mas de 6 créditos.

```
SELECT nombre  
FROM Profesor  
WHERE
```

```
cod_pro IN (SELECT cod_pro  
FROM Docencia  
WHERE
```


```
cod_asg IN (SELECT cod_asg  
FROM Asignatura  
WHERE (teoría+prac)>6))
```

¡las subconsultas se pueden anidar!

El lenguaje SQL.

Obtener el nombre de las asignaturas que tienen menos créditos.

```
SELECT AX.nombre
FROM Asignatura AX
WHERE (AX.teoría + AX.prac)
      <
      (SELECT AY.teoría + AY.prac
       FROM Asignatura AY)
```



subconsulta: devuelve una lista de valores (no puede ser utilizada como operando de un predicado de comparación).

El lenguaje SQL.

Predicados que admiten subconsultas como argumentos:

- ✓ predicados de comparación α : =, <>, >, <, >=, <=.

[expresión| subconsulta] α [expresión| subconsulta]

- ✓ predicado IN: expresión IN (subconsulta)

- ✓ predicados de comparación cuantificados:

expresión α [ALL| ANY] (subconsulta)

- ✓ predicado EXISTS: EXISTS (subconsulta)

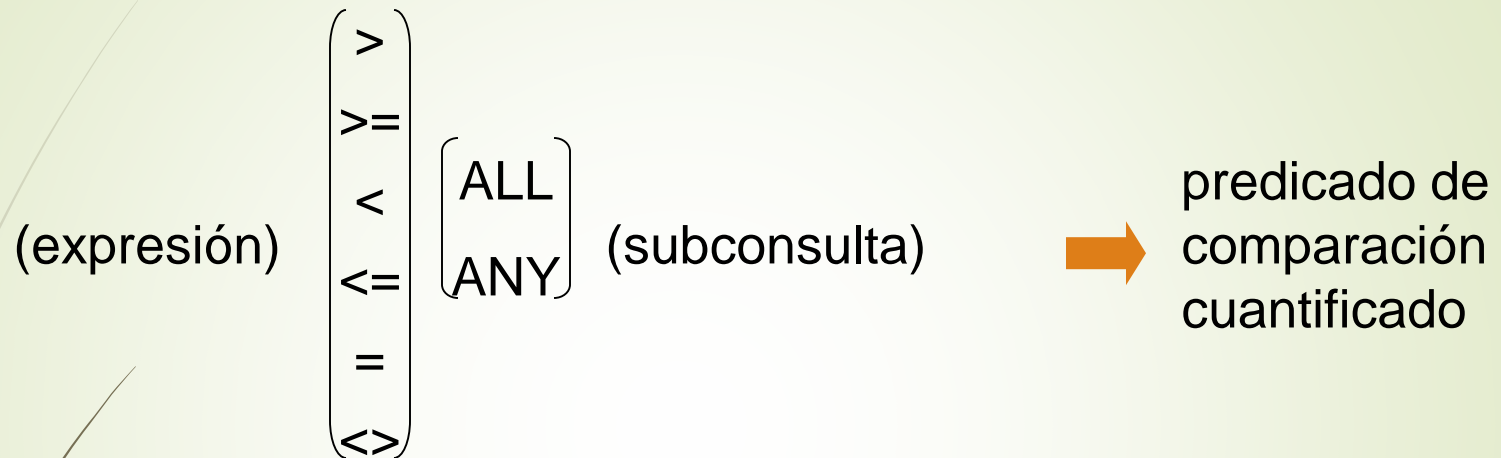
- ✓ predicado MATCH

- ✓ predicado UNIQUE

Si la evaluación de la subconsulta es vacía, se devuelve una tupla de valores nulos.

El lenguaje SQL.

Predicados de comparación cuantificados:



- ✓ Los predicados de comparación cuantificados con **ALL** se evalúan a CIERTO si la comparación se evalúa a CIERTO para todos los valores devueltos por la subconsulta.
- ✓ Los predicados de comparación cuantificados con **ANY** o **SOME** se evalúan a CIERTO si la comparación se evalúa a CIERTO para algún valor devuelto por la subconsulta.

El lenguaje SQL.

Obtener el nombre de las asignaturas que tienen el mayor número de créditos.

```
SELECT  AX.nombre  
FROM    Asignatura AX  
WHERE   (AX.teoría + AX.prac)  
          >= ALL
```


```
(SELECT DISTINCT (AY.teoría + AY.prac)  
  FROM Asignatura AY)
```

subconsulta: devuelve una lista de valores
(créditos de todas las asignaturas).

El lenguaje SQL.

SELECT AX.nombre
FROM Asignatura AX
WHERE (AX.teoría + AX.prac)
>= ALL

(SELECT DISTINCT (AY.teoría + AY.prac)
FROM Asignatura AY)



6
9

cod_asg	nombre	semestre	teoría	prac	cod_dep
BDA	Bases de Datos	2B	3	3	DSIC
AD1	Algoritmos y Estructuras de Datos 1	1A	4	2	DSIC
FCO	Fundamentos de computadores	1A	4,5	4,5	DISCA
MAD	Matemática Discreta	1A	3	3	MAT
INT	Inglés Técnico	1B	3	3	IDM
FFI	Fundamentos Físicos de la Informática	1A	3	3	FIS
EC2	Estructuras de Computadores 2	2A	3	3	DISCA

>=
ALL

6
9

El lenguaje SQL.

Obtener el nombre de las asignaturas que no son las de menor número de créditos.

```
SELECT  AX.nombre  
FROM    Asignatura AX  
WHERE   (AX.teoría + AX.prac)  
          > ANY
```


```
(SELECT DISTINCT (AY.teoría + AY.prac)  
  FROM Asignatura AY)
```

subconsulta: devuelve una lista de valores
(créditos de todas las asignaturas).

El lenguaje SQL.

SELECT AX.nombre
FROM Asignatura AX
WHERE (AX.teoría + AX.prac)
> ANY

(SELECT DISTINCT (AY.teoría + AY.prac)
FROM Asignatura AY)



6
9

cod_asg	nombre	semestre	teoría	prac	cod_dep
BDA	Bases de Datos	2B	3	3	DSIC
AD1	Algoritmos y Estructuras de Datos 1	1A	4	2	DSIC
FCO	Fundamentos de computadores	1A	4,5	4,5	DISCA
MAD	Matemática Discreta	1A	3	3	MAT
INT	Inglés Técnico	1B	3	3	IDM
FFI	Fundamentos Físicos de la Informática	1A	3	3	FIS
EC2	Estructuras de Computadores 2	2A	3	3	DISCA

> ANY

6
9

El lenguaje SQL.

```
SELECT AX.nombre
FROM Asignatura AX
WHERE (AX.teoría+AX.prac)
      >= ALL
      (SELECT DISTINCT (AY.teoría + AY.prac)
       FROM Asignatura AY)
```

```
SELECT AX.nombre
FROM Asignatura AX
WHERE (AX.teoría+AX.prac)
      >=
      (SELECT MAX (AY.teoría + AY.prac)
       FROM Asignatura AY)
```

```
SELECT AX.nombre
FROM Asignatura AX
WHERE (AX.teoría+AX.prac)
      > ANY
      (SELECT DISTINCT (AY.teoría + AY.prac)
       FROM Asignatura AY)
```

```
SELECT AX.nombre
FROM Asignatura AX
WHERE (AX.teoría+AX.prac)
      >
      (SELECT MIN (AY.teoría + AY.prac)
       FROM Asignatura AY)
```

El lenguaje SQL.

Evaluación de los predicados de comparación, el predicado IN, y los predicados de comparación cuantificados con subconsultas vacías:

[expresión| subconsulta] α [expresión| subconsulta] INDEFINIDO

expresión IN (subconsulta) FALSO

expresión α ALL (subconsulta) CIERTO

expresión α ANY (subconsulta) FALSO



Subconsultas relacionadas

El lenguaje SQL.

BD relacional

Departamento

cod_dep	nombre	director	teléfono
DSIC	Sistemas Informáticos y Computación	V. Botti	3500
DISCA	Ingeniería de Sistemas, Computadores y Automática	A. Crespo	5700
MAT	Matemática Aplicada	P. Pérez	6600
FIS	Física Aplicada	J. Linares	5200
IDM	Idiomas	B. Montero	5300
EIO	Estadística e Investigación Operativa	L. Barceló	4900
OEM	Org. de Empresas, Economía Financ. y Contabilidad	M. Pérez	6800

Asignatura

cod_asg	nombre	semestre	teoría	prac	cod_dep
BDA	Bases de Datos	2B	3	3	DSIC
AD1	Algoritmos y Estructuras de Datos I	1A	4	2	DSIC
FCO	Fundamentos de computadores	1A	4,5	4,5	DISCA
MAD	Matemática Discreta	1A	3	3	MAT
INT	Inglés Técnico	1B	3	3	IDM
FFI	Fundamentos Físicos de la Informática	1A	3	3	FIS
EC2	Estructuras de Computadores 2	2A	3	3	DISCA

Docencia

cod_asg	cod_pro	gteo	gpra
BDA	JCC	2	4
MAD	RFC	1	2
FCO	DGT	2	2
AD1	MAF	1	1
INT	CPG	1	0
EC2	JBD	2	0
BDA	MCG	1	3
AD1	JCC	1	1
FCO	JBD	2	2
AD1	MCG	1	1

Profesor

cod_pro	nombre	teléfono	cod_dep
JCC	Juan C. Casamayor Ródenas	7796	DSIC
RFC	Robert Fuster i Capilla	6789	MAT
JBD	José V. Benlloch Dualde	5760	DISCA
MAF	María Alpuente Frasnado	3560	DSIC
CPG	Cristina Pérez Guillot	7439	IDM
JTM	José M. Torralba Martínez	4590	OEM
IGP	Ignacio Gil Pechuán	3423	OEM
DGT	Daniel Gil Tomás	5679	DISCA
MCG	Matilde Celma Giménez	7756	DSIC

El lenguaje SQL.

Subconsultas relacionadas.

```
SELECT AX.nombre
FROM Asignatura AX
WHERE (AX.teoría+AX.prac)
      >=
      (SELECT MAX (AY.teoría + AY.prac) FROM Asignatura AY)
```

El lenguaje SQL.

Subconsultas relacionadas.

En la subconsulta se hace referencia a relaciones de la consulta más externa.

El lenguaje SQL.

Obtener el nombre de las asignaturas que tienen el mayor número de créditos.

```
SELECT AX.nombre  
FROM Asignatura AX  
WHERE (AX.teoría+AX.prac)
```

subconsulta independiente

la subconsulta se evalúa una única vez

```
SELECT MAX (AY.teoría + AY.prac) FROM Asignatura AY)
```

Obtener el nombre de las asignaturas que tienen el mayor número de créditos en su departamento.

```
SELECT AX.nombre  
FROM Asignatura AX  
WHERE (AX.teoría+AX.prac)
```

subconsulta dependiente

la subconsulta se evalúa una vez para cada tupla de Asignatura (AX)

```
SELECT MAX (AY.teoría + AY.prac) FROM Asignatura AY  
WHERE AY.cod_dep=AX.cod_dep)
```


El lenguaje SQL.

Predicado EXISTS

EXISTS (*subconsulta*)

El predicado **EXISTS** se evalúa a **CIERTO** si la subconsulta (sentencia SELECT) devuelve al menos una fila, en caso contrario se evalúa a **FALSO**.

El lenguaje SQL.

Obtener el nombre de los profesores que imparten más de 4 grupos en alguna asignatura.

```
SELECT PX.nombre
FROM Profesor PX
WHERE EXISTS (SELECT * FROM Docencia DX
              WHERE DX.cod_pro=PX.cod_pro
                 AND
                 (DX.gteo+DX.gpra)>4)
```

El lenguaje SQL.

```
SELECT PX.nombre
FROM Profesor PX
WHERE EXISTS (SELECT * FROM Docencia DX
              WHERE DX.cod_pro=PX.cod_pro
              AND
              (DX.gteo+DX.gpra)>4)
```



```
SELECT PX.nombre
FROM Profesor PX
WHERE PX.cod_pro IN (SELECT DX.cod_pro
                    FROM Docencia DX
                    WHERE (DX.gteo+DX.gpra)>4)
```

El lenguaje SQL.

¡El predicado EXISTS es equivalente al cuantificador existencial de la lógica!

EXISTS (SELECT * FROM R RX
WHERE *Condición*)

SQL



$\exists x (R(x) \wedge \textit{Condición})$

AR

El lenguaje SQL.

¡El predicado EXISTS es equivalente al cuantificador existencial de la lógica!

```
SELECT PX.nombre
FROM Profesor PX
WHERE EXISTS (SELECT * FROM Docencia DX
              WHERE DX.cod_pro= PX.cod_pro
                  AND
                  (DX.gteo+DX.gpra) > 4 )
```

SQL



```
PX.nombre | PX:Profesor ∧
    ∃ DX: Docencia
    PX.cod_pro ∧
    (DX.cod_pro =
    (DX.gteo+DX.gpra) > 4))
```

AR

El lenguaje SQL.

Predicado **EXISTS**: Cuantificación universal.

¡ El cuantificador universal no existe en SQL !

$$\forall X F(X) \equiv \neg \exists X \neg F(X)$$

$$A \rightarrow B \equiv \neg A \vee B$$

$$\forall X (A(X) \rightarrow B(X))$$

\equiv

$$\neg \exists X \neg (A(X) \rightarrow B(X))$$

\equiv

$$\neg \exists X (A(X) \wedge \neg B(X))$$

El lenguaje SQL.

Obtener el nombre de los profesores que imparten todas las asignaturas.

PX.nombre | PX:Profesor \wedge

$\forall AX: \text{Asignatura} \rightarrow \exists DX: \text{Docencia} \wedge$

$DX.\text{cod_pro} = PX.\text{cod_pro} \wedge DX.\text{cod_asg} = AX.\text{cod_asg}$)



$$\forall X (A(X) \rightarrow B(X)) \equiv \neg \exists X (A(X) \wedge \neg B(X))$$

PX.nombre | PX:Profesor \wedge

$\neg \exists AX: \text{Asignatura} \wedge \neg \exists DX: \text{Docencia} \wedge$

$DX.\text{cod_pro} = PX.\text{cod_pro} \wedge DX.\text{cod_asg} = AX.\text{cod_asg}))$

Obtener el nombre de los profesores que imparten todas las asignaturas.

AR

$$\begin{aligned} & \text{PX.nombre} \mid \text{PX:Profesor} \wedge \\ & \quad \neg \exists \text{AX: Asignatura} \quad \wedge \quad \neg \exists \text{DX: Docencia} \wedge \\ & \quad \quad \text{DX.cod_pro} = \text{PX.cod_pro} \wedge \text{DX.cod_asg} = \text{AX.cod_asg})) \} \end{aligned}$$

SQL

[illegible]

El lenguaje SQL.

Obtener el nombre de los profesores que imparten todas las asignaturas de su departamento.

PX.nombre | PX: Profesor

$\forall AX: \text{Asignatura} \wedge AX.\text{cod_dep} = PX.\text{cod_dep}$

\rightarrow

$\exists DX: \text{Docencia} \wedge$

$DX.\text{cod_pro} = PX.\text{cod_pro} \wedge DX.\text{cod_asg} = AX.\text{cod_asg})$



PX.nombre | PX: Profesor \wedge

$\neg \exists AX: \text{Asignatura} \wedge AX.\text{cod_dep} = PX.\text{cod_dep}$

\wedge

$\neg \exists DX: \text{Docencia} \wedge$

$DX.\text{cod_pro} = PX.\text{cod_pro} \wedge DX.\text{cod_asg} = AX.\text{cod_asg}$

El lenguaje SQL.

AR

PX.nombre | PX: Profesor \wedge

$\neg \exists \text{AX: Asignatura} \wedge \text{AX.cod_dep} = \text{PX.cod_dep}$

\wedge

$\neg \exists \text{DX: Docencia} \wedge$

$\text{DX.cod_pro} = \text{PX.cod_pro} \wedge \text{DX.cod_asg} = \text{AX.cod_asg}$

SQL

SELECT PX.nombre

FROM Profesor PX

WHERE NOT EXISTS (SELECT * FROM Asignatura AX

WHERE AX.cod_dep= PX.cod_dep

AND

NOT EXISTS (SELECT * FROM Docencia DX

WHERE DX.cod_pro=PX.cod_pro \wedge

DX.cod_asg=AX.cod_asg))

El lenguaje SQL.

Predicado UNIQUE

UNIQUE (*subconsulta*)

El predicado **UNIQUE** se evalúa a **CIERTO** si en la subconsulta (sentencia **SELECT**) no hay tuplas duplicadas, en caso contrario se evalúa a **FALSO**.

El lenguaje SQL.

Evaluación de los predicados EXISTS, UNIQUE con subconsultas vacías:

EXISTS (subconsulta)

FALSO

expresión UNIQUE (subconsulta)

CIERTO



Consultas agrupadas

El lenguaje SQL.

Consultas agrupadas.

```
SELECT [ALL | DISTINCT]  $A_{1i}, A_{2j}, \dots, A_{nk}$  *  
FROM  $R_1, R_2, \dots, R_n$   
[WHERE condición]  
[GROUP BY  $B_1, B_2, \dots, B_m$ ]  
[HAVING condición]
```

El lenguaje SQL.

Consulta de valores de grupo:

SELECT [**ALL** | **DISTINCT**] *funciones agregadas*
FROM R
[**WHERE** *condición*]

valores globales para el conjunto de tuplas que cumplen la condición.

AVG
MAX
MIN
SUM
COUNT

COUNT (*)

([**ALL** | **DISTINCT**] *expresión_escalar*)

El lenguaje SQL.

Obtener el número total de créditos de las asignaturas de primer curso.

```
SELECT SUM (teoría) + SUM (prac)  
FROM Asignatura  
WHERE (semestre='1A' OR semestre ='1B')
```

```
SELECT SUM (teoría + prac)  
FROM Asignatura  
WHERE (semestre='1A' OR semestre ='1B')
```

- Los valores nulos son ignorados para realizar los cálculos.
- Si el número de filas de la seleccionadas es 0, la función COUNT devuelve el valor 0 y las otras funciones el valor nulo.

El lenguaje SQL.

cod_asg	nombre	semestre	teoría	prac	cod_dep
BDA	Bases de Datos	2B	3	3	DSIC
AD1	Algoritmos y Estructuras de Datos I	1A	4	2	DSIC
FCO	Fundamentos de computadores	1A	4,5	4,5	DISCA
MAD	Matemática Discreta	1A	3	3	MAT
INT	Inglés Técnico	1B	3	3	IDM
FFI	Fundamentos Físicos de la Informática	1A	3	3	FIS
EC2	Estructuras de Computadores 2	2A	3	3	DISCA

SELECT SUM (teoría) + SUM (prac)
FROM Asignatura
WHERE (semestre="1A" OR semestre ="1B")

El lenguaje SQL.

Obtener el número total de profesores de cada departamento.

Profesor

cod_pro	nombre	teléfono	cod_dep
JCC	Juan C. Casamayor Ródenas	7796	DSIC
RFC	Robert Fuster i Capilla	6789	MAT
JBD	José V. Benlloch Dualde	5760	DISCA
MAF	María Alpuente Frasnado	3560	DSIC
CPG	Cristina Pérez Guillot	7439	IDM
JTM	José M. Torralba Martínez	4590	OEM
IGP	Ignacio Gil Pechuán	3423	OEM
DGT	Daniel Gil Tomás	5679	DISCA
MCG	Matilde Celma Giménez	7756	DSIC

cod_dep	
DSIC	3
MAT	1
DISCA	2
IDM	1
OEM	2

El lenguaje SQL.

Consultas agrupadas.

```
SELECT [ALL | DISTINCT]  $A_{1i}, A_{2j}, \dots, A_{nk}$  | *  
FROM  $R_1, R_2, \dots, R_n$   
[WHERE condición]  
[GROUP BY  $B_1, B_2, \dots, B_m$ ]  
[HAVING condición]
```

GROUP BY: define grupos de tuplas en el conjunto de tuplas seleccionadas por la condición WHERE. Los grupos se definen por la igualdad de valor en los atributos de agrupación (B_1, B_2, \dots, B_m).

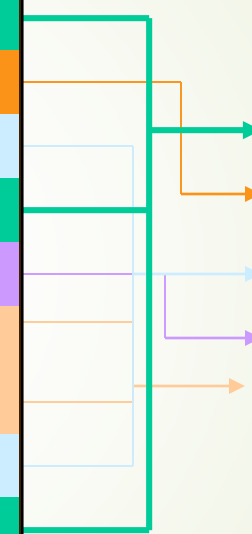
HAVING: de los grupos definidos se seleccionan aquellos que cumplen la condición expresada.

El lenguaje SQL.

Obtener el número total de profesores de cada departamento.

Profesor

cod_pro	nombre	teléfono	cod_dep
JCC	Juan C. Casamayor Ródenas	7796	DSIC
RFC	Robert Fuster i Capilla	6789	MAT
JBD	José V. Benlloch Dualde	5760	DISCA
MAF	María Alpuente Frasnado	3560	DSIC
CPG	Cristina Pérez Guillot	7439	IDM
JTM	José M. Torralba Martínez	4590	OEM
IGP	Ignacio Gil Pechuán	3423	OEM
DGT	Daniel Gil Tomás	5679	DISCA
MCG	Matilde Celma Giménez	7756	DSIC



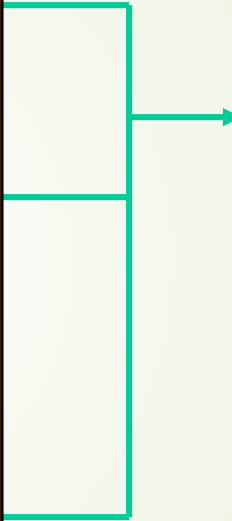
cod_dep	
DSIC	3
MAT	1
DISCA	2
IDM	1
OEM	2

```
SELECT cod_dep, COUNT (*)  
FROM Profesor  
GROUP BY cod_dep
```

El lenguaje SQL.

Obtener el número total de profesores de los departamentos que tienen mas de 2 profesores.

cod_pro	nombre	teléfono	cod_dep
JCC	Juan C. Casamayor Ródenas	7796	DSIC
RFC	Robert Fuster i Capilla	6789	MAT
JBD	José V. Benlloch Dualde	5760	DISCA
MAF	María Alpuente Frasnado	3560	DSIC
CPG	Cristina Pérez Guillot	7439	IDM
JTM	José M. Torralba Martínez	4590	OEM
IGP	Ignacio Gil Pechuán	3423	OEM
DGT	Daniel Gil Tomás	5679	DISCA
MCG	Matilde Celma Giménez	7756	DSIC



cod_dep	
DSIC	3

```
SELECT cod_dep, COUNT (*)  
FROM Profesor  
GROUP BY cod_dep  
HAVING COUNT (*) > 2
```

El lenguaje SQL.

Consultas agrupadas.

```
SELECT [ALL | DISTINCT]  $A_{1i}, A_{2j}, \dots, A_{nk}$  | *  
FROM  $R_1, R_2, \dots, R_n$   
[WHERE condición]  
[GROUP BY  $B_1, B_2, \dots, B_m$ ]  
[HAVING condición]
```

En una consulta agrupada:

- ✓ los datos seleccionados en la **SELECT** deben ser datos comunes al grupo: atributos de agrupación o funciones agregadas para obtener valores globales del grupo.
- ✓ en la condición del **HAVING** sólo se puede hacer referencia a datos comunes al grupo:.

El lenguaje SQL.

Obtener el número total de profesores y el nombre de los mismos, para aquellos departamentos que tienen mas de 2 profesores.

```
SELECT cod_dep, nombre, COUNT (*)  
FROM Profesor  
GROUP BY cod_dep  
HAVING COUNT (*) > 2
```

¡ el nombre del profesor es un dato a nivel de tupla
no es un dato a nivel de grupo de tuplas !

El lenguaje SQL.

Obtener para cada departamento el número total de asignaturas de mas de seis créditos que tienen adscritas.

```
SELECT cod_dep, COUNT (*)  
FROM Asignatura  
WHERE (teoría + prac) > 6  
GROUP BY cod_dep
```

Obtener los departamento que tienen mas de 3 asignaturas adscritas de primer curso, indicando el número de éstas.

```
SELECT cod_dep, COUNT (*)  
FROM Asignatura  
WHERE semestre IN ('1A', '1B')  
GROUP BY cod_dep  
HAVING COUNT (*) > 3
```


El lenguaje SQL.

Evaluación:

Obtener los departamento que tienen mas de 3 asignaturas adscritas de primer curso, indicando el número de éstas.

```
SELECT cod_dep, COUNT (*)  
FROM Asignatura  
WHERE semestre IN ('1A', '1B') (1)  
GROUP BY cod_dep (2)  
HAVING COUNT (*) > 3 (3)
```

El lenguaje SQL.

Obtener para cada departamento los créditos de docencia impartidos en asignaturas de primer curso adscritas al departamento.

```
SELECT A. cod_dep,  
        SUM(D.gteo * A.teoría) + SUM(D.gpra*A.prác)  
FROM Docencia D, Asignatura A  
WHERE A.cod_asg = D.cod_asg  
        AND  
        semestre IN ('1A', '1B')  
GROUP BY A.cod_dep
```

El lenguaje SQL.

SELECT [**ALL** | **DISTINCT**] $A_{1i}, A_{2j}, \dots, A_{nk}$ | *
FROM R_1, R_2, \dots, R_n
[WHERE *condición*]
[GROUP BY B_1, B_2, \dots, B_m]
[HAVING *condición*]

Evaluación:

- ✓ Se seleccionan n tuplas de las relaciones R_1, R_2, \dots, R_n que cumplan la condición de la cláusula **WHERE**.
- ✓ En el conjunto de tuplas seleccionadas se definen grupos basados en el valor de los atributos de agrupación: B_1, B_2, \dots, B_m .
- ✓ De los grupos definidos se seleccionan los que cumplen la condición de la cláusula **HAVING**.



Consultas sobre varias relaciones

El lenguaje SQL.

```
SELECT [ALL | DISTINCT]  $A_{1i}, \dots, A_{2j}, \dots, A_{nk} \mid ^*$   
FROM  $R_1, R_2, \dots, R_n$   
[WHERE condición]  
[GROUP BY  $B_1, B_2, \dots, B_m$ ]  
[HAVING condición]
```

Uso de varias relaciones en una SELECT:

- ✓ relaciones de la cláusula FROM: sólo se puede obtener datos de estas relaciones.
- ✓ relaciones en subconsultas de las cláusulas WHERE y HAVING: estas subconsultas sirven para expresar condiciones.

El lenguaje SQL.

Consultas sobre varias relaciones.

SELECT [ALL | DISTINCT] $A_{1i}, \dots, A_{2j}, \dots A_{nk} \mid ^*$
FROM $R_1, R_2, \dots R_n$
[WHERE *condición*]

condiciones de concatenación de
las relaciones $R_1, R_2, \dots R_n$

AND

condiciones de la consulta

la concatenación entre $R_1, R_2, \dots R_n$ se realiza
usualmente sobre los atributos de referencia
(claves ajenas) de las relaciones.

El lenguaje SQL.

Obtener la docencia de los profesores del DSIC: nombre del profesor, nombre de la asignatura impartida y grupos impartidos.

```
SELECT PX.nombre, AX.nombre, DX.gteo, DX.gpra
FROM Profesor PX, Docencia DX, Asignatura AX
WHERE PX.cod_pro = DX.cod_pro
        AND
        AX.cod_asg = DX.cod_asg
        AND
        PX.cod_dep = "DSIC"
```

← concatenación de
Profesor, Asignatura y
Docencia

← condiciones de la
consulta

El lenguaje SQL.

Concatenación de tablas.

SELECT [ALL | **DISTINCT**] $A_{1i}, \dots, A_{2j}, \dots, A_{nk} \mid ^*$
FROM R_1 , concatenación de tablas
[**WHERE** *condición*]
[**GROUP BY** B_1, B_2, \dots, B_m]
[**HAVING** *condición*]

- ✓ Producto cartesiano
- ✓ concatenación interna
- ✓ concatenación externa

El lenguaje SQL.

Producto cartesiano o CROSS JOIN

tabla1 CROSS JOIN *tabla2*



Equivalente a la inclusión en el componente FROM de las dos referencias de tablas separadas por comas:

FROM tabla1, tabla2

El lenguaje SQL.

Concatenación interna de tablas.

```
tabla1 [NATURAL] [INNER] JOIN tabla2  
[ON condición | USING ( $C_1, C_2, \dots, C_n$ ) ]
```



```
tabla1 NATURAL INNER JOIN tabla2
```

```
tabla1 INNER JOIN tabla2 USING ( $C_1, C_2, \dots, C_n$ )
```

```
tabla1 INNER JOIN tabla2 ON condición
```

El lenguaje SQL.

Concatenación interna de tablas: **NATURAL INNER JOIN**.

SELECT [ALL | DISTINCT] A_1, A_2, \dots, A_n | *
FROM *tabla1* **NATURAL INNER JOIN** *tabla2* (1)
[**WHERE** *condición*] (2)
[**GROUP BY** B_1, B_2, \dots, B_m] (3)
[**HAVING** *condición*] (4)

se concatenan las tuplas de *tabla1* y *tabla2* que tienen el mismo valor en los atributos del mismo nombre

tabla1 ⋈ *tabla2*

operador **Concatenación**
del Álgebra Relacional

El lenguaje SQL.

Concatenación interna de tablas: NATURAL INNER JOIN.

```
SELECT [ALL | DISTINCT] A1, A2,..., An | *  
FROM tabla1 NATURAL INNER JOIN tabla2  
[WHERE condición]  
[GROUP BY B1, B2,..., Bm]  
[HAVING condición]
```



si C_1, C_2, \dots, C_n son los atributos del mismo nombre de tabla1 y tabla2

```
SELECT [ALL | DISTINCT] A1, A2,..., An | *  
FROM tabla1, tabla2  
WHERE (tabla1. C1 =tabla2. C1) AND  
      (tabla1. C2 =tabla2. C2) AND  
      ... AND  
      (tabla1. Cn =tabla2. Cn) [AND condición]  
[GROUP BY B1, B2,..., Bm]  
[HAVING condición]
```

El lenguaje SQL.

```
SELECT PX.cod_pro, PX.nombre, COUNT(DX.cod_asg)
FROM Profesor PX, Docencia DX
WHERE PX.cod_pro = DX.cod_pro
GROUP BY cod_pro
```

concatenación de
Profesor y Docencia



```
SELECT cod_pro, nombre, COUNT (cod_asg)
FROM Profesor NATURAL INNER JOIN Docencia
GROUP BY cod_pro
```

El lenguaje SQL.

Concatenación interna de tablas: INNER JOIN USING

SELECT [ALL | **DISTINCT**] A_1, A_2, \dots, A_n | *

(1) **FROM** *tabla1* **INNER JOIN** *tabla2* **USING** (C_1, C_2, \dots, C_n)

(2) **WHERE** *condición*

(3) **GROUP BY** B_1, B_2, \dots, B_m

(4) **HAVING** *condición*

se concatenan las tuplas de *tabla1* y *tabla2* que tienen el mismo valor en los atributos comunes C_1, C_2, \dots, C_n

Es útil cuando no interesa que las relaciones se concatenen por todos los atributos del mismo nombre (NATURAL INNER JOIN).

El lenguaje SQL.

Concatenación interna de tablas: INNER JOIN USING.

```
SELECT [ALL | DISTINCT] A1, A2,..., An | *  
  FROM tabla1 INNER JOIN tabla2 USING (C1, C2,..., Cn )  
  [WHERE condición]  
  [GROUP BY B1, B2,..., Bm]  
  [HAVING condición]
```



```
SELECT [ALL | DISTINCT] A1, A2,..., An | *  
  FROM tabla1, tabla2  
  WHERE (tabla1.C1 =tabla2.C1) AND  
        (tabla1.C2 =tabla2.C2) AND  
        ... AND  
        (tabla1.Cn =tabla2.Cn) [AND condición]  
  [GROUP BY B1, B2,..., Bm]  
  [HAVING condición]
```

El lenguaje SQL.

```
SELECT cod_pro, Profesor.nombre, Departamento.nombre  
FROM Profesor, Departamento  
WHERE Profesor.cod_dep = Departamento.cod_dep
```

```
SELECT cod_pro, Profesor.nombre, Departamento.nombre  
FROM Profesor NATURAL INNER JOIN Departamento
```




```
SELECT cod_pro, Profesor.nombre, Departamento.nombre  
FROM Profesor INNER JOIN Docencia USING cod_dep
```


El lenguaje SQL.

Concatenación interna de tablas: INNER JOIN ON

```
SELECT [ALL | DISTINCT] A1, A2,..., An | *  
  FROM      tabla1 INNER JOIN tabla2 ON condición1 (1)  
  [WHERE condición2] (2)  
  [GROUP BY B1, B2,..., Bm] (3)  
  [HAVING condición] (4)
```



se concatenan las tuplas de tabla1 y tabla2 que cumplen condición1

Es útil cuando:

- ✓ interesa concatenar tuplas de tabla1 y tabla2 por condiciones distintas de la igualdad.
- ✓ los atributos por los que se desea concatenar no tienen el mismo nombre en ambas relaciones.

El lenguaje SQL.

Concatenación interna de tablas: INNER JOIN ON.

```
SELECT [ALL | DISTINCT] A1, A2, ..., An | *  
  FROM tabla1 INNER JOIN tabla2 ON condición1  
  [WHERE condición2]  
  [GROUP BY B1, B2, ..., Bm]  
  [HAVING condición]
```



```
SELECT [ALL | DISTINCT] A1, A2, ..., An | *  
  FROM tabla1, tabla2  
  WHERE (condición1) [AND condición2]  
  [GROUP BY B1, B2, ..., Bm]  
  [HAVING condición]
```

El lenguaje SQL.

Concatenación externa de tablas.

Obtener de cada departamento el número de profesores adscritos.

```
SELECT DX.cod_dep, COUNT(PX.cod_pro)
FROM Profesor PX, Departamento DX
WHERE PX.cod_dep = DX.cod_dep
GROUP BY DX.cod_dep
```

¡ los departamentos que no tienen profesores adscritos no salen en el resultado de la consulta !

El lenguaje SQL.

Concatenación externa de tablas.

Obtener de cada departamento el número de profesores adscritos.

```
SELECT Departamento.cod_dep, COUNT(cod_pro)
FROM (Profesor INNER JOIN Departamento USING (cod_dep) )
GROUP BY Departamento.cod_dep
```

¡ los departamentos que no tienen profesores adscritos no salen en el resultado de la consulta !

El lenguaje SQL.

Concatenación externa de tablas.

```
tabla1 [NATURAL]
      { LEFT [OUTER] |
        RIGHT [OUTER] |
        FULL [OUTER] } JOIN tabla2
      [ON condición | USING (C1, C2,..., Cn) ]
```



tabla1 [NATURAL] LEFT JOIN tabla2 [ON condición| USING (C₁, C₂,..., C_n)] tabla2

tabla1 [NATURAL] RIGHT JOIN tabla2 [ON condición| USING (C₁, C₂,..., C_n)] tabla2

tabla1 [NATURAL] FULL JOIN tabla2 [ON condición| USING (C₁, C₂,..., C_n)] tabla2

El lenguaje SQL.

SELECT [ALL | DISTINCT] A_1, A_2, \dots, A_n | *
FROM tabla1 [NATURAL] LEFT JOIN tabla2
[ON condición] USING (C_1, C_2, \dots, C_n)
[WHERE condición]
[GROUP BY B_1, B_2, \dots, B_m]
[HAVING condición]

(1)

(2)

(3)

(4)

{concatenación (NATURAL | ON | USING) de las tuplas de tabla1 y tabla2}

UNION

{tuplas de tabla1 que no pueden concatenarse con tuplas de tabla2 extendidas con valores nulos para los atributos de tabla2}

El lenguaje SQL.

Obtener de cada departamento el número de profesores adscritos.

```
SELECT Departamento.cod_dep, COUNT(cod_pro)
FROM Departamento LEFT JOIN Profesor ON (cod_dep)
GROUP BY Departamento.cod_dep
```


El lenguaje SQL.

cod_dep	Departamento.nombre	director	Departamento.teléfono	cod_pro	Profesor.nombre	Profesor.teléfono
DSIC	Sistemas Informáticos y Computación	V. Botti	3500	JCC	Juan Casamayor Ródenas	7796
DSIC	Sistemas Informáticos y Computación	V. Botti	3500	MAF	María Alpuente Frasnado	3560
DSIC	Sistemas Informáticos y Computación	V. Botti	3500	MCG	Matilde Celma Giménez	7756
DISCA	Ingeniería de Sistemas, Computadores y Automática	A. Crespo	5700	JBD	José Benlloch Dualde	5760
DISCA	Ingeniería de Sistemas, Computadores y Automática	A. Crespo	5700	DGT	Daniel Gil Tomás	5679
MAT	Matemática Aplicada	P. Pérez	6600	RFC	Robert Fuster i Capilla	6789
FIS	Física Aplicada	J. Linares	5200			
IDM	Idiomas	B. Montero	5300	CPG	Cristina Pérez Guillot	7439
EIO	Estadística e Investigación Operativa	L. Barceló	4900	NULO	NULO	NULO
OEM	Org. de Empresas, Economía Financ. y Contabilidad	M. Pérez	6800	JTM	José Torralba Martínez	4590
OEM	Org. de Empresas, Economía Financ. y Contabilidad	M. Pérez	6800	IGP	Ignacio Gil Pechuán	3423

DSIC 3

DISCA 2

MAT 1

FIS 1

IDM 1

EIO 0

OEM 2

Departamento LEFT JOIN Profesor ON (cod_dep)

El lenguaje SQL.

SELECT [ALL | DISTINCT] A_1, A_2, \dots, A_n | *
FROM tabla1 [NATURAL] RIGHT JOIN tabla2
[ON condición] USING (C_1, C_2, \dots, C_n)
[WHERE condición]
[GROUP BY B_1, B_2, \dots, B_m]
[HAVING condición]

(1)

(2)

(3)

(4)

{concatenación (NATURAL | ON | USING) de las tuplas de tabla1 y tabla2}

UNION

{tuplas de tabla2 que no pueden concatenarse con tuplas de tabla1 extendidas con valores nulos para los atributos de tabla1}

El lenguaje SQL.

Obtener de cada departamento el número de profesores adscritos.

```
SELECT Departamento.cod_dep, COUNT(cod_pro)
FROM Departamento LEFT JOIN Profesor ON (cod_dep)
GROUP BY Departamento.cod_dep
```



```
SELECT Departamento.cod_dep, COUNT(cod_pro)
FROM Profesor RIGHT JOIN Departamento ON (cod_dep)
GROUP BY Departamento.cod_dep
```

El lenguaje SQL.

SELECT [ALL | DISTINCT] A_1, A_2, \dots, A_n | *
FROM tabla1 [NATURAL] FULL JOIN tabla2 (1)
[ON condición] USING (C_1, C_2, \dots, C_n)
[WHERE condición] (2)
[GROUP BY B_1, B_2, \dots, B_m] (3)
[HAVING condición] (4)

{concatenación (NATURAL | ON | USING) de las tuplas de tabla1 y tabla2}

UNION

{tuplas de tabla2 que no pueden concatenarse con tuplas de tabla1 extendidas con valores nulos para los atributos de tabla1}

UNION

{tuplas de tabla1 que no pueden concatenarse con tuplas de tabla2 extendidas con valores nulos para los atributos de tabla2}

El lenguaje SQL.

Concatenación de tablas en ORACLE*.

Sintaxis del operador de SQL **LEFT [OUTER] JOIN** en ORACLE8:

```
SELECT [ALL | DISTINCT] A1, A2,..., An | *  
  FROM tabla1 LEFT JOIN tabla2 ON tabla1.Ai Op tabla2.Aj  
  .....
```

SQL



```
SELECT [ALL | DISTINCT] A1, A2,..., An | *  
  FROM tabla1, tabla2  
  WHERE tabla1.Ai Op tabla2.Aj (+)  
  .....
```

ORACLE

* En ORACLE sólo existen operadores para la concatenación externa.

El lenguaje SQL.

Concatenación de tablas en ORACLE.

Sintaxis del operador de SQL **RIGHT [OUTER] JOIN** en ORACLE8:

```
SELECT [ALL | DISTINCT] A1, A2,..., An | *  
      FROM tabla1 RIGHT JOIN tabla2 ON tabla1.Ai Op tabla2.Aj  
      ..... SQL
```



```
SELECT [ALL | DISTINCT] A1, A2,..., An | *  
      FROM tabla1, tabla2  
      WHERE tabla1.Ai (+) Op tabla2.Aj  
      ..... ORACLE
```

Si la condición de combinación del JOIN externo se define sobre varias columnas, el signo (+) deberá aparecer en cada columna que participa en la combinación en el mismo sentido: si es RIGHT en la columna de la tabla t1 y si es LEFT en la columna de la tabla t2.

El lenguaje SQL.

UNION JOIN

Obtiene una tabla que contiene :

- todas las columnas de tabla1 y tabla2
- todas las tuplas de tabla1 con nulos en las columnas de la tabla2
- y viceversa (todas las tuplas de tabla2 con nulos en las columnas de la tabla1)

El lenguaje SQL.

Operaciones conjuntistas en SQL.

- ✓ UNION (unión)
- ✓ INTERSECT (intersección)
- ✓ EXCEPT (diferencia)

Se definen entre relaciones compatibles (del mismo esquema).

El lenguaje SQL.



UNION



```
consulta1 UNION [ALL] consulta2
```

INTERSECT



```
consulta1 INTERSECT consulta2
```

EXCEPT



```
consulta1 EXCEPT consulta2
```



El lenguaje SQL.

UNION

Obtener el nombre de todo el personal (profesores y directores de departamento).

```
SELECT nombre  
FROM Director  
UNION  
SELECT nombre  
FROM Profesor
```

El lenguaje SQL.

UNION

Obtener de cada departamento el número de profesores adscritos.

```
SELECT Departamento.cod_dep, COUNT(cod_pro)
FROM Departamento LEFT JOIN Profesor ON (cod_dep)
GROUP BY Departamento.cod_dep
```



```
SELECT D.cod_dep, COUNT(P.cod_pro)
FROM Departamento D, Profesor P
WHERE D.cod_dep = P.cod_dep
GROUP BY D.cod_dep
UNION
SELECT D.cod_dep, 0
FROM Departamento D
WHERE D.cod_dep NOT IN (SELECT cod_dep FROM Profesor)
```

El lenguaje SQL.

INTERSECT

Obtener los departamentos que tienen adscritas asignaturas y profesores.

```
SELECT DISTINCT cod_dep  
FROM Profesor  
INTERSECT  
SELECT DISTINCT cod_dep  
FROM Asignatura
```

El lenguaje SQL.

EXCEPT

Obtener los departamentos que no tienen adscritas asignaturas.

```
SELECT cod_dep  
FROM Departamento  
EXCEPT  
SELECT DISTINCT cod_dep  
FROM Asignatura
```

En ORACLE, el operador EXCEPT se denomina MINUS.



COMBINACIONES DE TABLAS RESUMEN

- ▶ Existen, en definitiva, varias formas de combinar dos tablas en el lenguaje SQL:
 - Incluir varias tablas en la cláusula from.
 - Uso de subconsultas en las condiciones de las cláusulas where o having.
 - **Combinaciones conjuntistas de tablas:** utilizan para combinar las tablas operadores de la teoría de conjuntos.
 - **Concatenaciones de tablas:** combinan dos tablas utilizando diferentes formas variantes del operador concatenación del Álgebra Relacional.

ÁLGEBRA RELACIONAL -- SQL

A.R.

SQL ESTÁNDAR

SQL de ORACLE'8

• \cup	➡	• UNION	➡	• UNION
• $-$	➡	• EXCEPT	➡	• MINUS
• \cap	➡	• INTERSECT	➡	• INTERSECT
• \times	➡	• CROSS JOIN	➡	• (no hace falta, poner una coma)
• $ >< $	➡	• NATURAL JOIN	➡	• (no está, bastan '='s en el WHERE)

Otros

• \cup (dups.)	➡	• UNION ALL	➡	• UNION ALL
• \times	➡	• Left/Right/Full JOIN	➡	• WHERE TX.a1(+) = TY.a2 (eq. right join)
		• UNION JOIN	➡	• WHERE TX.a1 = TY.a2 (+) (eq. left join)

El lenguaje SQL.

Lenguaje SQL:

- ✓ manipulación de datos (consulta y actualización):
 - **SELECT** (consulta)
 - **INSERT** (inserción de tuplas)
 - **DELETE** (borrado de tuplas)
 - **UPDATE** (modificación de tuplas)
- ✓ definición de datos (definición del esquema)

El lenguaje SQL.

Operación de inserción de tuplas.

Inserción simple

```
INSERT INTO tabla [(columna1, columna2, ...columnak)]  
    {DEFAULT VALUES | VALUES (v1, v2, ..., vk) }
```

Inserción múltiple

```
INSERT INTO tabla [(columna1, columna2, ...columnak)]  
    consulta
```

- Si no se incluye la lista de columnas se deberán insertar filas completas de *tabla*.
- Si se incluye la opción default values se insertará una única fila en la tabla con los valores por defecto apropiados en cada columna (según la definición de *tabla*).

El lenguaje SQL.

Operación de inserción de tuplas.

Insertar un nuevo profesor:

INSERT INTO Profesor

VALUES ('EVL', 'Enrique Vidal López', 73333, 'DSIC')



{(cod_pro, 'EVL'), (nombre, 'Enrique Vidal López'), (teléfono, 73333), (cod_dep, 'DSIC')}

INSERT INTO Profesor (cod_pro, nombre, cod_dep)

VALUES ('EVL', 'Enrique Vidal López', 'DSIC')



{(cod_pro, 'EVL'), (nombre, 'Enrique Vidal López'), (teléfono, ?), (cod_dep, 'DSIC')}

El lenguaje SQL.

Operación de inserción de tuplas.

Insertar en la tabla Profesor los profesores de una tabla del mismo esquema Profesores_Alcoy :

Inserción múltiple

```
INSERT INTO Profesor
```

```
SELECT cod_pro, nombre, teléfono, cod_dep
```

```
FROM Profesores_Alcoy
```

El lenguaje SQL.

Operación de borrado de tuplas.

DELETE FROM tabla [**WHERE** *condición*]

Borrar el profesor de código 'EVL':

Borrado simple

DELETE FROM Profesor **WHERE** cod_pro = 'EVL'

Borrar los profesores del DSIC:

Borrado múltiple

DELETE FROM Profesor **WHERE** cod_dep = 'DSIC'

El lenguaje SQL.

Operación de actualización de tuplas.

UPDATE tabla

SET $columna_1 = \{DEFAULT | NULL | expresión1\},$

$columna_2 = \{DEFAULT | NULL | expresión2\},$

.....

[WHERE condición]

El lenguaje SQL.

Operación de actualización de tuplas.

Adscribir el profesor de código MCG al DISCA.

```
UPDATE Profesor  
SET cod_dep = 'DISCA'  
WHERE cod_pro='MCG'
```

Actualización
simple

Adscribir todos los profesores del departamento DSIC al DISCA.

```
UPDATE Profesor  
SET cod_dep = 'DISCA'  
WHERE cod_dep='DSIC'
```

Actualización
múltiple

El lenguaje SQL.

El lenguaje de manipulación de datos de ORACLE:

Las principales diferencias respecto al estándar SQL son:

- ✓ no incluye operadores de concatenación interna
- ✓ la concatenación externa tiene una notación específica
- ✓ en ORACLE el operador EXCEPT se denomina MINUS.