

4.3 SQL-92

LDD: definición de vistas

- Una vista es una tabla derivada de otras tablas
- Es una **tabla virtual**: no necesariamente existe en forma física

- Sentencia de definición o **creación** de una vista

```
CREATE VIEW <nombre_vista> [ (<lista_nombres_columnas>) ]  
AS <consulta_de_definición>
```

- La consulta de definición es una SELECT que...
 - determina el **contenido de la vista**
 - contiene las **tablas base**: **tablas** o vistas de las que se deriva la vista (también llamadas tablas de definición)

4.3 SQL-92

Dos tablas más del esquema de BD “Empresa”

EMPLEADO

| nombre | apellido | <u>nss</u> | dni | añonacim | direccion | ciudad | salario | nssjefe | dep |
|------------|----------|------------|-----|----------|-------------|---------|---------|---------|------|
| JONÁS | SOLANO | 123 | 11A | 1945 | C/PEZ, 10 | MURCIA | 1100 | 111 | D1 |
| RIGOBERTA | CALavera | 321 | 21C | 1974 | C/BOJ, 2 | YECLA | 900 | 333 | D3 |
| EUSEBIO | MULETAS | 222 | 22B | 1969 | C/RIF, 6 | TOTANA | 2100 | 123 | D2 |
| MACARENO | SOSO | 111 | 23D | 1944 | C/MAR, 4 | JUMILLA | 1100 | | D1 |
| CASIANA | FABERGÉ | 333 | 33B | 1943 | C/SOL, 8 | MURCIA | 920 | 123 | D3 |
| FILOMENA | RASCAS | 234 | 34E | 1970 | C/NUEZ, 3 | MURCIA | 1100 | 111 | D1 |
| GUMERSINDA | MIMOS | 543 | 45F | 1980 | C/QUINTO, 5 | PINOSO | 850 | NULL | NULL |

PROYECTO

| titulo | <u>codproy</u> | lugar | dep |
|------------|----------------|--------|-----|
| PROYECTO X | X | MURCIA | D2 |
| PROYECTO Y | Y | YECLA | D3 |
| PROYECTO Z | Z | TOTANA | D4 |

DEDICACION

| <u>empleado</u> | <u>proyecto</u> | horas |
|-----------------|-----------------|-------|
| 123 | X | 32.5 |
| 543 | Z | 40.0 |
| 321 | Y | 7.5 |
| 111 | X | 20.0 |
| 333 | Y | 10.0 |
| 222 | Z | 10.0 |
| 543 | Y | 20.0 |

4.3 SQL-92

LDD: definición de vistas

- Ejemplo

```
CREATE VIEW Empleado_en_Proyecto  
AS SELECT nombre, apellido, titulo, horas  
FROM Empleado, Dedicacion, Proyecto  
WHERE nss = empleado AND proyecto = codproy ;
```

```
SELECT *  
FROM Empleado_en_Proyecto;
```

| nombre | apellido | titulo | horas |
|------------|----------|------------|-------|
| JONÁS | SOLANO | PROYECTO X | 32.5 |
| GUMERSINDA | MIMOS | PROYECTO Z | 40.0 |
| RIGOBERTA | CALAVERA | PROYECTO Y | 7.5 |
| MACARENO | SOSO | PROYECTO X | 20.0 |
| CASIANA | FABERGÉ | PROYECTO Y | 10.0 |
| EUSEBIO | MULETAS | PROYECTO Z | 10.0 |
| GUMERSINDA | MIMOS | PROYECTO Y | 20.0 |

4.3 SQL-92

LDD: definición de vistas

- Las vistas pueden utilizarse como mecanismo de...
 - **Simplificación de consultas**
 - Consulta muy **frecuente**, a la que se le da un nombre
 - Consulta muy **compleja** (combina *joins*, agregados, *online views*, etc.)
 - **Seguridad**
 - Ocultación de ciertas columnas, del nombre de una tabla
 - Ocultación del propietario de la tabla (esquema)
 - **Adaptación** de la información a las necesidades de cada usuario o grupo de usuarios
 - Nuevos nombres para las columnas
 - Cálculos aritméticos, o de agregados, *joins*, etc.

4.3 SQL-92

LDD: definición de vistas

- Característica fundamental de las vistas

Actualización Permanente

- El responsable de esta característica es el SGBD

► La vista no se crea cuando se define, sino cuando se consulta

► Una vista no “contiene información”,
sino que “deja ver información”
que está almacenada en sus tablas base

4.3 SQL-92

LDD: definición de vistas

- Por defecto, la vista 'hereda' los nombres de las columnas seleccionadas desde las tablas base

```
CREATE VIEW Familiar_Empleado
AS SELECT E.nombre, F.nombre, parentesco
FROM Empleado E JOIN Familiar F ON nss = nssemp;
```

←nombres de columna que hereda la vista

- Definición de **nuevos nombres** para columnas de la vista

```
CREATE VIEW Familiar_Empleado (empleado, familiar, parentesco)
AS SELECT E.nombre, F.nombre, parentesco
FROM Empleado E JOIN Familiar F ON nss = nssemp;
```

- Obligatorio cuando alguna columna es el resultado de una operación aritmética o una función de agregados

```
CREATE VIEW Info_Depto (nombre_dep, num_emps, sal_total)
AS SELECT D.nombre, COUNT(*), SUM(salario)
FROM Departamento D, Empleado
WHERE coddep = dep
GROUP BY D.nombre;
```

4.3 SQL-92

LDD: definición de vistas

- Es posible seleccionar todas las columnas...

```
CREATE VIEW Veterano
AS SELECT *
   FROM Empleado
   WHERE año nacim < 1970;
```

```
CREATE VIEW Catalogo
AS SELECT *
   FROM Producto
   WHERE linea IN ('JUVENIL', 'MINIMALISTA', 'ETNICA');
```

4.3 SQL-92

LDD: definición de vistas

- ❑ En **SQL Oracle** la sentencia `CREATE VIEW` tiene la cláusula **OR REPLACE**, que permite la redefinición (recompilación) de la vista
- Esto evita tener que eliminarla y volver a crearla para cambiar algo en su definición
- Si se crea una vista cuya consulta de definición contiene un `*` (todas las columnas de las tablas base), y después se añade una o varias columnas a alguna de dichas tablas, la vista NO mostrará esas nuevas columnas automáticamente: es necesario recompilarla (o eliminarla y volver a crearla)
 - Si se ha añadido la columna 'categoria' a PRODUCTO y en el catálogo se desea sustituir la línea ETNICA por RUSTICA, hay que ejecutar esto:

```
CREATE OR REPLACE VIEW Catalogo
AS SELECT *
   FROM Producto
  WHERE linea IN ('JUVENIL', 'MINIMALISTA', 'RUSTICA');
```


4.3 SQL-92

LDD: **consulta** a través de vistas

- Las vistas **no** tienen ninguna **limitación** en operaciones de **consulta**
- El usuario **no** distingue si el elemento al que accede es una tabla base o una vista

* Nombres de los empleados y de sus hijos/as

```
SELECT empleado, familiar  
FROM Familiar_Empleado WHERE parentesco = 'HIJO' ;
```

* Datos del departamento 'Investigación'

```
SELECT * FROM Info_Depto WHERE  
nombre_dep= 'INVESTIGACIÓN' ;
```

* Nombres y apellidos de los empleados que trabajan en el proyecto 'PROYECTO X'

```
SELECT nombre, apellido, titulo  
FROM Empleado_en_Proyecto  
WHERE titulo= 'PROYECTO X' ;
```

4.3 SQL-92

LDD: consulta a través de vistas

- El SGBD **traduce** cualquier sentencia SQL sobre la vista a una expresión equivalente sobre sus tablas base: reemplaza el nombre de la vista por su consulta de definición, la combina con la sentencia en la que se usa, y la ejecuta

```
CREATE VIEW Veterano AS
```

```
SELECT nombre, dni, nss, año nacim, dep FROM Empleado
```

```
WHERE año nacim < 1970;
```

| Sentencia de usuario | Traducción |
|---|---|
| <pre>SELECT * FROM VETERANO WHERE ciudad='MURCIA';</pre> | <pre>SELECT nombre, dni, nss, año nacim, dep FROM EMPLEADO WHERE año nacim<1970 AND ciudad='MURCIA';</pre> |
| <pre>SELECT nombre, nssdire FROM VETERANO JOIN DEPARTAMENTO ON dep=coddep;</pre> | <pre>SELECT nombre, nssdire FROM EMPLEADO JOIN DEPARTAMENTO ON dep=coddep WHERE año nacim<1970;</pre> |
| <pre>SELECT nombre FROM VETERANO WHERE nss IN (SELECT nssemp FROM FAMILIAR WHERE parentesco <> 'HIJO');</pre> | <pre>SELECT nombre FROM EMPLEADO WHERE año nacim<1970 AND nss IN (SELECT nssemp FROM FAMILIAR WHERE parentesco <> 'HIJO');</pre> |

4.3 SQL-92

LDD: **modificación** a través de vistas

- El SGBD **traduce** (si puede) cualquier sentencia INSERT, UPDATE y DELETE sobre la vista a una expresión equivalente sobre sus tablas base
- Puesto que **una vista no contiene datos**, con un INSERT, UPDATE o DELETE sobre una vista **no** se modifica “el contenido de la vista”...
 - ▶ Hay que **modificar los datos almacenados en las tablas base**, para que la siguiente vez que se consulte la vista, los datos se vean así cambiados

| Sentencia de usuario | Traducción |
|--|---|
| <code>INSERT INTO VETERANO VALUES ('EVA', '12E', '234',1947, 'D4');</code> | <code>INSERT INTO EMPLEADO (nombre, dni, nss, año nacim, dep) VALUES ('EVA', '12E', '234',1947, 'D4');</code> |
| <code>UPDATE VETERANO SET dep= 'D1' WHERE dep='D2';</code> | <code>UPDATE EMPLEADO SET dep='D1' WHERE año nacim<1970 AND dep='D2';</code> |
| <code>DELETE FROM VETERANO WHERE dni = '123E';</code> | <code>DELETE FROM EMPLEADO WHERE año nacim<1970 AND dni='123E';</code> |

4.3 SQL-92

LDD: modificación a través de vistas

- La **actualización** de datos a través de vistas tiene **limitaciones**
 - Por un lado, algunas actualizaciones **carecen de sentido**
 - Por ejemplo, sobre columnas definidas mediante una **función de agregados**

```
UPDATE Info_Depto  
SET sal_total = 100000  
WHERE nombre_dep= 'INVESTIGACIÓN' ;
```

 - ▶ `sal_total` se calcula: es la *suma de salarios individuales* de los empleados y muchas actualizaciones de las tablas base satisfarían esta actualización
 - Por otro lado, actualizar a través de una **vista definida sobre varias tablas base** suele dar problemas, pues puede haber **ambigüedad**
 - Una modificación puede traducirse a dos o más actualizaciones distintas de las tablas base de la vista y el SGBD no tiene manera de saber cuál es correcta y cuál es la más adecuada

4.3 SQL-92

LDD: modificación a través de vistas

- Recordemos la definición de la vista EMPLEADO_EN_PROYECTO

```
CREATE VIEW Empleado_en_Proyecto
AS SELECT nombre, apellido, titulo, horas
FROM Empleado, Dedicacion, Proyecto
WHERE nss = empleado AND proyecto = codproy;
```

- Lo que muestra la vista, antes de intentar la modificación:

| nombre | apellido | titulo | horas |
|------------|----------|------------|-------|
| JONÁS | SOLANO | PROYECTO X | 32.5 |
| GUMERSINDA | MIMOS | PROYECTO Z | 40.0 |
| RIGOBERTA | CALAVERA | PROYECTO Y | 7.5 |
| MACARENO | SOSO | PROYECTO X | 20.0 |
| CASIANA | FABERGÉ | PROYECTO Y | 10.0 |
| EUSEBIO | MULETAS | PROYECTO Z | 10.0 |
| GUMERSINDA | MIMOS | PROYECTO Y | 20.0 |

- Veamos las DOS actualizaciones de las tablas base como traducción de...

```
UPDATE Empleado_en_Proyecto
SET titulo='PROYECTO Z'
WHERE nombre='JONÁS' AND apellido='SOLANO' AND titulo='PROYECTO X';
```

4.3 SQL-92

LDD: modificación a través de vistas

❑ Actualización de las tablas base ①

UPDATE Dedicacion

SET proyecto=(SELECT codproy FROM Proyecto
WHERE titulo='PROYECTO Z')

WHERE empleado=(SELECT nss FROM Empleado
WHERE nombre='JONÁS' AND apellido='SOLANO')
AND proyecto=(SELECT codproy FROM Proyecto
WHERE titulo = 'PROYECTO X');

- ▶ 😊 Modifica los vínculos en DEDICACION: cada fila que relacionaba la fila de 'JONÁS SOLANO' en EMPLEADO con 'PROYECTO X' en PROYECTO, pasa a relacionar ese empleado con la fila 'PROYECTO Z' de PROYECTO

❑ Actualización de las tablas base ②

UPDATE Proyecto SET titulo='PROYECTO Z'
WHERE titulo='PROYECTO X';

- ▶ 😞 Produce igual efecto que ① pero **modifica titulo** en PROYECTO: al calcular la vista, mostrará 'PROYECTO Z' para todos los empleados que antes aparecían junto con 'PROYECTO X'

4.3 SQL-92

LDD: modificación a través de vistas

- Así que **no se garantiza que “toda vista sea actualizable”**
 - Es decir, que permita cambios a través de ella
- Una **vista** sería **actualizable** si implicara una **única actualización posible** de las **tablas base**
- En general...
 - Una vista con **una sola tabla base**
 - **SÍ** es **actualizable** si sus columnas **contienen la clave primaria** u otra clave **candidata** de la **tabla base**
 - Pues se establece una correspondencia entre cada fila de la vista y una única fila de la tabla base
 - Una vista definida sobre **varias tablas** mediante reuniones
 - **NO** es **actualizable**
 - Una vista definida mediante **agrupación y funciones agregadas**
 - **NO** es **actualizable**

4.3 SQL-92

LDD: modificación a través de vistas

- **Opción de verificación de vistas**

```
CREATE VIEW Precario
  AS SELECT nombre,apellido,nss,dni,salario,dep
     FROM Empleado WHERE salario < 950 ;
```

*¿Qué pasaría al ejecutar estas sentencias?

```
INSERT INTO Precario
  VALUES ('Dimas', 'Pi', '444', '121D', 1025, 'D1');
```

```
UPDATE Precario
  SET salario = 985
  WHERE dni= '21C'; -- antes cobraba 900€
```


4.3 SQL-92

LDD: modificación a través de vistas

- Cláusula **WITH CHECK OPTION**

- Se debe incluir en la definición de toda **vista actualizable** que se vaya a utilizar para la modificación de datos
- Indica al SGBD que **debe comprobar cada INSERT y UPDATE sobre la vista**, y **rechazarlo** si su realización implicara que la fila nueva o modificada no cumpliera la condición de definición de la vista

```
CREATE VIEW Precario
```

```
AS SELECT nombre, apellido, nss, dni, salario, dep  
FROM Empleado WHERE salario < 950  
WITH CHECK OPTION ;
```

4.3 SQL-92

LDD: implementación de vistas

1. Estrategia de **actualización de consultas de definición**

- Cada **consulta** sobre la **vista** se traduce a una **consulta** sobre las **tablas base**
- La vista se **rellena de filas** a partir de la ejecución de la **consulta**
- ☹ **Poco eficiente** cuando la <consulta_de_definición> es **compleja**, con tiempo de ejecución apreciable, y se aplican **muchas consultas** sobre la vista **en poco tiempo**

2. Estrategia de **materialización de vistas**

- **1ª consulta** sobre la vista ⇒ **creación de tabla temporal física**
- Se conserva la tabla para posteriores consultas sobre la vista
 - Necesaria estrategia para actualización incremental de la tabla temporal tras cualquier modificación sobre las tablas base
- ⇒ **actualización permanente**
- Si no se hace referencia a la vista tras un tiempo, el sistema la eliminará (y la recalculará en una consulta futura)

4.3 SQL-92

LDD: **eliminación** de vistas

- Eliminación de una vista

Orden **DROP VIEW**

- Destruye una tabla derivada, junto con su definición en el INFORMATION_SCHEMA del catálogo

DROP VIEW <nombre_vista> ;

4.3 SQL-92

LDD: índices

- En el modelo relacional se indica que las filas no están ordenadas.
- Pero ¿cómo podemos entonces buscar de manera eficiente un elemento en una tabla?
 - Eficiente: significa leer menos bloques (y registros) de discos.

4.3 SQL-92

LDD: índices

- ¿Cuándo nació el actor con **codA=77**?
- ¿Qué actores están representados por la **agencia Actors**?
- **Ojo:** los datos se almacenan en cinco bloques de disco.

Fichero **ACTOR**

| Bloque | codA | nombre | agencia | fechaNacim |
|--------|------|-----------------|-----------|------------|
| B1 | 88 | Fele Martínez | Glam | 22/02/75 |
| | 11 | Najwa Nimri | Actors | 14/02/72 |
| B2 | 33 | Nancho Novo | Rol | 17/09/78 |
| | 22 | Santiago Segura | Amiguetes | 17/07/65 |
| B3 | 77 | Luis Tosar | BCN | 13/10/71 |
| | 55 | Candela Peña | Actors | 14/07/73 |
| B4 | 00 | Maribel Verdú | Glam | 02/10/70 |
| | 44 | Penélope Cruz | BCN | 28/04/74 |
| B5 | 99 | Javier Bardem | BCN | 01/03/69 |
| | | | | |

4.3 SQL-92

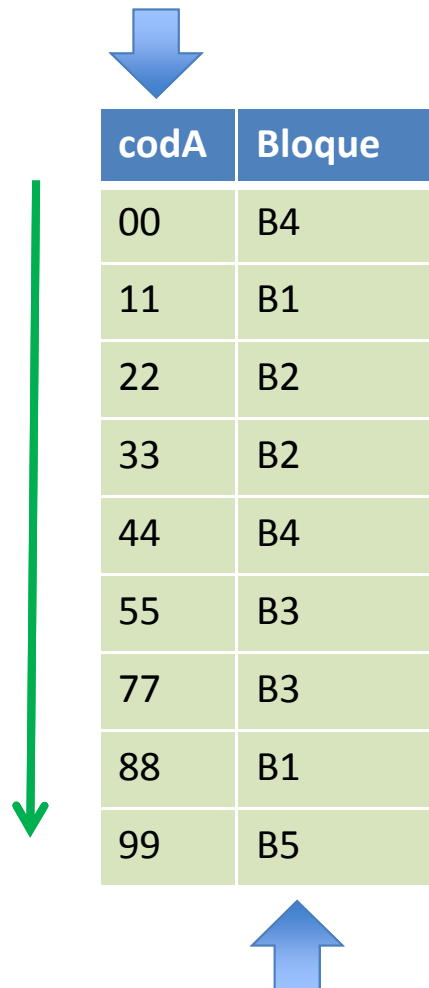
LDD: índices

- Un índice es una estructura de datos auxiliar que contiene entradas con dos campos, para almacenar...
 - Un **valor** de los almacenados en el atributo indexado
 - Un puntero al **bloque** que lo contiene
- Las entradas están **ordenadas** según el valor del campo indexado
 - De esta forma es posible realizar **búsquedas binarias** sobre el índice.
- Es posible tener índices sobre más de un campo de una tabla
- Es similar a un **índice en un libro**

4.3 SQL-92

LDD: índices

Ordenado por codA



| codA | Bloque |
|------|--------|
| 00 | B4 |
| 11 | B1 |
| 22 | B2 |
| 33 | B2 |
| 44 | B4 |
| 55 | B3 |
| 77 | B3 |
| 88 | B1 |
| 99 | B5 |

El índice cabe en un **único** bloque

Fichero **ACTOR**

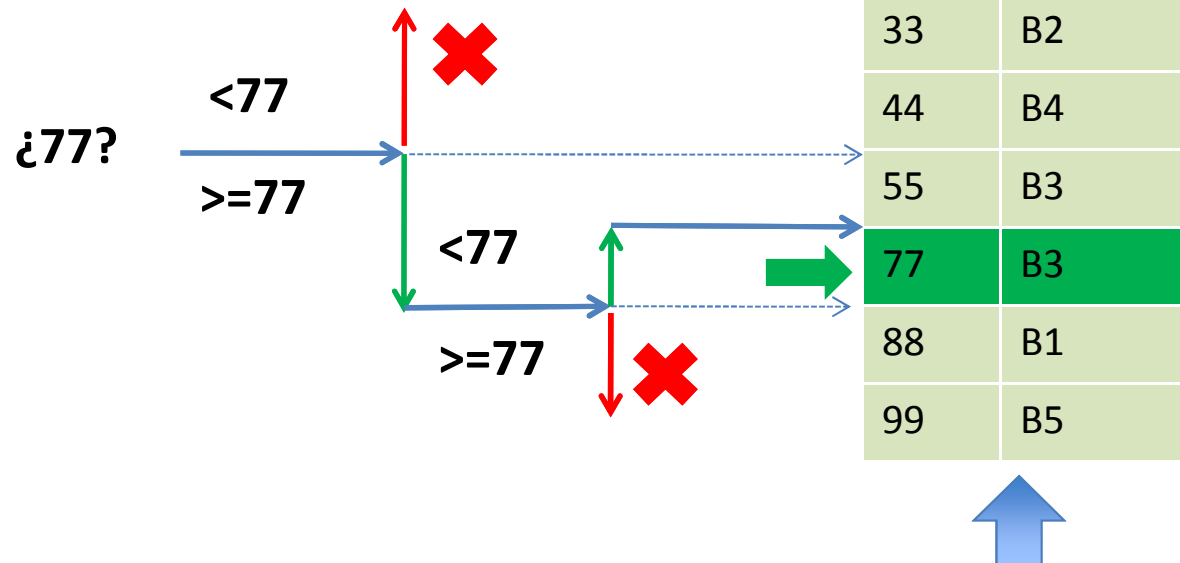
| Bloque | codA | nombre | agencia | fechaNacim |
|--------|------|-----------------|-----------|------------|
| B1 | 88 | Fele Martínez | Glam | 22/02/75 |
| | 11 | Najwa Nimri | Actors | 14/02/72 |
| B2 | 33 | Nancho Novo | Rol | 17/09/78 |
| | 22 | Santiago Segura | Amiguetes | 17/07/65 |
| B3 | 77 | Luis Tosar | BCN | 13/10/71 |
| | 55 | Candela Peña | Actors | 14/07/73 |
| B4 | 00 | Maribel Verdú | Glam | 02/10/70 |
| | 44 | Penélope Cruz | BCN | 28/04/74 |
| B5 | 99 | Javier Bardem | BCN | 01/03/69 |
| | | | | |

4.3 SQL-92

LDD: índices

- ¿Cuándo nació el actor con **codA=77**?

Búsqueda binaria



El índice cabe en un **único** bloque

Fichero **ACTOR**

| Bloque | codA | fechaNacim |
|--------|------|------------|
| B1 | 88 | 22/02/75 |
| | 11 | 14/02/72 |
| B2 | 33 | 17/09/78 |
| | 22 | 17/07/65 |
| B3 | 77 | 13/10/71 |
| | 55 | 14/07/73 |
| B4 | 00 | 02/10/70 |
| | 44 | 28/04/74 |
| B5 | 99 | 01/03/69 |
| | | |

4.3 SQL-92

LDD: índices

- Los índices son **recomendables** cuando
 - Un campo se usa frecuentemente...
 - en **condiciones** de restricción / selección
 - en operaciones de **reunión**
 - Una tabla ...
 - Es grande
 - Las consultas sobre ella recuperan un bajo porcentaje de los datos (menos del 20%)
- Así, se debe crear un índice sobre toda **clave** primaria (PRIMARY KEY) y clave alternativa (UNIQUE)
 - ❑ **Oracle** ya lo hace automáticamente
- Pero se puede crear índices sobre **columnas no clave**

4.3 SQL-92

LDD: índices

- ¿Cuáles son los actores de la agencia Actors?

Fichero **ACTOR**

Varios punteros
por entrada

| agencia | Bloque |
|-----------|------------|
| Actors | B1, B3 |
| Amiguetes | B2 |
| BCN | B3, B4, B5 |
| Glam | B1, B4 |
| Rol | B2 |



Ordenado por agencia
El índice aún cabe en un **único**
bloque

| Bloque | codA | nombre | agencia | fechaNacim |
|--------|------|-----------------|-----------|------------|
| B1 | 88 | Fele Martínez | Glam | 22/02/75 |
| | 11 | Najwa Nimri | Actors | 14/02/72 |
| B2 | 33 | Nancho Novo | Rol | 17/09/78 |
| | 22 | Santiago Segura | Amiguetes | 17/07/65 |
| B3 | 77 | Luis Tosar | BCN | 13/10/71 |
| | 55 | Candela Peña | Actors | 14/07/73 |
| B4 | 00 | Maribel Verdú | Glam | 02/10/70 |
| | 44 | Penélope Cruz | BCN | 28/04/74 |
| B5 | 99 | Javier Bardem | BCN | 01/03/69 |
| | | | | |

4.3 SQL-92

LDD: índices en Oracle

- Sentencia LDD para **crear** índices

```
CREATE INDEX Idx_Genero_Pelicula ON PELICULA(genero);
```

- Especificación del **orden** ascendente o descendente de las entradas

```
CREATE INDEX Idx_Año_Pelicula ON PELICULA(año DESC);
```

- Índice **compuesto**

```
CREATE INDEX Idx_Alumno ON ALUMNO(apell1,apell2,nombre);
```

- El índice incrementa la velocidad de consultas que acceden por un **prefijo**...
 - apell1
 - apell1 y apell2
 - apell1 y apell2 y nombre
- Pero el SGBD **no usará el índice** para consultas que acceden por...
 - apell2
 - nombre
 - apell2 y nombre

👁 En el WHERE pueden aparecer
en cualquier orden

4.3 SQL-92

LDD: índices en Oracle

- Sentencia LDD para destruir índices
DROP INDEX Idx_Genero_Pelicula ;
- Conviene eliminar un índice cuando ya no se espera realizar consultas basadas en el campo de indexación
- También si se detecta su escasa utilización en la ejecución de las consultas
- Desaparece el coste de mantenimiento del índice
- Se recupera el espacio de almacenamiento ocupado por el índice