

Realización de consultas

BDDA

UT4

Laurent/Rubén



Introducción

UT3 Diseñar tablas

UT2 Crear las tablas en Oracle

UT4 Consultar y operar con los datos que hay en la tabla.

- Select
- Operadores
- Consultas calculadas
- Funciones
- Agrupaciones
- Consultar de varias tablas
- Subconsultas, select dentro de select



Sentencia Select



POLITÉCNICO
ESTELLA

Select

Obligatorio usar SELECT y FROM. Debemos elegir qué seleccionar y de dónde.

SELECT * FROM CICLISTA

** significa elegir todos los campos de la tabla.*



Select

```
SELECT nombre FROM CICLISTA
```

Nos devuelve todos los atributos nombre de la tabla CICLISTA

```
SELECT * FROM CICLISTA WHERE nombre="Enric"
```

WHERE permite concretar qué estamos buscando.

La consulta selecciona todos los campos de ciclista donde el nombre es Enric

Select

SELECT nombre, apellido FROM CICLISTA

Separados por comas podemos elegir más de un atributo, columna, a devolver

SELECT CICLISTA.Nombre AS Nombre del ciclista, CICLISTA.Apellido AS Apellido del ciclista
FROM CICLISTA

WHERE peso < 60;

Podemos indicar el nombre de la tabla y nombre del atributo. TABLA.atributo para seleccionar esas columnas. En WHERE podemos usar comparadores =, <, >, >=, <=, <> (diferente)

También, usar AS para cambiar el nombre de la tabla que aparecerá como resultado.

| Nombre del ciclista | Apellido del ciclista |
|---------------------|-----------------------|
| Enric | Mas |

Select

```
SELECT C.Nombre AS Nombre del ciclista, C.Apellido AS Apellido del ciclista  
FROM CICLISTA AS C  
WHERE peso < 60;
```

Podemos dar un nombre distinto a la tabla para facilitar la selección de esos atributos. En estos casos no parece necesario, pero al mezclar varias tablas debemos indicar a qué tabla pertenece cada atributo que queremos mostrar.



Select

SELECT DISTINCT e.salida

FROM ETAPA AS e

Por defecto el select va seguido de ALL, es decir, muestra todos los resultados. Podemos cambiarlo e indicar DISTINCT, mostrará todos aquellos con distinto valor.

En caso de la vuelta a España, si hay 25 etapas y 2 repiten salida nos mostrará una lista con los 23 nombres que hay distintos.



Select

```
SELECT DISTINCT e.salida  
FROM ETAPA AS e  
ORDER BY e.salida ASC
```

Podemos añadir el ORDER BY para que se ordene el resultado por la columna indicada, puede ser alfabéticamente o numéricamente. Por defecto, se ordena en ASC, así que no haría falta indicarlo, pero si hace falta DESC.

En el ejemplo muestra las distintas salidas alfabéticamente.

ORDER BY 1 ASC; Ordena por la primera columna que devolvemos, el número indica la posición de la columna que utilizamos. En este caso solo hay una.

Select

```
SELECT [ ALL / DISTINCT ] [ * ] / [ListaColumnas_Expresiones] AS [Expresion]  
  
FROM Nombre_Tabla_Vista  
  
WHERE Condiciones  
  
ORDER BY ListaColumnas [ ASC / DESC ]
```

Con las explicaciones anteriores hemos visto todos los componentes que pueden participar en una consulta SELECT básica

Operadores



POLITÉCNICO
ESTELLA



Operadores de comparación y lógicos

Ya hemos visto algunos operadores =, <,>, =>,<=, <> para comparar valores.

Es posible incorporar AND, OR y NOT para agregar más campos en la consulta.

```
SELECT CICLISTA.Nombre AS Nombre del ciclista, CICLISTA.Apellido AS Apellido del ciclista  
FROM CICLISTA
```

```
WHERE peso < 60 AND sexo="H";
```

Peso menor de 60 y hombre.

```
WHERE NOT nombre="Enric";
```

Elige todos los que no se llamen Enric. Se puede usar con números

Operadores de comparación

BETWEEN

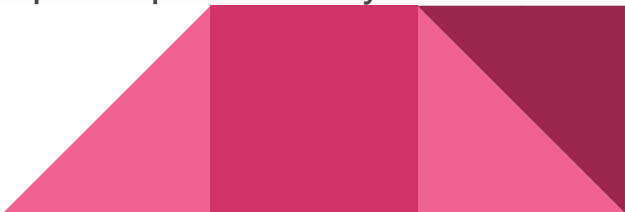
SELECT CICLISTA.Nombre AS Nombre del ciclista, CICLISTA.Apellido AS Apellido del ciclista

FROM CICLISTA

WHERE peso BETWEEN 50 AND 90;

LIKE

WHERE nombre LIKE 'a%'; Elige todos los nombres que empiecen con a, % implica que después puede venir cualquier valor. Podemos intercalar el % 'a%n' elige todos que empiezn en a y acaban con n.



Operadores de comparación

BETWEEN

```
SELECT CICLISTA.Nombre AS Nombre del ciclista, CICLISTA.Apellido AS Apellido del ciclista  
FROM CICLISTA
```

WHERE peso BETWEEN 50 AND 90; NOT BETWEEN lo opuesto

LIKE

WHERE nombre LIKE 'a%'; Elige todos los nombres que empiecen con a, % implica que después puede venir cualquier valor. Podemos intercalar el % 'a%n' elige todos que empiezn en a y acaban con n.

'a_b%' empiezan por a, otro valor, b y el resto del nombre. Es decir, primer y tercer caracter son a y b.

Operadores de comparación

NULL OR NOT NULL

SELECT CICLISTA.Nombre AS Nombre del ciclista, CICLISTA.Apellido AS Apellido del ciclista

FROM CICLISTA

WHERE peso IS NULL; Selecciona los datos de aquellos que no sabemos su peso. Si indicamos NOT NULL será lo contrario, mostrará aquellos que sabemos el peso.



Operadores aritméticos y concat.

Los aritméticos se entienden muy fácil, se aplican directamente al valor de salida.

```
SELECT nombre, precio*1.21 AS Productos con IVA  
FROM PRODUCTOS
```

Nos devuelve el precio aplicando el IVA de nuestra lista de productos



Operadores aritméticos y concat.

Los aritméticos se entienden muy fácil, se aplican directamente al valor de salida.

```
SELECT nombre, precio*1.21 AS Productos con IVA  
FROM PRODUCTOS
```

Nos devuelve el precio aplicando el IVA de nuestra lista de productos

```
SELECT nombre AS nombre empleado, apellido1 || apellido2 AS apellido empleado  
FROM EMPLEADOS
```

une apellido 1 y apellido 2 en una “columna” de salida



Precedencia (orden)



POLITÉCNICO
ESTELLA

Operadores aritméticos y concat.

Ocurre como en matemáticas, unos operadores se ejecutan antes que otros.

¿ $5*5+5=30$ Para hacerlo claro $(5*5)+5$.

Si queremos cambiar el orden $5*(5+5)=50$. El resultado es muy distinto, igual ocurre en las consultas.

1. Se evalúa la multiplicación (*) y la división (/) al mismo nivel
2. A continuación sumas (+) y restas (-).
3. Concatenación (||).
4. Todas las comparaciones (<, >, ...).
5. Después evaluaremos los operadores **IS NULL, IS NOT NULL, LIKE, BETWEEN.**
6. **NOT.**
7. **AND.**
8. **OR.**

Consultas calculadas

Consultas calculadas

Vistas en los operadores. Simplemente saber que no modifican los valores de la tabla.



Funciones y consultas resumen

Funciones

Explicamos solo algunas consultas resumen más utilizadas, funcionan todas igual.

```
SELECT MAX(Precio)
```

```
FROM Productos; // devuelve el producto con mayor precio, sólo 1.
```

```
SELECT COUNT(*)
```

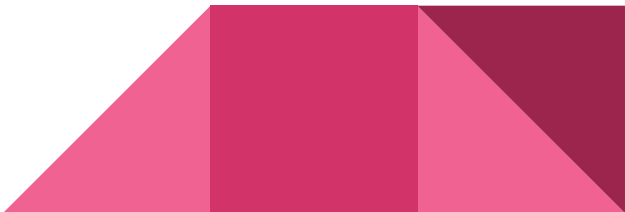
```
FROM Productos; // cuenta el número de productos, ya que cuenta todo. ¿Si queremos contar aquellos que valen más de 100 euros?
```

```
SELECT AVG(Price)
```

```
FROM Products; //media
```

```
SELECT SUM(Precios)
```

```
FROM Productos; // devuelve la suma de todos los productos
```



Agrupamientos



POLITÉCNICO
ESTELLA

Agrupamiento

Permite agrupar por datos de columna. Ejemplo

| VISITAS_MED | | | |
|-------------|--------|-------|------|
| dni | nombre | fecha | hora |
| | | | |
| | | | |

```
SELECT fecha, COUNT(*)  
FROM VISITAS_MED  
GROUP BY fecha;
```

Selecciona la fecha y cuenta todos los elementos agrupados en cada fecha.

| Fecha | Count |
|------------|-------|
| 20/11/2024 | 11 |
| 21/11/2024 | 14 |

Agrupamiento

Indicar condiciones con el HAVING

| VISITAS_MED | | | |
|-------------|--------|-------|------|
| dni | nombre | fecha | hora |
| | | | |
| | | | |

```
SELECT fecha, COUNT(*)  
FROM VISITAS_MED  
WHERE DNI <> `11111111A`  
GROUP BY fecha  
HAVING FECHA > '2024-11-20';
```

Selecciona la fecha y cuenta todos los elementos agrupados en fecha mayor que el día 20 para las personas que no tenga el dni 11111111A

| Fecha | Count |
|------------|-------|
| 21/11/2024 | 14 |

Consultas multitable



POLITÉCNICO
ESTELLA

Introducción

La consulta de múltiples tablas es la parte potente de las Bases de Datos Relacionales. Permite simplificar cada tabla ya que posteriormente podemos combinar el contenidos de todas ellas mediante los campos comunes. PK-FK

¿Cómo se pueden unir tablas?



Unión total de tablas.

Unir todas las combinaciones posibles de dos o más tablas (producto cartesiano). No se utiliza porque el resultado son tablas muy grandes.

```
SELECT ciclista.nombre, equipo.nombre  
FROM ciclista, equipo  
WHERE ciclista.id_equipo=equipo.id;
```



Composiciones externas

Se puede obtener información entre dos tablas, aunque una de ella contenga los valores NULL.
Veremos que esto es LEFT JOIN o RIGHT JOIN

```
SELECT ciclista.nombre, equipo.nombre
```

```
FROM ciclista, equipo
```

```
WHERE ciclista.id_equipo=equipo.id(+);
```

Tendremos todos los ciclistas, incluso aquellos que se han podido quedar sin equipo.



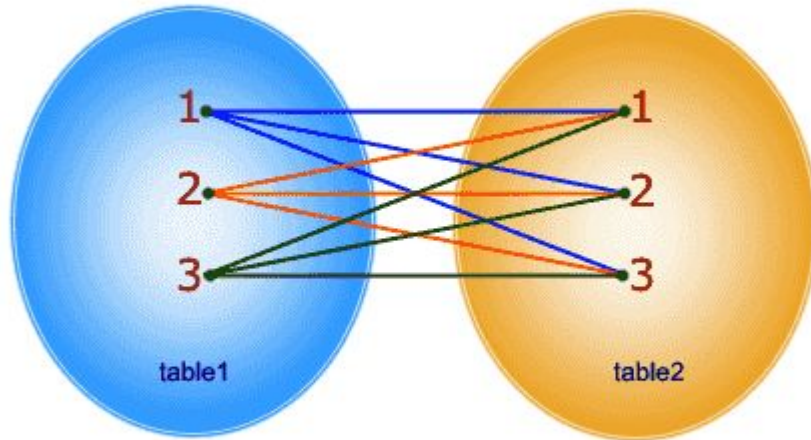
Diferentes JOINS

<https://www.w3resource.com/sql/joins>



CROSS JOIN

```
SELECT * FROM table1 CROSS JOIN table2;
```



In CROSS JOIN, each row from 1st table joins with all the rows of another table.
If 1st table contain x rows and y rows in 2nd one the result set will be $x * y$ rows.

```
SELECT *
```

```
FROM table1
```

```
CROSS JOIN table2;
```

```
ES LO MISMO
```

```
SELECT *
```

```
FROM table1, table2;
```


NATURAL JOIN



| ITEM_ID | ITEM_NAME | ITEM_UNIT | COMPANY_ID |
|---------|--------------|-----------|------------|
| 1 | Chex Mix | Pcs | 16 |
| 6 | Cheez-It | Pcs | 15 |
| 2 | BN Biscuit | Pcs | 15 |
| 3 | Mighty Munch | Pcs | 17 |
| 4 | Pot Rice | Pcs | 15 |
| 5 | Jaffa Cakes | Pcs | 18 |
| 7 | Salt n Shake | Pcs | - |

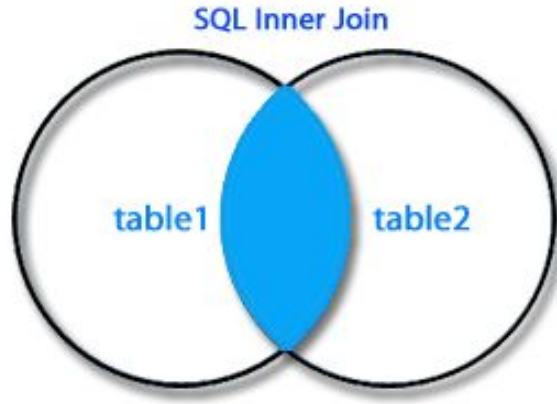
| COMPANY_ID | COMPANY_NAME | COMPANY_CITY |
|------------|---------------|--------------|
| 18 | Order All | Boston |
| 15 | Jack Hill Ltd | London |
| 16 | Akas Foods | Delhi |
| 17 | Foodies. | London |
| 19 | sip-n-Bite. | New York |

** Same column came once

| COMPANY_ID | ITEM_ID | ITEM_NAME | ITEM_UNIT | COMPANY_NAME | COMPANY_CITY |
|------------|---------|--------------|-----------|---------------|--------------|
| 16 | 1 | Chex Mix | Pcs | Akas Foods | Delhi |
| 15 | 6 | Cheez-It | Pcs | Jack Hill Ltd | London |
| 15 | 2 | BN Biscuit | Pcs | Jack Hill Ltd | London |
| 17 | 3 | Mighty Munch | Pcs | Foodies. | London |
| 15 | 4 | Pot Rice | Pcs | Jack Hill Ltd | London |
| 18 | 5 | Jaffa Cakes | Pcs | Order All | Boston |

SELECT *
FROM ITEMS
NATURAL JOIN COMPANIES;

JOIN O INNER JOIN



```
SELECT *
```

```
FROM EMPLEADO
```

```
JOIN PROYECTO ON EMPLEADO.proyecto_id =  
PROYECTO.id;
```

Devuelve solo los empleados que tienen proyecto asignado.

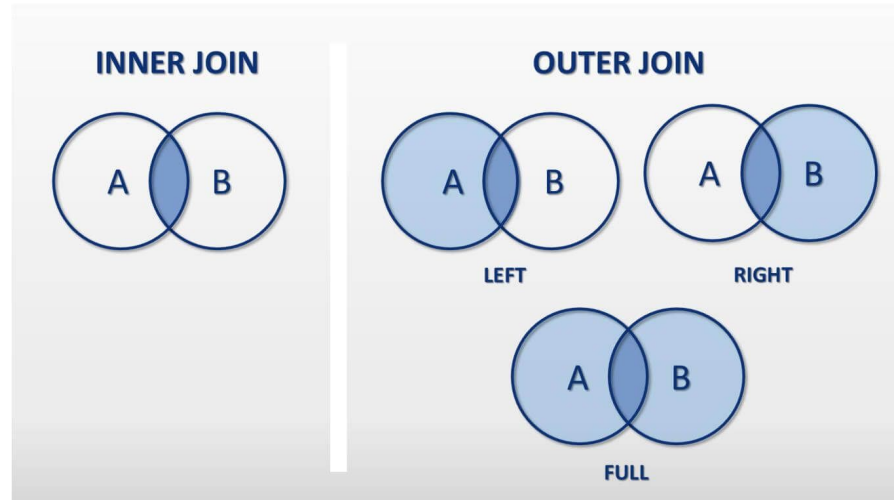
OUTER JOIN

Permite seleccionar todas las tuplas, aunque el valor sea null.

LEFT JOIN

RIGHT JOIN

FULL OUTER JOIN



LEFT/RIGHT OUTER JOIN

SELECT *

FROM EMPLEADO

LEFT JOIN PROYECTO

ON EMPLEADO.proyecto_id = PROYECTO.id;

Selecciona todas las filas de empleado porque está a la izquierda, aunque no tengan proyecto asociado.

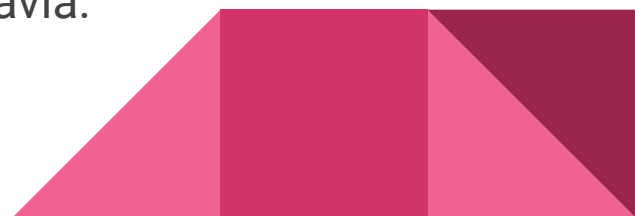
SELECT *

FROM EMPLEADO

RIGHT JOIN PROYECTO

ON EMPLEADO.proyecto_id = PROYECTO.id;

Selecciona todas las filas de proyecto porque está a la derecha, aunque no tengan empleados asignados todavía.



LEFT/RIGHT OUTER JOIN

SELECT *

FROM EMPLEADO

FULL OUTER JOIN PROYECTO

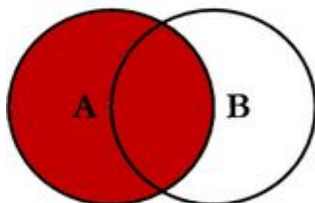
ON EMPLEADO.proyecto_id = PROYECTO.id;

Selecciona todas las filas de empleado y proyecto, tanto si tienen asignado alguno como no.

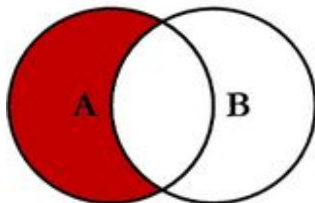
No es lo mismo que CROSS JOIN, cross join crea todas las combinaciones posibles. Aquí mostrará todos los empleados, empleados con proyecto asignado y todos los proyectos.



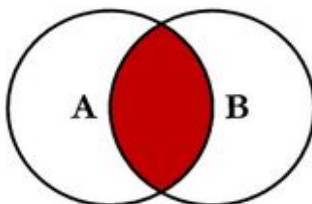
SQL JOINS



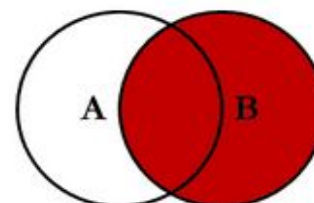
```
SELECT <select_list>  
FROM TableA A  
LEFT JOIN TableB B  
ON A.Key = B.Key
```



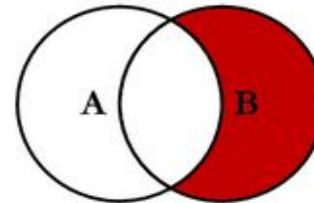
```
SELECT <select_list>  
FROM TableA A  
LEFT JOIN TableB B  
ON A.Key = B.Key  
WHERE B.Key IS NULL
```



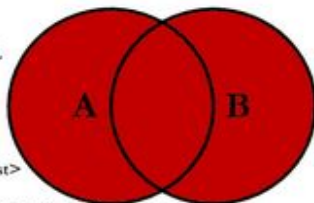
```
SELECT <select_list>  
FROM TableA A  
INNER JOIN TableB B  
ON A.Key = B.Key
```



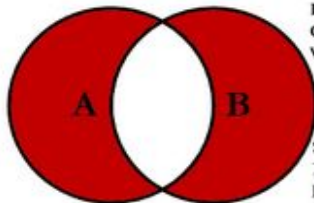
```
SELECT <select_list>  
FROM TableA A  
RIGHT JOIN TableB B  
ON A.Key = B.Key
```



```
SELECT <select_list>  
FROM TableA A  
RIGHT JOIN TableB B  
ON A.Key = B.Key  
WHERE A.Key IS NULL
```



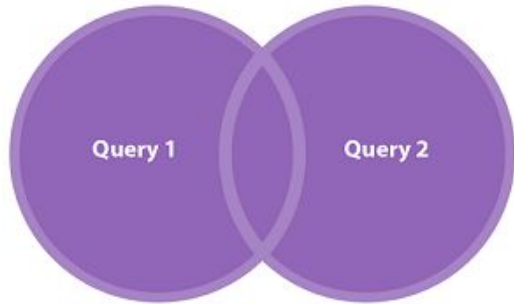
```
SELECT <select_list>  
FROM TableA A  
FULL OUTER JOIN TableB B  
ON A.Key = B.Key
```



```
SELECT <select_list>  
FROM TableA A  
FULL OUTER JOIN TableB B  
ON A.Key = B.Key  
WHERE A.Key IS NULL  
OR B.Key IS NULL
```

Consultas multitable II

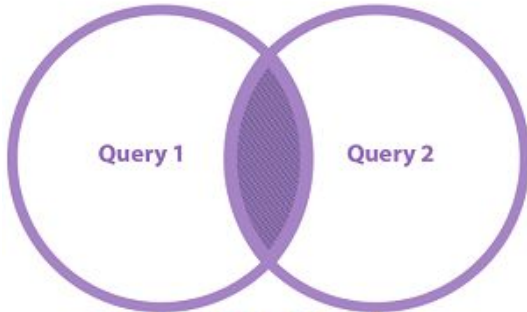
UNION, INTERSECT AND EXCEPT



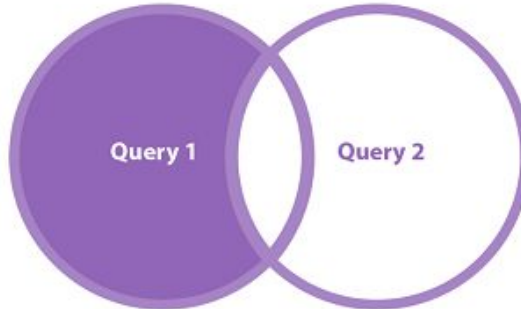
UNION



UNION ALL



INTERSECT



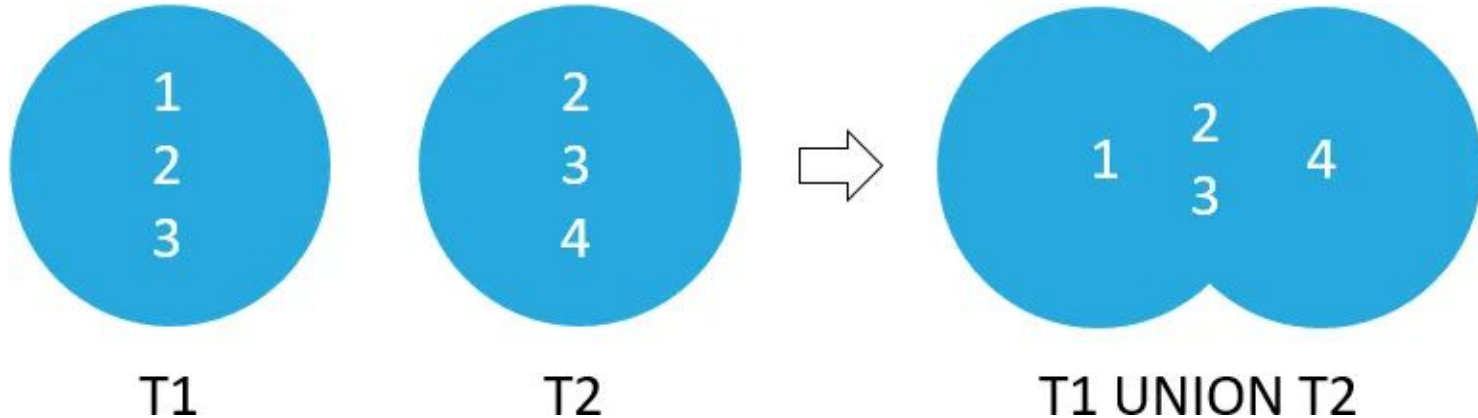
EXCEPT

Sirven para **combinar** datos de dos o más **consultas**. Es decir, mezclar varios select.



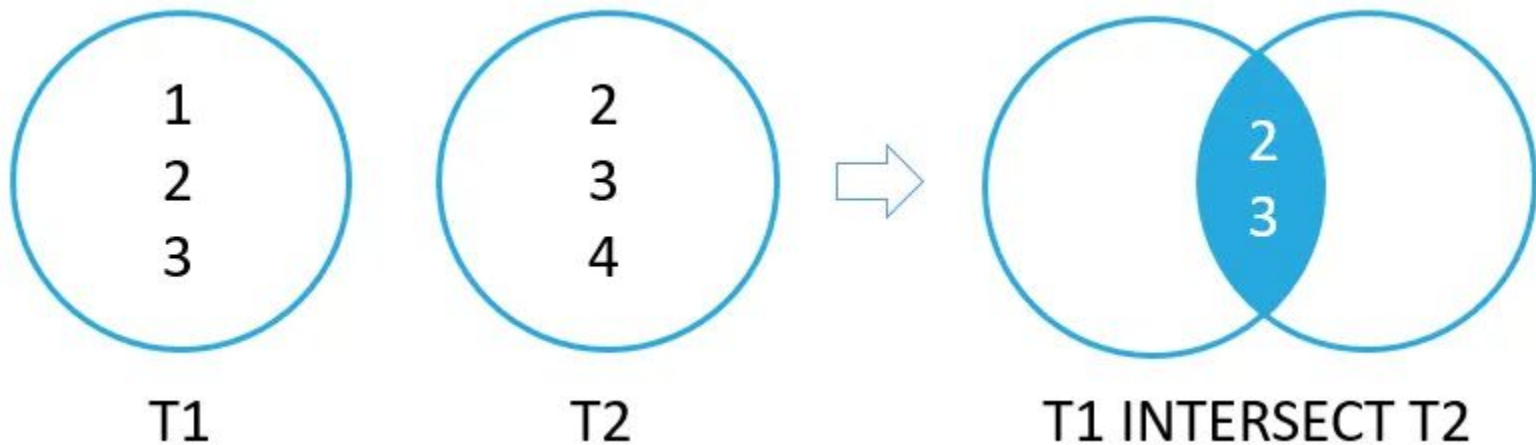
UNION

NO CREA DUPLICADOS

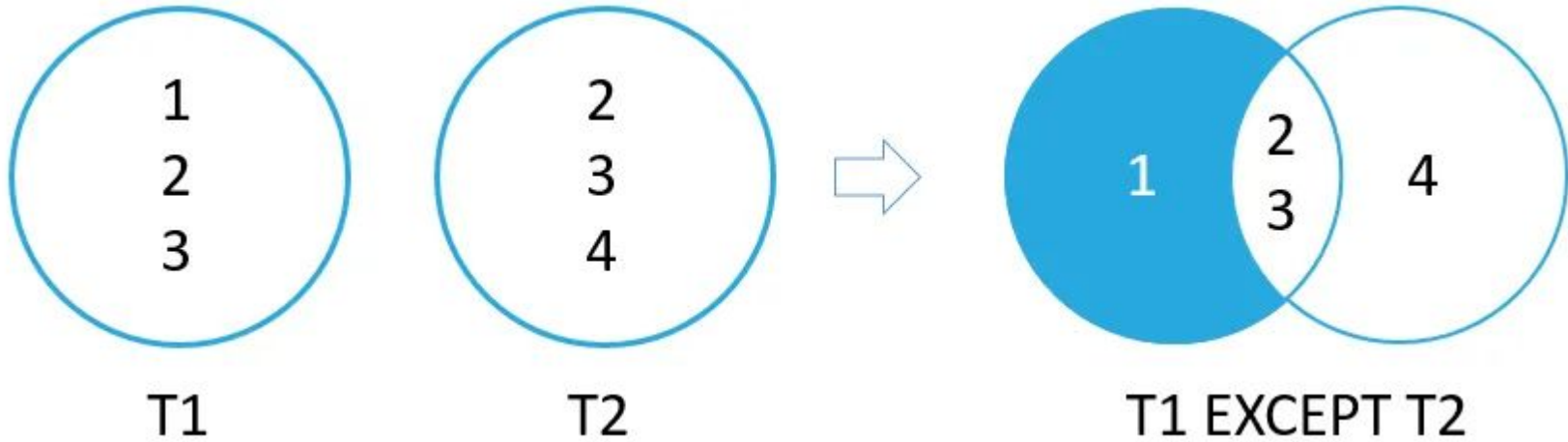


SELECT name FROM customers UNION SELECT name FROM employees; *Devuelve nombres de ambos sin duplicar*

INTERSECT



INTERSECT



Subconsultas

Introducción

Incluir una consulta dentro de la cláusula WHERE o HAVING de otra.



Introducción

SELECT nombre, salario

FROM EMPLEADOS

WHERE salario <= ALL (SELECT salario FROM EMPLEADOS);

Ejemplo de los apuntes. Podemos comparar con los operadores que conocemos y jugar con ANY, ALL, IN, NOT IN en caso de devolver más de una fila y nosotros querer solamente 1 valor. En el empleo solamente quiere el empleado o empleados con mayor salario

Introducción

ALTERNATIVA

SELECT nombre, salario

FROM EMPLEADOS

ORDER BY salario ASC

LIMIT 1;



VIEWS



POLITÉCNICO
ESTELLA

Introducción

