



## TEMA - IV:

### REALIZACIÓN DE CONSULTAS

**BD - (1º DAM)**



# ÍNDICE

1. SENTENCIA SELECT .....	4
2. LAS FUNCIONES.....	5
3. COMBINACION DE TABLAS .....	11
3.1. LA UNIÓN NATURAL .....	11
3.2. INNER JOIN – RIGHT JOIN – LEFT JOIN – FULL JOIN .....	12
4. SUBCONSULTAS .....	13
4.1. SUBCONSULTA QUE DEVUELVE UN VALOR.....	13
4.2. SUBCONSULTA QUE DEVUELVE UNA LISTA DE VALORES.....	13
4.3. SUBCONSULTA QUE DEVUELVE MÁS DE UNA COLUMNA.....	13
4.4. SUBCONSULTA CON CUALQUIER NUMERO DE FILAS Y COLUMNAS.....	13
5. UNION (INTERSECCIÓN y DIFERENCIA) .....	14
6. CONSULTAS CORRELACIONADAS .....	15
7. AGRUPAMIENTO .....	16
8. EJEMPLOS .....	17
9. EJERCICIOS.....	19
9.1. RESUELTOS.....	19
9.1.1. BASICOS .....	19
9.1.2. COMBINACION DE TABLAS .....	25
9.1.3. SUBCONSULTAS .....	26
9.1.4. AGRUPAMIENTO.....	27
9.2. PROPUESTOS.....	28
9.2.1. Bloque - A .....	28
9.2.2. Bloque - B .....	30
9.2.3. Bloque - C .....	32
9.2.4. Bloque - D .....	34

## 1. SENTENCIA **SELECT**.

```
SELECT [DISTINCT] A1 , A2, ...
    FROM B...
    [WHERE ... ]
    [LIMIT m,n]
    [INTO OUTFILE 'nombre_fichero']
    [GROUP BY...]
    [HAVING ... ]
    [ORDER BY ... ];
```

**SELECT** [ALL | DISTINCT] [\* , colum1,column2,..] AS Nuevo\_nombre

SELECT recupera información. \* recupera todas las filas. ALL recupera todas las filas incluidas las repetidas. DISTINCT recupera todas sin las repetidas. Colum1 puede ser una columna, cte, exp. aritmética, una o mas funciones.

**FROM** [tabla1, tabla2,...]

Especifica la/s tabla/s donde están los datos. Si el usuario no es el propietario de la tabla debe especificar el nombre: PROPIETARIO.TABLA1

**[WHERE condición]**

Obtiene las filas que cumplen la condición expresada.

**Condición = expresión operador expresión;**

Expresión puede ser una cte, exp. aritmética, valor nulo o columna

**Operador:**

- **Comparación**

- =, <, >, <>, >=, <=, !=,
- IN (grupo de valores separados por comas), NOT IN,
- BETWEEN v\_inicial AND v\_final, NOT BETWEEN,
- LIKE valor ( las cadenas de caracteres se escriben entre comillas simples, '%' representa cualquier cadena de 0 o mas caracteres, '\_' representa cualquier carácter.

- **Lógicos:** para construir condiciones múltiples: **AND, OR y NOT.**

- **Aritméticos:** sirven para formar expresiones aritméticas con ctes, valores de columnas y funciones de valores de columnas. +, -, \*, / **suma, resta, producto, división.**

Columna **IS NULL, IS NOT NULL** sirve para comprobar si la columna de una fila está vacía.

**[ORDER BY column1 DESC/ASC, column2...]**

Especifica el criterio de ordenación del resultado. ASC ascendente, DESC descendente.

<b>EJERCICIOS:</b>	-RESUELTOS: (BASICOS)	-PROPUESTOS: (1,2,3)
--------------------	-----------------------	----------------------

## 2. LAS FUNCIONES

<http://mysql.conclase.net/coste/?cap=011>

Son funciones incorporadas por el gestor y son muy utilizadas en SQL y dan mucha potencia al lenguaje. Estas funciones predefinidas devuelven un valor dependiendo del valor de un argumento que se pasa en la llamada.

Cabe subrayar que las funciones no modifican los valores del argumento, indicado entre paréntesis, sino que devuelven un valor creado a partir de los argumentos que se le pasan en la llamada, y ese valor puede ser utilizado en cualquier parte de una sentencia SQL.

Existen muchas funciones predefinidas para operar con todo tipo de datos. A continuación se indican las funciones predefinidas más utilizadas.

Vamos a ver estas funciones agrupadas según el tipo de datos con los que operan y/o el tipo de datos que devuelven.

*Nota: estas funciones pueden variar según el sistema gestor que se utilice.*

### Funciones numéricas o aritméticas

Función	Valor que devuelve.
ABS( <i>num</i> )	<b>Valor absoluto.</b> Valor absoluto de <i>num</i> .
CEIL( <i>num</i> )	<b>Función “Techo”</b> Devuelve el entero mas pequeño mayor que <i>num</i>
FLOOR( <i>num</i> )	<b>Función “Suelo”</b> Devuelve el entero mas grande menor que <i>num</i>
EXP( <i>num</i> )	<b>Potencia del número e</b> Devuelve el número e elevado a <i>num</i>
LN( <i>num</i> )	<b>Logaritmo neperiano</b> Devuelve el logaritmo en base e de <i>num</i>
LOG( <i>num</i> )	<b>Logaritmo</b> Devuelve el logaritmo en base 10 de <i>num</i>
MOD( <i>num1, num2</i> ).	<b>Módulo</b> Resto de la división entera de <i>num1</i> por <i>num2</i>
PI()	<b>Pi</b> Devuelve el valor de la constante PI
POWER( <i>num1, num2</i> ).	<b>Potencia</b> Devuelve <i>num1</i> elevado a <i>num2</i> .
RAND()	<b>Número aleatorio</b> Genera un número aleatorio entre 0 y 1
ROUND( <i>num1, num2</i> )	<b>Redondeo</b> Devuelve <i>num1</i> redondeado a <i>num2</i> decimales. Si se omite <i>num2</i> redondea a 0 decimales.
SIGN( <i>num</i> ).	<b>Signo</b> Si <i>num</i> < 0 devuelve -1. Si <i>num</i> = 0 devuelve 0 Si <i>num</i> > 0 devuelve 1.
SQRT( <i>num</i> ).	<b>Raíz cuadrada</b> Devuelve la raíz cuadrada de <i>num</i>
TRUNCATE( <i>num1, num2</i> ).	<b>Truncado</b> Devuelve <i>num1</i> truncado a <i>num2</i> decimales. Si se omite <i>num2</i> trunca a 0 decimales.

Funciones de caracteres

Función	Valor que devuelve.
ASCII( <i>cad1</i> ).	<b>ASCII</b> Código ASCII del carácter <i>cad1</i> .
CHAR( <i>num</i> )	<b>Carácter ASCII</b> Devuelve el carácter cuyo código ASCII es <i>num</i>
CONCAT( <i>cad1,cad2 [,cad3...]</i> )	<b>Concatenar</b> Concatena <i>cad1</i> con <i>cad2</i> . Si existiesen más cadenas, <i>cad3...</i> , las concatenaría a continuación
INSERT( <i>cad1, pos, ,len, cad2</i> )	<b>Insertar</b> Devuelve <i>cad1</i> con <i>len</i> caracteres desde <i>pos</i> en adelante sustituidos en <i>cad2</i>
LENGTH( <i>cad1</i> )	<b>Longitud</b> Devuelve la longitud de <i>cad1</i> .
LOCATE( <i>cad1,cad2,pos</i> )	<b>Localizar</b> Devuelve la posición de la primera ocurrencia de <i>cad1</i> en <i>cad2</i> empezando desde <i>pos</i>
LOWER( <i>cad1</i> )	<b>Minúsculas</b> La cadena <i>cad1</i> en minúsculas.

<b>LPAD(<i>cad1, n, cad2</i>)</b>	<b>Rellenar (Izquierda)</b> Añade a <i>cad1</i> por la izquierda <i>cad2</i> , hasta que tenga <i>n</i> caracteres. Si se omite <i>cad2</i> , añade blancos.
<b>LTRIM(<i>cad1</i>)</b>	<b>Suprimir (Izquierda)</b> Suprime blancos a la izquierda de <i>cad1</i> .
<b>REPLACE(<i>cad1,cad2,cad3</i>)</b>	<b>Reemplazar</b> Devuelve <i>cad1</i> con todas las ocurrencias de <i>cad2</i> reemplazadas por <i>cad3</i>
<b>RPAD(<i>cad1, n, cad2</i>)</b>	<b>Rellenar (Derecha)</b> Igual que LPAD pero por la derecha.
<b>RTRIM(<i>c1</i>)</b>	<b>Suprimir (Derecha)</b> Suprime blancos a la derecha de <i>c1</i> . Igual que LTRIM pero por la izquierda.
<b>SUBSTR(<i>c1, n, m</i>)</b>	<b>Subcadena</b> Devuelve una subcadena a partir de <i>c1</i> comenzando en La posición <i>n</i> tomando <i>m</i> caracteres.
<b>UPPER(<i>cad1</i>)</b>	<b>Mayúsculas</b> La cadena <i>cad1</i> en mayúsculas.

Funciones de fecha

Función	Valor que devuelve.
<b>ADDDATE(<i>Fecha</i>, <i>Num</i>)</b>	<b>Incremento de días</b> Devuelve <i>Fecha</i> incrementada en <i>Num</i> días
<b>SUBDATE(<i>Fecha</i>, <i>Num</i>)</b>	<b>Decremento de días</b> Devuelve <i>Fecha</i> decrementada en <i>Num</i> días
<b>DATE_ADD(<i>Fecha</i>, INTERVAL <i>Num</i> <i>Formato</i>)</b>	<b>Incremento</b> Devuelve <i>Fecha</i> incrementada en <i>Num</i> veces lo indicado en <i>Formato</i> <i>Formato</i> puede ser entre otros: DAY, WEEK, MONTH, YEAR, HOUR, MINUTE, SECOND
<b>DATE_SUB(<i>Fecha</i>, INTERVAL <i>num</i> <i>Formato</i>)</b>	<b>Decremento</b> Devuelve <i>Fecha</i> decrementada en <i>Num</i> veces lo indicado en <i>Formato</i> <i>Formato</i> puede ser entre otros: DAY, WEEK, MONTH, YEAR, HOUR, MINUTE, SECOND
<b>DATEDIFF(<i>Fecha1</i>,<i>Fecha2</i>)</b>	<b>Diferencia de fechas</b> Devuelve el número de días entre <i>Fecha1</i> y <i>Fecha2</i>
<b>DAYNAME(<i>Fecha</i>)</b>	<b>Nombre del día de la semana</b> Devuelve el nombre del día de la semana de <i>Fecha</i>
<b>DAYOFMONTH(<i>Fecha</i>)</b>	<b>Día del mes</b> Devuelve el número del dia del mes de <i>Fecha</i>
<b>DAYOFWEEK(<i>Fecha</i>)</b>	<b>Día de la semana</b> Devuelve el número del día de la semana de <i>Fecha</i> (1:Domingo, 2:Lunes.....7:Sábado)
<b>DAYOFYEAR(<i>Fecha</i>)</b>	<b>Día del año</b> Devuelve el número de día del año de <i>Fecha</i> (de 1 a 366)
<b>WEEKOFYEAR(<i>Fecha</i>)</b>	<b>Semana</b> Devuelve el número de semana de <i>Fecha</i> (de 1 a 53)
<b>MONTH(<i>Fecha</i>)</b>	<b>Mes</b> Devuelve el número de mes de <i>Fecha</i> ( de 1 a 12)
<b>YEAR(<i>Fecha</i>)</b>	<b>Año</b> Devuelve el número de año con 4 dígitos de <i>Fecha</i> (de 0000 a 9999)
<b>HOUR(<i>Tiempo</i>)</b>	<b>Hora</b> Devuelve la hora de <i>Tiempo</i> (de 0 a 23)
<b>MINUTE(<i>Tiempo</i>)</b>	<b>Minutos</b> Devuelve los minutos de <i>Tiempo</i> (de 0 a 59)
<b>SECOND(<i>Tiempo</i>)</b>	<b>Segundos</b> Devuelve los segundos de <i>Tiempo</i> (de 0 a 59)
<b>CURDATE()</b>	Devuelve la fecha actual con el formato 'YYYY-MM-DD'
<b>CURTIME()</b>	Devuelve la hora actual con el formato 'HH:MM:SS'
<b>SYSDATE()</b>	Devuelve la fecha y la hora actual con el formato 'YYYY-MM-DD HH:MM:SS'

Conversión de fechas a otro tipo de datos:

Función	Valor que devuelve.																																						
<b>DATE_FORMAT(Fecha,Formato)</b>	Devuelve una cadena de caracteres con la Fecha con el Formato especificado. El formato es una cadena de caracteres que incluye las siguientes máscaras: <b>Máscara Descripción</b>																																						
	<table border="0"> <tr> <td>%a</td><td>Abreviatura (3 letras) del nombre del día de la semana</td></tr> <tr> <td>%b</td><td>Abreviatura (3 letra ) del nombre mes</td></tr> <tr> <td>%c</td><td>Número del mes (1 a 12)</td></tr> <tr> <td>%e</td><td>Número del dia del mes (0 a 31)</td></tr> <tr> <td>%H</td><td>Número de la hora en formato 24 horas (00 a 23)</td></tr> <tr> <td>%h</td><td>Número de la hora en formato 12 horas (01 a 12)</td></tr> <tr> <td>%i</td><td>Número de minutos (00 a 59)</td></tr> <tr> <td>%j</td><td>Número del dia del año (001 a 366)</td></tr> <tr> <td>%M</td><td>Nombre del mes</td></tr> <tr> <td>%m</td><td>Número de mes (01 a 12)</td></tr> <tr> <td>%p</td><td>Am o PM</td></tr> <tr> <td>%r</td><td>Hora en formato 12 horas (hh:mm seguido de AM o PM)</td></tr> <tr> <td>%s</td><td>Número de segundos (00 a 59)</td></tr> <tr> <td>%T</td><td>Hora en formato 24 horas (hh:mm:ss)</td></tr> <tr> <td>%u</td><td>Número de semana en el año (00 a 53)</td></tr> <tr> <td>%W</td><td>Nombre del día de la semana</td></tr> <tr> <td>%w</td><td>Número del dia de la semana (0:domingo a 6:Sábado)</td></tr> <tr> <td>%Y</td><td>Número de año con cuatro dígitos</td></tr> <tr> <td>%y</td><td>Número de año con dos dígitos</td></tr> </table>	%a	Abreviatura (3 letras) del nombre del día de la semana	%b	Abreviatura (3 letra ) del nombre mes	%c	Número del mes (1 a 12)	%e	Número del dia del mes (0 a 31)	%H	Número de la hora en formato 24 horas (00 a 23)	%h	Número de la hora en formato 12 horas (01 a 12)	%i	Número de minutos (00 a 59)	%j	Número del dia del año (001 a 366)	%M	Nombre del mes	%m	Número de mes (01 a 12)	%p	Am o PM	%r	Hora en formato 12 horas (hh:mm seguido de AM o PM)	%s	Número de segundos (00 a 59)	%T	Hora en formato 24 horas (hh:mm:ss)	%u	Número de semana en el año (00 a 53)	%W	Nombre del día de la semana	%w	Número del dia de la semana (0:domingo a 6:Sábado)	%Y	Número de año con cuatro dígitos	%y	Número de año con dos dígitos
%a	Abreviatura (3 letras) del nombre del día de la semana																																						
%b	Abreviatura (3 letra ) del nombre mes																																						
%c	Número del mes (1 a 12)																																						
%e	Número del dia del mes (0 a 31)																																						
%H	Número de la hora en formato 24 horas (00 a 23)																																						
%h	Número de la hora en formato 12 horas (01 a 12)																																						
%i	Número de minutos (00 a 59)																																						
%j	Número del dia del año (001 a 366)																																						
%M	Nombre del mes																																						
%m	Número de mes (01 a 12)																																						
%p	Am o PM																																						
%r	Hora en formato 12 horas (hh:mm seguido de AM o PM)																																						
%s	Número de segundos (00 a 59)																																						
%T	Hora en formato 24 horas (hh:mm:ss)																																						
%u	Número de semana en el año (00 a 53)																																						
%W	Nombre del día de la semana																																						
%w	Número del dia de la semana (0:domingo a 6:Sábado)																																						
%Y	Número de año con cuatro dígitos																																						
%y	Número de año con dos dígitos																																						

Funciones de comparación

Función	Valor que devuelve.
<b>GREATEST(<i>lista de valores</i>)</b>	<b>Mayor de la lista</b> Devuelve el valor más grande de una lista de columnas o expresiones de columna
<b>LEAST(<i>lista de valores</i>)</b>	<b>Menor de la lista</b> Devuelve el valor más pequeño de una lista de columnas o expresiones de columna
<b>IFNULL(<i>exp1, exp2</i>)</b>	<b>Conversión de nulos</b> Si <i>exp1</i> es nulo devuelve <i>exp2</i> , sino devuelve <i>exp1</i>
<b>ISNULL(<i>exp</i>)</b>	<b>Comprobación de nulo</b> Devuelve 1( True) si <i>exp</i> es NULL y 0 (False) en caso contrario
<b>STRCMP(<i>cad1,cad2</i>)</b>	<b>Comparación de cadenas</b> Devuelve 1(True) si <i>cad1</i> y <i>cad2</i> son iguales, 0(False) si no lo son y NULL si alguna de ellas es nula

Otras funciones:

Función	Valor que devuelve.
DATABASE()	<b>Base de datos</b> Nombre de la base de datos actual
USER()	<b>Usuario</b> Devuelve el usuario y el host de la sesión usuario@host
VERSION()	<b>Versión</b> Devuelve una cadena indicando la versión que estamos utilizando

EJEMPLOS:FUNCIONES ARITMÉTICAS

- ABS(n)
  - . Propósito: devuelve el valor absoluto de n.
  - . Ejemplo: **SELECT ABS (-15)**
  - . Resultado: 15
- CEILING(n)
  - . Propósito: devuelve el valor entero superior o igual a n.
  - . Ejemplo: **SELECT CEIL (3,152)**
  - . Resultado: 4
- FLOOR(n)
  - . Propósito: devuelve el valor entero inferior o igual a n.
  - . Ejemplo: **SELECT FLOOR (3,152)**
  - . Resultado: 3
- POWER(m, n)
  - . Propósito: devuelve el valor m elevado a n.
  - . Ejemplo: **SELECT POWER (3,2)**
  - . Resultado: 9
- ROUND(n, [m])
  - . Propósito: devuelve el valor n redondeado a m decimales. Si m es negativo, el redondeo se hace a la izquierda de la parte decimal. Si se omite la m, se muestran 0 decimales
  - . Ejemplo: Ver apellidos y Salarios, pero salarios en miles.  
**SELECT Apellido, ROUND (Salario, -3)**  
**FROM Plantilla**
- SQRT(n)
  - . Propósito: devuelve la raíz cuadrada de n.
  - . Ejemplo: **SELECT ABS (4)**
  - . Resultado: 2
- TRUNCATE(n,m)
  - . Propósito: devuelve N truncado a m decimales, m puede ser negativo.
  - . Ejemplo: **SELECT TRUNC (3,156)**
  - . Resultado: 3

FUNCIONES DE CADENAS DE CARACTERES

- CHAR(n) . Propósito: devuelve el carácter cuyo valor en ASCII es n.

. Ejemplo: **SELECT CHR (48)**

. Resultado: 0

- CONCAT(cadena1, cadena2)

. Propósito: devuelve la concatenación de 2 cadenas = ||.

- LPAD(cadena1, m[,cadena2])

. Propósito: devuelve cadena1 con longitud m y ajustando a la derecha. Cadena2 es el carácter con que se rellena por la izquierda.

. Ejemplo: **SELECT LPAD ('P',5,'x')**

. Resultado: xxxxP

- RPAD(cadena1, m[,cadena2])

. Propósito: devuelve cadena1 con longitud m y ajustando a la izquierda. Cadena2 es el carácter con que se rellena por la derecha.

. Ejemplo: **SELECT RPAD ('P',5,'x')**

. Resultado: Pxxxx

Ejercicio: **SELECT LPAD (Apellido,12,'X') 'VIVE EN' RPAD (Direccion,20,'x')**  
FROM Enfermo

- REPLACE(cadena,ocurrencia[, cadena\_a\_reemplazar])

. Propósito: devuelve cadena con la ocurrencia sustituida por cadena\_a\_reemplazar.

. Ejemplo: **SELECT REPLACE ('JACK AND JUE','J','BL'))**  
FROM Dual

. Resultado: BLACK AND BLUE.

- SUBSTRING(cadena, m[, n])

. Propósito: devuelve la subcadena de cadena que abarca desde la posición número m hasta la número n. El valor de n no puede ser inferior a 1. El valor de m puede ser negativo.

. Ejemplo: **SELECT SUBSTRING ABCDEFG',3,2)**

. Resultado: CD.

FUNCIONES DE CADENAS DE CARÁCTERES QUE DEVUELVEN UN NÚMERO

- ASCII(carácter) . Propósito: devuelve el código ASCII del carácter introducido.

. Ejemplo: **SELECT ASCII ('A')**

. Resultado: 65

- LENGTH(cadena). Propósito: devuelve la longitud de la cadena.

. Ejemplo: **SELECT LENGTH ('12345')**

. Resultado: 5

### 3. COMBINACION DE TABLAS

Hay veces que una consulta necesita columnas de varias tablas, las cuales se expresarán a la derecha del FROM:

```
SELECT columnas de las tablas del "from"
FROM tabla1, tabla2, ...
WHERE tabla1.columna = tabla2.columna;
```

Cuando combinamos varias tablas hay que tener en cuenta unas **reglas**:

- 1) Es posible unir tantas tablas como deseemos
  - 2) En la cláusula SELECT se pueden citar columnas de todas las tablas
  - 3) Si hay columnas con el mismo nombre, en las distintas tablas de la cláusula FROM se deben identificar especificando NombreTabla.NombreColumna (usaremos alias)
  - 4) El criterio que se siga para combinar las tablas ha de especificarse en la cláusula WHERE. Si se omite esta cláusula, el resultado será un PRODUCTO CARTESIANO que emparejará todas las filas de una tabla con cada fila de la otra
- Sacar el Apellido y lugar de trabajo de todos los empleados que trabajen en una ciudad que empiece por M.

```
SELECT dept_no --> poner sinónimo sin nombre de tabla.
FROM dept d, emp e
WHERE (loc LIKE 'M%') and (d.dept_no=e.dept_no)
```

#### 3.1. LA UNIÓN NATURAL

```
SELECT T1.A , T1.B , T2.A , T2.C.
FROM Tabla1 T1 , Tabla2 T2
WHERE T1.A = T2.A
```

[http://www.aulaclic.es/sqlserver/t\\_4\\_5.htm](http://www.aulaclic.es/sqlserver/t_4_5.htm)

**EJERCICIOS:**

-RESUELTOS: (COMBINACIÓN DE TABLAS)

-PROPUESTOS: (4-13)

**3.2. INNER JOIN – RIGHT JOIN – LEFT JOIN – FULL JOIN**

PROFESOR		ALUMNO			AULA		
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
DNIP	nombrep	DNIA	nombrea	DNIP	CODA	nombreau	DNIP
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
P1	Pedro	A1	Jose	P1	A-101	1º ASIR	P1
P2	David	A2	Javier	P2	A-102	2º DAW	P2
P3	Ivan	A3	Antonio	NULL	A-103	LIBRE	NULL
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+

**INNER JOIN**

```
mysql> SELECT *
      -> FROM Profesor INNER JOIN Alumno ON
Profesor.DNIP=Alumno.DNIP;
```

+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
DNIP	nombrep	DNIA	nombrea	DNIP	CODA	nombreau	DNIP
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
P1	Pedro	A1	Jose	P1	A-101	1º ASIR	P1
P2	David	A2	Javier	P2	A-102	2º DAW	P2
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+

```
mysql> **** INNER JOIN (tres tablas) ****
mysql> SELECT *
```

```
> FROM (Profesor INNER JOIN Alumno ON Profesor.DNIP=Alumno.DNIP)
-> INNER JOIN Aula ON Profesor.DNIP=Aula.DNIP;
```

+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
DNIP	nombrep	DNIA	nombrea	DNIP	CODA	nombreau	DNIP
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
P1	Pedro	A1	Jose	P1	A-101	1º ASIR	P1
P2	David	A2	Javier	P2	A-102	2º DAW	P2
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+

**RIGHT JOIN**

```
mysql> SELECT *
      -> FROM Profesor RIGHT JOIN Alumno ON
Profesor.DNIP=Alumno.DNIP;
```

+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
DNIP	nombrep	DNIA	nombrea	DNIP	CODA	nombreau	DNIP
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
P1	Pedro	A1	Jose	P1	A-101	1º ASIR	P1
P2	David	A2	Javier	P2	A-102	2º DAW	P2
NULL	NULL	A3	Antonio	NULL			
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+

**LEFT JOIN**

```
mysql> SELECT *
      -> FROM Profesor LEFT JOIN Alumno ON
Profesor.DNIP=Alumno.DNIP;
```

+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
DNIP	nombrep	DNIA	nombrea	DNIP	CODA	nombreau	DNIP
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
P1	Pedro	A1	Jose	P1	A-101	1º ASIR	P1
P2	David	A2	Javier	P2	A-102	2º DAW	P2
P3	Ivan	NULL	NULL	NULL			
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+

**FULL JOIN**

```
mysql> SELECT *
      -> FROM Profesor RIGHT JOIN Alumno ON
Profesor.DNIP=Alumno.DNIP
      -> UNION
      -> SELECT *
      -> FROM Profesor LEFT JOIN Alumno ON
Profesor.DNIP=Alumno.DNIP;
```

+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
DNIP	nombrep	DNIA	nombrea	DNIP	CODA	nombreau	DNIP
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
P1	Pedro	A1	Jose	P1	A-101	1º ASIR	P1
P2	David	A2	Javier	P2	A-102	2º DAW	P2
NULL	NULL	A3	Antonio	NULL			
P3	Ivan	NULL	NULL	NULL			
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+

#### **4. SUBCONSULTAS**

Hasta ahora en la cláusula where hemos utilizado una constante o un grupo de constantes para comparar con los valores de las columnas. Pero si se desconocen los valores de las columnas se puede incluir una cláusula select completa dentro de la cláusula where. Esto es lo que se llama subconsulta o subconsulta anidada.

Ejemplo: Apellido de los empleados del hospital que tienen un salario superior a la media.

```
SELECT apellido
FROM Plantilla
WHERE salario > (SELECT AVG(salario)
FROM Plantilla);
```

##### **4.1. SUBCONSULTA QUE DEVUELVE UN VALOR**

El operador relacional utilizado puede ser cualquier operador relacional ( $<$ ,  $>$ ,  $=$ ,  $<>$ ,  $\leq$ ,  $\geq$ ).

Nombre y oficio de los empleados que tienen el mismo trabajo que Jiménez.

```
SELECT apellido , oficio
FROM Emp
WHERE oficio = (SELECT oficio
FROM emp
WHERE apellido = 'Jiménez');
```

##### **4.2. SUBCONSULTA QUE DEVUELVE UNA LISTA DE VALORES**

El operador de comparación será un operador lógico de conjuntos (**IN**, operador relacional más **ALL** o **ANY**).

Nombre de los empleados que trabajan en un departamento de Madrid o Barcelona.

```
SELECT apellido
FROM Emp
WHERE dept_no IN (SELECT dept_no
FROM Dept
WHERE loc IN ('Madrid', 'Barcelona'));
```

##### **4.3. SUBCONSULTA QUE DEVUELVE MÁS DE UNA COLUMNA**

Apellido de los empleados con el mismo oficio y salario que Jiménez.

```
SELECT apellido
FROM Emp
WHERE (oficio,salario) = (SELECT oficio, salario
FROM Emp
WHERE apellido = 'Jiménez');
```

##### **4.4. SUBCONSULTA CON CUALQUIER NUMERO DE FILAS Y COLUMNAS**

**WHERE [NOT] EXISTS subconsulta**

**EJERCICIOS:**

-RESUELTOS: (SUBCONSULTAS)

-PROPUESTOS: (14,15)

## 5. UNION (INTERSECCIÓN y DIFERENCIA)

**UNION** => combina los resultados de dos consultas. Las filas duplicadas que aparecen se reducen a una fila única (elimina repetidos). Éste es su formato

```
SELECT COL1, COL2, ... FROM TABLA 1 WHERE condicion  
UNION  
SELECT COL1, COL2, ... FROM TABLA2 WHERE condicion;
```

**UNION ALL** => combina los resultados de dos consultas. Cualquier duplicación de filas que se dé en el resultado final aparecerá en la consulta (no elimina repetidos):

### Reglas para la utilización

- 1) Las columnas de las dos consultas se relacionan en orden, de izquierda a derecha
- 2) Los nombres de columna de la primera sentencia SELECT no tienen por qué ser los mismos que los nombres de la segunda
- 3) Las SELECT necesitan tener el mismo número de columnas
- 4) Los tipos de datos deben coincidir, aunque la longitud no tiene que ser la misma

### Ejemplos:

- Sacar el Apellido de la Plantilla que sea interno o trabaje en el turno de mañana.

```
SELECT Apellido  
      FROM Plantilla  
      WHERE turno= 'M'  
      union  
      SELECT Apellido  
      FROM Plantilla  
      WHERE LOWER (Funcion)= 'interno'
```

## INTERSECT

## MINUS

## **6. CONSULTAS CORRELACIONADAS**

## 7. AGRUPAMIENTO

Los datos seleccionados en la sentencia SELECT que lleva el GROUP BY deben ser: una constante, una función de grupo (SUM, COUNT,...), una columna expresada en el GROUP BY.

La cláusula GROUP BY sirve para calcular propiedades de uno o más conjuntos de filas (grupos). Igual que existe la condición de búsqueda WHERE para filas individuales, también hay una condición de búsqueda para grupos de filas: HAVING => se emplea para controlar cuál de los conjuntos de filas se visualiza y no puede existir sin un GROUP BY.

Cuando se usa GROUP BY, todas las columnas de la SELECT distintas a funciones de agregación deben aparecer en la cláusula GROUP BY. Sin embargo, pueden aparecer columnas en el GROUP BY y no estar en la SELECT.

- **AVG(n)**: calcula de media de n.

- **COUNT(\*| expr)**: cuenta el número de elementos del grupo:

*COUNT(A) - COUNT(DISTINCT A) - COUNT(\*)*

- **MAX(expr)**: cálcula el valor máximo.

- **MIN(expr)**: cálcula el valor mínimo.

- **SUM(expr)**: suma.

- **STDDEV(expr)**: desviación típica de la expresión sin tener en cuenta los valores nulos.

Ejemplos:

1º- Sacar el salario medio de cada hospital de los empleados de la plantilla, si dicha media es mayor que 200.000 pesetas.

```
SELECT AVG(Salario)
FROM Plantilla
GROUP BY Hospital_cod
HAVING AVG(Salario) > 200.000
```

2º- Sacar la suma de los salarios de cada una de las funciones, siempre y cuando dicha suma sea > 500.000 y el empleado no pertenece al hospital 18.

```
SELECT SUM(Salario), Funcion
FROM Plantilla
WHERE Hospital_cod <> 18
GROUP BY Funcion
HAVING SUM(Salario) > 500.000
```

3º- Seleccionar el salario máximo, mínimo y la diferencia entre ambos del personal de plantilla.

```
SELECT MAX(Salario), MIN(Salario), MAX(Salario) - MIN(Salario)
FROM Plantilla
```

4º- Número de empleos que existen.

```
SELECT COUNT (DISTINCT Función)
FROM Plantilla
```

## 8. EJEMPLOS.

Selección de todas las Enfermas en la Tabla Enfermo en que su Apellido está ordenado de forma ascendente:

```
SELECT *
  FROM Enfermo
 WHERE S='F'
ORDER BY Apellido ASC
```

### a) Operadores Aritméticos

Son: (\*, +, -, /)

```
SELECT (Salario *10) + comisión
  FROM Emp
 WHERE (salario + comisión) >1000
```

### b) Operadores para comparación de cadenas de caracteres

[NOT] LIKE : Permite utilizar caracteres especiales o comodines (% , \_)

% : se refiere a cualquier número de caractéres.  
\_ : se refiere a un sólo carácter.

Saca todas las direcciones que tengan una A.

```
SELECT Direccion
  FROM Enfermo
 WHERE Direccion LIKE '%A %'
```

### c) Comparadores lógicos

1.- [NOT] BETWEEN Valor1 AND Valor2 : establece un rango.

Sacar todos los enfermos, nacidos entre 16/09/45 y 21/05/60.

```
SELECT Apellido  
      FROM Enfermo  
     WHERE Fecha_na BETWEEN '16/09/45' AND '21/05/60'
```

2.- [NOT] IN (Lista de valores entre comas):

3.- IS [NOT] NULL

Sacar todos los empleados que tienen comisión:

```
SELECT Apellido  
      FROM Emp  
     WHERE comision IS NOT NULL
```

Si le quitamos el NOT, sacaría todos los enfermos que no cobran comisión.

4.- <oper-arit> ANY

Saca todos los empleados que no cobran el mayor salario.

```
SELECT Apellido, Salario  
      FROM Emp  
     WHERE Salario < ANY (SELECT Salario  
                           FROM Emp)
```

5.- <oper-arit> ALL

Sacar el empleado que más cobra.

```
SELECT Apellido, Salario  
      FROM Emp  
     WHERE Salario >= ALL (SELECT Salario  
                           FROM Emp)
```

6.- DISTINCT

Salen sólo las tuplas que son distintas. De dos tuplas iguales sólo deja una de ellas.

**SELECT[DISTINCT] Atributos**

Sacar de la tabla de plantilla funciones de las enfermeras ordenadas por el turno.

```
SELECT DISTINCT Funcion, Turnos  
      FROM Plantilla  
     WHERE Función='Enfermera'  
          ORDER BY Turno
```

¿Cuando <oper-arit> ALL<=> NOT IN?  
NOT IN

Cuando sea distinto a todos != ALL <=>

¿ANY <=> IN?      =ANY <=> IN

## 9. EJERCICIOS

### 9.1. RESUELTOS

#### 9.1.1. BASICOS

1. Seleccionar todos los datos de todos los empleados

```
SELECT *  
FROM EMP;
```

2. Seleccionar los siguientes datos de todos los empleados: - *número - nombre - salario - comisión*

```
SELECT EMPNO, ENAME, SAL, COMM  
FROM EMP;
```

3. Seleccionar los siguientes datos de todos los empleados: - *número departamento - nombre - trabajo - fecha ingreso*

```
SELECT DEPTNO, ENAME, JOB, HIREDATE  
FROM EMP;
```

4. Seleccionar los siguientes datos de todos los empleados: - *número - nombre - salario - comisión. - salario bruto SALBR (= SALARIO + COMMISSION)*

```
SELECT DEPTNO, ENAME, SAL, COMM, SAL + COMM AS SALBR  
FROM EMP;
```

### Clausulas: distintos, igual, no igual, mayor, menor, etc

5. Seleccionar los distintos trabajos (JOB) de todos los empleados

```
SELECT DISTINCT JOB  
FROM EMP;
```

6. Seleccionar los distintos departamentos (DEPTNO) de todos los empleados

```
SELECT DISTINCT DEPTNO  
FROM EMP;
```

7. Seleccionar todos los datos de los empleados del departamento 20

```
SELECT *  
FROM EMP  
WHERE DEPTNO = 20;
```

8. Seleccionar los siguientes datos de los empleados del departamento 10: *deptno - empno - ename - sal - comm*

```
SELECT DEPTNO, EMPNO, ENAME, SAL, COMM  
FROM EMP  
WHERE DEPTNO = 10;
```

9. Seleccionar todos los datos de los empleados que no pertenezcan al departamento 30

```
SELECT *  
FROM EMP  
WHERE DEPTNO != 30;
```

10. Seleccionar los siguientes datos de los empleados que no pertenezcan al departamento 10: - *deptno* - *empno* - *ename* - *sal* - *comm*

```
SELECT DEPTNO, EMPNO, ENAME, SAL, COMM  
FROM EMP  
WHERE DEPTNO != 10;
```

11. Seleccionar todos los datos de los empleados cuyo salario es superior que 1600

```
SELECT *  
FROM EMP  
WHERE SAL > 1600;
```

12. Seleccionar todos los datos de los empleados cuyo salario no es superior que 1600

```
SELECT *  
FROM EMP  
WHERE SAL <= 1600;
```

13. Seleccionar todos los datos de los empleados cuyo salario es superior o igual que 5000

```
SELECT *  
FROM EMP  
WHERE SAL >= 5000;
```

14. Seleccionar todos los datos de los empleados cuyo trabajo es GESTOR (MANAGER)

```
SELECT *  
FROM EMP  
WHERE JOB = 'MANAGER';
```

15. Seleccionar todos los datos de los empleados cuyo trabajo no es GESTOR (MANAGER)

```
SELECT *  
FROM EMP  
WHERE JOB != 'MANAGER';
```

16. Seleccionar todos los datos de los empleados cuyo trabajo es ANALISTA

```
SELECT *  
FROM EMP  
WHERE JOB = 'ANALYST';
```

17. Seleccionar todos los datos de los empleados cuyo trabajo no es ANALISTA

```
SELECT *  
FROM EMP  
WHERE JOB != 'ANALYST';
```

18. Seleccionar todos los datos de los empleados cuyo nombre es ALLEN

```
SELECT *  
FROM EMP  
WHERE ENAME = 'ALLEN';
```

19. Seleccionar todos los datos de los empleados cuyo nombre no es ALLEN

```
SELECT *  
FROM EMP  
WHERE ENAME != 'ALLEN';
```

### Clausula IN

20. Seleccionar todos los datos de los empleados cuyo trabajo sea uno de los siguientes:  
*PRESIDENTE, ANALISTA*

```
SELECT *
  FROM EMP
 WHERE JOB IN ('PRESIDENT', 'ANALYST');
```

21. Seleccionar todos los datos de los empleados cuyo trabajo no sea ninguno de los siguientes: *PRESIDENTE, ANALISTA*

```
SELECT *
  FROM EMP
 WHERE JOB NOT IN ('PRESIDENT', 'ANALYST');
```

22. Seleccionar los siguientes datos de los empleados que pertenezcan a los departamentos 10 y 30: - *deptno - ename - sal*

```
SELECT DEPTNO, ENAME, SAL
  FROM EMP
 WHERE DEPTNO IN (10,30);
```

23. Seleccionar todos los datos de los empleados cuyo departamento no sea ninguno de los siguientes: 10, 20

```
SELECT *
  FROM EMP
 WHERE DEPTNO NOT IN (10,20);
```

### Clausula BETWEEN

24. Seleccionar todos los datos de los empleados cuyo salario se encuentre entre 1100 y 3000 (ambos valores inclusive)

```
SELECT *
  FROM EMP
 WHERE SAL BETWEEN 1100 AND 3000;
```

25. Seleccionar todos los datos de los empleados cuyo salario no se encuentre entre 2000 y 3000 (ambos valores inclusive)

```
SELECT *
  FROM EMP
 WHERE SAL NOT BETWEEN 2000 AND 3000;
```

### Clausula LIKE

26. Seleccionar los siguientes datos de los empleados cuyo nombre comience por S:  
*empno, ename, hiredate, mgr*

```
SELECT EMPNO, ENAME, HIREDATE, MGR  
FROM EMP  
WHERE ENAME LIKE 'S%';
```

27. Seleccionar los siguientes datos de los empleados cuyo nombre comience por M:  
*deptno, ename, sal, comm*

```
SELECT DEPTNO, ENAME, SAL, COMM  
FROM EMP  
WHERE ENAME LIKE 'M%';
```

28. Seleccionar todos los datos de los empleados cuyo nombre no comience por W

```
SELECT *  
FROM EMP  
WHERE ENAME NOT LIKE 'W%';
```

29. Seleccionar todos los datos de los empleados cuyo nombre no comience por K

```
SELECT *  
FROM EMP  
WHERE ENAME NOT LIKE 'K%';
```

30. Seleccionar todos los datos de los empleados cuyo trabajo finalice por T

```
SELECT *  
FROM EMP  
WHERE JOB LIKE '%T';
```

31. Seleccionar todos los datos de los empleados cuyo trabajo no finalice por T

```
SELECT *  
FROM EMP  
WHERE JOB NOT LIKE '%T';
```

32. Seleccionar todos los datos de los empleados cuyo trabajo contenga una E en cualquier posición

```
SELECT *  
FROM EMP  
WHERE JOB LIKE "%E%";
```

33. Seleccionar todos los datos de los empleados cuyo trabajo no contenga una E (en ninguna posición)

```
SELECT *  
FROM EMP  
WHERE JOB NOT LIKE "%E%";
```

34. Seleccionar todos los datos de los empleados cuyo trabajo contenga una A y una L en ese orden

```
SELECT *  
FROM EMP  
WHERE JOB LIKE "%AL%";
```

35. Seleccionar todos los datos de los empleados cuyo trabajo contenga una *L* y una *A* en ese orden

```
SELECT *
FROM EMP
WHERE JOB LIKE "%LA%";
```

### Operadores aritméticos

36. Seleccionar los siguientes datos de todos los empleados: - *deptno*, *empno*, *sal*, *comm* y el salario anual ( salario por 12 ). La columna del salario se llamará *SALANUAL*.

```
SELECT DEPTNO, EMPNO, SAL, COMM, SAL*12 "SALANUAL"
FROM EMP;
```

37. Seleccionar los siguientes datos de todos los empleados: *empno*, *ename*, *sal*, y el salario que percibe diariamente. Se considerará que el mes es de 30 días.

```
SELECT EMPNO, ENAME, SAL, SAL/30 SALDIARIO
FROM EMP;
```

### Clausula AND

38. Seleccionar todos los datos de los empleados que pertenezcan al departamento 30, además su trabajo deberá el de vendedor

```
SELECT *
FROM EMP
WHERE DEPTNO = 30 AND JOB = 'SALESMAN';
```

39. Seleccionar todos los datos de los empleados cuyo trabajo sea el de vendedor, además su salario deberá superar el importe de 1500

```
SELECT *
FROM EMP
WHERE JOB = 'SALESMAN' AND SAL > 1500;
```

### Clausula OR

40. Seleccionar todos los datos de los empleados que pertenezcan al departamento 30 y su comisión es igual o mayor que cero. También se deberán seleccionar aquellos empleados que posean un salario inferior que 1000

```
SELECT *
FROM EMP
WHERE (DEPTNO = 30 AND COMM >= 0) OR (SAL < 1000);
```

41. Seleccionar todos los datos de los empleados que realicen el trabajo de empleado (CLERK) y que su salario sea mayor que 1000, también se seleccionarán aquellos empleados cuyo trabajo no sea de empleado (CLERK) y que tengan una comisión igual o mayor que cero.

```
SELECT *
FROM EMP
WHERE (JOB = 'CLERK' AND SAL > 1000) OR (JOB != 'CLERK' AND COMM >= 0);
```

### Clausula ORDER BY

42. Ordenar los datos de los empleados por nombre.

```
SELECT *
FROM EMP
ORDER BY ENAME;
```

43. Ordenar los datos de los empleados por fecha de ingreso de manera descendente.

```
SELECT *
FROM EMP
ORDER BY HIREDATE DESC;
```

44. Ordenar los datos de los empleados por número de departamento.

```
SELECT *
FROM EMP
ORDER BY DEPTNO;
```

45. Ordenar los datos de los empleados primero por departamento y trabajo.

```
SELECT *
FROM EMP
ORDER BY DEPTNO, JOB;
```

46. Ordenar los datos de los empleados primero por trabajo y fecha de ingreso descendente.

```
SELECT *
FROM EMP
ORDER BY JOB, HIREDATE DESC;
```

**9.1.2. COMBINACION DE TABLAS**

1. Seleccionar el nombre, el oficio y la localidad de los departamentos donde trabajan los 'ANALYST'

```
SELECT E.ENAME, E.JOB, D.LOC
FROM EMP E, DEPT D
WHERE E.JOB = 'ANALYST' AND E.DEPTNO = D.DEPTNO;
```

2. Obtener el número de empleado, nombre, salario, número de empleado de su jefe y nombre de su jefe

```
SELECT EMPLE.ENAME, EMPLE.SAL, EMPLE.MGR, JEFE.ENAME
FROM EMP EMPLE, EMP JEFE
WHERE EMPLE.MGR=JEFE.EMPNO;
```

3. Obtener el número de empleado, nombre, salario, número de empleado de su jefe, nombre de su jefe y nombre de departamento del empleado

```
SELECT EMPLE.ENAME, EMPLE.SAL, EMPLE.MGR, JEFE.ENAME, D.DNAME
FROM EMP EMPLE, EMP JEFE, DEPT D
WHERE EMPLE.MGR=JEFE.EMPNO AND D.DEPTNO=EMPLE.DEPTNO;
```

4. Seleccionar el apellido de los empleados que lleven más de 23 años trabajando en el departamento 'SALES'

```
SELECT E.ENAME
FROM EMP E, DEPT D
WHERE MONTHS_BETWEEN (SYSDATE, E.HIREDATE)/12 >23
AND D.DEPTNO=E.DEPTNO AND D.DNAME='SALES';
```

5. Seleccionar el nombre de todos los departamentos que tenemos en la tabla DEPT y los nombres y salarios de los empleados que tiene asignados cada departamento.

```
SELECT D.DNAME, E.ENAME, E.SAL
FROM DEPT D, EMP E
WHERE D.DEPTNO = E.DEPTNO
ORDER BY D.DNAME, E.ENAME DESC;
```

6. Visualizar todas las asignaturas que contengan tres letras 'o' en su interior y tengan alumnos matriculados de 'Madrid'

```
SELECT ASIG.NOMBRE
FROM ASIGNATURAS ASIG, ALUMNOS AL, NOTAS N
WHERE ASIG.NOMBRE LIKE "%o%o%o%" AND AL.POBLA ='Madrid' AND
ASIG.COD = N.COD AND N.DNI = AL.DNI;
```

7. Visualizar los nombres de alumnos que tengan una nota entre 7 y 8 en la asignatura de 'FOL'

```
SELECT AL.APENOM
FROM ALUMNOS AL, ASIGNATURAS ASIG, NOTAS N
WHERE N.NOTA BETWEEN 7 AND 8
AND ASIG.NOMBRE LIKE 'FOL'
AND ASIG.COD=N.COD AND N.DNI=AL.DNI;
```

**9.1.3. SUBCONSULTAS**

1. Presentar aquellos empleados cuyo salario es inferior a la media de los salarios de la empresa

```
SELECT *
FROM EMP
WHERE SAL < (SELECT AVG (SAL) FROM EMP);
```

2. Presentar aquellos empleados cuyo salario no es inferior a la media de los salarios de la empresa

```
SELECT *
FROM EMP
WHERE SAL >= (SELECT AVG (SAL) FROM EMP);
```

3. Presentar aquellos empleados cuyo salario coincide con el mayor de los salarios de la empresa

```
SELECT *
FROM EMP
WHERE SAL = (SELECT MAX (SAL) FROM EMP);
```

4. Presentar aquellos empleados cuyo salario coincide con la media de los salarios de la empresa

```
SELECT *
FROM EMP
WHERE SAL = (SELECT AVG (SAL) FROM EMP); => Ninguna fila
```

5. Presentar empleados cuyo salario supere al valor de tres veces la maxima comision

```
SELECT *
FROM EMP
WHERE SAL > 3*(SELECT MAX(COMM) FROM EMP);
```

6. Presentar empleados con un salario superior en un 50 % al del empleado CLARK

```
SELECT *
FROM EMP
WHERE SAL > 1.5*(SELECT SAL FROM EMP WHERE ENAME='CLARK');
```

7. Presentar los empleados con un salario igual o superior al menor de los salarios del departamento 30

```
SELECT *
FROM EMP
WHERE SAL >= (SELECT MIN(SAL)
                FROM EMP
                WHERE DEPTNO=30);
```

8. Mostrar los empleados (nombre, trabajo, salario y fecha de ingreso) que desempeñen el mismo trabajo que ALLEN o que tengan un salario mayor o igual que JONES

```
SELECT ENAME, JOB, SAL, HIREDATE
FROM EMP
WHERE JOB = (SELECT JOB
                FROM EMP
                WHERE ENAME = 'ALLEN')
OR SAL >= (SELECT SAL
                FROM EMP
                WHERE ENAME = 'JONES');
```

**9.1.4. AGRUPAMIENTO**

1. Para cada trabajo determinar:
- la cantidad de empleados que realizan dicho trabajo
  - la suma de los salarios brutos ( salario más comisión )

```
SELECT JOB, COUNT (*) NUM_EMPLEADOS, SUM (SAL+NVL(COMM,0))
SALARIOS FROM EMP
GROUP BY JOB;
```

2. Para cada departamento obtener el total de los salarios anuales.

```
SELECT DEPTNO, SUM (SAL*12) SALARIO_ANUAL
FROM EMP
GROUP BY DEPTNO;
```

3. Para cada departamento obtener:

- |                            |                           |                            |
|----------------------------|---------------------------|----------------------------|
| - la cantidad de empleados | - la media de salarios    | - el mayor de los salarios |
| - el menor de los salarios | - la suma de los salarios |                            |

```
SELECT DEPTNO, COUNT (EMPNO) NUM_EMPLEADOS, AVG (SAL)
MEDIA_SALARIOS, MAX (SAL) SALARIO_MAXIMO, MIN (SAL)
SALARIO_MINIMO, SUM (SAL) SUMA_SALARIOS
FROM EMP
GROUP BY DEPTNO;
```

4. Para cada trabajo, determinar qué departamentos y cuántos trabajadores lo realizan.

```
SELECT JOB, DEPTNO, COUNT (*) NUM_TRABAJADORES
FROM EMP
GROUP BY JOB, DEPTNO;
```

5. Para cada departamento, determinar cuales y cuántos trabajadores realizan esos trabajos

```
SELECT DEPTNO, JOB, COUNT (*) NUM_TRABAJADORES
FROM EMP
GROUP BY DEPTNO, JOB;
```

6. Para cada Jefe (manager) determinar la cantidad de empleados subordinados.

```
SELECT MGR, COUNT (*) EMPLE_SUBORD
FROM EMP
GROUP BY MGR;
```

7. Para cada Jefe (manager) determinar la media de salarios de sus empleados.

```
SELECT MGR, AVG (SAL) SALARIO_MEDIO
FROM EMP
GROUP BY MGR;
```

8. Para cada Jefe (Manager) determinar la media de salarios de sus empleados.  
Restricción: no se puede utilizar la función AVG.

```
SELECT MGR, (SUM(SAL)/COUNT(SAL)) SALARIO_MEDIO
FROM EMP
GROUP BY MGR;
```

**9.2. PROPUESTOS****9.2.1. Bloque - A****Base de datos COMERCIAL****ART**

Columna	Tipo	Nulo	Pred
<i>na</i>	char(2)	No	
desa	varchar(20)	Sí	NULL
color	varchar(10)	Sí	NULL
talla	int(2)	Sí	NULL

**PED**

Columna	Tipo	Nulo	Pred
<i>np</i>	char(2)	No	
<i>na</i>	char(2)	No	
<i>nf</i>	char(2)	No	
cantidad	int(3)	Sí	NULL

A1	retales	rojo	10
A2	paños	azul	10
A3	botones	negro	10

**FAB**

Columna	Tipo	Nulo	Pred
<i>nf</i>	char(2)	No	
nomf	varchar(15)	Sí	NULL
ciudadf	varchar(15)	Sí	NULL

P1	A2	F1	580
P2	A1	F1	200
P2	A2	F3	908
P4	A1	F1	236
P4	A2	F1	467
P4	A2	F2	456
P4	A3	F2	349
P5	A2	F1	567
P5	A2	F2	123

F1	RETASA	León
F2	PAÑASA	Madrid
F3	TEXTASA	Lisboa

**PRO**

Columna	Tipo	Nulo	Pred
<i>np</i>	char(2)	No	
nomp	varchar(15)	Sí	NULL
ciudadp	varchar(15)	Sí	NULL

P1	Macario	Palencia
P2	Salustio	León
P4	Endelecio	Zamora
P5	Constancio	León

- 1)- **Nombres** de las Fábricas situadas en *Madrid*.
- 2)- **Proveedores** que suministran a la Fábrica *F1*.
- 3)- **Proveedores** que suministran a la Fábrica *F1* el Artículo *A1*.
- 4)- **Nombres** de las Fábricas a las que suministra el Proveedor *P1*.
- 5)- **Colores** de los Artículos que suministra el proveedor *P1*.
- 6)- **Proveedores** que suministran Artículos *azules* a la Fábrica *F1*.
- 7)- **Artículos** suministrados a las Fábricas de *Madrid*.
- 8)- **Proveedores** que suministran algún Artículo *azul* a las Fábricas de *Madrid* o *Lisboa*.
- 9)- **Artículos** suministrados por Proveedores de *Zamora* a las Fábricas de *Madrid*.
- 10)- **Fábricas** abastecidas por algún Proveedor de distinta ciudad.
- 11)- **Parejas de ciudades** tales que un Proveedor de la 1<sup>a</sup> abastece a una Fábrica de la 2<sup>a</sup>.
- 12)- **Tripletas de valores <CIUDAD,NA,CIUDAD>** tales que un Proveedor de la 1<sup>a</sup> ciudad abastece el artículo NA a una Fábrica de la 2<sup>a</sup> ciudad.
- 13)- Idem, pero sin obtener las tripletas en las que coinciden ambas ciudades.
- 14)- **Fábricas** que usan al menos un Artículo suministrado por el Proveedor *P1*.
- 15)- Proveedores que suministran al menos un Artículo suministrado por al menos otro Proveedor que suministra al menos un Artículo azul.
- 16)- Proveedores que suministran a las Fábricas *F1* y *F2*.
- 17)- Fábricas que **no** son abastecidas de Artículos azules por Proveedores de *Madrid*.
- 18)- Fábricas que usan **sólo** Artículos que pueden ser suministrados por el Proveedor *P1*.
- 19)- Para cada Fábrica:
  - El número de proveedores que la suministran. -Las cantidades totales suministradas.
- 20)- De cada artículo obtener:
  - la cantidad total suministrada. - la cantidad media suministrada. - el mayor de los envíos.
- 21)- Artículos (NA) tales que ningún otro tiene *talla* más pequeña.
- 22)- Para cada proveedor, el número de envíos realizados.
- 23)- Nombre de los proveedores que realizan más de tres envíos.
- 24)- Artículos suministrados a todas las fábricas.
- 25)- Proveedores que suministran, al menos un mismo Artículo, a **todas** las Fábricas.
- 26)- Artículos que son suministrados a **todas** las Fábricas de *Madrid*.
- 27)- Fábricas que usan, al menos, **todos** los Artículos suministrados por el Proveedor *P1*.
- 28)- Fábricas abastecidas por **todos** los Proveedores que suministran algún Artículo de color azul.
- 29)- Artículo suministrado a más fábricas.
- 30)- Proveedores que realizan un número de envíos superior a la media.
- 31)- Fábricas que reciben un determinado artículo de un mayor número de proveedores.

9.2.2. Bloque - B

## Base de datos OBRAS

### CONDUCTORES

Columna	Tipo	Nulo	Pred
<b>CODC</b>	char(3)	No	
NOMBRE	char(25)	No	
LOCALIDAD	char(20)	Sí	NULL
CATEG	int(11)	Sí	NULL

### TRABAJOS

Columna	Tipo	Nulo	Pred
<b>CODC</b>	char(3)	No	
<b>CODM</b>	char(3)	No	
<b>CODP</b>	char(3)	No	
<b>FECHA</b>	date	No	
TIEMPO	int(11)	Sí	NULL

C01	José Sánchez	Arganda	18
C02	Manuel Díaz	Arganda	15
C03	Juan Pérez	Rivas	20
C04	Luis Ortiz	Arganda	18
C05	Javier Martín	Loeches	12
C06	Carmen López	Rivas	15

### MAQUINAS

Columna	Tipo	Nulo	Pred
<b>CODM</b>	char(3)	No	
NOMBRE	char(20)	No	
PRECIOHORA	int(11)	Sí	NULL

C01	M02	P02	14/09/2002	120
C01	M02	P04	20/09/2002	NULL
C01	M03	P04	18/09/2002	180
C02	M03	P01	10/09/2002	100
C02	M03	P01	21/09/2002	NULL
C02	M03	P02	17/09/2002	NULL
C02	M03	P03	15/09/2002	30
C03	M01	P02	11/09/2002	200
C03	M01	P04	16/09/2002	300
C04	M03	P02	13/09/2002	90
C05	M03	P02	12/09/2002	150
C05	M03	P04	19/09/2002	90

M01	Excavadora	15000
M02	Hormigonera	10000
M03	Volquete	11000
M04	Apisonadora	18000

### PROYECTOS

Columna	Tipo	Nulo	Pred
<b>CODP</b>	char(3)	No	
DESCRIP	char(25)	No	
LOCALIDAD	char(20)	Sí	NULL
CLIENTE	char(25)	Sí	NULL
TELEFONO	char(9)	Sí	NULL

P01	Garaje	Arganda	Felipe Sol	600111111
P02	Solado	Rivas	José Pérez	912222222
P03	Garaje	Arganda	Rosa López	666999666
P04	Techado	Loeches	José Pérez	913333333
P05	Buhardilla	Rivas	Ana Botijo	NULL

1. Obtener el nombre de los conductores con categoría 15.
2. Obtener la descripción de los proyectos en los que se haya realizado trabajos durante los días 11 al 15 de septiembre de 2002.
3. Obtener el nombre de los conductores que hayan trabajado con una Hormigonera, ordenados descendenteamente.
4. Obtener el nombre de los conductores que hayan trabajado con una Hormigonera en proyectos de Arganda.
5. Obtener el nombre de los conductores y descripción del proyecto, para aquellos conductores que hayan trabajado con una Hormigonera en proyectos de Arganda durante los días 12 al 17 de Septiembre.
6. Obtener los conductores que trabajan en los proyectos de José Pérez.
7. Obtener el nombre y localidad de los conductores que NO trabajan en los proyectos de José Pérez
8. Obtener todos los datos de los proyectos realizados en Rivas o que sean de un cliente llamado José.
9. Obtener los conductores que habiendo trabajado en algún proyecto, figuren sin horas trabajadas.
10. Obtener los empleados que tengan como apellido Pérez y hayan trabajado en proyectos de localidades diferentes a las suyas
11. Obtener el nombre de los conductores y la localidad del proyecto, para aquellos conductores que hayan trabajado con máquinas con precio hora comprendido entre 10000 y 15000 ptas.
12. Obtener el nombre y localidad de los conductores, y la localidad del proyecto para aquellos proyectos que sean de Rivas y en los que no se haya utilizado una máquina de tipo Excavadora o una máquina de tipo Hormigonera.
13. Obtener todos los datos de los proyectos, y para aquellos proyectos realizados el día 15 de Septiembre, además incluir el nombre y localidad de los conductores que hayan trabajado en dicho proyecto.
14. Obtener el nombre de los conductores y el nombre y localidad de los clientes, en los que se haya utilizado la máquina con precio hora más elevado.
15. Obtener los datos de los proyectos que siempre han utilizado la máquina de precio más bajo.
16. Obtener los proyectos en los que haya trabajado el conductor de categoría más alta menos dos puntos, con la máquina de precio hora más bajo.
17. Obtener por cada uno de los clientes el tiempo total empleado en sus proyectos.
18. Obtener por cada uno de los proyectos existentes en la BD, la descripción del proyecto, el cliente y el total a facturar en ptas y en euros. Ordenar el resultado por uno de los totales y por cliente.
19. Obtener para el proyecto que más se vaya a facturar la descripción del proyecto, el cliente y el total a facturar en Ptas. y en euros
20. Obtener los conductores que hayan trabajado en todos los proyectos de la localidad de Arganda.
21. Obtener el tiempo máximo dedicado a cada proyecto para aquellos proyectos en los que haya participado más de un conductor diferente.
22. Obtener el número de partes de trabajo, código del proyecto, descripción y cliente para aquél proyecto que figure con más partes de trabajo.
23. Obtener la localidad cuyos conductores (al menos uno) haya participado en más de dos proyectos diferentes.
24. Eliminar las tablas de la base de datos.

9.2.3. Bloque - C

# Base de datos AGENCIAMATRI

## HOMBRES

Columna	Tipo	Nulo	Pred
<i>nomh</i>	varchar(15)	No	
edad	int(2)	Sí	NULL

Ambrosio	53
Gumersindo	60
Lucio	50
Macario	22
Rodolfo	25
Segismundo	27
Sigiloso	50
Tiburcio	23

## MUJERES

Columna	Tipo	Nulo	Pred
<i>nomm</i>	varchar(15)	No	
edad	int(2)	Sí	NULL

Andrea	24
Cipriana	35
Clodomira	45
Fulgencia	21
Lucrecia	48
Pepita	34
Salustia	28

## MATRIMONIOS

Columna	Tipo	Nulo	Pred
<i>nomh</i>	varchar(15)	No	
<i>nomm</i>	varchar(15)	No	

Rodolfo	Andrea
Segismundo	Cipriana
Tiburcio	Fulgencia

## HSIM

Columna	Tipo	Nulo	Pred
<i>nomh</i>	varchar(15)	No	
<i>nomm</i>	varchar(15)	No	

Hombre cae Simpatico a la Mujer

Ambrosio	Fulgencia
Lucio	Andrea
Lucio	Salustia
Macario	Andrea
Macario	Salustia
Rodolfo	Andrea
Segismundo	Andrea
Sigiloso	Salustia
Tiburcio	Andrea
Tiburcio	Fulgencia
Tiburcio	Salustia

## MSIH

Columna	Tipo	Nulo	Pred
<i>nomh</i>	varchar(15)	No	
<i>nomm</i>	varchar(15)	No	

Mujer cae Simpatica al Hombre

Gumersindo	Fulgencia
Lucio	Andrea
Lucio	Cipriana
Lucio	Fulgencia
Lucio	Salustia
Macario	Andrea
Macario	Fulgencia
Macario	Salustia
Rodolfo	Andrea
Segismundo	Cipriana
Tiburcio	Fulgencia
Tiburcio	Lucrecia

- A. Parejas de hombres y mujeres que se caen mutuamente simpáticos.
- B. Parejas de hombres y mujeres que se caen mutuamente simpáticos, con edades entre 20 y 30 años y que no estén casados entre sí.
- C. Matrimonios en que ambos esposos se caen mutuamente simpáticos.
- D. Mujeres casadas y sufridas a quienes no cae simpático su marido.
- E. Hombres "raritos" a quienes no cae simpática ninguna mujer.
- F. Hombres y mujeres asociales a quienes no cae nadie simpático.
- G. Mujeres casadas que caen simpáticas a algún hombre.
- H. Hombres y mujeres que no caen bien a nadie.
- I. Hombres a quienes sólo caen simpáticas mujeres casadas.

9.2.4. Bloque - D

## Base de datos VIDECLUBS

### SOCIO

Columna	Tipo	Nulo	Pred
<i>aficionado</i>	varchar(20)	No	
<i>videoclub</i>	char(2)	No	

### GUSTA

Columna	Tipo	Nulo	Pred
<i>aficionado</i>	varchar(20)	No	
<i>pelicula</i>	varchar(20)	No	

César	V1
Hierro	V2
José Pérez	V2
José Pérez	V3
Raúl González	V1
Roberto Carlos	V1
Roberto Carlos	V2

César	LA PILLE
Hierro	PELIS
José Pérez	EL ESCANDALO
José Pérez	LA PILLE
Raúl González	EL FOLLON
Roberto Carlos	VAYA LIO
Roberto Carlos	Y AHORA QUE

### VIDEOCLUB

Columna	Tipo	Nulo	Pred
<i>videoclub</i>	char(2)	No	
<i>pelicula</i>	varchar(20)	No	

V1	EL ESCANDALO
V1	EL FOLLON
V2	PELIS
V3	LA PILLE
V3	VAYA LIO
V5	Y AHORA QUE

- I. Videoclubes que disponen de alguna película que le guste a José Pérez.
- II. Aficionados que son socios al menos de un videoclub que dispone de alguna película de su gusto.
- III. Aficionados que son socios solamente de videoclubes que disponen de alguna película de su gusto.
- IV. Aficionados que no son socios de ningún videoclub donde tengan alguna película de su gusto.