tisBDdocs

P05 Funciones y subconsultas

CONTENIDOS

Listas

Mediante el operador IN se puede buscar un determinado valor en una lista construida usando constantes.

```
expresión [NOT] IN (listaValores)
```

Descripción de las asignaturas FBD y DGBD.

```
select descripcion
from asignaturas
where codigo in ('FBD', 'DGBD')
```

descripcion
DISEÑO Y GESTION DE BASES DE DATOS
FUNDAMENTOS DE LAS BASES DE DATOS

Nombre de los profesores que no imparten HI, FBD o DGBD.

```
select nombre
from profesores p, imparte i
where p.dni = i.dni
      and asignatura not in ('HI', 'FBD', 'DGBD')
```

nombre
RAFAEL ROMERO

Fijémonos en que MANUEL PALOMAR, que no imparte ninguna de las asignaturas objeto de la búsqueda, tampoco aparece en la tabla resultado puesto que su dni no aparece en la tabla IMPARTE.

El operador IN también funciona con subconsultas, esto es, la lista se genera a partir de los propios datos de la base de datos.

Obtener todos los datos de los profesores que imparten alguna asignatura.

```
select * from profesores
where dni IN (select dni from imparte)
```

dni	nombre	categoria	ingreso
21111222	EVA GOMEZ	TEU	1993-10-01
21333444	RAFAEL ROMERO	ASO6	1992-06-16

O dicho de otra manera: "datos de los profesores cuyo dni aparece en la tabla imparte". Igual que antes, podemos pensar que se va recorriendo la tabla PROFESORES fila a fila y, en cada una de ellas, se pregunta si su DNI se encuentra en la lista generada a partir de la subconsulta. Si la respuesta es afirmativa, el profesor aparece como resultado y si no es así se ignora.

En este caso daría lo mismo procesar la orden

```
select distinct p.*
from profesores p, imparte i
where p.dni = i.dni
```

Se verá más clara la utilidad de este operador si preguntamos justo lo contrario.

Obtener todos los datos de los profesores que no imparten asignaturas.

```
select * from profesores
where dni NOT IN (select dni from imparte)
```

dni	nombre	categoria	ingreso
21222333	MANUEL PALOMAR	TEU	1989-06-16

Funciones

La lista completa de funciones disponibles en MySQL es mejor consultarla en su manual. Téngase en cuenta que cada SGBD puede proporcionar sus funciones que coincidirán o no con las que maneja MySQL.

Redondea 15.1297 a dos decimales

```
select round(15.1297,2) redondeo;
```

redondeo
15.13

Obviamente, para realizar y mostrar un cálculo como este no hace falta ninguna tabla.

Fechas y tiempo

La administración y manejo de valores temporales es la parte de los motores de base de datos menos estandarizada y con más diferencias entre uno y otro. Aparte, y junto con la codificación de caracteres y las características regionales (por ejemplo, formato de las fechas) forma parte de un conjunto de parámetros globales del sistema que el administrador del mismo debe comprender y tener presente dependiendo del medio en el que se vayan a ver los resultados de una consulta. Si fuera a formar parte de un conjunto de páginas web dinámicas no solo el servidor de base de datos debe estar correctamente configurado sino que debe tener en cuenta el servidor http, el del lenguaje huesped (php, por ejemplo) e incluso los clientes (navegadores, ordenadores, etc.). Es muy posible que la misma fecha, en SQL Developer, vista en la pestaña resultados o en la script output tenga formato diferente (la primera está controlada por el programa cliente y la segunda muestra los datos tal cual se los envía el servidor.

Puestas así las cosas, aquí se va a tratar la superficie de todo lo que se puede hacer con fechas y tiempos y siempre desde el punto de vista estrictamente de la consulta SQL. MySQL ofrece varios tipos de datos relacionados con el tiempo:

DATETIME '0000-00-00 00:00:00'
DATE '0000-00-00'
TIMESTAMP 00000000000000
TIME '00:00:00'
YEAR 0000

Lo que se muestra en la lista anterior da una idea de qué datos maneja cada tipo. En realidad, lo anterior es la relación de "valores cero" que pueden almacenar y que tienen carácter de valor por defecto o valor de prueba. Dejando de lado el tipo TIMESTAMP que

tiene unas propiedades y aplicaciones propias, todos los tipos están relacionados y difieren en cuanto a las limitaciones de almacenamiento y, más importante, de conversión automática de tipos.

De cadenas de caracteres, números y fechas y tiempos

Profesores que han ingresado antes de 1990.

```
select *
from profesores
where ingreso < '1990-01-01';
```

dni	nombre	categoria	ingreso
21222333	MANUEL PALOMAR	TEU	1989-06-16

Nótese que se está comparando un tipo date con una cadena de caracteres. Lo que ocurre es que MySQL analiza la cadena y determina si tiene un formato adecuado para el tipo de datos y la procesa si así es. De hecho, aunque la salida genérica de una fecha siempre es aaaa-mm-dd, la cadena de caracteres que usamos para la comparación asume cierta libertad de formato:

```
ingreso < '1990@01@01'
ingreso < '1990/01/01'
ingreso < '1990.01.01'
ingreso < '1990:01:01'
```

En todos los casos el resultado es idéntico al anterior. Lo que ya no funciona es

```
select *
from profesores
where ingreso < '01-01-1990'
```

```
0 rows selected
```

Y, además, no se genera ningún mensaje de error, simplemente, no muestra fila alguna en el resultado. Tampoco se comporta como nosotros esperaríamos

```
select *
from profesores
where ingreso < 1990-01-01;
```

```
0 rows selected
```

Pero sí de

```
select *
from profesores
where ingreso < 19900101;
```

dni	nombre	categoria	ingreso
21222333	MANUEL PALOMAR	TEU	1989-06-16

Otro aspecto en el que se da cierta libertad en es la forma del año:

```
select *
from profesores
where ingreso < '90-01-01';
```

dni	nombre	categoria	ingreso
21222333	MANUEL PALOMAR	TEU	1989-06-16

Ahora bien, hay que tener en cuenta que MySQL hace una interpretación de esa parte de la fecha de tal forma que años en el rango 00-69 los convierte en 2000-2069, y los 70-99 en 1970-1999. Si manejamos años anteriores o posteriores debemos usar los 4 caracteres.

Funciones de fecha y hora

En el manual de referencia de MySQL 5.0 puede consultarse la funcionalidad completa relativa al manejo del tiempo. Aquí solo expondremos algunas:

now(), curdate(), curtime()
date_format(), str_to_date()
day(), dayofweek(), dayname(), month(), year(), hour(), minute(), second()

Ahora

La función `now()` devuelve la fecha y hora del servidor en formato datetime. Las funciones `curdate()` y `curtime()` hacen lo mismo pero con la fecha y la hora respectivamente.

```
select now(), curdate(), curtime();
```

now()	curdate()	curtime()
2010-12-03 19:40:17	2010-12-03	19:40:17

Formato

Las funciones principales de formato podemos decir que son date_format() y str_to_date(). Una es la inversa de la otra: dateformat() transforma la fecha a un formato de texto determinado y str_to_date() una cadena de caracteres en un formato concreto a fecha. Las dos trabajan con 2 parámetros, una expresión y una cadena de formato.

La cadena de formato indica a la función que aspecto tiene o queremos que tenga (depende de si es una u otra función) el dato que le suministramos en el primer parámetro:

Profesores, con la fecha de ingreso en formato "dd/mm/aaaa".

```
select dni, nombre, date_format(ingreso, '%d/%m/%Y') ingreso from profesores;
```

dni	nombre	ingreso
21111222	EVA GOMEZ	01/10/1993
21222333	MANUEL PALOMAR	16/06/1989
21333444	RAFAEL ROMERO	16/06/1992

Profesores que han ingresado antes de 1/1/1990.

```
select *
from profesores
where ingreso < str_to_date('1/1/90','%d/%m/%y');.
```

dni	nombre	categoria	ingreso
21222333	MANUEL PALOMAR	TEU	1989-06-16

Así, la cadena de formato para la primera consulta representa "cómo queremos la salida" y la de la segunda "cómo está escrita la fecha que quiero comparar". La totalidad de los códigos posibles se puede consultar en la descripción de la función date_format() en http://dev.mysql.com/doc/refman/5.0/es/date-and-time-functions.html.

Extracción

Las otras funciones a las que vamos a prestar atención son las que extraen parte de la expresión temporal.

```
select day(ingreso) día, month(ingreso) mes, year(ingreso) año
from profesores
where nombre='EVA GOMEZ';
```

día	mes	año
1	10	1993

```
select dayname(ingreso) día, dayofweek(ingreso) ndía, monthname(ingreso) mes
from profesores
where nombre='EVA GOMEZ';
```

d	nd	m
Friday	6	October

```
select date_format(now(),'%Y%m%d -- %H:%i:%s') ahora;
```

ahora
20101203 -- 20:12:49

BDgite (GITE-11014-UA),
Departamento de Lenguajes y Sistemas Informáticos,
Universidad de Alicante