


# U.T.6.2

## Creación, supresión y modificación de tablas, vistas y otros objetos

 Views o Vistas

View: Los views son como tablas virtuales que contienen una búsqueda pre-armada muy utiles a la hora de generar reportes.

```
CREATE [OR REPLACE] [ALGORITHM = {UNDEFINED | MERGE | TEMPTABLE}] VIEW
nombre_vista [(columnas)] AS sentencia_select [WITH [CASCADED | LOCAL] CHECK
OPTION]
```

La ventaja de los views, es que están precompilados en el motor, lo que su acceso y ejecución se torna más rápido.

```
DROP VIEW IF EXISTS `NombreSchema`.`viewabono_control`;

CREATE VIEW `Nombreschema`.`nombreview` AS SELECT * FROM tabla;
```

Las vistas comparten el mismo espacio de nombre que las tablas, por lo que no pueden tener los mismo nombres

Las vistas pueden ser usadas en otros comandos SELECT con todas sus opciones

# VISTAS

- A veces, para obtener datos de varias tablas hay que construir una sentencia SELECT compleja y, si en otro momento necesitamos realizar esa misma consulta, hay que volver a escribir la sentencia SELECT. Las vistas solucionan este problema: mediante una consulta simple de una vista cabe la posibilidad de obtener datos de una consulta compleja. Una vista es una tabla lógica que permite acceder a la información de una o de varias tablas. No contiene información por sí misma, sino que su información está basada en la que contienen otras tablas, llamadas *tablas base*, y siempre refleja los datos de estas tablas; es, simplemente, una sentencia SQL.
- Una Vista actualizable puede aceptar sentencias UPDATE, DELETE. Algunas vistas pueden permitir INSERT. Los cambios de los datos se propagan automáticamente a la tabla base.

# VISTAS

- Es el Resultado de una consulta SQL sobre una o varias tablas
- También se llaman tablas virtuales
- Tienen la misma estructura que una tabla
- Bajo ciertas condiciones podemos insertar, actualizar, borrar y seleccionar datos
- Proporcionan un mecanismo de seguridad potente y flexible al ocultar partes de la base de datos a distintos usuarios y aplicaciones
- Permiten a los usuarios y aplicaciones acceder a los datos de forma personalizada según sus necesidades
- Permiten simplificar operaciones complejas sobre relaciones base al poder representarlas con un simple nombre

# VISTAS

- Una vista es una **tabla *virtual***. Desde el punto de vista del usuario, es indistinguible de una tabla de la base de datos.
- Se maneja como una tabla, con la diferencia de que no contiene datos, dado que el sistema únicamente almacena la definición de la vista.
- La sintaxis general de la sentencia DDL para definir una vista es la siguiente:

```
CREATE [OR REPLACE] VIEW nombre AS  
consulta  
[WITH [CASCADED|LOCAL] CHECK OPTION];
```

# VISTAS

- Los modificadores de la cláusula CHECK OPTION establecen el alcance de la comprobación:
  - CASCADED**: Si se extiende a aquellas vistas incluidas en la actual. Es la opción por defecto.
  - LOCAL**: Sólo se comprueba la vista actual.
- En la creación de vistas hay que tener en cuenta que algunas vistas son *Actualizables*, es decir, podemos emplear sentencias INSERT, UPDATE o DELETE para modificar las tablas subyacentes. Esto se permite en aquellas vistas que:
  - Proceden de una única tabla.
  - En cuya definición no aparece la cláusula GROUP BY.
  - No se ha establecido una cláusula de sólo lectura.
- Además, hay que tener en cuenta el alcance de los cambios, si se ha definido la vista con una cláusula CHECK OPTION.

# VISTAS

- En SQL, las instrucciones DCL básicas para la gestión de vistas nos facilitan las tareas de:
  - Creación:  
CREATE VIEW
  - Modificación:  
ALTER VIEW
  - Borrado:  
DROP VIEW
- Las vistas complejas contienen consultas que se definen sobre más de una tabla, agrupan filas usando las cláusulas GROUP BY o DISTINCT, y contienen llamadas a funciones.
- Se pueden crear vistas usando funciones, expresiones en columnas y consultas avanzadas, pero únicamente se podrán consultar estas vistas.

# VISTAS

- La cláusula CHECK OPTION, impide, por ejemplo insertar un registro que no cumpla la condición de la creación de la vista.
- Si se crea una vista para listar los usuarios que empiezan por l, no se permitirá insertar un usuario que no cumpla esa condición:

**create view vista1 as select \* from usuarios  
where nombre like 'l%' WITH CHECK OPTION;**

```
mysql> INSERT INTO VISTA1 VALUES(NULL,'MARTA','MAR',DEFAULT);  
ERROR 1369 (HY000): CHECK OPTION failed 'prueba.vista1'  
mysql>
```

# USOS DE LAS VISTAS

- Las vistas pueden ser usadas en lugar de definir restricciones utilizando por ejemplo NOT NULL, BETWEEN, etc.
- Pueden usarse en la reestructuración de tablas, ocultando o moviendo columnas.
- Permiten implementar seguridad en los datos al ocultar registros.



# MODIFICACIÓN DE DATOS EN VISTAS

- Algunas vistas son actualizables, se pueden emplear en sentencias update, delete o insert para modificar el contenido de las tablas subyacentes. Además las vistas reflejan instantáneamente los cambios producidos en las tablas.
- **Para que una vista sea actualizable, debe haber una relación uno a uno entre los registros de la vista y los registros de la tabla subyacente.**
- Existen además ciertas restricciones:
  - Que **no** incluyan **funciones de agregado**
  - Que **no** incluyan cláusulas **distinct**
  - Que **no** incluyan **subconsultas**
  - Que **no** usen **tablas temporales**
  - **No** usar cláusulas **group by** no having
  - **No** usar **uniones** ni **reuniones externas**
  - **No** usar **consultas correlacionadas**

# MODIFICACIÓN DE DATOS EN VISTAS

- Para el caso de reuniones internas (tipo INNER) podemos actualizar o insertar siempre y cuando los campos afectados sean únicamente los de una de las tablas implicadas en el JOIN.
- Para agregar registros mediante INSERT, es necesario que las columnas de la vista actualizable también cumplan los siguientes requisitos adicionales:
- No debe haber nombres duplicados entre las columnas de la vista
- La vista debe contemplar todas las columnas de la tabla que **no tengan indicado un valor por defecto**
- Las columnas de la vista deben ser referencias a columnas simples y **no columnas derivadas**

# MODIFICACIÓN DE DATOS EN VISTAS

- No podemos insertar registros en una vista conteniendo una combinación de columnas simples y derivadas, pero podemos actualizarla si actualizamos únicamente las columnas no derivadas.

- Ejemplo:

```
mysql> create view vista1 as select saldo, 2*saldo as dup  
from pp;
```

-- Se puede modificar el campo no derivado

```
mysql> update vista1 set saldo=100;
```

-- No el derivado:

```
mysql> update vista1 set dup=100;  
ERROR 1348 (HY000): Column 'dup' is not updatable  
mysql>
```

No se pueden insertar nuevos registros:

```
mysql> insert into vista1 values (199,123);  
ERROR 1471 (HY000): The target table vista1 of the INSERT is not insertable  
-into  
mysql>
```

# MODIFICACIÓN DE DATOS EN VISTAS

Crear una vista con las columnas nombre, emp\_no, salario y salario\*2 que contenga los empleados de emple3 que empiecen por 'P':

```
mysql> create or replace view vemple1 as select  
emp_no,nombre,salario,salario*2 as doble from emple3 where  
nombre like 'p%';
```

- Actualizar el salario del emp\_no=7788 a 7777:

```
mysql> update vemple1 set salario=7777 where emp_no=7788;
```

- Comprobar que no se puede actualizar el campo derivado:

```
mysql> update vemple1 set doble=7777;
```

```
ERROR 1348 (HY000): Column 'doble' is not updatable
```

- Comprobar que no se puede añadir nueva información a la vista:

```
mysql> insert vemple1 values(7777,'Pepe',1990,2000);
```

```
ERROR 1471 (HY000): The target table vemple1 of the INSERT is  
not insertae-into
```

# MODIFICACIÓN DE DATOS EN VISTAS

Ejemplo: Crear una vista vemple2 con los campos emp\_no, salario y salario\*2 con los empleados de emple3 que pertenezcan al departamento con mayor número de empleados:

```
mysql> create or replace view vemple2 as select emp_no,  
    salario, salario*2 from emple3 where dept_no IN (select  
    dept_no from emple3 group by dept_no having count(*) >=  
    ALL (select count(*) from emple3 group by dept_no));
```

Esta vista no podrá ser modificada ya que contiene subconsultas.

```
mysql> update vemple2 set salario=7777 where  
    emp_no=222;
```

```
ERROR 1288 (HY000): The target table vemple2 of the  
    UPDATE is not updatable
```

# Ejemplos vistas

Vista que contenga los amigos que no pertenezcan al departamento 30:

```
mysql> create or replace view vamigo1 as select * from amigos3 where  
    idamigo not in (select emp_no from emple3 where dept_no=30);
```

- Si se intenta insertar un registro que tiene como emp\_no uno correspondiente a un empleado del dept\_no =30:

**No se verá en la vista, pero si en la tabla original**

```
mysql> insert vamigo1 values(7777,'pepe, Gomez', '917776655',  
'ventas');
```

Query OK, 1 row affected (0.00 sec)

Sin embargo, se podrá insertar un registro que no pertenezca al dept\_no 30:

```
mysql> insert into vamigo1 values (7782,'Pepe, Rodriguez', '987676543',  
'profesor');
```

Query OK, 1 row affected (0.00 sec)

# Ejemplos vistas con check option

Vista que contenga los amigos que no pertenezcan al departamento 30:

```
mysql> create or replace view vamigo1 as select * from amigos3 where idamigo  
not in (select emp_no from emple3 where dept_no=30) with check option;
```

- Si se intenta insertar un registro que tiene como emp\_no uno correspondiente a un empleado del dept\_no =30, dará el siguiente error:

```
mysql> insert into vamigo1 values (7788,'Pepe, Rodriguez', '987676543',  
'profesor');
```

ERROR 1369 (HY000): CHECK OPTION failed 'empleados2.vamigo1'

**No se insertará en la vista, ni en la tabla original**

- Sin embargo, se podrá insertar un registro que no pertenezca al dept\_no 30:

```
mysql> insert into vamigo1 values (7782,'Pepe, Rodriguez', '987676543',  
'profesor');
```

- **Con la opción local:**

```
mysql> create or replace view vamigo2 as select * from vamigo1 where nombre  
not like 'p%' with local check option;
```

- Permitirá añadir un empleado del dept\_no 30, si no empieza por 'P' en la tabla base, aunque no en las vistas:

```
mysql> insert vamigo2 values(7521,'ana, Gomez','917776655','vendedor');
```

# Ejemplos vistas

- Si se elimina, se eliminará de la tabla y por lo tanto no se visualizará con la vista:

```
mysql> delete from vamigo1 where idamigo=7782;  
Query OK, 1 row affected (0.00 sec)
```

```
mysql> select * from vamigo1 where idamigo=7782;  
Empty set (0.00 sec)
```

```
mysql> select * from amigos3 where idamigo=7782;  
Empty set (0.00 sec)
```



# GESTION DE INDICES MYSQL

- Los índices son objetos que permiten un acceso más rápido a ciertos campos de una tabla.
- Una vez creados, su uso correcto es utilizando esos campos en las condiciones WHERE o HAVING de las consultas.
- La creación de índices permite optimizar la consultas.
- **Para crear un índice usamos:**
- Es posible crear un índice para varios campos a la vez:

```
CREATE INDEX nombre_indice ON Tabla  
(campo)
```

```
CREATE INDEX nombre_indice ON Tabla  
(campo1, campo2)
```

# GESTION DE INDICES MYSQL

- Es posible crear un índice para ciertos caracteres de un campo de tipo cadena:

```
CREATE INDEX nombre_indice ON Tabla (campo  
(numero caracteres))
```

- También podemos crear el índice con ALTER TABLE:

## **Para borra un índice:**

```
ALTER TABLE tabla ADD INDEX nombre_indice ON  
Tabla (campo)
```

```
DROP INDEX nombre_indice On tabla;
```

# Ejemplos de índices

- 1. Teniendo en cuenta que se consulta mucho por el campo nombre de departamentos, crear un índice inomdepto sobre ese campo en esa tabla:

```
CREATE INDEX inomdepto on departamentos(nombre);
```

- 2. Crear un índice inomapel sobre la tabla empleados para su nombre y apellidos:

```
CREATE INDEX inomapel ON empleados(nombre, apellidos);
```

- 3. Crear un índice ipuesto sobre la tabla empleados para su puesto con los 10 primeros caracteres:

```
CREATE INDEX ipuesto ON empleados(puesto(10));
```

- 4. Borra este último índice.

```
DROP INDEX ipuesto On empleados;
```

# RESUMEN

- Sólo son actualizables las vistas que proceden de una **única tabla y en cuya definición no aparezcan las cláusulas GROUP BY o HAVING.**

# ENLACES

- <http://uno-de-piera.com/introduccion-las-vistas-en-mysql/>
- [http://www.w3schools.com/sql/sql\\_create\\_index.asp](http://www.w3schools.com/sql/sql_create_index.asp)
- <http://manuales.guebs.com/mysql-5.0/views.html>
- <http://es.slideshare.net/duartoduar/vistas-en-mysql-34713140>
- <http://tallerdebasededatos.obolog.es/resumen-vistas-mysql-comandos-444806>
- <http://ftp.nchu.edu.tw/MySQL/doc/refman/5.0/es/create-index.html>
- <http://ftp.nchu.edu.tw/MySQL/doc/refman/5.0/es/create-index.html>
- <http://dev.mysql.com/doc/refman/5.0/es/connection-access.html>
- <http://dev.mysql.com/doc/refman/5.0/en/drop-user.html>
- <http://dev.mysql.com/doc/refman/5.0/en/create-user.html>
- <http://dev.mysql.com/doc/refman/5.0/en/grant.html>