Bases de Datos

Unidad 8:

Programación de bases de datos

Sesión 9

EVENTOS

Se introdujeron a partir de MySql 5.1.

Permiten la ejecución planificada (indicando hora de ejecución, intervalo, repeticiones, etc) de Procedimientos almacenados o conjuntos de sentencias SQL.

Proporcionan una gran funcionalidad a las bases de datos, permitiéndonos por ejemplo, borrar una tabla y cargarla con nuevos datos al cumplirse una condición determinada y de manera repetitiva.

Para poder trabajar con eventos hay que tener activado el planificador o Scheduler. Para ello tiene que estar a ON la variable global **event_sheduler**

Hay que modificar su valor en el archivo de configuración my.ini y reiniciar el servidor mysqld.

9.- Eventos

Sintaxis para crear un evento

```
CREATE EVENT    [IF NOT EXISTS]    event_name
  ON SCHEDULE schedule
  [ON COMPLETION [NOT] PRESERVE]
  [ENABLE | DISABLE | DISABLE ON SLAVE]
  [COMMENT 'comment']
  DO event_body;
```

schedule:

```
  AT timestamp [+ INTERVAL interval] ...
| EVERY interval
  [STARTS timestamp [+ INTERVAL interval] ...]
  [ENDS timestamp [+ INTERVAL interval] ...]
```

Interval:

```
  quantity {YEAR | QUARTER | MONTH | DAY | HOUR | MINUTE |
        WEEK | SECOND | YEAR_MONTH | DAY_HOUR | DAY_MINUTE |
        DAY_SECOND | HOUR_MINUTE | HOUR_SECOND | MINUTE_SECOND}
```

ON SCHEDULE

- Sirve para indicar cuando y con que frecuencia se producirá el evento.
- Presenta dos posibilidades. **La primera es**:
  - AT dato de tipo timestamp**.**
    - Se usa para un evento que sucede una única vez. NO PERIÓDICO.
    - Se llevará a cabo en la fecha y hora que marca TIMESTAMP (puede ser valor real o expresión compatible con el tipo)
    - CURRENT_TIMESTAMP. Indica fecha y hora actual (se ejecuta lo antes posible)
    - +INTERVAL intervalo. Me permite crear un evento del tipo anterior pero que sucederá dentro de un tiempo.
    - EJEMPLOS:
      - AT '2017-06-01 00:00:00'
      - AT CURRENT_TIMESTAMP +INTERVAL '2:10' MINUTE_SECOND (2 minutos y diez segundos después de ahora)
      - AT CURRENT_TIMESTAMP + INTERVAL 3 WEEK + INTERVAL 2 DAY (3 semanas y dos días a partir de ahora)

ON SCHEDULE

- **La segunda posibilidad es**:
  - EVERY interval
    - Se usa para repetir acciones a intervalos regulares.
    - Va seguido de un intervalo que puede usar lo mismo que indicamos para +INTERVAL, aunque esta última sintaxis no está permitida.
    - Puede llevar un STARTS + TIMESTAMP para indicar cuando deberá comenzar la opción a repetir y puede llevar +INTERVAL de la siguiente manera:
      - EVERY 3 MONTHS STARTS CURRENT_TIMESTAMP + INTERVAL 1 WEEK (cada 3 meses comenzando dentro de una semana a partir de ahora)
      - EVERY 2 WEEK STARTS CURRENT_TIEMSTAMP + INTERVAL '6:15' HOUR_MINUTE (cada 2 semanas y comienza 6 horas y quince minutos después de ahora)
    - Sin STARTS comienza a partir de la creación
    - Puede llevar ENDS. Indica cuando finaliza. En caso contrario se ejecuta de infinitamente.
      - EVERY 12 HOURS STARTS CURRENT_TIMESTAMP + INTERVAL 30 MINUTE ENDS CURRET_TIMESTAMP + INTERVAL 4 WEEK (cada 12 horas comenzando dentro de 30 minutos y acabando tras 4 semanas desde ahora)

Saber lo que ocurre

- SHOW EVENT
  - Permite ver los eventos que tenemos y cuando se ejecutarán

- SHOW PROCESSLIST
  - Permite ver los procesos abiertos que tenemos, así como su status en cada instante de tiempo

Ejemplo 1

- En la Base de Datos eBanca.
- Haz un evento que bonifique con 100 euros aquellas cuentas que se hayan dado de alta en el último mes.
- Para ello investiga por internet el funcionamiento de las funciones DATE_ADD() y NOW() de MySql.

Ejemplo 1. Solución

```
/*CREA UN EVENTO PARA BONIFICAR A LAS CUENTAS CREADAS DURANTE  EL ÚLTIMO MES*/
DELIMITER //
CREATE EVENT bonificacion
ON SCHEDULE AT CURRENT_TIMESTAMP + INTERVAL 1 MONTH
DO
      BEGIN
            UPDATE ebanca.cuenta SET saldo=saldo + 100
            WHERE fecha_creación between
                  date_add(now(),interval -1 month) and now();
      END;
//

DELIMITER ;

DECLARE V TIMESTAMP

SELECT DATE_SUB('2014-11-22 13:23:44.657', INTERVAL NOW() -1 MONTH);
```

Ejemplo 2

- En la Base de Datos revista nos vamos a crear una nueva tabla llamada Monográficos.
- La estructura es la siguiente:

```
CREATE TABLE  test.eMonografico(
     id SMALLINT(5) UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
    nombre_Mono VARCHAR(50),
     fecha_publica DATETIME NOT NULL DEFAULT  '0000-00-00 00:00:00'
    );
```

- Posteriormente creamos un evento que nos inserte datos dentro de esta tabla. Los datos son los siguientes:

```
INSERT INTO test.eMonografico VALUES
    (DEFAULT, 'BASES DE DATOS NATIVAS', '2012-02-25 01:20:15' ),
    (DEFAULT, 'DOM CON BD NATIVAS', '2014-06-25 01:20:15' ),
    (DEFAULT, 'INTERFACE Y REALIDAD VIRTUAL', '2016-05-22 10:20:15'),
    (DEFAULT, 'SCRATCH DEL MIT A LAS AULAS', '2016-05-25 11:20:15' );
```

Ejemplo 2. Continuación

- Vamos ahora a crearnos un evento que haga otra tabla como eMonográficos, pero esta vez la llamaremos eMonográficosHis. Los campos son los mismos que la anterior, dado que es su histórico.

- Por último tenemos que hacer un evento que borre de eMonograficos y los pase al histórico a aquellos monográficos que tengan más de un mes de antigüedad.

- NOTA: mirad función DATE_SUB()

SOLUCION EVENTOS

Ejemplo 2. Solución

```
/*CREO UN EVENTO QUE ME CREE LA TABLA INDICADA. ESTE SE LLEVA A EJECUCIÓN 2 MINUTOS DESPUÉS DE CREARLO. ESTA TABLA ALMACENARÁ DESPUÉS LOS MONOGRÁFICOS*/
DELIMITER //

CREATE EVENT eCrearMono
ON SCHEDULE AT CURRENT_TIMESTAMP + INTERVAL '2' MINUTE
DO
BEGIN
    /*TABLA PARA CREARLA MEDIANTE EVENTO*/
    CREATE TABLE  test.eMonografico(
        id SMALLINT(5) UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
        nombre_Mono VARCHAR(50),
        fecha_publica DATETIME NOT NULL DEFAULT  '0000-00-00 00:00:00’
    );
END;

//
delimiter ;
```

Ejemplo 2. Solución

```
/*CREO UN EVENTO UN MINUTO MÁS TARDE QUE EL DE CREACIÓN DE LA TABLA, PARA
CARGAR LOS DATOS QUE ME SERVIRAN PARA EL EJEMPLO*/
DELIMITER //

CREATE EVENT eCargaDatos
ON SCHEDULE AT CURRENT_TIMESTAMP + INTERVAL '4' MINUTE
DO
BEGIN
      /*CARGA DE LOS DATOS PARA REALIZAR EL EJERCICIO*/
      INSERT INTO test.eMonografico VALUES
            (DEFAULT, 'BASES DE DATOS NATIVAS', '2012-02-25 01:20:15' ),
            (DEFAULT, 'DOM CON BD NATIVAS', '2014-06-25 01:20:15' ),
            (DEFAULT, 'INTERFACE Y REALIDAD VIRTUAL', '2016-05-22 10:20:15' ),
            (DEFAULT, 'SCRATCH DEL MIT A LAS AULAS', '2016-05-25 11:20:15' );
END;
//

delimiter ;
```

Ejemplo 2. Solución

```
/*CREO UN EVENTO QUE CREE LA TABLA INDICADA. ESTE SE LLEVA A EJECUCIÓN 2 MINUTOS DESPUÉS DE INSERTAR LOS DATOS EN EL FICHERO DE MONOGRÁFICOS. SIRVE PARA HACER EL HISTORIAL*/
delimiter //

CREATE EVENT eCrearMonoHis
ON SCHEDULE AT CURRENT_TIMESTAMP + INTERVAL '6' MINUTE
DO
BEGIN
    /*TABLA PARA CREARLA MEDIANTE EVENTO*/
    CREATE TABLE  test.eMonograficoHis(
    id SMALLINT(5) UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
    nombre_Mono VARCHAR(50),
    fecha_publica DATETIME NOT NULL DEFAULT  '0000-00-00 00:00:00');
END;
//

delimiter ;
```

Ejemplo 2. Solución

```
/*CREO UN TRIGGER PARA QUE CUANDO BORRE SE CARGUEN EN EL HISTORICO AQUELLOS
QUE REGISTROS QUE SE HAN BORRADO*/
/*CREACIÓN TRIGGER DELETE*/
delimiter //

DROP TRIGGER IF EXISTS tr_Monografico_delete;
CREATE TRIGGER  tr_Monografico_delete  BEFORE    DELETE   ON test.eMonografico
FOR EACH ROW
BEGIN
      INSERT INTO test.eMonograficoHis VALUES
      (OLD.id,OLD.nombre_Mono,OLD.fecha_publica);
END;
//

delimiter ;
```

Ejemplo 2. Solución

```
/*EVENTO QUE SE ENCARGA DE MIRAR LOS ARTICULOS MÁS ANTIGUOS*/
delimiter //
CREATE EVENT archivo_noticias
ON SCHEDULE EVERY 1 MONTH STARTS CURRENT_TIMESTAMP + INTERVAL '8' MINUTE
DO
BEGIN
	INSERT INTO test.eMonograficoHis (
	SELECT * FROM test.eMonografico
		WHERE  fecha_publica <= DATE_SUB(CURRENT_DATE(),INTERVAL 30 DAY));
	DELETE FROM test.eMonografico  WHERE fecha_publica <=
		DATE_SUB(CURRENT_DATE(), INTERVAL 30 DAY);
END;
//

delimiter;
```