ANÁLISIS DE SQLite

**Álvaro Rodríguez Vila, Óscar Fariña Campelo,
Juan Sobral González, Ángel Rivas Nieto,
Alejandro Carballo Alonso**

BASES DE DATOS:
Usa sistema de tipos dinámicos, a diferencia de los demás que usan rígidos. Cualquier columna puede almacenar cualquier tipo de dato, pero algunas columnas pueden preferir usar un tipo en vez de otro. El orden de operaciones en una query también afecta las conversiones de tipos resultantes.

FUNCIONES DE FECHA Y HORA:
Todas las funciones de fecha y hora toman un valor de tiempo (usualmente string) como argumento y cero o más modificadores. Los modificadores ponen de izquierda a derecha (orden importante).

MANEJO DE NULLS EN SQLITE VS OTROS MOTORES DE BASES DE DATOS:
Se encarga de nulls de forma cómoda, pero los documentos no son claros sobre cómo debe hacerse. SQLite fue modificado para tratar null haciéndolos iguales en statements SELECT DISTINCT y operador UNION en un SELECT. Nulls siguen siendo distintos en una columna UNIQUE.

SOPORTE DE CLAVE FORÁNEA:
Las constraints de clave foránea se usan para forzar relaciones "EXISTS" entre tablas y están desactivadas por defecto. Para que no haya casos de filas huérfanas que no corresponden con ninguna fila de otra tabla a la que deberían referenciar, ya que esa pudo haber sido eliminada. Puede reforzarse el vínculo con "NOT NULL".

BLOBS INTERNOS VS EXTERNOS:
Para blobs (Binary Large Objects) más pequeños que 100KB es más eficiente almacenarlo directamente en la base de datos. Si son mayores, es mejor almacenarlos de forma fragmentada.

---------- **FAQ** ----------

(1) ¿Cómo crear un campo AUTOCINREMENTADO?
A diferencia de todos los demás motores de databases, si pones un numérico como clave primaria, SQLite lo autoincremente automáticamente.
Pregunta 2: Mientras que en MySQL debes concretar que un campo es de autoincremento, en SQLite no.

(2) ¿Qué tipo de datos soporta SQLite?
Los campos pueden ser almacenados como INTEGER, REAL, TEXT, BLOB o NULL.

(3) ¡SQLite me permite insertar un string a una columna de tipo integer!
Es una característica, no un bug. Los tipos de datos que asignes a una columna no restringen el tipo de dato que se pueden poner en esa columna.

(4) ¿Por qué SQLite no me permite usar '0' y '0.0' como la clave primaria en dos filas diferentes de la misma tabla?
Porque al compararlos numéricamente, estos tienen el mismo valor.

(5) ¿Pueden múltiples aplicaciones o múltiples instancias de la misma aplicaciones acceder a una misma base de datos al mismo tiempo?
Sí, múltiples personas pueden acceder al mismo tiempo a una misma base de datos, pero solo una persona puede alterar una tabla a la vez.

(9) ¿Cuál es el tamaño máximo de un VARCHAR en SQLite?
SQLite permite mantener el contenido sin perderlo, mientras que el resto truncan el valor. Realmente no hay un valor máximo.

(10) ¿SQLite tiene tipo BLOB?
SQLite te permite almacenar BLOBs en cualquier columna, incluso pueden ser claves primarias.

(11) ¿Cómo añado, elimino o renombro columnas desde una tabla ya existente en SQLite?
Para tareas simples puedes ejecutar ALTER TABLE, pero para cosas más complejas vas a tener que volver a hacer la tabla.

(12) He borrado un montón de datos pero el archivo de la base de datos no se hace más pequeña. ¿Es esto un bug?
SQLite guarda un espacio de disco para los datos, por lo que aunque los borres, el espacio seguirá ahí por si se añaden nuevos datos. No es un bug.

(13) ¿Puedo usar SQLite en mi producto comercial sin pagar los derechos de autor?
Sí, porque SQLite es de dominio público.

(14) ¿Cómo uso una string literal que contiene una comilla simple integrada?
Funciona igual que Pascal, tienes que usar una comilla de escape.

(18) Comparaciones de caracteres insensibles al caso Unicode no funciona.
SQLite compara únicamente con ASCII de manera insensible al caso. Para comparar todo lo relacionado con Unicode hay que proporcionar los métodos necesarios.

(19) INSERT es muy lento, solo puedo hacer unas pocas docenas de INSERT por segundo.
La velocidad depende de tu disco duro. (Solo tenemos en cuenta HDDs).

(20) He eliminado información importante de mi base de datos SQLite. ¿Cómo la puedo recuperar?
Más te vale tener una copia de seguridad, porque sino será casi imposible. Con muchísima suerte podrás recuperar trozos de datos string.

(22) ¿SQLite soporta claves foráneas?
Sí, pero tienes que activarlo tú, porque viene desactivado por defecto.

(24) Mi cláusula WHERE *column1="column1"* no funciona. Causa que cada fila de la tabla sea devuelta, en vez de solo las filas donde la column1 tenga el mismo valor que "column1".
Para comparar contenido tienes que usar comillas simples, no dobles. Si le pones comillas dobles es como si no pusieses comillas y lo trata como su fuese una booleana, por lo que será siempre true.

(26) El estándar de SQL requiere que un constraint UNIQUE sea forzado incluso si una o más columnas en el constraint son NULL, pero SQLite no hace esto. ¿Es un bug?
Un constraint UNIQUE está satisfecho si y SOLO si ninguna de las filas en una tabla tienen los mismos valores no NULL en columnas únicas.

(28) Mi query no devuelve el nombre de la columna que espero. ¿Es un bug?
No es un bug. Si las columnas de tu resultado se llaman como las cláusulas, entonces devuelve el identificador de la derecha como palabra clave, sino no.