SQLite 别名 SQLite 索引

SQLite 触发器（Trigger）

SQLite **触发器（Trigger）**是数据库的回调函数，它会在指定的数据库事件发生时自动执行/调用。以下是关于 SQLite 的触发器（Trigger）的要点：

- SQLite 的触发器（Trigger）可以指定在特定的数据库表发生 DELETE、INSERT 或 UPDATE 时触发，或在一个或多个指定表的列发生更新时触发。
- SQLite 只支持 FOR EACH ROW 触发器（Trigger），没有 FOR EACH STATEMENT 触发器（Trigger）。因此，明确指定 FOR EACH ROW 是可选的。
- WHEN 子句和触发器（Trigger）动作可能访问使用表单 **NEW.column-name** 和 **OLD.column-name** 的引用插入、删除或更新的行元素，其中 column-name 是从与触发器关联的表的列的名称。
- 如果提供 WHEN 子句，则只针对 WHEN 子句为真的指定行执行 SQL 语句。如果没有提供 WHEN 子句，则针对所有行执行 SQL 语句。
- BEFORE 或 AFTER 关键字决定何时执行触发器动作，决定是在关联行的插入、修改或删除之前或者之后执行触发器动作。
- 当触发器相关联的表删除时，自动删除触发器（Trigger）。
- 要修改的表必须存在于同一数据库中，作为触发器被附加的表或视图，且必须只使用 **tablename**，而不是 **database.tablename**。
- 一个特殊的 SQL 函数 RAISE() 可用于触发器程序内抛出异常。

语法

创建 **触发器（Trigger）** 的基本语法如下：

```
CREATE  TRIGGER trigger_name [BEFORE|AFTER] event_name
ON table_name
BEGIN
 -- Trigger logic goes here....
END;
```

在这里，**event_name** 可以是在所提到的表 **table_name** 上的 *INSERT*、*DELETE* 和 *UPDATE* 数据库操作。您可以在表名后选择指定 FOR EACH ROW。

以下是在 UPDATE 操作上在表的一个或多个指定列上创建触发器（Trigger）的语法：

```
CREATE  TRIGGER trigger_name [BEFORE|AFTER] UPDATE OF column_name
ON table_name
BEGIN
 -- Trigger logic goes here....
END;
```

实例

让我们假设一个情况，我们要为被插入到新创建的 COMPANY 表（如果已经存在，则删除重新创建）中的每一个记录保持审计试验：

```
sqlite> CREATE TABLE COMPANY(
   ID INT PRIMARY KEY     NOT NULL,
   NAME           TEXT    NOT NULL,
   AGE            INT     NOT NULL,
   ADDRESS        CHAR(50),
   SALARY         REAL
);
```

为了保持审计试验，我们将创建一个名为 AUDIT 的新表。每当 COMPANY 表中有一个新的记录项时，日志消息将被插入其中：

```
sqlite> CREATE TABLE AUDIT(
    EMP_ID INT NOT NULL,
    ENTRY_DATE TEXT NOT NULL
);
```

在这里，ID 是 AUDIT 记录的 ID，EMP_ID 是来自 COMPANY 表的 ID，DATE 将保持 COMPANY 中记录被创建时的时间戳。所以，现在让我们在 COMPANY 表上创建一个触发器，如下所示：

```
sqlite> CREATE TRIGGER audit_log AFTER INSERT
ON COMPANY
BEGIN
   INSERT INTO AUDIT(EMP_ID, ENTRY_DATE) VALUES (new.ID, datetime('now'));
END;
```

现在，我们将开始在 COMPANY 表中插入记录，这将导致在 AUDIT 表中创建一个审计日志记录。因此，让我们在 COMPANY 表中创建一个记录，如下所示：

```
sqlite> INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY)
VALUES (1, 'Paul', 32, 'California', 20000.00 );
```

这将在 COMPANY 表中创建如下一个记录：

```
ID          NAME        AGE         ADDRESS     SALARY
----------  ----------  ----------  ----------  ----------
1           Paul        32          California  20000.0
```

同时，将在 AUDIT 表中创建一个记录。这个纪录是触发器的结果，这是我们在 COMPANY 表上的 INSERT 操作上创建的触发器（Trigger）。类似的，可以根据需要在 UPDATE 和 DELETE 操作上创建触发器（Trigger）。

```
EMP_ID      ENTRY_DATE
----------  -------------------
1           2013-04-05 06:26:00
```

列出触发器（TRIGGERS）

您可以从 **sqlite_master** 表中列出所有触发器，如下所示：

```
sqlite> SELECT name FROM sqlite_master
WHERE type = 'trigger';
```

上面的 SQLite 语句只会列出一个条目，如下：

```
name
----------
audit_log
```

如果您想要列出特定表上的触发器，则使用 AND 子句连接表名，如下所示：

```
sqlite> SELECT name FROM sqlite_master
WHERE type = 'trigger' AND tbl_name = 'COMPANY';
```

上面的 SQLite 语句只会列出一个条目，如下：

```
name
----------
audit_log
```

删除触发器（TRIGGERS）

下面是 DROP 命令，可用于删除已有的触发器：

```
sqlite> DROP TRIGGER trigger_name;
```

SQLite 别名 SQLite 索引

1 篇笔记

写笔记

上面没有关于 for each row 和 when 的实例，这里补充一下。

for each row 是操作语句每影响到一行的时候就触发一次，也就是删了 10 行就触发 10 次，而 for each state 一条操作语句就触发一次，有时没有被影响的行也执行。sqlite 只实现了 for each row 的触发。when 和 for each row 用法是这样的：

```
CREATE TRIGGER trigger_name
AFTER UPDATE OF id ON table_1
FOR EACH ROW
WHEN new.id>30
BEGIN
UPDATE table_2 SET id=new.id WHERE table_2.id=old.id;
END;
```

上面的触发器在 table_1 改 id 的时候如果新的 id>30 就把 表table_2 中和表table_1 id 相等的行一起改为新的 id

absin 1年前 (2017-10-30)