lsqlthw11

Robert A. Stevens

March 23, 2016

http://sql.learncodethehardway.org/book/

Learn SQL The Hard Way

http://sql.learncodethehardway.org/book/ex11.html

Exercise 11: Replacing Data

I'm going to show you an alternative way to insert data which helps with atomic replacement of rows. You don't necessarily need this too often, but it does help if you're having to replace whole records and don't want to do a more complicated UPDATE without resorting to transactions.

In this situation, I want to replace my record with another guy but keep the unique id. Problem is I'd have to either do a DELETE/INSERT in a transaction to make it atomic, or I'd need to do a full UPDATE.

Another simpler way to do it is to use the REPLACE command, or add it as a modifier to INSERT. Here's some SQL where I first fail to insert the new record, then I use these two forms of REPLACE to do it:

```
/* This should fail because 0 is already taken. */
INSERT INTO person (id, first_name, last_name, age)
    VALUES (0, 'Frank', 'Smith', 100);

/* We can force it by doing an INSERT OR REPLACE. */
INSERT OR REPLACE INTO person (id, first_name, last_name, age)
    VALUES (0, 'Frank', 'Smith', 100);

SELECT * FROM person;

/* And shorthand for that is just REPLACE. */
REPLACE INTO person (id, first_name, last_name, age)
    VALUES (0, 'Zed', 'Shaw', 37);

/* Now you can see I'm back. */
SELECT * FROM person;
```

What You Should See

In this exercise I'm going to enter these commands in the sqlite3 console rather than run them from the command line:

```
sqlite> /* This should fail because 0 is already taken. */
sqlite> INSERT INTO person (id, first_name, last_name, age)
   ...>     VALUES (0, 'Frank', 'Smith', 100);
Error: PRIMARY KEY must be unique
```

```
sqlite>
sqlite> /* We can force it by doing an INSERT OR REPLACE. */
sqlite> INSERT OR REPLACE INTO person (id, first_name, last_name, age)
   ...>     VALUES (0, 'Frank', 'Smith', 100);
sqlite>
sqlite> SELECT * FROM person;
0|Frank|Smith|100
sqlite>
sqlite> /* And shorthand for that is just REPLACE. */
sqlite> REPLACE INTO person (id, first_name, last_name, age)
   ...>     VALUES (0, 'Zed', 'Shaw', 37);
sqlite>
sqlite> /* Now you can see I'm back. */
sqlite> SELECT * FROM person;
0|Zed|Shaw|37
sqlite>
sqlite>
```

You can see on line 4 that I get an "Error: PRIMARY KEY must be unique" because the record has the id 0. I want to do a combination DELETE/INSERT with the new record, so then I do an INSERT OR REPLACE next. After that I put the record back with REPLACE which is shorthand for INSERT OR REPLACE.

Extra Credit

- Go to the SQLite3 INSERT page and look at the other INSERT OR options you have.
- Practice REPLACE by replacing the Unicorn with a pet Parrot.

Portability Notes

Some databases might not have the REPLACE command or even the INSERT OR REPLACE syntax. In that case, you'll just have to wait until I show you transactions.