Updating and Deleting Records

We'll learn how to update and delete records in SQLite using Python

Being able to update your database records is key to keeping your data accurate. If you can't update, then your data will become out of date and pretty useless very quickly. Sometimes you will need to delete rows from your data too. We'll be covering both of those topics in this section. First, let's do an update!

```
import sqlite3

conn = sqlite3.connect("mydatabase.db")
cursor = conn.cursor()

# Show rows BEFORE update
cursor.execute("SELECT * from albums WHERE title='Glow';")
print(cursor.fetchall())

sql = """
UPDATE albums
SET artist = 'John Doe'
WHERE artist = 'Andy Hunter'
"""
cursor.execute(sql)
conn.commit()

# Show rows AFTER update
cursor.execute("SELECT * from albums WHERE title='Glow';")
print(cursor.fetchall())
```

Here we use SQL's **UPDATE** command to update out albums table. You can use **SET** to change a field, so in this case we change the artist field to be *John Doe* in any record **WHERE** the artist field is set to *Andy Hunter*. Wasn't that easy? Note that if you don't commit the changes, then your changes won't be written out to the database. The **DELETE** command is almost as easy. Let's check that out!

```
import sqlite3

conn = sqlite3.connect("mydatabase.db")
cursor = conn.cursor()

sql = """
DELETE FROM albums
WHERE artist = 'John Doe'
"""
cursor.execute(sql)
conn.commit()

# Show rows AFTER delete. No row will be shown
cursor.execute("SELECT * from albums WHERE title='Glow';")
print(cursor.fetchall())
```

Deleting is even easier than updating. The SQL is only 2 lines! In this case, all we had to do was tell SQLite which table to delete from (albums) and which records to delete using the WHERE clause. Thus it looked for any records that had “John Doe” in its artist field and deleted it.