CRUD WITH SQL

CRUD

- **C**reate
- **R**ead
- **U**pdate
- **D**elete

OUR EXAMPLE : A DOCTORS TABLE

TABLE SCHEMA

```
CREATE TABLE `doctors` (
  `id`  INTEGER PRIMARY KEY AUTOINCREMENT,
  `name` TEXT,
  `age` INTEGER,
  `specialty` TEXT
);
```

SAMPLE DATA

```
sqlite> .mode column
sqlite> .headers on
sqlite> SELECT * FROM doctors;
id          name        age         specialty
----------  ----------  ----------  ----------------
1           John Smith  39          Anesthesiologist
2           Emma Reale  31          Cardiologist
```

You can download 02_sql_crud_doctors.db

READ

SQL keyword: **`SELECT`**

`SELECT` keyword to **fetch records** from the DB.

SELECTING ALL THE ROWS

```
SELECT * FROM doctors
```

SELECTING A SPECIFIC ROW

```
SELECT * FROM doctors WHERE id = 2
```

No quotes around 2 as it's a number.

CREATE

SQL keyword: **`INSERT`**

`INSERT` keyword to **add records** to a table.

```
INSERT INTO doctors (name, age, specialty)
VALUES ('Dr. Dolladille', 45, 'Dentist')
```

inserts a new record:

```
sqlite> SELECT * FROM doctors;
id          name          age         specialty
----------  ----------  ----------  ----------------
1           John Smith  39          Anesthesiologist
2           Emma Reale  31          Cardiologist
3           Dr. Dollad  45          Dentist
```

AUTOINCREMENT

Remember the schema definition:

```
CREATE TABLE `doctors` (
  `id`  INTEGER PRIMARY KEY AUTOINCREMENT,
  `name` TEXT,
  `age` INTEGER,
  `specialty` TEXT
);
```

LET THE DB HANDLE `ID`

Ensure that all records have a different id.

UPDATE

SQL keyword: **`UPDATE`**

`UPDATE` keyword to **update records** in a table.

UPDATE ONE RECORD

```
UPDATE doctors SET age = 40, name = 'John Smith' WHERE id = 1
```

DELETE

SQL keyword: **`DELETE`**

`DELETE` keyword to **delete records** from a table.

DELETE A SINGLE RECORD

The most used

```
DELETE FROM doctors WHERE id = 32
```

DELETE EVERY RECORD

```
DELETE FROM doctors
```

Deletes every record in the `doctors` table.

SQLITE AND RUBY

Some tips for the exercise

MODEL

We will use a model class (the M in MVC).

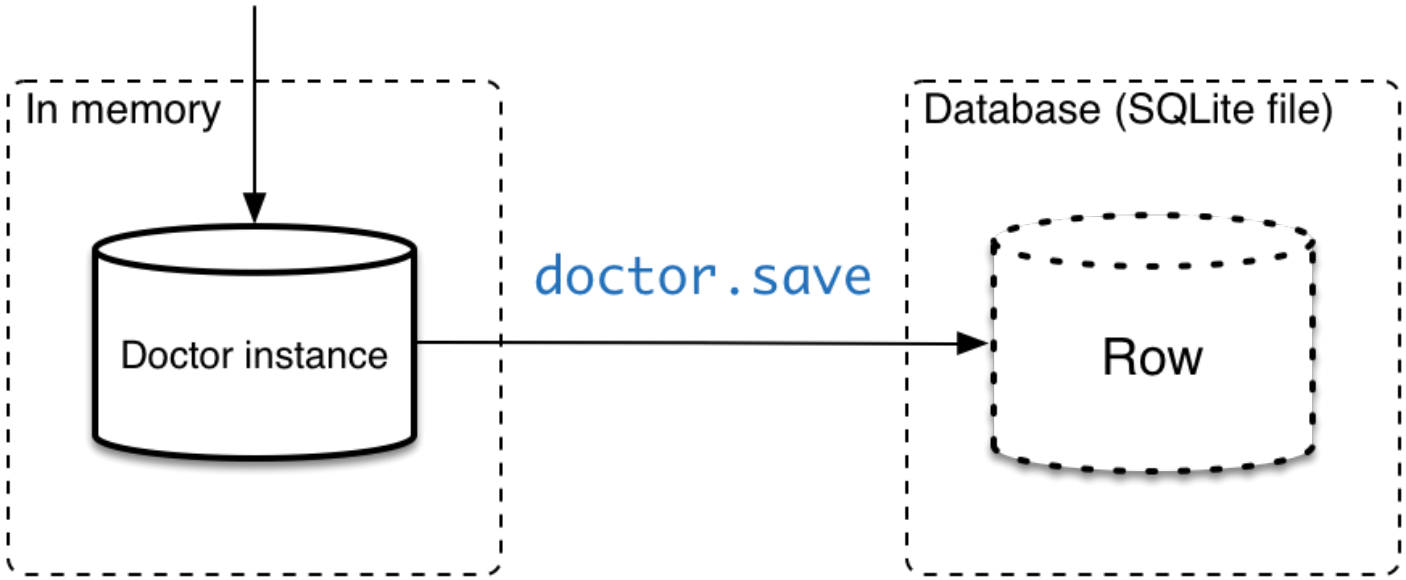
```ruby
# app/models/doctor.rb
class Doctor
  attr_reader :id

  def initialize(attributes = {})
    @id = attributes[:id]
    # TODO: store other attributes as instanced variable (exe
  end
end
```

Example

```ruby
doctor = Doctor.new(name: 'John', age: 42)
doctor.id
# => nil
```

IN-MEMORY VS STORED IN DATABASE

```
doctor = Doctor.new(name: "John")
```



You'll have to write the `save` method in your model.

DB CONNECTION

For today, we'll provide a global variable named `DB`

```
require 'sqlite3'
DB = SQLite3::Database.new("db/doctors.db")
```

Then you can use it in the model to run queries:

```
rows = DB.execute('SELECT * FROM doctors')
```

WHAT ABOUT THE ID?

Not your job to set it. Your job is to retrieve it and update the `@id` accordingly upon first save.

```
DB.execute("INSERT INTO doctors (name, age) VALUES ('John', 4
DB.last_insert_row_id
```

SOMETHING USEFUL...

```
doctors = DB.execute("SELECT name, age FROM doctors")
# => [
#      [ "John Smith", 39 ],
#      [ "Emma Reale", 31 ]
#    ]
```

```
DB.results_as_hash = true
doctors = DB.execute("SELECT name, age FROM doctors")
# => [
#      { "name" => "John Smith", "age" => 39 , 0 => "John Sm
#      { "name" => "Emma Reale", "age" => 31 , 0 => "Emma Rea
#    ]
```

```
doctor = doctors.first
name = doctor["name"]
age = doctor["age"]
puts "Doctor #{name} is #{age} years old"
```