JÖNKÖPING UNIVERSITY

School of Engineering

SQLITE

Peter Larsson-Green
Jönköping University
Autumn 2018

SQLITE

A very simple relational database.

- The entire database is stored in a single file.
- No users.
- Runs as part of the application.

Different SQLite managers exists.

- DB Browser for SQLite http://sqlitebrowser.org/

SQL CREATING TABLES

```
CREATE TABLE tableName (
   columnA datatype,
   columnB datatype,
   ...
)
```

```
CREATE TABLE Humans (
   Name TEXT,
   Age INTEGER,
   City TEXT
)
```

Some available datatypes:

- INTEGER
- REAL
- TEXT

Name	Age	City
Alice	10	Atlanta
Belle	15	Buenos Aires
Chloe Clair	20	Cairo

Table: Humans

Database: Test

SQL CREATING TABLES

```
CREATE TABLE IF NOT EXISTS tableName (
   columnA datatype,
   columnB datatype,
   ...
)
```

```
CREATE TABLE IF NOT EXISTS Humans (
   Name TEXT,
   Age INTEGER,
   City TEXT
)
```

Name	Age	City
Alice	10	Atlanta
Belle	15	Buenos Aires
Chloe Clair	20	Cairo

Table: Humans

Database: Test

SQL INSERTING DATA

```
INSERT INTO tableName (columnA, columnB, ...)
VALUES (valueA, valueB, ...)
```

```
INSERT INTO Humans (Name, Age, City)
VALUES ("Alice", 10, "Atlanta")
```

```
INSERT INTO Humans (Name, Age, City)
VALUES ("Belle", 15, "Buenos Aires")
```

```
INSERT INTO Humans (Name, Age, City)
VALUES ("Chloe Clair", 20, "Cairo")
```

Name	Age	City
Alice	10	Atlanta
Belle	15	Buenos Aires
Chloe Clair	20	Cairo

Table: Humans

Database: Test

SQL RETRIEVING DATA

```
SELECT * FROM tableName
```

```
SELECT * FROM Humans
```

Name	Age	City
Alice	10	Atlanta
Belle	15	Buenos Aires
Chloe Clair	20	Cairo

Name	Age	City
Alice	10	Atlanta
Belle	15	Buenos Aires
Chloe Clair	20	Cairo

Table: Humans

Database: Test

SQL RETRIEVING DATA

```
SELECT columnA, columnB, ... FROM tableName
```

```
SELECT Name, Age FROM Humans
```

Name	Age
Alice	10
Belle	15
Chloe Clair	20

Name	Age	City
Alice	10	Atlanta
Belle	15	Buenos Aires
Chloe Clair	20	Cairo

Table: Humans

Database: Test

SQL RETRIEVING DATA

```
SELECT columnA, columnB, ... FROM tableName
WHERE columnX op value
```

```
SELECT Name FROM Humans WHERE
Age >= 15
```

Some available op:

- < <= > >= == !=
- AND OR

Name
Belle
Chloe Clair

Name	Age	City
Alice	10	Atlanta
Belle	15	Buenos Aires
Chloe Clair	20	Cairo

Table: Humans

Database: Test

SQL RETRIEVING DATA

```
SELECT columnA, columnB, ... FROM tableName
WHERE columnX op value ORDER BY columnY sort
```

```
SELECT Name FROM Humans WHERE
Age < 20 ORDER BY Age DESC
```

Name
Belle
Alice

Name	Age	City
Alice	10	Atlanta
Belle	15	Buenos Aires
Chloe Clair	20	Cairo

Table: Humans

Database: Test

Available `sort`:

- `DESC` (Descending)
- `ASC` (Ascending)

SQL UPDATING DATA

```
UPDATE tableName SET columnA = valueA, columnB = valueB, ...
WHERE columnX op value
```

```
UPDATE Humans SET Age = 10
WHERE Name == "Alice"
```

Name	Age	City
Alice	5	Atlanta
Belle	15	Buenos Aires
Chloe Clair	20	Cairo

Table: Humans

Database: Test

SQL DELETING DATA

```
DELETE FROM tableName
WHERE columnX op value
```

```
DELETE FROM Humans
WHERE Name == "Alice"
```

Name	Age	City
Alice	10	Atlanta
Belle	15	Buenos Aires
Chloe Clair	20	Cairo

Table: Humans

Database: Test

SQL PRIMARY KEY

```
CREATE TABLE Humans (
   Id INTEGER PRIMARY KEY AUTOINCREMENT,
   Name TEXT,
   Age INTEGER
)
```

Id	Name	Age
1	Alice	10
2	Alice	10
3	Chloe Clair	20

Table: Humans

Database: Test

SQL UNIQUE CONSTRAINTS

```
CREATE TABLE Accounts (
   Id INTEGER PRIMARY KEY AUTOINCREMENT,
   Name TEXT UNIQUE,
   Age INTEGER
)
```

Id	Name	Age
1	Alice	10
2	Bob	15
3	Chloe Clair	20

Table: Accounts

Database: Test

SQL FOREIGN KEYS

```
CREATE TABLE Humans (
  Id INTEGER PRIMARY KEY AUTOINCREMENT,
  Name TEXT UNIQUE,
  Age INTEGER
)
```

Id	Name	Age
1	Alice	10
2	Bob	15
3	Chloe Clair	20

Table: Humans

Id	Name	HumanId
1	Doggy	1
2	Catty	2
3	Fishy	1

Table: Pets

Database: Test

```
CREATE TABLE Pets (
  Id INTEGER PRIMARY KEY AUTOINCREMENT,
  Name TEXT,
  HumanId INTEGER,
  FOREIGN KEY(HumanId) REFERENCES Humans(Id) ON DELETE CASCADE
)
```

PRACTICAL DEMONSTRATION

Go through DB Browser for SQLite.