SQLite

Relational Database

- A `database` is an organised collection of structured information, or data, typically stored electronically in a computer system.
- For examples, all students and lecturers are stored in databases.
- Data in a database are stored in tables. For example, a table called students will contain the details of all the students in the university or a particular school.

- `Relational database:` This is an approach where we consider our collections of information about objects as **Tables** with columns and rows. A database can have many tables. Relational database technology provides the most efficient and flexible way to access structured information.

- `Object-oriented databases`. Information in an object-oriented database is represented in the form of objects, as in object-oriented programming.

Relational Database

You will notice that more than one student takes the same class. In most databases you will find repetitive data such as this. To make database work more efficiently, the repeated data is often stored in a separate table. In this case there is a **class** table which would store all the information about each class to save having to repeat all the class details for each student.

Table: students

ID	name	class	grade
1899877D	Mary	Python	67
2223998M	John	Maths	34
2348990M	Anne	Python	70

Table: class

class	lecturer
Python	Jack
Maths	Laurie
Java	Joe

By splitting the data into two tables, if we need to update the lecturer, it will only need to be updated once rather than updating it several times, which would have happened if it was all stored in one table.

This is known as **one-to-many** relationship as on class can have many students taking it.

Primary Keys

A `primary key` is the field (usually first one) in each table that stores unique identifier for that record.

- For example, in the students table, the primary key will be the `ID` column and in the class table, the primary key is the `class` column.

- Each item in the supermarket has a bar-code, The bar code represents the product-code. The product-code is the key of the item in the database.

- Your telephone has a unique number which no-one else has. This is the key of your phone's record held by your service provider

- When creating a table, you need to identify the following for each field:
  - The name of the field (field name cannot contain spaces and must follow the same rules as variables names)
  - If it is a primary key
  - The data type for that field

Data types for fields

- `Integer`: the value is an integer value
- `Real`: the value is a floating-pint value
- `Text`: the value is a string text
- `Blob`: the value is stored exactly as it was input.
- You can also specify if the field cannot be left blank by adding `NOT NULL` to the end of the field when you create it

SQLite

- `SQL` stands for "`Structured Query Languages`" and is the main language that large database packages use. Open source relational database management system based on SQL.

- SQLite is free and can be downloaded from www.sqlite.org

- To download select the "`Precomplied Binaries`" for your operating system (Mac OS, Windows or Linux)

- To use SQLite you need to load the "DB Browser for SQLite from https://sqlitebrowser.org

SQLite3 - Example code

- `import sqlite3` Allows Python to use the SQLite3 library

- `with sqlite3.connect("company.db") as db:`
  `cursor=db.cursor()` Connects to the `company` database. If no such database exists, it will create one. The file will be stored in the same folder as the programme.

```
cursor.execute("""CREATE TABLE IF NOT EXIST students(
  id integer PRIMARY KEY,
  name text NOT NULL,
  class text NOT NULL,
  grade integer);""")
```

creates a table called `students` which has four fields (id, name, class and grade). It specifies the data type for each field, defines which field is the primary key and which field cannot be left blank. The triple speech marks allow the code to be split over several lines to make it easier to read rather than having it all displayed in one line.

Example code

```
cursor.execute("""INSERT INTO students(id,name, class,grade)
  VALUES(1,Mary,"Python","67")""")
```

`db.commit()` Inserts data into the `students` table. The `db.commit()` line saves the changes.

```
newID = input ("Enter ID number: ")
newName = input("Enter name: ")
newClass = input("Enter class: ")
newGrade = input("Enter grade: ")
cursor.execute("""INSERT INTO students(id, name, class, grade)
  VALUES(?, ?, ?, ?)""", (newID, newName, newClass, newGrade))
```

`db.commit()` allows a user to enter new data which is then inserted into the `students` table

```
cursor.execute("SELECT * FROM students")
Print(cursor.fetchall()
```

Displays all the data from the `students` table.

`db.close()` This must be the last line in the programme to close the database.

```
cursor.execute("SELECT * FROM students") for x in cursor.fetchall():
  print(x)
```
Displays all the data from the `students` table and displays each record on a separate line

```
cursor.execute("SELECT * FROM students ORDER By name") for x in
cursor.fetchall():
  print(x)
```
Selects all the data from the `students` table, sorted by name and displays each record on a separate line.

```
cursor.execute("SELECT * FROM students WHERE grade>50")
```
Selects all the data from the `students` table where the grade is over 50.

```
cursor.execute("SELECT * FROM students WHERE class = 'Python'")
```
selects all the data from the `students` table where the class is "Python".

```
cursor.execute("""SELECT students.id, students.name, students.lecturer
 FROM students, class WHERE students.class=class.class
 AND students.grade > 70""")
```
Selects the ID and name fields from the `students` table and the lecturer field from the `class` table if the grade is over 70.

```
cursor.execute("SELECT id, name, grade FROM students")
```
Selects the ID, name and grade from the `students` table.

```
whichClass = input(Enter a class: ”)
cursor.execute(“SELECT * FROM employees WHERE class=?”,
[whichClass])
for x in cursor.fetchall():
   print (x)
```
allows the user to enter a class and displays the records of all the students in that class.

```
cursor.execute(“””SELECT students.id, students.name,
class.lecturer
  FROM students,class WHERE students.class= class.class”””)
```
selects the ID and name fields from the students table and the lecturer filed from the class table, using the class filed to link the data. If you do not specify how the tables are linked, Python will assume every students takes every class and you will not get the results you are expecting.

```
cursor.execute(“UPDATE students SET name = ‘Richard’ WHERE id
=1”) db.commit()
```
updates the data in the table(overwriting the original) to change the name to “Richard” for student ID 1

```
cursor.execute(“DELETE students WHERE id=1”)
```
deletes any data in the `students` table where the id is 1

Example code

Create an SQL database called PhoneBook that contains a table called Names with the following data as seen in the code

```
import sqlite3
# Connect to the database called PhoneBook or create one if there is none
with sqlite3.connect("PhoneBook.db") as db:
    cursor = db.cursor()

#    Create a table called Names with foru fields
cursor.execute(""" CREATE TABLE IF NOT EXISTS Names(
id integer PRIMARY KEY,
firstname text,
surname text,
phonenumber text); """)

# Insert data into the table
cursor.execute(""" INSERT INTO Names(id,firstname,surname,phonenumber)
VALUES ("1", "Simon","Pierre","0141647 1367")""")
db.commit() # Saves the changes

# Insert data into the table Names
cursor.execute(""" INSERT INTO Names(id, firstname,surname,phonenumber)
VALUES ("2", "Rita","McVey","0141887 2354")""")
db.commit() # saves the chnages

# Insert data into a table called Names
cursor.execute(""" INSERT INTO Names(id,firstname,surname,phonenumber)
VALUES ("3", "Marc", "Blondel", "0123456 7987")""")
db.commit() # saves the changes

# Select everything from the table called Names and prints one row per line.
cursor.execute(" SELECT * FROM Names")
for x in cursor.fetchall():
    print(x)

db.close() # close the database
```

```
In [146]: runfile('C:/Users/mireilla/.s
(1, 'Simon', 'Pierre', '0141647 1367')
(2, 'Rita', 'McVey', '0141887 2354')
(3, 'Marc', 'Blondel', '0123456 7987')
```