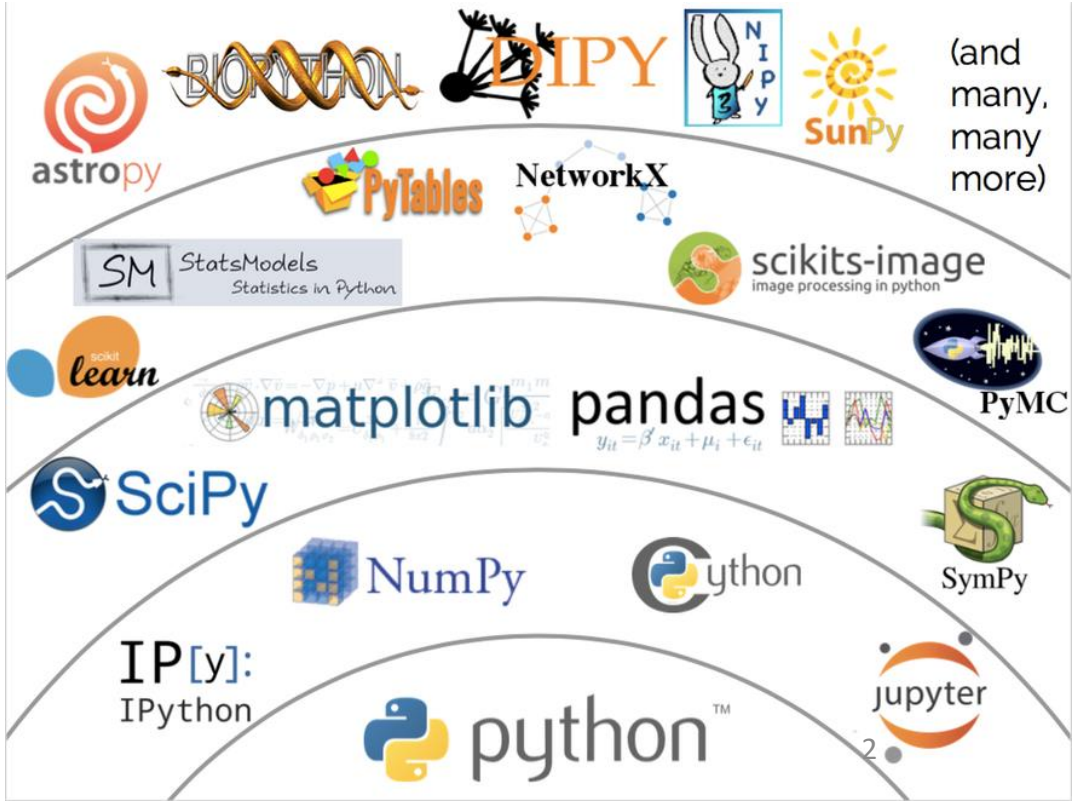
Python-SQLite database

Dr. Sarwan Singh

NIELIT Chandigarh

Agenda

- Introduction
- Database Management System
- SQLite vs MySQL
- Creating database
- Database operations – creating tables, inserting data, selection



One guiding principle of Python code is that

"explicit is better than implicit"

Relational Database Management System

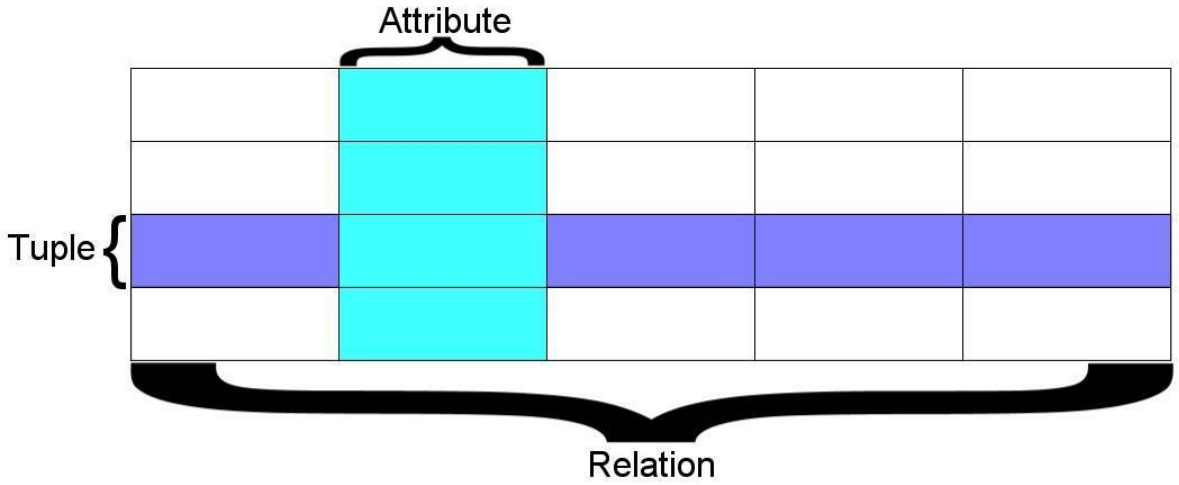
- Relational databases model data by storing rows and columns in tables.
- The power of the relational database lies in its ability to efficiently retrieve data from those tables and in particular where there are multiple tables and the relationships between those tables involved in the query.

Source :en.wikipedia.org/

Terminology

- Database - contains many tables
- Relation (or table) - contains tuples and attributes
- Tuple (or row) - a set of fields that generally represents an “object” like a person or a music track
- Attribute (also column or field) - one of possibly many elements of data corresponding to the object represented by the row

- A relation is defined as a set of tuples that have the same attributes.
- A tuple usually represents an object and information about that object. Objects are typically physical objects or concepts.
- A relation is usually described as a table, which is organized into rows and columns.
- All the data referenced by an attribute are in the same domain and conform to the same constraints.



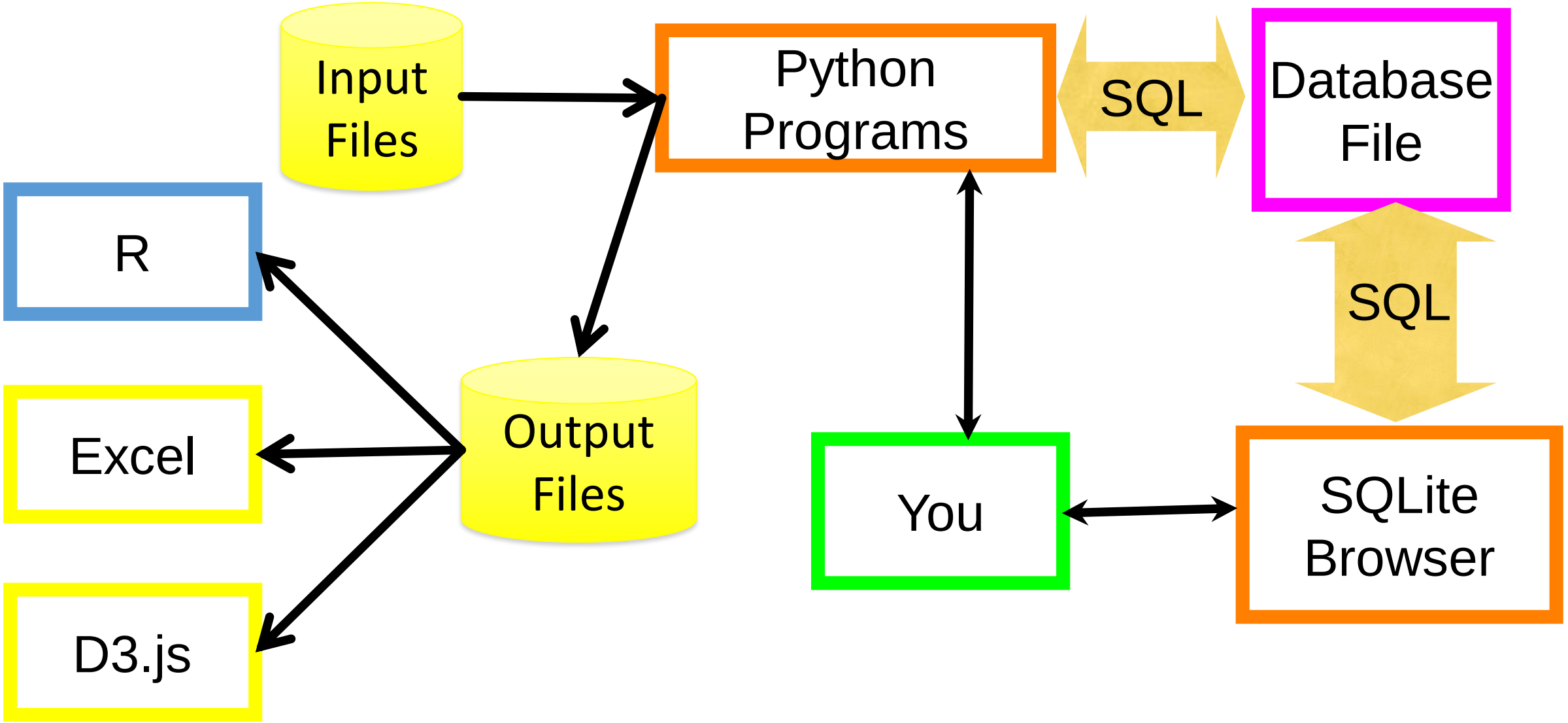
Structured Query Language (SQL)

- Structured Query Language is the language we use to issue commands to the database
- Create a table
- Retrieve some data
- Insert data
- Delete data

Roles in Big database projects

- Application Developer - Builds the logic for the application, the look and feel of the application - monitors the application for problems
- Database Administrator - Monitors and adjusts the database as the program runs in production
- Often both people participate in the building of the “Data model”

Data Analysis Structure



Database Model

- A database model or database schema is the structure or format of a database, described in a formal language supported by the database management system.
- In other words, a “database model” is the application of a data model when used in conjunction with a database management system.

Common databases

- Three Major Database Management Systems in wide use
  - Oracle - Large, commercial, enterprise-scale, very very tweakable
  - MySql - Simpler but very fast and scalable - commercial open source
  - SqlServer - Very nice - from Microsoft (also Access)
- Many other smaller projects, free and open source
  - HSQL, SQLite, Postgress, ...

SQLite is in lots of software...

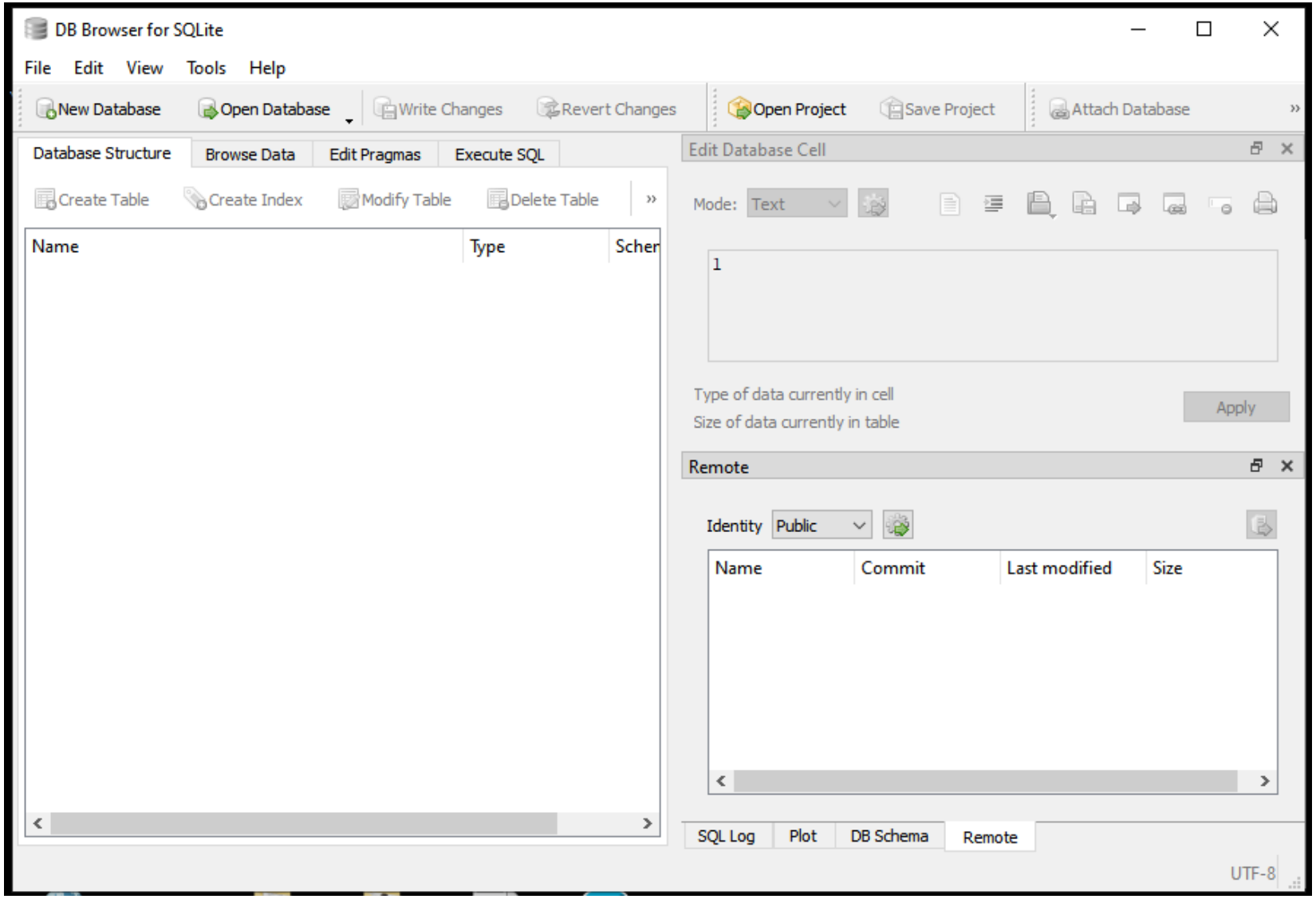
http://www.sqlite.org/famous.html

SQLite Browser

- SQLite is a very popular database - it is free and fast and small
- SQLite Browser allows us to directly manipulate SQLite files
- http://sqlitebrowser.org/
- SQLite is embedded in Python and a number of other languages

SQLite Browser

- http://sqlitebrowser.org/

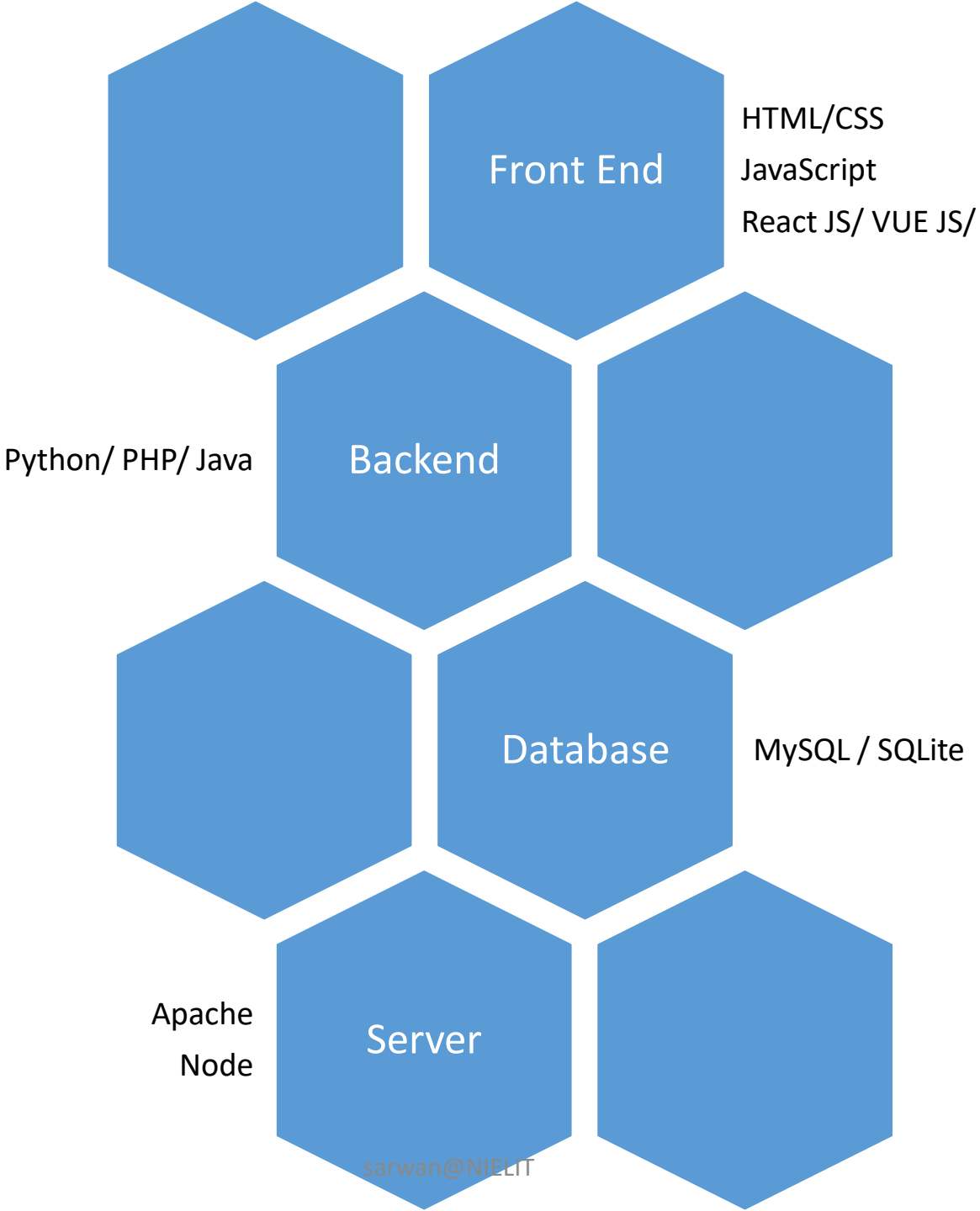


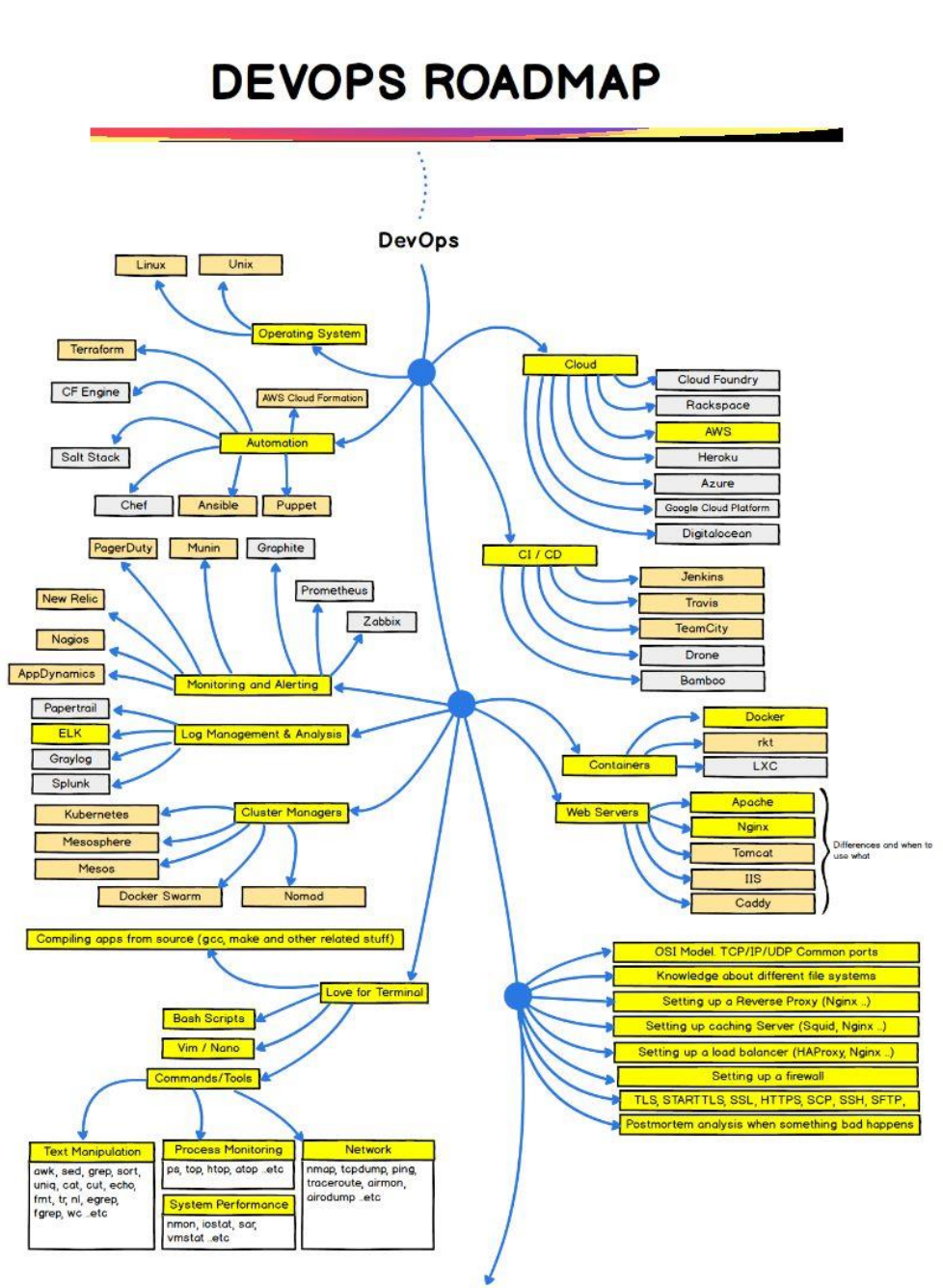
Create table

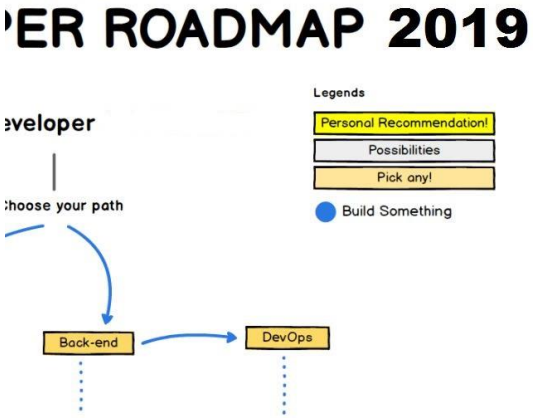
```
CREATE TABLE Users(
  name VARCHAR(128),
  email VARCHAR(128)
)
```

Full Stack Developer

- https://www.edureka.co/blog/how-to-become-a-full-stack-web-developer/
- https://redblink.com/become-full-stack-developer-2019-roadmap/


Front End
HTML/CSS
JavaScript
React JS/ VUE JS/ Angular JS
Backend
Python/ PHP/ Java
Database
MySQL / SQLite
Server
Apache
Node





Source : redblink

```
import sqlite3

dbase = sqlite3.connect('Our_data.db') # Open a database File
print 'Database opened'



dbase.close()
print ' Database Closed'
```

```
import sqlite3
dbase = sqlite3.connect('Our_data.db') # Open a database File
print 'Database opened'

dbase.execute(''' CREATE TABLE IF NOT EXISTS employee_records(
ID INT PRIMARY KEY NOT NULL,
NAME TEXT NOT NULL,
DIVISION TEXT NOT NULL,
STARS INT NOT NULL) ''')

print 'Table created'



dbase.close()
print ' Database Closed'
```

```
import sqlite3

dbase = sqlite3.connect('Our_data.db') # Open a database File
print 'Database opened'

dbase.execute(''' CREATE TABLE IF NOT EXISTS employee_records(    ID INT PRIMARY KEY NOT NULL,
   NAME TEXT NOT NULL,    DIVISION TEXT NOT NULL,    STARS INT NOT NULL) ''')
print 'Table created'
dbase.execute(''' INSERT INTO employee_records(ID,NAME,DIVISION,STARS)
      VALUES(5,'James','Maintenance',4)   ''')

dbase.commit()
print 'REcord inserted'

dbase.close()
print ' Database Closed'
```

```
def read_Data():
    # from math import *
    data = dbase.execute(''' SELECT * FROM employee_records ORDER BY NAME''')
    for record in data:
        print 'ID : '+str(record[0])
        print 'NAME : '+str(record[1])
        print 'DIVISION : '+str(record[2])
        print 'STARS : '+str(record[3])+'\n'
```

```
def update_record():
    dbase.execute(''' UPDATE employee_records set STARS=3 WHERE ID=2 ''')
    dbase.commit()
    print 'Updated'
```