INTRODUCTION TO SQL

- ## What is SQL :

  SQL Stand for Structural Query Language it is used for storing, retrieving and manipulating data from database.

- ## What is Database :

  Data Database is a ordered collection of data, Which is stored in electronic format.

- ## What is DBMS :

  It is a System software which is able to create and managing databases. It store data in table form also called in the form of rows and columns

- Im using MySql and python for handling queries

- ## Import Libraries

In [1]:
```
# import required libraries
import mysql.connector as connector
import getpass
```

- ## Make a connection

In [2]:
```
def make_connection():
    try :
        host = input("Enter host name : ")
        user = input("Enter user name : ")
        passws = getpass.getpass('Enter password : ')

        #create connection object
        mysql = connector.connect(host=host,user=user,passwd=passws,use_pure=T
rue)
        return mysql
    except Exception as e:
        print(e)

mysql = make_connection()
print('Connection is established : ',mysql.is_connected())
```

```
Enter host name : 127.0.0.1
Enter user name : root
Enter password : ........
Connection is established :  True
```

- ## Creating a cursor

In [3]:
```
cur = mysql.cursor()
```

SQL Queries

- ## Show databases

In [4]:
```
cur.execute('show databases')
cur.fetchall()
```

Out[4]:
```
[('bank',),
 ('information_schema',),
 ('mysql',),
 ('performance_schema',),
 ('rushi21',),
 ('sqlintro',),
 ('sys',),
 ('try_rush',)]
```

- ## Create a own database name as 'sqlintro'

In [ ]:
```
cur.execute('create database sqlintro')
mysql.commit()
```

- ## Now, check database list

In [5]:
```
cur.execute('show databases')
cur.fetchall()
```

Out[5]:
```
[('bank',),
 ('information_schema',),
 ('mysql',),
 ('performance_schema',),
 ('rushi21',),
 ('sqlintro',),
 ('sys',),
 ('try_rush',)]
```

- ## Use 'sqlintro' database

In [6]:
```
cur.execute('use sqlintro')
```

- ## Check Table in 'sqlintro'

We did not add any table means our expected output is null

In [7]:
```
cur.execute('show tables')
cur.fetchall()
```

Out[7]:
```
[('student',)]
```

- ## Create a table 'student'

In [ ]:
```
cur.execute('Create table student(s_id int,f_name varchar(20),l_name varchar(2
0),s_age int,s_course varchar(20),duration int,fees int,join_date date, primar
y key(s_id))')
cur.execute('show tables')
cur.fetchall()
```

- ## Describe student table

In [8]:
```
cur.execute("desc student")
cur.fetchall()
```

Out[8]:
```
[('s_id', b'int', 'NO', 'PRI', None, ''),
 ('f_name', b'varchar(20)', 'YES', '', None, ''),
 ('l_name', b'varchar(20)', 'YES', '', None, ''),
 ('s_age', b'int', 'YES', '', None, ''),
 ('s_course', b'varchar(20)', 'YES', '', None, ''),
 ('duration', b'int', 'YES', '', None, ''),
 ('fees', b'int', 'YES', '', None, ''),
 ('join_date', b'date', 'YES', '', None, '')]
```

- ## Insert values into table 'student'

In [ ]:
```
cur.execute("insert into student values(1,'Mick','COX',23,'BE',4,93000,'2018-04-09')")
cur.execute("insert into student values(2,'Allen','Smith',22,'BSc',3,23900,'2019-05-21')")
cur.execute("insert into student values(3,'Jane','Brown',23,'BSc',3,31900,'2018-07-02')")
cur.execute("insert into student values(4,'Mary','Jones',31,'BCom',3,23000,'2013-12-30')")
cur.execute("insert into student values(5,'Ella','Garcia',25,'BE',4,89056,'2017-08-19')")
cur.execute("insert into student values(6,'Isla','Taylor',20,'BCom',3,30567,'2020-02-17')")
cur.execute("insert into student values(7,'Robert','Lee',24,'ME',2,234000,'2018-11-29')")
cur.execute("insert into student values(8,'John','Moore',21,'BCom',3,48770,'2019-06-27')")
cur.execute("insert into student values(9,'Jack','Davis',20,'BE',4,91064,'2021-07-11')")
cur.execute("insert into student values(10,'Noah','Jones',22,'BE',4,95890,'2019-04-15')")
cur.execute("insert into student values(11,'Alfie','White',21,'BSc',3,27980,'2020-02-05')")
cur.execute("insert into student values(12,'Leo','Johnson',29,'ME',2,198700,'2015-06-19')")
```

- ## The SQL SELECT QUERY

- Read all data in student table

In [9]:
```
mysql.commit()
cur.execute('select * from student')
data = cur.fetchall()
data
```

Out[9]:
```
[(1, 'Mick', 'COX', 23, 'BE', 4, 93000, datetime.date(2018, 4, 9)),
 (2, 'Allen', 'Smith', 22, 'BSc', 3, 23900, datetime.date(2019, 5, 21)),
 (3, 'Jane', 'Brown', 23, 'BSc', 3, 31900, datetime.date(2018, 7, 2)),
 (4, 'Mary', 'Jones', 31, 'BCom', 3, 23000, datetime.date(2013, 12, 30)),
 (5, 'Ella', 'Garcia', 25, 'BE', 4, 89056, datetime.date(2017, 8, 19)),
 (6, 'Isla', 'Taylor', 20, 'BCom', 3, 30567, datetime.date(2020, 2, 17)),
 (7, 'Robert', 'Lee', 24, 'ME', 2, 234000, datetime.date(2018, 11, 29)),
 (8, 'John', 'Moore', 21, 'BCom', 3, 48770, datetime.date(2019, 6, 27)),
 (9, 'Jack', 'Davis', 20, 'BE', 4, 91064, datetime.date(2021, 7, 11)),
 (10, 'Noah', 'Jones', 22, 'BE', 4, 95890, datetime.date(2019, 4, 15)),
 (11, 'Alfie', 'White', 21, 'BSc', 3, 27980, datetime.date(2020, 2, 5)),
 (12, 'Leo', 'Johnson', 29, 'ME', 2, 198700, datetime.date(2015, 6, 19))]
```

- Print first name ,course name and joining date

In [10]:
```
cur.execute('select f_name,s_course,join_date from student')
data = cur.fetchall()
data
```

Out[10]:
```
[('Mick', 'BE', datetime.date(2018, 4, 9)),
 ('Allen', 'BSc', datetime.date(2019, 5, 21)),
 ('Jane', 'BSc', datetime.date(2018, 7, 2)),
 ('Mary', 'BCom', datetime.date(2013, 12, 30)),
 ('Ella', 'BE', datetime.date(2017, 8, 19)),
 ('Isla', 'BCom', datetime.date(2020, 2, 17)),
 ('Robert', 'ME', datetime.date(2018, 11, 29)),
 ('John', 'BCom', datetime.date(2019, 6, 27)),
 ('Jack', 'BE', datetime.date(2021, 7, 11)),
 ('Noah', 'BE', datetime.date(2019, 4, 15)),
 ('Alfie', 'BSc', datetime.date(2020, 2, 5)),
 ('Leo', 'ME', datetime.date(2015, 6, 19))]
```

- Print all DISTINCT courses in table

In [11]:
```
cur.execute('select distinct s_course from student')
data = cur.fetchall()
data
```

Out[11]:
```
[('BE',), ('BSc',), ('BCom',), ('ME',)]
```

- **The SQL WHERE CLAUSE**

- Print data of students whose course is BE

In [12]:
```
cur.execute("select * from student where s_course='BE'")
data = cur.fetchall()
data
```

Out[12]:
```
[(1, 'Mick', 'COX', 23, 'BE', 4, 93000, datetime.date(2018, 4, 9)),
 (5, 'Ella', 'Garcia', 25, 'BE', 4, 89056, datetime.date(2017, 8, 19)),
 (9, 'Jack', 'Davis', 20, 'BE', 4, 91064, datetime.date(2021, 7, 11)),
 (10, 'Noah', 'Jones', 22, 'BE', 4, 95890, datetime.date(2019, 4, 15))]
```

- print id,first name,course name of student whose course duration is less than equal to 2

In [13]:
```
cur.execute("select s_id,f_name,s_course from student where duration  <= 2")
data = cur.fetchall()
data
```

Out[13]:
```
[(7, 'Robert', 'ME'), (12, 'Leo', 'ME')]
```

- ## A WHERE clause with AND , OR, NOT

- print details of student whose course duration is greater than 2 and id is even

In [14]:
```
cur.execute("select * from student where duration >2 and s_id % 2=0")
data = cur.fetchall()
data
```

Out[14]:
```
[(2, 'Allen', 'Smith', 22, 'BSc', 3, 23900, datetime.date(2019, 5, 21)),
 (4, 'Mary', 'Jones', 31, 'BCom', 3, 23000, datetime.date(2013, 12, 30)),
 (6, 'Isla', 'Taylor', 20, 'BCom', 3, 30567, datetime.date(2020, 2, 17)),
 (8, 'John', 'Moore', 21, 'BCom', 3, 48770, datetime.date(2019, 6, 27)),
 (10, 'Noah', 'Jones', 22, 'BE', 4, 95890, datetime.date(2019, 4, 15))]
```

- print id and course whose course is BCom and joining date is previous than '2015-01-01'

In [15]:
```
cur.execute("select s_id,s_course from student where s_course='BCom' and join_
date<'2015-01-01'")
data = cur.fetchall()
data
```

Out[15]:
```
[(4, 'BCom')]
```

- Print the data of student whose course duration is greater then 3 or course is ME

In [16]:
```
cur.execute("select * from student where s_course='ME' or duration>3")
data = cur.fetchall()
data
```

Out[16]:
```
[(1, 'Mick', 'COX', 23, 'BE', 4, 93000, datetime.date(2018, 4, 9)),
 (5, 'Ella', 'Garcia', 25, 'BE', 4, 89056, datetime.date(2017, 8, 19)),
 (7, 'Robert', 'Lee', 24, 'ME', 2, 234000, datetime.date(2018, 11, 29)),
 (9, 'Jack', 'Davis', 20, 'BE', 4, 91064, datetime.date(2021, 7, 11)),
 (10, 'Noah', 'Jones', 22, 'BE', 4, 95890, datetime.date(2019, 4, 15)),
 (12, 'Leo', 'Johnson', 29, 'ME', 2, 198700, datetime.date(2015, 6, 19))]
```

- Print the id,first name , last name and course where id is odd or course is Bsc

In [17]:
```
cur.execute("select s_id,f_name,l_name,s_course from student where s_course='BSc' or s_id%2=1")
data = cur.fetchall()
data
```

Out[17]:
```
[(1, 'Mick', 'COX', 'BE'),
 (2, 'Allen', 'Smith', 'BSc'),
 (3, 'Jane', 'Brown', 'BSc'),
 (5, 'Ella', 'Garcia', 'BE'),
 (7, 'Robert', 'Lee', 'ME'),
 (9, 'Jack', 'Davis', 'BE'),
 (11, 'Alfie', 'White', 'BSc')]
```

- print the details of student whose course is not a BE

In [18]:
```
cur.execute("select * from student where NOT s_course='BE'")
data = cur.fetchall()
data
```

Out[18]:
```
[(2, 'Allen', 'Smith', 22, 'BSc', 3, 23900, datetime.date(2019, 5, 21)),
 (3, 'Jane', 'Brown', 23, 'BSc', 3, 31900, datetime.date(2018, 7, 2)),
 (4, 'Mary', 'Jones', 31, 'BCom', 3, 23000, datetime.date(2013, 12, 30)),
 (6, 'Isla', 'Taylor', 20, 'BCom', 3, 30567, datetime.date(2020, 2, 17)),
 (7, 'Robert', 'Lee', 24, 'ME', 2, 234000, datetime.date(2018, 11, 29)),
 (8, 'John', 'Moore', 21, 'BCom', 3, 48770, datetime.date(2019, 6, 27)),
 (11, 'Alfie', 'White', 21, 'BSc', 3, 27980, datetime.date(2020, 2, 5)),
 (12, 'Leo', 'Johnson', 29, 'ME', 2, 198700, datetime.date(2015, 6, 19))]
```

- Print details of student whose didn't join course before '2018-12-31'

In [19]:
```
cur.execute("select * from student where NOT join_date<'2018-12-31'")
data = cur.fetchall()
data
```

Out[19]:
```
[(2, 'Allen', 'Smith', 22, 'BSc', 3, 23900, datetime.date(2019, 5, 21)),
 (6, 'Isla', 'Taylor', 20, 'BCom', 3, 30567, datetime.date(2020, 2, 17)),
 (8, 'John', 'Moore', 21, 'BCom', 3, 48770, datetime.date(2019, 6, 27)),
 (9, 'Jack', 'Davis', 20, 'BE', 4, 91064, datetime.date(2021, 7, 11)),
 (10, 'Noah', 'Jones', 22, 'BE', 4, 95890, datetime.date(2019, 4, 15)),
 (11, 'Alfie', 'White', 21, 'BSc', 3, 27980, datetime.date(2020, 2, 5))]
```

- ## The SQL ORDER BY

- Print all deatils is alphabetical order based on first name

In [20]:
```
cur.execute("select * from student order by f_name")
data = cur.fetchall()
data
```

Out[20]:
```
[(11, 'Alfie', 'White', 21, 'BSc', 3, 27980, datetime.date(2020, 2, 5)),
 (2, 'Allen', 'Smith', 22, 'BSc', 3, 23900, datetime.date(2019, 5, 21)),
 (5, 'Ella', 'Garcia', 25, 'BE', 4, 89056, datetime.date(2017, 8, 19)),
 (6, 'Isla', 'Taylor', 20, 'BCom', 3, 30567, datetime.date(2020, 2, 17)),
 (9, 'Jack', 'Davis', 20, 'BE', 4, 91064, datetime.date(2021, 7, 11)),
 (3, 'Jane', 'Brown', 23, 'BSc', 3, 31900, datetime.date(2018, 7, 2)),
 (8, 'John', 'Moore', 21, 'BCom', 3, 48770, datetime.date(2019, 6, 27)),
 (12, 'Leo', 'Johnson', 29, 'ME', 2, 198700, datetime.date(2015, 6, 19)),
 (4, 'Mary', 'Jones', 31, 'BCom', 3, 23000, datetime.date(2013, 12, 30)),
 (1, 'Mick', 'COX', 23, 'BE', 4, 93000, datetime.date(2018, 4, 9)),
 (10, 'Noah', 'Jones', 22, 'BE', 4, 95890, datetime.date(2019, 4, 15)),
 (7, 'Robert', 'Lee', 24, 'ME', 2, 234000, datetime.date(2018, 11, 29))]
```

- odder a table in such way that new joined student at top

In [21]:
```
cur.execute("select * from student order by join_date desc")
data = cur.fetchall()
data
```

Out[21]:
```
[(9, 'Jack', 'Davis', 20, 'BE', 4, 91064, datetime.date(2021, 7, 11)),
 (6, 'Isla', 'Taylor', 20, 'BCom', 3, 30567, datetime.date(2020, 2, 17)),
 (11, 'Alfie', 'White', 21, 'BSc', 3, 27980, datetime.date(2020, 2, 5)),
 (8, 'John', 'Moore', 21, 'BCom', 3, 48770, datetime.date(2019, 6, 27)),
 (2, 'Allen', 'Smith', 22, 'BSc', 3, 23900, datetime.date(2019, 5, 21)),
 (10, 'Noah', 'Jones', 22, 'BE', 4, 95890, datetime.date(2019, 4, 15)),
 (7, 'Robert', 'Lee', 24, 'ME', 2, 234000, datetime.date(2018, 11, 29)),
 (3, 'Jane', 'Brown', 23, 'BSc', 3, 31900, datetime.date(2018, 7, 2)),
 (1, 'Mick', 'COX', 23, 'BE', 4, 93000, datetime.date(2018, 4, 9)),
 (5, 'Ella', 'Garcia', 25, 'BE', 4, 89056, datetime.date(2017, 8, 19)),
 (12, 'Leo', 'Johnson', 29, 'ME', 2, 198700, datetime.date(2015, 6, 19)),
 (4, 'Mary', 'Jones', 31, 'BCom', 3, 23000, datetime.date(2013, 12, 30))]
```

- ## The SQL SELECT TOP CLAUSE

- select top 5 records of students

In [22]:
```
cur.execute("Select * from student limit 5")
cur.fetchall()
```

Out[22]:
```
[(1, 'Mick', 'COX', 23, 'BE', 4, 93000, datetime.date(2018, 4, 9)),
 (2, 'Allen', 'Smith', 22, 'BSc', 3, 23900, datetime.date(2019, 5, 21)),
 (3, 'Jane', 'Brown', 23, 'BSc', 3, 31900, datetime.date(2018, 7, 2)),
 (4, 'Mary', 'Jones', 31, 'BCom', 3, 23000, datetime.date(2013, 12, 30)),
 (5, 'Ella', 'Garcia', 25, 'BE', 4, 89056, datetime.date(2017, 8, 19))]
```

- odder a table in such way that new joined student at top and print details of top 3 students

In [23]:
```
cur.execute("select * from student order by join_date desc limit 3")
cur.fetchall()
```

Out[23]:
```
[(9, 'Jack', 'Davis', 20, 'BE', 4, 91064, datetime.date(2021, 7, 11)),
 (6, 'Isla', 'Taylor', 20, 'BCom', 3, 30567, datetime.date(2020, 2, 17)),
 (11, 'Alfie', 'White', 21, 'BSc', 3, 27980, datetime.date(2020, 2, 5))]
```

- ## The SQL LIKE-OPERATOR

- Selects all columns of the customer with a first_name starting with "J"

In [24]:
```
cur.execute("select * from student where f_name LIKE 'J%'")
cur.fetchall()
```

Out[24]:
```
[(3, 'Jane', 'Brown', 23, 'BSc', 3, 31900, datetime.date(2018, 7, 2)),
 (8, 'John', 'Moore', 21, 'BCom', 3, 48770, datetime.date(2019, 6, 27)),
 (9, 'Jack', 'Davis', 20, 'BE', 4, 91064, datetime.date(2021, 7, 11))]
```

- Selects all columns of the customer with a first_name Ending with "E"

In [25]:
```
cur.execute("select * from student where f_name Like '%e'")
cur.fetchall()
```

Out[25]:
```
[(3, 'Jane', 'Brown', 23, 'BSc', 3, 31900, datetime.date(2018, 7, 2)),
 (11, 'Alfie', 'White', 21, 'BSc', 3, 27980, datetime.date(2020, 2, 5))]
```

- Selects all columns of the customer with a last _name that have "on" in any position.

In [26]:
```
cur.execute("select * from student where l_name like '%on%'")
cur.fetchall()
```

Out[26]:
```
[(4, 'Mary', 'Jones', 31, 'BCom', 3, 23000, datetime.date(2013, 12, 30)),
 (10, 'Noah', 'Jones', 22, 'BE', 4, 95890, datetime.date(2019, 4, 15)),
 (12, 'Leo', 'Johnson', 29, 'ME', 2, 198700, datetime.date(2015, 6, 19))]
```

- Selects all columns of the customer with a last name that starts with "j" and ends with "s"

In [27]:
```
cur.execute("select * from student where l_name like 'j%s'")
cur.fetchall()
```

Out[27]:
```
[(4, 'Mary', 'Jones', 31, 'BCom', 3, 23000, datetime.date(2013, 12, 30)),
 (10, 'Noah', 'Jones', 22, 'BE', 4, 95890, datetime.date(2019, 4, 15))]
```

- Selects all columns of the customer with a first_name that starts with "m" and are at least four characters in length:

In [28]:
```
cur.execute("select * from student where f_name like 'M___'")
cur.fetchall()
```

Out[28]:
```
[(1, 'Mick', 'COX', 23, 'BE', 4, 93000, datetime.date(2018, 4, 9)),
 (4, 'Mary', 'Jones', 31, 'BCom', 3, 23000, datetime.date(2013, 12, 30))]
```

- **The SQL IN AND NOT IN OPERATORS**

- Selects all the details of student whose s_id in (1,2,3):

In [29]:
```
cur.execute("select * from student where s_id in (1,2,3)")
cur.fetchall()
```

Out[29]:
```
[(1, 'Mick', 'COX', 23, 'BE', 4, 93000, datetime.date(2018, 4, 9)),
 (2, 'Allen', 'Smith', 22, 'BSc', 3, 23900, datetime.date(2019, 5, 21)),
 (3, 'Jane', 'Brown', 23, 'BSc', 3, 31900, datetime.date(2018, 7, 2))]
```

- Selects all the details of student whose s_id not in (1,2,3)

In [30]:
```
cur.execute("select * from student where s_id not in (1,2,3)")
cur.fetchall()
```

Out[30]:
```
[(4, 'Mary', 'Jones', 31, 'BCom', 3, 23000, datetime.date(2013, 12, 30)),
 (5, 'Ella', 'Garcia', 25, 'BE', 4, 89056, datetime.date(2017, 8, 19)),
 (6, 'Isla', 'Taylor', 20, 'BCom', 3, 30567, datetime.date(2020, 2, 17)),
 (7, 'Robert', 'Lee', 24, 'ME', 2, 234000, datetime.date(2018, 11, 29)),
 (8, 'John', 'Moore', 21, 'BCom', 3, 48770, datetime.date(2019, 6, 27)),
 (9, 'Jack', 'Davis', 20, 'BE', 4, 91064, datetime.date(2021, 7, 11)),
 (10, 'Noah', 'Jones', 22, 'BE', 4, 95890, datetime.date(2019, 4, 15)),
 (11, 'Alfie', 'White', 21, 'BSc', 3, 27980, datetime.date(2020, 2, 5)),
 (12, 'Leo', 'Johnson', 29, 'ME', 2, 198700, datetime.date(2015, 6, 19))]
```

- ## The SQL BETWEEN OPERATOR

- Select all the columns from the student with s_id between 3 to 8

In [31]:
```
cur.execute("select * from student where s_id between 3 and 8")
cur.fetchall()
```

Out[31]:
```
[(3, 'Jane', 'Brown', 23, 'BSc', 3, 31900, datetime.date(2018, 7, 2)),
 (4, 'Mary', 'Jones', 31, 'BCom', 3, 23000, datetime.date(2013, 12, 30)),
 (5, 'Ella', 'Garcia', 25, 'BE', 4, 89056, datetime.date(2017, 8, 19)),
 (6, 'Isla', 'Taylor', 20, 'BCom', 3, 30567, datetime.date(2020, 2, 17)),
 (7, 'Robert', 'Lee', 24, 'ME', 2, 234000, datetime.date(2018, 11, 29)),
 (8, 'John', 'Moore', 21, 'BCom', 3, 48770, datetime.date(2019, 6, 27))]
```

- Select all the columns from the student with s_id, not between 3 to 9

In [32]:
```
cur.execute("select * from student where s_id not between 3 and 9")
cur.fetchall()
```

Out[32]:
```
[(1, 'Mick', 'COX', 23, 'BE', 4, 93000, datetime.date(2018, 4, 9)),
 (2, 'Allen', 'Smith', 22, 'BSc', 3, 23900, datetime.date(2019, 5, 21)),
 (10, 'Noah', 'Jones', 22, 'BE', 4, 95890, datetime.date(2019, 4, 15)),
 (11, 'Alfie', 'White', 21, 'BSc', 3, 27980, datetime.date(2020, 2, 5)),
 (12, 'Leo', 'Johnson', 29, 'ME', 2, 198700, datetime.date(2015, 6, 19))]
```

- Select all the columns from the student with fees betweem 20K to 40K

In [33]:
```
cur.execute("select * from student where fees between 20000 and 40000")
cur.fetchall()
```

Out[33]:
```
[(2, 'Allen', 'Smith', 22, 'BSc', 3, 23900, datetime.date(2019, 5, 21)),
 (3, 'Jane', 'Brown', 23, 'BSc', 3, 31900, datetime.date(2018, 7, 2)),
 (4, 'Mary', 'Jones', 31, 'BCom', 3, 23000, datetime.date(2013, 12, 30)),
 (6, 'Isla', 'Taylor', 20, 'BCom', 3, 30567, datetime.date(2020, 2, 17)),
 (11, 'Alfie', 'White', 21, 'BSc', 3, 27980, datetime.date(2020, 2, 5))]
```

- ## The SQL ALIAS

- Creates two aliases, one for the first_name column and one for the last_name column

In [34]:
```
cur.execute("select f_name as first_name, l_name as last_name from student")
cur.fetchall()
```

Out[34]:
```
[('Mick', 'COX'),
 ('Allen', 'Smith'),
 ('Jane', 'Brown'),
 ('Mary', 'Jones'),
 ('Ella', 'Garcia'),
 ('Isla', 'Taylor'),
 ('Robert', 'Lee'),
 ('John', 'Moore'),
 ('Jack', 'Davis'),
 ('Noah', 'Jones'),
 ('Alfie', 'White'),
 ('Leo', 'Johnson')]
```

- Create an alias for the student table

In [35]:
```
cur.execute("Select s.f_name,s.l_name from Student as s")
cur.fetchall()
```

Out[35]:
```
[('Mick', 'COX'),
 ('Allen', 'Smith'),
 ('Jane', 'Brown'),
 ('Mary', 'Jones'),
 ('Ella', 'Garcia'),
 ('Isla', 'Taylor'),
 ('Robert', 'Lee'),
 ('John', 'Moore'),
 ('Jack', 'Davis'),
 ('Noah', 'Jones'),
 ('Alfie', 'White'),
 ('Leo', 'Johnson')]
```

- ## Aggregate Functionds

- Find the small age

In [36]:
```
cur.execute("select min(s_age) from student")
cur.fetchall()
```

Out[36]:
```
[(20,)]
```

- Hight fees

In [37]:
```
cur.execute("select max(fees) from student")
cur.fetchall()
```

Out[37]:
```
[(234000,)]
```

- Find total count of student

In [38]:
```
cur.execute("select count(*) from student")
cur.fetchall()
```

Out[38]:
```
[(12,)]
```

- Find average fees

In [39]:
```
cur.execute("select AVG(fees) from student")
cur.fetchall()
```

Out[39]:
```
[(Decimal('82318.9167'),)]
```

- Find Sum of age of all student

In [40]:
```
cur.execute("Select sum(s_age) from student")
cur.fetchall()
```

Out[40]:
```
[(Decimal('281'),)]
```

- ## The SQL GROUP BY STATEMENT

- Count the number of student in each course

In [41]:
```
cur.execute("select s_course,count(s_id) from student group by s_course")
cur.fetchall()
```

Out[41]:
```
[('BE', 4), ('BSc', 3), ('BCom', 3), ('ME', 2)]
```

- group by duration

In [42]:
```
cur.execute("select count(duration) from student group by duration")
cur.fetchall()
```

Out[42]:
```
[(4,), (6,), (2,)]
```

- ## The SQL HAVING CLAUSE

- count the course whose course duration is greater than 2

In [45]:
```
cur.execute("select s_course,count(*) from student group by s_course having co
unt(duration)>2")
cur.fetchall()
```

Out[45]:
```
[('BE', 4), ('BSc', 3), ('BCom', 3)]
```

- find the average age of student according to there courses

In [49]:
```
cur.execute("select s_course,avg(s_age) from student group by s_course")
cur.fetchall()
```

Out[49]:
```
[('BE', Decimal('22.5000')),
 ('BSc', Decimal('22.0000')),
 ('BCom', Decimal('24.0000')),
 ('ME', Decimal('26.5000'))]
```

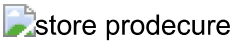
- List the number of students which has a age more than 23

In [52]:
```
cur.execute("select * from student group by(s_course) having s_age>23")
cur.fetchall()
```

Out[52]:
```
[(4, 'Mary', 'Jones', 31, 'BCom', 3, 23000, datetime.date(2013, 12, 30)),
 (7, 'Robert', 'Lee', 24, 'ME', 2, 234000, datetime.date(2018, 11, 29))]
```

- ## The SQL STORED PROCEDURE

- create a store procedure with no parameter



- ## Create a new table student2

In [61]:
```
cur.execute('Create table student2(s_id int,f_name varchar(20),l_name varchar
(20),s_age int,s_course varchar(20),duration int,fees int,join_date date, prim
ary key(s_id))')
cur.execute('show tables')
cur.fetchall()
```

Out[61]:
```
[('student',), ('student2',)]
```

- ## Insert values in student2 table

In [62]:
```
cur.execute("insert into student2 values(1,'Mick','COX',24,'BE',4,93000,'2018-
04-09')")
cur.execute("insert into student2 values(2,'Harry','Leo',22,'BCA',3,12390,'201
9-05-21')")
cur.execute("insert into student2 values(3,'Ada','Freya',25,'BCA',3,15900,'201
7-02-22')")
cur.execute("insert into student2 values(4,'Paul','Alice',27,'MCA',2,23700,'20
19-02-03')")
cur.execute("insert into student2 values(5,'Ella','Garcia',25,'BE',4,89056,'20
17-08-19')")
cur.execute("insert into student2 values(6,'Isla','Taylor',20,'BCom',3,3056
7,'2020-02-17')")
cur.execute("insert into student2 values(7,'Robert','Lee',24,'ME',2,234000,'20
18-11-29')")
cur.execute("insert into student2 values(8,'Anne','Noah',23,'MCA',2,18770,'202
1-01-27')")
```

In [64]:
```
mysql.commit()
cur.execute('select * from student2')
data = cur.fetchall()
data
```

Out[64]:
```
[(1, 'Mick', 'COX', 24, 'BE', 4, 93000, datetime.date(2018, 4, 9)),
 (2, 'Harry', 'Leo', 22, 'BCA', 3, 12390, datetime.date(2019, 5, 21)),
 (3, 'Ada', 'Freya', 25, 'BCA', 3, 15900, datetime.date(2017, 2, 22)),
 (4, 'Paul', 'Alice', 27, 'MCA', 2, 23700, datetime.date(2019, 2, 3)),
 (5, 'Ella', 'Garcia', 25, 'BE', 4, 89056, datetime.date(2017, 8, 19)),
 (6, 'Isla', 'Taylor', 20, 'BCom', 3, 30567, datetime.date(2020, 2, 17)),
 (7, 'Robert', 'Lee', 24, 'ME', 2, 234000, datetime.date(2018, 11, 29)),
 (8, 'Anne', 'Noah', 23, 'MCA', 2, 18770, datetime.date(2021, 1, 27))]
```

- ## Joining Operator

  - For joining i will considered student and student2 tables

- find the unique records in table student and student2 without duplication

In [65]:
```
cur.execute("Select * from student union select * from student2")
cur.fetchall()
```

Out[65]:
```
[(1, 'Mick', 'COX', 23, 'BE', 4, 93000, datetime.date(2018, 4, 9)),
 (2, 'Allen', 'Smith', 22, 'BSc', 3, 23900, datetime.date(2019, 5, 21)),
 (3, 'Jane', 'Brown', 23, 'BSc', 3, 31900, datetime.date(2018, 7, 2)),
 (4, 'Mary', 'Jones', 31, 'BCom', 3, 23000, datetime.date(2013, 12, 30)),
 (5, 'Ella', 'Garcia', 25, 'BE', 4, 89056, datetime.date(2017, 8, 19)),
 (6, 'Isla', 'Taylor', 20, 'BCom', 3, 30567, datetime.date(2020, 2, 17)),
 (7, 'Robert', 'Lee', 24, 'ME', 2, 234000, datetime.date(2018, 11, 29)),
 (8, 'John', 'Moore', 21, 'BCom', 3, 48770, datetime.date(2019, 6, 27)),
 (9, 'Jack', 'Davis', 20, 'BE', 4, 91064, datetime.date(2021, 7, 11)),
 (10, 'Noah', 'Jones', 22, 'BE', 4, 95890, datetime.date(2019, 4, 15)),
 (11, 'Alfie', 'White', 21, 'BSc', 3, 27980, datetime.date(2020, 2, 5)),
 (12, 'Leo', 'Johnson', 29, 'ME', 2, 198700, datetime.date(2015, 6, 19)),
 (1, 'Mick', 'COX', 24, 'BE', 4, 93000, datetime.date(2018, 4, 9)),
 (2, 'Harry', 'Leo', 22, 'BCA', 3, 12390, datetime.date(2019, 5, 21)),
 (3, 'Ada', 'Freya', 25, 'BCA', 3, 15900, datetime.date(2017, 2, 22)),
 (4, 'Paul', 'Alice', 27, 'MCA', 2, 23700, datetime.date(2019, 2, 3)),
 (8, 'Anne', 'Noah', 23, 'MCA', 2, 18770, datetime.date(2021, 1, 27))]
```

- find the unique records in table student and student2 with duplication

In [66]:
```
cur.execute("select * from student union all select * from student2")
cur.fetchall()
```

Out[66]:
```
[(1, 'Mick', 'COX', 23, 'BE', 4, 93000, datetime.date(2018, 4, 9)),
 (2, 'Allen', 'Smith', 22, 'BSc', 3, 23900, datetime.date(2019, 5, 21)),
 (3, 'Jane', 'Brown', 23, 'BSc', 3, 31900, datetime.date(2018, 7, 2)),
 (4, 'Mary', 'Jones', 31, 'BCom', 3, 23000, datetime.date(2013, 12, 30)),
 (5, 'Ella', 'Garcia', 25, 'BE', 4, 89056, datetime.date(2017, 8, 19)),
 (6, 'Isla', 'Taylor', 20, 'BCom', 3, 30567, datetime.date(2020, 2, 17)),
 (7, 'Robert', 'Lee', 24, 'ME', 2, 234000, datetime.date(2018, 11, 29)),
 (8, 'John', 'Moore', 21, 'BCom', 3, 48770, datetime.date(2019, 6, 27)),
 (9, 'Jack', 'Davis', 20, 'BE', 4, 91064, datetime.date(2021, 7, 11)),
 (10, 'Noah', 'Jones', 22, 'BE', 4, 95890, datetime.date(2019, 4, 15)),
 (11, 'Alfie', 'White', 21, 'BSc', 3, 27980, datetime.date(2020, 2, 5)),
 (12, 'Leo', 'Johnson', 29, 'ME', 2, 198700, datetime.date(2015, 6, 19)),
 (1, 'Mick', 'COX', 24, 'BE', 4, 93000, datetime.date(2018, 4, 9)),
 (2, 'Harry', 'Leo', 22, 'BCA', 3, 12390, datetime.date(2019, 5, 21)),
 (3, 'Ada', 'Freya', 25, 'BCA', 3, 15900, datetime.date(2017, 2, 22)),
 (4, 'Paul', 'Alice', 27, 'MCA', 2, 23700, datetime.date(2019, 2, 3)),
 (5, 'Ella', 'Garcia', 25, 'BE', 4, 89056, datetime.date(2017, 8, 19)),
 (6, 'Isla', 'Taylor', 20, 'BCom', 3, 30567, datetime.date(2020, 2, 17)),
 (7, 'Robert', 'Lee', 24, 'ME', 2, 234000, datetime.date(2018, 11, 29)),
 (8, 'Anne', 'Noah', 23, 'MCA', 2, 18770, datetime.date(2021, 1, 27))]
```

- **SQL JOIN TYPEs**

- print the comman record between student and student2 table
- inner join

- Find the id,name and age of student whoes age is same

In [80]:
```
cur.execute("select s.s_id,s.f_name,s1.s_id,s1.f_name,s.s_age,s1.s_age from st
udent as s inner join student2 as s1 on s.s_age=s1.s_age")
cur.fetchall()
```

Out[80]:
```
[(1, 'Mick', 8, 'Anne', 23, 23),
 (2, 'Allen', 2, 'Harry', 22, 22),
 (3, 'Jane', 8, 'Anne', 23, 23),
 (5, 'Ella', 5, 'Ella', 25, 25),
 (5, 'Ella', 3, 'Ada', 25, 25),
 (6, 'Isla', 6, 'Isla', 20, 20),
 (7, 'Robert', 7, 'Robert', 24, 24),
 (7, 'Robert', 1, 'Mick', 24, 24),
 (9, 'Jack', 6, 'Isla', 20, 20),
 (10, 'Noah', 2, 'Harry', 22, 22)]
```

- find all the record in student2 and matching record in student

In [85]:
```
cur.execute("select s.s_id,s.f_name,s1.s_id,s1.f_name,s.s_course,s1.s_course f
rom student2 as s left join student as s1 on s.s_course=s1.s_course")
cur.fetchall()
```

Out[85]:
```
[(1, 'Mick', 10, 'Noah', 'BE', 'BE'),
 (1, 'Mick', 9, 'Jack', 'BE', 'BE'),
 (1, 'Mick', 5, 'Ella', 'BE', 'BE'),
 (1, 'Mick', 1, 'Mick', 'BE', 'BE'),
 (2, 'Harry', None, None, 'BCA', None),
 (3, 'Ada', None, None, 'BCA', None),
 (4, 'Paul', None, None, 'MCA', None),
 (5, 'Ella', 10, 'Noah', 'BE', 'BE'),
 (5, 'Ella', 9, 'Jack', 'BE', 'BE'),
 (5, 'Ella', 5, 'Ella', 'BE', 'BE'),
 (5, 'Ella', 1, 'Mick', 'BE', 'BE'),
 (6, 'Isla', 8, 'John', 'BCom', 'BCom'),
 (6, 'Isla', 6, 'Isla', 'BCom', 'BCom'),
 (6, 'Isla', 4, 'Mary', 'BCom', 'BCom'),
 (7, 'Robert', 12, 'Leo', 'ME', 'ME'),
 (7, 'Robert', 7, 'Robert', 'ME', 'ME'),
 (8, 'Anne', None, None, 'MCA', None)]
```

- find all the record in student2 and matching record in student
- using right join

In [86]:
```
cur.execute("select s.s_id,s.f_name,s1.s_id,s1.f_name,s.s_course,s1.s_course f
rom student as s right join student2 as s1 on s.s_course=s1.s_course")
cur.fetchall()
```

Out[86]:
```
[(10, 'Noah', 1, 'Mick', 'BE', 'BE'),
 (9, 'Jack', 1, 'Mick', 'BE', 'BE'),
 (5, 'Ella', 1, 'Mick', 'BE', 'BE'),
 (1, 'Mick', 1, 'Mick', 'BE', 'BE'),
 (None, None, 2, 'Harry', None, 'BCA'),
 (None, None, 3, 'Ada', None, 'BCA'),
 (None, None, 4, 'Paul', None, 'MCA'),
 (10, 'Noah', 5, 'Ella', 'BE', 'BE'),
 (9, 'Jack', 5, 'Ella', 'BE', 'BE'),
 (5, 'Ella', 5, 'Ella', 'BE', 'BE'),
 (1, 'Mick', 5, 'Ella', 'BE', 'BE'),
 (8, 'John', 6, 'Isla', 'BCom', 'BCom'),
 (6, 'Isla', 6, 'Isla', 'BCom', 'BCom'),
 (4, 'Mary', 6, 'Isla', 'BCom', 'BCom'),
 (12, 'Leo', 7, 'Robert', 'ME', 'ME'),
 (7, 'Robert', 7, 'Robert', 'ME', 'ME'),
 (None, None, 8, 'Anne', None, 'MCA')]
```

- self join

In [113]:
```
cur.execute("select s1.f_name,s2.f_name,s1.s_course from student as s1, studen
t2 as s2 where s1.s_course=s2.s_course")
for i in cur.fetchall():
    print(i,end='\n')
```

```
('Mick', 'Ella', 'BE')
('Mick', 'Mick', 'BE')
('Mary', 'Isla', 'BCom')
('Ella', 'Ella', 'BE')
('Ella', 'Mick', 'BE')
('Isla', 'Isla', 'BCom')
('Robert', 'Robert', 'ME')
('John', 'Isla', 'BCom')
('Jack', 'Ella', 'BE')
('Jack', 'Mick', 'BE')
('Noah', 'Ella', 'BE')
('Noah', 'Mick', 'BE')
('Leo', 'Robert', 'ME')
```

- **NOTE :**
  - **MySQL don't support :**

- EXCEPT operator
- INTERSECT operator
- FULL OUTER JOIN

SQL DATABASE

- ## The SQL CREATE DATABASE STATEMENT

- Let's create a database and give name as testdb

In [123]:
```
cur.execute("create database testdb")
```

- check the databases in MySQL

In [124]:
```
cur.execute("show databases")
cur.fetchall()
```

Out[124]:
```
[('bank',),
 ('information_schema',),
 ('mysql',),
 ('performance_schema',),
 ('rushi21',),
 ('sqlintro',),
 ('sys',),
 ('testdb',)]
```

- Use testdb database

In [125]:
```
cur.execute("use testdb")
```

- ## The SQL CREATE TABLE

- Let's create a customer table

In [126]:
```
cur.execute("create table customer(id int, first_name varchar(10),last_name va
rchar(10), city varchar(10), country varchar(10), phone varchar(10))")
```

- Check the schema of the table

In [127]:
```
cur.execute("desc customer")
cur.fetchall()
```

Out[127]:
```
[('id', b'int', 'YES', '', None, ''),
 ('first_name', b'varchar(10)', 'YES', '', None, ''),
 ('last_name', b'varchar(10)', 'YES', '', None, ''),
 ('city', b'varchar(10)', 'YES', '', None, ''),
 ('country', b'varchar(10)', 'YES', '', None, ''),
 ('phone', b'varchar(10)', 'YES', '', None, '')]
```

- **The SQL INSERT INTO STATEMENT**

- INSERT INTO query

In [129]:
```
cur.execute("insert into customer values(2, 'Ana', 'Trujillo', 'Mexico', 'Mexi
co', 55554729)")
cur.execute("insert into customer values(3, 'Antonio', 'Moreno', 'Mexico', 'Me
xico', 55553932)")
cur.execute("insert into customer values(4,'Thomas','Hardy','London','UK',5557
788)")
cur.execute("insert into customer values(5,'Christina','Berglund','Lulea','Swe
den',09211234)")
cur.execute("insert into customer values(6,'Hanna','Moos','Mannheim','German
y',62108460)")
cur.execute("insert into customer values(7,'Frederique','Citeaux','Strasbour
g','France',88661531)")
cur.execute("insert into customer values(8,'Martin','Sommer','Madrid','Spain',
5552280)")
cur.execute("insert into customer values(9,'Laurence','Lebihan','Marseille','F
rance',91244540)")
cur.execute("insert into customer values(10,'Elizabeth','Lincoln','Tsawasse
n','Canada',5554729)")
cur.execute("insert into customer values(11,'Victoria','Ashworth','London','U
K',5551212)")
mysql.commit()
```

- print all records in customer table

In [130]:
```
cur.execute("select * from customer")
cur.fetchall()
```

Out[130]:
```
[(2, 'Ana', 'Trujillo', 'Mexico', 'Mexico', '55554729'),
 (3, 'Antonio', 'Moreno', 'Mexico', 'Mexico', '55553932'),
 (4, 'Thomas', 'Hardy', 'London', 'UK', '5557788'),
 (5, 'Christina', 'Berglund', 'Lulea', 'Sweden', '9211234'),
 (6, 'Hanna', 'Moos', 'Mannheim', 'Germany', '62108460'),
 (7, 'Frederique', 'Citeaux', 'Strasbourg', 'France', '88661531'),
 (8, 'Martin', 'Sommer', 'Madrid', 'Spain', '5552280'),
 (9, 'Laurence', 'Lebihan', 'Marseille', 'France', '91244540'),
 (10, 'Elizabeth', 'Lincoln', 'Tsawassen', 'Canada', '5554729'),
 (11, 'Victoria', 'Ashworth', 'London', 'UK', '5551212')]
```

- **The SQL NULL VALUES**

- Insert the NULL values in tables

In [131]:
```
cur.execute("INSERT INTO customer VALUES(11, 'Victoria', 'Ashworth', 'London',
NULL, 5551212)")
```

- check for NULL Values

In [132]:
```
cur.execute("select * from customer where country IS NULL")
cur.fetchall()
```

Out[132]:
```
[(11, 'Victoria', 'Ashworth', 'London', None, '5551212')]
```

- check for not NULL Values

In [133]:
```
cur.execute("select * from customer where country IS NOT NULL")
cur.fetchall()
```

Out[133]:
```
[(2, 'Ana', 'Trujillo', 'Mexico', 'Mexico', '55554729'),
 (3, 'Antonio', 'Moreno', 'Mexico', 'Mexico', '55553932'),
 (4, 'Thomas', 'Hardy', 'London', 'UK', '5557788'),
 (5, 'Christina', 'Berglund', 'Lulea', 'Sweden', '9211234'),
 (6, 'Hanna', 'Moos', 'Mannheim', 'Germany', '62108460'),
 (7, 'Frederique', 'Citeaux', 'Strasbourg', 'France', '88661531'),
 (8, 'Martin', 'Sommer', 'Madrid', 'Spain', '5552280'),
 (9, 'Laurence', 'Lebihan', 'Marseille', 'France', '91244540'),
 (10, 'Elizabeth', 'Lincoln', 'Tsawassen', 'Canada', '5554729'),
 (11, 'Victoria', 'Ashworth', 'London', 'UK', '5551212')]
```

- ## The SQL UPDATE STATEMENT

- update country = 'Maxico' where country is null

In [135]:
```
cur.execute("update customer set country='Maxico' where country is NULL")
```

Out[135]:
```
[]
```

- Check table

In [136]:
```
cur.execute("select * from customer")
cur.fetchall()
```

Out[136]:
```
[(2, 'Ana', 'Trujillo', 'Mexico', 'Mexico', '55554729'),
 (3, 'Antonio', 'Moreno', 'Mexico', 'Mexico', '55553932'),
 (4, 'Thomas', 'Hardy', 'London', 'UK', '5557788'),
 (5, 'Christina', 'Berglund', 'Lulea', 'Sweden', '9211234'),
 (6, 'Hanna', 'Moos', 'Mannheim', 'Germany', '62108460'),
 (7, 'Frederique', 'Citeaux', 'Strasbourg', 'France', '88661531'),
 (8, 'Martin', 'Sommer', 'Madrid', 'Spain', '5552280'),
 (9, 'Laurence', 'Lebihan', 'Marseille', 'France', '91244540'),
 (10, 'Elizabeth', 'Lincoln', 'Tsawassen', 'Canada', '5554729'),
 (11, 'Victoria', 'Ashworth', 'London', 'UK', '5551212'),
 (11, 'Victoria', 'Ashworth', 'London', 'Maxico', '5551212')]
```

- ## The SQL ALTER TABLE STATEMENT

- add a new column in a table

In [137]:
```
cur.execute("alter table customer add emai_id varchar(25)")
cur.execute("desc customer")
cur.fetchall()
```

Out[137]:
```
[('id', b'int', 'YES', '', None, ''),
 ('first_name', b'varchar(10)', 'YES', '', None, ''),
 ('last_name', b'varchar(10)', 'YES', '', None, ''),
 ('city', b'varchar(10)', 'YES', '', None, ''),
 ('country', b'varchar(10)', 'YES', '', None, ''),
 ('phone', b'varchar(10)', 'YES', '', None, ''),
 ('emai_id', b'varchar(25)', 'YES', '', None, '')]
```

In [138]:
```
cur.execute("alter table customer add DOB varchar(25)")
cur.execute("desc customer")
cur.fetchall()
```

Out[138]:
```
[('id', b'int', 'YES', '', None, ''),
 ('first_name', b'varchar(10)', 'YES', '', None, ''),
 ('last_name', b'varchar(10)', 'YES', '', None, ''),
 ('city', b'varchar(10)', 'YES', '', None, ''),
 ('country', b'varchar(10)', 'YES', '', None, ''),
 ('phone', b'varchar(10)', 'YES', '', None, ''),
 ('emai_id', b'varchar(25)', 'YES', '', None, ''),
 ('DOB', b'varchar(25)', 'YES', '', None, '')]
```

- change the data type of column DOB to date in a table

In [143]:
```
cur.execute("alter table customer modify DOB date")
cur.execute("desc customer")
cur.fetchall()
```

Out[143]:
```
[('id', b'int', 'YES', '', None, ''),
 ('first_name', b'varchar(10)', 'YES', '', None, ''),
 ('last_name', b'varchar(10)', 'YES', '', None, ''),
 ('city', b'varchar(10)', 'YES', '', None, ''),
 ('country', b'varchar(10)', 'YES', '', None, ''),
 ('phone', b'varchar(10)', 'YES', '', None, ''),
 ('emai_id', b'varchar(25)', 'YES', '', None, ''),
 ('DOB', b'date', 'YES', '', None, '')]
```

- **The SQL DELETE RECORDS**

- Delete all records where city is london

In [145]:
```
cur.execute("delete from customer where city='London'")
cur.execute("select * from customer")
cur.fetchall()
```

Out[145]:
```
[(2, 'Ana', 'Trujillo', 'Mexico', 'Mexico', '55554729', None, None),
 (3, 'Antonio', 'Moreno', 'Mexico', 'Mexico', '55553932', None, None),
 (5, 'Christina', 'Berglund', 'Lulea', 'Sweden', '9211234', None, None),
 (6, 'Hanna', 'Moos', 'Mannheim', 'Germany', '62108460', None, None),
 (7, 'Frederique', 'Citeaux', 'Strasbourg', 'France', '88661531', None, Non
e),
 (8, 'Martin', 'Sommer', 'Madrid', 'Spain', '5552280', None, None),
 (9, 'Laurence', 'Lebihan', 'Marseille', 'France', '91244540', None, None),
 (10, 'Elizabeth', 'Lincoln', 'Tsawassen', 'Canada', '5554729', None, None)]
```

- ## The SQL DELETE COLUMNS (ALTER TABLE - DROP COLUMN)

- Drop emai_id columns in table

In [149]:
```
cur.execute("alter table customer drop column emai_id ")
cur.execute("desc customer")
cur.fetchall()
```

Out[149]:
```
[('id', b'int', 'YES', '', None, ''),
 ('first_name', b'varchar(10)', 'YES', '', None, ''),
 ('last_name', b'varchar(10)', 'YES', '', None, ''),
 ('city', b'varchar(10)', 'YES', '', None, ''),
 ('country', b'varchar(10)', 'YES', '', None, ''),
 ('phone', b'varchar(10)', 'YES', '', None, ''),
 ('DOB', b'date', 'YES', '', None, '')]
```

- ## The SQL DROP TABLE STATEMENT

- drop an existing customer table in a database

In [150]:
```
cur.execute("DROP TABLE customer")
```

- check tables in database testdb

In [151]:
```
cur.execute("show tables")
cur.fetchall()
```

Out[151]:
```
[]
```

- ## The SQL DROP DATABASE STATEMENT

- Let's drop the created database

In [152]:
```
cur.execute("drop database testdb")
```

- check the databases in MySQL

In [153]:
```
cur.execute("show databases")
cur.fetchall()
```

Out[153]:
```
[('bank',),
 ('information_schema',),
 ('mysql',),
 ('performance_schema',),
 ('rushi21',),
 ('sqlintro',),
 ('sys',)]
```

Thank You!