FILE I/O

File input / output has a basic structure of logic:

Open a file

Read or Write to the file

Close the file

Another source of input is a text file. Example:

configFile = open("C:\\temp\\file.txt","r")

The open function's first argument is the path. The second argument is r which stands for read-only.

MODE	DESCRIPTION
r	Opens the file as read-only
r+	Opens a file for reading and writing
w	Opens a file for writing. Overwrites an existing file if it exists
w+	Opens a file for reading and writing
a	Appends to an existing file

To read a file, you can read one line at a time. Example:

```
configFile = open("C:\\temp\\file.txt","r")
data = configFile.readline()
configFile.close()
```

Or, you can read the entire file all at once. Example:

```
configFile = open("C:\\temp\\file.txt","r")
data = configFile.readlines()
configFile.close()
```

To write to a file, you can write one line at a time. Example:

```
configFile = open("C:\\temp\\file.txt","w")
configFile.write("Hello")
configFile.close()
```

To write to a file, you can write multiple lines at a time. Example:

```
configFile = open("C:\\temp\\file.txt","w")
temp = [1, 10, "Burnaby"]
configFile.writelines(temp)
configFile.close()
```

To append to a file, you can write multiple lines at a time. Example:

```
configFile = open("C:\\temp\\file.txt","a")
configFile.write("hi bob")
configFile.close()
```

DATABASE

Relational database has similar logic to file I/O

Open database connection

Insert, Delete, Update or Query

Close database connection

A relational database is a collection of tables

A table is a collection of rows and columns like a spreadsheet.

A row in a table is a record.

A column in a table is a field.

A cursor is pointer to a record.

A relationship is a link between two or more tables. Relationships make it possible to find data in one table that pertains to a specific record in another table. This relationship is linked by a primary key.

If one record in a table is linked with only one row in another table, then this is called a one-to-one relationship (1:1). Example: Capital city to Country

If one record in a table is linked with many in another table, then this is called a one-to-many relationship (1:M). Example: Mother has many children.

There is also many-to-many relationships(M:M) where man records can be linked with many records in another table and vice versa. Example: Author can write many books. Books can have many authors.

A primary key is a field or group of fields to identify a record. A foreign key is a field or group of fields to primary keys in other tables.

Datatypes are the table's columns that specify the type of data that can exist in that column. For example, phoneNum might be defined as INT. This means only integer numbers can be saved in the phoneNum column

To query a table, use the Select statement. Example:

Select <column name(s)> from <table name>

One can add a wildcard for the column name. Example:

Select * from <table name>

One can add conditions to the statement like the where clause. Example:

Select * from <table name> where name = “John”

Order by is another clause. Example:

Select * from <table name> order by name asc

For more, goto https://www.webucator.com/tutorial/learn-sql/relational-database-basics.cfm

To insert a record into a database, use the Insert statement. Example:

Insert into <table name> (column1, column2) values (value1, value2)

To delete a record into a database, use Delete statement. Example:

Delete from <table name> where <condition>

To update a record in a database, use Update statement. Example:

Update <table name> set <column name>, <column name> where <condition>

Python has a database already pre-installed called SQLite. To create a database, use the connect function. If it doesn't exist, SQLite will create one for you. **Note other vendors (Oracle, MS, etc.) may have different methods in creating databases.**

```
import sqlite3

conn = sqlite3.connect("c:\\metrotown.db")

curr = conn.cursor()

curr.execute("Create table if not exists ken(id int, name text))")

conn.close()
```

To insert a record, we add a commit statement and change the SQL string. Example:

```
import sqlite3

conn = sqlite3.connect("c:\\metrotown.db")

curr = conn.cursor()

sql = "insert into al(id, name) values(5,'jericho')"

curr.execute(sql)

conn.commit()

conn.close()
```

To update a record, it is the same as insert except the SQL string. Example

```
sql = "Update al set id = 1 where id = 5""
```

To delete a record, it is the same except the SQL statement. Example:

```
sql = "delete from al where id = 1"
```

When searching for a record, use the select statement, put the data in a list and loop through the list. Example:

```
import sqlite3

conn = sqlite3.connect("c:\\metrotown.db")

curr = conn.cursor()

sql = "select * from al"

curr.execute(sql)

data = curr.fetchall()

item = []

for item in data:

        print(item)

conn.close()
```