1. ***10 minute intro -- Loops, If statements, string manipulation, datatype conversion***

   String Manipulation

   ```
   str = "1999, 2002, 2011, 2012, 2015, 2016, 2017 and 2020"
   str = str.replace(" and", ",")
   print str
   dateList = str.split(", ")
   print dateList
   ```

   Datatype Conversion in Python

   Convert Ints into Strings: str(1)

   Convert Strings into Ints: int('100')

   Loops and If Statements

   ```
   for i in dateList:
           d = int(i)
           if d % 4 == 0:
                   print i
   ```

2. **40 Minutes JSON**
   - **a.** What is JSON?
     - i. JavaScript Object Notation
     - ii. a lightweight format that is used for moving data.
     - iii. JSON is built on two structures:
       1. A collection of name/value pairs. In various languages, this is realized as an object, record, struct, dictionary, hash table, keyed list, or associative array.
       2. An ordered list of values. In most languages, this is realized as an array, vector, list, or sequence.
   - b. Why JSON?
     - i. Really quick to access because no joins
     - ii. Flexible and allows for structural changes easily
     - iii. Great for read only access APIs
     - iv. Most of the reasons why not SQL
   - c. Simple JSON Example:

     http://www.person.com/api/?firstname=jason&lastname=bourne

     ```
     {
             "age" : "24",
             "hometown" : "Missoula, MT",
             "gender" : "male"
     }
     ```
   - d. Real fake example: https://randomuser.me/api/?results=1&nat=EN
   - e. Queries Can return Arrays:

     http://www.person.com/api/?lastname=bourne

     ```
     [{
     ```

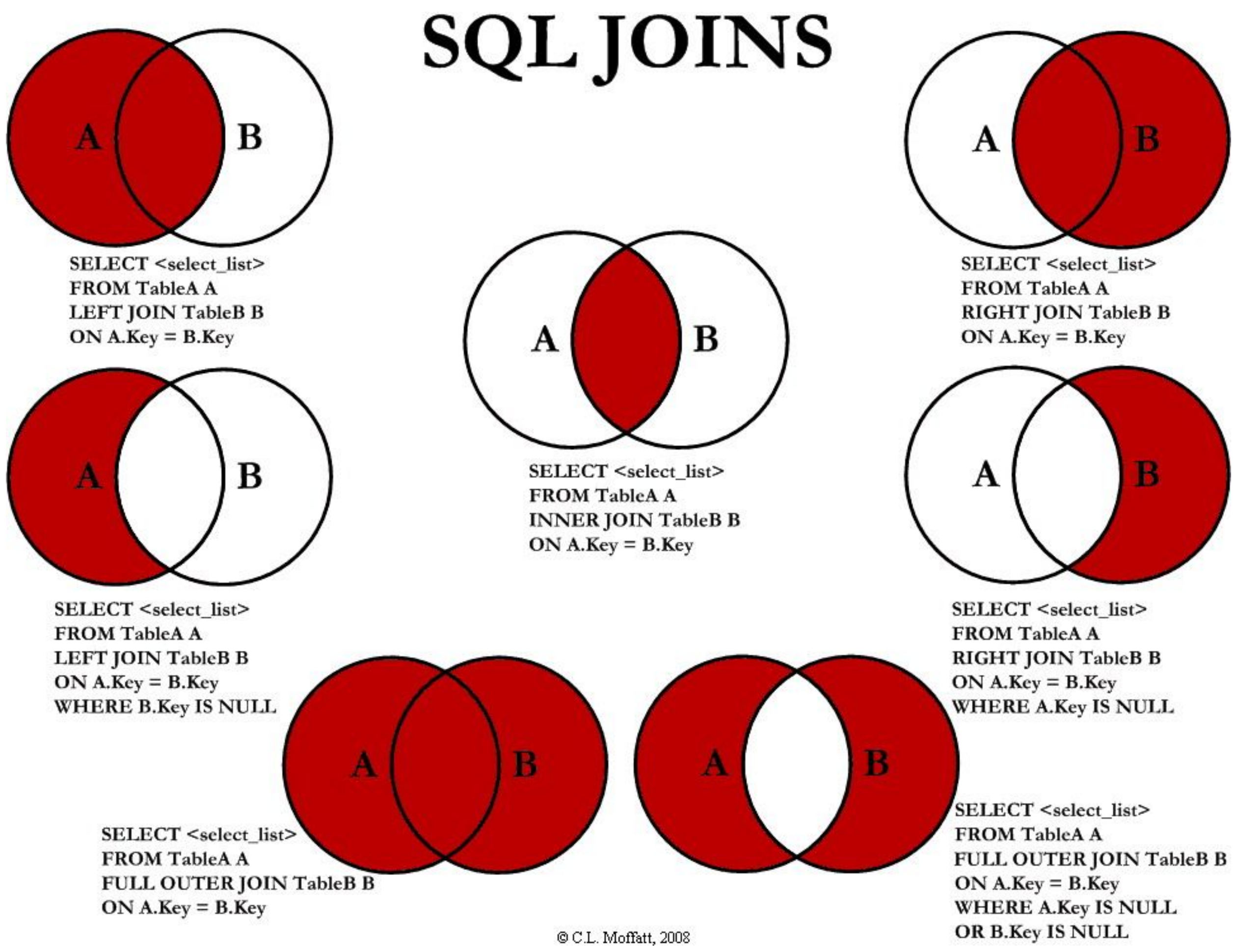
```
            "name" : "Jason Bourne",
            "age" : "24",
            "gender" : "male"
    },{
            "name" : "Kyle Bourne",
            "age" : "21",
            "gender" : "male"
    }]
```

f. Real fake example: https://randomuser.me/api/?results=5&nat=EN
g. Examples on how to use and manipulate JSON with python

```
import json
import requests
import collections

output_file = "output.txt"

data = requests.get("https://randomuser.me/api/?",
params={'results':'10', "nat":'english'}).json()

results = []

#how to access list
print data['results']
print data['results'][0]
print data['results'][0]['gender']
print data['results'][0]['name']['first']

#how to iterate over a list
for i in data['results']:
     print i['gender']

#how to enumerate through a list
for i, val in enumerate(data['results']):
     print i, val['gender']

#how to count from a list
results = []
for i in data['results']:
     results.append(i['gender'])
print results

counter=collections.Counter(results)
```

```
final = list(counter.items())
print final

#how to sort a list
names = []
for i in data['results']:
     names.append(i['name']['last'])
print names
print sorted(names)

#how to sort a list using a function
names = []
for i in data['results']:
     person = []
     person.append(i['name']['last'])
     person.append(i['name']['first'])
     person.append(i['location']['city'])
     names.append(person)
print names

def getKey(item):
     return item[2]

print sorted(names, key=getKey)
```

3. **30 Minutes --SQL Queries**
    a. CREATE TABLE EXAMPLE
        i. Designating Primary Keys, Foreign Keys, Incrementing
            1. Add Primary Keys and Foreign Keys to EXAMPLE
    b. *Write the syntax for a table from the example*
    c. Dropping Tables
        i. DROP TABLE EXAMPLE
    d. Simple SQL Queries
        i. SELECT * FROM table_name WHERE column = value
        ii. QUERY EXAMPLE
    e. Joining SQL Tables



```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```

```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```

```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```

```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```

```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```

```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```

Examples:

```
select courseId, class, count(studentId) from Student_Courses sc
     join Courses c on c.id = sc.courseId
     group by courseId
     order by courseId;

select studentId, firstName, lastName, birth, grad, courseId, class,
department from Student_Courses sc
     join Students s on s.id = sc.studentId
     join Courses c on c.id = sc.courseId
     join Department d on d.id = c.departmentId
     order by studentId, courseId;
```