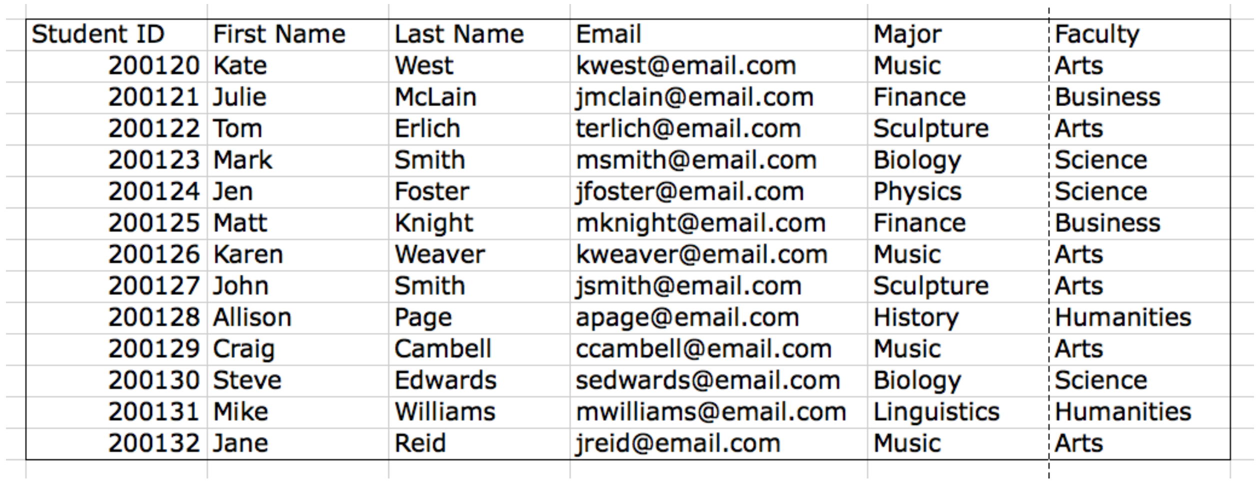
Reading and Writing from a File

Databases:

Databases are made up of files, or tables. Each file is made up of records. Each record is made up of fields. Each field is made up of characters.

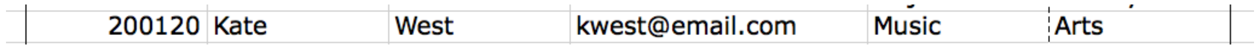
Database File/Table:



Student ID	First Name	Last Name	Email	Major	Faculty
200120	Kate	West	kwest@email.com	Music	Arts
200121	Julie	McLain	jmclain@email.com	Finance	Business
200122	Tom	Erlich	terlich@email.com	Sculpture	Arts
200123	Mark	Smith	msmith@email.com	Biology	Science
200124	Jen	Foster	jfoster@email.com	Physics	Science
200125	Matt	Knight	mknight@email.com	Finance	Business
200126	Karen	Weaver	kweaver@email.com	Music	Arts
200127	John	Smith	jsmith@email.com	Sculpture	Arts
200128	Allison	Page	apage@email.com	History	Humanities
200129	Craig	Cambell	ccambell@email.com	Music	Arts
200130	Steve	Edwards	sedwards@email.com	Biology	Science
200131	Mike	Williams	mwilliams@email.com	Linguistics	Humanities
200132	Jane	Reid	jreid@email.com	Music	Arts

Record:

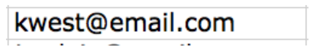
In this database, there are 13 records. One example is:



200120	Kate	West	kwest@email.com	Music	Arts

Field:

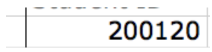
In this database, there are 6 fields. Each record is divided into fields.



kwest@email.com

Primary Key

In a database, each record has a primary key which is a field that uniquely identifies the record.



200120

File Storage

There are two methods of storing files, they are random access files and sequential files. The method we will be looking at is sequential.

Sequential File
Sequential files work in the same way a tape cassette does. To play the fifth song, you need to fast forward past the first four. Similarly with a sequential file, in order to read the third record, you need to read the first and second record as well.

Random Access Files
Random access files work in the same way a CD does. The fifth record can be read without needing to read the previous records.

For your work in this class, please save any data files with extension .dat.

Code for Sequential Files

COMMAND	EXAMPLE	EXPLANATION
open	numFile = open(“in.dat”, “r”)	The variable numFile now represents in.dat. “r” - file is only for reading “w” - file will be written to only and if it exists before hand, it is completely wiped out
readline()	text = numFile.readline()	Will read in the next line from the file and assign it to the variable text
close()	numFile.close()	Will close the file. Never good to leave open files around. May not read properly next time.
write(info)	numFile.write()	Will write whatever info is to the next line in the file.

Example #1

Write a program that accepts positive numbers from the user until they enter a negative number. All valid numbers are written to a file called in.dat.

```
# Writing to a file
numFile = open ("in.dat", "w")
user = True
while user == True:
    num = int(input("Please enter a positive number. (-1 to finish)"))
    if num < 0:
        user = False
    else:
        numFile.write(str(num) + "\n")  #\n forces the input onto another line

numFile.close()
```

Example #2

Write a program that reads in the numbers from the previous file in.dat and displays them.

```
# Reading from a file
numFile = open("in.dat", "r")

while True:
    text = numFile.readline()
    #rstrip removes the newline character read at the end of the line
    text = text.rstrip("\n")
    if text=="":
        break
    print (text, end = "\t")


numFile.close()
```

Exercises:

1) Write a program that reads from the file in.dat from the examples and outputs the numbers doubled.

2) Write a program that reads a file called ages.dat and finds the average of the ages. Create ages.dat in an editor like notepad with the following data:

```
Billy 16
Jimmy 15
Janet 16
Jeremiah 17
Minah 14
Karen 15
```

Hint: do a search for **split** for breaking the fields apart
e.g. values = text.split(“ “) # breaks up data into elements of the list values

3) Write a program that reads an unknown number of records from a file called file.dat in which each record has the following fields:

First Name - alphanumeric of up to 20 characters
Last Name - alphanumeric of up to 20 characters
Math Mark - integer between 0 and 100
English Mark - integer between 0 and 100
Science Mark - integer between 0 and 100
Computers Mark - integer between 0 and 100

Output the information in appropriate columns. Calculate the overall average for each student.

4) Write a program that writes a random number of random numbers between 1 and 1000 to a file called random.dat.

5) Write a program that reads an unknown number of numbers from the file called random.dat. Output how many of the numbers are:

a) prime numbers
b) composite numbers (not primes)

c) ugly numbers (any number who has only the prime factors of 2, 3 and 5)