CSC 365 - Databases
Lab 1 part 2

Team members: Cole Cummins, Jasmine Patel

Initial Decisions:

Programming Language: Python3
Environment: local machine

We debated using Pandas dataframes, but that would have been excessive considering the scope of the project. We used GitHub for collaboration.

Changes to Part1:

We decided to start from scratch with part 2 of the lab. We then parsed through list.txt and teachers.txt. Because we were reading from two files instead of one, we created a function that read one file and then called that function once for list.txt and one for teachers.txt.

Internal Architecture:

As in part 1, we chose to create a "Student" class that represents one student. A student has the following fields:

- first name
- last name
- Grade
- Classroom
- Bus
- Gpa

We then formed student objects as well as a dictionary to map teachers to their respective classrooms. We created a function called prompt_loop which uses a while True loop to manage the recurring prompt. Inside the loop there is a switch statement that contains all of the logic for each user input. Some of the queries were the same as the previous lab so we just copied them over where they fit in and made some minor adjustments.

Additions (Analytics):

For our analytics query, we found the average gpa for every variation of the attribute. For example, 'Analytics: Bus' would return a list of [bus_number] : [average_gpa]

Task log:

Task	Who	Start Time	End Time	How Long (approx in hours)
Import code from schoolsearch part 1	Cole	11:00 am	11:45 am	45 min
Finish Analytics commands	Cole	11:45 am	12:15 pm	30 min
Add additional search commands	Cole	12:15 pm	12:45 pm	30 min
Work on write-up	Jasmine	7:00 am	7:45 am	45 min
Update test script	Jasmine	7:45 am	8:15 am	30 min

Notes on Testing:

	Jasmine	Cole
When	When writing tests	Each new command
How long	~15	~10 mins
How many bugs	1	5-6
Total fixing time	5 min	20 mins

Final Notes:

Having two initial tables rather than one made calculations easier. Each teacher was only stored once instead of once per student, so that was more efficient. The prompt_loop function was more complex with the additional requirements. In general, this program was cleaner and more organized than Part1.