lh3119: HW0 and HW1

Step 3: Test File Import

Replace the UNI in the steps with your UNI.

In [17]:
```
import lh3119_HW0
```

In [18]:
```
lh3119_HW0.t1()
```

Out[18]:
```
'lh3119 says Hello World'
```

The text above should look like my example, but with you UNI.

Note: Any time you change the underlying Python file, you must restart the kernel using the menu. You must then re-import and rerun any cells.

Step 4: Install PyMYSQL and iPython-SQL

- You run the commands below in an Anaconda terminal window.
- Install `pymysql` in your Anaconda environment.
- Install `iPython-SQL` in your Anaconda environment.
- Restart the notebook Kernel.
- The following cell should execute.

In [21]:
```
import pymysql
pymysql.__version__
```

Out[21]:
```
'1.0.2'
```

- In the cell below, replace `dbuser:dbuserdbuser` with your MySQL user ID and password.

In [22]:
```
%load_ext sql

%sql mysql+pymysql://root:dvuserdvuser@localhost
```

Out[22]:
```
The sql extension is already loaded. To reload it, use:
  %reload_ext sql
'Connected: root@None'
```

- The following is a simple test. You should get similar results, but your might be slightly

different.

In [23]:
```
%sql show tables from information_schema
```

```
 * mysql+pymysql://root:***@localhost
79 rows affected.
```

Out[23]:

Tables_in_information_schema
ADMINISTRABLE_ROLE_AUTHORIZATIONS
APPLICABLE_ROLES
CHARACTER_SETS
CHECK_CONSTRAINTS
COLLATION_CHARACTER_SET_APPLICABILITY
COLLATIONS
COLUMN_PRIVILEGES
COLUMN_STATISTICS
COLUMNS
COLUMNS_EXTENSIONS
ENABLED_ROLES
ENGINES
EVENTS
FILES
INNODB_BUFFER_PAGE
INNODB_BUFFER_PAGE_LRU
INNODB_BUFFER_POOL_STATS
INNODB_CACHED_INDEXES
INNODB_CMP
INNODB_CMP_PER_INDEX
INNODB_CMP_PER_INDEX_RESET
INNODB_CMP_RESET
INNODB_CMPMEM
INNODB_CMPMEM_RESET
INNODB_COLUMNS
INNODB_DATAFILES
INNODB_FIELDS
INNODB_FOREIGN
INNODB_FOREIGN_COLS
INNODB_FT_BEING_DELETED

Tables_in_information_schema
INNODB_FT_CONFIG
INNODB_FT_DEFAULT_STOPWORD
INNODB_FT_DELETED
INNODB_FT_INDEX_CACHE
INNODB_FT_INDEX_TABLE
INNODB_INDEXES
INNODB_METRICS
INNODB_SESSION_TEMP_TABLESPACES
INNODB_TABLES
INNODB_TABLESPACES
INNODB_TABLESPACES_BRIEF
INNODB_TABLESTATS
INNODB_TEMP_TABLE_INFO
INNODB_TRX
INNODB_VIRTUAL
KEY_COLUMN_USAGE
KEYWORDS
OPTIMIZER_TRACE
PARAMETERS
PARTITIONS
PLUGINS
PROCESSLIST
PROFILING
REFERENTIAL_CONSTRAINTS
RESOURCE_GROUPS
ROLE_COLUMN_GRANTS
ROLE_ROUTINE_GRANTS
ROLE_TABLE_GRANTS
ROUTINES
SCHEMA_PRIVILEGES
SCHEMATA
SCHEMATA_EXTENSIONS
ST_GEOMETRY_COLUMNS
ST_SPATIAL_REFERENCE_SYSTEMS
ST_UNITS_OF_MEASURE

Tables_in_information_schema
STATISTICS
TABLE_CONSTRAINTS
TABLE_CONSTRAINTS_EXTENSIONS
TABLE_PRIVILEGES
TABLES
TABLES_EXTENSIONS
TABLESPACES
TABLESPACES_EXTENSIONS
TRIGGERS
USER_ATTRIBUTES
USER_PRIVILEGES
VIEW_ROUTINE_USAGE
VIEW_TABLE_USAGE
VIEWS

Step 5: Load Sample Data

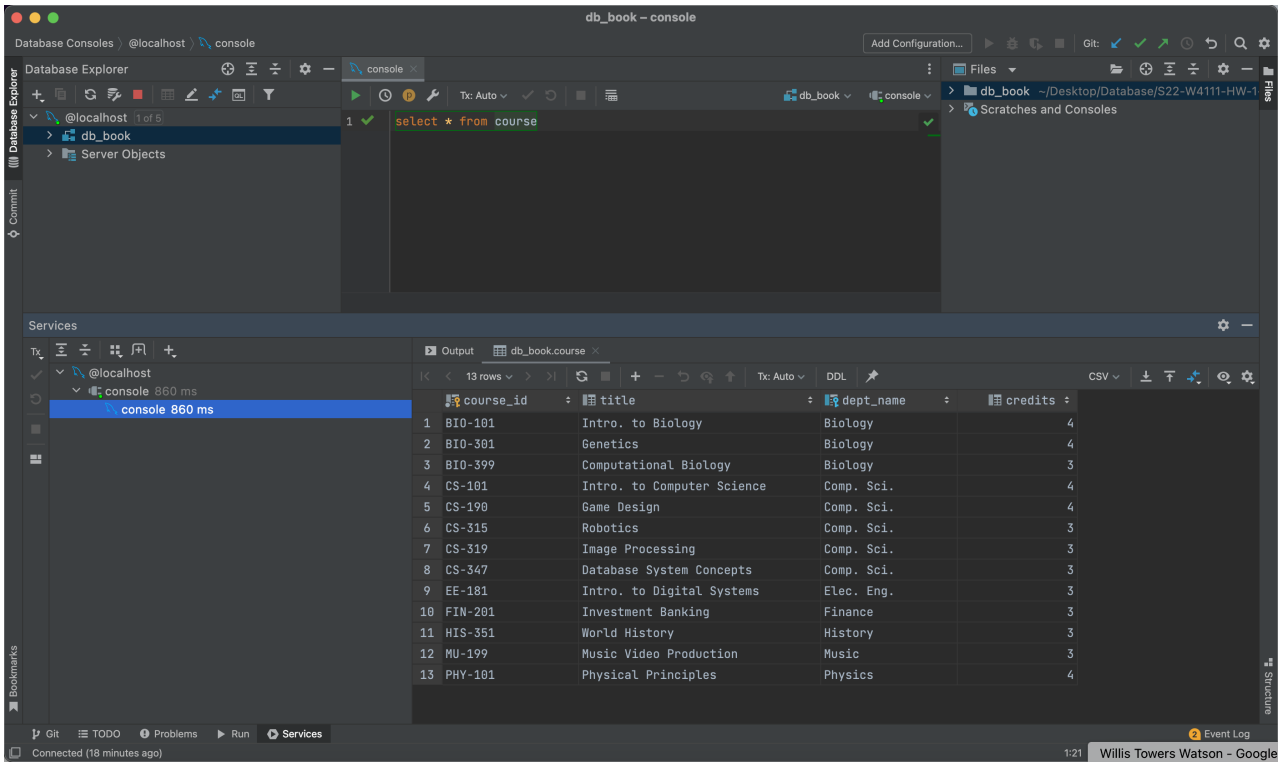
- In the directory where you cloned the project, there is a sub-folder `db_book.`
- Start DataGrip.
- In DataGrip, choose `File->New DataSource->MySQL.`
  - Accept the default name for the data source.
  - Set the MySQL user ID and password.
  - You may see a message stating that you need to install database drives. Install the drivers.
- Select the newly created data source. The name will `Run SQL Script` . Navigate to and choose the file `DDL_drop.sql` .
- Do the same for `smallRelationsInsertFile.sql` .
- You will see an icon/text on the side bar labelled `db_book.` It may be greyed-out. Right click on the entry and choose `New query console.` You may see a message `Current schema not introspected` and `Introspect schema` on the far right. Click on `Introspect schema.`
- Enter `select * from course` in the query console window. Click on the little green arrow to run the query.

- Take a screen show of your DataGrip window and save the screen show into the folder of the form `dff9_src` using your UNI. Remember the name of the file.
- Set your file name in the cell below replacing the example and run the cell. You should see your screenshot below. Yours will look a little different from mine. As long as yours shows the query result, you are fine.

In [24]:
```
file_name = 'queryresult.png'

print("\n")
from IPython.display import Image
Image(filename=file_name)
```

Out[24]:



Step 6: Very %sql

- Execute the cell below. Your answer will be similar to mine but may not match exactly.

In [25]:
```
%sql select * from db_book.course
```

```
 * mysql+pymysql://root:***@localhost
13 rows affected.
```

Out[25]:

course_id	title	dept_name	credits
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4

course_id	title	dept_name	credits
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3
EE-181	Intro. to Digital Systems	Elec. Eng.	3
FIN-201	Investment Banking	Finance	3
HIS-351	World History	History	3
MU-199	Music Video Production	Music	3
PHY-101	Physical Principles	Physics	4

Step 7: Pandas, CSV and SQL

- Run the cell below.

In [26]:
```
import pandas
pandas.__version__
```

Out[26]:
```
'1.3.4'
```

- Install SQLAlchemy using an Anaconda prompt.
- Restart the notebook kernel and rerun all cells. Then run the cell below.

In [27]:
```
from sqlalchemy import create_engine
```

- Go into DataGrip. Select your local database, e.g. `@localhost` .
- Open a query console and execute `create database lahmansdb` . Then execute the cell below.

In [28]:
```
%sql show tables from lahmansdb;
```

```
 * mysql+pymysql://root:***@localhost
1 rows affected.
```

Out[28]:

Tables_in_lahmansdb
people

- There is a folder `data` in the project you cloned. There is a file in the folder `People.csv` .
- Execute the following code cell. If you are on Windows, you may have to change the path to the file and may have to replace `/` with `\\` in paths.

- You should see a result similar to mine below.

In [29]:
```
df = pandas.read_csv('../../data/People.csv')
df
```

Out[29]:

	playerID	birthYear	birthMonth	birthDay	birthCountry	birthState	birthCity	deathYear
0	aardsda01	1981.0	12.0	27.0	USA	CO	Denver	NaN
1	aaronha01	1934.0	2.0	5.0	USA	AL	Mobile	2021.0
2	aaronto01	1939.0	8.0	5.0	USA	AL	Mobile	1984.0
3	aasedo01	1954.0	9.0	8.0	USA	CA	Orange	NaN
4	abadan01	1972.0	8.0	25.0	USA	FL	Palm Beach	NaN
...	...	...	...	...	...	...	...	...
20353	zupofr01	1939.0	8.0	29.0	USA	CA	San Francisco	2005.0
20354	zuvelpa01	1958.0	10.0	31.0	USA	CA	San Mateo	NaN
20355	zuverge01	1924.0	8.0	20.0	USA	MI	Holland	2014.0
20356	zwilldu01	1888.0	11.0	2.0	USA	MO	St. Louis	1978.0
20357	zychto01	1990.0	8.0	7.0	USA	IL	Monee	NaN

20358 rows × 24 columns

- We will now save the data to MySQL. Run the cells below. You will have to change `dbuser:dbuserdbuser` to your MySQL user ID and password.

In [30]:
```
engine = create_engine("mysql+pymysql://root:dvuserdvuser@localhost")
```

In [31]:
```
df.to_sql('people', con=engine, index=False, if_exists='replace', schema='lahman
```

- Test that you wrote the information to the databases.

In [32]:
```
%sql select * from lahmansdb.people where nameLast='Williams' and bats='L'
```

```
 * mysql+pymysql://root:***@localhost
19 rows affected.
```

Out[32]:

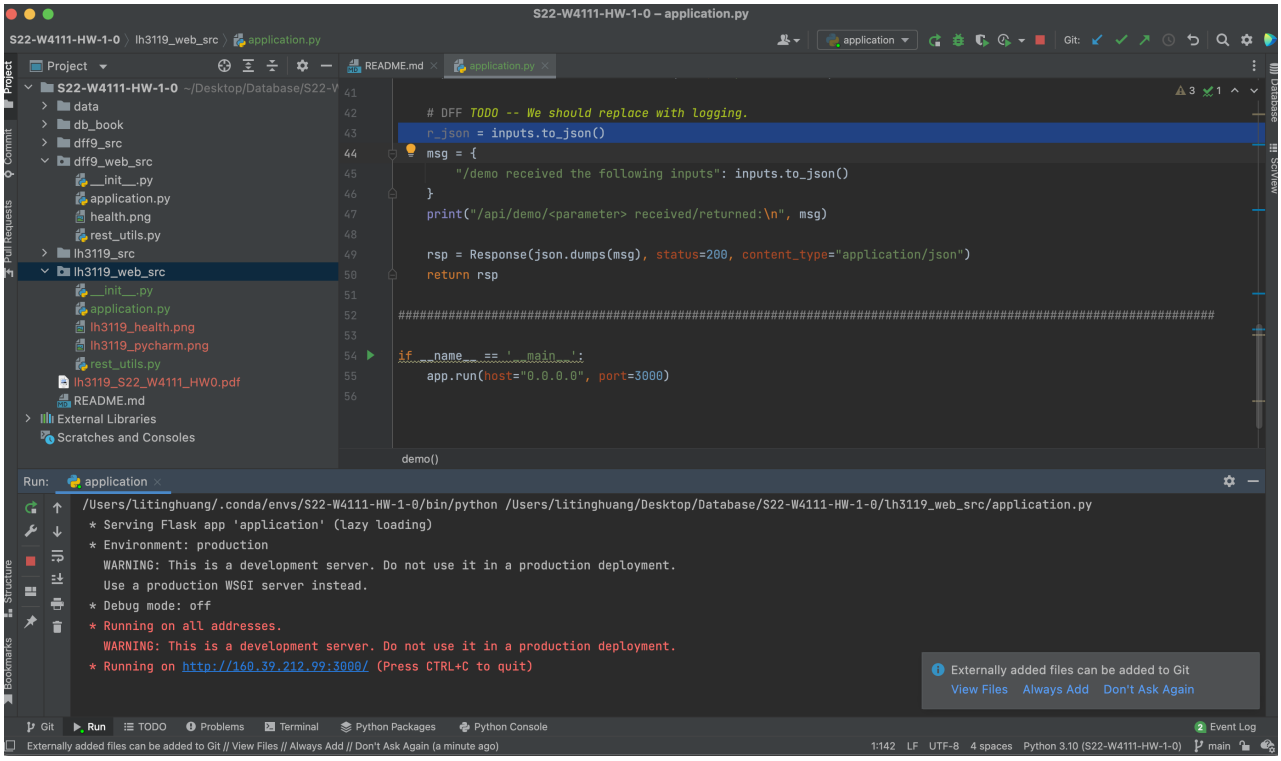
playerID	birthYear	birthMonth	birthDay	birthCountry	birthState	birthCity	deathYear	deathM
williar01	1877.0	8.0	24.0	USA	MA	Somerville	1941.0	
willibi01	1938.0	6.0	15.0	USA	AL	Whistler	None	
willibi02	1932.0	6.0	13.0	USA	SC	Newberry	2013.0	
willicy01	1887.0	12.0	21.0	USA	IN	Wadena	1974.0	
willida05	1958.0	2.0	28.0	USA	NY	Brooklyn	None	
willida07	1979.0	3.0	12.0	USA	AK	Anchorage	None	
willide01	1896.0	12.0	13.0	USA	OR	Portland	1929.0	
willigu02	1888.0	5.0	7.0	USA	NE	Omaha	1964.0	
williju02	1995.0	8.0	20.0	USA	LA	Houma	None	
willike01	1890.0	6.0	28.0	USA	OR	Grants Pass	1959.0	
willile03	1905.0	12.0	2.0	USA	GA	Macon	1984.0	
willima02	1953.0	7.0	28.0	USA	NY	Elmira	None	
willima07	1991.0	8.0	21.0	USA	RI	Pawtucket	None	
willimi02	1964.0	11.0	17.0	USA	CA	Santa Ana	None	
willini01	1993.0	9.0	8.0	USA	TX	Galveston	None	
willira01	1975.0	9.0	18.0	USA	TX	Harlingen	None	
williri02	1893.0	12.0	18.0	USA	CA	Santa Cruz	1966.0	
willist01	1892.0	1.0	31.0	USA	MT	Cascade	1979.0	
willite01	1918.0	8.0	30.0	USA	CA	San Diego	2002.0	

Step 7: Done

- You are done.

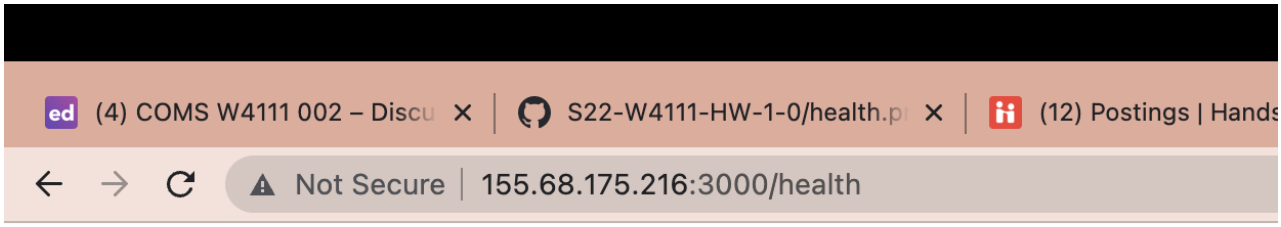
In [33]:
```
file_name = 'lh3119_pycharm.png'
print("\n")
from IPython.display import Image
Image(filename=file_name)
```

Out[33]:



In [34]:
```
file_name = 'lh3119_health.png'
print("\n")
from IPython.display import Image
Image(filename=file_name)
```

Out[34]:



{"status": "healthy", "time": "2022-01-25 20:19:06.902813"}

In [ ]:

In [ ]:

In [ ]: