Introduction to SQLAlchemy

Data Boot Camp

Lesson 10.1

© 2019 Trilogy Education Services, Inc.

Class Objectives

By the end of today's class you will be able to:

Connect to a SQL database with SQLAlchemy.

Perform a SQL query with SQLAlchemy.

Create Python classes and objects.

Use a Python class to model a SQL table.

Activity: Looking Into SQLAlchemy

In this activity you will break into groups of two or three and research a few questions...

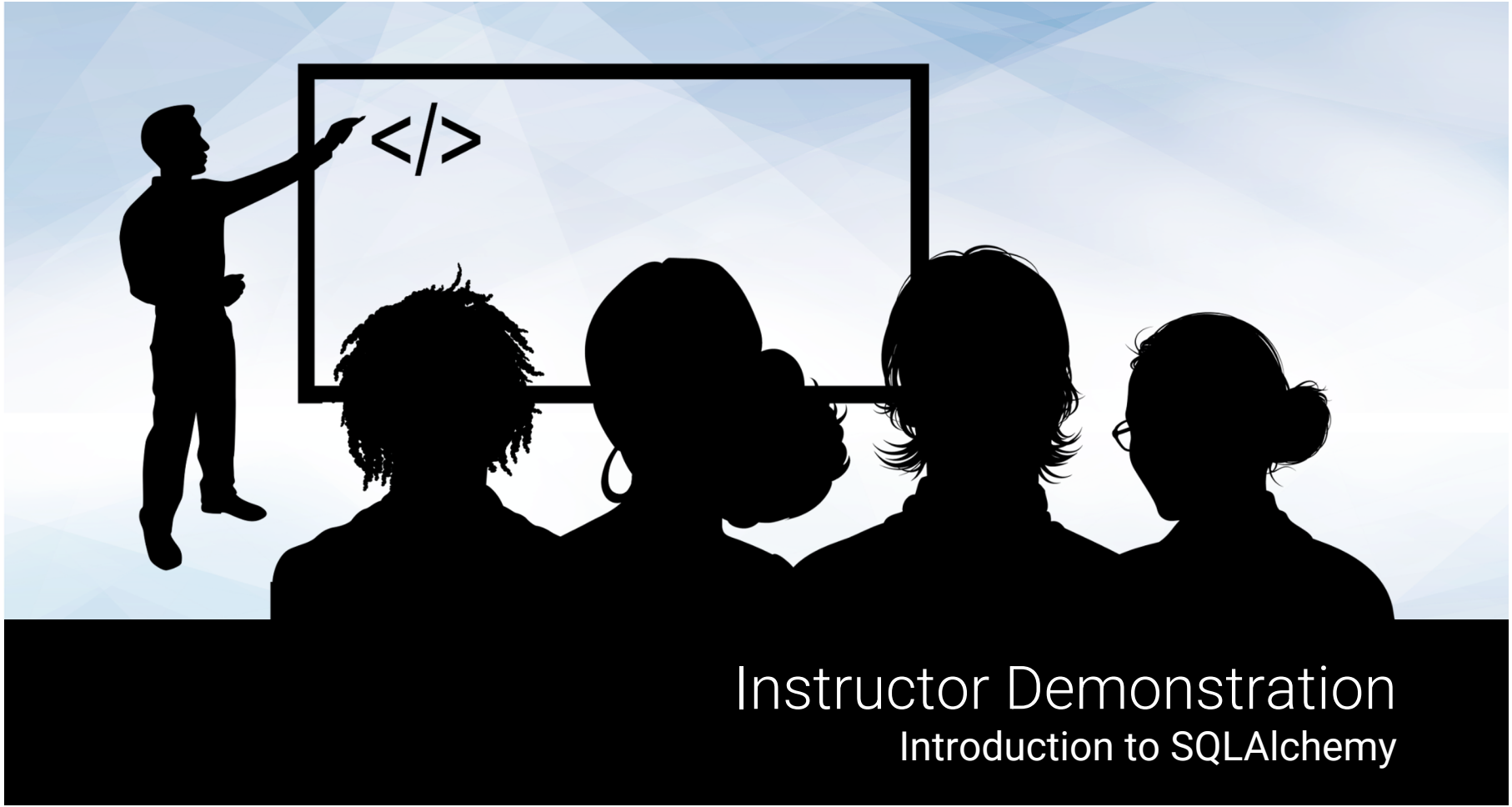
(Instructions sent via Slack.)

Suggested Time:
3 Minutes

Looking Into SQLAlchemy Instructions

- Within your group, take a few minutes to research the answers to the following:
  - What is an ORM?
  - What are some of the benefits to using an ORM?
  - What are some of the disadvantages of using an ORM?

Time’s Up! Let’s Review.



Instructor Demonstration

Introduction to SQLAlchemy

SQLAlchemy is a Python library that works across a variety of SQL dialects.

Write the query once, run it anywhere!

SQLAlchemy ORM Is Flexible

It’s possible to query a database using more SQL...

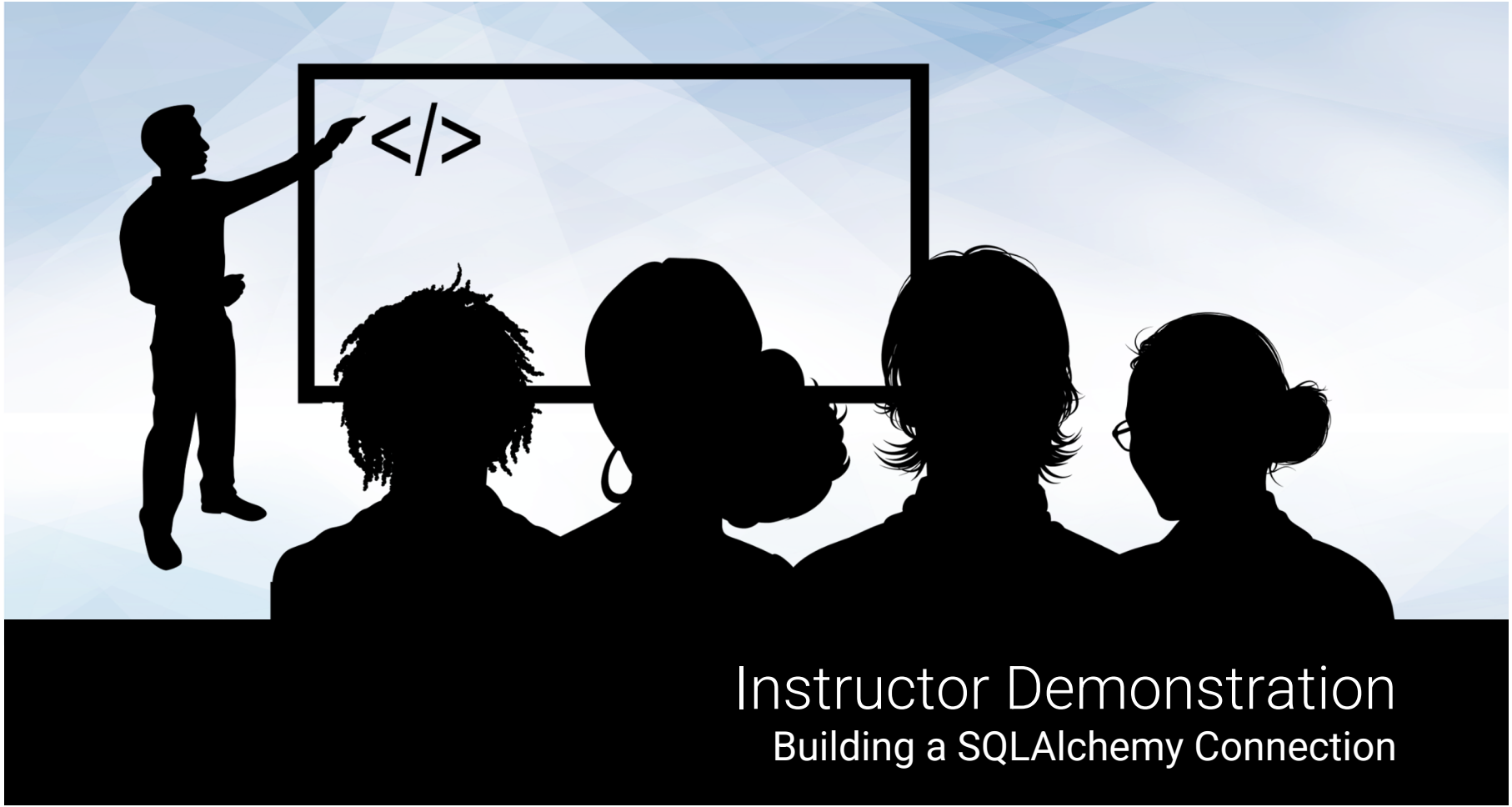
```
data = engine.execute("SELECT * FROM BaseballPlayer")
```

...or more Python!

```
players = session.query(BaseballPlayer)
for player in players:
  print(player.name_given)
```

Looking at SQLAlchemy Documentation

Let’s take a moment to look at the SQLAlchemy documentation!

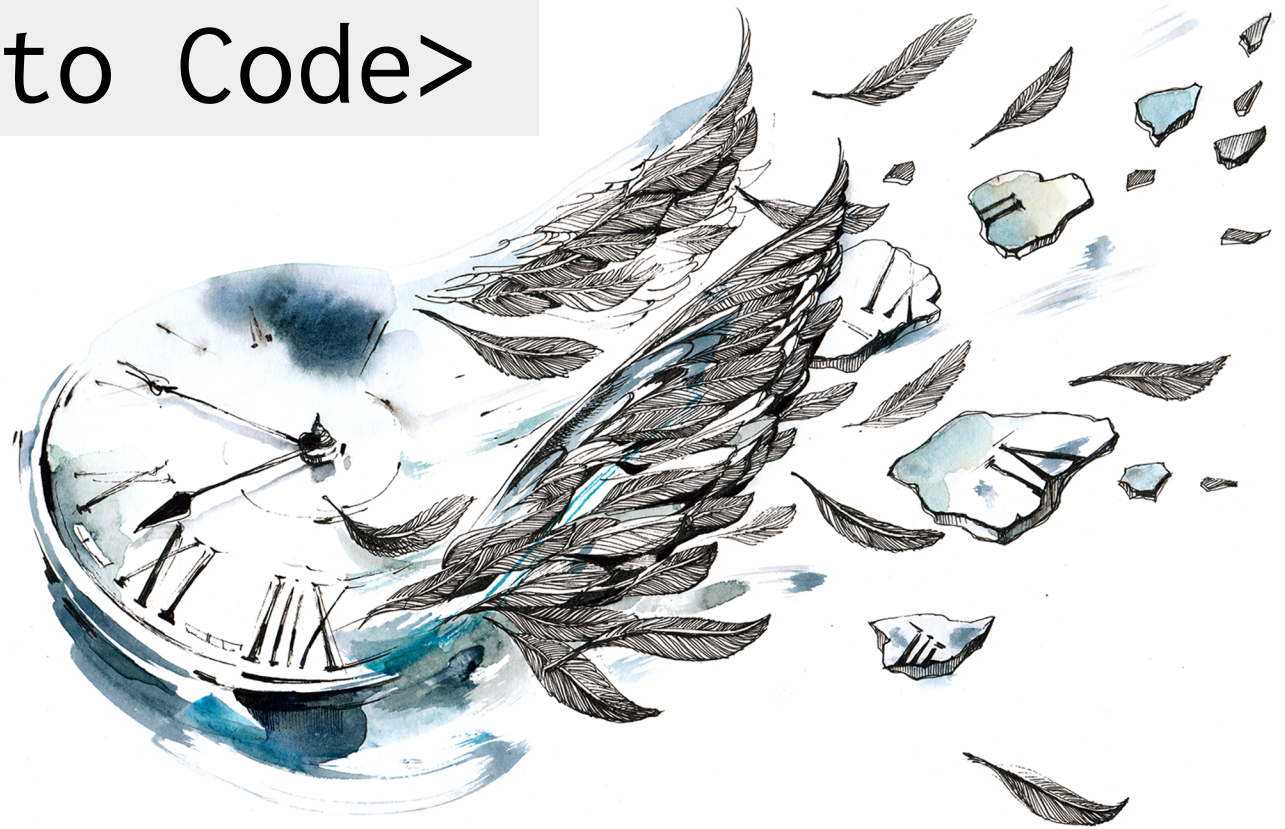


Instructor Demonstration

Building a SQLAlchemy Connection

Today we will only be working with one SQL dialect - **SQLite**!

<Time to Code>



Activity: Ice Cream Connection

In this activity, you will create, connect and insert data into a new database using SQLAlchemy.

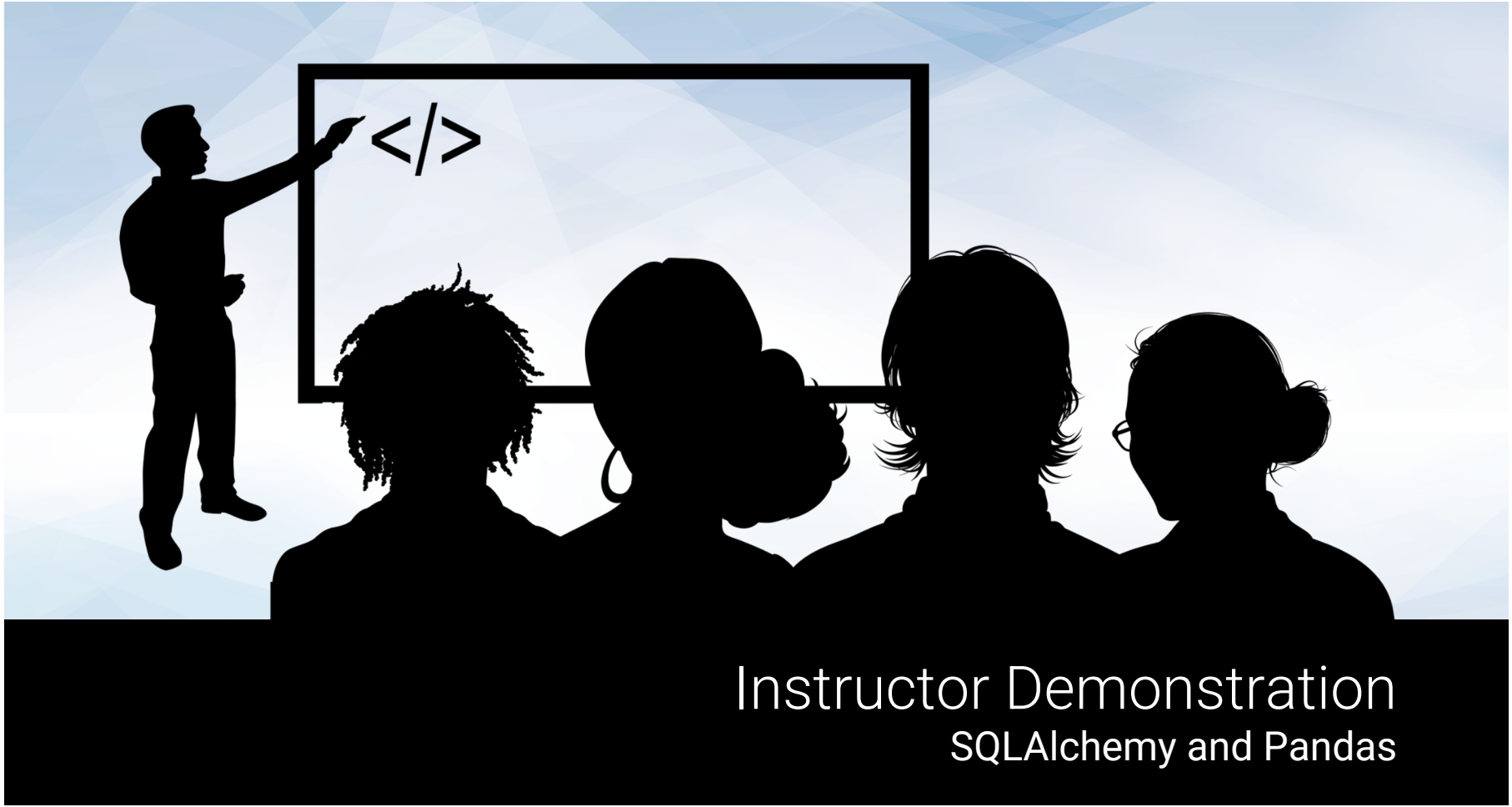
(Instructions sent via Slack.)

Suggested Time:
15 Minutes

Ice Cream Connection Instructions

- Use the database path to create a sqlite engine
- Use the engine to select all of the rows and columns from the table icecreamstore.csv
- Create a new query that finds the ice cream flavors that cost $1.25 or greater

Time’s Up! Let’s Review.



Instructor Demonstration

SQLAlchemy and Pandas

One of the most impressive aspects of **SQLAlchemy...**

...is how it integrates with **Pandas!**

Pandas integrates with SQLAlchemy

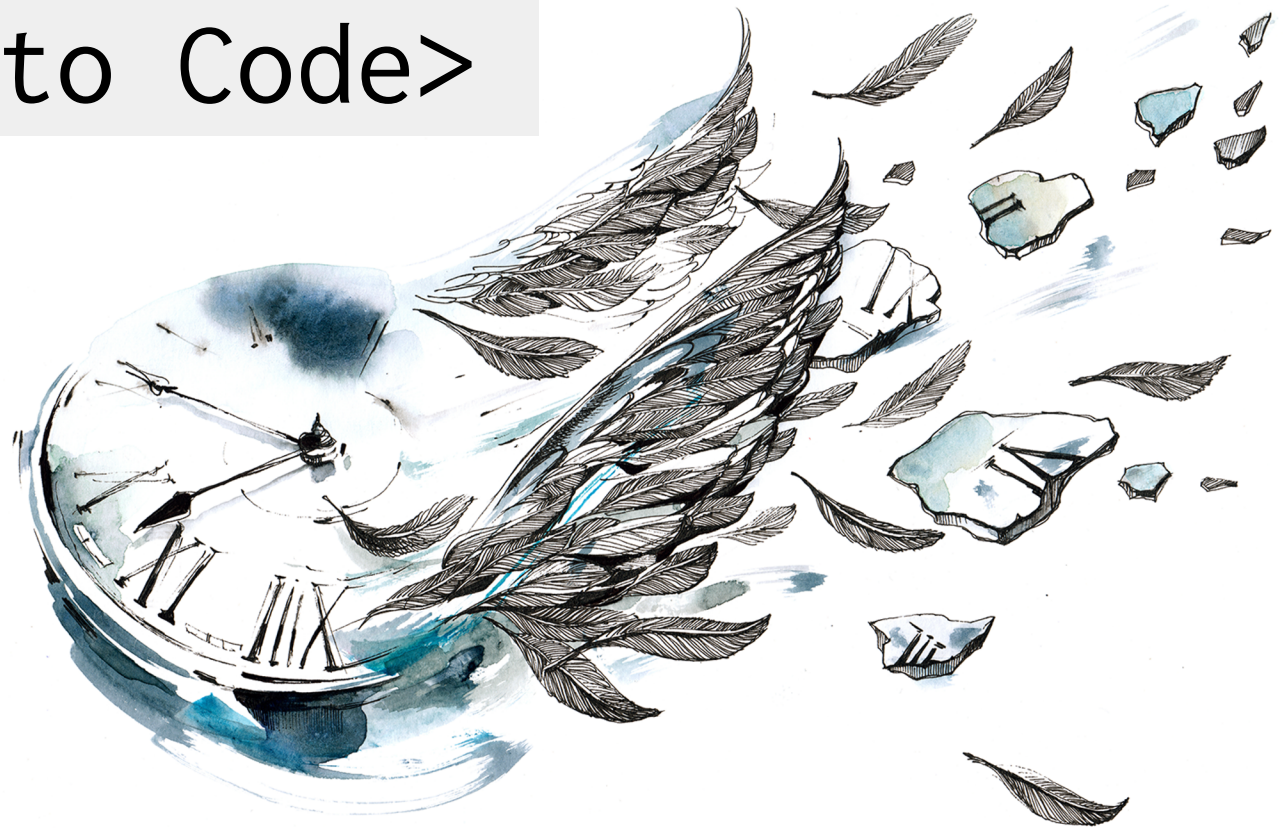
- Once we connect to our SQL database using SQLAlchemy

- We can query directly using pandas

```
# Create Engine
engine = create_engine(f"sqlite:///{database_path}")
conn = engine.connect()
```

```
# Query All Records in the the Database
data = pd.read_sql("SELECT * FROM Census_Data", conn)
```

<Time to Code>



Activity: Read All the SQL

In this activity, you query an external server using Pandas and SQLAlchemy to create new dataframes based on US census data.

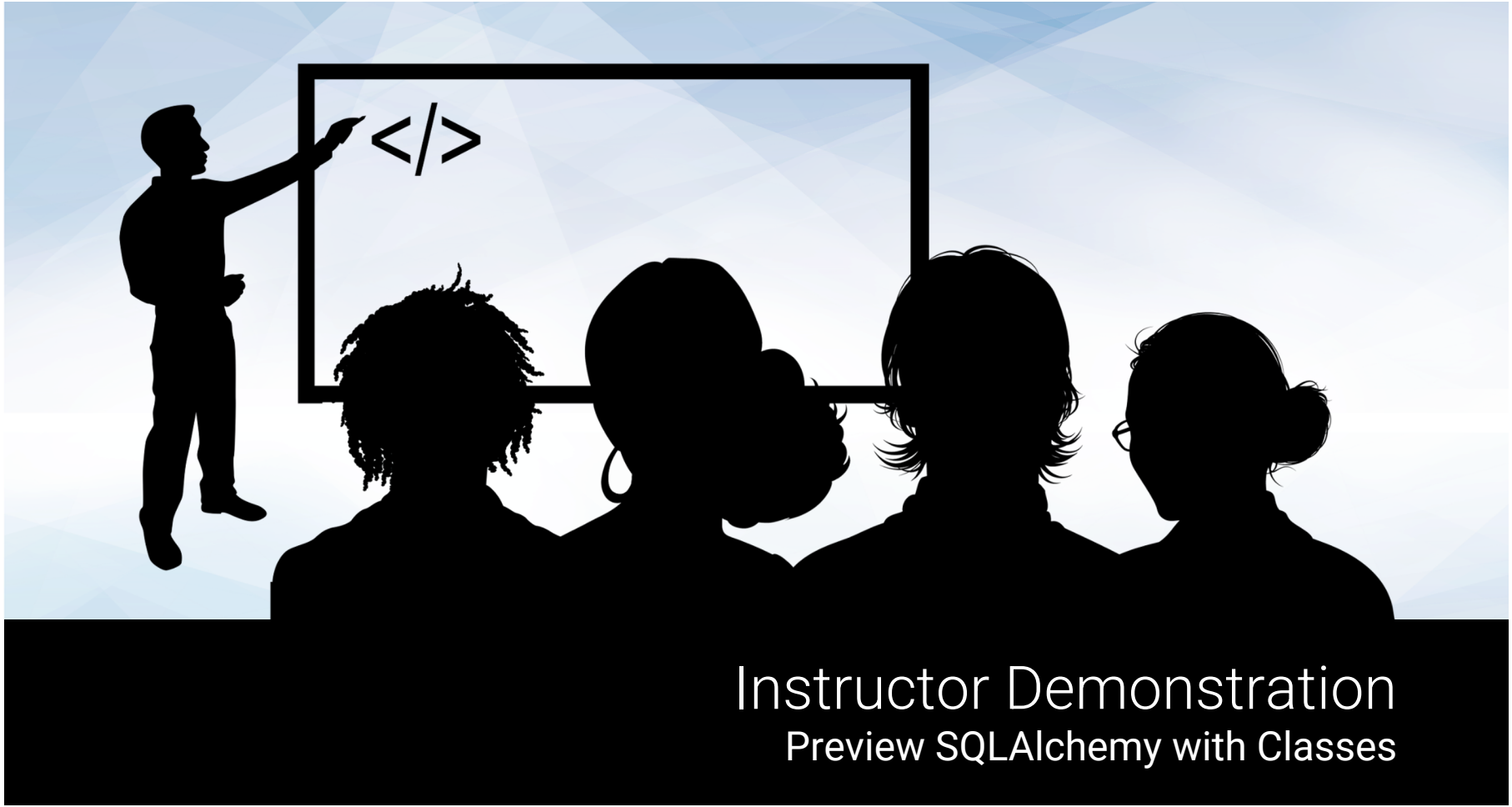
(Instructions sent via Slack.)

Suggested Time:
10 Minutes

Read All the SQL Instructions

- Create an engine to connect to the census database.
- Query all the data from the Census_Data table and load into pandas.
- Create an engine to connect to the zip database.
- Query all the data from the Zip_Census table and load in pandas.
- Show the .head() of your newly imported data.

Time’s Up! Let’s Review.



Instructor Demonstration

Preview SQLAlchemy with Classes

SQLAlchemy with Classes

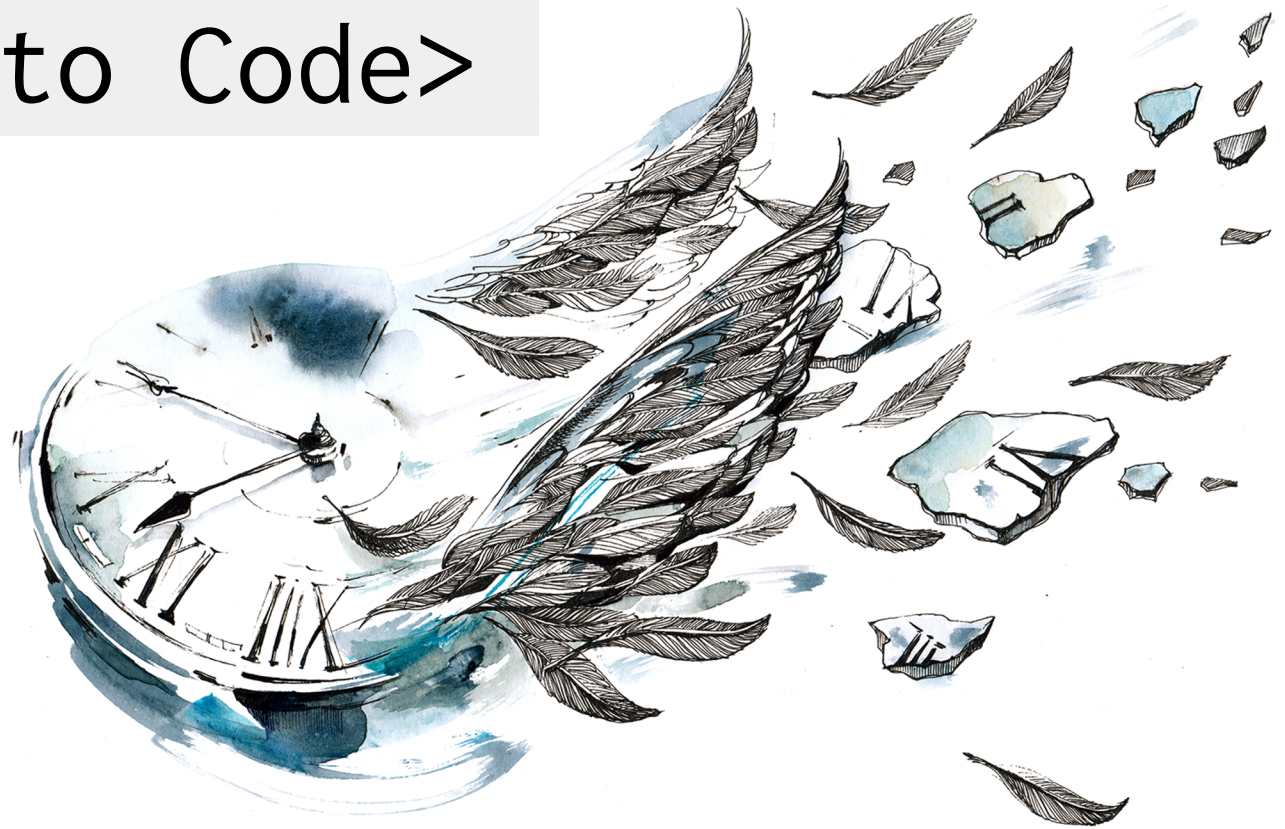
- SQLAlchemy is not just for making SQL queries in Python
  - It can also **update** a SQL database using Python classes

- Python classes are traditionally used to bundle data and functions together
  - In SQLAlchemy they are used to define structures

```
# Create Dog and Cat Classes
# ----------------------------------
class Dog(Base):
    __tablename__ = 'dog'
    id = Column(Integer, primary_key=True)
    name = Column(String(255))
    color = Column(String(255))
    age = Column(Integer)
```

```
class Cat(Base):
    __tablename__ = 'cat'
    id = Column(Integer, primary_key=True)
    name = Column(String(255))
    color = Column(String(255))
    age = Column(Integer)
```

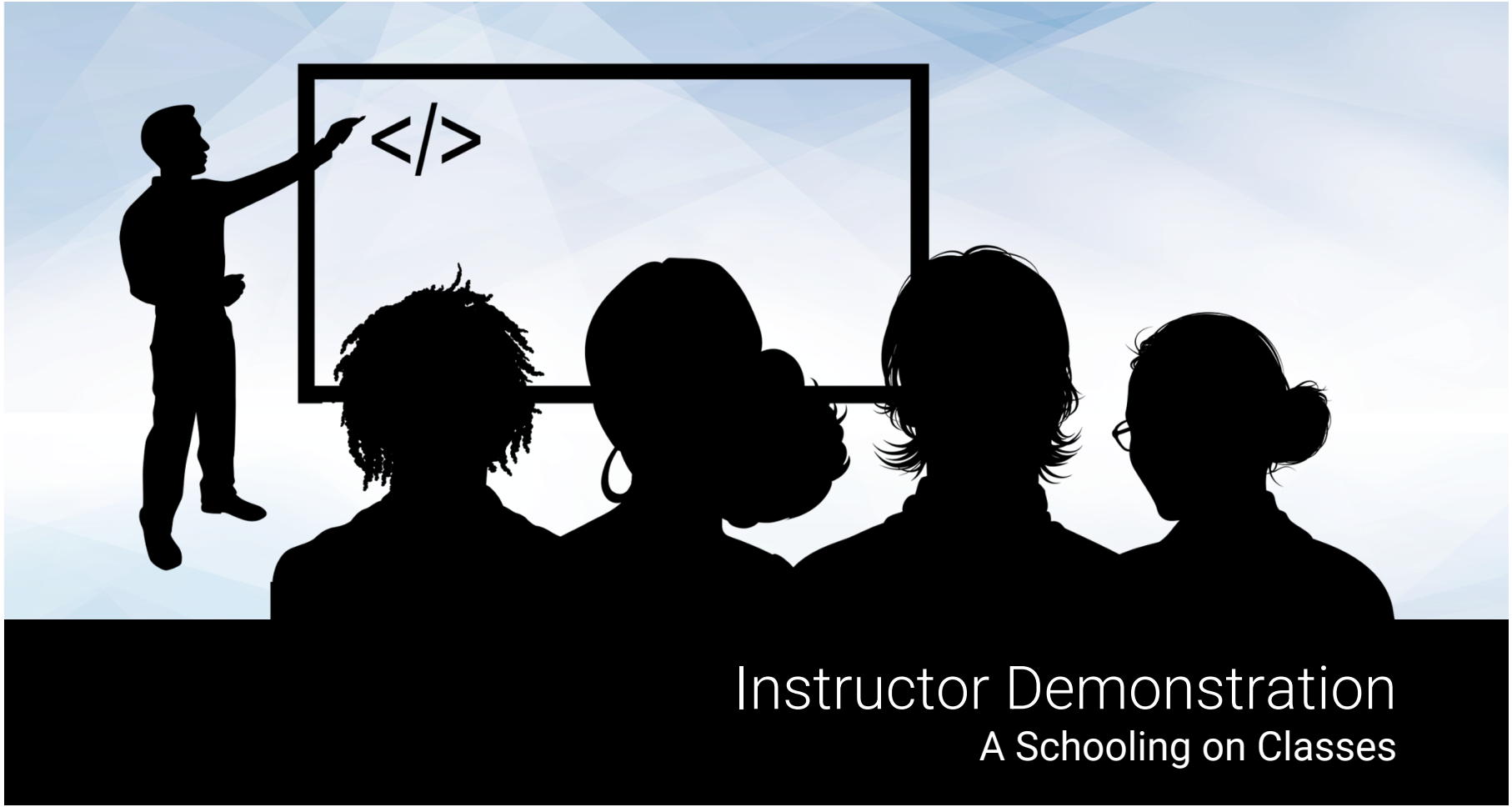
<Time to Code>



Countdown timer

15:00

(with alarm)



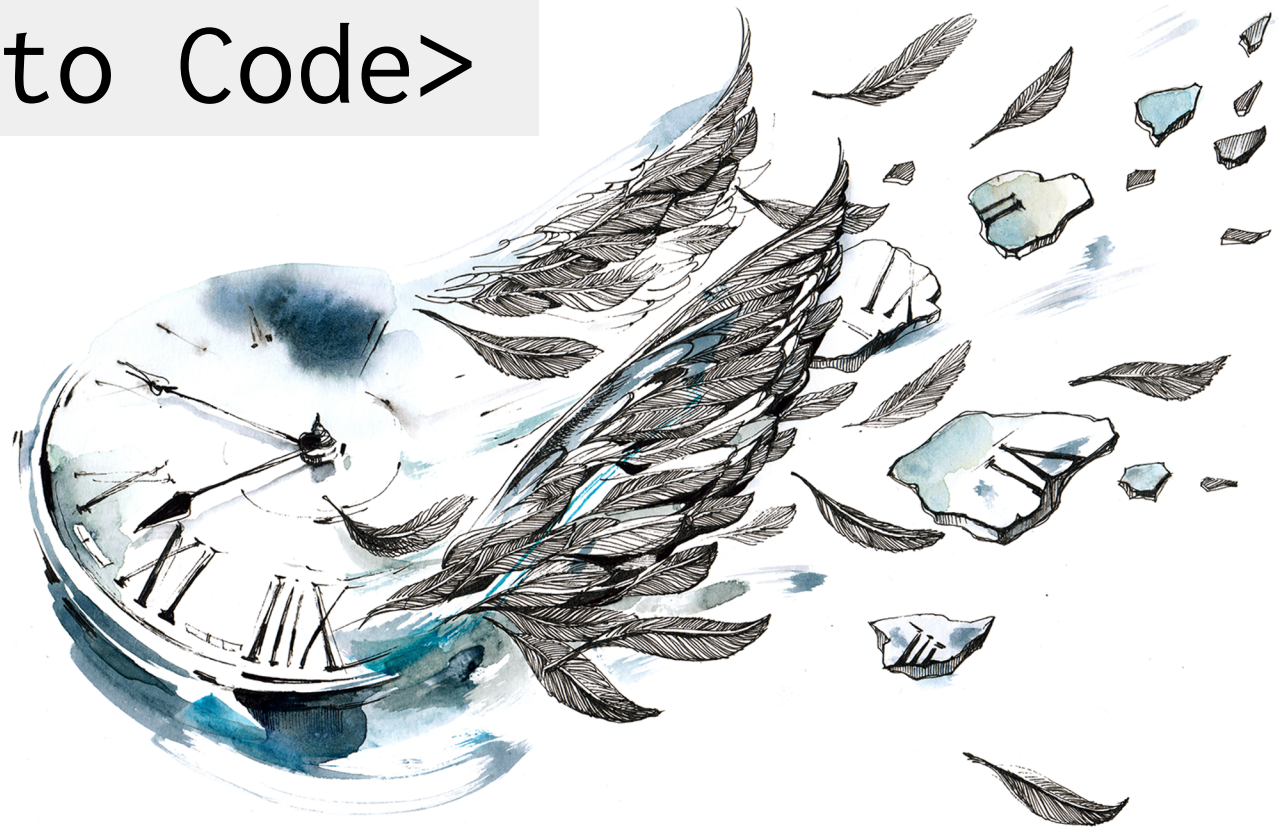
Instructor Demonstration

A Schooling on Classes

Time For a Crash Course in Programming!

- Object oriented programming
  - Style of coding based around “objects”
- Objects may contain:
  - Attributes (data)
  - Methods (functions)
- Python is an object oriented programming language
  - Classes are used to interact and create objects
  - Makes code more reproducible/ adaptable

<Time to Code>



Activity: Surfer Class

In this activity, you will work on creating your own classes in Python.

(Instructions sent via Slack.)

Suggested Time:
15 Minutes

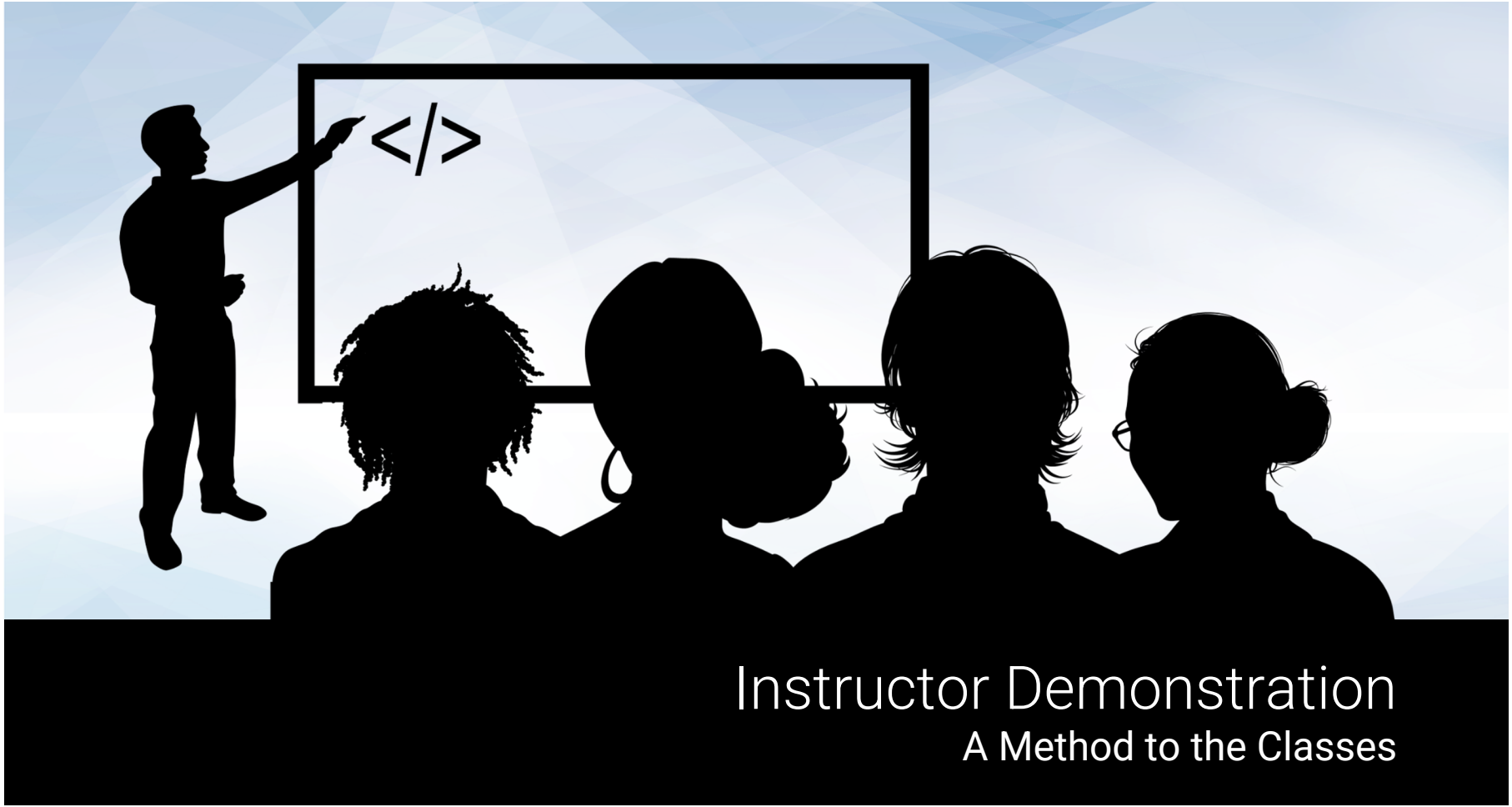
Surfer Class Instructions

- Create a class Surfer and initialize it with name, hometown, and rank instance variables.
- Create an instance of a surfer.
- Then print the name, hometown and rank of your surfer object.

Bonus:

- Create a while loop that will allow you to continuously create new instances of surfers using input().
- Keep the loop going until the user specifies otherwise.

Time’s Up! Let’s Review.



Instructor Demonstration

A Method to the Classes

Adding Methods to Python Classes is Easy as 1, 2, 3!

1. Define the function using `def`

1. Provide a name and list of parameters

1. Use `class.method()` to run the method in your script!

```
# Define the Expert class
class Expert():

    # A required function to initialize the class object
    def __init__(self, name):
        self.name = name

    # A method that takes another object as its argument
    def boast(self, obj):

        # Print out Expert object's name
        print("Hi. My name is", self.name)

        # Print out the name of the Film class object
        print("I know a lot about", obj.name)
        print("It is", obj.length, "minutes long")
        print("It was released in", obj.release_year)
        print("It is in", obj.language)
```

<Time to Code>

Activity: Surfer Class Extended

In this activity, you will be reworking your Surfer script from earlier as you add in methods to perform specific tasks.

(Instructions sent via Slack.)

Suggested Time:
10 Minutes

Surfer Class Extended Instructions

- Create a Surfer class that has name, hometown, rank, and wipeouts instance variables.
- Create a method called speak that prints "Hangs loose, bruh!"
- Create a method called biography that prints the surfer's name and hometown.
- Create a method called cheer that will print "I totally rock man, no wipeouts!" if the surfer has no wipeouts. Otherwise, it prints 'Bummer bruh, keep on keeping on!'.
- Create two surfers that print out all their info and run all the methods.

Time’s Up! Let’s Review.

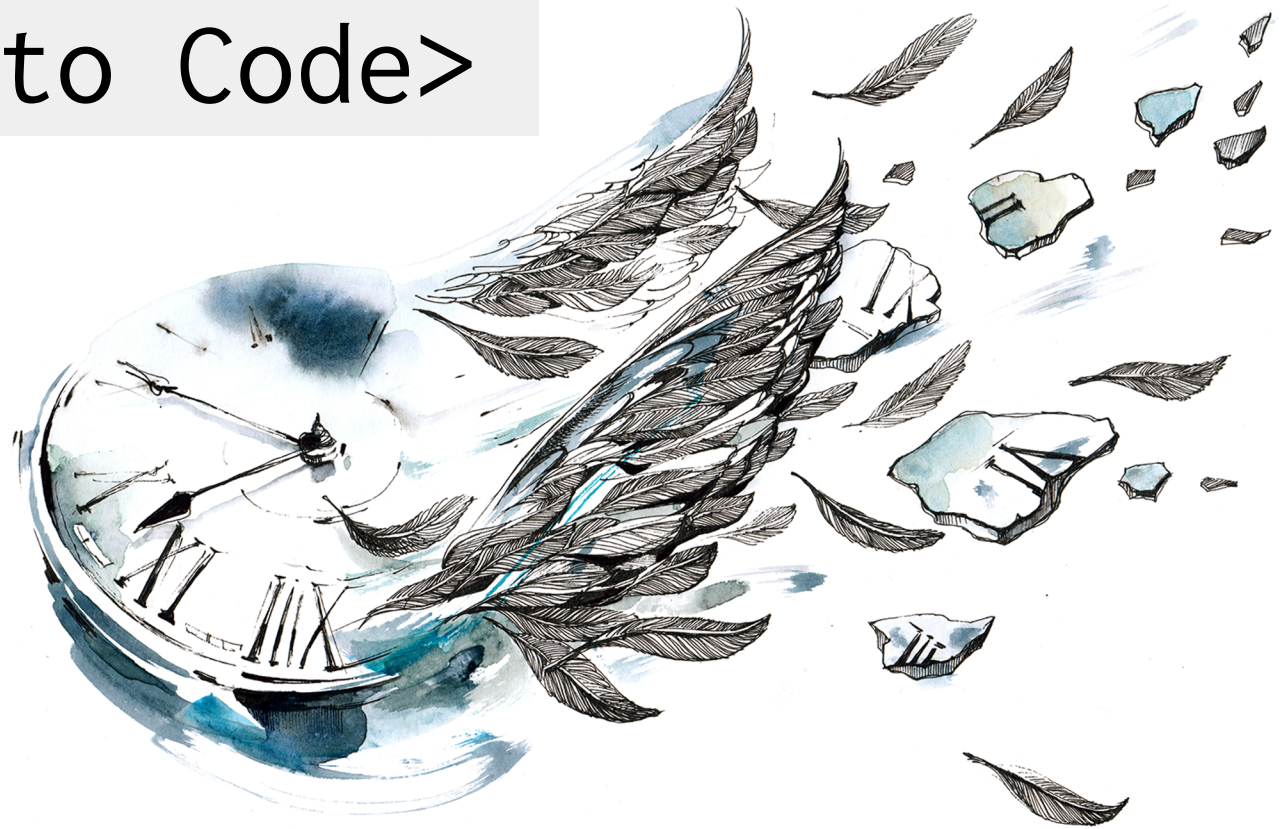
Everyone Do: Back to the SQL

In this activity, we will bring all of our SQLAlchemy concepts together and remake the pets database using classes.

(Instructions sent via Slack.)

Suggested Time:
20 Minutes

<Time to Code>



Activity: Surfing SQL

In this final activity, you will test your SQLAlchemy skills and update your Surfer database.

(Instructions sent via Slack.)

Suggested Time:
20 Minutes

Surfing SQL Instructions

- Modify the Surfer class created during the previous activity so that it will function with SQLAlchemy.
- Create a new class called Board which will function with SQLAlchemy and has the following parameters: __tablename__, id, surfer_id, board_name, color, length
- Pull a list of all of the surfers and surfboards already inside the database
- Push a new surfer and surfboard to the tables on the database

Time’s Up! Let’s Review.