MIDS W205

Storing and Retrieving Data

Instructors:

Jari Koister, jari@ischool.berkeley.edu

Arash Nourian , nourian@ischool.berkeley.edu

Amit Bhattacharyya , amitbl@ischool.berkeley.edu

Uri Schoenfeld, shuri@ischool.berkeley.edu

Edward Fine, edward.fine@gmail.com

Exercises and Labs Overview

In this course we have 2 exercises and 12 labs. The exercises are larger programming assignments spanning multiple weeks. We expect each exercise to take students anywhere from 20 – 40 hours to complete. Labs are smaller assignments that are intended to illustrate a point from a module, or to prepare to help prepare students for the exercises. A lab should take 1-2 hours to complete. Labs are to be completed by students individually. Students can do exercises as group projects. If an exercise is completed as a group project each individual should be prepared to present the exercise to an instructor.

Prerequisites

- Previous experience with Python

Evaluation

1. 10 labs (spread through out the course): [25% of grade]
2. 2 exercises, spanning Weeks 1–7 and 8–14: (20% ex 1, 20% ex 2 respectively) [total 40% of grade]
3. Final project: [total 35% of grade]

Submission

Submissions of Exercises, Labs and Projects should be done according to the following guidelines.

- For submission of labs with questions are to done using ISVC.
- Exercises and Labs with code submissions are done in Github according to instructions below.
- Students will sign-up for private GitHub repository. There are free private repositories provided for students.
- Student will invite one or more instructors to collaborate on repo.
- Submission is done by Student pushing submitted code to repo and alerting instructor that submission is ready for review.

Exercises

Exercise 1: Storage and Retrieval in Hadoop and RDBMS.

In this Exercise, you will implement end to end storage and retrieval Processes, covering the following features:

- Build a Data Model consisting of transactional data and master data.
- Load data into two types of storage systems : Hadoop and Postgres
- Profile data in order to learn about a dataset that you have just received
- Build aggregate datasets for retrievals or reporting
- Benchmark system/tools for your purposes
- Build Capacity Plan for storing operational and aggregate data

Exercise 2: Data Stream Processing and Visualization

In this Exercise, you will implement end to end real time data processing system using twitter data streams, Apache Storm, Redis and Tableau for real time visualization. You will build the followings:

- Understand Twitter Data feed and collect Tweets using Twitter API
- Use Apache Storm to collect and analyze Tweets
- Save the data into Redis
- Visualize Tweets using Tableau

Labs

Lab 1: AWS, EBS, EC2, AMI etc

In this introductory lab we will familiarize ourselves with the environment that we will use for the upcoming labs and exercises in this course. Today, we will learn about the following:

- Amazon EC2 Environment and your Account
- What an AMI Is
- How to Find an AMI and Launch a Server
- How to Choose a Server
- How to Check for Already Installed Software on a Server
- Understand storage options such as EBS, S3 etc.
- Understand the AWS dashboard

Lab 2: Hadoop and Spark

In this lab, we will get introduced to Hadoop and Spark. We will learn the basics in running both systems. We will learn how to troubleshoot and use the systems. We will also review the processes and components that constitute these two systems:

- Running and using Hadoop on AWS.
- Hadoop components, processes and tools.
- Running and using Spark on AWS.
- Spark components, processes and tools.

Lab 3: Hive and Postgres

In this lab, we will go over basic features in Hive and Postgres and how to run these tools on AWS.

- Running, loading data and querying Hive.
- Running, loading data and querying Postgres.

- Architectural components and processes.

Lab 4: Spark basics

In this lab, we will go over Apache Spark features and common commands as below:

- Accessing Spark SQL – CLI
- Accessing through Java database connectivity (JDBC)
- Accessing through Java database connectivity
- Let's create a table and load data using CSV file
- Accessing Spark-SQL in Python Code
- Caching tables and Un-caching tables

Lab 5: Data Warehousing with SQL

Queries are the fundamental way of extracting knowledge from data. In this lab you will write a number of SQL queries to generate reports from data stored in PostgreSQL. Additionally, you will measure the runtime of queries and add additional structures to optimize performance.

Lab 6: Streaming with Apache Storm

In this lab, we will go over data streaming using Apache Storm. Specifically the following features:

- Apache Storm Spout
- Apache Storm Bolt
- Monitor data being generated and analyzed real time

Lab 7: Visualization with Tableau

In this lab, we will go over data visualization using Tableau. We will go over the following features:

- Build a Hive Tables and load data
- Tableau Installation for your desktop
- Connect Tableau from your desktop to your Hive Server in EC2
- Plot a few meaningful Visualization Charts

Lab 8: Data cleaning with OpenRefine

In this lab you will get some hands on experience cleaning data using OpenRefine. You will load data, do basic transformations and try some fuzzy matching features.

Lab 9: Graph Models and Analysis

There are many interesting applications of graph-based processing models. In this lab, we will use Gephi to explore the Marvel Comics Universe social graph. We will examine important characters in the universe, as well as plot visualizations of the graph data.

Lab 10: Streaming with Apache Spark

In this lab, we will go over streaming data management and analysis, using Apache Spark. We will go over the following features:

- Build a stream generator for the lab
- Run Apache Spark to extract and analyze streaming data

Exercise Grading Guidelines

Grading
This guide lays out the criteria used to grade W205 assignments. Please email the instructor if you have any questions about this document. The grading of assignments are broken down into the following categories:

1-Program Correctness: Your code(s) must meet the specifications written in the assignment and function correctly as accepted. If a specification is ambiguous or unclear, you either make a reasonable assumption about what is required, based on what makes the most sense to you, or you can ask the instructor. If you make an assumption about an ambiguous specification, you should mention that in your architecture design document.

2- *Architecture Design*: You need to prepare a document (max one page) explaining all the relevant information as mentioned in the Table 1.

3- *Resiliency/Documentation /Readability*: Your code needs to consider all the relevant possible errors, exceptions, interrupts, and framework limitations by providing sufficient handlers for such cases (Resiliency). You code needs to be readable to both you and a knowledgeable data scientist (Readability). For example, adding whitespaces where appropriate to help separate distinct parts of the code or giving variables meaningful names. Your code should be well organized such as defining functions in one section of the program separated from the main program. Your code(s) should start with a header comment containing your name and a brief description of what the code does. Use comments to explain your code (Documentation). Any section of your code whose purpose isn't completely clear (not to you, but rather to someone else reading your code) should have a comment explaining it.

4-Submission Specifications: Assignments will usually contain specifications for submission as some of them are mentioned in Table 1. Other requirements may be asked as well, so please read the assignments carefully.

Due Date
To receive full credit for your assignment, the assignment must be turned in by 11:59PM on the due date.

Late Policy
Every student has up to 2 late days he may use for any assignment throughout the semester. For example, a student may hand in Assignment 1 one day late and Assignment 2 one day late, but then all the remaining assignments must be handed in by the deadline. Assignments that are turned in after the due date (assuming you used your 2 late days) will be penalized 15 % of the points assigned per day and no assignment is accepted more than two days late. For example if you are two days late an above the budget, you still can get up to 70% credit for an assignment.

Grading Standard

Criterion	%	Expectations
Program Correctness	80%	-Program runs correctly and meets the assignment's specification(s). -Program provides all the required outputs as outlined in the assignment's specification
Architecture Design	5%	-Providing a document, explaining the high-level design of the program, assumptions made, possible use of external libraries, how to run the program, requirements and related issues that need to be considered by the instructor
Resiliency Documentation Readability	10%	-Addressing the possible errors/exceptions/interrupts/framework limitations (like rate limit) that may occur. -Code is clean, understandable, and well organized. -Code is well commented providing concise and meaningful comments
Submission Specifications	5%	-Providing the submission requirements as specified -Correct Github pull request with sufficient permissions

		-Correct S3 bucket address and permissions if required by the assignment -Readable outputs (figures)

Honor Code and Collaboration Policy

Students are encouraged to discuss the assignments and final project implementation with other students; it's fine to discuss overall strategy, as both giving and receiving advice will help students to learn. However, students must their own code and they should not share code or write code collaboratively. If they do not do so, this will be considered a violation of the UC Berkeley Honor Code - ASUC. If students consult any resources outside of the materials provided in class, they must cite these sources. Otherwise, they receive a penalty if their codes are substantially derivative.

Appeals

Instructors sometimes can miss something. If you discover such issues when you review the comments, you should contact the instructor directly.