**Course introduction**

Welcome to the course! You've begun one of the most complete overviews on data science tooling that you’ll currently find on the internet. This doesn’t mean that we cover each and every tool, but later in the course we’ll introduce a comprehensive list of tasks a data scientist needs to perform and give you the top two or three open source and commercial tools available to complete them. We also explain how the tools overlap in functionality, what their pros and cons are, and how these tools can address the whole data science pipeline. Let’s start with data. Data is obviously central to data scientists. In this course, we’ll show you how to manage, extract, transform, analyze, and visualize data. Now, you might be able to survive data science without programming skills if you use the right set of tools. However, we highly recommend getting familiar with programming and the related programming frameworks for data science. To help you along, we’ll introduce you to the most commonly used programming languages and frameworks available for data science. That said, there is a lot of automation available in the latest tooling that a data scientist can use. In this course, we’ll explain how to make use of those tools to save time and uncover inspiration. Visual programming is available in many tools. In this course, you’ll learn how visual programming can be used to speed up development time and to help non-programmers enter the field of data science. Open source software is leading the field of data science, but its total costs of ownership, or "TCO," can be higher at times due to configuration, customization and maintenance costs. As a result, commercial software also has its place, especially since the new generation of commercial data science software leverages open source software and open standards. This makes it easy to migrate between tools and can reduce overall TCO. In this course, we’ll introduce you to both open source and commercial software and point out their strengths and weaknesses for data science. We'll also show you ways that you can take advantage of their respective strengths. Finally, we'll show you how cloud computing can be used to further speed up and facilitate data scientists' work. We'll introduce you to the most commonly used and newly emerging cloud tools for data science. In addition to lectures, this course, has numerous labs to make you more familiar with the material and get hands-on experience. There are also multiple quizzes to test your learning. Nothing more to say than we’re glad to have you in the course and happy learning. In case you have trouble in any way, please don’t hesitate to contact us in the discussion forum. There's nothing left but to begin! We're very happy to have you with us as you start your data science journey. If you have any trouble with any of the course material, please don’t hesitate to contact us in the discussion forum. Let's get started!

**Languages of data science**

The languages of Data Science For anyone just getting started on their data science journey, the range of technical options can be overwhelming. There is a dizzying amount of choice when it comes to programming languages. Each has it's own strengths and weaknesses and there is no one right answer to the question of which one you should learn first. The answer to that question depends largely on your needs, the problems you are trying to solve, and who you are solving them for. Python, R, and SQL are the languages that we recommend you consider first and foremost. But there are so many others that have their own strengths and features. Scala, Java, C++, and Julia are some of the most popular. Javascript, PHP, Go, Ruby, and Visual Basic all have their own unique use cases as well. The language you choose to learn will depend on the things you need to accomplish and the problems you need to solve. It will also depend on what company you work for, what role you have, and the age of your existing application. We’ll explore the answers to this question as we dive into the popular languages in the data science industry. There are many roles available for people who are interested in getting involved in data science. Business Analyst Database Engineer Data Analyst Data Engineer Data Scientist Research Scientist Software Engineer Statistician Product Manager Project Manager and many more. Let’s dive into what we will learn in Lesson 1. We will put most of our focus on the top three Data Science languages: Python, R, and SQL, which each have their own lessons. Then we will go on to highlight other noteworthy languages and what makes them special. Then we’ll finish with a short quiz to test your knowledge! ● 1.1.1 - Python ● 1.1.2 - R ● 1.1.3 - SQL ● 1.1.4 - Other Noteworthy Data Science Languages ● 1.1.5 - Practice Quiz

**Introduction to Python**

In this video, we will review the high-level features of the Python programming language. Python is a powerhouse language. It is by far the most popular programming language for data science. According to the 2019 Kaggle Data Science and Machine Learning Survey, 75% of the over 10,000 respondents from around the world reported that they use Python on a regular basis. Glassdoor reported that in 2019 more than 75% of data science positions listed included Python in their job descriptions. When asked which language an aspiring data scientist should learn first, most data scientists say Python. You are probably thinking, why on earth is Python so popular? Well, let’s start with the people who use Python. If you already know how to program, then Python is great for you because it uses clear, readable syntax. You can do many of the things you are used to doing in other programming languages but with Python you can do it with less code. If you want to learn to program, it’s also a great starter language because of the huge global community and wealth of documentation. In fact, several different surveys in 2019 found that over 80% of data professionals worldwide use Python. Python is useful for many situations, including data science, AI and machine learning, web development, and IoT devices like the Raspberry Pi. Large organizations that use Python heavily include IBM, Wikipedia, Google, Yahoo!, CERN, NASA, Facebook, Amazon, Instagram, Spotify, and Reddit. Python is a powerful general-purpose programming language that can do a lot of things. It is widely supported by a global community and shepherded by the Python Software Foundation. 1. Python is a high-level general-purpose programming language that can be applied to many different classes of problems. 2. It has a large, standard library that provides tools suited to many different tasks, including but not limited to databases, automation, web scraping, text processing, image processing, machine learning, and data analytics. 3. For data science, you can use Python's scientific computing libraries such as Pandas, NumPy, SciPy, and Matplotlib. 4. For artificial intelligence, it has TensorFlow, PyTorch, Keras, and Scikit-learn. 5. Python can also be used for Natural Language Processing (NLP) using the Natural Language Toolkit (NLTK). Another great selling point is the Python community, which has a well documented history of paving the way for diversity and inclusion efforts in the tech industry as a whole. The Python language has a code of conduct executed by the Python Software Foundation that seeks to ensure safety and inclusion for all, in both create spaces for people interested in Python to learn in safe and inclusive environments. PyLadies is an international mentorship group with a focus on helping more women become active participants and leaders in the Python open source community.

**Introduction to R language**

In this video, we will give a brief overview of the R programming language. After our last video on Python, where we discussed its wide adoption, you might be wondering why on earth you should consider learning any other language. Well, according to the results of the 2019 Kaggle Data Science survey, which had over 10k respondents from around the world, learning up to three languages can increase your salary! And R has a lot to offer you. Like Python, R is free to use, but it's a GNU project -- instead of being open source, it's actually free software. So if Python is open source and R is free software, what’s the difference? Well, Both open source and free software commonly refer to the same set of licenses. Many open source projects use the GNU General Public License, for example. Both open source and free software support collaboration. In many cases (but not all), these terms can be used interchangeably. The Open Source Initiative (OSI) champions open source while the Free Software Foundation (FSF) defines free software. Open source is more business focused, while free software is more focused on a set of values. Back to why you should learn R. Because this is a free software project, you can use the language in the same way that you contribute to open source, and it allows for public collaboration and private and commercial use. Plus, R is another language supported by a wide global community of people passionate about making it possible to use the language to solve big problems. Who is R for? It's most often used by statisticians, mathematicians, and data miners for developing statistical software, graphing, and data analysis. The language’s array-oriented syntax makes it easier to translate from math to code, especially for someone with no or minimal programming background. According to Kaggle’s Data Science and Machine Learning Survey, most folks learn R when they're a few years into their data science career, but it remains a welcoming language to those who don’t have a software programming background. R is popular in academia but companies that use R include IBM, Google, Facebook, Microsoft, Bank of America, Ford, TechCrunch, Uber, and Trulia. ● R has become the world’s largest repository of statistical knowledge. ● As of 2018, R has more than 15,000 publicly released packages, making it possible to conduct complex exploratory data analysis. ● R integrates well with other computer languages, such as C++, Java, C, .Net, and Python. ● Common mathematical operations such as matrix multiplication work straight out of the box. ● R has stronger object-oriented programming facilities than most statistical computing languages. There are many ways to connect with other R users around the globe. Communities such as user!, WhyR?, SatRdays, and R-Ladies are all great to connect with. And you can also check out the R project website for R conferences and events.

**Introduction to SQL**

In this video, we'll take a high-level look at SQL. SQL is a bit different from the other languages we’ve covered so far. First off, it's formally pronounced “ess cue el,” although some people say “sequel.” While the acronym stands for “Structured Query Language,” many people do not consider SQL to be like other software development languages because it's a non-procedural language and its scope is limited to querying and managing data. While it is not a “data science” language per se, data scientists regularly use it because it's simple and powerful! Another couple of neat facts about SQL: it's much older than Python and R, by about 20 years, having first appeared in 1974. And, SQL was developed at IBM! This language is useful in handling structured data; that is, the data incorporating relations among entities and variables. SQL was designed for managing data in relational databases. Here you can see a diagram showing the general structure of a relational database. A relational database is formed by collections of two-dimensional tables; for example, datasets and Microsoft Excel spreadsheets. Each of these tables is then formed by a fixed number of columns and any number of rows. BUT! Even though SQL was originally developed for use with relational databases, because it's so pervasive and easy to use, SQL interfaces for many NoSQL and big data repositories have also been developed. The SQL language is subdivided into several language elements, including clauses, expressions, predicates, queries, and statements. So what makes SQL great? Knowing SQL will help you do many different jobs in data science, including business and data analyst, and it's a must in data engineering. When performing operations with SQL, you access the data directly. There's no need to copy it beforehand. This can speed up workflow executions considerably. SQL is the interpreter between you and the database. SQL is an American National Standards Institute, or "ANSI," standard, which means if you learn SQL and use it with one database, you will be able to easily apply that SQL knowledge to many other databases. There are many different SQL databases available, including MySQL, IBM Db2, PostgreSQL, Apache OpenOffice Base, SQLite, Oracle, MariaDB, Microsoft SQL Server, and more. The syntax of the SQL you write might change a little bit based on the relational database management system you’re using. If you are looking to learn SQL you would be best served to focus on a specific relational database and then plug into the community for that specific platform. There are also many great introductory courses on SQL available!

**Other languages**

So far, we’ve reviewed Python, R, and SQL. In this video, we will review some other languages that have compelling use cases for data science. Ok, so indisputably, Python, R, and SQL are the three most popular languages that data scientists use. But, there are many, many other languages that are worth your time when considering which language to use to solve a particular data science problem. Scala, Java, C++, and Julia are probably the most traditional data science languages on this slide. But JavaScript, PHP, Go, Ruby, Visual Basic, and others have all found their place in the data science community as well! I won’t dive as deeply into each of these languages, but I'll mention some notable highlights.

**Java** is a tried-and-true general-purpose object oriented programming language. It's been widely adopted in the enterprise space and is designed to be fast and scalable. Java applications are compiled to bytecode and run on the Java Virtual Machine, or "JVM." Some notable data science tools built with Java include Weka, for data mining; Java-ML, which is a machine learning library; Apache MLlib, which makes machine learning scalable; and Deeplearning4j, for deep learning. Apache Hadoop is another Java-built application. It manages data processing and storage for big data applications running in clustered systems. Scala is a general-purpose programming language that provides support for functional programming and a strong static type system. Many of the design decisions in the construction of the Scala language were made to address criticisms of Java.

**Scala** is also interoperable with Java, as it runs on the JVM. The name "Scala" is a combination of "scalable" and "language." This language is designed to grow alongwith the demands of its users. For data science, the most popular program built using Scala is Apache Spark. Spark is a fast and general-purpose cluster computing system. It provides APIs that make parallel jobs easy to write, and an optimized engine that supports general computation graphs. Spark includes Shark, which is a query engine; MLlib, for machine learning; GraphX, for graph processing; and Spark Streaming. Apache Spark was designed to be faster than Hadoop.

**C++** is a general-purpose programming language. It is an extension of the C programming language, or "C with Classes.” C++ improves processing speed, enables system programming, and provides broader control over the software application. Many organizations that use Python or other high-level languages for data analysis and exploratory tasks still rely on C++ to develop programs that feed that data to customers in real-time. For data science, a popular deep learning library for dataflow called TensorFlow was built with C++. But while C++ is the foundation of TensorFlow, it runs on a Python interface, so you don’t need to know C++ to use it. MongoDB, a NoSQL database for big data management, was built with C++. Caffe is a deep learning algorithm repository built with C++, with Python and MATLAB bindings.

A core technology for the World Wide Web, **JavaScript** is a general-purpose language that extended beyond the browser with the creation of Node.js and other server-side approaches. Javascript is NOT related to the Java language. For data science, the most popular implementation is undoubtedly TensorFlow.js. TensorFlow.js makes machine learning and deep learning possible in Node.js as well as in the browser. TensorFlow.js was also adopted by other open source libraries, including brain.js and machinelearn.js. The R-js project is another great implementation of JavaScript for data science. R-js has re-written linear algebra specifications from the R Language into Typescript. This re-write will provide a foundation for other projects to implement more powerful math base frameworks like Numpy and SciPy of Python. Typescript is a superset of JavaScript.

**Julia** was designed at MIT for high-performance numerical analysis and computational science. It provides speedy development like Python or R, while producing programs that run as fast as C or Fortran programs. Julia is compiled, which means that the code is executed directly on the processor as executable code; it calls C, Go, Java, MATLAB, R, Fortran, and Python libraries; and has refined parallelism. The Julia language is relatively new, having been written in 2012, but it has a lot of promise for future impact on the data science industry. JuliaDB is a particularly useful application of Julia for data science. It's a package for working with large persistent data sets. That's as far as we’ll dig into the many languages that are used to solve data science problems. If you have experience with a particular language, I recommend you do a web search to see what might already be possible in terms of using it for data science. You might be surprised at the possibilities!