**Welcome to SQL for Data Science**

The demand for data scientists is high, boasting a median base salary of $110,000 and job satisfaction score of 4.4 out of five. It's no wonder that it's the top spot on Glassdoor's best jobs in America. Glassdoor analyzed data from data scientist job postings on Glassdoor and found that SQL is listed as one of the top three skills for a data scientist. Before you step into the field of data science, it is vitally important that you set yourself apart by mastering the foundations of this field. One of the foundational skills that you will require is SQL. SQL is a powerful language that's used for communicating with databases. Every application that manipulates any kind of data needs to store that data somewhere; whether it's big data, or just a table with a few simple rows for government, or a small startup, or a big database that spans over multiple servers or a mobile phone that runs its own small database. Here are some of the advantages of learning SQL for someone interested in data science. SQL will boost your professional profile as a data scientist, as it is one of the most sought after skills by hiring employers. Learning SQL will give you a good understanding of relational databases. Tapping into all this information requires being able to communicate with the databases that store the data. Even if you work with reporting tools that generate SQL queries for you, it may be useful to write your own SQL statements so that you need not wait for other team members to create SQL statements for you. In this course, you will learn the basics of both the SQL language and relational databases. The course includes interesting quizzes and hands on lab assignments, where you can get experience working with databases. In the first few modules, you work directly with the database and develop a working knowledge of SQL. Then, you will connect to a database and run SQL queries like a data scientist typically would, where you will use Python and Jupyter notebooks to connect to relational databases to access and analyze data. There is also an assignment included towards the end of the course, where you will get an opportunity to apply the concepts that you learned.

**Introduction to Databases**

First, we will talk a little bit about what you'll learn in this course. This course teaches you the basics of the SQL language and the relational database model. There will be some lab exercises, and at the end of each section there are a few review questions. And at the end, there is a final exam. By the end of this course, you will be able to discuss SQL basics and explain various aspects of the relational database model. In this video, we will learn about SQL and relational databases. By the end of this video, you will be able to describe SQL, data, database, a relational database, and list five basic SQL commands. But wait, what is SQL and what is a relational database? What is SQL? SQL is a language used for relational databases to query or get data out of a database. SQL is also referred to as SQL and is short for its original name Structured English Query Language. So SQL is a language used for a database to query data. But what is data and what is a database? Data is a collection of facts in the form of words, numbers, or even pictures. Data is one of the most critical assets of any business. It is used and collected practically everywhere. Your bank stores data about you, your name, address, phone number, account number et cetera. Your credit card company and your paypal accounts also store data about you. Data is important; so, it needs to be secure, and it needs to be stored and accessed quickly. The answer is a database. So, what is a database? Databases are everywhere and used every day, but they are largely taken for granted. A database is a repository of data. It is a program that stores data. A database also provides the functionality for adding, modifying, and querying that data. There are different kinds of databases of different requirements. The data can be stored in various forms. When data is stored in tabular form, the data is organized in tables like in a spreadsheet, which is columns and rows. That's a relational database. The columns contain properties about the item such as last name, first name, email address, city. A table is a collection of related things like a list of employees or a list of book authors. In a relational database, you can form relationships between tables. So a database is a repository of data. A set of software tools for the data in the database is called a database management system or DBMS for short. The terms database, database server, database system, data server, and database management systems are often used interchangeably. For relational databases, it's called a relational database management system or RDBMS. RDBMS is a set of software tools that controls the data such as access, organization, and storage. And RDBMS serves as the backbone of applications in many industries including banking, transportation, health, and so on. Examples of relational database management systems are my SQL, Oracle Database, DB2 Warehouse, and DB2 on Cloud. For the majority of people using a database, there are five simple commands to create a table, insert data to populate the table, select data from the table, update data in the table, delete data from the table. So those are the building blocks for SQL for data science. You can now describe what is SQL, what is data, what is a database, and what is a relational database. You know that RDBMS stands for Relational Database Management System, and you can list five basic SQL commands to create a table, insert data to populate the table, select data from the table, update data in the table, and delete data from the table.

**SELECT Statement**

In this video, we will learn about retrieving data from a relational database table by selecting columns of a table. At the end of this lesson, you will be able to retrieve data from a relational database table, to find the use of a predicate, identify the syntax of the SELECT statement using the WHERE clause, and list the comparison operators supported by a relational database management system. The main purpose of a database management system, is not just to store the data but also facilitate retrieval of the data. So, after creating a relational database table and inserting data into the table, we want to see the data. To see the data, we use the SELECT statement. The SELECT statement is a data manipulation language statement. Data Manipulation Language statements or DML statements are used to read and modify data. The SELECT statement is called a query, and the output we get from executing this query is called a result set or a result table. In its simplest form, a SELECT statement is select star from table name. Based on the book entity example, we would create the table using the entity name book and the entity attributes as the columns of the table. The data would be added to the book table by adding rows to the table using the insert statement. In the book entity example, select star from book gives the result set of four rows. All the data rows for all columns in the table book are displayed. In addition, you can also retrieve all the rows for all columns by specifying the column names individually in the SELECT statement. You don't always have to retrieve all the columns in a table. You can retrieve just a subset of columns. If you want, you can retrieve just two columns from the table book. For example book\_id and title. In this case, the select statement is select book\_id, title from book. In this case, only the two columns display for each of the four rows. Also notice that the order of the columns displayed always matches the order in the SELECT statement. However, what if we want to know the title of the book whose book\_id is B1. Relational operation helps us in restricting the result set by allowing us to use the clause WHERE. The WHERE clause always requires a predicate. A predicate is conditioned evaluates to true, false or unknown. Predicates are used in the search condition of the WHERE clause. So, if we need to know the title of the book whose book\_id is B1, we use the WHERE clause with the predicate book\_id equals B1. Select book\_id title from book where book\_id equals B1. Notice the result set is now restricted to just one row whose condition evaluates to true. The previous example used comparison operator equal to in the WHERE clause. There are other comparison operators supported by a relational database management system: equal to, greater than, less than, greater than or equal to, less than or equal to, and not equal to. Now you can retrieve data and select columns from a relational database table, define the use of a predicate, identify the syntax of the SELECT statement using the WHERE clause, and list the comparison operators supported by a relational database management system.

**COUNT, DISTINCT, LIMIT**

In this video, we'll briefly present a few useful expressions that are used with select statements. The first one is COUNT, COUNT is a built-in database function that retrieves the number of rows that match the query criteria. For example, get the total number of rows in a given table, select COUNT(\*) from tablename. Let's say you create a table called MEDALS which has a column called COUNTRY, and you want to retrieve the number of rows where the medal recipient is from Canada. You can issue a query like this: Select COUNT(COUNTRY) from MEDALS where COUNTRY='CANADA.' The second expression is DISTINCT. DISTINCT is used to remove duplicate values from a result set. Example, to retrieve unique values in a column, select DISTINCT columnname from tablename. In the MEDALS table mentioned earlier, a country may have received a gold medal multiple times. Example, retrieve the list of unique countries that received gold medals. That is, removing all duplicate values of the same country. Select DISTINCT COUNTRY from MEDALS where MEDALTYPE = 'GOLD'. The third expression is LIMIT, LIMIT is used for restricting the number of rows retrieved from the database. Example, retrieve just the first 10 rows in a table. Select \* from tablename LIMIT 10. This can be very useful to examine the results set by looking at just a few rows instead of retrieving the entire result set which may be very large. Example, retrieve just a few rows in the MEDALS table for a particular year. Select \* from MEDALS where YEAR = 2018 LIMIT 5. In this video we looked at some useful expressions that are used with select statements, namely the COUNT, DISTINCT, and LIMIT built-in functions.

**INSERT Statement**

In this video, we will learn about populating a relational database table. At the end of this video, you'll be able to identify the syntax of the INSERT statement and explain two methods to add rows to a table. After table is created, the table needs to be populated with data. To insert data into a table, we use the INSERT statement. The INSERT statement is used to add new rows to a table. The INSERT statement is one of the data manipulation language statements. Data manipulation language statements or DML statements are used to read and modify data. Based on the author entity example, we created the table using the entity name author, and the entity attributes as the columns of the table. Now we will add the data to the author table by adding rows to the table. To add the data to the author table, we use the INSERT statement. The syntax of the INSERT statement looks like this, insert into table name, column name, values. In this statement, table name identifies the table, the column name list identifies each column in the table, and the values clause specifies the data values to be added to the columns in the table. To add a row with the data for Raul Chong, we insert a row with an author underscore ID of A one, the last name is Chong, the first name as Raul, the email as RFC@IBM.com, the city is Toronto, and the country as CA for Canada. The author table has six columns, so the INSERT statement lists the six column names separated by commas, followed by a value for each of the columns also separated by commas. It is important that the number of values provided in the values clause is equal to the number of column names specified in the column name list. This ensures that each column has a value. Tables do not need to be populated one row at a time, multiple rows can be inserted by specifying each row in the values clause. In the values clause, each row is separated by a comma. For example, in this INSERT statement we are inserting two rows, one for Raul Chong and one for Rav Ahuja. Now you can identify the syntax of the INSERT statement, and explain the two methods to add rows to a table. One row at a time or multiple rows.

**UPDATE and DELETE Statements**

In this video, we will learn about altering and deleting data in a relational database table. At the end of this lesson, you will be able to identify the syntax of the UPDATE statement and DELETE statement and explain the importance of the WHERE clause in these statements. After a table is created and populated with data, the data in a table can be altered with the UPDATE statement. The UPDATE statement is one of the data manipulation language or DML statements. DML statements are used to read and modify data. Based on the author entity example, we created the table using the entity name Author and the entity attributes as the columns of the table. Rows were added to the Author table to populate the table. Sometime later, you want to alter the data in the table. To alter or modify the data in the Author table, we use the UPDATE statement. The syntax of the UPDATE statement looks like this, UPDATE [TableName] SET [ColumnName] = [Value] ]> <WHERE [Condition] >. In the statement, TableName identifies the table. The ColumnName identifies the column value to be changed, as specified in the <WHERE [Condition] >. Let's look at an example. In this example, you want to update the FIRSTNAME and LASTNAME of the author with AUTHOR\_ID A2 from Rav Ahuja to Lakshmi Katta. In this example, to see the UPDATE statement in action, we start by selecting all rows from the author table to see the values. To change the first name and last name to Lakshmi Katta where the AUTHOR\_ID = A2, enter the UPDATE statement as follows. UPDATE AUTHOR SET LAST NAME = KATTA, FIRST NAME = LAKSHMI WHERE AUTHOR\_ID = A2. Now, to see the result of the update, select all rows again from the Author table and you will see that in row to the name changed from Rav Ahuja to Lakshmi Katta. Note that if you do not specify the WHERE clause, all the rows in the table will be updated. In this example, without specifying the WHERE clause all rows in the table would have changed the first and last names to Lakshmi Katta. Sometime later, there might be a need to remove one or more rows from a table. The rows are removed with the DELETE statement. The DELETE statement is one of the data manipulation language statements used to read and modify data. The syntax of the DELETE statement looks like this, DELETE FROM [TABLEName] <WHERE [Condition] >. The rows to be removed are specified in the WHERE condition. Based on the author entity example, we want to delete the rows for AUTHOR\_ID A2 and A3. Let's look at an example. DELETE FROM AUTHOR WHERE AUTHOR\_ID IN ('A2','A3'). Note that if you do not specify the WHERE clause, all the rows in the table will be removed. Now you can identify the syntax of the UPDATE statement and DELETE statement and explain the importance of the WHERE clause in these statements.