Khaled El Tannir, M: Hello! Good evening. Thanos Michailopoulos: Hello! Khaled El Tannir, M: How are you today? Great? Do you have any question, any any news, any request. Cristal Cortez: No, I have a question. Will. Homework won't be available today at night, right. Khaled El Tannir, M: Correct after the class. Cristal Cortez: The class. How will be submitted? Will it? Will it be uploaded as a sip file, or something like that? Khaled El Tannir, M: Yeah, it is out. Spark. Cristal Cortez: Okay. Khaled El Tannir, M: And you have one week to submit. It should be submitted on next Wednesday before the next class. Wednesday midnight. So the day just before the class. Cristal Cortez: Okay. Thank you. Khaled El Tannir, M: I will explain this after the break I will give you the instruction. I will show you everything. After the break. Sam Wanis: So it it is only one week, right? Not 2 weeks, because the the. Khaled El Tannir, M: No, you have only one week. Sam Wanis: Yeah. The what is mentioned in the assessment section or assignment section is mentioned. The 8th of June is the Let me check this. Khaled El Tannir, M: Let me check this, please. Sam Wanis: It's mentioned due on June 4.th Khaled El Tannir, M: Yeah. So June 4, yeah. So you have one week. Sam Wanis: Oh, okay, yeah. I thought, because it's mentioned. May 29th to June 8, th okay? Okay. Khaled El Tannir, M: Let me open this. And okay. So since it is your 1st homework due on June 4, th today, we are. Sam Wanis: Yeah, right? Yeah. June 8, th yeah. June 8th is the date. Like the 3rd line. I mean. Cristal Cortez: This is the next Sunday, I think. Khaled El Tannir, M: Maybe. But I this is due on June 4.th Cristal Cortez: Okay. So only one week. Okay, yeah. Because June 8th is next Sunday. Yeah, it's a. Khaled El Tannir, M: Maybe I don't know what is at this date, but this is what you need to take. Cristal Cortez: Okay. Thank you. Jeff Liu: Well. Khaled El Tannir, M: Yes. Jeff Liu: So our last course will be on. Khaled El Tannir, M: Is that last course? So you you talk about week 12. Jeff Liu: Yes, yes, so if would I? Khaled El Tannir, M: Sorry. So it's the end of July. We will let me just give you the dates. Just give me one second should be July. July 24.th Jeff Liu: Okay. Khaled El Tannir, M: And your last homework last, which was a team homework. It will start in July 17th this day. So the day for the class before the last one, and you have 3 day after to submit. You have 10 days for the last homework you have 10 days. Jeff Liu: Instead of one week, one week and a half. Khaled El Tannir, M: So you have 2 weekends to work on it. Jeff Liu: Okay, second. Khaled El Tannir, M: So can we please. 1st assignment extended, who? No, we cannot, because we have a very, very tight time. I have all the schedule. I cannot move this now. and if you, if you do not submit, you will get penalized. So. No, I'm sorry I cannot do this. because one week after. Yes, you have one week in week, 6. You have your weeks. Yeah, week 7. You have your second homework. So you have one week to submit, and then I need to explain to you and give you the case. and you need to prepare. You have one week to prepare the next topic, so no, we cannot extend this. Sorry. Okay, any other question. No. So are you now? Did you have to get the chance to work a little bit with our spark tutorial spark, the workshop. Little bit handsome, because your 1st homework is the. It is based on spark Scala. You need to write 6 or 7 line of code. It is not something difficult, but it is your 1st homework. So yes, I know you have one week to to submit. but you need to be more comfortable with this, because this will be recurrent. the next homework. And later and later and later, okay. so today, we will be starting our 7, a second part of the course. Okay. So this we covered and completed our part. One part one was about how to store your data on a distributed manner, how you can process your data in a distributed manner. So you are now more comfortable with this and we. You have an idea. You practice it a little bit how we can manage file on Hdfs. So this is our 1st block. our 1st phase, our 1st step. 1st milestone, if you want. So this now will be the foundation for the next part part 2 and part 3, because all the files will be working with will be stored on Hdfs and the output. It will be outputted also to Hdfs. Okay, so I want you to be more very comfortable with how to upload and how to download data from or files from Hdfs how to explore Hdfs. So this is very important. So the data analysis part, which is our second part in the course. So we have here 3 classes. So the today class, the next class and the after deck. So we have here, we will be focused on the data analysis. Now, the data is here. The data can be process, it clean it. This is what we are doing with the spark. We store our file on Hdfs, we load files or our input files using spar, we do some cleansing, some filtering, some preparation. And today we will add a new feature which is partitioning. So we're working with partition and bucketing. So I will explain what it is about today. The next week we continue with this data analysis, and we will be completing this part 2, using and talking about file format, different file formats. Okay, so here we'll be working more about SQL and simile. SQL, so if you are already familiar with SQL, this part will be very easy for you. and you will be very comfortable with it. So we start today, our data analysis part using hive, which is the foundation. you will learn what is partitioning, how to partition, how to organize partition, how to organize data into partition. All this stuff the next week will be completing with SQL. With Trino. and of course you already know this is what we covered last class how to access data using spark and how to use the data frame Api, how to use the data set spark dataset Api or use SQL statement in spark. So this is what about Spark, SQL. And in the latest class? So the 3rd class of this second part also, I will introduce Apache superset. So you will be able to build the your end-to-end solution with data at rest. Data is not moving so data at rest so that you put the your file manually on Hdfs. And we start all this process manually, because we didn't now cover any orchestrator, and you will be able to do this end to end great. And of course, for now we're still using Zeppelin for all these tools except Apache superset, of course, which is a dedicated and independent tool. Okay, great. So part one was, as I said, just this is a refresher distributed storage distributed processing. We store our data on Hdfs with low velocity, for instance. And we can process our data on disk using mapreduce and in memory using spark. Today, we'll start with something complicated. We start with structured data and you will learn later how we can deal with unstructured data. But today, because it is easier to understand and is easier to to practice, we'll start with structured data. Okay, so this is our process pipe model. So this is, you, remember, you recall this, we start by collecting the data. This is what you are doing when you upload your data to Hdfs. This is a kind of data collection. Starting Week 7, you will learn something new how we can connect to different sources. How we can connect to the Internet, how can we connect and grab data from the Internet or from any endpoint on the Internet? For example, data in motion or whatever. So, but, for instance, we are collecting the data manually. But this is what we are doing. You upload your data file input data file to Hdfs, and from here you prepare this data. You clean, you filter, you sort to do whatever you normalize whatever you want, column names and whatever you want using spark. Usually, this is most common scenario using a spark today, we'll complete with this now. data or insight getting insight from the data using high level language, such as SQL and simile. SQL, later, we'll continue with the superset later. Okay, so what is hive hive is our tour for today we will be talking about the hive architecture, the hive data model, the hive query language not very complicated. But we have a combination and subset of the SQL. Language with a new keyword, and maybe a little bit for some synth. So for some statement, a new syntax. So we need to just to be more familiar with this tool. Okay, so this is our landscape. So we start, we collect data, the highest velocity or low velocity wherever and for instance, we'll focus on data storage and the distributed file system. Hdfs, so all the data we'll be working with will be stripped it. So this mean binary or text. For instance, we are working with text file, and later we'll be working with binary file. But, for instance, we are working only with text file. So the text file. This mean, maybe this is a Csv file. So with or without header, we will learn how we can remove this header if it exists, and how to deal with all this kind of data. Also, we have some hands on to learn how to deal with semi-structured data, which is Json file. Json needs a little bit more. not advance it, maybe add some, or use some features from hive. To be able to load this kind of data and be able to query this type of data. But we are focusing on structured data, and we'll be using spark SQL hive. And the next week, Trino. So these tools, for example, Trino and Spark sequel are very, very common. So. for example, if you are already familiar with, or you know about bigquery or Athena, on Amazon Aws, or Bigquery on on Google Cloud. These in the back end use Atrino or presto, which is the the legacy version of Trino. So what we learn and practice with Hive and Reno, you will be able to practice this and use all the concept and these fundamentals using bigquery on Google or using this concept on Athena, on aws exactly the same, maybe a little bit, some differences, but almost the same. So just to give you an idea about the popular tool. There are a lot of very, very common tool about how to analyze data. So this was a very common landscape. I can check this. Maybe it's not the latest version, but you can check the latest version from the URL. I specify on this screen. But what it is important here is just to give you an idea about the landscape. So on the left you have all the frameworks. So the frameworks. Now we know about spark. We know about mapreduce mesos, just which is our not our spark cluster manager, but was created to run spark. But mesos and yarn are very close to each other. So we are using yarn on Hadoop, as you can see here, you can see Hadoop Hdfs, you can see Hadoopreduce spark. Flink is very similar to spark. But it is a real time processing framework while spark is near real time processing framework. So a little bit difference. And in the file format you can see we have different file formats. We have the iceberg, the parquet, HUD Orc, Rho and Delta Lake. We'll not be covering all of this. We will be covering some of this file, especially Parquet and Orc. We have also another Avro, which is not listed here. but these are very, very common, and you will understand why we should use this file format and not text formats, such as Csv or or whatever. But this is, we have a dedicated class for this file format for the query and the data flow. We can use spark, pig, hive, presto, and trino, you can see. Presto and Trino. You have drill, Apache beam, Apache hook. But all these tools are used to query your data. You can use spark and use the data frame Api data. Set Api and SQL. Api. To query your data. Everything can be done with the spark pig is a tool dedicated to process unstructured data, huge volume of unstructured data. But pig is now a little bit deprecated, since, I would say 5 years, for now there is no new version, but we can run. We can use spark as an execution engine to run pig so pig can run you. It was created to run on mapreduce. It was designed to run on. Mapreduce, hadoop mapreduce, and later new features to to change or to replace the execution engine. So we could use mapreduce version 2, which is Apache test. And also we have the possibility to change or to replace the mapreduce original execution engine by spark execution engine. But the main problem is now the Java Library versions, because the recent version of Spark needs more recent Java Library and Java Virtual Machine, while Pig, still in the 1.6 or 1.7 version, and this is now a problem to run both. This is why I used to introduce pig in our course, but I removed this due to this incompatibility of version, libraries, and so on. But it is very, very good tool, and it provide its its own language. Pig Latin, which is a script very inspired from the SQL. Language, but it is not sequel. It is a script, but it is inspired from the SQL. Language, so hive will be covering today. Presto and Trino are very similar. Trino added a lot of new features to the presto. which is the initial version, and presto! Is used as a back end for the Athena on Aws. Trino is used also in many, many different other large big query engines. For to query large data set. so graphql drill is another drill will let you connect to any data, to any file and query this file. So, for example, you can connect to a Json file, and you will be able always to use, for example, SQL. To, to to query the file. So it is. We can do everything now with spark, hive, entry. All this all analysis type we want to do. We can do with these tools we can prepare. We can filter, we can normalize. We can we do anything we want using these 3 tools for the data access. So it is not our parts databases. You can see, we have different databases. So this is very common. So here you can see. But there is some error in this part, so you can see in the databases you can see the 3rd line. We have a flume plume is the second icon from the left. It is not a database. This flume is data collection engine. It is a tool that you can use to connect to a source and ingest data from this source. So it is not a database, and it will not never be a database. But again, this is not a problem. Just to give you an idea about what we will be covering. So this is why, in the query and data flow, we can see. Now, this is very, very popular and very common tools with hive, trino and spark. So the question for you today is how to analyze a distributed file or data set. So what do I mean by analyze this means? I want to use a dedicated language such as SQL, how can I do it. Now you uploaded your file on Hdfs. You can access or you know how to load these files using spark, you know, or you have an idea how connect or to load this file into spark engine and use the Apis data frame that I said SQL. To query this, but if I don't want to use a spark, as you can see, spark. Need you need to be a Scala or Java developer. Not maybe this, for the sequel part is a little bit different because you are using SQL. Statement, but you still need to load your spark or you package your application when you go to production. It is not exactly how you are doing this. Using Zeppelin, you are using this using Zeppelin. It is very similar to what we are doing in a development environment. This is very similar to a data scientist using the Jupyter notebook and create and running code. The Jupyter notebook is not what you will use in production. So here you are doing the same in our environment. In the sandbox, you are doing something similar to what you do in a development environment. You are running in ripple mode. So line by line you evaluate, you run, you see the the evaluation. You see the output. and you continue sell new cell, the plane and add a new code, and so on. So this is not what you are doing in. You will be doing in the production environment, in the production environment. This should run without any human interaction until the administrator needs just to, to, maybe to to to maintain, or whatever. But now not all of us are Java developer. Not all of us are a python developer when we go to packaging. So this is why we need to be more close to the sequel. And this is where these tools come in hand. So how to analyze, I don't know how to analyze instead using spark or this engine. So this is why hive was introduced. So to be able to let you use a very similar or semi SQL language to be able to query your data, because more many of us know already. SQL. Or can learn SQL. I would say easily than Scala or Java, or maybe Python. And now the question, how can we do, or what should I do to analyze billions of rocks? Remember, we are not thinking on a single machine. We are not thinking how to analyze a small data set. It is not a very small file that you can load and analyze. So here we need also to worry about the time it will take to analyze your data. The one very simple statement on SQL. Is the count. So when you do a count, select Count star from table name. This very simple SQL. Statement is very high time consuming. So this will consume a lot of time, processor time and resource to compute. So how can we optimize this? How can we do? What should we do to be able to reduce this time to compute the count of the number of records. So do we need. How can I query my very large data set with billions of rows? How can I do this if I just do select on and in the back end my engine do a scan table. It is not something very optimal. So this question we need to answer it. So how to analyze a distributed file system. So I need a distributed query engine. So this distributed query engine will distribute my query and collect the result for me. what if I have to analyze billions of row? I need to use different techniques, new techniques. We will be talking about partitioning. about file format and about data bucketing. So this is the 3 foundation. To worry about when you need to to process to analyze very large data set in the distributed manner. Okay, so let's see thesis. So we have this 3 main functionality to focus on the data format. For instance, we're still working with, see text clear text. As I said later in week 6 we will. I will introduce this different file format. But, for instance, let's keep it as a text file. But when it comes to production. never think text file. you will see later why, but never think. The text file are not optimized for speed and for distributed way to to process your data. Okay? So also, we need to think how to reduce the size of our data. It's very important. If you have a file data set that you store on Hdfs. And this is file. The size of this file is, let's say, 10 GB or 3 GB. If I had to do a scan table with 3 GB. It will take, let's say, 10 min on our sandbox, for example. But if I can find a way to reduce this size and make it maybe a half a gigabyte or 500 MB. It will be faster. So we need to worry about the file format and the size of the data. Very important. So, but this will be, I will go into details in week 6, the second functionality. We need to focus on is the data partitioning. When you have very large foil. Usually you don't need to explore or to get insight from all this data set I just give. We'll give you an example if you are working one on e-commerce website, and you have many, many customers from all around the world when you want to. for example, know how many customers you have from this country, you need to create a query. And this query will filter your data or your data set based on the square on this country. So you say, Okay, I want to know how many customer I have from, for example, Canada or from the United States. So here you are doing filtering. So you are excluding all the rows that not belongs to the Canada or Us country. If you want to do. For example, if you want to get the sales amount for particular month, you can say, I want to know, for example, the sales amount for the month of May. In this case you will say, Okay, get me get the sum. Select sum of all the records from my table, but you will filter again. You will be doing aware, close to just limit to this month, year, maybe, and month. So this is where partitioning will be very helpful. So think about partitioning your data this way. So I will say, I can partition my data based on a country. I can partition my data based on a date or any other parameter to to use as partitioning K. So here, when you need to extract this insight from your partition, the system will go directly to this partition. There is no need to scan the full table or the billions of row, so partition will help to decompose your data set into more manageable parts, and this part will be physically stored on the distributed device. And whenever you need to query based on a particular partition. K. This, the the engine will retrieve the data for you from this partition. So partitioning is very important, and and before partitioning we need to think what we call and prepare our partitioning strategy. And this is base it on what the users needs to query to do with the data. So, for example, for a particular. Let's say, financial data set. Maybe they want to know this particular stock on this day, or the amount of open and close value. So maybe you need here to partition. Your data set based on the stock name could be. Also, maybe you don't need to use the stock name, but maybe use the date to partition your data. So the so this is the partition strategy should be prepared, based on what the users will be using to query. And also very important. We can also think this, how later you want to build your dashboard. I want, for example, to know the average of the sales for this month or whatever. So you need also to think about that. So this data, partitioning and data organization is very, very important to to to optimize the querying time, and especially when you have very large data set. Now, when we come to spark and hive. we have also something to to worry about, and which is, can be very helpful to optimize your querying time, which is bucketing. So what it is about. So here you are organizing your data physically inside a partition. So, for example, now I have my sales website, and I have different customer coming from different countries. So I have, for example, customers from United States, and I have also customers from Cyprus. So if I compare the population. United States and Cyprus are very Cyprus, very, very small country, and United States is a very large country. So for sure, if all the population buy from my website, I will get 2 different sizes for this partition. If I partition it my data set based on the country. So here I will have a very big partition for the United States country, and I will get very small partition for the Cyprus partition for a country. So to make this more, I would say, homogene. we will be using bucketing. So bucketing is a way to distribute physically the data inside the partition. So this will help the engine to be more efficient when it will comes to read your data. Even this partition has different sizes. So partitioning is here to optimize your query and bucketing is here to help you optimize physical distribution data inside your partition. So the bucket size is defined and set by the administrator. So you when you create your bucket so you can decide physically how to store and how many bucket you want to store. Usually we don't use more than 64. This is very, very rare, usually very, very common. Scenario. 32 bucket, very common scenario. Now in the I say in the previous version of Hive. When we come to create buckets, it it initially create the number of physical files inside each partition and use a segregation function based on hash code and whatever something like that to distribute the data over these physical files, the buckets. Now, since our in the latest version, this is our this. Now it is different. It doesn't need to create these physical files, but it will create the necessary number of files and distribute your data inside this bucket to balance the size of your partition. So 3 main function you need to focus on when you come to prepare your data to be analyzed as at scale, the data format which will impact the size and the format itself and the data organization, the data partitioning strategy and the data bucketing strategy. So just to summarize what I said, the data format you can use any any data format, for example, Csv, just G, so. Xml, but these are not optimized. Okay, while Avro, Parquet and Orc are very optimized to to be used at scale, and for a very, very large numbers of file. Again. These are not the only file format. We have different other file. Format, Hoodie and Delta Lake are. Hoodie is dedicated for the data lakes. The Delta Lake is another file format used by spark, which will store only the difference, the increment between the data. So this is the Delta Delta Lake. A row is another way to to create an abstraction, and you can use any file format. So again, we will be focusing on Avro, Orc and parquet. But the same concept can be used in iceberg and Hoodie just reading the documentation. You will understand the data partitioning also. Hdfs does not have the concept of index until now. When you explore the comment of Hdfs, it focus mainly on managing files and directories. How to rename File, how to create directory, how to move, file, how to delete, how to change the access policy, and so on. But we don't have this concept of index, though the similar to what you have in database management system. And remember, even if you want to read one file, we need to read the entire block number of blocks from the start to the end. We cannot just choose one block and read it randomly from this for this particular file, so here the partitioning will provide a way to read. Only a subset of the data partitioning will be based on a partition K. Usually this is the column name. So, for example in my table. I have my in my table. I have different columns, and I have the colon country. I will be using this colon country to partition my data. So the partition K is my colon company. Okay? Now the partition value is filled from the rose you have in your table in the column country. You have different countries. You have United States, you have Canada, you have Cyprus, you have France wherever. So the value is extracted from the row itself. Okay, so this will create your partitions. You have one partition for the United States, and it will group physically all the rows from having the the value U.S.A. In this column. So the partition we have the present on the colon partition K, which is the colon, and the partition value is the colon value for this column extracted from the row. We can also create or use multiple attribute to partition. I can decide to start, for example, by a year. then a subpartition, the months. the subpartition, the day, maybe hour and minutes, depends on the granularity you are looking to to to your data when you query your data again, this should be prepared, based on what the users are commonly querying your data, using as query to query your data, so this will be very efficient. But if your partition strategy does not meet the user queries, you will not get this efficiency. Okay, very, very important. So physically, talking, the partitions are directories, so this will create one directory per partition and store. Inside this directory the physical bucket, which contains all the values from the rows, exactly from the rows for this partition. You will see this in the workshop, which is better to understand and help you to understand how this is physically organized. So the data bucketing. Again. As I said, this is a way to organize and balance the size between the partitions. So partition is optimal when attribute has a small number of unique values. So countries I know that I do not have more than 2,000 countries in the world, so I will not exceed 200 partition. But if I say, I want to create one partition per product Id and I have 3,000 product. It will create 3,000 partition which is not optimal. So this is where we need to use bucketing again. This is a way to balance the distribution of the data inside your partition and bucketing is here to say it is very efficient, and especially when you have very large file of very large of unique values. For example, you can create a bucket using this product. Id, so we can see, okay, I want to physically bucket these products together because we want to. For example, to package this product together so it can be done. Again, bucketing is very similar to the partitioning, and help to distribute the data efficiently. The data can be partitioned without to be bucketed. The data can be bucketed without to be partitioned. But again. we used to, when we create partition, we need, we used to create buckets, organize with buckets. So in in the workshop, we'll be doing. Just understand, with this, if you didn't apply always bucketing. It does not big deal, but want you to understand? The the difference of partitioning, partitioning, and bucketing, and how to use each of them, and when to use each of them. Okay, so this is about bucketing. So now we have an idea about the concepts we need to apply with foundation to when you need to analyze data at scale and very large data set. Okay, so now, Hive will implement all of these features. So Hive is a tool called the data Warehouse on top of Hadoop. So it will. it will aggregate or create your data group, your data physically on Hdfs. And it is called the Data Warehouse. Okay, it can deal with different file storage, so it will. it can be extended also. This extension, based on what hive call a Saturday Surde, is a short for serializer and deserializer. So this is just the way or the techniques hive will be using to read data from the physical storage and store data on the physical storage. So if you are reading, for example, a Csv text, so Hive will use the survey dedicated to read Csv file, so it will read the Csv file. It will deserialize what it will read from Hdfs and serialize when it needs to output the data to Hdfs. If you are using, for example, Jason, we will be using Json Survey, the G. 7 Serializer. to read, to write, to disk, and Json Deserializer, when to read from disk. So this is just the terminology in in hive, and the 3rd day will define or determine the input and output format. Not nothing else. So hive can summarize big data, and you can query very easily, as we are using a simile. SQL. Language very, very simple, very familiar, and it's scalable and extends. We can extend the feature of hive. Basically, initially, it is written in Java, and of course it is open source. So now you have a better idea. So Hive need to store the schemas of your data in a Meta store. We'll talk about this in a few minutes. So when you create when you store a file on Hdfs. And you need to read it. Remember, Hdfs will not ask you anything. You just put the data, and it is up to you to define your schema on. Read. remember. So it is up to you. It is your responsibility to define the schema. So here I've the to to read a file or data on Hdfs. The schema is mandatory. You cannot read, just say load, and it will infer for you the schema. Now this works in spark, but not in hive hive. You need to explicitly define your schema. So what is the schema? The schema is nothing but the definition of your table. So you say. create table, my table customer, and in my table I have different columns. Name, 1st name, for example, last name. I have phone number, address, Zip code, and so on. So this is my schema. So this schema is stored into a meta store. This is what we call the hive metastore. So everything you will be doing. create partition, create a schema, extend the future of hive, set the engine whatever you want to do, everything is stored in the hive. Metastore. Okay? So hive can be extended using what we call the Udf user, define it function so you can add new keywords. You can add whatever you want, and another course I gave I used to. It was advanced hive course. We used to write our own Udf. so we have different function that can aggregate that can explode. So we have. We can do a lot a lot lot a lot with the hive Udf presented in Java. But it is not part of our course. Just understand that we can do it, and I will give you an example of already packaged library that extend your hive functions using machine learning. So this will add some machine learning function to hive. But I will show you this as just by the end of the class. Okay, so hive provide the hive. Query, language, hive, Qa. Which is a subset of the sequel. It is not the full sequel. Just a subset of the sequel, and I've added some keyword, and some say some new keywords, a little bit syntax, different syntax. To to be able to read these different file formats, to be able to remove or define some characters in the code whatever. So to extend. How we can read the idea here of these new keywords is to deal with the survey. For example, you are reading a Csv file. How you can remove the header if the Colon are double quoted. If you are not using the comma, but you are using semicolon to to as a colon separator or the tab as a colon separator. All these parameters are set for the survey, and we are using this collection of keywords some of them are already in can be already inferred. So you don't need to to type this each time some time. Now you need to get to provide this explicitly. So this is why it is not the full sequel subset of the SQL. Language with new keyword dedicated to to the hive engine. So also Spark can read and execute all your hive query language. So this is very competitive. Spark can read your hive query language. So it will. And also the hive query, language, provide 3 data, rich data type. So this rich data type are struct lists and maps. So the structs list and map here are here to let you manipulate different column types. For example, the maps are dedicated to to manipulate mapreduce output because my previous outlook, remember, is K value format in a tuple. So the map is nothing but a tuple. So here we have the K. And the value. But the K is always a string. The K is always a string list is here to let you define an array or a collection the column as a collection. For example, you can say I have a colon package, and inside this column you can say I can. In this package. I have the charger. I have the USB cable and I have my phone. This is a package and this package. These 3 information are stored as a list in this column. the struct will let you in bend more than one or nest more than one column inside one column. For example, this is very useful when you come to object oriented programming model. You say, I have, for example, the object user. And this user, we have the name of the user, the date of birth. And, for example, phone number and address. So we have here 5 or 6 column or attributes to define the user object using the strict data type in hive. You can nest all of these attributes in a single color. So it is very useful for this so very efficient, and a hive is query is able to query petabyte of data very, very efficient. It was not designed to be interactive. Very, very important. Hive was not designed to be interactive. So it is not the tool you will be using to run a query and wait to get the result in seconds or milliseconds. This is not the case. So Hive is here to query your data and run very long job. The interactive query is Trino. This will be covered the next week. But here we are running this in the background. We'll be able to interact with it with hive. But hive. keep in mind. It is not an active tool. So it is not designed to be interactive. It is designed to run very long job and process and analyze your your data and query, your data set. So how it's work. Very simple architecture, very simple. So the data should be stored on Hdfs. Okay, once the data is on Hdfs, you can use the different hive can use different engines. So spark sorry mapreduce, test or spark in our sandbox. So initially it was created, or it was designed to run over. Mapreduce test is optimized. Version of mapreduce so and hive can be ex, can execute or can use spark to execute its as execution. Engine, so it will say, enhance a lot the time, execution, and optimizing the hive execution. So in our sandbox, in our environment, high views test run on test. Don't try if even you are. You see this on the Internet. Don't try to change the engine to run high, overspire. This will not work for the simple reason spark and the hive engine spark is compatible with, I would say. old version of spark, not the most recent version of spark. So if you want to run hive on spark, we need to downgrade our spark version, which is not acceptable for our course. But earlier, maybe 4 or 5 weeks 5 years ago. We I use it hive to run on to to on spark in our course, but now it's running on test. But you will not see any difference, for our at our level we don't see any difference. The execution time may be a little bit slower than spark, but it is acceptable for our cost, but just to know that hive can use mapreduce or tes, or spark as an execution engine. Now you write your query and hive. Ql. Query, and you submit. Hive will take your query, will create or convert this to Java package. Java archive and submit this archive to the hadoop cluster. Do you remember your workshop with Mapreduce Hadoop. you took the Java Archive and choose one application from this achieve submitted to the cluster and wait back to the result. This is what hive is doing in the background. So I will take your query. Your convert into Java package. Submit as a Java application mapreduce application to the hadoop, cluster and monitor the result, and once it is ready it will give you back. This is why hive is not interactive because it needs to create the Java Archive, submit to the cluster, monitor the progress and return back the result. This is why. But this is how it works. Okay. So now to have to get more better idea about the hive architecture. So on the bottom. you can see the storage. So we have our distributed storage on the bottom, which is Hdfs. So, and some who say the cases we can use the some keywords to let us read from the local system, but it is only for developer, and it will not works when you are in production. so don't try to do it even I can. I will show you the instruction, but maybe it will not work, because hive here is this is configured to run as a distributed engine as it runs in production. So maybe reading from the local file. So from outside, Hdfs, maybe not, will not work. Okay. So on top of Hdfs. So we have our yarn now produce and test. So these are the execution engine could be spark also. Again, here we are using test which is enhanced version or optimized version of mapreduce. Now on top of the execution engine we have the hive core. So here we have the hive driver, the hive metastore, and the hive thrift server. So the hive driver is, I would say, for the administrator. the hive metastore is, this is where hive will store everything. You create a schema, you define the index, which is different from the index you already know. In the Rdbms you create a partition, you create a bucket. Whatever you are doing, everything is stored in the Metastore. So the Metastore by default is Provide is you use Apache Derby, Apache Derby is, let's say, very small database, very similar to sqlite. but accept only one concurrent connection which is not doable in a production environment. So in in the sandbox. we are simulating a production environment. And the meta store here is a real Mysql server. So we are using my SQL. Server to to replace this metastore because it accept concurrent access, and it will. You can run different queries different. So we are doing this as in production environment, so the high thrift server is here to let you communicate with from the with different tools or 3rd party application and the hive core. So this will let you communicate. So if you want later to connect my Microsoft excel, you want to connect tableau, or you can. You want to connect? Don't know whatever tool using Gdbc or Odbc connector. This will use the hive thrift server, and you have in the guide in the sandbox. You have the instruction. if you want to install the Odbc connector, so you can connect your excel to, for example, your excel to to hive to the hive engine in this sandbox. It is doable, because everything is already configured for you. Just install the audac driver for Microsoft or from Cloudera, and you will be able to connect and use this connection. On top of this client application we have some administrator tools, such as the hive client interface which is deprecated. Don't use it. It is. I think it is disabled, and I don't remember if I disabled. But it is not really, really nothing. You it will. It is not sexy, not no, so we have the hue. A hue is the hadoop user experience, which is a web tool developed by Cloudera, and give you a kind of an editor to create and to edit your query and submit to to the hike. We do the same with Zeppelin. Exactly the same. So we have here the Zipline you, you'll be able to interact with different in different cell. You just use the hive interpreter, and you'll be able to query your hive table. Also, we have the hive web interface, which is it was provided in the I would say, 1st versions of hive. Now it is deprecated, and it is no more available in the hive package. So now, if we take a look to the hive, how hive do this? Read and write operation. Remember, it used the survey, the techniques to serialize and deserialize the data so to read from disk. So from Hdfs. You say, I want to read the Csv file. I want to read the Csv file. I want to read the parquet Avro, or C, so you define your input format. This is the responsibility of the Deserializer. So it will deserialize this format into a memory record internal memory record. When you want to output to store on disk, for example, you are using hive to create a partition. When you create the partition, you are outputting data to the disk to the Hdfs. So here it will use the serializer. So from the input from the memory in memory record, it will use this serializer. create the output format and store it on Hdfs. So again, the 3rd day is nothing but the library uses by hive to read and write data from the storage space. No. if you talk about the hive metastore, which is very, very important, part of the hive ecosystem, because the meta hive metastore is a single point of failure in case of you do not have, does not have access to the metastore. You will not have access to your data, you you cannot run anything from hive, nothing because Hive needs to read everything from the Metastore everything is stored. The metastore, the database name, physical location on Hdfs. The tables, names, location, input format, output, format column names, type everything. The column, the partitions, survey functions, statistics. Everything is in stored on the metastore. And again, the metastore in our environment is a database dedicated in the Mysql server. The hive data model use very common terminology. Database is exactly the same as you already know, in the Rdbms. So database is nothing but a namespace, a kind of a directory that contain your table. So, physically speaking, Hive will use Hdfs Directory to create the database. So, for example, if you create database tutorials. so Hive will create on Hdfs Physical Directory tutorials. Dot dB, to say, this is a database inside you create a table. and the table is very, very similar to the table in the rational database management server, the so physically here, it will create a table, a directory to store the data of this table. Okay, so this is how it is organized. It's a partition. Each partition in a table, because partition belongs to a table. So in each partition you create or you add to a table, it will create a physical directory on Hdfs, because here we are talking about Hdfs. The buckets are piles, not directories, files inside the partition directory, so the database is a directory. The table is a directory. The partition or each partition is directory, a subdirectory database table partition. and the bucket is a file inside the direct partition directory. If the table is partitioning. If the table is not partitioning, the bucket file will be stored in the table directory. you see. So let me, and by default by default. When you install hive it will create on Hdfs the warehouse inside the user hive directory. So on Hdfs, we have user hive warehouse. This is where hive will store any, everything. But again, don't go to this directory. You can explore if you want to see how it is stored. But don't, delete. Don't change anything because it will impact your hive, and you maybe will not be able to access your your data. So this is an example. So here we are creating or organizing our data partition table. So I have, for example, my date, which is the partition you can see date equal. I have a value. So this is how the developer create the partition. So the column name here is date. This is the partition K, the value you can see this is to 2080 and 0 5, 5, may and 13 may. So the developer chose the value like this. So it is not mandatory to be like this. It is up to the developer. Okay, so here, this is the value inside, you can see different also subdirectories, the hours equal, 18 equal 90. And inside you have a file physical file. So here we have. This is 0 basis index 5. So the 1st bucket start with 0 number index 0, and the latest one is 31. So I have 32 buckets in this partition. So this is how it is physically organized. You will see this in the workshop. I can. You can explore on Hdfs. But we will be using our custom directories and not custom location if you want and will not go and explore the default directory user hive warehouse. which is better. So this is just an overview about how the data model in hive is organized. So our database contains all our table definition. The table definition will contain all the partition definition and the partition will contain the bucket. Again. the partition and the buckets are optional. It is not mandatory to partition any table or all the tables. But, again, if your data is used in production, and this table is used by many, many users. You are using this table to. To to get to, to connect to an external tools or an external application. you need to think about a partitioning strategy. Okay, but this is again optional. So sometimes, if you are working on a development environment, maybe you don't need to partition. If you are maybe doing some temporary table, maybe it doesn't need to be partitioned. And so so it depends again on the use case, but always think partitioning strategy. You need to partition your data until if if it is going to production environment, of course. or even it is not going. Just think about this. So you will be more familiar with partitioning strategy and how to do it. Now, Hive, provide 2 tables type. This is very, very, very important to understand. So this type, 2 types is called internal or external. This is the short name, but the technical name is hive manager, table or user manager table. So this is very important to understand this tables type, and what the difference between these different table types. So when we say internal. this means hive manage it. Does this mean hive will take care or will manage the table schema and the data of this table. So both table schema and tables. Data are managed by hyph. while in the external or the user management type hive will manage or will take care only about the schema. While the data is the responsibility of the user user manager. Table means hive will take care of the schema, but not the data. The user will take care of the data and the internal hive will manage. Both the schema and the data by default hive table are internal by default. So when you say, create table hive will create a hive manager table. So Hive will take care of the schema and the data. This is not what always we want, and especially when we go to production. Why? Because 1 1 second, because when we work with hive manager table, if you drop the table hive will drop the scheme or delete the schema and the data. while we in the external or user management hive will delete the schema only, not the data. Yes, Sam. Sam Wanis: Yeah. So I I just want to know, for the external or user defined table user managed. I mean, I understand that the data is managed by user. But again, the schema is dependent on what? Input, what data files, what the structure. So the user should have control over the schema of the tables, external tables to to have it structured like, add column, or alter the table to remove columns. So this is still, I see it user managed. But I'm not sure. How how come the? It's only managed by hive with no control from user. Khaled El Tannir, M: So I've in the internal type or hive management. When you create a table, you provide a schema. And also now the schema is empty. Nothing is in the the table is empty. You need to load the data into this table. So now Hive will take care of this. This schema you created and the data you loaded into the table. while in the external type or the user management type, the user will provide also the schema. So because when you read data with hive, it is the schema is mandatory. So you provide the schema. So Hive will take care of the schema. No, the data itself. Hi will not touch it if you drop the table. Data is still here while in the internal type. If you drop the table. Oh, the data is managed by hive. This data is deleted. Now you still be able in the user manager type, you still be able. Even you delete your data as a user if you remove the data, but your schema is still here. All you will get is an empty table. But you're still able to manage exactly the schema the same way. You can do it normally. Say you can alter table using to alter, to change columns, alter partition, to change partition, and so on. But if you if the user delete the data hive will get just an empty table. And why, if you drop the schema. the data still on Hdfs. It is not deleted while in the internal. If you do a drop table, this will drop the schema from the Metastore and the data from India. This is why it is very important to understand when to use the hive internal table type and when to use the hive, external or user management table type. You use the internal or the default type table type. When data is temporary. data is not going to production. Data is just temporary are using. We create a table. Just maybe this is a staging table. Because we are reading. We need to change some format. We need to use this data to be partitioned into another table. So the data is temporary. You don't need the table after deletion. If you delete the table, it is not needed later. and hive is the only tool that using the data, because in many, many situation, you maybe you need to create your data using or load your data using hive. But this data will be used with another tool, maybe spark, maybe an I 5, whatever. So we can. If data, if this table or this data is not used in in any other tool. You can then use internal tables while the external tables or user management table. especially when you are on production always, we think, external tables when the data is already on Hdfs available. So, for example, you upload your data on Hdfs. you prepare the data, using spark spark now outputted the clean data on Hdfs. So the data now is ready. It is available on Hdfs. It is not hive that you prepare the data with or whatever it is already here for you. So in this case, you need to use the external type or user manager type, because this is already here. Of course, you can just create an internal table, but it is very, very dangerous. If you drop the table you will lose your data, and especially, it is critical when you are in production, and the data is prepared for you using other tools. This is very, very important to know. So when the data is available and prepare it by another tool. We always always use X type, external type or user management table type. And again, if you need to output using hive. And this will be used with another tool. Also, we need to use this as an external. So the idea is very simple. You have 2 table types. internal and external hive, manage it and user manage it when it is temporary, when data is not needed. After this, when data is just hype is only the only tool is using your data. You can use the default type, which is internal. So create table means it will create a hive manager table. If your data is already prepared for you is other tools are using your data. use the user management table types, which is the external type. So this is how we can and decide what type table type to use. Now, now, let's see a very, very simple example, very, very simple. So create table products. I just use the table, create, create statement to create a table. And the table name is product. And I provide the schema which is here in this example for the column, and followed by the type, so id name, price, and category. All these are the table column names, followed by each type of each column, the integral, the string float, and a string for category. No, I didn't specify anything between, or explicitly the external keyword. So this mean, it is an internal table hive manager table. When you create hive manager table, you need to explicitly load the data into this, the table. You need to explicitly populate the table with your data. Hive does not know anything about where your data is. How? What does it know? Just give me the data. So Hive will ask you to give him the data. So this is why you need to when you'd say, create table. which is internal table type you need to follow with load data in path or use insert into insert into this table. So if we load this from Hdfs, we can use load data. Input, this is the keyword or the instruction statement to load from a path on Hdfs. so load data. And so here we are loading the file product Tsv. which is a tab separated value product, file name products. And this file is located into the Temp directory. So this is on Hdfs and where to load the data which table I want to populate with this data into table products. You see very simple create table and load data in part. So this is for internal tables. And again, if I go and do drop table. the table products will be deleted and products Csv will be deleted. Both will be deleted. Now, when I'm I use the user manager table. The data is already available on Hdfs. This mean, you outputted the data using, for example, spark, or you created a directory on Hdfs and you uploaded the file to Hdfs. So this mean, the data is already on Hdfs. Okay, so in in this case you can use the great table, but the same. And this time you can add explicitly the keyword external. So create external table table name is product. The schema is exactly the same. Nothing changes here. And I added. because Hive doesn't know where is the data located on Hdfs. And remember, in the internal. You need to load explicitly. So when you say load data in path. So now the location is known by hive from the from the path. But in the external table type the user knows where is the data, but not hive. So this is why you need to provide the location explicitly. So create external table and the schema for the table location 10. The the location is always a directory cannot be a file. always, always, always, always a directory. So this mean, you can put more than one file physical file into this directory, and all this will be loaded. Remember, then, our mapreduce, the Mapreduce might, ex produce more than one physical file in the output directory. And remember, I said, we don't need to merge these files together to get the final result, as if you are using any tools from the hadoop, ecosystem or spark ecosystem. They are able to load all these files without need to merge them into a single file. So here we have one example. So you provide the directory, and even you have more than one file. And of course all these files should be the same schema. Very important can be loaded without any additional action. So here I do not need to load the data explicitly. So so there is no need to do load data because you provided already the location. So the user management table here is just we don't need to do to do. Load hive will point directly to the location you provided, and will be able to read directly your data in in the internal type you. The location is not known. So you need to provide this location explicitly and the external type. The location is known because you provided so Hi doesn't need from you to load explicitly the data it can read directly the data from this location. Now, if I do drop table. so only the schema will be dropped. Nothing in this example, nothing from the Temp Directory or the Temp Directory itself will be deleted. Nothing but the schema table product, table products in the store, the meta store of hive will be deleted in the tutorials. You have different exercise and different hands on that will let you practice and see physically the difference between these 2 types, and I recommend really to to go over these 2 tutorials to understand better and practice by yourself these 2 types. Now, yes, Jeff. Jeff Liu: Yes, just one quick question. That's the give a location. So it automatically merge all of the file under this location data and put into the table now. But can I just like I just want to those one of this file in the in that location? I just set the location pass, and 5 or less. Yeah. Khaled El Tannir, M: The location can be any directory you create on Hdfs. Jeff Liu: Yeah, but the give a location. One same location. There are maybe a file, but I just want to load one of these files. Khaled El Tannir, M: If you don't want to load all the files in this location, this is your question. Jeff Liu: Don't want to load all the value. but in this case you need to remove. Khaled El Tannir, M: The file. You don't want to to be loaded in the hive from this directory. Jeff Liu: Okay, okay, so I cannot. Khaled El Tannir, M: By default. Hive will read all the files in this directory. but if you don't want, you need to remove it and store it into another directory. Jeff Liu: All good. Is it one of them. Khaled El Tannir, M: What is it? Jeff Liu: Yeah. Yeah. So it's auto- automatically load all of the files in this location. I cannot select the location. For for example, location. I don't. Khaled El Tannir, M: What you can do. You can create different tables to point to different locations. Jeff Liu: Yeah, yeah. Khaled El Tannir, M: Yeah, if you if you have, if you want to separate these files, you can create different tables and specify these different locations. This will let you also maybe give different access level to this data. Jeff Liu: Okay. Khaled El Tannir, M: No about partitioning. So hive provide partitioning types. So we have 2 tables type in internal and external user management and hive manager and I provide partitioning type, static and dynamic. So in the terminology of hive static is known as strict mode. and the dynamic is known as non-strict mode. No dynamic partitioning is used only, I would say, in a development environment never, never, never in production, environment by default, partitioning, direct partitioning is disabled in the recent version of Hive it is enabled, but it is limited because it will consume a lot of memory. It will. The dynamic partitioning what it is, and static partitioning what it is. The static partitioning means you as a user, you provide the partition to hive. Remember, if we have, for the example, I gave it to you earlier. I have my sales data. and I have different countries. And I choose to build my partitioning strategy based on the country column. So here I can do a select star from my table where country equal Canada, so it will return all the rose for this for the Canada country you can now store physically, this result as a partition. This is what about is about static partitioning. So you provide the work, clause and store the result as a physical partition. This is very efficient. It can be scripted, of course, but this is very efficient, and the static partition are faster. Then the dynamic partition and static partition can be altered. you can modify while dynamic partition you cannot. Now the dynamic partition. This means you have a file, or that table. You load the table into memory and say, do for me the partitioning. So it needs to load all the data into memory. Do a group by and then split into these groups. Physically, then, store physically, this group as a partition, so it consume a lot of memory, and by default it is limited because it consumes a lot of memory, and it is not very, very efficient. It takes a long time. It's a very time consuming. So this is why dynamic partitioning or partition. Your data, do it with a spark. And this is what we used to do. We have our data set, we take, we load the data using spark we clean, we prepare, and partition within spark spark will output the partition for us. Dynamic partition can be used to to do some testing. It is not used in production environment. It is just used to be to do some testing. So it will let you as a developer load the data just to check. If the structure is good, you can sub instead, using a hundred 1,000 of row, you can just apply your maybe so 100 rows from your table and see how it behaves. If everything is fine. Now then, you can write your script to partition statically, and provide this in in this to have as a static partition. so static partitioning are preferred, they are faster, and it is up to you to provide the work. Close the result of the workloads to be stored as a physical partition, dynamic partitioning. Again, it loads everything in memory, and here there is no need to provide any where closes as it is done by the system in memory, it will do a group by, and then it will extract these groups and store this on the disk as a partition which is time consuming, and it is memory consuming. and by default it is limited, limited. If I'm not wrong to 100 200 partition, it is so if you are again, if you are doing very, very small data set. Yes, it can be done in the development environment, but not in production environment. the bucketing. Now, again, this is the technique. How to decompose your to organize physically, your data inside this partition and to crate buttons will be using the clustered by close for partition, we use partition by for clustering. We need to use the clustered by just to make it some difference between groups and partition. and again, it can be done with or without partitioning, so it is not mandatory to do bucketing whenever you do a partition. No, you can. A table can be not bucketed and not partitioned can be partitioned, but not bucketed, could be bucketed, but not partitioned, or can be both bucketed and partitioned. So we have all the combination for this partition and buckets. So here again the bucketing is not enabled by default, and inside a bucket we can enforce sorting. I'll give you an example. So you are a real estate agent, and you always query your State data set. And you want to know, for example, the price sorted ascended in ascending order. So what you can do, for example, you can take this data, set a real state data set. You can partition, base it on, not the county, let's say, a state, or a town, or or a city inside the city. I can cluster the data base. It may be on a street on the street name and inside the bucket. I say, I want the price to be sorted ascending in ascending order. So this way, when you query your data, you say, Okay, select star from my real estate table, where city is Sacramento. So it will return all the data for these cities already sorted. So you see, this is but by default enforcing the sorting is disabled because it takes resources and bucketing. In the previous versions of of hive. It was disabled. But now, in the recent version of Hive, it is enabled because they changed the algorithm, to create buckets and organize data, and so on. But this is good to know it is enabled. But sorting so enforced sorting should be enabled by the user. So this is an example of partitioning strategy. So here I have my sales table and my sales table is partitioning based on the date. So here this is my 1st level of partition. second level of the partition is the month. Okay, and the table is bucketed and the bucket number is 6. So now you can see each directory, which is because it is a physical directories one per year, and a subdirectory one per month. and for each partition I have 6 bucket. Now, whenever you query. you say, select star from where, from sales where year equal, for example, 2,010, and where and month equal. 5, it will go directly to this part Directory and read from these buckets. We can also choose from which buckets to read. But let's say it is already optimize it, because here we just go directly to this physical location, and we don't need to do a scan table filter in memory whatever. Nothing. Just go to this location and read from the buckets. So here we are optimizing the the way to to get the answer or the result of the query. So you see where it says import how important is to partition your data and organize correctly your data based on good. It's partitioning strategy. When you output hive hive can read different file formats. can read Orc. Limited text, Csv or Tsv Avro parquet sequence file can be. You can read Json, Xml. So, but some are built in, but not all in our environment, we'll be able to read over C. Arrow, parquet. Sequence. Json. All this you can. You can read, not not sure for Xml. But Json, for sure now how to load data into hive. We need just maybe 5 min, and then we can go to the break. So how to load data into hive. Very simple. You need. If you are creating internal table, just use the load statement. Load statement will load data from Hdfs, and you need to specify the format, or use the one, use it by default by half. If you are running text file or or binary file, we need to choose the corresponding format. If you are reading text, we need to use the text survey. If you are reading Json, you need to use the text Csv or Tsv. If you are reading Json, you need to use the Json Survey survey or C or C survey, and so on. To, you can populate a table using the lot statement. populate a table using an insert query, you can say, create table as select. The result of the select statement will be used to create the schema for the new table, and the data from the select result will be used to populate the new table as select all or create table and insert into, you can use, say, insert into table the table, name and the select from the source table. Also you will be using spark or knife later, starting from week 7 or week 8, you'll be using knife to provide data to your hive or trino or spark. Okay, so different ways to to put the data on Hdfs, and then it will be available to hive to be loaded on in hive from Hdfs. So the hive query language very simple is provide the select the where join there is no join. Don't think relational. If you think as a schema, for example, I want to retrieve my product, name and do as you are doing in your rdbms creating relational table. This will. I would say, this will kill the performance in hype. Don't think relational. Think independent table. But sometime we need to join 2 tables to get data from this from the second foreign table. But don't think when you build or you create your tables or models. Don't think a relational model. This is completely different should be independent. Okay? And you can use, of course, Ddl and Dml data definition language to create database, create tables, create views. You can use the data manipulation language to select, to update, to drop. So all of these are provided. Again, it is not the complete sequel. Okay, the complex and primitive types. So teeny int small int int big int is for long the long type Boolean float, double string, timestamp, decimal binary. The most complex part is when you deal with dates, dates, and time. because the hive needs a particular date. Time formats. and this make little bit more complex or difficult to deal with, because hive needs the information of the millisecond. the millisecond. If it does not, it is not provided. hive will not be able to parse this value or the value as daytime column. So this is why we need to deal with different function convert into unix time. And so you have a dedicated tutorial on the sandbox will to let you practice this part because it is a little bit difficult, but it is important to know how to deal with, because hive needs the millisecond information which is not always provided by the timestamp you can read from Internet, or wherever this about the complex data type, struct map and array. Again, struct will let you nest more than one column into one column map, which is a K value to represent a K value and the array just kind of a list. The column will be as a list. So before continuing the example, we can go to the break, and after the break we can just go deeper into the hive. Query, example, do you have any question before we're going to the break. Thanos Michailopoulos: That's okay. Khaled El Tannir, M: Okay, great. So let's go to the break, and after the break we'll go deep into our hive model. Hive query language, and so on. So I have question here. In other words, it's good to create specific directory for adjusting data. Yes. it is always good to organize ingested data, and especially when the data has different schema. For example, let's say we have to analyze the Bixie data, the shared bike in Montreal. So you have the file for the stations, names and location, and you have the file for the rides, which is different. You have a starting time ending time. Member, not member the so these are 2 files, but they have different schema. So you need to create separate directories to this for each schema for each type, because otherwise hive will not be able to load everything from the single directory. When the data or the file are in single directory, all the files should be the same, or should have the same schema. This is very important. Okay, so let's say our, let's see our 1st hive. Query, language example. Very simple, very very simple, so I can use short tables, so the short tables will list everything, all, all the tables in the current database. So here I didn't specify any database, so it will list the current database. So the current database by default is default. The name of the default. Database in hive is default, and most commonly it is empty. because always we create our custom or own database, and in which we put our tables. So here we assume that this table or this create table will be created in this database, as I didn't provide any database name to be created inside this database. So now you can see, create table. Shakespeare frequency, integral word, string a raw. Just take a look to the 1st line, create table, Shakespeare, frequency and word string. This is enough to create a table schema. This table type is internal or external. Yes, no. Jeff Liu: You have. Khaled El Tannir, M: Sorry. Jeff Liu: This table is the interim. Khaled El Tannir, M: Correct. So it is internal, because I didn't specify anything else as external keywords. So this means the data is managed by hive. The schema is managed by hive. So when you create table Shakespeare frequency world and and hit, enter hive will create your table. But in the back end, in the back office hive will add many, many meta information. One or some of this Meta information are. row format, delimited row format. Delimited fields terminated by tab characters store it as text file. No, this is added by default. When you create your table, I recommend always to put this information, because when you come to maintain your table. It is easier to understand the file type binary or text. In this example it is a text file. What is the character. Use it to separate column. In this example is the tab character, which is the default character. and it use row format delimited the limited field. Okay. So if you type create table Shakespeare, it is enough to create your table. But if you hive will add this line. But I recommend always to use this full definition. So create table row format limited. So we understand this data is organized in rows. the limited format or the limited fields. We understand that colons are separated using a separated character. Character, separator. column, separator. Sorry. And this by default is tab. If you want to use the comma, just change this to comma, replace tab by comma. store it as a text file later, will you learn how to use or to load other format than the text file. Now, as it is internal, I need to populate the data explicitly, because I need to to to load the data explicitly into this table. So load data in path. Okay, here I am providing the Directory. Okay, Directory or or the file name doesn't matter. In this example. It could be the directory name or the file name, but it should be located and the current directory of the current user, because I didn't provide the full Hdfs path. I just provided Shakespeare frequencies. So this, if it is a file, it will, should be located physically in the current directory of the current user. If you are running as training should be. User training and Shakespeare frequency should be here. If you are using Zeppelin should be user Zeppelin. And this file of all this directory should be here into Table Shakespeare. Now we can run very simple statements, select star, select all the rows from Shakespeare, where frequency greater than 100 sort by frequency. Ask is by default. You didn't specify anything else. Limit to 10. Always, always, always limit. Think to limit your result always because you are. We're dealing with very large data set, and we don't want to get 100 or 1,000 of rows to the credit, but by default. When you are running with Zeppelin. It is limited, so it is limited to 1,000 when you load and limited to 20 when we when we show in in cell. But this is, let's say, always sync to to to limit. So we don't have to put the file extension when loading. No, it is, if you can put it. if it is, if if the file exists with dot, for example, Tsv, you need to put it, because otherwise the system will not find it so. It should be exactly as it is stored in this directory, or where to find the the files. But here we don't. We are not using this extension, but we could. If the extension exists in the file name, you should specify this if you are loading the file. but also, if you don't specify an extension, it could be used as a directory, so a hive will try both directory, name or file name. So let's see another example. So on the left create, we create a hive manager table. So create table products. row format, delimited fields terminated by comma. So here we understand that we are reading a Csv file. And this Csv file has no header because I didn't specify anything to remove the header. But I specified only the colon separator, which is the comma. Okay in the right example on the right, the we are creating a user management table. So create external. You can add, of course, for both. You can add, if not exist, for external or internal tables. You can add also, if not, or use, if not, exist, for databases, not a problem for views, not not a problem. So it is applicable for views, tables, and databases and indexes also function if I'm not wrong, so don't you can use it. No problem. So I here, I declare, or I enumerate my schema. Take a closer look to the year column. Name so column name here is enclosed into back ticks. Why? Because here is reserved keyword, and I want that hive. Interpret this as a literal string and not as a keyword. So I enclose this into 2 back ticks. You can add comments. comma, just a kind of a documentation or description. Then you can add to the table, and then you can, as you can see, row format, delimited fields terminated by comma store it as a text file location. Where the data is located. From this location the data will be loaded again. Here, Hive will read all the files into this directory. We don't need. We are specifying a directory. We don't need to specify the extension if you want to specify to the extension. But in the load statement, not in the location, location is always a directory, but in the load statement you can specify the file. Name and extension location is always directory, always. Okay. So now let's see, for example, how we can load data into a partition. This is a way to load an existing partition into an existing table. So load data in path. So I have my my file already in the text format, I'd say, into table. The table name is page view the partition because the table is partitioned. Okay, so I need to specify the partition colon partition K and partition value. So, partition date equal. Okay, 2820 and 8, 6, 8, June 2,008. So this is date. Equal date is our partition. K equal is our equal sign, and I have the date, which is the value partition value country equal us. So here we understand we have 2 levels. It is hierarchical partitioning. So the 1st level is date in sub level is country. Okay? And now hive will load this input file. Pv underscore, 2,008 June 8th US. Text. This is the data stored physically, and it will be loaded as a partition. This is a way. If you have already the data outputted by another tool. But we'll be using other techniques. I will show you this in the tutorial more easier and faster if you need to output or export the content of a table from hive. so you can use the insert, override for, or not override. If to replace the the existing. If already exists. it is insert directory, overwrite to replace, if any already data exist and you specify the output path. But here we need to pay attention to the access policy. The account running the hive should have right access to this output directory. So you are using, or otherwise you'll get a permission denied. and the result of the selected statement will be stored as a file into this directory. You'll see many, many example. We don't choose this insert Directory from directly from hive, because in our pipe. in the pipe model, in the data model we collect, we prepare with the spar. We will be using hive and trino to analyze only. and later we will see what are the best techniques to connect 3rd party tools, for example, power bi tableau whatever, but not using this insert directory. This is, if you are just want to export and need this data, you can. You can do it. But again, for all this task we can use spark. So this one word about the survey Saturday, as I said, is used to just to to to serialize and deserialize data from memory, and disk to read from disk into memory and from memory to go back to the disk. Okay? So again, some of these survey are already embedded to read the Csv file to read many format parquet of row Orc, all already configured for you, so you don't need to do it. The G. 7. It is external, but you need to specify it. It is not loaded by default, but it already exists in the tutorial and workshop. I show you some examples. If you want to to practice or to load some Json file. How to connect. to, to hive, so you can use the Zipline interface. You can use the B line. This is the tool for the administrator. It is not very sophisticated tool. This just come online with very basic text, display everything in text but it. Let the administrator use this tool to connect, to, to check and validate. If the connection is working, if everything is fine about the accessibility, the authorization authentication, whatever. So it can be used. It is available in the sandbox, but I will say it will not bring a big value because you are doing everything from Zephyrin with a better interface. Also, we can connect to hive using external tools and the tutorial in the guide. You have the instruction to connect the dB visualizer, which is a free database visualization tool, so it can connect to many, many different databases. Hive also, so you can connect to the hive sandbox hive from this tool also, sometime this the administrator needs to script some and automate some instruction it can use directly from a file using the hive command, hive hyphen, F command. It can run some hive. Query, language tool. Okay. Now for the workshop. I will just show you this little bit. And after I will go to the tutorial in the in the sandbox. So I've provide you the Ddl type. And Dml, so, data, definition, language, and data modification, language. We have all these comments so, and keywords select from very similar to the to the SQL. But it is not. Everything is just a sub sub system. Again join. You can join 2 tables or more, but it is not very efficient to think as relational word. So just here, we need to denormalize, I will say, to be very performant and efficient. Okay, so these are our complex struct map and array. We have also some examples of how to use this type in the workshop. Now, the create table syntax. You're already familiar with this in this previous example. Create table, provide the schema column. Name and type always my recommendation. Use the full create statement. So row format, delimited fields, even though maybe it is not mandatory to add a comment, but always use row format. The limited fields terminated by. If it is the case, store it as we need to you to use the dedicated format. Yes, some. Sam Wanis: Yes. So for Let's assume that we have a Csv file, and it has a header means that it has column names in the 1st line. And is it possible that to map the names of the file directly to the column names in the table, instead of using, create table or to use it in a different syntax, I mean, or this is not like, not not not possible to be that. Khaled El Tannir, M: Do you want to infer the column names from the Csv. Sam Wanis: Yes, based on using the 1st line. It's not. Khaled El Tannir, M: Nothing happened. Sam Wanis: Yeah, yeah. Khaled El Tannir, M: In spark. Yes, but not in hive. Sam Wanis: Oh, okay. Okay. Thank you. Khaled El Tannir, M: Okay. So now, this is an example how to use the struct or the complex type provided by hive. For example, create tables, employees. The subordinates is a list. The list column is an array type. So array string so you can provide the different names. But here also we need to specify the collection separator. So, for example, inside the array, we cannot use the same as column separators. It should be different. We can say collection separator, for example, columns. 2 points dot the current. So here you can use an array and separator and to separate element. Inside this array the deduction map, which is just a map. The K is always a string, and the value can be any type. And struct, as you can see here, you can nest different or more than one column into one single column. So the address, here is Street City. This is a column. Store it into a single column. a street, city, State and Z row format. The limited fields terminated by tab. The tab is the default character used to separate columns stored as a text file. Now for external tables, so create external, very important to specify the keyword external tables. You can provide your your column name and column type, row format, delimited fields terminated by tab. This is our default character stored as a text file location. This is the Directory. From where we need to load the data or populate the data. So great table, we have. When you do a crate table so internal table hive will move. When you run after the load statement hive will move the data from its original location. The table location on Hdfs, for example, you are loading the data from you created the Directory Lab one. and you put your file inside lab, one on Hdfs and you say, create table customer, load customer load data in path from Lab one into table customer. So hive will move the customer file. use it to populate the table from the lab one directory to the location inside the hive warehouse. So the data is moving. This is why, when you drop the table, have dropped the directory of the table and everything inside this directory, which now the case is our data file, it will be deleted. create external tables. Nothing is data is not moving. Stay where the user define the location. So this is why, when you delete the table or drop the table data. Still. an original place the partitioning. When you create a partition. this will make your query faster. Of course, physically it will divide your database on partition, Colon. You will be using the partition by close when creating your table partition table. and if you want to load explicitly data from a physical file. You need to use the partition clause. No, I will show I will show you this in the in the case study and the tutorial. When we have many, many partition, you can add this manually alter table, add partition, as many as you want as many as you have. but in case you have hundreds or 1,000 of partition. how can you do it manually? It is not possible I used to do. What I used to do is to use the Ms. Ck. Command. So the Ms. Ck. Command. I will show you this in the workshop is here to add every missing partition. So Ms. Ck. Repair table, add all the missing partition. So this means, if you are running this from hive to load partitioning table. And this table was partitioning, using spark or whatever it will load all the partition for you. using just one single comment. So which is this is what I use to to use when we. When I load part sheet into hype. you can list the partition of a table when you run the show partition, show tables will show all the tables in the database and show partitions will show all the partitions in particular table for the bucket. It will speed your query. Okay, use the clustered by it can be added also to a sorted bias. You can say my data is clustered and sorted, so both we can use both K. Words, and if I want to get data randomly from any buckets, this is useful for machine learning to create or extract randomly rows from a particular table, we can use the table sample table sample will check randomly, get data randomly from buckets. So this is the example how to do sampling and run extract randomly data. So you say, select. So insert over, insert overwrite tables. This mean the output of the select statement will be inserted into a table. So select star from Pv. Gender sum, table sample bucket, 3 out of 32. This means it will use the buckets starting from 3 to 32 randomly extract data from these buckets. So this very basic commands. Just show tables. Show tables. Page. So this will give you a regex to to list the tables. Names, partition describe is very important. Command, describe, extended and describe, formatted, so describe will show you the command or the the information about the table column. Names and column type describe extended will give you more information about the partition, about the bucket, about the file format, the physical number of file format, but also it will show comments if you use the comment statement in the create table, but it is. return it in a single column. It's not easy to read. So we have described formatted, which will show you in a table. You have this example in the workshop. I will show you this in a moment. and describe partition. You can describe the information about a particular partition, so do you have any question. Before I went I go to this to the, to the workshop. I was showing. So let me open this. Okay. So from from the quick links. So here you have, and the hive, we have 2 links. The 1st one is. So here you don't. It is only for information. There is nothing to do just for information. It kind of logs, for in the hive server. So when this user interface shows up this mean, your hive server is running, which is a good thing. And here you can just have all the history about all your query. You can just go to look at logs. But again, this is more for the administrator. But it is here to to show you that everything is good. You can go to particular query, and you can show the stay, execution stage the plan. What I've executed to be able to return your data. This is what the Administrator will use to optimize the query. We can analyze this to see where time is wasted, which part is time consuming. So we can analyze and get. say, to prepare to be optimized, based on this plan and the performance logging. So again, this is more for information. Here you have the test web interface. So this will show you the execution graph of each hive. Query, because not all the hive, for example, a select query will not generate a mapreduce. Application will not use test because select just listing the content of a table. But when you use, for example, account, or an aggregation or something else. This will generate an entry in this table, this interface, and here you can analyze again better than the hive server interface, and you can see that the configuration is the timeline, where time is consuming, what part is taking more or less time, and so on. So again, this will give you an idea, but this more used by the administrator. But it will give you an idea. Okay, so now let's go to the guide before going. So here you have the connect database to dB visualizer. dB, visualizer is a free tool. You have the community edition and the pro edition communication is free. And here from here you can just install it straight forward. And this is how to connect, hive to to to this tool you can now explore hive from the external this external tool. Okay, you can also install the Odbc driver you can choose. And here you can use the Microsoft Odbc driver for hive, and once it is installed, you can connect, you can just test the connection, and you can connect any tool support, external application supporting Odbc connectors such as, for example, excel. Okay, now, in the tutorial, the tutorial, we have here many tutorials. So let's start with hive. So here we start focusing on the user management type tables and and hive manager tables. So you have here both. So hive user management tables and the hive manager tables. So here you you, I will repeat or explain what I explained. In in the class you can work with, you have an example. and by the end you have your zip zippering note. Okay, so it will open and you can practice like, follow. create a database. I will show you this. For example, let me do this little bigger, for example, create database if not exist. So here we are using the hive interpreter. Okay, you can run this. Okay, finish it. We can list. The databases exist already exist in hive. Okay, so here we have 3. Okay, maybe you do not have exactly the same as me. So high create table, you see, create table ex. This is an internal. I didn't provide any external keyword or whatever. So it is created. So row format, delimited show tables in tutorials. So this will list all the tables in this particular database. Let's run this. Okay, we have customer and parquet. Okay, another table. Now, I want to upload the data to Hdfs. So this already familiar to you. So put and the customer, you see, load data in path from this location. So I want to to read this file single file, so I specify the name and the extension overwrite. If already the data is already loaded into the table and the table name into tutorials. Customer. I recommend to use always this notation. because it is easier to know and to to where the table is located. You can use the use keyword. But later you don't remember where the table is located. So I recommend to use always this notation table database, period, table name, database, name, period, table name. Okay. Now, the data. It is not loaded. No, for I didn't run this one. Okay. okay. And now we have our data. You can see the data is loaded correctly. No problem again, for I will show you how you can remove the header, but it is only logical. Now, if you do a describe, disk or describe. This is exactly the same, so you can see it will show you the column, name and column type for this table. Now we can do our select statement. You see nothing special here. Okay. in the in the tutorial. You have also the second one, the high user manager table. and you it will show you. When you delete here we create the table, and in this one you will create the external, and you, delete, drop the table. And you compare the result. Okay, this very, very similar. But here we are using the where it is. So okay, we are using the external keyword. So let me open this in hive in the sandbox. So from hive and this user management table, okay, so create database show database. Okay. and here we are loading a different file product. Csv, okay, let me load this. So here we are loading the the creating the table. So create external table tutorial product external. I used to ex use this extension because it is more again when you come to maintain it is more readable, and you know already if it is an external table, just by reading the name. Okay? So I define all the column name as a schema raw format, delimited field terminated by comma stored. But as a text file. And this is the location where, from where to read the data and remember, that has been already uploaded to Hdfs if I do this select. So I have my road, my data already loaded. And here. So you see, we have the drop table. So lets me. Let me drop the table. Okay, it is drop. And if I go to my browsing Hdfs. Sorry. So here the data should stay in the home. Sorry tutorials hive external. Okay, so, tutorials, let me go. Do this little bigger. Hi, external and product. You see, the data is still here. which is not the case if the table was internal. Okay, now I can recreate the schema. and I can have again my, that if I drop the file from Hdfs I will get the empty table so I can go to my Hdfs. I can delete this. Okay, and when I rerun I get an empty table, I just re upload my data to Hdfs. okay? And I rerun. I get my data back. So this is the difference main difference between external and internal tables. Okay? Now. yeah. And so this is a way, if you really want. But again, and only in development. If you really want that, when you drop an external table hive. drop also or delete also your data. You need to use this property table property external table push. True. So this will delete the data from Hdfs if you are using the external table. But again, this should be only doable in the development environment, not in not in production environment. Okay. So now, this is just about the hive manager tables. Now let's go to partitioning. So you have all. Where is my tutorials? So you have the partitioning here, data, partitioning and bucketing in hive. So you have everything you need to know about how to partition here in this tutorial. So I will go directly to this hive data, partitioning and bucketing. Okay, so here we are loading. Here we are loading data into an internal table which is a staging table. Okay, I can read this. So I am uploading data to Hdfs. Okay, I create the database. If does it exist. And here, as you can see. I create a table. So Olympics and raw format limited for by T dogs, this is my, and I load the data from Hdfs. Okay. this is meant. This is my source table. Now, I just will check. If the data is loaded, the data is loaded correctly. Now let's create a partitioning table. create table. I provide the schema and the colon partitioning Colon. So here, if we compare. you can see in the order in the staging table. the country column is part of the schema. When you go to partition, the country Colon has been removed and added to the partition by close. Okay. So when I run this, it will create the table. And now, to populate the partition I will use insert into country equal Canada, and the result of the select statement I provide the where close. So the result of this select the statement will be used to populate the table Olympics part, which is partition table. And this is static partitioning. Okay? So now it is loading and partitioning physically on the disk, so we can just here finish it. Let me just show you where it is located tutorial hive, and by default here I just created my Olympics part, what is my Olympics part? Okay? My. So if I go to browse hive user hive, warehouse tutorials, Olympics part and I have my colon partition, colon, partition K and partition value inside. I have one bucket. Okay, one bucket. This is a file, we can browse this content. Sorry? Okay. Now, I can add a new partition. Okay, you can see, show partitions. And see. Now, I have one partition which is county equal Canada. This is case sensitive, very important case sensitive. Now I can add a new partition. It's exactly the same query. I just change it the partition. They were closed, only the criteria of the were closed. So here country equal United States. And this is also, if I go to my browse. I just go back. And now I have 2 partitions. If I refresh partition, I see that I have 2 partition. Okay, so we can. Here I am using the dynamic mode. So it is enabled. As I said, it is disabled by default. But again, this should be always only use it in development environment. Okay? So I will let you just continue this. Nothing special. All, everything is similar to what I did in the partitioning part. Okay, now, I want to show you what is the very. So this is, let me show you this. Okay, so this is the 1st part of the workshop. So here, these are these main 2 tutorials about hive manager and user management tables and data partitioning and bucketing. So we have the same partitioning approach with Spark. How we can do partitioning with the spark, and I will move to this case. Study in this in a moment. So if you go to spark in the tutorial page from spark. We have also the partitioning. So data partitioning in spark. The concept is the same. We load the data. But here you have more features. You can infer from the header. You can change color names can do whatever you want, and let me show you the output. So here we have our file. So here we are loading our input, we specify the header, the schema, if it is coded, not coded escape delimiter. Everything is here where it is located physically on Hdfs. And let me show you the partitioning. So it is here. The so the repartition colon you can. This is, I would say, not mandatory. But this will impact how your data is partitioned in memory, how spark will use the data to partition in memory. So here the repartition colon will be used to define the output. You have all this information in this tutorial, but what it is important because this is optional, this part, we go bank. The data frame name, write function, mode, overwrite. Okay, partition by partition by here it will do exactly the same what we did in hive. So here it is dynamic partitioning. This is spark. It is more optimized than the Hadoop hive. So here you can load everything in memory. Do the group by organize and output to this directory. Okay, so it will output everything to this directory. So and again. this is not a file name. This is a directory name output directory name. Okay, I will run this. This is the explanation about the repartition function. Again, it is more optional because we are going more. One step in the level for further in the level. But again, just focus on the right function, mode partitioning by and the output. So here we are using the output as a parquet, because I want you to be prepared for the next classes, and the parquet is nothing but for now the binary format, and this is the default format of spark spark. We can output as a text, but it is not the default format. The default format in spark is parking. Just use it now as is, don't worry. We will have all the explanation in Class 6. Okay, so go over. Focus on the partition by not necessarily the repartition, but understand what it is used for. Okay. So here, if I go back to my spark and I go partitioning data, not aggregation, but as partitioning. Okay, so this is, this is hive. I want spark. spark, participant data partitioning and spark. Okay, so I can load here our file. Okay? And we can read this file. Okay? So this will load all the contacts. The driver for spark and load the the file, so finish it. We can check the schema if you want to check this out. Okay, this is working. We have all the data. And here, as you can see. we can run this partitioning. Okay, so let's go to the browser. And we are tutorial spark data frame tutorials, spark data frame. And I have partition, and I have my output one directory. And, as you can see, I have my 4 partition based on the marital column, which is our partition. K and I have my different K value partition values. And this is the file outputted by hive, by sorry by spark, which is the snappy parking. Okay, so go over. It is very important. It will. because we'll be using all of this concept later in the course. Now, the for the case study. This is a very important one. It will. You will learn a lot from this one. So here, this case study, we have 2 parts. One part is for hive, and the second part is for Trino. So as you can see on the. So here we have everything, and here you have both. Zepri. Note, one for hive and one for Trino. For this week we will focus on the hive part. The next week you will go to the 3 row part. But what it is important here is who this we can deal with this. So we are loading the file. It is compressed file, zip, file. So we expand the file. We create a directory. We load on this directory on Hdfs, we load the file using spark. So here I am using a cache to put this into memory to improve the input output operation access. I do some normalization. And here I am outputting to to to Hdfs using the partition K year, a particular K and the in parquet format. Okay? And from hive. This is very important here. Create a database, create an external. As the data is already outputted by spark. So here we just change it, store it as parquet. It is not stored as a text file. Okay, to store it as parquet as from spark. We output it as a packet we can use, save it as text from spark, but let you can try it. But but here I provided you as a parking, because it is smaller. As you can see, the file is very smaller. Okay, but as the table is parquet, we need to specify the compression by because spark. output parquet file with snappy compression. So we need to tell Hive that the park parquet compression is snappy. This is case sensitive. So should be written this way and from here. This is the command, Ms. Ck. Repair table, which which will let you add all the missing a partition. So from here you will be able to load all the partition created by spark into the when you do the crate table. The table is empty. Okay, because it is partition table. and once you run the Ms Ck. Repair table and show partition, you will see all the partition already loaded, otherwise it will be empty. And now you can just run the queries. Now, after running this before running any query. this is the what we call preparing the data to be analyzed. So analyze table parquet computer statistics. So this will let you, or let hive prepare the table to be query one. You can do it without this compute statistics, but it will take a little bit longer. So here, let me show you the output. Okay? So I don't remember if I have already describe formatted. And let's do this. Okay? So you see, if you do a describe formatted here, it will give us all the column name Schema, and we can see here the physical location on Hdfs. The table type, which is an external table bucketing version number of file, physical file. It is 9 number of partitions, 6 number of rows, parquet compression snappy. You see, we have all this information. But if you don't run this analyze table compute statistics, you cannot have all this Meta information, so this will help hive to improve the performance of your query. and then you just here, you just can run any query, and this should return a result. So again, hive was not designed to be interactive. So in the background it is creating a Java application submitted to the hadoop cluster and then return back. The result. Later in the next week we'll be using Trino. And this is interactive tool. These are result returned in seconds milliseconds. So I will let you go over all these examples and practice and understand. So here also in this hive examples, here you have in this let me show you where. if you want, for example, to remove the hive, this header from a Csv. You can choose. I don't remember which one exactly, maybe this one but the in one of these tutorial. You have this no, not this one, maybe this one location not this one. I don't. Remember where it is you have in in one of these we have the example of the syntax how to remove the header column. Header line. No, this is spark no, this is a spot where it is my hive. Yes, this is so. This is a Csv table. and this is an external table. And here we say, table properties, keep header, line count equal one. This is logical skipping. It is not removing physically the header from the file. so just adding this table property properties, this will remove the header of a Csv file. Okay, so this is everything for hive and practices work on it. It is very important because it will be used. This concept will be used from starting from today until the end of the class and the next class will continue. But with the interactive actor interactive query with Trino. So today it was data at rest with very long job with hive. And the next week we'll be using Trino. It is trino will will be using our, I would say, main querying engine from starting from the next week. So do you have any question before I show you your 1st homework. Yes, in in the exercise you have this exam exercise. I already uploaded for you on my courses. The the case try to do it before jumping to the case, and just copying or regard. Take taking a look to the solution, try to do it before and when finish or very, you really are struggling, or you're not able to continue. Take a look to the case, but try to do it before jumping to the to the case. Okay, so here you are just loading the data and preparing the data from Hdfs and reading, using hype. Nothing special but everything you have in the tutorials and tutorial section. You have everything to help you to do it. Now, regarding your this time. Yeah, it is time. So regarding the 1st homework, it is very simple. So we will. You unload 2 files in in in spark, you will be using the spark data set Api to analyze this to this data set. Okay, so this is about restaurant inspections. So the schema for the 1st one restaurant is very simple. You have the restaurant Id, the restaurant name and address and cuisine type. The restaurant inspection is restaurant, id, the restaurant restaurant, inspection, date, inspection, violation code, violation description and the rate grade and score for this a restaurant. So what we need to do preparing the data environment, so load the file to Hdfs and loading the file into spark. Okay. nothing special. You can use the cache just to improve the read and write operation, so it will enhance the processing time. Now, the 1st part in task. One is, use, Scala always Scala, no python, no, Java, nothing scala to. And the data set Api to complete this 4 query, nothing but 4 queries. So make sure to import in a cell. And Zeppelin the implicit SQL. Type and date and start timestamp type because you need to define your case class. So the 1st task is to create the define, the appropriate case class for both data set and then run. These 4 queries display the 1st 10 row from each data set total number of rows in very simple. But using the data set. Api most top top 10, most cuisine types and top 10. Most frequent violation code. Okay, now, using the SQL. Api. So here you are using SQL. Query. you have 3 query in SQL. Find the top 5 restaurant which highest number of inspection you can include any column you think it is relevant. Identify your top 5 cuisine type associated with the most violation type of description. compute the average inspection score per bottle. 4. Simple query in data. Set Api, and 3 simple query in SQL. Api. This is your 1st homework, and you have one week to submit. So what do you will submit? You will submit 2 documents. 2 items. The 1st is a word document with this name. summer 2025, homework one include the full name, the past relevant code from your Zeppelin note limit to 5 or 10 rows, not more clearly. Label each task, and provide your comment and explanation where applicable in French or in English. and a copy of your Zipline notebook, so I can read and can rerun and execute your statement. So submission, policy, no submission or withdraw grade 0 late submission. So if you submit after Wednesday, midnight. 15% penalty per day, and if you use a robot to generate your coach, adpi deepseek gopilot, wherever you get 50% penalty, don't do it. so do not package, please, your file or submission in a Zip file, because we are not able to read it on my courses. We need to download and expand, so we don't. We cannot read this on my courses, and do not ask to debug your code. because this will be treated as a submission. so try to do it. You have all you need in the tutorials and the exercise everything you need to solve this very, very simple. You have everything. and of course this is an individual work. Please don't copy, don't provide your solution to your classmate. Even you are very good friends. Don't do it because it is not good. This, considered as plagiarism, and it will. You both will be penalized. So don't do it, please, and don't share this assignment with others, so do you have any question. Rajesh Kamaraj: So tell you the the due date is 4, th or is it 8? th Okay. Khaled El Tannir, M: No, it is 4 4 June you have one week today we are Thursday until Wednesday, midnight after Wednesday midnight. This be considered as a late submission any other question. Sam Wanis: So the submission on the portal, or or by email. Khaled El Tannir, M: Only all my courses. Only do not submit anything. Do not send me anything by mail. It is not accepted only submission by all my courses. Only let me just reconnect because I was disconnected. I need to reconnect as I show you. multifactor. Okay, so now we have accesses. Okay, okay. So here you, when you submit here you it will be. I will see your submission, name and everything. And this is, you see, due on June 4.th This is the date after this date. Date time will be considered as delayed submission, and it will be shown in red in your in your submission. otherwise it will be considered as regular submission. Rajesh Kamaraj: Can you? Can. You just show. Khaled El Tannir, M: No, no, no submission by email. If any code doesn't work for the assignment, can we send your you an email? No. because it is not a workshop, it is a graded workshop, so I cannot provide any individual debug or answer your code. If it was a workshop, an exercise, yes, but not for the graded one. and if you send me and ask me, it doesn't work. How I cannot do it. I do not have the expected result. This will be considered as your submission, so don't do it. please. Sam Wanis: And just worst case scenario. If we manage to finish like part of the assignment we will submit it before the deadline, but for the other part can we submit it later with penalty, only applying for the remaining. Khaled El Tannir, M: If you submit more. No, if you submit anything submitted after the due date, we get penalties. But if you submit in more than one part we, we will take only the latest of submissions. So if you put 2 submission, and in the first, st submission is your task, one and second partition. Submission is task 2 only. The latest submission will be taken in consideration. So don't split your task. Submit one document a word, explain everything you want. Don't submit after the due date, and everything will be fine. Rajesh Kamaraj: Any other question, can you can just show how to access those Csvs from. Khaled El Tannir, M: It is. Yeah, let me show you this. Csv are already in the on the sandbox and let me show you so this is the location. Rajesh Kamaraj: Okay. Sure. Khaled El Tannir, M: Can go to my. So you don't need to explore using mobile external. But just provide this location in your Hdfs command, and you should be able to load this. But let me show you. So in home training. data and restaurant, you should see this, these 2 files. There is more than the files we need for this homework. So you have here we have both restorancy Csv. And restorancy inspection. Csv. This is the 2 file you need to to process. The 1st one is 2 MB, and the second one is almost 78 MB. So. and you will see spark will be able to load this very easily, and there is no as you can. I can show you the next week later, when I will show you the case for this Homer, you will see. Spark will be able to load this in seconds. But this this is where the file are located. You don't need to download anything from outside the sandbox. Everything is already in the sandbox. Rajesh Kamaraj: Okay? Oh, I'm not sure. But I have a group. How do you access that? Because I have a very different interface. Khaled El Tannir, M: Is it? What interface you? You have. Rajesh Kamaraj: It's it's still that you know the ubuntu. Khaled El Tannir, M: About this interface. Rajesh Kamaraj: Yes, yes. Khaled El Tannir, M: But you don't need to go here, because this is only for logging. So this is our external Ssh tool. You can from the guide. You can go here. and this is how to install it. Rajesh Kamaraj: Okay. Khaled El Tannir, M: So you have all the instruction to connect mobile, external to the sandbox. This is a free tool, but only available for windows. It is not available for Mac. If you are using a Mac you need, we can use any other Ssh tool, putty or or whatever. So you just straightforward and start and connect and get your screen. Nothing difficult here. And I have this tabby, for if you are running a map free and do the same, but it is. The interface is a little bit different. But again, you don't need to go using Ssh, but all you need is providing the directory where this, that file are located on your local system? Any other questions. Hammed Akande: Oh, okay, so just a quick one. Khaled El Tannir, M: Yeah. Hammed Akande: Only need to provide these on the Zeppelin, and it should be able to connect as long as we. We are connected. Right. Khaled El Tannir, M: You. Hammed Akande: Oh, okay. Khaled El Tannir, M: You as you already did, for now for these tutorials and workshop. and when you finish on my courses you submit your Zeppelin, you can export your Zeppelin. Remember you have all the instruction how to export. For example, you want to export this one click on export as a Zeppelin. and this will export your Zeppelin note. Hammed Akande: Okay? So we submit only the sapling notes. Okay, so it's not. Khaled El Tannir, M: Zeppelin and the word document both. Let let me show you a game here, this homework, what to submit a word document. This word document will contain your full name and the relevant code and result you copy from your Zeppelin, but should be readable, not not. Don't copy paste very small. I cannot read this, and for your task, just explain comments and explain what you did. You can write in French or in English, wherever and the second item you need to. I want you to submit is your Zipline note. So I can compare. I can check your result. Maybe use in the word document. There is not. Maybe you updated later, your Zepri Node, but you didn't update your documents so I can check and see if there is some differences or something like that. Hammed Akande: Okay? Well, okay, because I mean to my understanding Zeppelin or Markdown notebook. They kind of solve essentially the same thing as word document right? So. And if we on Zeppelin or Markdown can comment and do all of those things, so if we duplicate the same thing on word, document. Khaled El Tannir, M: Yeah, from the Zeppelin. This is your code, your implementation. And in the word document you just copy part of your implementation and comment what you did. I need to understand what you did. Okay, okay, that should be in the word. Okay? Sounds good. And I need to to know your name. Hammed Akande: Thank you. Khaled El Tannir, M: So everything. This is why the word document can be loaded from my courses directly, but not my, but not the Zeppelin. This is why? We ask, because if the faculty can sometime they ask an external grader to check and grade your work not. It is not always me, or maybe most of the time it is me. But the faculty or the administration can ask an external grader to do the job. So this is why the grader should see exactly a word document, and compared to your implementation in the Zeppelin. Not okay. Hammed Akande: Yeah, thank, you.