database management system is one of the most used technique but why do we need to manage our data and where this data comes from so if you see everywhere we are actually generating data to manage those data we need to have a proper database management tools so keep that in mind we came up with a tutorial where we will learn about sql and that will help you to manage your data so let's get started let's get started and let's have a look at what agenda we have for this tutorial so first we will have a look at what is database right and what are the example of databases we have now we will come up to why do we need database and what is database management system and how this database and database management system are related to each other then we will have a look at the history of database management system then we will have a look at what are the most used database softwares we have and then we will jump into the types of database systems then we will have a look at what are the advantages if we go for using database management system lastly we will have a look at what is relational database management system and what's the difference between dbms versus normal traditional file system and we will also have a look at what is the difference between dbms versus relational database management system then we will have a look at the key concerns concepts we have in RDBMS. Then, we will also have look at what are the types of key we have in RDBMS, the relational database management system. Then, we will have a look at the difference between super key and candidate key. Then, we will look at what is normalization and why do we need normalization and what are the types of normalization we have in SQL. Then, we will have a brief look about what is SQL stands for and how can you install SQL in your local system. Then, we will have a look at the data types we have in MySQL. Then, types of commands we are going to use in MySQL to maintain your data, to manage your data. Then, we will see how can you filter your data in MySQL and what are the operators we have in MySQL. Lastly, we will see how can you use pattern matching concepts in MySQL. Now, we will have a look at what is database. A database is a database that is based on a container where a data can be collected systematically. Managing and manipulating data with the concept of database are very easy. Now, let's have a look at why do we need database. What are the examples of databases we have? First, suppose you have an online telephone directory. If you need to maintain that online telephone directory, how can you do that? That time, you need to have a database where you can store your data. The data is like name, address, and password. You can store your data in that database. You can also store the address, phone numbers, and other contact details. This is where the database concepts comes into the picture. Let's have a look at the other examples of database. Suppose, you have an online library. In that time, if you want to have a look at what are the books you have and what is the count of your book and what are the domain or genre you have in your library, in that time, you need to have a database because in a library, you will have millions of books and maintaining them in a traditional way, you need to have is not possible so that time the database is coming to the picture and the database can help you to maintain your data or for millions of books now let's have a look at what is database management systems so basically a database management system which is called as dbms is basically a software where you can store retrieve define and manage your data in a database so basically we need this system to manage our data to retrieve data from a huge numbers of data and you can manage and define your data as well right now let's have a look at what is the history of database

management system how it came into the picture so basically in 1960 charles batchman designed first dbms system and then 1970 got introduced ibm's information management system that is known as ims so then 1976 peter chen coined and defined the entity relationship model also known as er model then in 1980 relational model becomes a widely accepted database component then the revolution comes in 1985 the object oriented dbmf develops then a huge uh development again in database management system then in 1990s in incorporation of object oriented in relational dbms then in 1991 microsoft ships in this access a personal dbms and that displaces all other personal dbms products in 1995 first internet database applications and in 1997 xml applied to database processing many vendors begin to integrate xml into database or dbms management system products right now let's have a look at what is the most used database management systems we have these are the software we use to manage our data we have mysql we have oracle then we have postgre sql then sq light we also have mario db that is also similar to mysql and we also have some other examples as well now we will have a look at what is database management system, right? So, I already informed that relational database management system is one of the DBMS way that it will help you to make your data is like more easily accessible and there are lots of features we have for our DBMS. So, first let's have a look at what is a relational database management system. So, basically relational database management system which is in short known as RDBMS is used for database management system that we all know. The concept is based on the relation model as introduced by E.F. Codd, right? Now, we will have a look at what is basically stands for and what are the properties we have for RDBMS. The relationship between the data files is relational in RDBMS. They connect the data and the different files by using common data numbers or maybe using key concepts. So, let's see what is a relational database management system. So, let's see what are the key concepts. So, we will have a look at after this what are those key concepts and how we are going to use that key concepts, right? So, let's have a look at what are the properties we have for RDBMS. So, we have values are atomic in RDBMS. Each row is different and columns are also different. Each column can have a common name and the integrity constants help to maintain the data consistency for multiple tables. So, these are the properties we have for RDBMS. And there are lots of other properties also there in RDBMS and that will basically help you to have a proper structured way to solve your or to store your data, right? Now, we will have a look at so what are the difference between database management system and the file system and why should you go for database management system, right? So, DBMS is a collection of data but user is not required to write the procedures to manage the databases. So, whenever you want to manage your database, that time DBMS has its own features which will actually help you to proceed your data. But in the case of file system, it is basically a collection of data obviously but any management with the file system has to write the procedures. So, now we will have a look at the difference between DBMS versus file system and why should you integrate your database to DBMS and what are the features and advantages we get when we use DBMS, right? So, let's have a look at the file system is basically a collection of data that we all know and any management with the file system has to write the procedures, right? Whenever you are going to make your data stored in file system or you want to manage them, you need to have a manual effort. In the case of DBMS, this is also a collection of data but user is not required to write the procedures to

manage the data. So, DBMS basically helps you to reduce the human effort and it is more of an automatic task. So, let's have a look at the second reason. File systems are not efficient for storing and tournament. Let me give you an example that we know we are going to store our ”ز ˆ¶ë ½؟retrieving ofï data in the Excel. We know for the most role of the case that we go for Excel. If you want to go for Excel, when you have a huge number of data, from that data, if you want to retrieve your data, if you save your data, then it reduces your data quality. Well, or if you want to find a data from Excel, it is quite time-consuming and quite difficult. And this also takes a human effort. Whereas, DBMS is efficient to use as there are large varieties of techniques to store and retrieve the data. So, this is why DBMS is accepted all over the world because of its flexibility and large varieties of techniques. Then, we will have a look at the file system. The third reason, the file system does not have a crash recovery mechanism. Example, while we are entering some data into the file, if the system crashes, then the content of that file will be lost. But, whereas, DBMS, in the case of DBMS, if you have a crash recovery option, we already have a crash recovery option which protects the user from the effects of system failure. Right? So, this is why and this is how DBMS is helping us to protect our data every way. And protecting the file using file system is very difficult. So, already we have talked about that file system is not secure. Whereas, DBMS has its own security process to save the data. And you have lots of options by which you can make your data more protected. Right? Now, let's have a look at what is the difference between DBMS and RDBMS. Okay? So, we know there are lots of DBMS we have. For the part of, one of the part is RDBMS. But, why RDBMS are using all over the world and why it is so popular? So, let's have a look at that. So, DBMS stores data as file. In the case of RDBMS, stores the data in a tabular form. In DBMS, data is stored in two way. One is hierarchical form and or a navigational form. But, in the case of RDBMS, the table have a classifier known as primary key. So, we will also have a look at what is primary key. And then, the data values are stored in a tabular format. So, these are the main two key points and the cache for RDBMS. So, basically I will give you a little brief about primary key. So, primary key helps you to make your data less redundant. Right? And your data most compact and in a structured way. Okay? Now, let's come back to RDBMS. Let's talk about the concepts of normalization. So, normalizing a data is one of the most important part wherever you are going to store your data. So, we will have a look at that as well. So, in the case of DBMS, the normalization is not needed for the database. So, normalization is the most important part whenever we are going for the storing our data. We will have a look at that as well. Why normalization is required when you go for storing your data? Right? And in the case of RDBMS, you need to follow all the normalization rules to store your data. Okay? Now, let's have a look at DBMS does not have any security for data manipulation. Okay? But in the case of RDBMS, it has the integrity constraint for the purpose of ACID. But what is ACID? So, basically ACID is basically if you want to break them. So, A for automaticity. C for consistency. I for isolation. And B for durability. Right? So, this property makes RDBMS one of the most popular using data-based management system. Right? Now, let's have a look at the second option. So, in DBMS, there are no relations between the tables. So, if you want to make a relation between your table, you cannot do that. But in the case of RDBMS, there should be a relationship between the data values. It will help you to retrieve your data. To structure your data more efficiently. Right? Now, have a look at DBMS

is used for small organization. We can understand from the features itself only. And always deals with small data. It also supports single user. But, when we go for RDBMS, it is basically used for large data. Because why? Because whenever we have large data, we can make relations between them. And it also supports multiple users. So, this is one of the features that we can use for large data. So the examples of DBMS is basically XML files and file systems, but for the RDBMS, we have MySQL, SQL Server, Oracle, Postgre, etc. Right? So I hope you can understand what's the difference between DBMS and RDBMS. So we will see what are the key concepts we have in DBMS. Right? So, what is key in RDBMS? So, key concepts has an important role in relational database management system. This technique is used for identifying the unique rows from table and also help to establish relationship among the tables. The name itself key is quite mysterious, right? Why this is named as key? But if you can think properly, you can understand making a relationship between two tables is basically the same. So, key terms is used to make those tables, like those tables have their own relationship. And from using those relationships, we can retrieve our data, right? And that makes our life so easy. I will show you how can you actually make a big data set into converting them into RDBMS and how it's going to help you to retrieve your data, to manipulate your data, right? To get the information from the data. From those huge numbers of data. Now, we will have a look at the types of key we have in RDBMS. So, we have primary key, we have super key, candidate key, alternate key, composite key, and foreign key. So, totally we have six types of key. Now, we will have a look at each of them differently and particularly and we will take an example for each of them and we will try to understand how this key concepts are working under this RDBMS. Now, we will start with primary key. So, basically primary key concepts is a technique to classify unique tables or rows in a table, right? So, give me an example with that. So, if you go for using a primary key, it will help you to make sure that your data, whatever data you have in your database, they are quietly unique, right? So, for the each and every primary key, that should be unique values, right? If you are going to give a repeated value for a primary key, that is not going to take your data. So, that time it is making sure that whatever the data you have in your dataset, they are unique. There is no repeated data in your dataset and that's how it makes your dataset more constructed and more structured, right? Now, let's have a look at some of the keys of primary key. So, primary key does not contain null value. Value should be unique. And primary keys are not always to be single attribute or column. And it can be also be a set of more than one attributes or column. But what are the attributes we stand for? So, if you look at the table, so basically attributes are the column name. So, whenever you are trying to work with DBMS, there are some words or some keywords will come into your way which maybe make you confused. So, I will try to cover all those things as well. So, when it comes to attributes, these attributes are basically your column name, right? Now, you can see in the table, we have three columns. We have basically three attributes. One is student ID, second one is name and third one is age. If you look at the table more closely, you can see the name can be repeated, right? Age can be same for two person. But what cannot be same for a two person? What can make your data more unique? So, in that case, student ID cannot be same. So, for that, we are going to take student ID as a primary key, right? So, whenever you are going to use primary key concepts for your data set, you have to make sure that which column making your data more unique, right? For the case of student ID, it's not

going to be same for any student. It's always be different for different students, right? So, this column can be your data. So, if you want to make primary key using two column or two attributes like maybe student ID and name, that can also be done using the two columns, right? So, these are the basic concepts. How can you use primary key for your table and make your data more constructed and more structured? Now, there comes the second concepts of super key. But what is super key? A super key is a set of one of more columns or attributes to uniquely classify rows in a table. But how we are going to do that? So, super key is a superset of candidate key. In the next part, we will have a look at what is candidate key, right? So, in the example, you can see that student ID and student name, okay? So, already we know student ID is something that uniquely you can use, right? So, that can be your primary key. But in the case of name and student ID, again, that will make sure your data is not redundant or your data is not duplicate. So, you can combine both of them, that student ID and name, and you can make a super key, okay? So, in the student table, student ID and name, the name of two student can be same. But their student ID cannot be same. So, this combination can also be a key of super key, right? Now, we will have a look at what is candidate key. So, candidate key is also a set of one or more columns or attributes to uniquely classify rows in a table. Now, you guys will like start thinking that what is the difference between super key and the candidate key, right? So, let me give you an example for that. The all remaining attributes or columns except for the primary key are considered as a candidate key. The candidate keys are also as strong as the primary key, right? So, if you look at the table, we have four attributes or four column names. So, in this case, we can say student ID will be unique. Passport number is also unique. And license number will also be unique, right? So, in that case, except student ID, your passport number and license number also be a part of candidate key, right? So, this is also unique. So, you can use them as a candidate key. Now, let's have a look at the next step. So, look at what's the difference between super key and candidate key. So, in the super key, all the candidate's keys are super keys. But when comes the candidate keys, the candidate's keys are taken from super keys, right? So, all the super keys cannot be a candidate key. But all the candidate keys should be super key. So, this is the catch over here. You need to understand. So, you can say all the super key are not candidate key. But any super key can be called as candidate key, right? So, combination of various super keys make the criteria to choose the candidate keys. Whereas, combination of various candidate keys make the criteria to choose the primary keys. So, these are the catch between super key and the candidate key. In a table, number of super keys are always greater than number of candidate's keys. And also in the table, number of super keys are always less than the number of candidate keys. Right? Now, let's have a look at the super keys attribute can contain null values. Whereas, candidate key's attributes can also contain null values. So, both of the cases, this is same. Right? Now, we will have a look at what is alternate key. So, alternate key is out of all the candidate keys, only one gets null. So, candidate key is selected for primary key. And raised keys are known as alternate or secondary keys. Right? So, let's have a look at what actually we want to mean by alternate key. So, you can see we have candidate key like student ID, passport number and license number. So, we take student ID as a primary key. But in the case of passport number and license number, this is also a candidate key. But this is also known as alternate key. Now, let's have a look at what is composite key. So, composite key consists of

greater than one attribute to uniquely classify rows or records and tuple in a table. Right? So, none of the column can perform as a primary key. So, the combination of key can be considered as a composite key. Right? So, these are the concepts we already know. Now, we will look at the last concept. What is foreign key? So, basically, in the real scenario, we know that. We use two key mostly. The two keys is primary key and foreign key. We also use the other keys as well. But for the most of the cases, we go for using primary key and foreign key. So, what is foreign key? Foreign keys are the columns of a table which refers to the primary key or another table. And basically, they act as a cross reference between tables. Right? So, whenever you are going to use a table relation. That time, we use foreign key. So, if you can look at, we have two tables over here. One is student table and second one is course table. So, in the student table, we have attributes like student ID, name, passport number and license number. But in the case of course table, we have student ID and course ID. Now, how we are going to relate them? Right? So, you can see one attribute is common over here. So, one attribute that is student ID is common in student table. And student ID is also common in course table. Right? So, we can say in the student table, student ID is the primary key that we all know. So, whenever it comes to the student ID in course table, that time we can say student ID is a foreign key. Right? So, basically, it is act as a cross reference between the tables. So, using student ID, you can connect this two table. So, this is how foreign key concepts works. We will understand this concept more deeply whenever we are going to use our demo part. That time, I will show you how can you actually use foreign key concepts to join two tables. Right? We will also look how can you make those relations between two tables. Now, we will have a look at what is normalization. So, I was saying that normalization is one of the most useful process and most wanted process whenever you are going to store your data in database management system. So, what is normalization? Normalization is the process of organizing data to avoid duplication and redundancy. It helps to minimize duplicate data. Also, it try to minimize or avoid data modifications issues. To simplify the queries also, we use normalization techniques. Now, let us have a look at what are the types of data. What types of normalization techniques we have? So, normalization rules are divided into few categories. So, let us have a look at what are the categories we have for normalization. We have first normal form which is known as 1NF in short. We also know as second normal form that is known as 2NF. We also have third normal form. And lastly, we have voice and code normal form that is DCNF. So, you already seen. What are the types of normalization we have or the categories we have. Now, we will have a look at how can you implement them. Right? And how can you use them in your databases as well. So, let us start with first normal form which is known as 1NF. So, each set of columns should have a unique value. It also helps to prevent using the multiple columns to face the same row. It should contain a primary key that identifies all the rows as a unique data. The primary key is usually a single column but if needed then more than one column can be combined to create a single primary key that we already know. Now, let us have a look at how can you use them in our databases. So, let us have an example of that. Suppose, we have a table called student age and course. Right? Now, you can see in the first table we have two courses under Raj. Right? So, basically from that we can say. Maybe Raj is enrolled for two courses that is CR001 and CR005. Right? But whenever we go for using our first NF or first normal form that time we are not allowed to have more than one value in one column. Right?

So, in that case you need to make two rows for Raj. So, first row will contain the first course he has enrolled for and the second rows you can see. It will contains the second course he had enrolled for. You can see your data or your table has changed after using the one NF concepts that it has two rows for Raj. Right? So, if you see the course. So, in that case your columns or your rows are basically increased because you are putting each and every data in different rows. Right? And here you can see after implementing the first normal form maybe the data redundancy increases but each row as a whole will be unique. So, this is the main like focus on one NF. Right? So, using first normal form basically help us to get the each row that will be whole unique row. So, your data will be unique. But maybe there will be some data redundancy are there and it will maybe sometimes increase the data redundancy. Now, let's have a look at what is two NF. Right? So, whenever you are going for using two NF rules for your database be sure your database is already in one NF. So, this is the first and foremost criteria whenever you are going to use second normal form to your data. You made a lot of tape! The recording was longer than 30 minutes. We hope our transcription made your workday more enjoyable. If it did, consider trying out our pro version on goodtape.io: – All your transcriptions* – Skip the queue = minimal waiting time – We will store your transcriptions (including this one) safely *) Well, up to 20 hours/month, which is kind of a lot. If you need to transcribe more let us know on yourfriends@goodtape.io