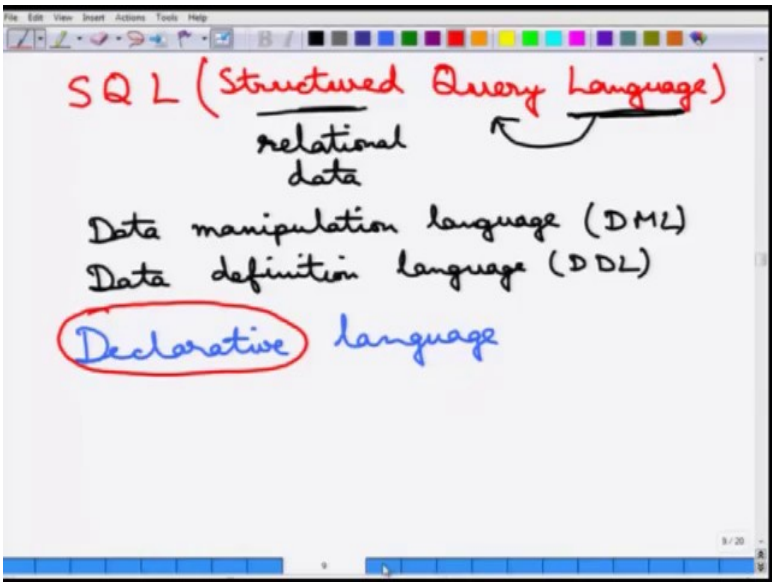
Fundamentals of Database Systems
Prof. Arnab Bhattacharya
Department of Computer Science and Engineering
Indian Institute of Technology, Kanpur

Lecture - 08
SQL: Introduction and Data Definition

So, let us start on SQL today.

(Refer Slide Time: 00:12)



So, **SQL** stands for **Structured Query Language**. So this is probably the most famous term for databases. Many people by database understands what SQL is. This means, probably a layman understands database as SQL. SQL is almost synonymous to databases and we will go over SQL in some depth. So start up with what is SQL? As you can see it is a language, fine; but it is a language to specify queries in a structured manner.

Now, for this what is a structured manner? This is a relational data model, so this has to be a relational data. So SQL is a language to specify queries in a relational database. So, that is the, what is about SQL. It is also what is called a data manipulation language (DML), because this lets one manipulate it or update values in a database, which is called a data manipulation language, as well as this is also a data definition language. (DDL)

So, it allows to define, how the data is structured in a relational database. So, when this is

both. SQL is based upon relational algebra as we already have said and SQL has many, many, many, many versions currently. So, when it was first incepted, after that there have been many, many versions; Many of them are non standard. So, different vendors which provide SQL and so MySQL, there are many other options and PostgreSQL, all those things they are… So, there are differences between them, there are some non standard operators that are provided by some of those vendors or systems, not by the others. What we will try to do is we will stick to the basic versions of the SQL as far as possible and one very, very important thing about SQL which is a big departure from languages such as C or Java is that, this is a *declarative language*.

Now, what does a *declarative language* mean? A declarative language means, this is not a procedural language, this is a declarative language. So, a declarative language essentially means is that, it let us you declare what you want to do. So, for example, in even a relational algebra, we just said select every tuple with the value of the attribute column A is 1, but it does not tell you how to select those. So, should you go over the tuples one by one, should you look at their attribute values or should you hash those attribute values, how it is being done that is not said.

Similarly, in SQL it is not specified how to do it, it is only specified what to do it. So, you just declare the intent, which is declared what the query is supposed to be doing, what it is supposed to be returning and not how it is being done. C, for example, you must specify each and every thing. So, suppose there is a table which is a relation and you are trying to manipulate it in C, so you must say, you go to the first element, go to the first attribute of it, check it with one, if yes you output it, if not you go to the second and so on and so forth.

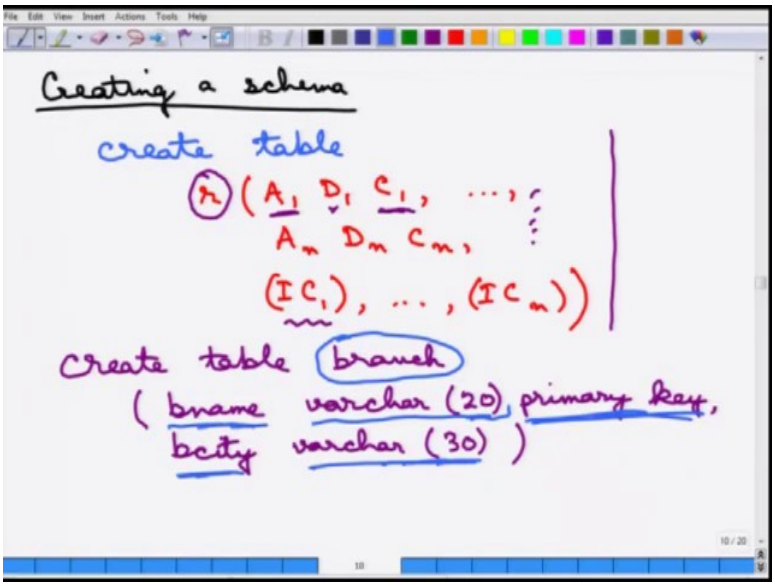
So, it must be specified in very minute detail, in SQL it is not done. So, a question comes is, then how it is being actually performed, so the database engine is very powerful actually. So, when you write an SQL query, it first parses it; then it figures out it has got it is internal algorithms and it tries to select an algorithm which will be the best for that particular query. So, then it applies that algorithm, finds the answer and returns it.

So, now, what is the, so this is a very nice feature of SQL, this is an advantage of SQL. Now this has got con side as well. The disadvantage is that, even if you know that a particular algorithm is not good or if you want to apply a particular algorithm for a query, the database engine does not let you do so, it will take over. So, you cannot say I want the answer of this

query in this manner; you are completely at the mercy of the database engine.

Having said that, in most cases the database engine is quite smart and there are lots of research that has gone into it and going into it every day and they are, they do a very, very good job in general. Let us start off with some examples of SQL.

(Refer Slide Time: 05:00)



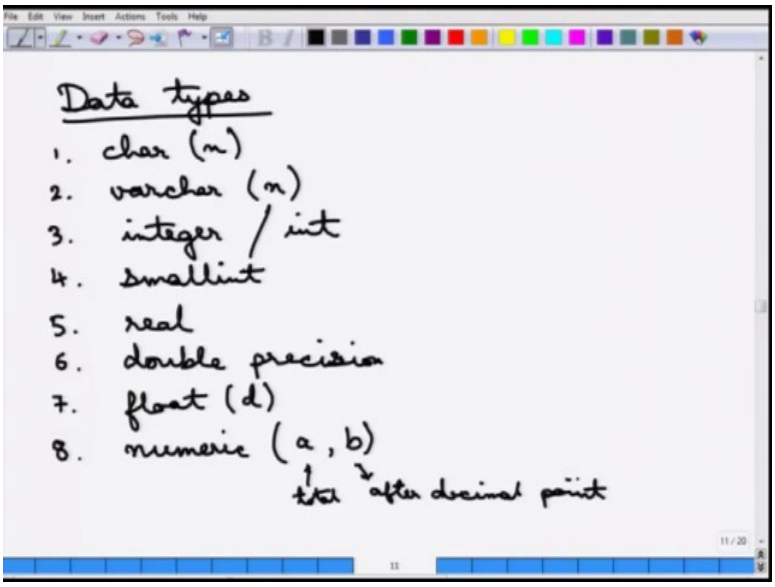
And, so the first thing is how to create a relational schema. So, how do you create, creating a schema? So, the way to create is that, there is a CREATE TABLE construct. So, the create table you say *create table*, then the specify the parameters in this manner A_1 , D_1 , C_1 then some A_n , D_n , C_n , then some constraints. So, let me go over this, what does it mean. So, this construct says that create a table with the name r; the first attribute of which is called A_1 whose domain is D_1 and the constraint, if there is any C_1 on that particular attribute and it let us you create n such.

So, you can create whatever number of attributes that you want as part of this *r*, in the end you can also specify certain integrity constraints, some other kinds of constraints if there are any on each of these attributes. So, you can say there is an integrity constraint 1 on this, there is an integrity constraint 2 on this etcetera, etcetera, etcetera. So, this is the way to create a table.

For example, if we just take an example from the earlier banking example that we were doing, what does it mean, it means create a table branch, the first attribute of which is called a

branch name, this is of character up to is can go up to 20 length. And this is the primary key of this table, then you create another attribute which is called branch city which is again a character which can go, I mean which is a string whose length can go up to 30 and that is it, that is about create table. So, now, let us discuss a little bit about the data types that you have.

(Refer Slide Time: 07:00)



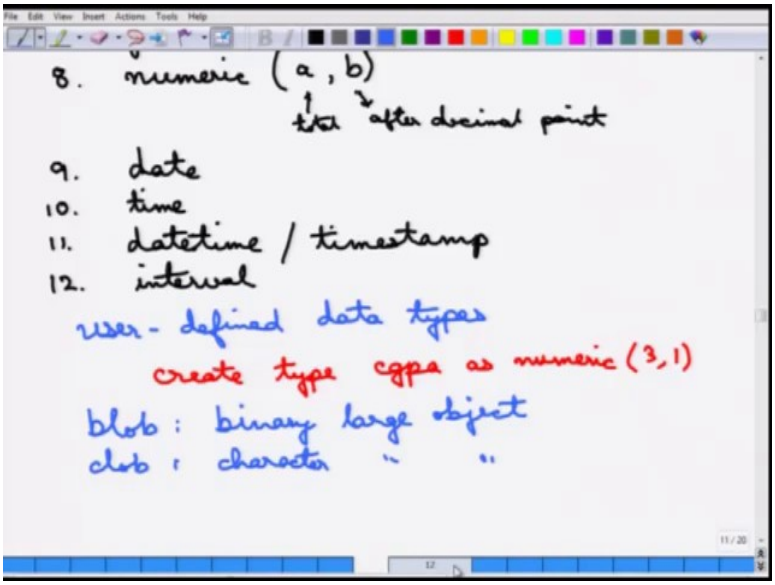
So, the data types in SQL, so the data type is we already saw one example. So, the first is a character n, this is a string whose length is exactly equal to n. As opposed to our varchar, so this is actually you can read it as variable character n. So, this is a string whose length can go up to n, not necessarily exactly equal to n, but it can go up to n. Then, you can say it is an integer or sometimes you can also specify it as an int, so simple this is an integer, then you can specify a small integer.

So, this is a small integer, just like this is a short in C, it says an integer, but it takes lesser amount of space to store. You can say it is a real, so it is a real number, you can say it is a double precision real, so you can say it is a double precision. So, essentially it means it is a real number, but with a higher precision; you can say it is a float with some precision d. So, the float will have at least d digits and finally, you can say it is a numeric.

So, this is a floating point number with a total of a digits of which b is after the decimal point, so this is the total and this is after decimal point. So, these are this and it actually little painful to go over all the syntax and all the data types of SQL. A good standard text book or the SQL manual will be probably a better way to look at this. So, these are the basic data types, but it

does not end here, there are many other data types in the newer versions of SQL, which makes it much more powerful.

(Refer Slide Time: 08:58)

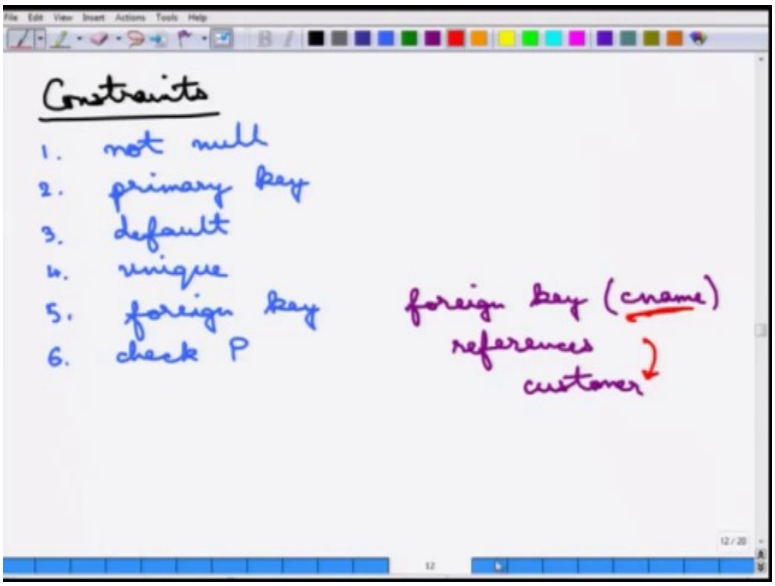


So, the first one is date, so it is specifies the date; now this is important because a date can be in the month, year, day format, etcetera, it lets you compare the two dates, take the difference and all of these. Similarly there is a time, then there is a date time which is a combination of these two, so which is actually sometimes called time stamp, so on and so forth. Then, there is you can also specify the time in terms of interval, so this is the time interval.

So, all these are different things that enhances the power of the data types, in addition SQL also lets one create user defined data types. So, one can create their own data types. For example, you can say create type; let us say cgpa as numeric, so it is just like a type definition. So, you are saying well, so every cgpa must be of three digits, at most three digits, so it is one digit after that. This is just a type name for this create, but more importantly it also for storing large objects, it also let us create something called a blob.

So, this stands for **b**inary **l**arge **ob**ject. So, for example, if you want to store an image etcetera in an SQL table, you can store it as a blob and similarly there is something called a clob, which is character large object. So, it store a document etcetera, it can be stored as a character. So, essentially these are not stored inside the table, they are pointed to these objects that are stored, so that is about the data types. Now, let us go over a little bit about the constraints that can be done in SQL.

(Refer Slide Time: 11:01)

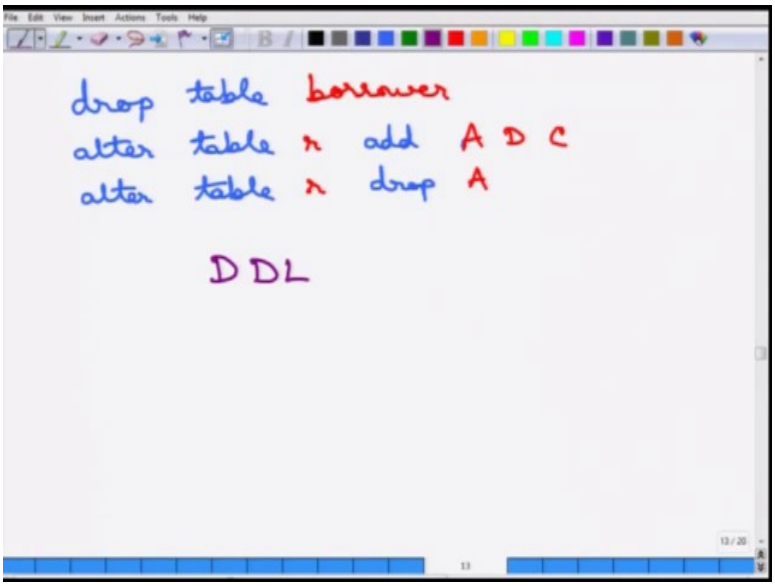


So, you can specify constraints saying something can be set as not null, so a particular attribute can be set as not null. So, whenever a new tuple is inserted, the attribute cannot be null. Similarly, we have already seen an example of primary key, we can specify what the primary key is, we can specify the default value. So, if nothing is specified, the default value takes over, we can specify whether it is unique or not.

So, when you say primary key of course, it is unique, but not the other way around, we can say you must make it unique. The fifth is probably very important, the fifth is important, it is called a foreign key. So, you can specify that this attribute is a foreign key to some other attribute and then, you can also say check p. So, this is like a domain constraint, so you check if the predicate p is satisfied or not.

So, something what we can do is, an example of may be the foreign key constraint, because this is important, is that you can say foreign key for a particular thing, such as let us say cname in some table, you can say this references the customer table. So, if such a definition is inside a table; that means, the attribute cname in this table references the primary key. It references the customer table which means of course, it references the primary key of the customer table, so this can be specified. Question comes is table can be created, can it be modified or can it be deleted of course, it can be deleted and the deletion is simply drop table.

(Refer Slide Time: 12:47)



So, you can simply say **drop table** whatever borrower say, let us say you can drop a particular table. You can also update a table. The syntax is **alter table** and essentially what you can do is, you can say alter table r; then you can say for example, add. So, I want to add the attribute A with domain constraint D and constraint C to this table that can be done. We can also say alter table r, drop a particular attribute A from this and again lots of other things can be done etcetera, etcetera, but it is the way to check up the manual.

So, this is about the part of the data definition language that we start over; how the table structure is defined. It is essentially just defining the table structure and not doing, not defining about any data etcetera. So, we will see about the insertion, deletion etcetera into the data later, but let us first go to why SQL is famous for, SQL is famous for this basic query structure.