Carbon Accounting and Sustainable Designs in Product Lifecycle Management

Prof. Deepu Philip
Department of Management Sciences

Dr. Amandeep Singh Oberoi
Imagineering Laboratory

Dr. Prabal Pratap Singh
Department of Management Sciences

Indian Institute of Technology, Kanpur

Week 11

Lecture 50

Database Normalization (Part-1)

Hello, everyone. Welcome back to the course on Carbon Accounting and Sustainable Designs in Product Lifecycle Management. I am Dr. Prabal Pratap Singh, and we are co-teaching this course with Professor Deepu Philip and Dr. Amandeep Singh. Till now we have talked about various kinds of carbon accounting databases and then we dive deeper into the database management systems.

Outline

→ MariaDB → RDBMS
→ Install MariaDB
→ Database Normalization
→ Queries Database Server
Normalize an unstructured Database Table.

Carbon Accounting Databases
DBMS
Database design process
ER diagram
Translation of ER to schema
SQL →

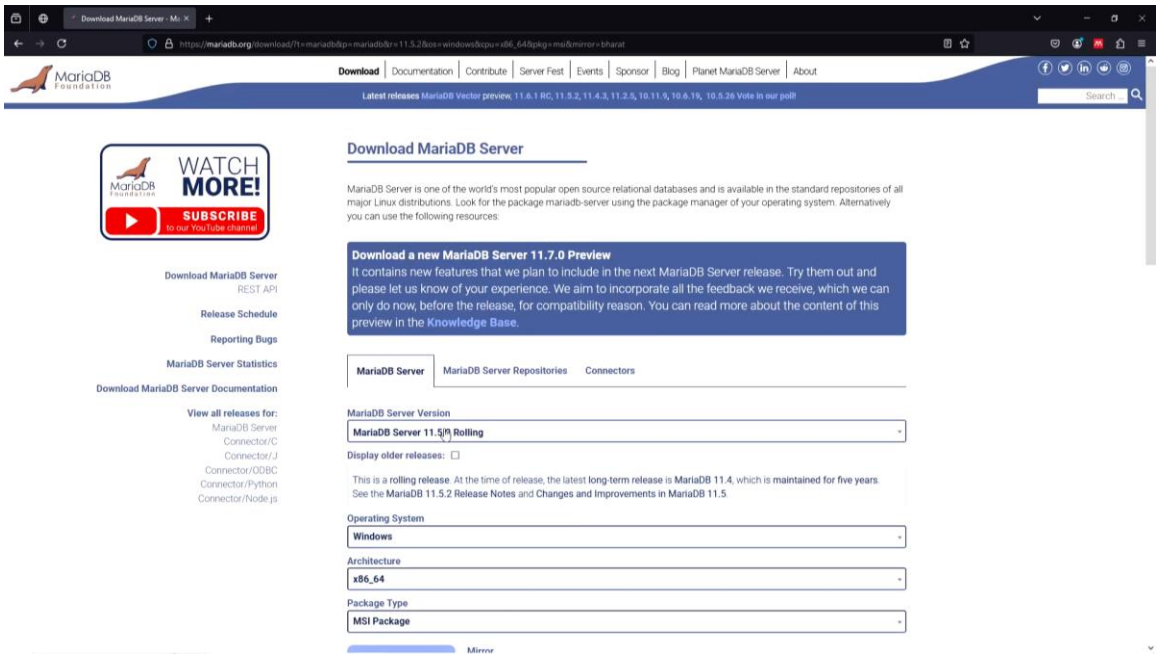
So, till now we have covered carbon accounting databases. We have talked about database management systems. We saw what is the actual database design process. So this process will be useful to design your own carbon accounting database. Then we talked about entity relationship diagram, which is a high level design of the complete database. And in the last video, we talked about how to translate these ER diagrams, translation of ER diagrams to schema of the database.

We have also looked upon these structured query language which is useful for communicating with the different kinds of databases. Since we are using rdbms, so we have looked upon what are the different kinds of queries that we can run on our database servers. So for today the outline is mainly is that we will start by first installing and first knowing about what is a MariaDB database, which is a kind of RDBMS, Relational Database Management System. Then we will install MariaDB on our machines. Since most of us are using Windows, so we will be installing Windows on Windows only but the process of installing on Mac OS and Linux is a bit different but usually same for both Linux and Mac OS.

After that we will talk about what are database normalizations why it is useful and how they make our database tables efficient. And then at the end, we will try to write queries in a freshly installed database server. And we will try to see how we can normalize, how to normalize, an unstructured or unnormalized unstructured database table. So this is the agenda for today.

MariaDB

→ Open-source RDBMS
→ Fork of MySQL
→ Developed in 2009 → after the acquisition of MySQL by Oracle
→ Adds several performance improvements and security enhancements.
→ Licensed under open-source licenses.
→ Supports parallel queries, thread pooling
→ Provides user management and data-at-rest encryption
→ Allows handling of semi-structured data
→ Perfect for dynamic websites
→ Supports data warehousing
→ Can handle large user loads and real-time data



So let us start with MariaDB. So MariaDB is nothing but an open source RDBMS, further it is a fork of MySQL.

So this was the original database server that was heavily used by the community. But then it was taken up by the Oracle. So this open source MariaDB, RDBMS is a fork of, is

just a copy of MySQL before getting it acquired by Oracle. So this MariaDB was developed in 2009. And why it was developed?

After the acquisition of MySQL by Oracle. So Oracle is an organization, well-known organization. And this MariaDB adds several performance improvements. And security enhancements it is licensed under open source licenses. So it is free to use also it supports parallel queries that is multiple users and multiple queries can be run together and also supports thread pooling.

So these features are useful to have a performant database server. And it also provides user management and encryption so data at rest encryption. It allows handling of semi-structured data. It is perfect for dynamic websites. It also supports data warehousing so web applications like wordpress. They use it heavily also it can handle large user loads and real time data.

So if you remember our previous discussion of carbon accounting databases that are prevalent and what are the needs while creating a carbon accounting database. You may remember that few of the features that are available or most of the features that are available in MariaDB are useful for creating a carbon accounting database. So let me highlight those features like the handling of real-time data, handling of large user loads, right. Or the availability of management of users or this parallel queries like multiple organizations are using the same carbon accounting database. So these kinds of features these they are available and that's why we can use MariaDB as our database server for creating our own relational database management system right.

So let us now switch to install our MariaDB server on our machines. So let us see how we can do that. So first of all, let's open your browser and do a web search by writing MariaDB. So this website with the web address mariadb.org, this is the actual website of MariaDB. You can directly click on download MariaDB server.

And something like this will open on your systems as well. And here it is asking that what is the version of the server that you want to install. So currently it is highlighting MariaDB server 11.5.2 rolling. So rolling are the cutting edge solutions that are available by this organization. If we are a beginner, then we should not use these kinds of versions.

So we can use MariaDB server 11.4.3. All other versions are in the build stages, right. This is the stable version. Now you can further choose on which operating system you

want to install. So if you want to install on Linux you can download it or if you want to install it from source then also you have the possibility of downloading it.

Further if you are on a Mac OS then you can use a package manager and can directly install this server. So, since we are using a Windows 10, we can install this, keep the architecture as same. This is a 64-bit version and use the MSI package if you are okay and just click on download. It will start downloading. So our file has successfully downloaded.

It's about 74 MB right now. And we can just click on it to open it. And something like this will open up. This is the installation window of the MariaDB server. So clicking on next, you can accept these terms.

Press the next button and then you can see that all these things will get installed. Just click on next again and here it is asking for a root password. So this is the main password of the MariaDB database server that you will need to provide every time when you want to open it. Let us just write it a very simple password ROOT123. Again provide the same password ROOT123.

And if you want to access this server from a different machine than this by connecting it to this machine then you can utilize this check option. So that you can use the root user from that remote machine. However, it is usually not prescribed that you should use root user from different machines. So let us skip this checkbox and you can click this use UTF-8 as default servers character set. And just click on next.

Now it is asking you that should we install this server as a service on this machine. So you can take this and also just this is the default TCP port. That is how this MariaDB server will connect to other applications if you want to. So we will in the upcoming lectures, we will see how we can connect our database and retrieve and modify the information inside the database using other programming languages like Python, JavaScript or something like that. So, while connecting with other programming languages, you need to use this port.

So, and the default port is 3306. So, just keep it as a default right now and just click on next. Now install it. So your server is installed and you can now access it. But before doing that, we need a console to access it.

So Windows provide different kinds of console like Command Prompt, PowerShell. So we will be using the Windows terminal that is available in Windows 10 and 11 by

default. If it is not available, then you can install it. You should edit your environment variables. So this is the Windows for system properties.

You can click on environment variables. And here is the user variables or system variables. So you can add at any of these. Let us just write in these user variable. So before that, where is the actual path of the MariaDB server that was installed?

So you can press Windows plus E to open the Explorer. Go to this PC. C drive where your actual windows installation is there and search for MariaDB this is the location and in this find the bin directory. Now you need to use this path C program files MariaDB 11.4 bin copy this path and open this window. Now Find the path variable and use edit option.

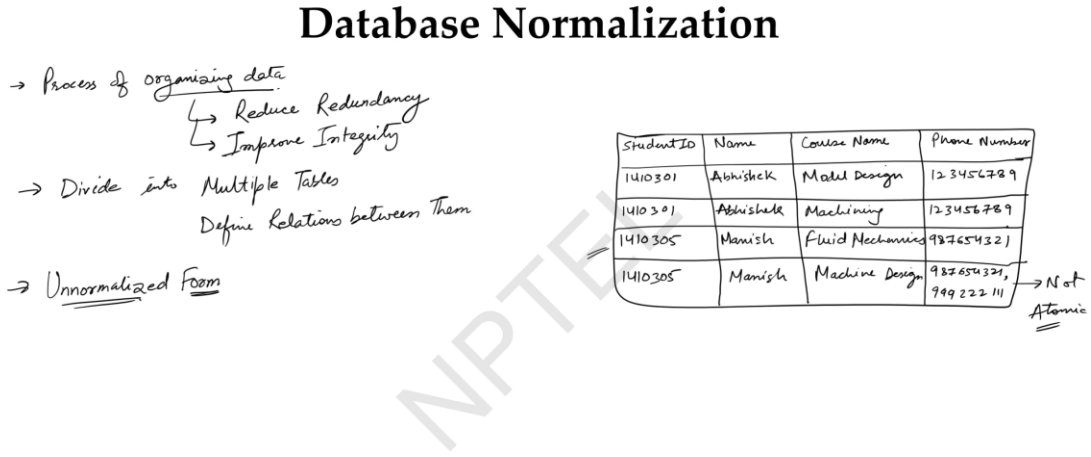
Now click on new and just paste the location of this installed MariaDB server. Click on OK. Again OK. And now you can use this MariaDB server from your terminal. So you can open a terminal window by searching for the terminal window and just click on open.

Now this is a terminal window. And you can write here the command mysql u root p. Now it is asking for the same password that you have provided while installing the MariaDB server. So it was ROOT123. So now we have entered into the MariaDB server using a Windows terminal and we have installed it on a Windows machine. The same steps are little bit different but same steps can be followed on your other machines like Linux or Mac OS, right.

So let us go back again to the topics and our installation is complete. So let us switch to the next topic which we want to cover and which we want to learn with the actual MariaDB server. So this is database normalization. So what happens is we learned how to create a database entity diagram for a database. Then we learned how to translate it to a database schema.

And from that, we also learned how to develop the database tables or database instances, right. Now the main thing is when you are trying to capture the real world during the requirement analysis phase. And you want to create a table in your database you can create a single table for all the data but it is not an efficient way to perform all the database table operations. So that is why we want to create efficient tables that is we need to understand why and how much database fields are required in a particular table. Can we reduce those fields, can we divide a single table into n number of tables, so that the

operations of a database management systems gets efficient. So, all of these decisions can be made through the database normalization task, right.




```
Windows PowerShell
MariaDB [(none)]> use db_normal;
Database changed
MariaDB [db_normal]> alter table student_unf rename column phoneNumber to phoeNumber;
Query OK, 0 rows affected (0.026 sec)
Records: 0  Duplicates: 0  Warnings: 0

MariaDB [db_normal]> desc student_unf;
+-------------+--------------+------+-----+---------+-------+
| Field       | Type         | Null | Key | Default | Extra |
+-------------+--------------+------+-----+---------+-------+
| studentId   | int(11)      | YES  |     | NULL    |       |
| studentName | varchar(100) | YES  |     | NULL    |       |
| courseName  | varchar(200) | YES  |     | NULL    |       |
| phoeNumber  | varchar(100) | YES  |     | NULL    |       |
+-------------+--------------+------+-----+---------+-------+
4 rows in set (0.030 sec)

MariaDB [db_normal]> alter table student_unf rename column phoeNumber to phoneNumber;
Query OK, 0 rows affected (0.025 sec)
Records: 0  Duplicates: 0  Warnings: 0

MariaDB [db_normal]> desc student_unf;
+-------------+--------------+------+-----+---------+-------+
| Field       | Type         | Null | Key | Default | Extra |
+-------------+--------------+------+-----+---------+-------+
| studentId   | int(11)      | YES  |     | NULL    |       |
| studentName | varchar(100) | YES  |     | NULL    |       |
| courseName  | varchar(200) | YES  |     | NULL    |       |
| phoneNumber | varchar(100) | YES  |     | NULL    |       |
+-------------+--------------+------+-----+---------+-------+
4 rows in set (0.031 sec)

MariaDB [db_normal]> insert into student_unf(
    -> studentId, studentName, courseName, phoneNumber)
    -> values( 1410301, 'Abhishek', 'Model Design', '123456789'
```

So, let us understand how to do all of these database normalizations. So, database normalization is a process of organizing data. Why we are organizing the data? To reduce or to improve integrity.

So, we can divide a single table into divide into multiple tables and further define relations between them. So the most basic table that anybody can draw off by getting the requirement analysis is the unnormalized form, right. So let us consider a very simple database of students. So we can create a table like this. So, these are the fields or attributes of our table.

And let us say the first entry or the first tuple has a student ID with 1410301. The name is Abhishek. Course under which this student is enrolled is model design. And let us say the phone number is 12345. Similarly, let us just fill this table.

Again, it is Abhishek. Another course he is doing is machining. And the mobile numbers remain the same. The next student is 1410305. That is Manish.

He is doing a course on fluid mechanics. His mobile number is 987654321. Again it is about Manish. And he is also doing a machine design course now while providing this detail he provided two numbers. So the first remains the same 987 654321 and there is one more number let's say 999222111 okay.

So this is the most basic form of table where all the data is present in a single table. So you can see there are different kinds of repetitions like there are repetitions in student ID, the names and the column of the phone number is containing more than two phones. So let us start using our MariaDB server that we installed and try to see how to create this kind of table into the server. So let us switch to our server right. Now it is showing that there is nothing so none is being highlighted.

So you can use simple command show databases with a semicolon and it will show that these four are the default databases that were installed while installing your MariaDB server. Now you can create a database so let's say create database. Now you have to provide the name of the database. So let it be DB normal. So it is showing that query okay one row affected.

That means your query has been successfully executed by the MariaDB server. Now again you can write show databases. Or instead of writing it again if you know that you have written a command and you have executed it before. So you can just use the arrow keys on your keyboard and use the upper arrow to get into the history of the commands you have wrote. So you can see that you have just written a create database db normal.

You can go one more up and you can see that this is the same command that you want to run it again. So just press enter. Now you can see instead of 4 now it is showing 5 rows

that means db normal is also a database. Which has been created by you but this kind of arrow key commands will only be available in the same while until and unless you do an exit command all the history will remain intact otherwise the history will get removed okay. So now you know that your database has been created you have to switch to that database.

Because currently MariaDB is showing that there are no active databases because it is showing none. So write use DB_normal semicolon. So it is showing that database has been changed. You should also see that I am using the lowercase letters. While discussing in the previous lectures, I was using uppercase letters as well.

So right now, the configuration of database server that you have installed changes. Has case insensitivity that means you can use any type of case but you can change that behavior by changing the configuration of the database right. So now you have a database in a database you can have multiple tables so let's see if you want to check how many tables are currently there you should see that there are no tables. Because you have just created this database but you can confirm this by writing show tables. So it is an empty set.

That means there are no tables. Now you need to create a table. So you have seen this is the table that you want to create, right. This one, so we first need to create the database relations. That means the attributes of the relations.

And after that we need to fill this data. So first let us create the schema of the database inside the MariaDB server. So you should write create create table name of the table is student UNF that means unnormalized form UNF and start providing the scheme of the table. So you can press enter this arrow is showing that the command you are writing is not yet complete because you have not provided a semicolon. So I am continuing my command otherwise it will take a long space horizontally, okay.

So the first field of the table was student id so you can write student id and the data type of the variable is int next student. Name data type of the variable is where care that means string the length of the string should be 100 characters maximum of 100 characters next is course name. It is also a string and let us just or let us put this as 200 course name could be long so the last attribute is phone number this let us just put it as a string as well right. Now so that we can have multiple phone numbers and currently it is violating the atomicity of the database. If we are using it as a multiple so this was my database schema and the query is again okay.

That means I should have a table in my database with the name DB normal. And the table name should be student UNF, unnormalized form. And it should have a student ID, student name, course name, and phone number. So let us see what is the describe. We can check the schema of a table by using describe function, describe statement student UNF.

So we made a mistake of creating the name of the field as a phone number. So we can alter this column name by using this command alter the alter what table which table student UNF. And then what we want to do we want to rename a column so rename a column and then we can rename the wrong value to the correct value that we want to fill. So, now if you describe this table you can see that student_UNF. This is the scheme of this table.

Now you have the fields of the table into your database. Now you want to fill the data in the table. So, the data, once again, this is the data, these four rows that we want to fill. So let us start writing the command for filling this data. So if you remember, we have discussed that insert into command is the command for filling the data.

So let us write the first insert into student_UNF. And the student id student name course name and phone number. So these are all the four fields, right. And now the next part of this command is what are the values that you want to fill in these fields? So you can write values.

The first is the student ID that is 1410301. Since it is a number, we are not using the single quotation marks. Then the next is a string. So you should use single quotation mark. Start with it.

Write the name Abhishek. Then the course name that is model design. And finally, the phone number of the student. And since you can see in the schema of the table that phone number is also a string. So we need to put this as a string.

We can write the phone number. Now just check it again that you are using an insert into command the name of the table is student_UNF. And fields of the student table is student id student name course name and phone number. And these are the four values corresponding to these fields so you can use a semicolon and just press enter. So this is how you have filled your table you can see what are the contents of your table.

So to see the schema you can write describe statement above and to see the content of the whole table you can use the select statement that we have discussed select star from where from the student table. So student_unf semicolon. So now you can see that you

have first defined the scheme of the table then you are start filling the tuples in the table. So this is the first tuple. Now use the same statement to again fill the next row.

So the next table also is about the student Abhishek. So the roll number remains the same name is same course name will be different it is machining and phone number is also same. So you can just press enter so now you should have two rows for the same student again use the statement insert into. Now fill the details for manish he is taking the course on fluid mechanics. And his phone number is one more thing you can provide multiple values as well in the same command of insert into.

So just use a comma instead of semicolon and press enter and again write the values that you want to fill, okay. So the next and the last row of this table is again about Manish. 1410305 name is Manish. Another course he is taking is about machine design. And now he provided two phone numbers here.

So now you should press the semicolon and so there are two rows and the your database management system is also showing that there are two records zero duplicates and zero warnings right. Now see the status of your table and the details in it so these are the four rows that we have seen here. And all the phone numbers course name and table so this is the unnormalized form of the table and this is not the correct form of creating a table because it is not efficient in a single table. There are multiple student ids multiple student names in a student name column and phone number is also storing composite values. So to make your tables more efficient there is a complete process of creating from going from an unnormalized form to a normalized form.

So there are different stages of normalized form one first normalized form second normalized form third normalized form. And there are more advanced normalized form as well like bc nf and all so we will study only up to the third normalized forms.

Thank you.