

Carbon Accounting and Sustainable Designs in Product Lifecycle Management

Prof. Deepu Philip
Department of Management Sciences

Prof. Amandeep Singh Oberoi
Imagineering Laboratory

Dr. Prabal Pratap Singh
Department of Management Sciences

Indian Institute of Technology, Kanpur

Week 12

Lecture 52

Carbon Accounting User Interface (Part-1)

Hello everyone, welcome 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 Prof. Deepu Philip and Dr. Amandeep Singh. Today we will talk about Carbon Accounting User Interfaces, but before that let us first revise what are the topics that we have majorly covered.

Carbon Accounting User Interface

Carbon Accounting & Sustainable Designs in Product Lifecycle Management

- Carbon Accounting Database
- DBMS
- Database design
- Database Schema
- Database Normalization

Dr. Prabal Pratap Singh
Indian Institute of Technology Kanpur

Initially, we covered carbon accounting databases, a brief introduction of what are the different types of carbon accounting databases that exist in India and around the world. Then we talked about what are the database management systems, what are their advantages and how they are being used for different kinds of purposes.

After that, we talked about the database design, the six different processes of designing a database. Then we also talked about the schema, how to develop the database schema while following the design process of the databases. So we talked about database schema. Then we talked about database normalizations. Now, today, we will try to develop user interfaces by incorporating the databases that we have developed, right.

Outline

- MariaDB and Python
- Historical emissions dataset
- Installing Python, Visual Studio Code.
- Connect Python and MariaDB.
- User interface development.

So, the outline for the today majorly covers the first we will talk about the connection between MariaDB and Python. Now, we have talked about MariaDB. We have seen in our previous lectures how to create new databases, show what is available in our currently available databases in MariaDB on Windows machines. After that, we will also try to talk about what is Python, what is this programming language. How we can utilize this Python to create different kinds of scripts so that we can connect it with our MariaDB databases.

And then transfer data to and fro between the user and the database. After that, we will discuss about a database that is historical emissions data set. So we will utilize this data set to develop our user interface and try to calculate various statistics based on the data. Then we will start developing our user interface by first installing Python, a code editor known as Visual Studio Code. And we will try to install various Python packages that we will utilize while developing our user interface.

Then we will try to develop a script to connect Python and MariaDB. After that, we will start developing our complete user interface for carbon accounting system.

MariaDB & Python

- Python is a general-purpose programming language.
 - Web development, Data Science, automation, etc.
- Developed in 1991.
- Executes code line-by-line (interpreted language)
- Python allows seamless integration with DBMS.
- provide dynamic content for web application.
 pip install mariadb

So let us start by introducing MariaDB and Python. So we all know about MariaDB based on our previous lectures. Now Python is a general purpose programming language.

So you can build different kinds of scripts of programs using this python. So its use cases could be web development most famous use case is in the applications of data science then you can also use it for automation. You can create different kinds of scripts that can be utilized to conduct various analysis behind the user that is automatically and various other tasks like game development etc. So this programming language was first developed in 1991. It was developed by Van Rosum and it executes code line by line so it is not a compiled language it is an interpreted language.

It is also a high-level language and we have also covered this basic introduction to Python in our previous courses with the name Advanced Business Decision Support Systems. So, you can utilize those resources to get to know more details about the Python programming language. We will not cover these details in this course and we will try to directly develop our carbon accounting system today. Also, Python allows seamless integration with various kinds of database management system. So, we have learned about different kinds of DBMS that are available and since today we are using MariaDB, we have a dedicated package named as MariaDB.

That is available in the Python Packaging Index (PyPi). And anybody can install by using pip install MariaDB. As a command to install this connector package that can provide different kinds of functions to connect to a MariaDB database server using Python scripts, right. So with these types of connections using different kinds of DBMS system, this Python scripts can provide dynamic content for web applications. And we will try to develop our web application today by developing our own user interface for the carbon accounting system.

Historical Emissions Data

- Climate Watch historical emissions data from 194 countries.
- Reports emissions between 1990 and 2021.
- Contains data about various sectors
 - Energy
 - Agriculture
 - Waste
 - Buildings, etc.
- Provides data on GHG gases like CH₄, N₂O, CO₂ and F-gases.
- Non-CO₂ gases are reported with CO₂-equivalent emissions using 100 year Global Warming Potential (GWP)

So now let us talk about the historical emissions data that we are trying to utilize today for developing our system. So this data set is sourced from a climate watch project. And

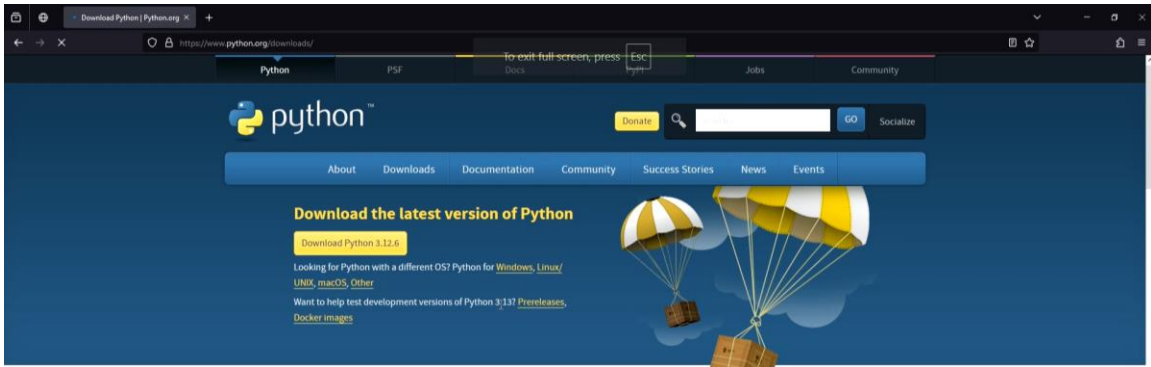
it provides historical emissions data from 194 countries, right. Also this data reports emissions between 1990 and 2021. It contains various sectors like,

it contains data about various sectors could be energy sector agriculture sector or waste buildings. And there are many more also it provides data on all major GHG gases. That is greenhouse gases GHG gases like methane N₂CO₂ carbon dioxide and other F gases. Also, whatever the numerical measures that this data set is reporting. If the gas is a non-carbon dioxide gas.

Then non-CO₂ gases are reported with CO₂ equivalent emissions using 100 year global warming potential or GWP. So, we have discussed all these terms in the previous weeks of this course and whatever the data that is available between 1919 and 2021 for different kinds of gases. They are based on the CO₂ equivalent emissions, right. We will see this data set and we have already pre-filled the data set in our MariaDB data for easy process of creating the user interface.

UI Development

- Install Python
- Install VS code.
- Connect the database with Python script.
- Demo of the complete historical emissions dataset ← MariaDB
- Packages
 1. Flask → lightweight web applications framework.
 2. MariaDB → allows access of MariaDB datasets using Python language.
 3. Pandas → Analysis and manipulation library.
 4. matplotlib → Comprehensive library for creating visualizations.



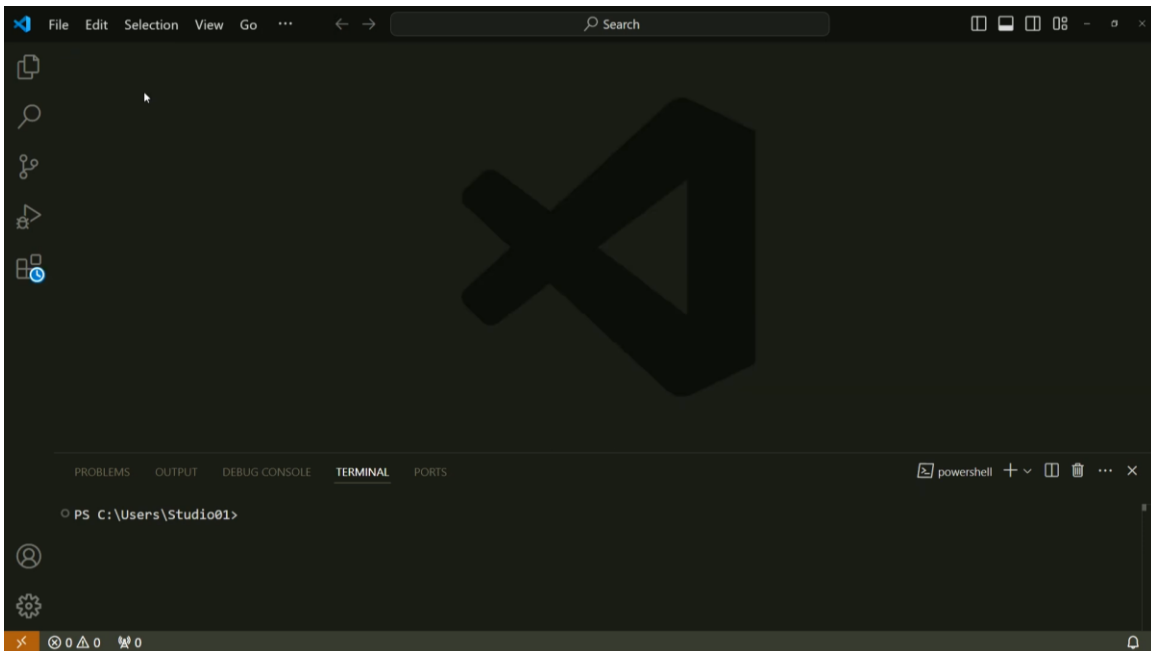
Active Python Releases

For more information visit the [Python Developer's Guide](#).

Python version	Maintenance status	First released	End of support	Release schedule
3.13	pre-release	2024-10-01 (planned)	2029-10	PEP 739
3.12	bugfix	2023-10-02	2028-10	PEP 693
3.11	security	2022-10-24	2027-10	PEP 684
3.10	security	2021-10-04	2026-10	PEP 679
3.9	security	2020-10-05	2025-10	PEP 596
3.8	security	2019-10-14	2024-10	PEP 569

<https://www.python.org/download/pre-releases/>

Looking for a specific release?



So now to develop a UI, we first need to install Python on our systems. Till now we have installed MariaDB on our machines. So it is a prerequisite or requirement for doing every task that we are doing today. And today we will install Python. Then we will install VS Code. And then first we will try to connect.

The database, any database that are available in our MariaDB server with Python script. Then we will try to get the demo of the complete historical animations data set. So this is available in our MariaDB server. We retrieved the data from internet sources and then we developed a MariaDB database and we will show you how this data is available in a table in the database. And for our actual UI development, we require the following packages today.

So we will also install these packages once our python gets installed on our machines. So the first package we require is flask is nothing but a lightweight web applications framework. So we will heavily utilize this to create our web pages. Then we will use MariaDB connector package named as MariaDB itself. And it allows, this package allows access of MariaDB data set using python language, then we will also utilize a famous library pandas it is a analysis and manipulation library.

At last we will utilize comprehensive visualization library known as Matplotlib. It is a comprehensive library for creating visualizations. So now let us switch to our machines and try to first install python then VS code. So to install python just write download python.

And download the latest version which is a stable release not the pre-releases like 3.13 so the current stable release is 3.12.6. So once it gets downloaded just open it and it will start its installer so once the installer starts try to click this checkbox, add Python.exe to path. So this variable we have also manually utilized for our MariaDB installation. If you switch back to our previous discussions, we added the path information of MariaDB manually. And this installer allows you to add Python.exe to path in this installer itself.

So just click on install now. Now our python is installed and we can check it by opening a terminal window and try to write python hyphen version. And you can see that it is showing that python 3.12.6 is installed now the next thing we need to install is visual studio code you can write vs code download. So now you will accept the agreement. Click on next.

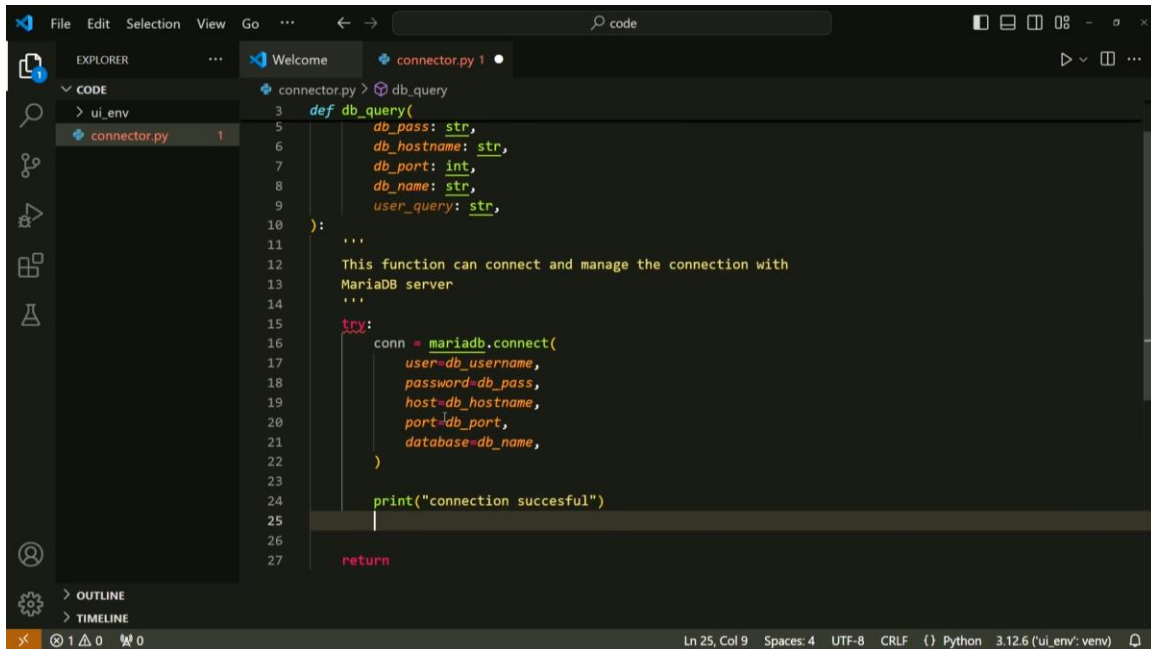
This is the default directory where it will install. And you can create a desktop icon and click on these checkboxes. Also make sure that you click on the add to path checkbox as well. So our code editor is also installed. Now let me give you a quick overview of the user interface of Visual Studio Code.

So these are all the file menus, different kinds of menus like file, edit, selection, view and this is our terminal that is open. We are currently in this directory and we will change this directory right now. Also these are the different kinds of options available for you like explorer will show you what are the different kinds of files that has been opened. So now we are in this directory where we will code today's user interface. Now, first of all, we need to create a environment for our packages.

So to do that, let us again check whether we have Python available in this terminal or not. So you can write Python version and Python is working fine. So let us now create our environment for today's session python hyphen. This is virtual environment. And create a name for your virtual environment.

So let's say today's session is uienv. So the name of our environment will be ui_env. It will take some time to create this environment. And now you can see in the code explorer window you have a folder named as uienv, right. And it has different kinds of folders and files so to activate this is this environment has been created.

But it is still not active so to create activate this environment. So now once your development environment gets activated, it will be shown as this. So the name of the environment has been highlighted in different color and now your environment is active. Now we need to install different kinds of packages which we talked about. So first start installing the MariaDB package. Install MariaDB also we need to install flask.



```
3 def db_query(  
4     db_pass: str,  
5     db_hostname: str,  
6     db_port: int,  
7     db_name: str,  
8     user_query: str,  
9 ):  
10  
11     '''  
12     This function can connect and manage the connection with  
13     MariaDB server  
14     '''  
15     try:  
16         conn = mariadb.connect(  
17             user=db_username,  
18             password=db_pass,  
19             host=db_hostname,  
20             port=db_port,  
21             database=db_name,  
22         )  
23  
24         print("connection succesful")  
25  
26  
27     return
```

So let us now create a new file name as connectors.py. Now we can hide our terminal for some time and we will start coding our connector script that will connect the MariaDB database. And fetch some data from the pre-existing data sets. So let us start by importing the MariaDB package that we just installed.

And we will try to create a function in the Python programming language by using the def keyword. So let us create the name of the function as db_query because this function can provide different kinds of queries and the dbms system will send back the data sets. So let us also create a doc string for this function this function can connect and manage the connections with maria db server. So whenever we try to connect with our MariaDB server using our terminal. We tend to provide some argument or some information to connect with the correct database.

So, these informations was like the name of the database. So, db_username or db_password. The hostname of the server of the database and the port of the database. So this port will be useful today which we defined while installing the MariaDB server the name of the database that we are trying to connect and the user query that the user wants to run while after creating this connection. So username will be a string.

So we can mark it as a string this is also string the password the host name will be a string as well the port will be an integer. So it will be int the name of the database is also

a string and finally the user query is again string. Now let us try to first open our database, MariaDB database, by running a Windows terminal. And we can write here `mysql-u`. The user is root and hyphen p. So the password was root123.

And you should change these passwords. Now show database self. So, we have different kinds of databases available and we created a DB normal database in our previous lectures. And this `cw_emissions` database is the data set which we will try to utilize today. And we have already prefilled this data set.

So, let us peek into this data set. So, use `cw_emissions` database. Now, let us first try to see how many tables are available in this data set. So, there is only one table named as `historical data`. So, we will try to see the schema of this database table.

So, we can write `describe historical_data`. So, there are different kinds of columns available like `country` which is a string and `sector` is also a string, `gas type` is also a string. And then these are double precision data sets with different numbers of years from 1990 to 2021. So, this is usually a unnormalized form of the data set and this is how we received the data set from the internet sources. So, we will try to utilize this data set.

Let us try to find out how many rows are available in this database that we have created. So we can write this query `select count star from` which the table name that is `historical data`. So we mistyped the name of the database table. So, there are 10,928 rows in this data set right and we can also try to see the first few rows. So, we can write `select star from historical_data fetch first row`.

Five rows only. So this is how your data set looks like the first column is `country`. So this is `AFG` or Afghanistan. Then we have a `sector`. So this is another value, `total excluding LUCF`.

And for the next row, it is `energy or industrial processes or agriculture`. And there are different kinds of gases like all GHG gases or `CO2`. So if you try to retrieve more rows we will try to see different kinds of data set available in each column now we need to connect this data set with our Python programming script. So now let us get back to this window again and to connect this Python script with the MariaDB package we need to first write a `try accept` statement. So what we need to try we need to try the sub function available in the MariaDB package, `mariadb.connect`.

So this function will try to connect it with the user's database available. So user is DB_username. And the next argument is password. It is db_pass host is equal to db_host name port is equal to db_port. And finally the database which is the db_name.

Now, this code will try to connect with the provided user argument for the database. We can store the output of this new variable name as connection. And we can try to print here that the connection is successful. You can also try to log this, but we can also use a print statement connection. So if this part of the code works, then this print statement will tell you that your connection with the MariaDB server is successful, right.

After that we will try to execute the query. So we can provide comments by using the pound symbol. So we can write execute the user query so query is saved in the user query argument here. So we can utilize it directly and we can use before executing the user query, we need to first create a cursor on the connection. So we can write connection.cursor.

And then we will execute the query by using cursor.query, user query. So whatever query the user will provide, this connection will execute it on the server. Now what if this connection does not happen? That is either MariaDB server is not available or the Python is not able to connect it to the currently running server. So we need to write our accept statement as well.

So this should contain accept. Maria DB.error as E and we can write print the errors. And whatever happens whether it can connect it or not you must run this command after to close this connection. So we can write if there is a connection that is this cursor variable was created by the script. So if this is available then we should close this connection cursor.close.

Otherwise else or we can write another if statement if only connection then connection close that is a successful connection was there. But we could not developed our cursor query so we should close our connection. So this is a function that will try to connect with the MariaDB server and will fetch your data from the server. So let us try to call this function with our arguments like the username and password and try to see whether it will print something or not. So the name of the function was db_query.

So we can call this function. And let us just copy these arguments quickly. So the username of our server is root. Then the password is root 123 hostname of the server is localhost and port was 3306. So if you try to remember the previous discussion while

installing the MariaDB server this is the default port that was created while installing it right.

Now the name of the database so it is cw_emissions and here we can provide a query. So let us just write select count star from historical data so let us open our terminal again. And we can run this script by using python connector.py okay. So we forgot a comma here so we can run this script again. So connection is successful but it is still not printing anything.

That is the user query ran successfully but there is nothing printing on the screen. So what we can do is once the query gets executed successfully we can write here. Let us say the cursor has a sub function known as fetch all and we can utilize it to get all the information that has been queried from the server into a new variable named as rows. And this row has multiple rows of whatever that we have retrieved so we can write for row in rows print row. So, now let us try to run this script again and you can see that we have mentioned in our query that we need to count all the total number of rows that are available in this historical data table that is present in the CW emissions data set.

So there are 10,928 rows that are available and we can confirm this since we have already executed this command directly into the MariaDB server and we have found the same number of rows. So if the user wants to change this query, so it can change the query here while calling this function and the output will be different. So let us now stop here for a quick pause. And we will try to start developing our user interface in the next upcoming lecture. And till now we have covered that how to create and connect the connection with our MariaDB server using Python programming language.

Thank you.