

# Project Documentation

# OurDB DataBase

---

Zeel sheladiya

Mihir Surati

Parth Dabheliya

## Introduction

Sr .no	Topic Name:	Page no
1	<u>Scope Of The Project</u>	3
2	<u>Overview Of Existing Systems and Technologies</u>	4
3	<u>Overview Of the Project</u>	5
4	<u>Process flow of Ourdb</u>	6
5	<u>Objectives of Ourdb</u>	6
6	<u>Feasibility</u>	7
7	<u>Consideration</u>	8
8	<u>Getting started</u>	9
9	<u>Syntax and Uses</u> (create database)	10
	<b>Topic name</b>	
	<b>Page no</b>	
	Select database	
	11	
	Create table	
	12	
	Current database	
	13	
	Rename database	
	14	
	Show database	
	15	
	Show table	
	16	
	Show column in table	
	17	
	Rename table in database	
	18	
	Rename column in table	
	19	
	Insert data into the table	
	20	
	Adding new column in to the table	
	21	
	Updating a data in to the table	
	22	
	Delete the database	
	23	
	Delete the table	
	24	
	Delete the particular table data	
	25	
	Delete the column	
	26	
	Select data from table	
	27	

10	<u>Implementation of database as console GUI</u>	28
11	<u>Encryption and Decryption</u>	31
12	<u>How actually data stored in table</u>	31
13	Future Enhancement	32

## Scope Of The Project



**Database Management** systems are widely used by companies and organizations to maintain and manage their knowledge and information resources. After completing a database management course, students might work in any number of industries including,

1. Automotive
2. Banking
3. Education
4. Legal
5. Insurance
6. Government
7. Pharmaceutical
8. Retail

## Overview Of The Existing Systems And Technologies

### Types of Database Management Systems

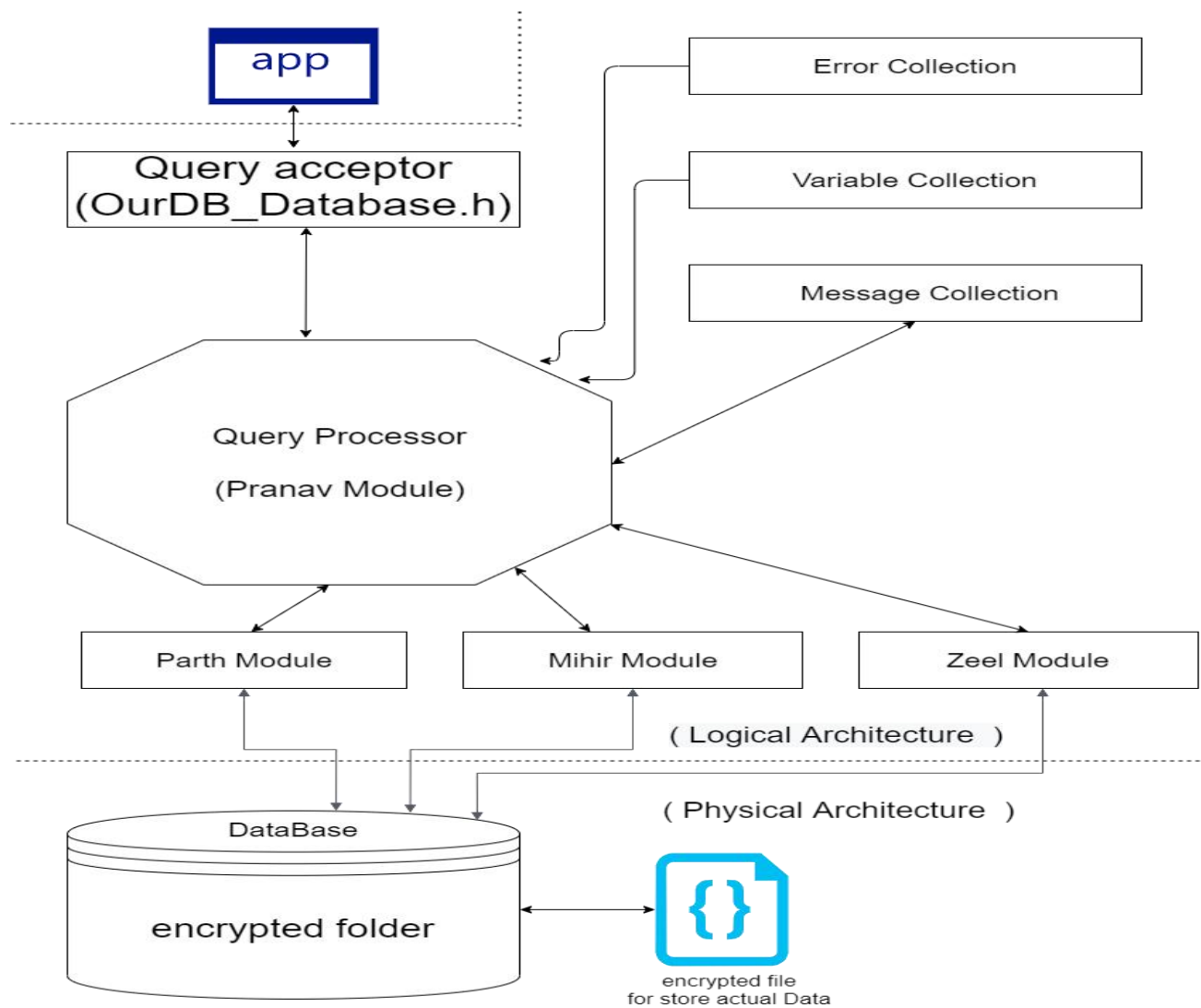
There are several types of database management systems. Here is a list of seven common database management systems:

1. Hierarchical databases
2. Network databases
3. Relational databases
4. Object-oriented databases
5. Graph databases
6. ER model databases
7. Document databases
8. NoSQL databases

### List Of The Top Database Management Software

SolarWinds Database Performance Analyzer , DbVisualizer , ManageEngine Applications Manager , Altibase , Oracle RDBMS , IBM DB2 , Microsoft SQL Server , SAP Sybase ASE , Teradata , ADABAS , MySQL , FileMaker , Microsoft Access , Informix , SQLite , PostgreSQL , AmazonRDS , MongoDB , Redis , CouchDB , Neo4j , OrientDB , Couchbase , phpMyAdmin , SQL Developer , MariaDB and many more.

## Overview Of The Project



Architecture Of OurDB

## Process Flow Of OurDB

In this Database management system , after connecting the database with your application. Whenever you use `run_query()` , through this function your query will be gone for process via `OurDB_Database.h` to `query_process.h` . In the `query processor` your query gets some operation on it and through these operations, it will be selected by **three main modules**. After the module operation if there is data for insertion or updation, it will write in encrypted form into the **physical storage area**. However , if there is data for viewing purpose then the module will perform operation on actual data and then present you in the form of string.

## Objectives Of The Project

- You don't need to install this database to use. It's make this database super portable.
- You can make your own query syntax very easily . it's make your query code super protected.
- It has all data files in encrypted format.
- You can connect this database with your application with just one line of code.
- This database works in both application console and GUI.
- Database import and export is super easy.
- You can make a shared database without any connection string or any extra connection.
- You can add your own module in the database for the query process.
- It's very lite in terms of size.
- To run this database you don't need any extra specification requirement in your system. So it can work smoothly in low specs machines.

# Feasibility Study

## 1. Financial Feasibility :

It is too light weight and it doesn't need any high end requirements to run. So it can be too cheap or very affordable for all kinds of developers.

## 2. Technical feasibility :

OurDB is a completely data operation based application. The main technologies and tools that are associated with OurDB are ,

- C++ Compiler

This technology is freely available and technical skills required are manageable. From These it's clear that the project OurDB is technically feasible.

## 3. Resource and Time Feasibility :

Resource that are required for the OurDB project includes,

1. C++ compiler
2. Space in physical disk to record the data
3. C and C++ supported environment





## Consideration

1. Performance
2. Security
3. Usability and ease of use
4. Capacity and Scalability
5. Availability
6. Maintainability



## GETTING STARTED

This will tell you about the OurDB database. And how to use it. Its so simple to use this database system you don't need to install whole database system like any other. But just add some database files not more than (under 1.5 MB) and need to install c++17 compiler (minimum requirement).

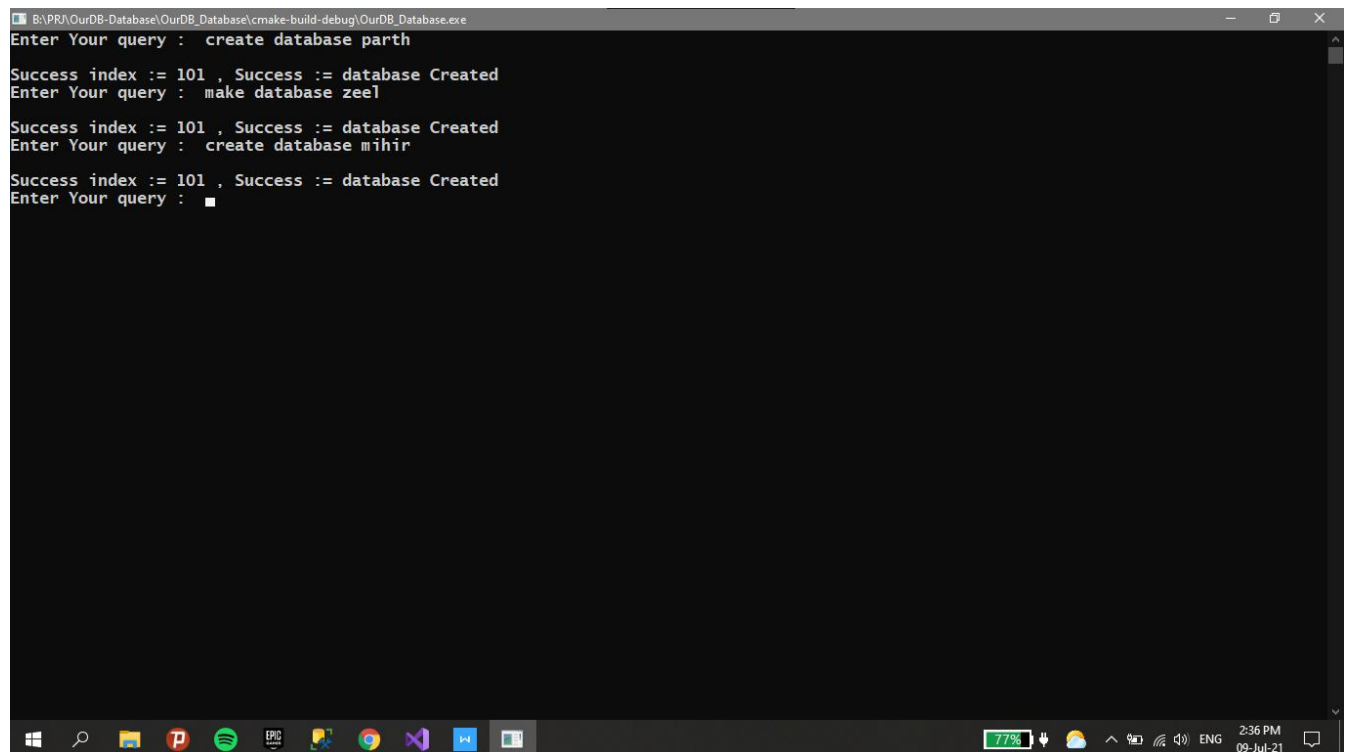
- We need to follow some step to get started with database..
  1. You have to include Ourdb\_Database.h header file to your application so that you can use database.
  2. Then you have to create a database.
  3. After creating database, you can create table and do all table related operation.
  4. Then you can insert into the table.
  5. Even you can perform update and delete operation with & and | operators.
- Additional functionality:
  - a) We have added functionality for the developer to convert data to json object.
  - b) Convert the data to stringTable (represent data in table format in console application)
  - c) Convert the data to map
- Developer can easily change the filter to convert to the above three format, by default it is string.

## Syntax and uses

- **Create database:**

First step to enter in the database system is to create database.  
A physical storage area where all table resides.

Syntax : **create/make database <database\_name>**



```
B:\PRJ\OurDB-Database\OurDB_Database\cmake-build-debug\OurDB_Database.exe
Enter Your query : create database parth
Success index := 101 , Success := database Created
Enter Your query : make database zeel
Success index := 101 , Success := database Created
Enter Your query : create database mihir
Success index := 101 , Success := database Created
Enter Your query : █
```

Above example shows that it is very simple to create a database..

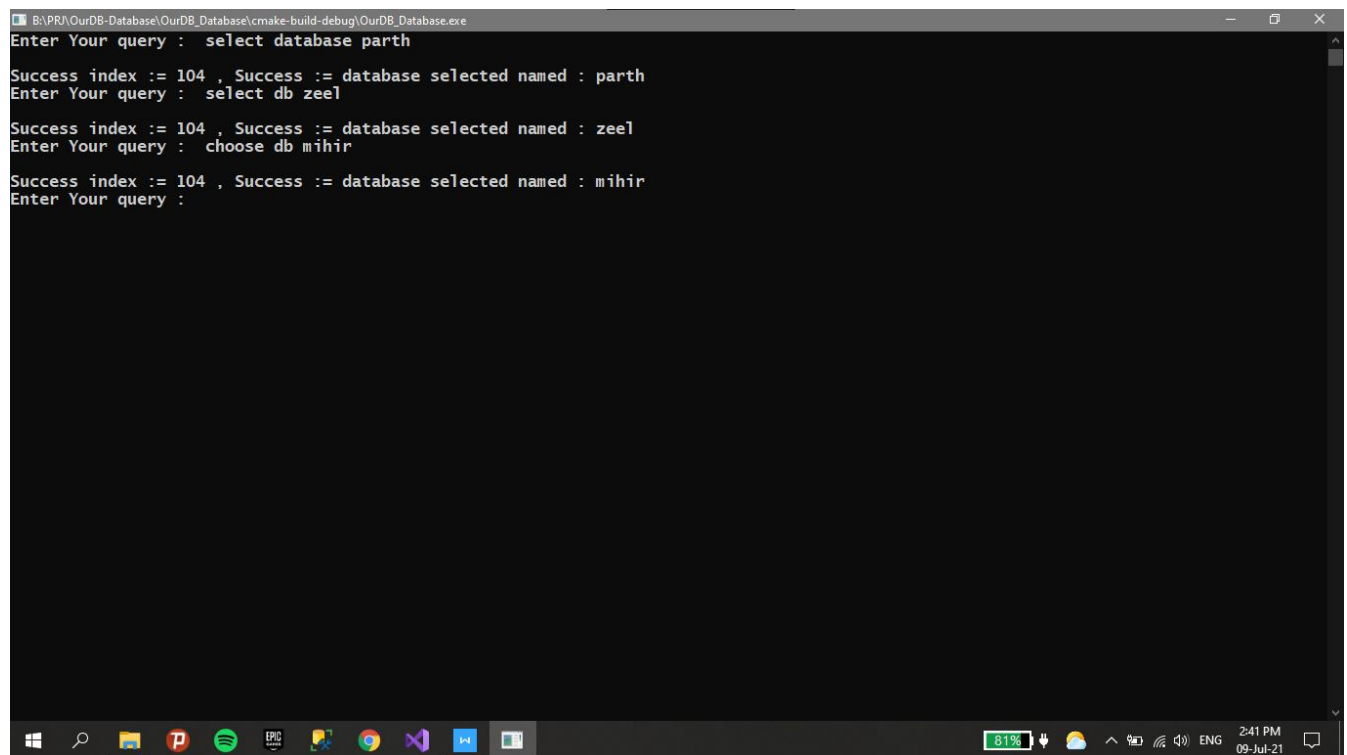
Even you can change the (create) keyword in the code(instruction for developer). its dynamic add or remove keyword etc.

- **Select database :**

select the database among the existing databases else returns the error.

**Syntax : select/choose database/db <database\_name>**

After creating database you have to compulsory select the database in order to create table and perform some action/operations.



```
B:\PRA\OurDB-Database\OurDB_Database\cmake-build-debug\OurDB_Database.exe
Enter Your query : select database parth
Success index := 104 , Success := database selected named : parth
Enter Your query : select db zeel
Success index := 104 , Success := database selected named : zeel
Enter Your query : choose db mihir
Success index := 104 , Success := database selected named : mihir
Enter Your query :
```

Note : database is so flexible that developer can change any keyword..

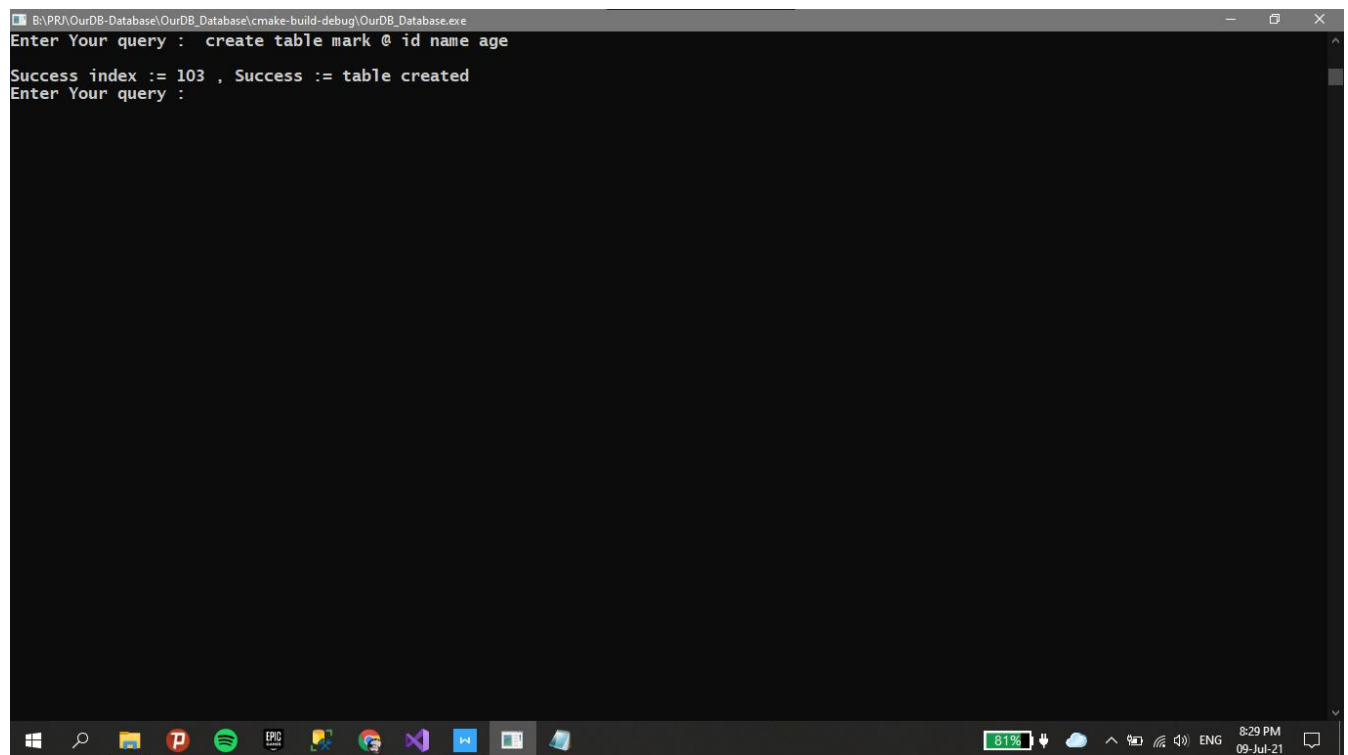
- **Create table :**

Creates table in the database along with the column in it.

**Keyword :** create or make

**Syntax :** `create/make table <table_name> @ <column1_name> <column2_name>...`

**NOTE :** After creating database you have to compulsory select the database in order to create table and perform some action/operations.



The screenshot shows a terminal window with the title bar "B:\PRJ\OurDB-Database\OurDB\_Database\cmake-build-debug\OurDB\_Database.exe". The terminal content is as follows:

```
Enter Your query : create table mark @ id name age
Success index := 103 , Success := table created
Enter Your query :
```

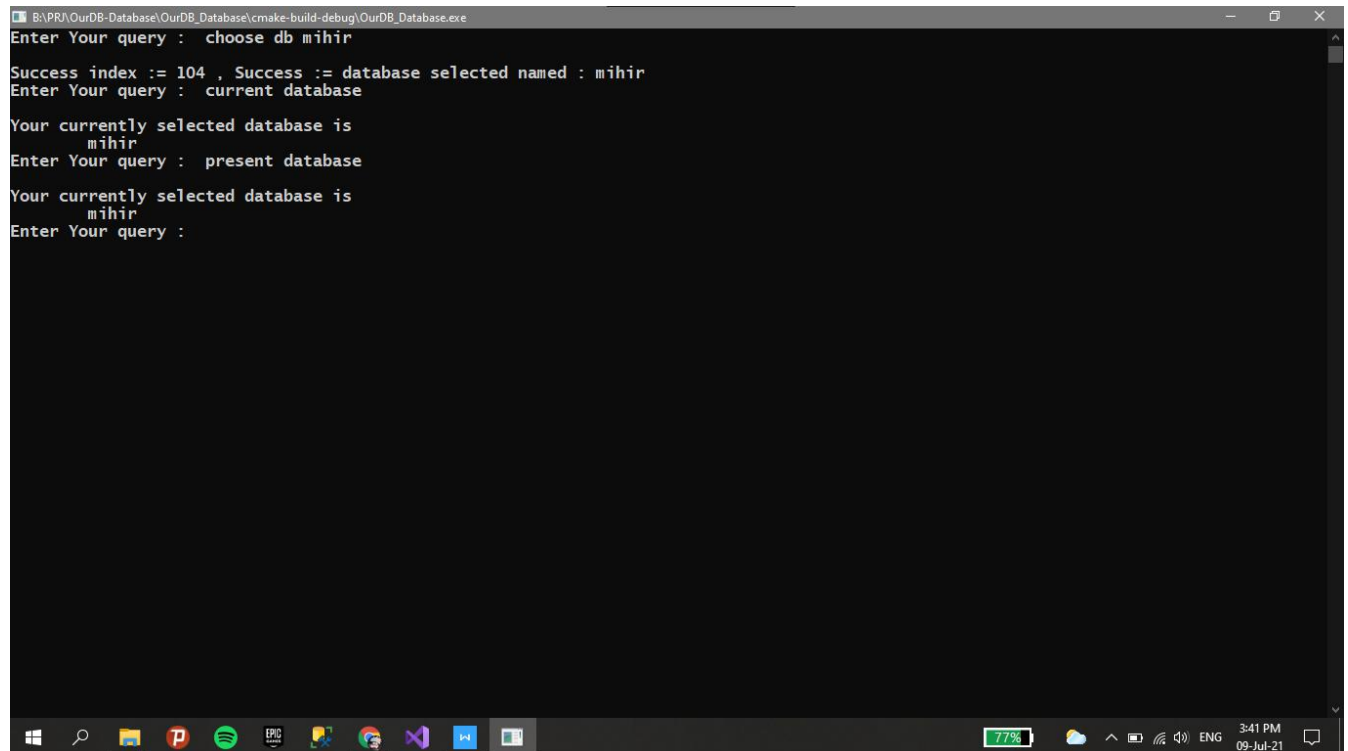
The terminal window is running on a Windows operating system, as evidenced by the taskbar at the bottom which includes icons for various applications and the system clock showing 8:29 PM on 09-Jul-21.

**NOTE :** with this simple syntax we can easily create the table in database.

- **Current database :**

it will show the selected database/the database you are in right now.

**Syntax : current/present database/db**



```
B:\PRA\OurDB-Database\OurDB_Database\cmake-build-debug\OurDB_Database.exe
Enter Your query : choose db mihir
Success index := 104 , Success := database selected named : mihir
Enter Your query : current database
Your currently selected database is
    mihir
Enter Your query : present database
Your currently selected database is
    mihir
Enter Your query :
```

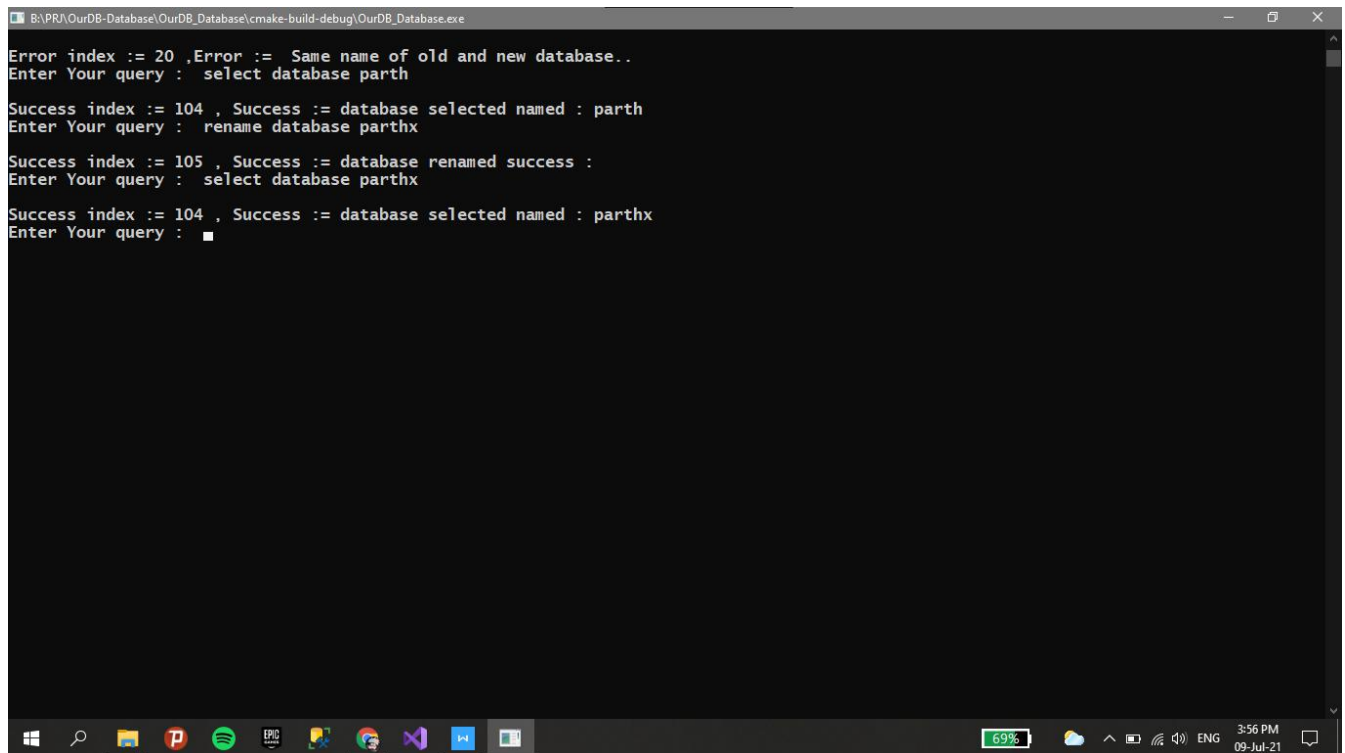
Note: this query used when you don't know what database you are in right now.

- **Rename database :**

It will rename the database in the system..

NOTE : first you have to select the database then and then you can perform rename query.and after renaming you have to again select the database to perform the operation.

**Syntax : rename database <database\_name>**

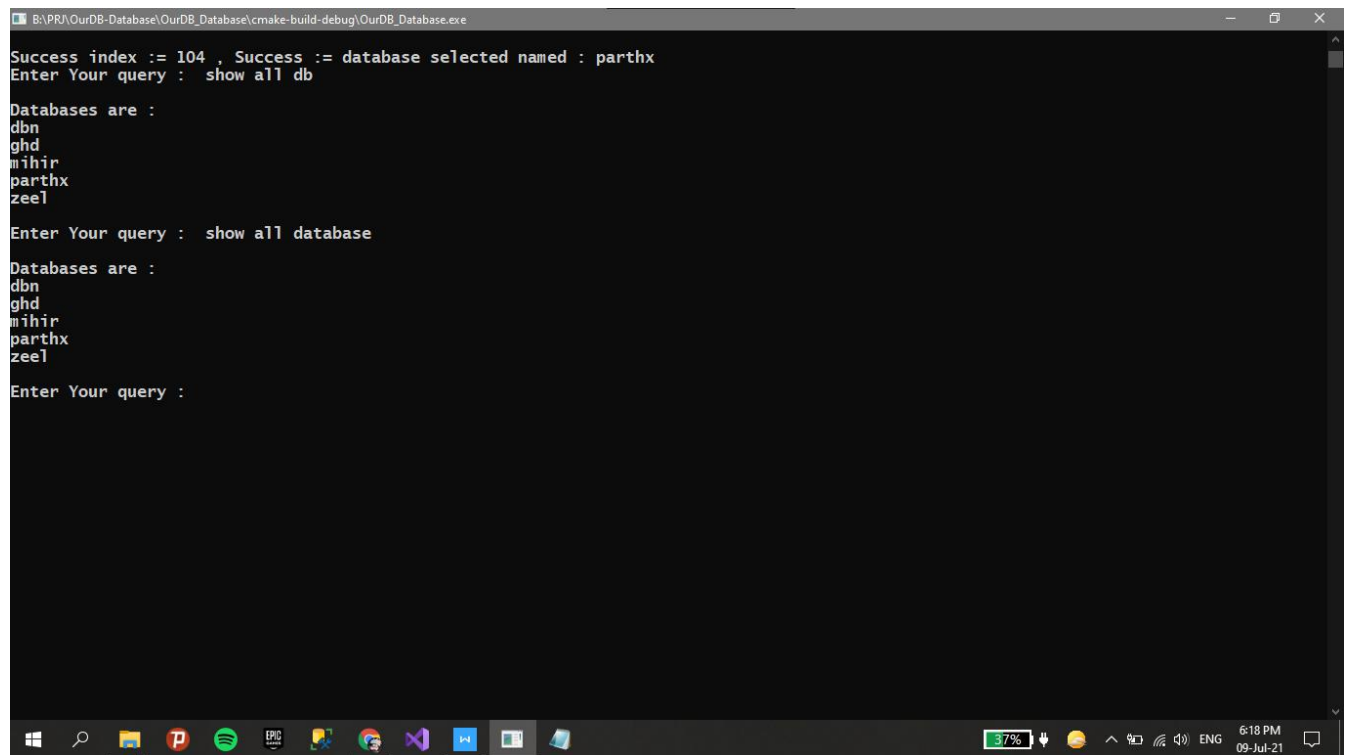
A screenshot of a Windows terminal window titled "B:\PRJ\OurDB-Database\OurDB-Database\cmake-build-debug\OurDB-Database.exe". The terminal shows a sequence of database operations. It starts with an error message: "Error index := 20 ,Error := Same name of old and new database..". Then, the user enters "select database parth", and the terminal responds with "Success index := 104 , Success := database selected named : parth". Next, the user enters "rename database parthx", and the terminal responds with "Success index := 105 , Success := database renamed success :". Then, the user enters "select database parthx", and the terminal responds with "Success index := 104 , Success := database selected named : parthx". Finally, the user enters a prompt character "■". The Windows taskbar is visible at the bottom, showing various application icons and the system clock indicating 3:56 PM on 09-Jul-21.

Note: we have given this facility who wants to change the name of the database but it is not good practice to do so..

- **Show databases :**

It will show all the database resides in the system.

**Syntax : show/display/view all db/database**



The screenshot shows a terminal window titled "B:\PRA\OurDB-Database\OurDB\_Database\cmake-build-debug\OurDB\_Database.exe". The terminal displays the following text:

```
Success index := 104 , Success := database selected named : parthx
Enter Your query : show all db

Databases are :
dbn
ghd
mihir
parthx
zeel

Enter Your query : show all database

Databases are :
dbn
ghd
mihir
parthx
zeel

Enter Your query :
```

The terminal window has a Windows taskbar at the bottom with various application icons and a system tray showing the time as 6:18 PM on 09-Jul-21.

NOTE : it is advisable to run query prior to any other query so that you can see all the database exists in the system

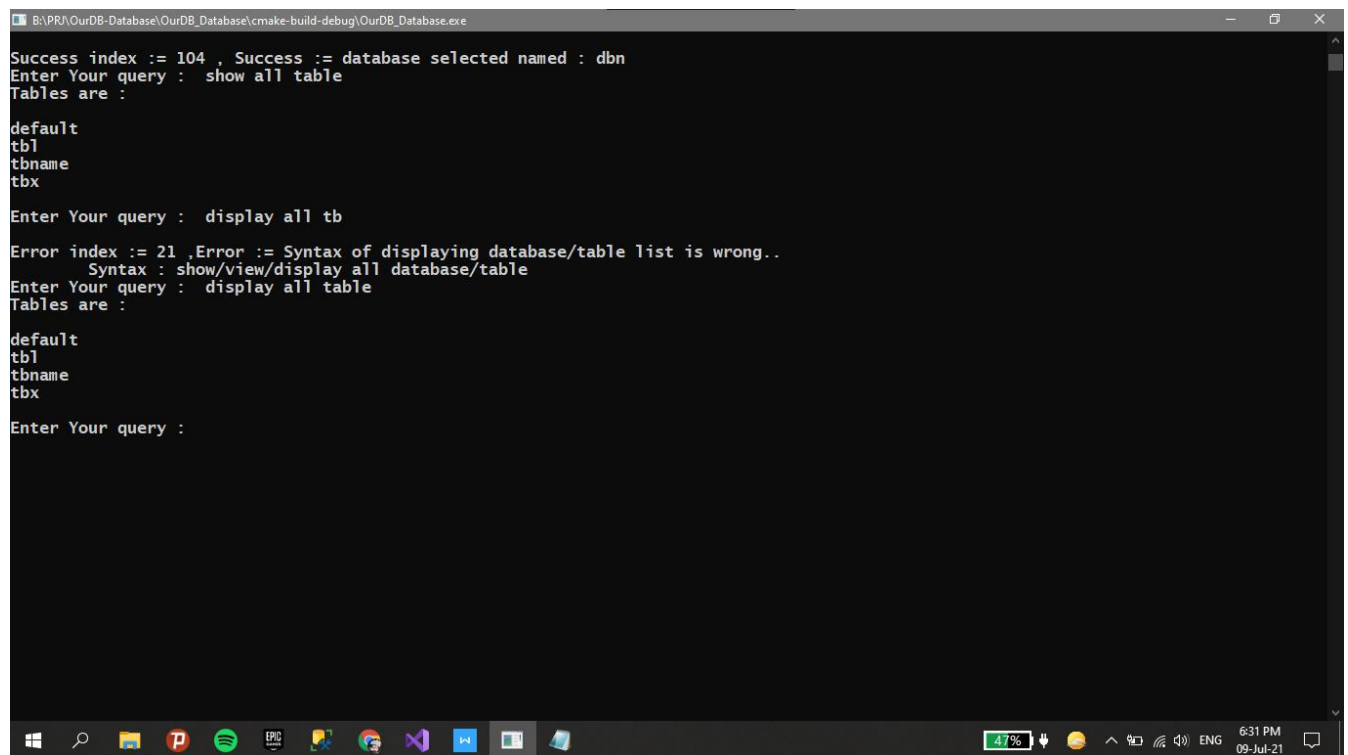


- **Show tables :**

It will show all the tables resides in the selected database.

Note : first we have to select the database so that we can see the table resides in it.

**Syntax : show/display/view all table**



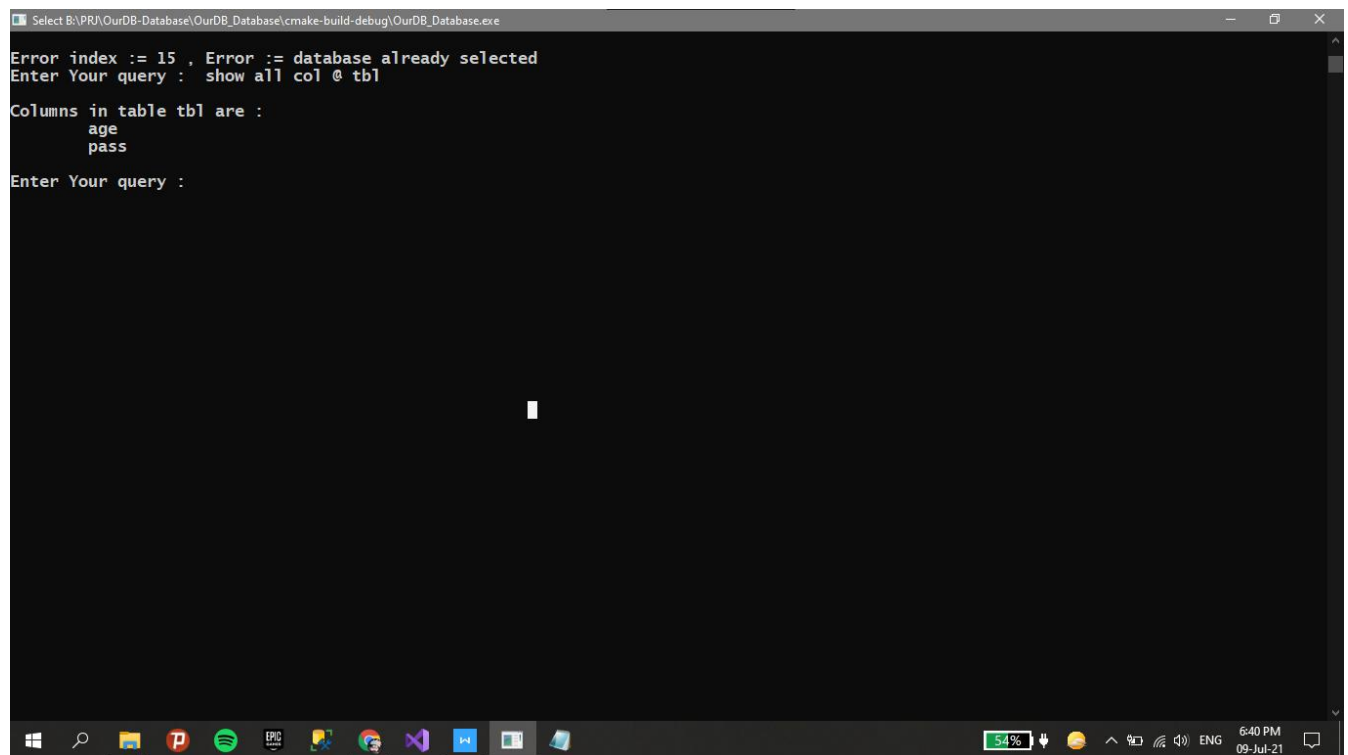
```
B:\PRA\OurDB-Database\OurDB_Database\cmake-build-debug\OurDB_Database.exe
Success index := 104 , Success := database selected named : dbn
Enter Your query : show all table
Tables are :
default
tbl
tbname
tbx
Enter Your query : display all tb
Error index := 21 ,Error := Syntax of displaying database/table list is wrong..
Syntax : show/view/display all database/table
Enter Your query : display all table
Tables are :
default
tbl
tbname
tbx
Enter Your query :
```

- **Show column in table :**

It will list out all the column which is present in the table

Note : After selecting database , create table and then you will be able to use this feature.

Syntax : `show all col @ <table_name>`

A screenshot of a terminal window titled "Select B:\PRJ\OurDB-Database\OurDB\_Database\cmake-build-debug\OurDB\_Database.exe". The terminal shows an error message: "Error index := 15 , Error := database already selected". Below this, it says "Enter Your query : show all col @ tbl". The output shows "Columns in table tbl are :" followed by a list of columns: "age" and "pass". The prompt "Enter Your query :" is shown again. The terminal window is overlaid on a Windows desktop with various icons in the taskbar and a system tray at the bottom right showing the time as 6:40 PM on 09-Jul-21.

```
Select B:\PRJ\OurDB-Database\OurDB_Database\cmake-build-debug\OurDB_Database.exe
Error index := 15 , Error := database already selected
Enter Your query : show all col @ tbl
Columns in table tbl are :
    age
    pass
Enter Your query :
```

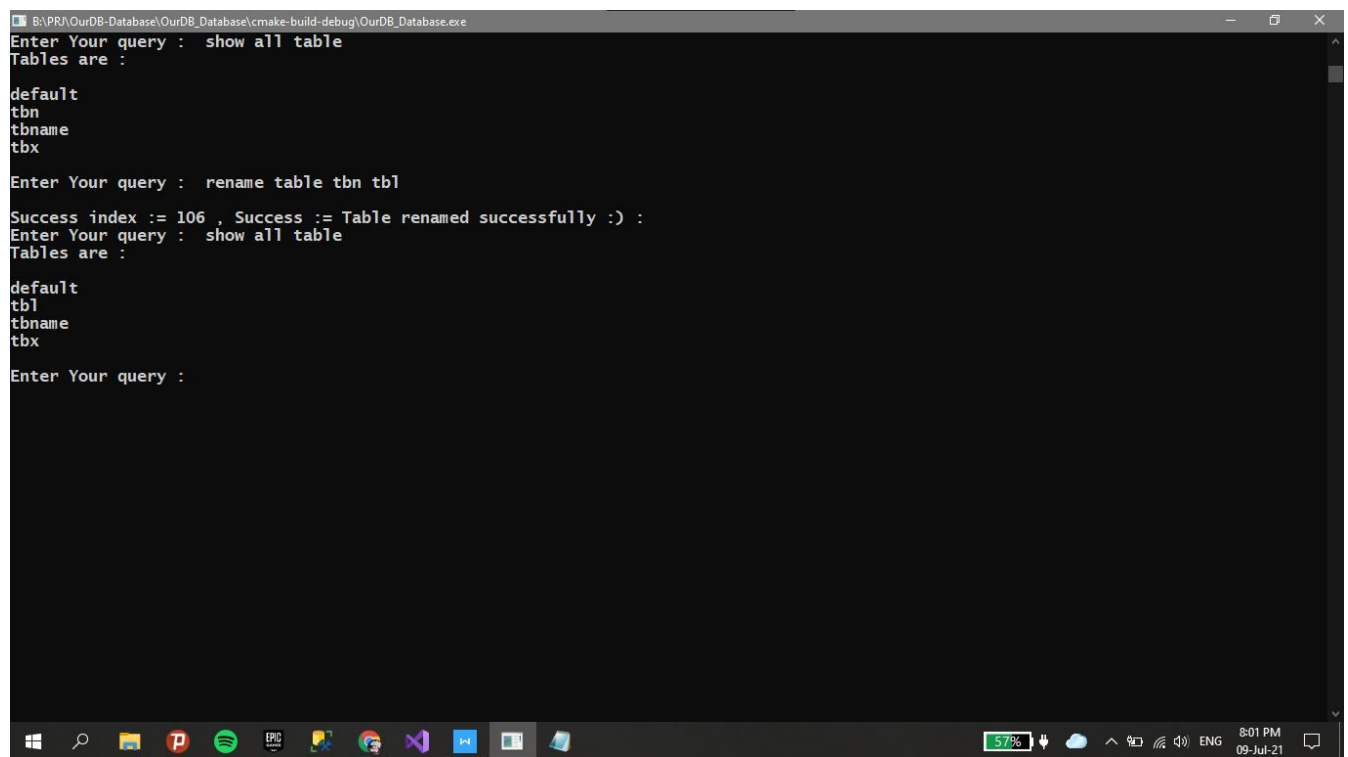
Note: show column in table never return null value because column in the table decide at the creation of table.

- **Rename table in database :**

This command will rename the table exist in the particular database.

Note : make sure that table must exist before renaming the table.

Syntax : **rename table <old\_table\_name> <new\_table\_name>**



```
B:\PRA\OurDB-Database\OurDB_Database\cmake-build-debug\OurDB_Database.exe
Enter Your query : show all table
Tables are :
default
tbn
tbname
tbx

Enter Your query : rename table tbn tbl

Success index := 106 , Success := Table renamed successfully :)
Enter Your query : show all table
Tables are :
default
tbl
tbname
tbx

Enter Your query :
```

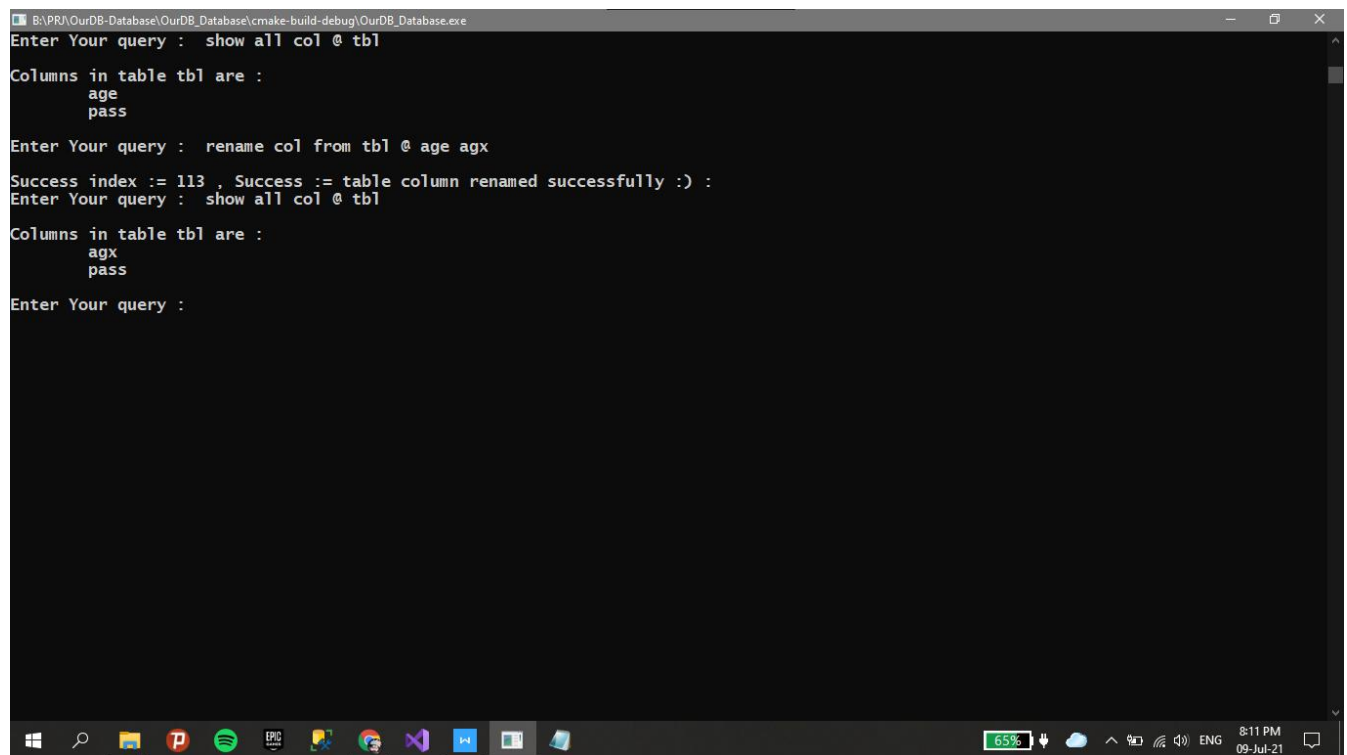
- **Rename column in table :**

This command will rename the column in the particular table

Note : make sure to check that column exist in the table by show all column @ table.

Keyword : column/col both will be accepted but not simultaneously.

**Syntax :** rename column/col from <table\_name> @ <old\_col\_name>  
<new\_col\_name>



```
B:\PRA\OurDB-Database\OurDB_Database\cmake-build-debug\OurDB_Database.exe
Enter Your query : show all col @ tbl
Columns in table tbl are :
    age
    pass

Enter Your query : rename col from tbl @ age agx
Success index := 113 , Success := table column renamed successfully :) :
Enter Your query : show all col @ tbl
Columns in table tbl are :
    agx
    pass

Enter Your query :
```

Note : this feature looks so simple but most important while working with the database .

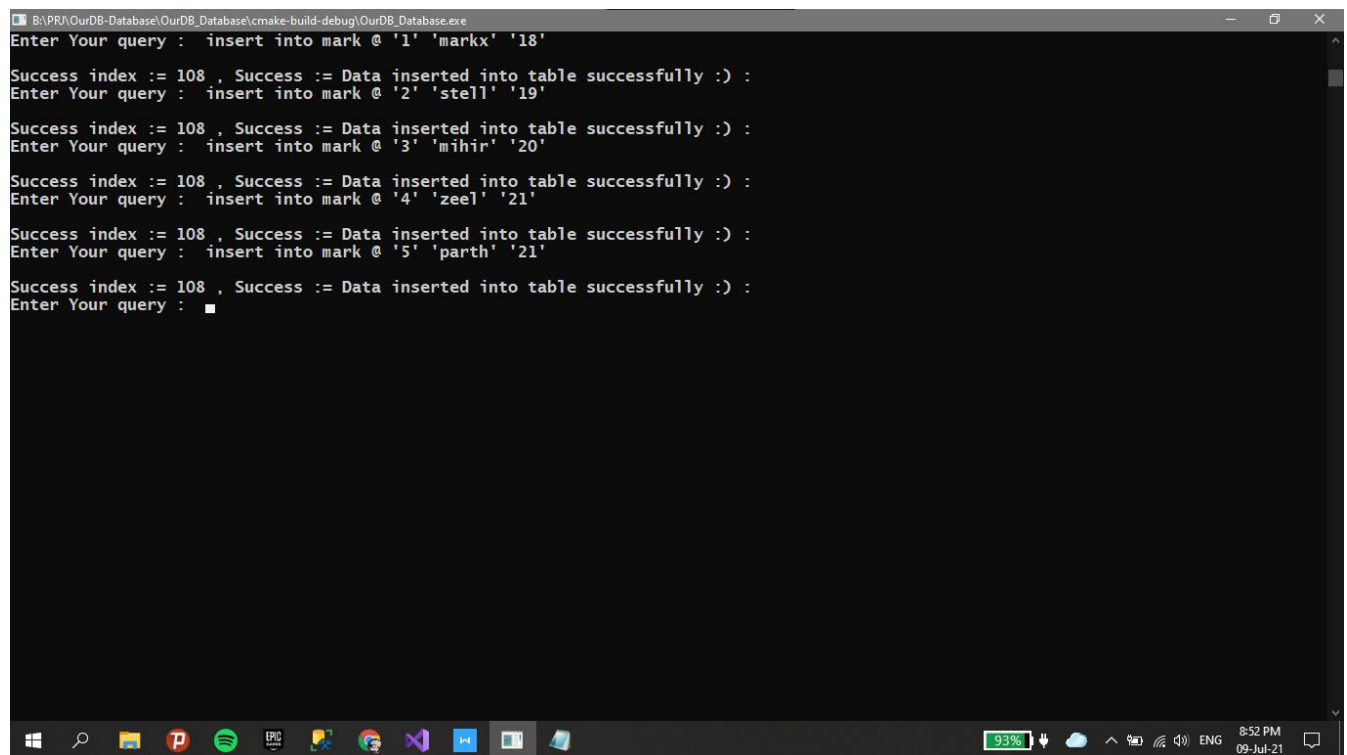
- Insert data in to the table :

Insert query is used to enter the data in the empty column which was created with the table during creation of table.

Note : make sure in which table you are going to insert the data ,that must be created before insertion of data.

Keyword :insert or add both will be accepted but not simultaneously.

**Syntax : insert into <table\_name> @ '<col1\_value>' '<col2\_value>' ...**



```
B:\PRA\OurDB-Database\OurDB_Database\cmake-build-debug\OurDB_Database.exe
Enter Your query : insert into mark @ '1' 'markx' '18'
Success index := 108 , Success := Data inserted into table successfully :) :
Enter Your query : insert into mark @ '2' 'stell' '19'
Success index := 108 , Success := Data inserted into table successfully :) :
Enter Your query : insert into mark @ '3' 'mihir' '20'
Success index := 108 , Success := Data inserted into table successfully :) :
Enter Your query : insert into mark @ '4' 'zeel' '21'
Success index := 108 , Success := Data inserted into table successfully :) :
Enter Your query : insert into mark @ '5' 'parth' '21'
Success index := 108 , Success := Data inserted into table successfully :) :
Enter Your query : █
```

Note : you can see the data in the table by select command we will look for it later on.

Single quote is necessary for inserting the value in to the table.

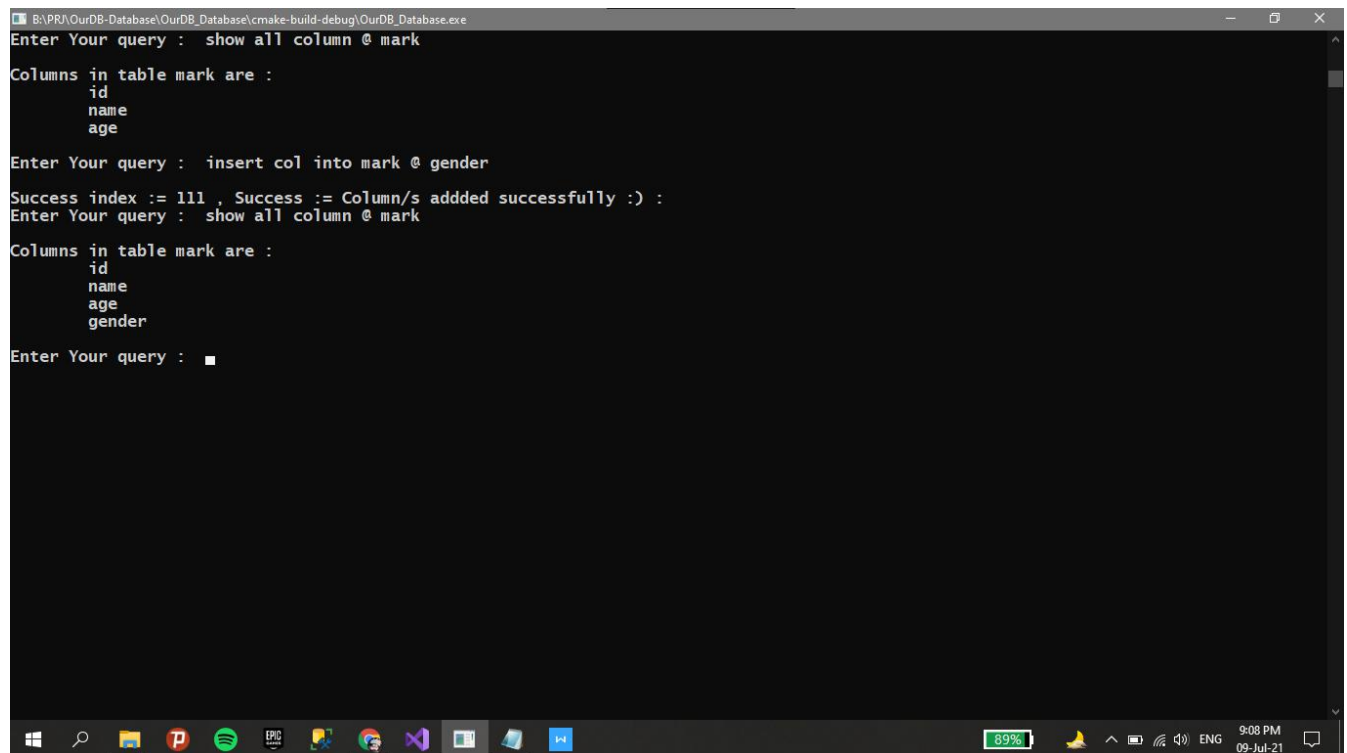
## ● Adding new column in the table :

This command will add a new column in the existing table.

Note : make sure in which table you are going to insert the column ,that must be created before insertion of column.

Keyword :insert or add both will be accepted but not simultaneously.

**Syntax : insert/add col/column into <table\_name> @ <col\_name>**



```
B:\PRA\OurDB-Database\OurDB_Database\cmake-build-debug\OurDB_Database.exe
Enter Your query : show all column @ mark
Columns in table mark are :
    id
    name
    age

Enter Your query : insert col into mark @ gender
Success index := 111 , Success := Column/s added successfully :) :
Enter Your query : show all column @ mark
Columns in table mark are :
    id
    name
    age
    gender

Enter Your query : █
```

- **Updating a data in to the table :**

It used to update a data in to the table ..

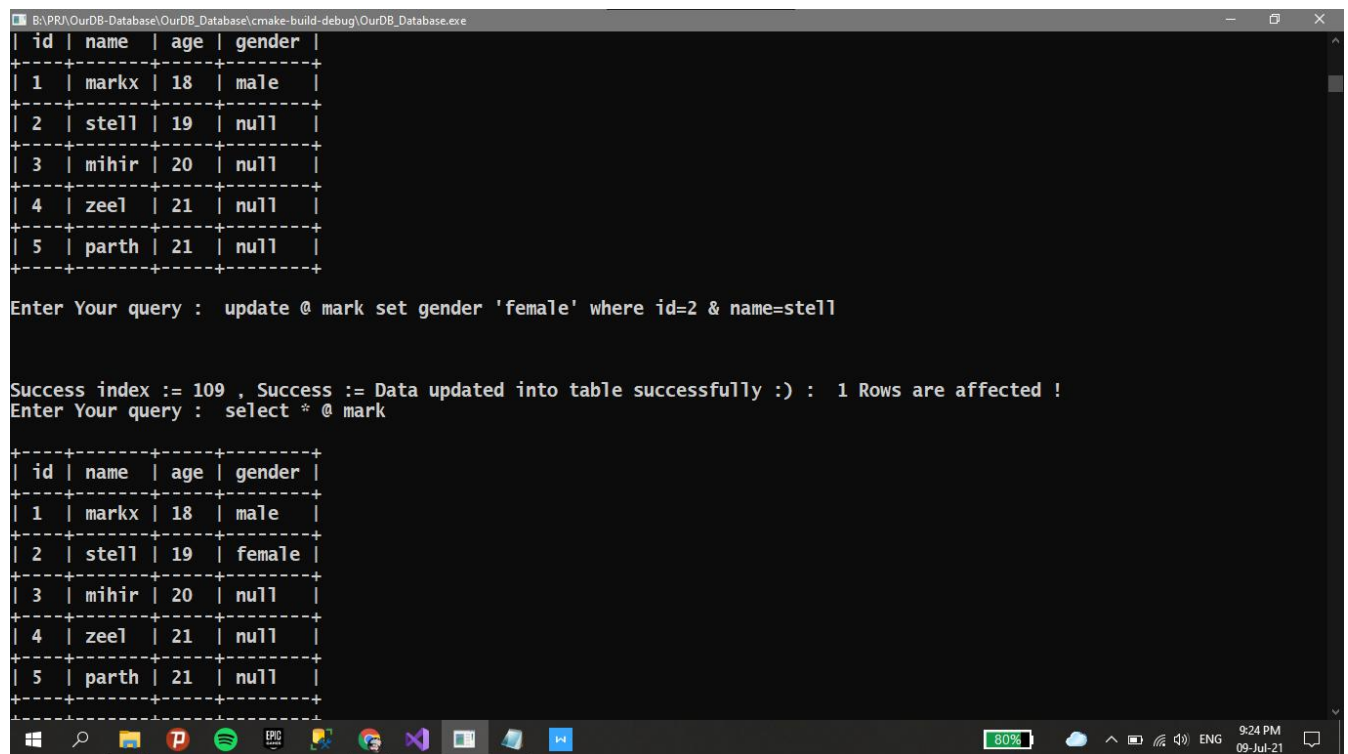
Here at particular row/rows you can update a data.

Here after where table field and value connected with the = operator.

Or !(not equal) operator.

And for more than one condition you can concatenation condition with either &(and) or |(or)

**Syntax : update @ <table\_name> set/put <table\_field> '<field\_value>'  
where <table\_field1>(<=|!><field\_value1> &  
<table\_field2>(<=|!><field\_value2>**



```
B:\PRJ\OurDB-Database\OurDB_Database\cmake-build-debug\OurDB_Database.exe
+---+---+---+---+
| id | name | age | gender |
+---+---+---+---+
| 1  | markx | 18  | male   |
+---+---+---+---+
| 2  | stell | 19  | null   |
+---+---+---+---+
| 3  | mihir | 20  | null   |
+---+---+---+---+
| 4  | zeel  | 21  | null   |
+---+---+---+---+
| 5  | parth | 21  | null   |
+---+---+---+---+

Enter Your query : update @ mark set gender 'female' where id=2 & name=stell

Success index := 109 , Success := Data updated into table successfully :) : 1 Rows are affected !
Enter Your query : select * @ mark

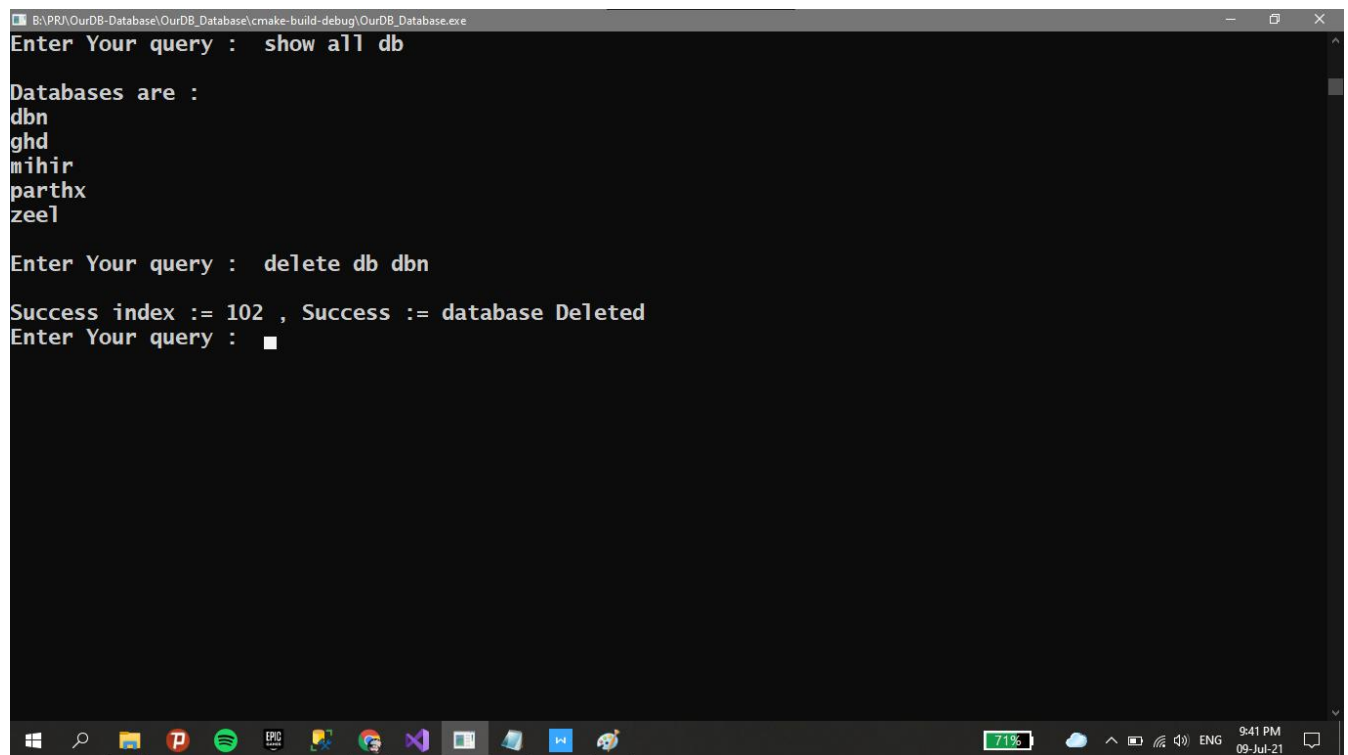
+---+---+---+---+
| id | name | age | gender |
+---+---+---+---+
| 1  | markx | 18  | male   |
+---+---+---+---+
| 2  | stell | 19  | female |
+---+---+---+---+
| 3  | mihir | 20  | null   |
+---+---+---+---+
| 4  | zeel  | 21  | null   |
+---+---+---+---+
| 5  | parth | 21  | null   |
+---+---+---+---+
```

- Delete the database :

It will delete the database from the system.

Keyword : delete or destroy both are accepted but not simultaneously

**Syntax : delete/destroy database <database\_name> // parth module**

A screenshot of a terminal window titled "B:\PRJ\OurDB-Database\OurDB\_Database\cmake-build-debug\OurDB\_Database.exe". The terminal shows a sequence of commands and outputs: "Enter Your query : show all db" followed by "Databases are : dbn, ghd, mihir, parthx, zeel"; then "Enter Your query : delete db dbn" followed by "Success index := 102 , Success := database Deleted"; and finally "Enter Your query : " with a cursor. The Windows taskbar is visible at the bottom with various application icons and system status indicators like battery level (71%) and time (9:41 PM, 09-Jul-21).

```
B:\PRJ\OurDB-Database\OurDB_Database\cmake-build-debug\OurDB_Database.exe
Enter Your query : show all db

Databases are :
dbn
ghd
mihir
parthx
zeel

Enter Your query : delete db dbn

Success index := 102 , Success := database Deleted
Enter Your query : █
```

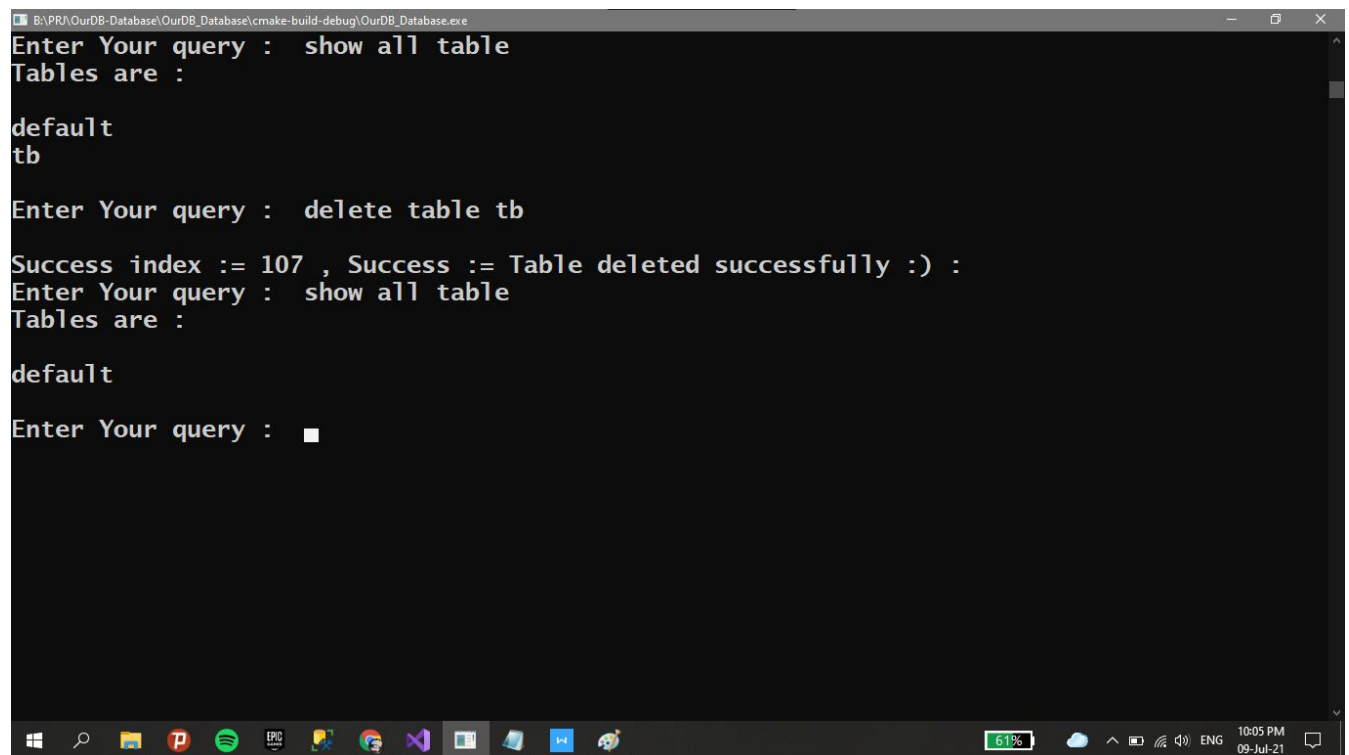
NOTE : Before deleting the database always make sure that your all the tables are deleted or you may loss important data.



- **Delete the table :**  
**Delete the table from the database.**

Note: make sure to select the database before deleting table.

**Syntax : delete table <table\_name>**



```
B:\PRJ\OurDB-Database\OurDB_Database\cmake-build-debug\OurDB_Database.exe
Enter Your query : show all table
Tables are :

default
tb

Enter Your query : delete table tb

Success index := 107 , Success := Table deleted successfully :) :
Enter Your query : show all table
Tables are :

default

Enter Your query : █
```

The screenshot shows a terminal window with a dark background. The title bar indicates the file path: B:\PRJ\OurDB-Database\OurDB\_Database\cmake-build-debug\OurDB\_Database.exe. The user enters the command 'show all table', and the output lists 'default' and 'tb'. Then, the user enters 'delete table tb', and the output shows a success message with an index of 107. Finally, the user enters 'show all table' again, and the output shows only 'default'.

- Delete the particular table data :

Delete the data from the table at particular row/rows .

Note: make sure to select the database before deleting table.

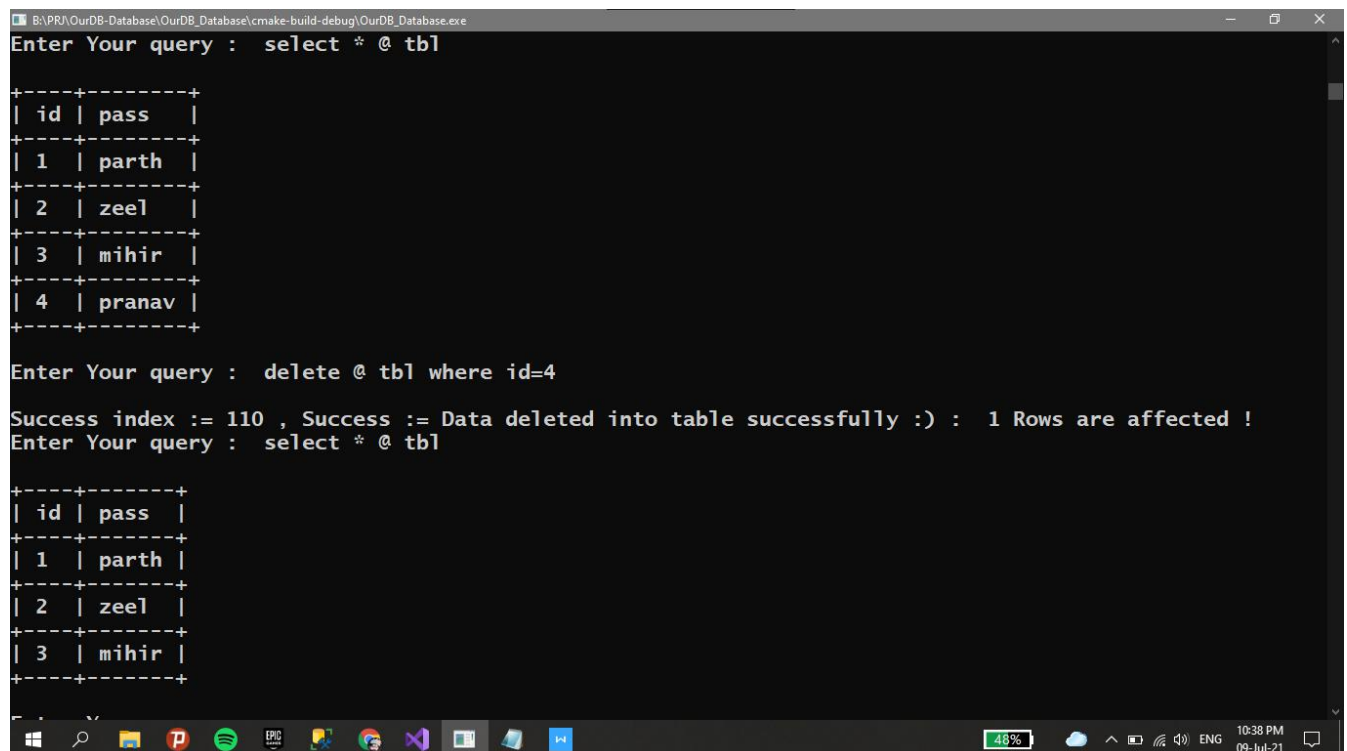
Here at particular row/rows you can delete a data.

Here after where table field and value connected with the = operator.

Or !(not equal) operator.

And for more than one condition you can concatenation condition with either &(and) or |(or)

**Syntax : delete/destroy @ <table\_name> where <column\_name> (=/!) <column\_value> (&/|) <column\_name1> (=/!) <column\_value1>**



```
B:\PRJ\OurDB-Database\OurDB_Database\cmake-build-debug\OurDB_Database.exe
Enter Your query : select * @ tbl

+-----+
| id | pass |
+-----+
| 1 | parth |
+-----+
| 2 | zeel |
+-----+
| 3 | mihir |
+-----+
| 4 | pranav |
+-----+

Enter Your query : delete @ tbl where id=4

Success index := 110 , Success := Data deleted into table successfully :) : 1 Rows are affected !
Enter Your query : select * @ tbl

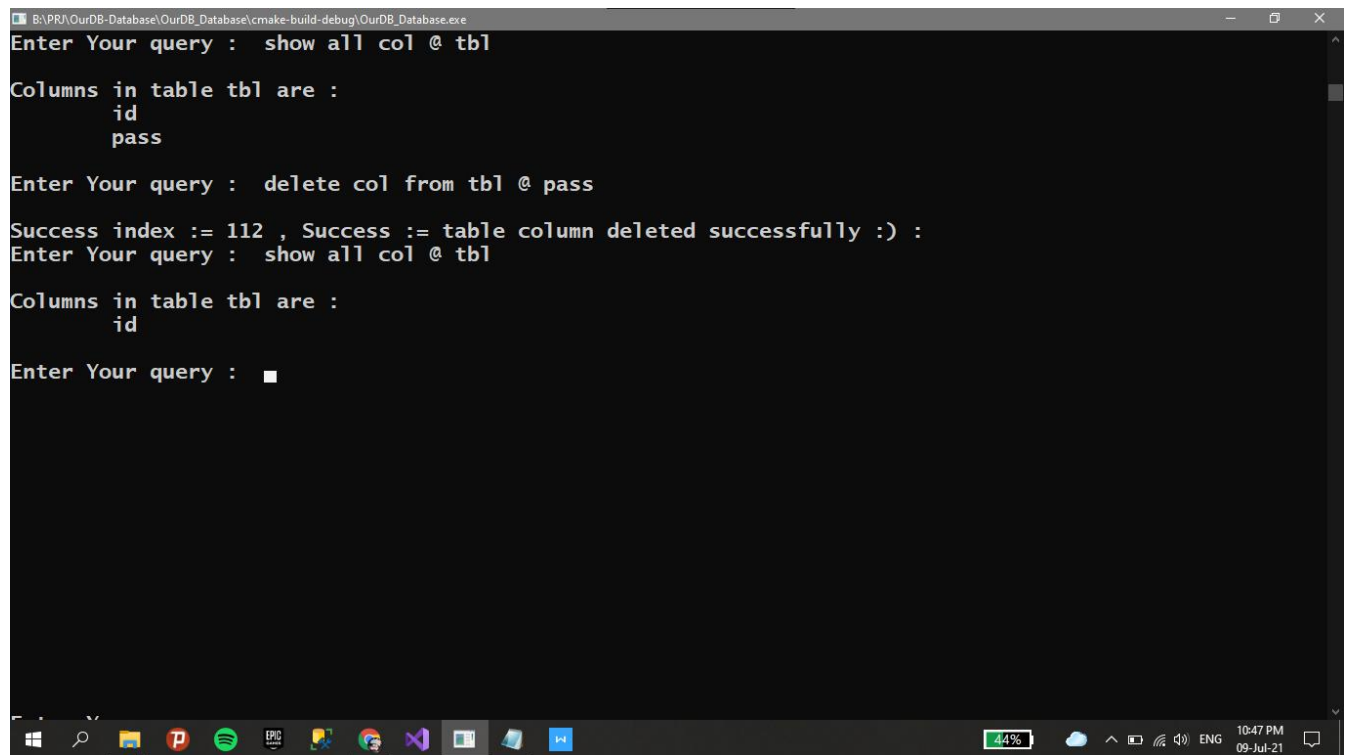
+-----+
| id | pass |
+-----+
| 1 | parth |
+-----+
| 2 | zeel |
+-----+
| 3 | mihir |
+-----+
```

- **Delete the column :**

**This will delete the column from the table .**

Note : make sure in which table you are going to insert the column ,that must be created before insertion of column.

**Syntax : delete/destroy column/col from <table\_name> @ <col\_name>**



```
B:\PRJ\OurDB-Database\OurDB_Database\cmake-build-debug\OurDB_Database.exe
Enter Your query : show all col @ tbl

Columns in table tbl are :
    id
    pass

Enter Your query : delete col from tbl @ pass

Success index := 112 , Success := table column deleted successfully :) :
Enter Your query : show all col @ tbl

Columns in table tbl are :
    id

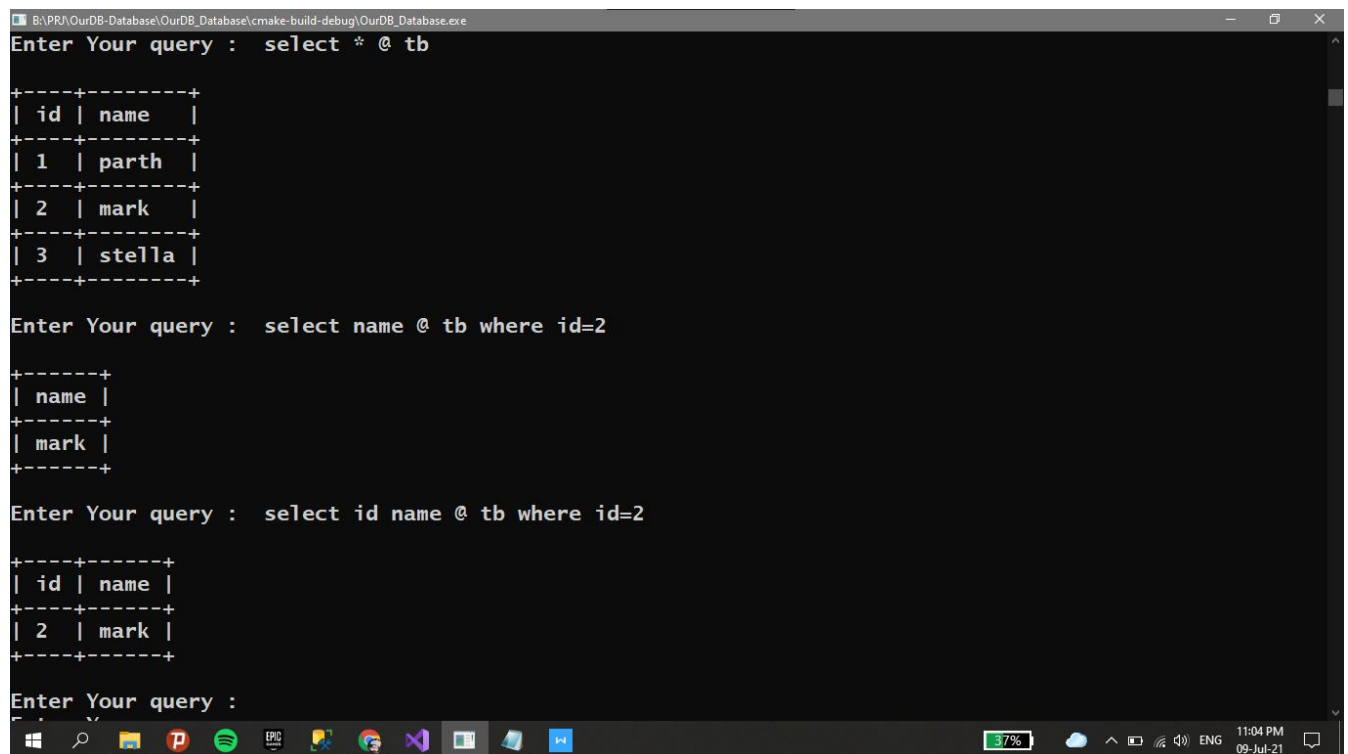
Enter Your query : █
```

- **Select data from table:**

It will select and display the the data into the table format.

**Syntax : select/choose \* @ <table\_name>**

**Syntax : select/choose col1 col2 @ <table\_name> where  
<column\_name> (=!) <column\_value> (&/|) <column\_name1>  
(=!) <column\_value1>**



```
B:\PRA\OurDB-Database\OurDB_Database\cmake-build-debug\OurDB_Database.exe
Enter Your query : select * @ tb

+-----+
| id | name |
+-----+
| 1 | parth |
+-----+
| 2 | mark |
+-----+
| 3 | stella |
+-----+

Enter Your query : select name @ tb where id=2

+-----+
| name |
+-----+
| mark |
+-----+

Enter Your query : select id name @ tb where id=2

+-----+
| id | name |
+-----+
| 2 | mark |
+-----+

Enter Your query :
```

The screenshot shows a terminal window with a Windows taskbar at the bottom. The terminal displays three SQL queries and their results in a table format. The first query 'select \* @ tb' returns a table with 3 rows. The second query 'select name @ tb where id=2' returns a single row. The third query 'select id name @ tb where id=2' returns a single row. The taskbar shows various application icons and system status information like 37% battery and 11:04 PM on 09-Jul-21.

## Encryption And Decryption:

The data stored in the database should be secure and can't be read by the user or any other, While they are stored in the database.

So that we created our own encryption and decryption method to encrypt and decrypt data respectively.

The encryption and decryption use in this we gave name as **69Crypton**.

This **69Crypton** method encrypt the data before stored in the json format And decrypt the data after user ask to retrieve it. So that user can see the encrypted data.

## How actually data stored in the table :

The Encrypted data are stored in to the <file\_name>.Ourdb (here Ourdb extension is especially for our database system).

While user run query "create table" , the <file\_name>.Ourdb is stored in specified database. Which was previously selected by the user.

While user uses "insert into" query, the data stored in the file as Json format.

## Future Enhancement :

We have code in our github directory. we will put directory as public so every one will be able to take advantage of database and also then can add more feature . we have created in such a way that , changes can be easily made.

So that we want to create a big community which can enhance the code.