ORMEDIAN RESEARCH INSTITUTE

TOPIC:

DATABASE

SUBTOPIC: SQL FUNDAMENTALS (PART 7)

By:

ADEKOLA OLUWASEUN O.

JANUARY 4, 2023

CONTENT COVERED

- *Understanding What SQL DEFAULT Constraint is*
- *Applying DEFAULT Constraint during Table creation and alteration*
- *Working with INDEXES in SQL*
- *Using AUTO INCREMENT in SQL*
- *Working with DATES in SQL and How to implement it*

DEFAULT CONSTRAINT

*The **DEFAULT** constraint is used to automatically assign default value to any column in which no other value has been specified. This technique allows us to set values for a given field automatically even if no other value has been specified by the user while entering their data. It is not only limited to user defined fields, DEFAULT constraint can still be used on some inbuilt functions like GETDATE() function.*

For example, Departmental registration form may require that only students from Electronic and Computer Engineering (ECE) can submit their entries. Hence, this form could have been given an initial condition as at the time of creation such that it saves the department field of the form as ECE. In this case, it doesn't matter whether the user enters value for this or not. The system assumes that the user trying to submit details is from the ECE department.

Assuming we need to create a table called ECE_records to hold students FirstName, MatricNo, Department and Level.

Syntax for this can be written in two different ways depending on whether we are creating a new table or we just want to alter an existing table.

DEFAULT CONSTRAINT CONTINUATION

To implement that scenario,

CREATE TABLE ECE_records (ID INT AUTO_INCREMENT PRIMARY KEY, NAME VARCHAR(255), LEVEL INT, DEPARTMENT VARCHAR(220) DEFAULT "ECE");

SETTING DEFAULT CONSTRAINT DURING TABLE ALTERATION

Let us assume that the ECE_records table does not have DEFAULT constraint place in it during the table creation process, and we have just decided to alter the table and add a DEFAULT constraint to the Department Field of the table.

SYNTAX

ALTER TABLE ECE_records ALTER Department SET DEFAULT 'ECE';

REMOVAL OF DEFAULT CONSTRAINT

Just like other constraints, the DROP keyword can be used to remove the DEFAULT constraint from a given field attached to a table if we see that we don't need such DEFAULT constraint again.

SYNTAX: ***ALTER TABLE DEF ALTER DEPARTMENT DROP DEFAULT;***

OR

ALTER TABLE DEF ALTER COLUM DEPARTMENT DROP DEFAULT;

INDEXES IN SQL

This is a functionality that are used to retrieve data. They make searching of data faster. Instead of using other keyword in SQL to search for data in the database, we can use indexes to get a good result in a little time. Creation of indexes during the table creation process is a great way to make to conquer searching and updation problems.

One advantage of using indexes is that we can create duplicate values for it and also we can create unique values where duplication is not allowed. This depends strictly on our choice. It is important to remember that once a particular index name has been used for a given column or combined column, it cannot be used for another column.

It is worth noting that syntax could be different depending on which database we are using.

FOR CASES WHERE WE WANT TO ALLOW FOR DUPLICATE VALUES

CREATE INDEX index_name ON Tablename (column1, ccolumn2,…..columnN);

FOR CASES WHERE WE DON'T WANT TO ALLOW FOR DUPLICATE VALUES

The previous code is further modified to the form shown below:

CREATE UNIQUE INDEX index_name on Tablename (column1, column2,…….columnN);

The keyword UNIQUE is the only difference that makes the two approaches different. The former can handle duplication of date while the later does not.

INDEXES

Continuation…….

Another advantage of index is that it can be applied on a single column or on multiple columns.

On a single column: ***CREATE INDEX index_name ON Tablename (column_name);***

On multiple columns: ***CREATE INDEX index_name ON Tablename (column1, column2, ….columnN);***

EXAMPLE

Let us consider a table SALES with the following fields: ID, CUSTOMER'S_NAME, PRODUCT_REQUESTED, ORDERNO, AMOUNT

If our decision is to place an index on the customer's_name then we can have our syntax written in this form by simply specifying the name of the column to be indexed.

SYNTAX BECOMES

CREATE INDEX sales_custindex ON SALES (customer's_name);

From the syntax written above, it can be inferred that we have placed sales_custindex as the index_name on the customer's_name field of the SALES table.

But if our intension is to combine is to give index name to multiple columns, we can modify our code to this form:

Assuming we are interested in columns customers's_name and product_requested.

CREATE INDEX sales_custindex ON SALES (customer's_name, product_requested);

INDEXES

Continuation…….

There is always an opportunity to remove indexes given to a particular column or combined columns. To achieve this, we need to use the DROP keyword.

Syntax to implement this works differently on different databases.

To drop an index on SQL, dot operator is used between the tablename and the indexname.

SYNTAX: ***DROP INDEX table_name.index_name;***

While in MySQL the syntax take the form of :

ALTER TABLE table_name DROP INDEX index_name;

Let's say we want to drop the index_name ***sales_custindex*** *attached to the SALES table we created previously.*

ALTER TABLE SALES DROP INDEX sales_custindex;

OR

DROP INDEX SALES. ***sales_custindex;***

AUTO_INCREMENT IN SQL

This is a functionality in SQL that allows for a unique identifier to be generated automatically whenever a new record is inserted into the database table. This functionality by default uses increment order of (+1) as the record in the table increases, meaning that the starting value is 1. But this does not mean that we can not use the increment order of choice. We may decide that the increment should be in order of +5 or +anything positive.

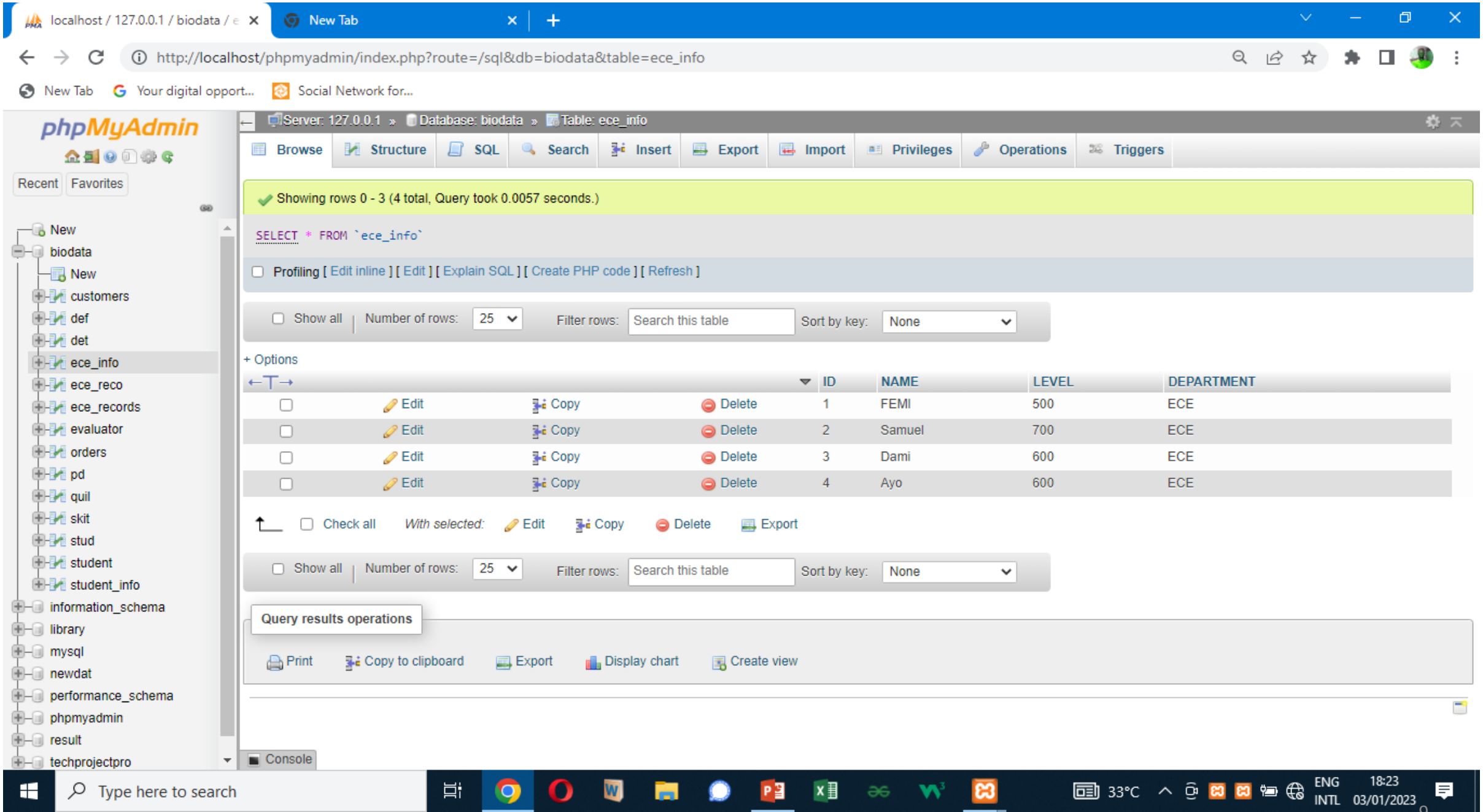
Let us create a new table called ece_info table and implement the auto increment on the id field the table.

SCENARIO 1: *Assuming in the sales table, we want the ID field to have AUTO_INCREMENT functionality. Then the syntax will become:*

CREATE TABLE ECE_INFO (ID INT AUTO_INCREMENT PRIMARY KEY, NAME VARCHAR(255), LEVEL INT, DEPARTMENT VARCHAR(220));

Result shown in the next slide was obtained after inserting some records into the table. To demonstrate the (+1) automatic increment of the ID. This is the syntax that works well on MySQL Database.

Link to this implementation is available @ https://github.com/SafersTechnologies/SQL-Tutorials/blob/master/ece_info.sql


localhost / 127.0.0.1 / biodata /
New Tab
http://localhost/phpmyadmin/index.php?route=/sql&db=biodata&table=ece_info
New Tab
Your digital opport...
Social Network for...
phpMyAdmin
Recent
Favorites
New
biodata
New
customers
def
det
ece_info
ece_reco
ece_records
evaluator
orders
pd
quil
skit
stud
student
student_info
information_schema
library
mysql
newdat
performance_schema
phpmyadmin
result
techprojectpro
Server: 127.0.0.1 » Database: biodata » Table: ece_info
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Operations
Triggers
Showing rows 0 - 3 (4 total, Query took 0.0057 seconds.)
SELECT * FROM `ece_info`
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all
Number of rows: 25
Filter rows: Search this table
Sort by key: None
+ Options
ID NAME LEVEL DEPARTMENT
Edit Copy Delete 1 FEMI 500 ECE
Edit Copy Delete 2 Samuel 700 ECE
Edit Copy Delete 3 Dami 600 ECE
Edit Copy Delete 4 Ayo 600 ECE
Check all With selected: Edit Copy Delete Export
Show all
Number of rows: 25
Filter rows: Search this table
Sort by key: None
Query results operations
Print
Copy to clipboard
Export
Display chart
Create view
Console
Type here to search
33°C
ENG
INTL
18:23
03/01/2023

AUTO INCREMENT IN SQL

Continuation…….

We can have some specifics to make auto increment work in a certain way. For example if we choose to have a different starting value for the auto increment unlike the usual way. Let us assume that we want the starting value to be 200, maybe we have some special records up to 199 somewhere in our database and we wish to further add more to later. It is a matter of choice. We can definitely start our sequence with any number that we feel like.

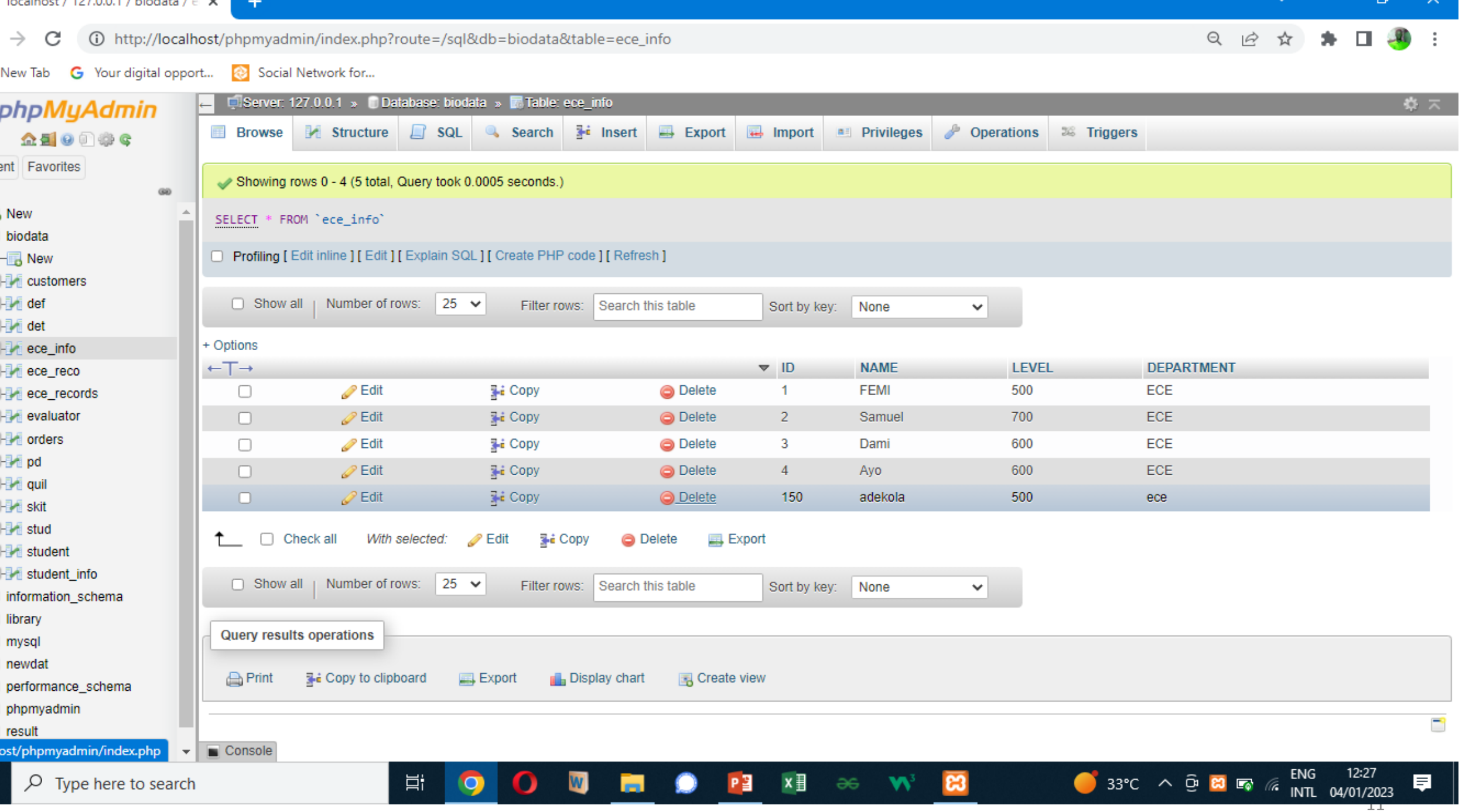
We might decide to change the starting value of the ID to 150 or anything. I am going to implement this on the previously created ECE_info table.

SYNTAX

ALTER TABLE ece_info AUTO_INCREMENT=150;

Result obtained from this implementation is displayed in the next slide.

Link to this implementation is available @ https://github.com/SafersTechnologies/SQL-Tutorials/blob/master/ece_info_auto_increment_to_150.sql


http://localhost/phpmyadmin/index.php?route=/sql&db=biodata&table=ece_info
phpMyAdmin
Server: 127.0.0.1 » Database: biodata » Table: ece_info
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Operations
Triggers
Showing rows 0 - 4 (5 total, Query took 0.0005 seconds.)
SELECT * FROM `ece_info`
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None
+ Options
ID NAME LEVEL DEPARTMENT
1 FEMI 500 ECE
2 Samuel 700 ECE
3 Dami 600 ECE
4 Ayo 600 ECE
150 adekola 500 ece
Check all With selected: Edit Copy Delete Export
Query results operations
Print Copy to clipboard Export Display chart Create view
Console
Type here to search
33°C
ENG INTL
12:27 04/01/2023

AUTO INCREMENT IN SQL

Continuation…….

Although the auto increment syntax differs from database to database. The way MySQL implements it is not the same as that of either SQL or Oracle or Access database.

*In **MS SQL,** the identity KEYWORD is used to achieve the auto increment functionality. The starting point and the steps(value it should be incremented by) must be stated for the order to be well taken care of.*

For example IDENTITY (30, 3) implies that the starting point should be 10, and every new record must have its ID incremented by 3.

Taking the ECE_info Table as an example again, auto increment can be included in MS SQL this way:

CREATE TABLE ECE_INFO (ID INT IDENTITY (1,1) PRIMARY KEY, NAME VARCHAR(255), LEVEL INT, DEPARTMENT VARCHAR(220));

*In **ACCESS DATABASE**, this is very much related to the way the implementation is done in MySQL, the only difference is that in Access database, the underscore symbol separating the AUTO INCREMENT is being replaced by a space.*

Whereas in Oracle database, the logic behind the auto increment is entirely different.

CREATE TABLE ECE_INFO (ID INT AUTO_INCREMENT PRIMARY KEY, NAME VARCHAR(255), LEVEL INT, DEPARTMENT VARCHAR(220));

*In **ORACLE DATABASE**, before we can successfully implement the auto increment in the Oracle database we basically need to specify some parameters (MINVALUE, START, INCREMENT, CACHE (this is needed because we need to specify how many values of the sequence of the database preallocates and keeps in the memory for faster access)) before we can successfully achieve the desired result. the most important part of it is the CREATE SEQUENCE statement.*

WORKING WITH DATES IN SQL

Date is a data type that serves as a very useful functionality in databases generally.

For example, if Queens University Belfast develops on online application system for admission processing. It is expected that for good record tracking, registration slips after completion of online form should have the submission date and time assigned to it. For this to be achieved, the backend program for the database must have been structured in such a way that it will attach date functionality to the form automatically after submission.

Common Date data type in databases and their formats are give below:

DATE DATA TYPE	FORMAT
YEAR	YYYY or YY
DATETIME/SMALL DATETIME	YYYY-MM-DD HH:MI:SS
TIMESTAMP	YYYY-MM-DD HH:MI:SS
DATE	YYYY-MM-DD

DATE DATA TYPE IN SQL

Continuation

SYNTAX for using the date data type differs from one database system to another. In this presentation, examples to be practiced will simply be carried out on MySQL database.

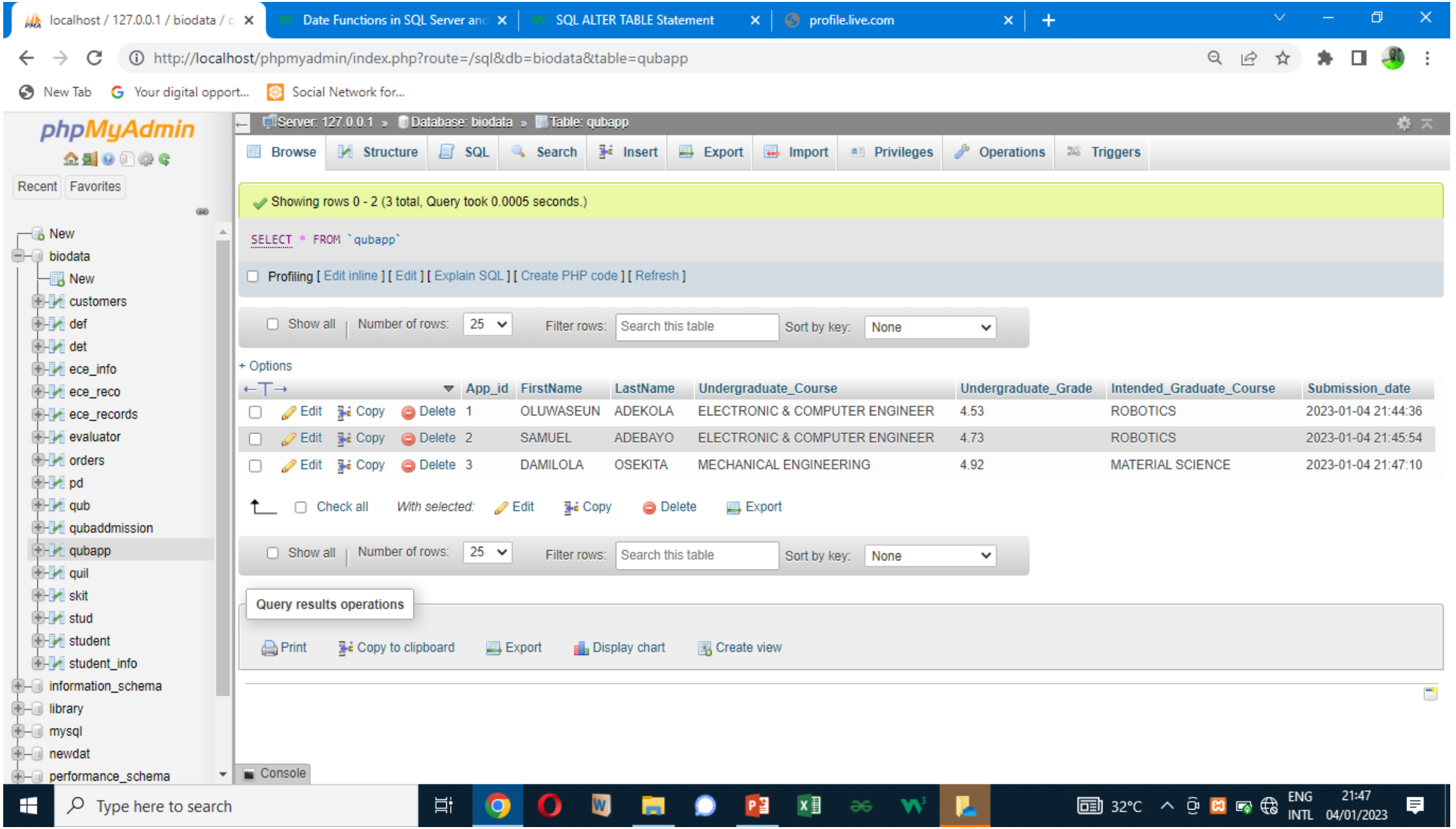
SCENARIO

Let us build a database system that stores online information submitted by applicants for graduate school admission processing into Queens University Belfast. Our assumption is that such system must be able to record the time at which any of the applicants submits his/her information.

CREATE TABLE QUBAPP (App_id int NOT NULL AUTO_INCREMENT PRIMARY KEY, FirstName varchar(20), LastName varchar(20), Undergraduate_Course varchar(30), Undergraduate_Grade FLOAT, Intended_Graduate_Course varchar (50), Submission_date datetime default now());

After inserting values into the table created above, the sample of result obtained is shown in the next slide.

Link to this implementation is available @ https://github.com/SafersTechnologies/SQL-Tutorials/blob/master/qubapp.sql


localhost / 127.0.0.1 / biodata /
Date Functions in SQL Server and
SQL ALTER TABLE Statement
profile.live.com
http://localhost/phpmyadmin/index.php?route=/sql&db=biodata&table=qubapp
New Tab
Your digital opport...
Social Network for...
phpMyAdmin
Recent
Favorites
New
biodata
New
customers
def
det
ece_info
ece_reco
ece_records
evaluator
orders
pd
qub
qubaddmission
qubapp
quil
skit
stud
student
student_info
information_schema
library
mysql
newdat
performance_schema
Server: 127.0.0.1 » Database: biodata » Table: qubapp
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Operations
Triggers
Showing rows 0 - 2 (3 total, Query took 0.0005 seconds.)
SELECT * FROM `qubapp`
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all
Number of rows: 25
Filter rows: Search this table
Sort by key: None
+ Options
App_id FirstName LastName Undergraduate_Course Undergraduate_Grade Intended_Graduate_Course Submission_date
Edit Copy Delete 1 OLUWASEUN ADEKOLA ELECTRONIC & COMPUTER ENGINEER 4.53 ROBOTICS 2023-01-04 21:44:36
Edit Copy Delete 2 SAMUEL ADEBAYO ELECTRONIC & COMPUTER ENGINEER 4.73 ROBOTICS 2023-01-04 21:45:54
Edit Copy Delete 3 DAMILOLA OSEKITA MECHANICAL ENGINEERING 4.92 MATERIAL SCIENCE 2023-01-04 21:47:10
Check all
With selected:
Edit
Copy
Delete
Export
Show all
Number of rows: 25
Filter rows: Search this table
Sort by key: None
Query results operations
Print
Copy to clipboard
Export
Display chart
Create view
Console
Type here to search
32°C
ENG
INTL
21:47
04/01/2023

THANKS FOR VIEWING

NOTE: ***Subsequent topics under SQL will be discussed in Part 8 Also, all Implementation as far as this presentation is concerned are in line with MySQL Syntax. All have been tested on XAMPP Server.***

THE NEXT PART WILL COVER FEW MORE TOPICS SUCH AS VIEWS, SQL INJECTION, HOSTING and DATA TYPE.