Introduction to Database

MySQL

phpMyAdmin

C# & MySQL

http://www.codeproject.com/Articles/43438/Connect-C-to-MySQL

โครงสร้างฐานข้อมูลเชิงสัมพันธ์

Relational Data Model

- ข้อมูลถูกจัดเก็บในฐานข้อมูลในรูปแบบตาราง (Table)
  - ในแต่ละตารางประกอบ Field (หรือ Attribute) ต่างๆ และแต่ละ Field อาจจะมีชนิดของข้อมูล (Data Type) ที่แตกต่างกันไป
  - ข้อมูลที่จัดเก็บอยู่ในแต่ละตาราง ถูกจัดเก็บในรูปแบบของ Record (หรือ Row หรือ Tuple)

ชนิดของข้อมูล (Data Type)

Data Type

- Numeric data types for integers and real numbers
- Characters
- Booleans
- Fixed-length strings
- Variable-length strings
- Date, time, timestamp
- Money
- Other special data types

ตัวอย่าง Table

(Fields)

Attributes

Relation Name

STUDENT

(Records)

Tuples

Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25

Figure 1
The attributes and tuples of a relation STUDENT.

<u>หมายเหตุ:</u> NULL หมายถึง ไม่มีข้อมูล (Empty)

MySQL

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation.

MySQL databases are relational.

ฐานข้อมูลที่สร้างขึ้นโดยใช้ MySQL **ถูกจัดเก็บอยู่ในรูปแบบของ** Relational Database

<u>หมายเหตุ:</u> การสร้างฐานข้อมูล MySQL ในที่นี้ใช้ phpMyAdmin และ phpMyAdmin สามารถสืบค้นข้อมูลด้วย QBE

1. เข้าใช้งาน

- ชุดซอฟต์แวร์ที่ใช้

http://www.wampserver.com/en/

1. เข้าใช้งาน (ต่อ)

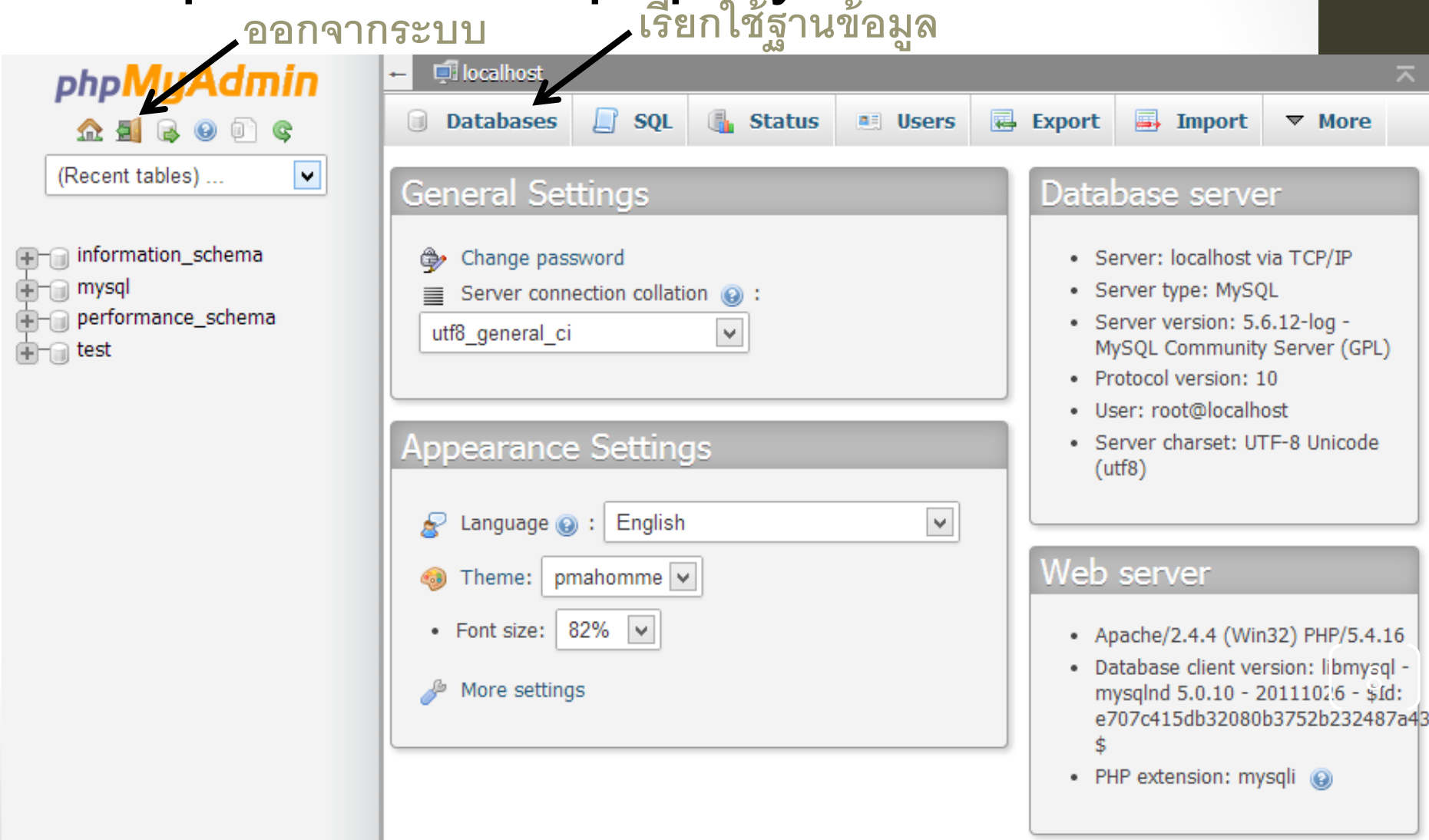
- http://localhost/phpmyadmin/

แต่ต้องมั่นใจว่า WAMPSERVER online

(Start WampServer)

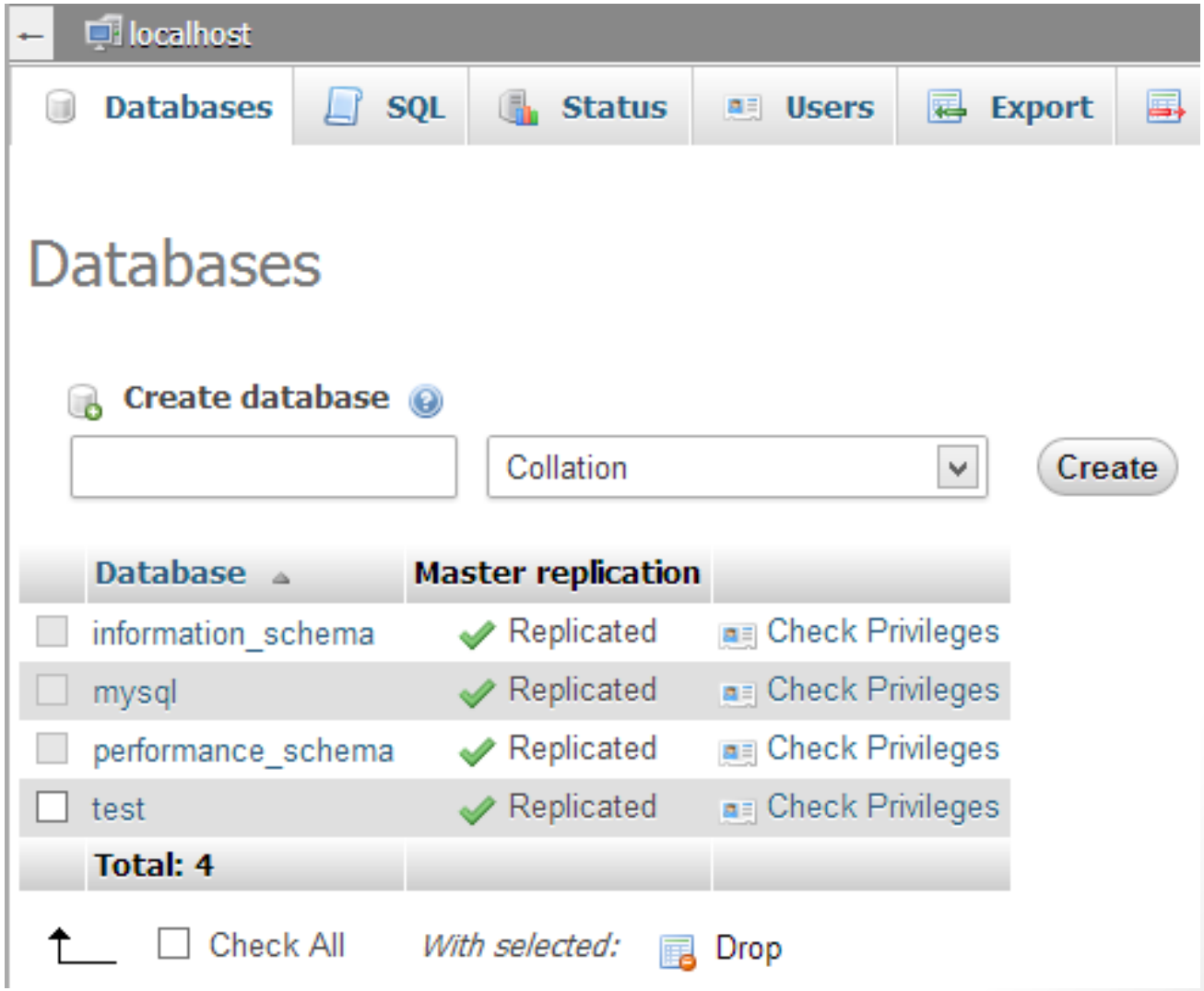
1. เข้าใช้งาน (ต่อ)

- http://localhost/phpmyadmin/



1. เข้าใช้งาน (ต่อ)

- เลือกฐานข้อมูลที่ต้องการหรือสร้างฐานข้อมูลใหม่



2. สร้างฐานข้อมูลใหม่

- สร้างฐานข้อมูลชื่อ TestDB

Databases

Create database

TestDB | Collation | Create

เลือก utf8_general_ci
เพื่อให้ Support ภาษาไทย

(Recent tables) ...

- information_schema
- mysql
- performance_schema
- test
- testdb

Databases

Create database

TestDB | Collation | Create

Database	Master replication	
information_schema	Replicated	Check Privileges
mysql	Replicated	Check Privileges
performance_schema	Replicated	Check Privileges
test	Replicated	Check Privileges
TestDB	Replicated	Check Privileges
Total: 5		

Check All *With selected:* Drop

วิธีลบฐานข้อมูล

- สิ่งที่ได้

เลือกใช้ฐานข้อมูลได้ 2 ที่

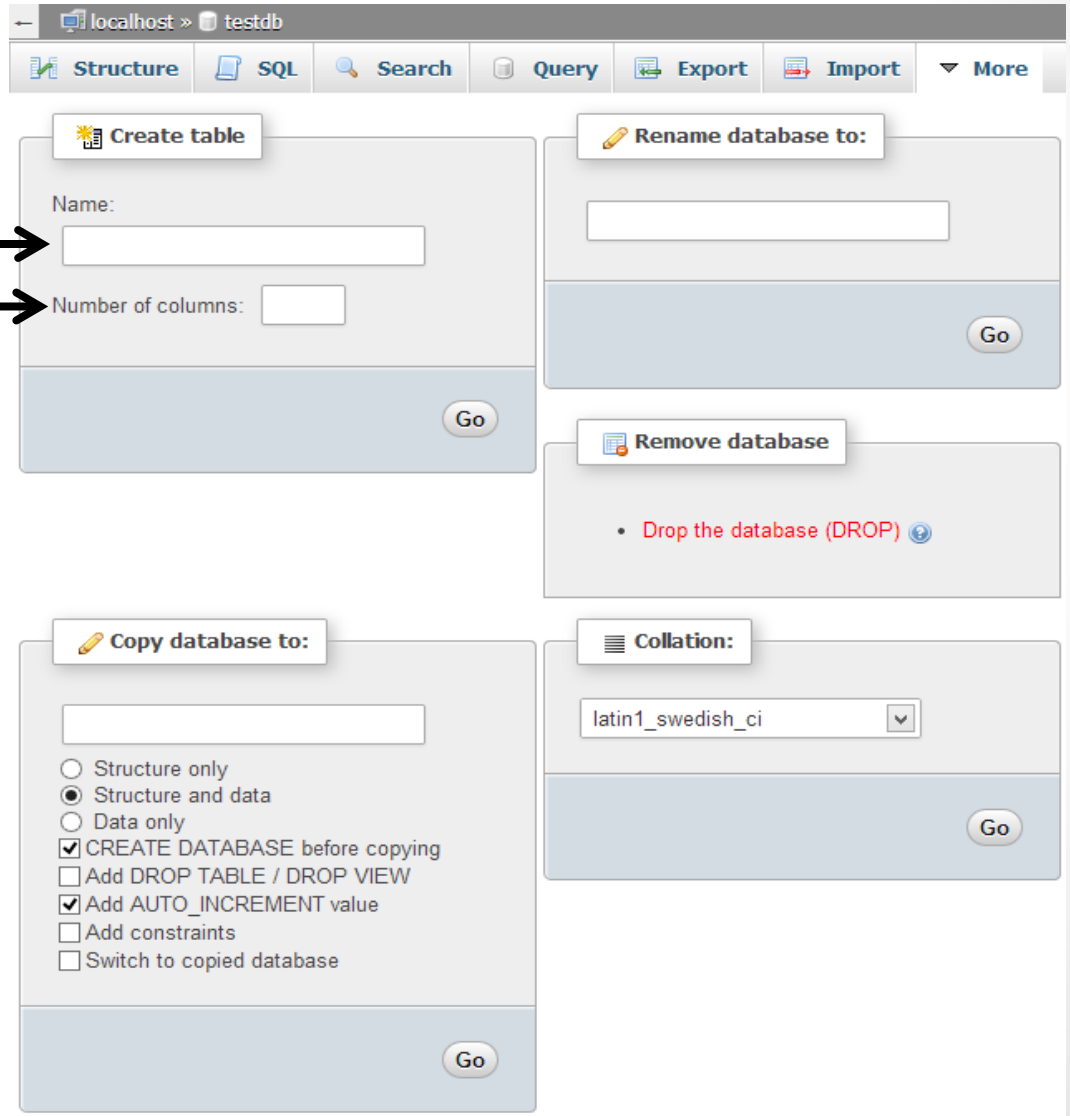
3. สร้างตารางเก็บข้อมูล

- กรณีศึกษา: นักศึกษาสังกัดภาควิชา
  - ข้อมูลนักศึกษาประกอบด้วย
    - รหัสนักศึกษา
    - คำนำหน้าชื่อ
    - ชื่อ
    - นามสกุล
    - วันเกิด
    - เกรดเฉลี่ยสะสม
    - ภาควิชาที่สังกัด
  - คณะวิทยาศาสตร์ประกอบด้วย 8 ภาควิชา
  - นักศึกษาแต่ละคนจะสังกัดเพียงภาควิชาเดียว
- ระบบฐานข้อมูลประกอบด้วย 2 ตาราง คือ
  - STUDENT
  - DEPARTMENT

3. สร้างตารางเก็บข้อมูล (ต่อ)

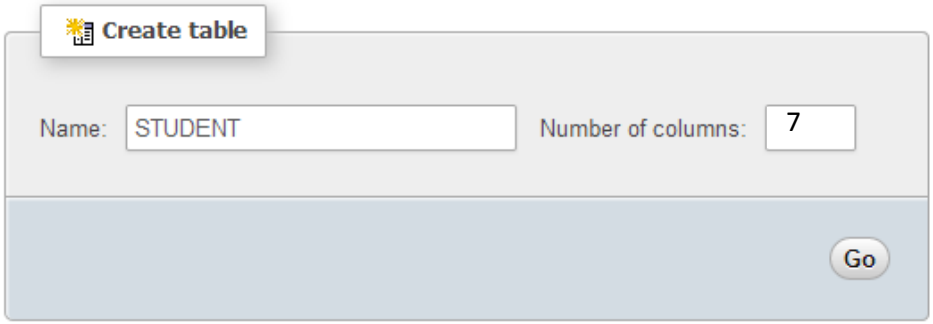
- เลือกฐานข้อมูล TestDB

สร้างตาราง และ
ระบุจำนวนคอลัมน์



3. สร้างตารางเก็บข้อมูล (ต่อ)

- สร้างตาราง STUDENT ประกอบด้วย 7 ฟิลด์



- ฟิลด์ (Field) หรือแอตตริบิวต์ (Attribute) หรือคุณสมบัติตาราง

ชื่อฟิลด์	คุณสมบัติ	ความหมาย
studentID	Char (9)	รหัสนักศึกษา กำหนดเป็น Primary Key
title	Varchar(15)	คำนำหน้าชื่อ
firstname	Varchar(20)	ชื่อนักศึกษา
lastname	Varchar(25)	นามสกุลนักศึกษา
birthdate	Date	วันเกิด
gpa	Float	เกรดเฉลี่ยสะสม
major	Char(4)	ชื่อย่อภาควิชาที่สังกัดจาก DEPARTMENT

3. สร้างตารางเก็บข้อมูล (ต่อ)

- คุณสมบัติของฟิลด์ที่เป็น Primary Key คือ
  - แต่ละระเบียนจะมีข้อมูลในฟิลด์นี้จะต้องไม่ซ้ำกัน
    - เช่น รหัสนักศึกษา เลขที่บัตรประชาชน e-mail ของนศ. แต่ละคน
  - เมื่ออ้างถึงข้อมูลในฟิลด์นี้จะรู้ว่าเป็นระเบียนใด
    - เช่น 3825177 คือ นายสมชาย จริงใจ

3 สร้างตารางเก็บข้อมูล (ต่อ)

- ชนิดข้อมูลใน MySQL

DATA TYPES	
CHAR	String, length 0 - 255
VARCHAR	String, length 0 - 255
TINYTEXT	String, length 0 - 255
TEXT	String, length 0 - 65535
BLOB	String, length 0 - 65535
MEDIUMTEXT	String, length 0 - 16777215
MEDIUMBLOB	String, length 0 - 16777215
LONGTEXT	String, length 0 - 4294967295
LONGBLOB	String, length 0 - 4294967295
* TINYINT	Integer, -128 to 127
* SMALLINT	Integer, -32768 to 32767
* MEDIUMINT	Integer, -8388608 to 8388607
* INT	Integer, -2147483648 to 2147483647
* BIGINT	Int, -9223372036854775808 to 9223372036854775807
FLOAT	Decimal (precise to 23 digits)
DOUBLE	Decimal (24 to 53 digits)
DECIMAL	"DOUBLE" stored as string
DATE	YYYY-MM-DD
DATETIME	YYYY-MM-DD HH:MM:SS
TIMESTAMP	YYYYMMDDHHMMSS
TIME	HH:MM:SS
ENUM	One of preset options
SET	Selection of preset options

** Note: "UNSIGNED" TINYINT, SMALLINT, MEDIUMINT, INT, BIGINT have the same range of values but start at 0, e.g. TINYINT UNSIGNED is between 0 and 255.*

3 สร้างตารางเก็บข้อมูล (ต่อ)

- สร้าง Field (ฟิลด์) หรือ Attribute (แอตทริบิวต์)

Server: mysql wampserver » Database: testdb » Table: Student

Browse | Structure | SQL | Search | Insert | Export | Import | Privileges | Operations | Triggers

Table name: Student Add 1 column(s) Go

ระบุ Primary Key

Structure

Name	Type	Length/Values	Default	Collation	Attributes	Null	Index	A_I	Comments
studentID	CHAR	9	None			☐	PRIMARY	☐	รหัสประจำตัวนักศึกษา
title	VARCHAR	15	None			☐	---	☐	คำนำหน้าชื่อ
firstName	VARCHAR	15	None			☐	---	☐	ชื่อนักศึกษา
lastName	VARCHAR	20	None			☐	---	☐	นามสกุลนักศึกษา
birthDate	DATE		None			☑	---	☐	วันเกิด
gpa	FLOAT	4	As defined: 0.00			☑	---	☐	GPA
major	CHAR	4	None			☑	---	☐	

Table comments: ข้อมูลนักศึกษา

Storage Engine: InnoDB

Collation:

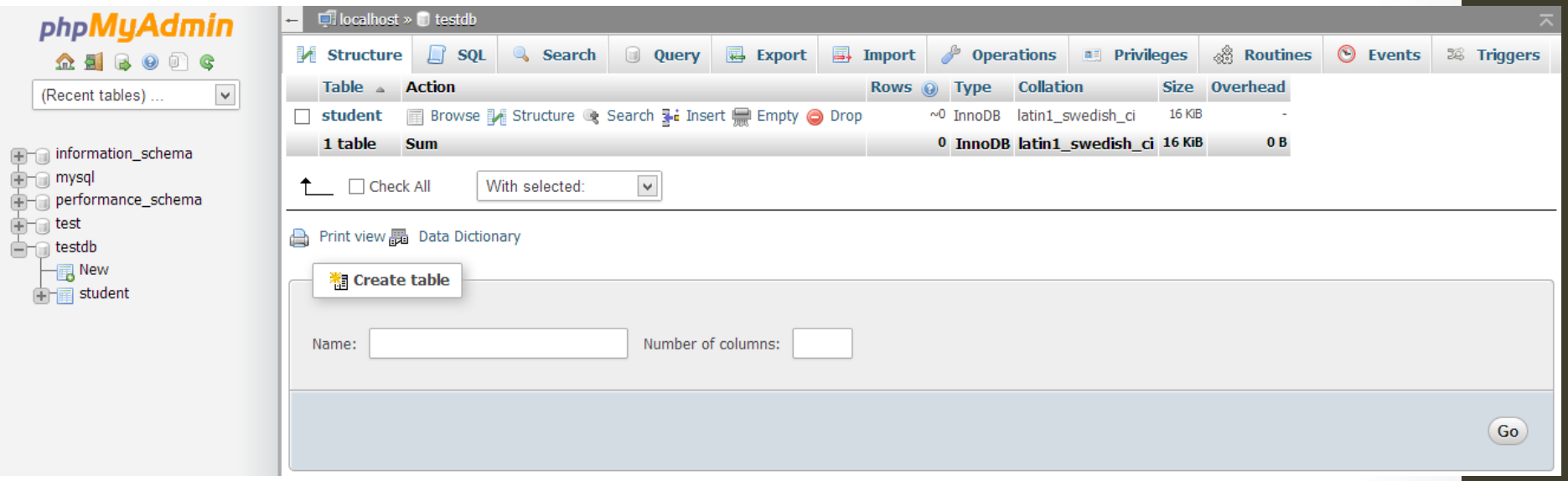
PARTITION definition:

บางคอลัมน์อนุญาต ให้ข้อมูลเป็น null ได้

Save

3 สร้างตารางเก็บข้อมูล (ต่อ)

- ผลลัพธ์ที่ได้



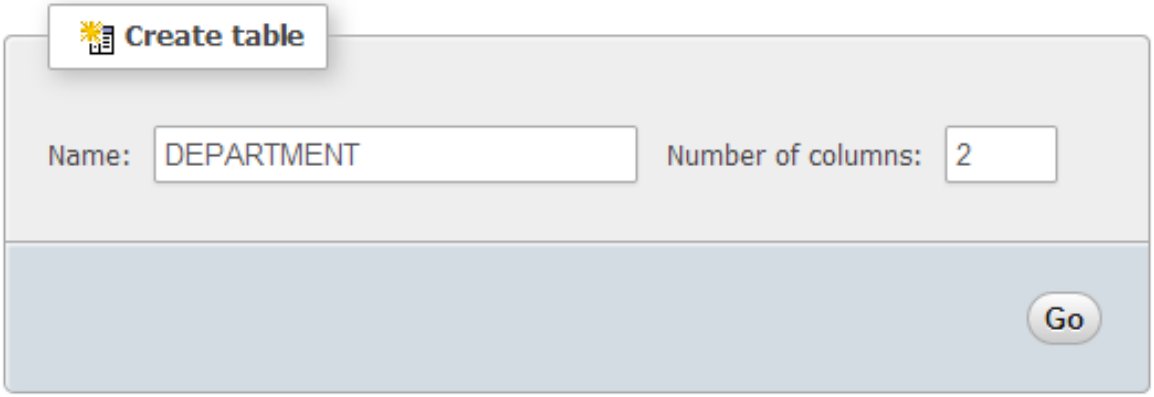
3 สร้างตารางเก็บข้อมูล (ต่อ)

- คำสั่งภาษา SQL ในการสร้างตาราง

```
CREATE TABLE Student (
        studentID       char(9)   NOT NULL,
        title           varchar(15)       NOT NULL,
        firstname       varchar(15)       NOT NULL,
        lastname        varchar(20)       NOT NULL,
        birthdate       date,
        gpa             float,
        major           char(4),
        PRIMARY KEY (studentID)
)
```

3 สร้างตารางเก็บข้อมูล (ต่อ)

- สร้างตาราง DEPARTMENT ประกอบด้วย 2 ฟิลด์



ชื่อฟิลด์	คุณสมบัติ	ความหมาย
deptCode	char (4)	ชื่อย่อภาควิชา เป็น Primary Key
deptName	Varchar(30)	ชื่อภาควิชา

3. สร้างตารางเก็บข้อมูล (ต่อ)

- สร้างฟิลด์

Table name: DEPARTMENT Add 1 column(s) Go

Structure

Name	Type	Length/Values	Default	Collation	Attributes	Null	Index	A_I	Comments
deptCode	CHAR	4	None			☐	PRIMARY	☐	ชื่อย่อภาควิชา
deptName	VARCHAR	30	None			☐	---	☐	ชื่อเต็มภาควิชา

Table comments: ข้อมูลภาควิชา

Storage Engine: InnoDB

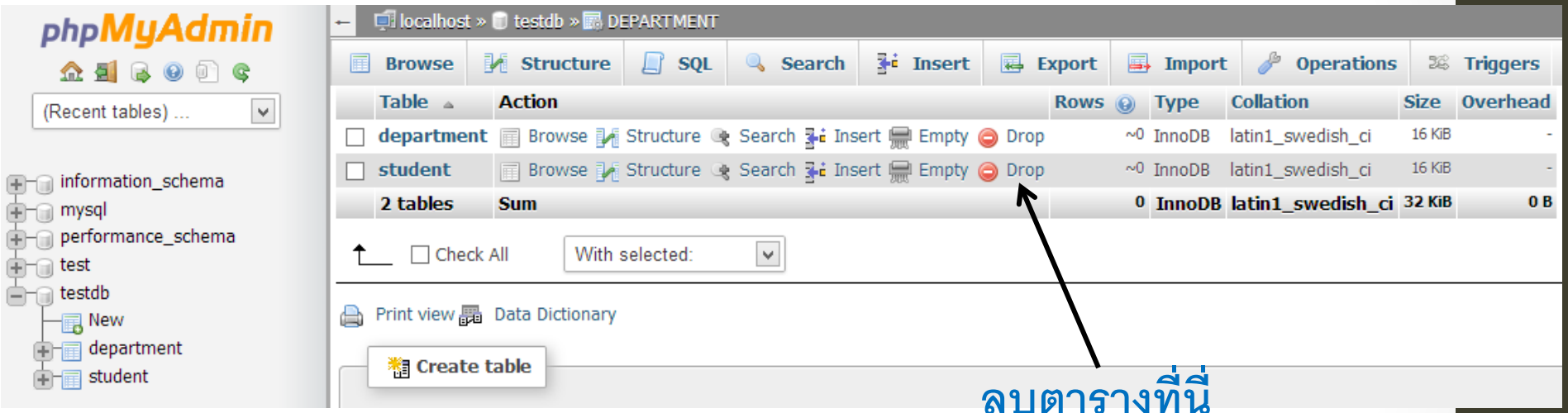
Collation:

PARTITION definition:

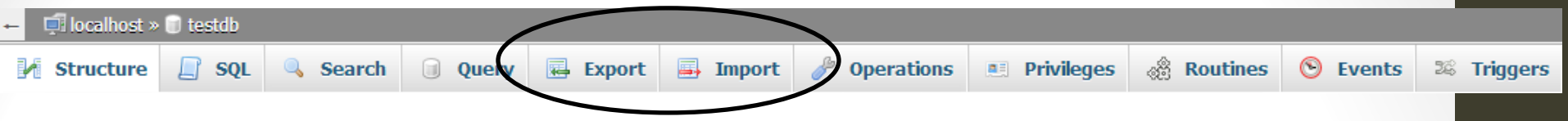
Save

3. สร้างตารางเก็บข้อมูล (ต่อ)

- ผลที่ได้และสถานะของฐานข้อมูล TestDB ปัจจุบัน



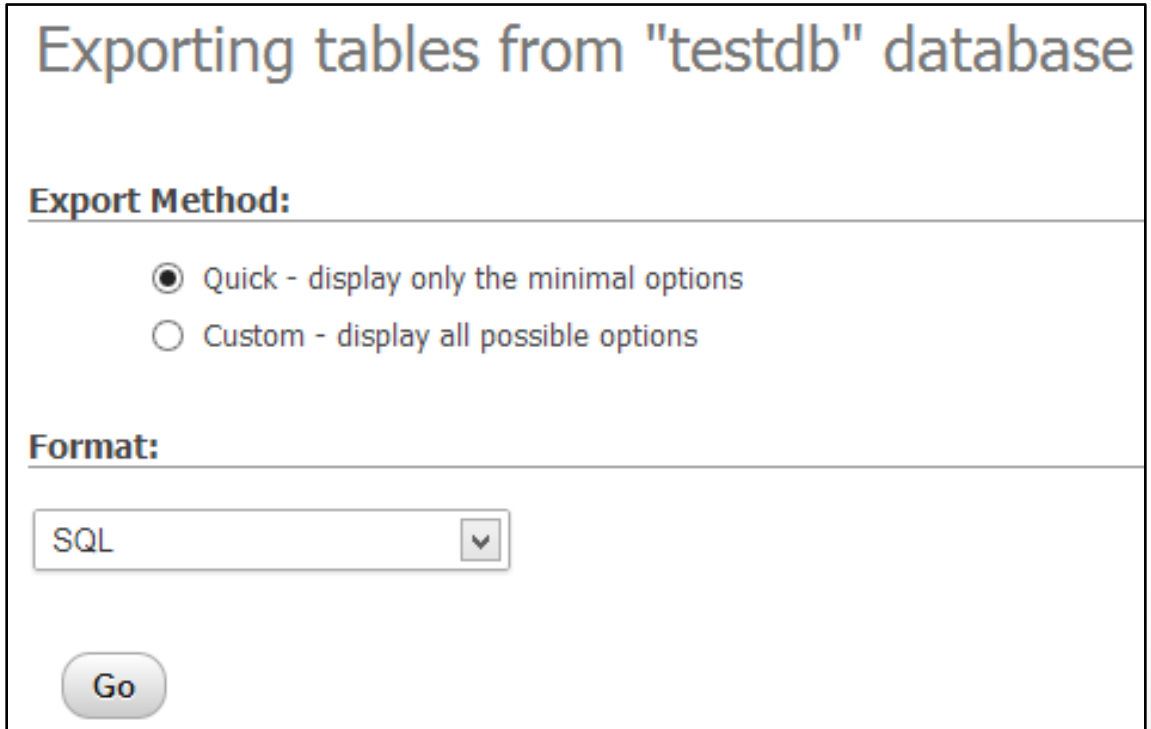
4. Backup & Restore



- Backup การสำรองฐานข้อมูล
  เก็บโครงสร้างและข้อมูลในตารางด้วย Export หรือส่งออก
- Restore การนำฐานข้อมูลที่สำรองไว้กลับมาใช้
  นำฐานข้อมูลที่เก็บสำรองไว้กลับมาติดตั้งใหม่ หรือ Import

4. Backup & Restore (ต่อ)

- การสำรองฐานข้อมูล เลือก Export
- เลือก Quick และ Go
- บันทึกไฟล์ที่ต้องการ เลือกแบบ Quick



4. Backup & Restore (ต่อ)

- **สิ่งที่ได้**

```
--
-- Database: `testdb`
--
CREATE DATABASE IF NOT EXISTS `testdb` DEFAULT CHARACTER SET latin1 COLLATE utf8_general_ci ;
USE `testdb`;

-- --------------------------------------------------------

--
-- Table structure for table `department`
--

CREATE TABLE IF NOT EXISTS `department` (
  `deptCode` char(4) NOT NULL COMMENT 'ชื่อย่อภาควิชา',
  `deptName` varchar(30) NOT NULL COMMENT 'ชื่อเต็มภาควิชา',
  PRIMARY KEY (`deptCode`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 COMMENT='ข้อมูลภาควิชา';

-- --------------------------------------------------------

--
-- Table structure for table `student`
--

CREATE TABLE IF NOT EXISTS `student` (
  `studentID` char(9) NOT NULL COMMENT 'รหัสประจำตัวนศ.',
  `firstname` varchar(15) NOT NULL COMMENT 'ชื่อนศ.',
  `lastname` varchar(20) NOT NULL COMMENT 'นามสกุลนศ.',
  `birthdate` date DEFAULT NULL COMMENT 'วันเกิด',
  `gpa` float DEFAULT NULL COMMENT 'เกรดเฉลี่ยสะสม',
  `major` char(4) DEFAULT NULL COMMENT 'ภาควิชาที่สังกัด',
  PRIMARY KEY (`studentID`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 COMMENT='ข้อมูลนศ.';
```

คำสั่งสร้างฐานข้อมูล

ระบุฐานข้อมูลที่ใช้

หากไม่ต้องการให้คำสั่งใดรันให้ใส่ comment → - -

คำสั่งสร้างตาราง

4. Backup & Restore (ต่อ)

- หากเลือกแบบ Custom

Exporting tables from "testdb" database

Export Method:

- (●) Quick - display only the minimal options
- ( ) Custom - display all possible options

Format:

SQL

Go

Table(s):

Select All / Unselect All

department
student

Format:

SQL

Format-specific options:

- [x] Display comments *(includes info such as export timestamp, PHP version, and server version)*
  - Additional custom header comment (\n splits lines): [ ]
  - [ ] Include a timestamp of when databases were created, last updated, and last checked
- [ ] Enclose export in a transaction
- [ ] Disable foreign key checks

Database system or older MySQL server to maximize output compatibility with: NONE

- ( ) structure
- ( ) data
- (●) structure and data

5. Structured Query Language (SQL)

- เป็นภาษาที่ใช้สำหรับบริหารจัดการข้อมูลในฐานข้อมูลเชิงสัมพันธ์ (Relational Database Management System: DBMS)

- มีมาตรฐาน ANSI 1986 เรียกว่า SQL-86 เป็นพื้นฐาน
  - The American National Standard Institute

- มีมาตรฐานอื่นเพิ่มเติม ได้แก่ SQL-89, SQL-92, SQL-99, SQL-2003, SQL-2006, SQL-2008, และ SQL-2011

5. Structured Query Language – SQL (ต่อ)

SELECT	Data retrieval
INSERT UPDATE DELETE	Data Manipulation Language (DML)
CREATE ALTER DROP RENAME TRUNCATE	Data Definition Language (DDL)

จะได้เรียนในวิชา **Fundamentals of Database System (204222)**

6. DESC ดูโครงสร้างตาราง

- หากต้องการทราบโครงสร้างตาราง DEPARTMENT

เรียกใช้ SQL

Structure | SQL | Search | More

Run SQL query/queries on database testdb:

```
DESC Department;
```

พิมพ์คำสั่ง

Clear

[Delimiter ;] Show this query here again Retain query box

Go

Show query box

Showing rows 0 - 1 (2 total, Query took 0.0156 sec)

DESC department

[Edit] [Create PHP Code]

Show : Start row: 0 Number of rows: 30 Headers every 100

+ Options

	Field	Type	Null	Key	Default	Extra
Edit Copy Delete	deptCode	char(4)	NO	PRI	NULL	
Edit Copy Delete	deptName	varchar(30)	NO		NULL	

Check All With selected: Change Delete

Show : Start row: 0 Number of rows: 30 Headers every 100

7. สร้างตารางด้วย SQL

- ใช้ Data Definition Language (DDL)
- CREATE, ALTER, RENAME, DROP, TRUNCATE
- สร้างตาราง

```
CREATE  TABLE  example (
        field1   INTEGER,
        field2   VARCHAR(50),
        field3   DATE  NOT  NULL,
        PRIMARY  KEY (field1, field2)
);
```

7. สร้างตารางด้วย SQL (ต่อ)

- **ดูโครงสร้างตารางที่สร้างขึ้น**

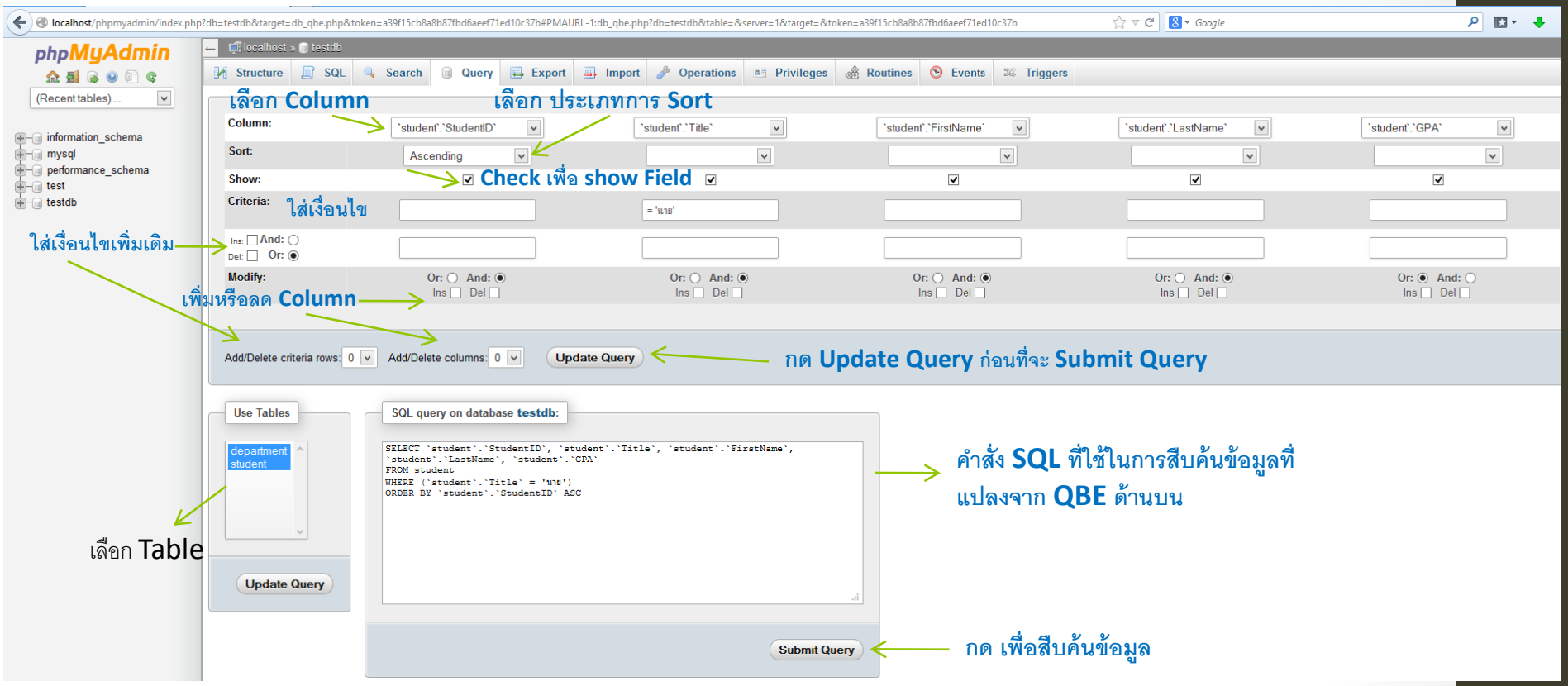
  DESC example

- **ลบตาราง**

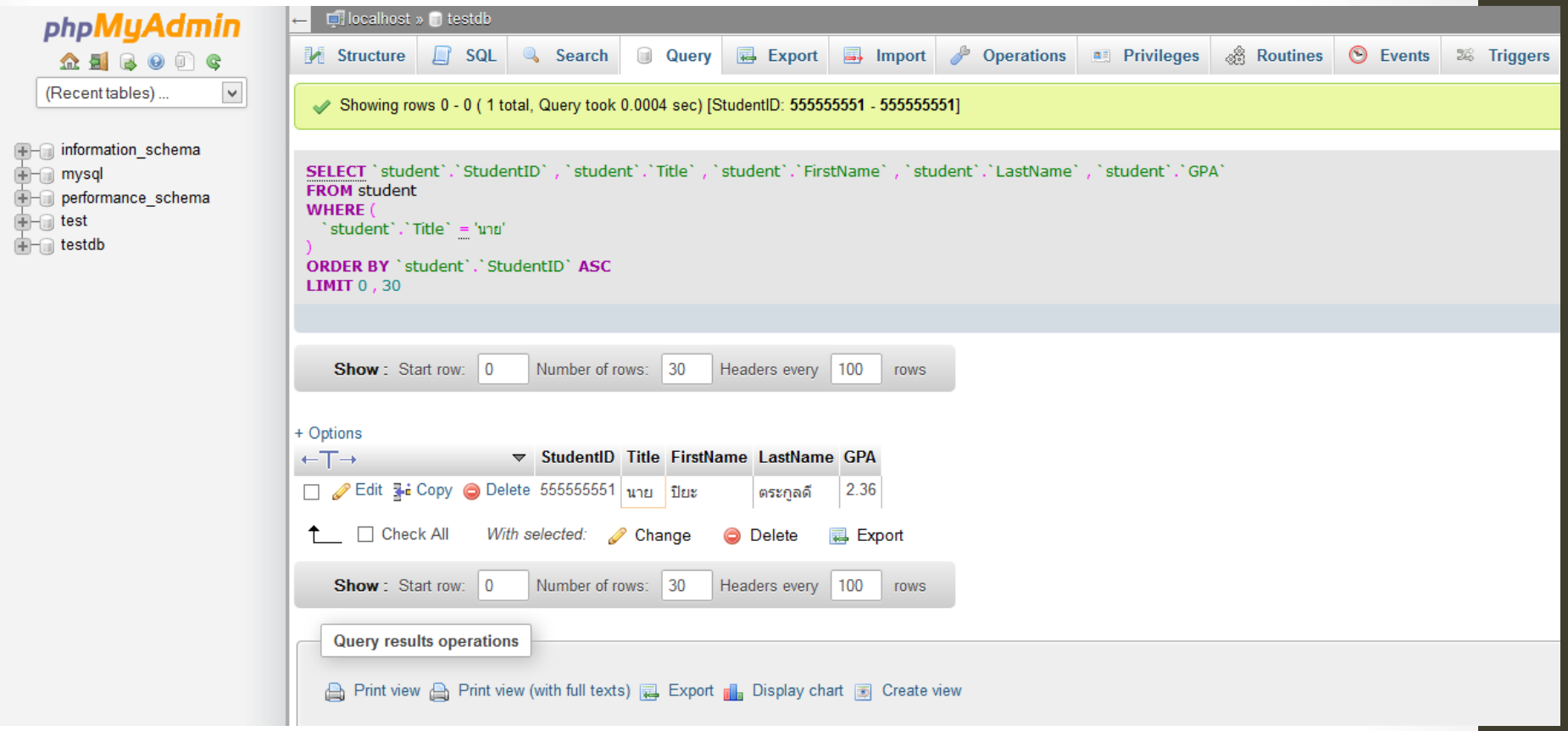
  DROP TABLE example

8. การสืบค้นข้อมูลโดยใช้ Query Tab

(Query By Example – QBE)



8. ผลลัพธ์จากการสืบค้นข้อมูลด้วย QBE



9. Maintaining Data using SQL Command - Insert

Insert Command

Approach 1

Insert into Table Name

Values ($data_1$, $data_2$, …, $data_n$)

Example: Insert Into Student

Values (‘555555555’, ‘นาย’, ‘วิษณุ’, ‘ชื่นบาน’, ‘1997-12-31’, 3.25, ‘SCCS’);

Approach 2

Insert into Table Name ($Field_1$, $Field_2$, …, $Field_n$)

Values ($data_1$, $data_2$, …, $data_n$)

Example: Insert Into Student (studentID, title, firstname, lastname)

Values (‘666666666’, ‘นางสาว’, ‘วิไลลักษณ์’, ‘สุขสงวน’,);

10. Maintaining Data using SQL Command - Update

Update Command

Update Table Name

Set Attribute Name = New Value

Where Condition; // ถ้าไม่มีเงื่อนไข จะ update ทุก record (ระวัง!!!)

Example:

Update Student

Set lastname = ‘สุขสำราญ’

Where studentID = ‘666666666’;

Maintaining Data using SQL Command - Delete

Delete Command

Delete From Table Name

Where Condition; // ถ้าไม่มีเงื่อนไข จะ delete ทุก record (ระวัง!!!)

Example:

Delete

From Student

Where studentID = ‘666666666’;

การสืบค้นข้อมูล (Query) - Select

Select Command

Select *
From Table Name; // มาทุก Record, ทุก Field

Select * From Student;

Select Attribute List
From Table Name; // มาทุก Record, เฉพาะ Field ที่ระบุหลัง Select

Select studentID, gpa From Student;

Select *
From Table Name;
Where Condition; // มาทุก Field แต่มาเฉพาะ Record ที่ตรงตามเงื่อนไข

Select * From Student Where gpa < 2.00;

Select Attribute List
From Table Name;
Where Condition; // มา เฉพาะ Field ที่ระบุหลัง Select และมาเฉพาะ Record ที่ตรงตามเงื่อนไข

Select studentID, title, firstname, lastname, gpa
From Student
Where gpa >= 3.50;

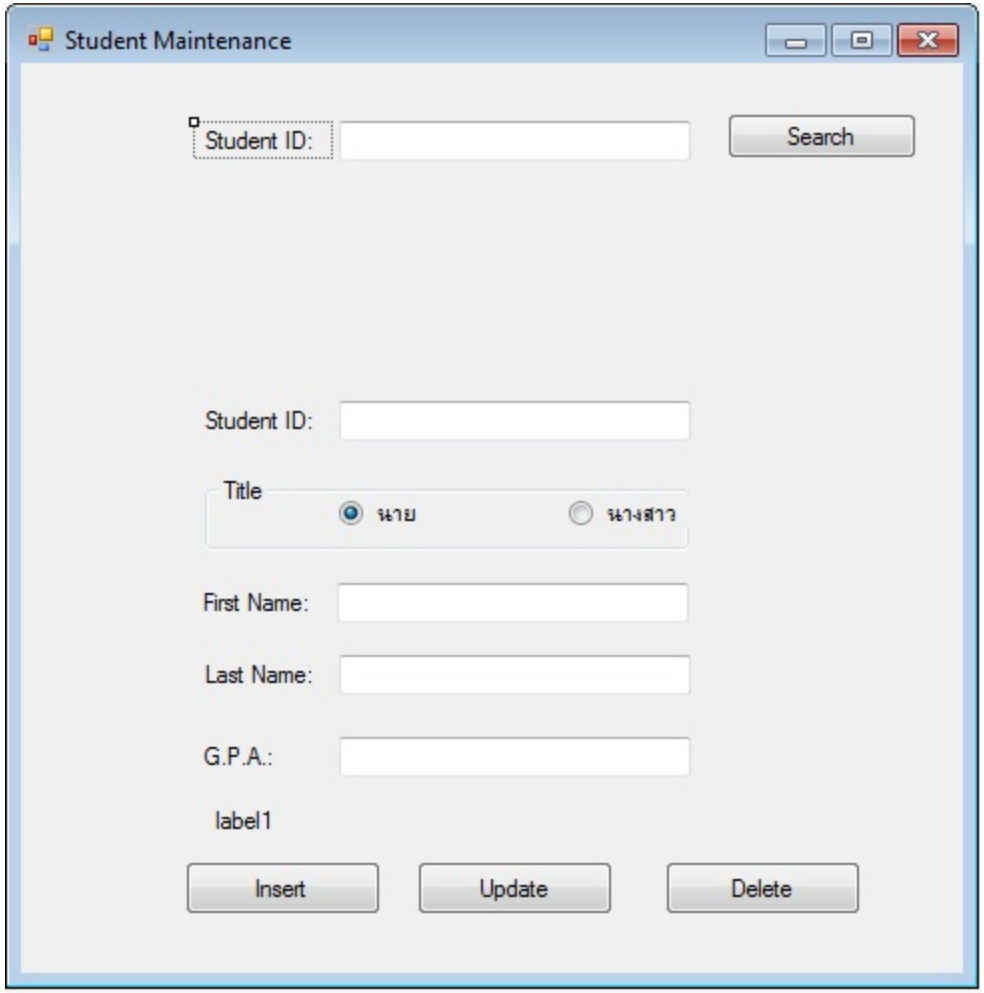
Downloading Connector/Net

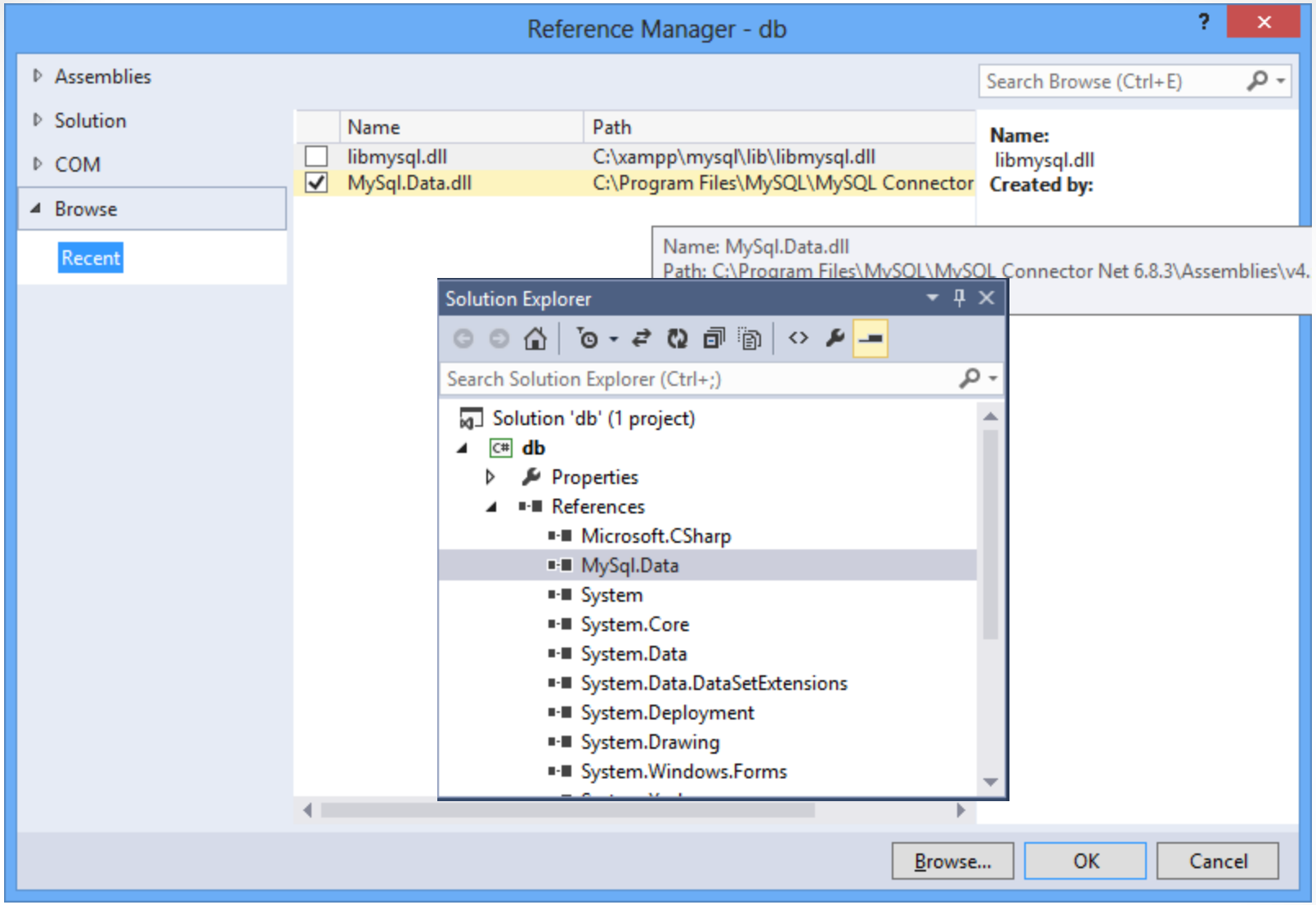
- First make sure you have downloaded and installed the MySQL Connector/NET (Database Driver) from the MySQL

  http://dev.mysql.com/downloads/connector/net/6.1.html

  mysql-connector-net-6.8.3.msi

สร้าง Form


Student Maintenance
Student ID:
Search
Student ID:
Title
นาย
นางสาว
First Name:
Last Name:
G.P.A.:
label1
Insert
Update
Delete



- ตัวอย่าง Browse ไปที่ C:\Program Files\MySQL\MySQL Connector Net 6.8.3\Assemblies\v4.5

```
//Add MySql Library
using MySql.Data.MySqlClient;
```

Then declaring and initializing the variables that we will use:

- connection: will be used to open a connection to the database.
- server: indicates where our server is hosted, in our case, it's localhost.
- database: is the name of the database we will use, in our case it's the database we already created earlier which is connectcsharptomysql.
- uid: is our MySQL username.
- password: is our MySQL password.
- connectionString: contains the connection string to connect to the database, and will be assigned to the connection variable.

```
private MySqlConnection connection;
private string server;
private string database;
private string uid;
private string password;
private string title_value;
public Form1()
{
     InitializeComponent();
     server = "localhost";
     database = "testdb";
     uid = "root";
     password = "";
     string connectionString;
     connectionString = "SERVER=" + server + ";" + "DATABASE=" +
                         database + ";" + "UID=" + uid + ";" +
                        "PASSWORD=" +password+ ";charset=utf8;";
     connection = new MySqlConnection(connectionString);
}
```

Working with Insert, Update, Select, Delete

Usually, Insert, update and delete are used to write or change data in the database, while Select is used to read data.
For this reason, we have different types of methods to execute those queries.

The methods are the following:

- ExecuteNonQuery: Used to execute a command that will not return any data, for example Insert, update or delete.
- ExecuteReader: Used to execute a command that will return 0 or more records, for example Select.
- ExecuteScalar: Used to execute a command that will return only 1 value, for example Select Count(*).

```
private bool OpenConnection()
{
     try
     {
         connection.Open();
         return true;
     }
     catch (MySqlException ex)
     {
         //When handling errors, you can your application's response based on the error number.
         //The two most common error numbers when connecting are as follows:
         //0: Cannot connect to server.
         //1045: Invalid user name and/or password.
         switch (ex.Number)
         {
             case 0:
                 MessageBox.Show("Cannot connect to server.  Contact administrator");
                 break;
             case 1045:
                 MessageBox.Show("Invalid username/password, please try again");
                 break;
         }
         return false;
     }
 }
```

```
private bool CloseConnection()
{
    try
    {
        connection.Close();
        return true;
    }
    catch (MySqlException ex)
    {
        MessageBox.Show(ex.Message);
        return false;
    }
}
```

C# Programming

Steps

Start with Insert, update and delete, which are the easiest. The process to successfully execute a command is as follows:

1. Open connection to the database.
2. Create a MySQL command.
3. Assign a connection and a query to the command. This can be done using the constructor, or using theConnection and the CommandText methods in the MySqlCommand class.
4. Execute the command.
5. Close the connection.

```
private void insert_Click (object sender, EventArgs e)
{
     //open connection
     if (this.OpenConnection() == true)
     {
        if (male.Checked)
           title_value = "zëǎ";
        else
           title_value = "zëhçëĉ";
        string query = "INSERT INTO Student (studentID, title, firstName,
                        lastName, gpa) VALUES('" + studentID.Text + "', '"
                       + title_value + "', '" + firstName.Text + "', '"
                       + lastName.Text + "', " + double.Parse(gpa.Text)
                       + ");";
        //stringQuery.Text = query;
        //create command and assign the query and connection from the constructor
        MySqlCommand cmd = new MySqlCommand(query, connection);
        //Execute command
        cmd.ExecuteNonQuery();
        //close connection
        this.CloseConnection();
     }
}
```

Steps for select

To execute a Select statement, we add a few more steps, and we use the ExecuteReader method that will return a dataReader object to read and store the data or records.

1. Open connection to the database.
2. Create a MySQL command.
3. Assign a connection and a query to the command. This can be done using the constructor, or using theConnection and the CommandText methods in the MySqlCommand class.
4. Create a MySqlDataReader object to read the selected records/data.
5. Execute the command.
6. Read the records and display them or store them in a list.
7. Close the data reader.
8. Close the connection.

```
private void search_Click(object sender, EventArgs e)
{
    string query = "SELECT * FROM Student WHERE studentID = " + studentID1.Text + ";";
    //Open connection
    if (this.OpenConnection() == true)
    {
        //Create Command
        MySqlCommand cmd = new MySqlCommand(query, connection);
        //Create a data reader and Execute the command
        MySqlDataReader dataReader = cmd.ExecuteReader();
        //Read the data and store them in the list
        while (dataReader.Read())
        {
            studentID.Text = dataReader["StudentID"].ToString();
            if (dataReader["title"].ToString() == "zëå")
                male.Checked = true;
            else
                female.Checked = true;
            firstName.Text = dataReader["firstname"].ToString();
            lastName.Text = dataReader["lastname"].ToString();
            gpa.Text = dataReader["gpa"].ToString();
        }
        //close Data Reader
        dataReader.Close();
        //close Connection
        this.CloseConnection();
    }
  }
```

```
private void update_Click(object sender, EventArgs e)
{
     //Open connection
     if (this.OpenConnection() == true)
     {
        if (male.Checked) title_value = "zëå";
        else title_value = "zëhçëĉ";
        string query = "UPDATE Student SET title='" + title_value +
                       "', firstname = '" + firstName.Text +
                       "', lastname = '" + lastName.Text + "', gpa=" +
                       gpa.Text +
                       " WHERE studentID=" + studentID.Text + ";";
        //stringQuery.Text = query;
        //create mysql command

        MySqlCommand cmd = new MySqlCommand();
        //Assign the query using CommandText
        cmd.CommandText = query;
        //Assign the connection using Connection
        cmd.Connection = connection;
        //Execute query
        cmd.ExecuteNonQuery();
        //close connection
        this.CloseConnection();
     }
 }
```

```
private void delete_Click(object sender, EventArgs e)
{
    string query = "DELETE FROM Student WHERE studentID = "+studentID.Text;

    if (this.OpenConnection() == true)
    {
        MySqlCommand cmd = new MySqlCommand(query, connection);
        cmd.ExecuteNonQuery();
        this.CloseConnection();
    }
}
```