บทที่ 19
ฐานข้อมูล MySQL

วัตถุประสงค์ของการศึกษา

1. เพื่อให้ทราบและเข้าใจถึงฐานข้อมูล MySQL
2. เพื่อให้ทราบถึงลักษณะของฐานข้อมูล
3. เพื่อให้ทราบถึงชนิดข้อมูลของคอลัมน์
4. เพื่อให้ทราบคำสั่ง SQL ต่าง ๆ และสามารถใช้คำสั่ง SQL จัดการกับฐานข้อมูล MySQL ได้

เนื้อหาในบทนี้ประกอบด้วย

- ฐานข้อมูล MySQL
- ลักษณะของฐานข้อมูล
- ชนิดข้อมูลของคอลัมน์
- คำสั่งเกี่ยวกับการจัดการฐานข้อมูล MySQL

ฐานข้อมูล MySQL

เนื่องจากในปัจจุบันแทบทุกหน่วยงานจำเป็นต้องมีการจัดเก็บข้อมูลบางอย่างเอาไว้ ซึ่งถ้าจะจัดเก็บในรูปแบบเอกสารอาจจะไม่สะดวกต่อการใช้งาน ถ้าข้อมูลมีปริมาณมาก ๆ โอกาสที่จะเกิดข้อผิดพลาดก็จะมีสูงตามไปด้วย ดังนั้นในปัจจุบันการจัดเก็บข้อมูลส่วนใหญ่จึงใช้โปรแกรมที่เรียกว่า ฐานข้อมูล (Database) ในการจัดเก็บข้อมูล ซึ่งปัจจุบันมีโปรแกรมฐานข้อมูลอยู่เป็นจำนวนมาก เช่น Oracle, MS SQL Server, Informix, Sysbase, DB/2, Access, MySQL เป็นต้น โปรแกรมเหล่านี้ส่วนใหญ่ถ้าจะใช้งานให้ถูกต้องจริง ๆ นั้นจะต้องเสียเงินซื้อซึ่งมีราคาค่อนข้างสูง แต่ก็มีบางตัวที่สามารถใช้งานได้ฟรี เช่น MySQL แต่อย่างไรก็ตามในปัจจุบัน MySQL ได้กลายเป็นโปรแกรมฐานข้อมูลแบบกึ่งพาณิชย์ไปแล้ว เนื่องจากมีทั้งแบบใช้งานได้ฟรี (Free License)และแบบที่ต้องเสียเงินซื้อ (Commercial License)

สาเหตุที่ต้องพูดถึง MySQL ก็เพราะว่าเป็นโปรแกรมฐานข้อมูลที่ภาษา PHP นิยมใช้งานร่วมมากที่สุด ทั้งนี้เนื่องจากเป็นฐานข้อมูลขนาดกลางที่มีประสิทธิภาพในการทำงานสูง นอกจากนี้ยังเป็นฐานข้อมูลที่ถูกสร้างขึ้นมาเพื่อรองรับการทำงานบนอินเทอร์เน็ตโดยเฉพาะ ในภาษา PHP เองก็ได้จัดเตรียมฟังก์ชั่นต่าง ๆ ในการจัดการฐานข้อมูล MySQL ไว้มากมาย และการติดต่อไปยังฐานข้อมูล MySQL จะใช้วิธีการติดต่อโดยตรงผ่านทางฟังก์ชั่นที่จัดเตรียมไว้ ทำให้สามารถทำงานได้รวดเร็วและมีประสิทธิภาพ

MySQL ถูกพัฒนาขึ้นในปี ค.ศ.1995 โดยกลุ่มโปรแกรมเมอร์ชาวสวีเดน โดยในเวอร์ชั่นแรก ๆ MySQL ยังไม่มีความสามารถที่โดดเด่นมากนัก แต่ในเวอร์ชั่นต่อ ๆ มาได้มีการปรับปรุงแก้ไขและเพิ่มเติมความสามารถใหม่ ๆ เข้าไปเรื่อย ๆ และเมื่อ PHP ได้มีฟังก์ชั่นในการจัดการและเชื่อมต่อไปยังฐานข้อมูล MySQL โดยเฉพาะ จึงทำให้มีผู้หันมาให้ความสนใจ MySQL กันอย่างกว้างขวาง และในเวลาต่อมา PHP และ MySQL ได้กลายเป็นของคู่กันแทบจะแยกกันไม่ออก เพราะโดยส่วนใหญ่ผู้ที่ศึกษา PHP ก็มักจะศึกษา MySQL ควบคู่กันไปด้วย ทั้ง ๆ ที่ PHP นั้นสามารถใช้งานร่วมกับฐานข้อมูลอื่น ๆ ได้เกือบทั้งหมด และ MySQL ก็สามารถใช้งานร่วมกับภาษาคอมพิวเตอร์อื่น ๆ ได้หลายภาษา แต่ก็ไม่ได้รับความนิยมเหมือนกับการใช้งานร่วมกันระหว่าง PHP และ MySQL

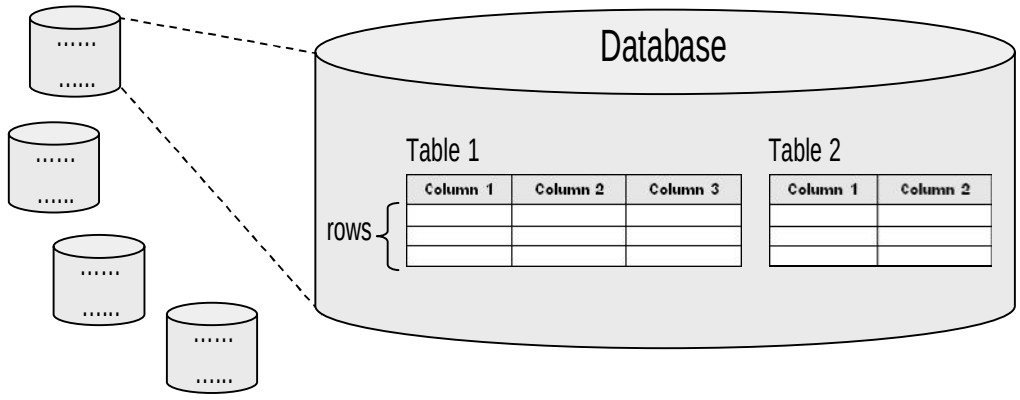
ลักษณะของฐานข้อมูล

โดยทั่วไปแล้วฐานข้อมูล MySQL จะมีส่วนประกอบหลัก ๆ ดังต่อไปนี้

1. ฐานข้อมูล (Database)
2. ตาราง (Table)
3. คอลัมน์ (Column)
4. แถวข้อมูล (Row)

ในการที่จะจัดเก็บข้อมูลลงใน MySQL ได้ จะต้องเริ่มต้นที่การสร้างฐานข้อมูลก่อน ลำดับถัดไปคือสร้างตาราง โดยฐานข้อมูลหนึ่ง ๆ จะประกอบไปด้วยตารางหลายตารางได้ ภายในตารางจะประกอบไปด้วยคอลัมน์ต่าง ๆ เช่น ตารางที่เก็บข้อมูลสินค้า จะมีคอลัมน์ รหัสสินค้า ชื่อสินค้า ราคา และประเภทสินค้า เป็นต้น ส่วนข้อมูลสินค้าที่จะนำเข้าไปเก็บในตารางจะกลายเป็นแถวข้อมูล เช่น มีสินค้าเก็บอยู่ 3 รายการ ก็แสดงว่ามี 3 แถวข้อมูล เป็นต้น ดังรูปดังต่อไปนี้

ฐานข้อมูล MySQL



ชนิดข้อมูลของคอลัมน์

จากที่ได้ทราบมาแล้วว่าตารางจะประกอบไปด้วยคอลัมน์สำหรับเก็บข้อมูล ซึ่งข้อมูลเหล่านี้อาจจะมีมากมายหลายชนิดแตกต่างกันออกไป เช่น ถ้าเป็นชื่อสินค้าจะเป็นข้อมูลชนิดสตริง ถ้าเป็นราคาสินค้าจะเป็นชนิดตัวเลข และถ้าเป็นวันเดือนปีจะเป็นข้อมูลชนิดวันเวลา เป็นต้น ทั้งนี้เนื่องจากเรามักจะมีวิธีในการจัดการกับข้อมูลเหล่านี้ในลักษณะที่แตกต่างกันออกไปตามลักษณะของข้อมูล ดังนั้นทุกคอลัมน์ของตารางจะต้องทำการกำหนดชนิดข้อมูลเสมอ สำหรับ MySQL ได้แบ่งชนิดข้อมูลของคอลัมน์ออกเป็นดังนี้

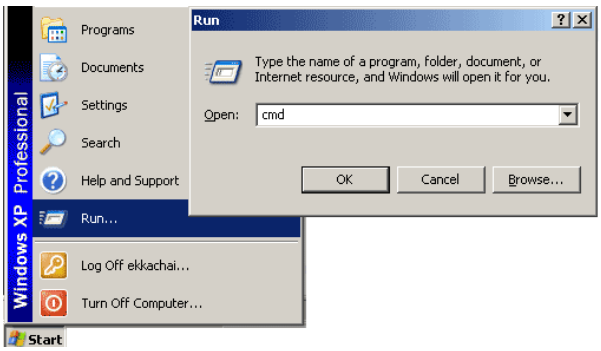
ชนิดข้อมูล	คำอธิบาย
ข้อมูลชนิดสตริง	
CHAR	ข้อมูลชนิดสตริง ความยาวไม่เกิน 255
VARCHAR	ข้อมูลชนิดสตริง ความยาวไม่เกิน 255
TINYTEXT	ข้อมูลชนิดสตริง ความยาวไม่เกิน 255
TEXT	ข้อมูลชนิดสตริง ความยาวไม่เกิน 65535
MEDIUMTEXT	ข้อมูลชนิดสตริง ความยาวไม่เกิน 16777215
LONGTEXT	ข้อมูลชนิดสตริง ความยาวไม่เกิน 4294967295
ข้อมูลชนิดตัวเลข	
TINYINT	ข้อมูลชนิดจำนวนเต็ม ระหว่าง 0 - 255
SMALLINT	ข้อมูลชนิดจำนวนเต็ม ระหว่าง 0 - 65535
MEDIUMINT	ข้อมูลชนิดจำนวนเต็ม ระหว่าง 0 - 16777215
INT	ข้อมูลชนิดจำนวนเต็ม ระหว่าง 0 - 4294967295
BIGINT	ข้อมูลชนิดจำนวนเต็ม ระหว่าง 0 - 18446744073709551615
FLOAT	ข้อมูลชนิดทศนิยม
DOUBLE	ข้อมูลชนิดทศนิยม

ข้อมูลชนิดวันเวลา	
DATE	เก็บข้อมูล วันเดือนปี ระหว่าง 1000-01-01 ถึง 9999-12-31
TIME	เก็บข้อมูล เวลา ระหว่าง -838:59:59 ถึง 838:59:59
DATETIME	เก็บข้อมูล วันเดือนปีและเวลา ระหว่าง 1000-01-01 00:00:00 ถึง 9999-12-31 23:59:59
ข้อมูลชนิด BLOB (Binary Large Object)	
TINYBLOB	ใช้เก็บข้อมูลประเภทรูปภาพหรือมัลติมีเดียที่มีรูปแบบข้อมูลเป็น Binary ขนาดไม่เกิน 255 ไบต์
BLOB	ใช้เก็บข้อมูลประเภทรูปภาพหรือมัลติมีเดียที่มีรูปแบบข้อมูลเป็น Binary ขนาดไม่เกิน 65535 ไบต์
MEDIUMBLOB	ใช้เก็บข้อมูลประเภทรูปภาพหรือมัลติมีเดียที่มีรูปแบบข้อมูลเป็น Binary ขนาดไม่เกิน 16777215 ไบต์
LONGBLOB	ใช้เก็บข้อมูลประเภทรูปภาพหรือมัลติมีเดียที่มีรูปแบบข้อมูลเป็น Binary ขนาดไม่เกิน 4294967295 ไบต์
ข้อมูลชนิด set และ enum	
SET	ใช้เก็บข้อมูลเป็นกลุ่ม รูปแบบเช่น SET(“member1”, “member2”, ..., “member N”) มีสมาชิกได้ไม่เกิน 64 ตัว
ENUM	ใช้เก็บข้อมูลเป็นกลุ่ม รูปแบบเช่น SET(“member1”, “member2”, ..., “member N”) มีสมาชิกได้ไม่เกิน 65535 ตัว

คำสั่งเกี่ยวกับการจัดการฐานข้อมูล MySQL

เมื่อได้ทำการติดตั้ง MySQL เรียบร้อยแล้ว (ได้ทำการติดตั้งไปพร้อมกับโปรแกรม AppServ แล้วในบทที่ 2) จากนั้นต้องทำการทดสอบการใช้งาน โดยในบทนี้จะเป็นการทดสอบการจัดการฐานข้อมูล MySQL ในลักษณะที่ใช้คำสั่งจัดการโดยตรง (Command Line) ไปยังฐานข้อมูล ผ่านทาง Dos ของระบบปฏิบัติการวินโดวส์ โดยมีขั้นตอนดังนี้

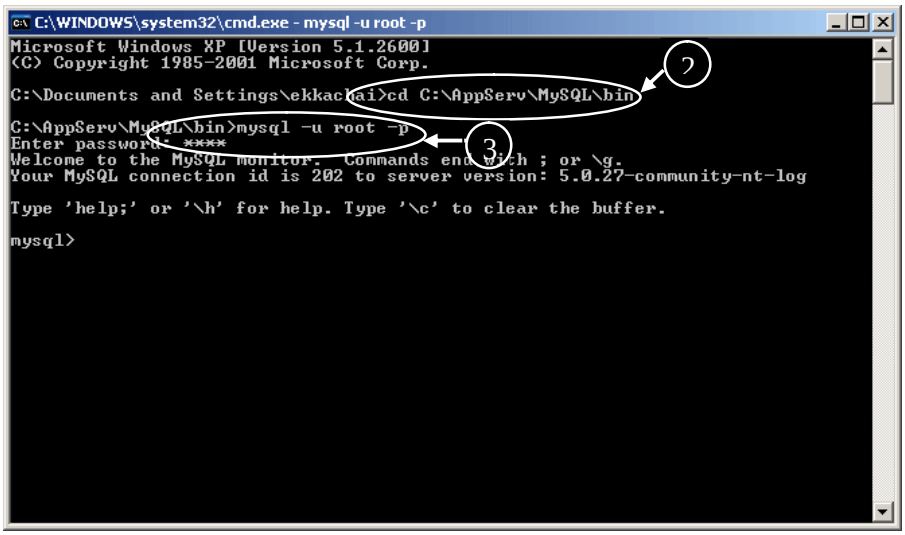
1. คลิ๊กที่ปุ่ม Start ไปที่ Run แล้วพิมพ์ cmd ตอบ OK ดังรูป



2. เข้าไปยังไดเร็กทอรี่ของ MySQL โดยพิมพ์คำสั่ง cd C:\AppServ\MySQL\bin

3. จากนั้น Login เข้าสู่ MySQL โดยพิมพ์คำสั่ง mysql –u root –p กดปุ่ม Enter แล้วใส่รหัสผ่าน ในที่นี้เราได้กำหนดรหัสผ่านของ root ไว้ตั้งแต่ตอนติดตั้งคือ 1234 ถ้าเข้าสู่ระบบได้จะแสดงดังรูป

หมายเหตุ : ก่อนเข้าสู่ระบบควรตรวจสอบดูก่อนว่า MySQL ได้ Start พร้อมใช้งานหรือไม่



ตารางสรุปคำสั่ง SQL ต่าง ๆ

คำสั่ง	คำอธิบาย
SHOW DATABASES;	แสดงรายชื่อฐานข้อมูลทั้งหมด
USE ชื่อฐานข้อมูล;	เลือกใช้ฐานข้อมูลที่จะทำงานด้วย
SHOW TABLES;	แสดงรายชื่อตารางทั้งหมดภายในฐานข้อมูล
SHOW COLUMNS FROM ชื่อตาราง;	แสดงชื่อฟิลด์และรายละเอียดของตาราง
CREATE DATABASE ชื่อฐานข้อมูล;	สร้างฐานข้อมูลใหม่
DROP DATABASE ชื่อฐานข้อมูล;	ลบฐานข้อมูล
CREATE TABLE (ชื่อฟิลด์ และรายละเอียดต่างๆ); เช่น CREATE TABLE customer(id int(4), name varchar(20) not null, address varchar(50) not null, primary key(id));	สร้างตารางใหม่
DROP TABLE ชื่อตาราง;	ลบตาราง

INSERT INTO ชื่อตาราง(ชื่อฟิลด์) VALUES(ค่าของฟิลด์); เช่น INSERT INTO customer VALUES(1111, 'Mr.Charnchai', 'U of Mahasarakham');	เพิ่มข้อมูลลงไปในตาราง
SELECT ชื่อฟิลด์ FROM ชื่อตาราง [WHERE เงื่อนไข]; เช่น SELECT id,name,address from customer; หรือ SELECT * FROM customer;	ดึงข้อมูลจากตารางที่กำหนด และตามเงื่อนไขที่ต้องการ
UPDATE ชื่อตาราง SET ชื่อฟิลด์ = ค่าที่แก้ไข [WHERE เงื่อนไข]; เช่น UPDATE customer SET name="Mr.Charnchai S.", address="Mahasarakharm University" WHERE id=1111;	แก้ไขข้อมูลในตารางตามเงื่อนไขที่ต้องการ
DELETE FROM ชื่อตาราง WHERE เงื่อนไข; เช่น DELETE FROM customer WHERE id = 1111;	ลบข้อมูลจากตารางที่ระบุ และตามเงื่อนไขที่ต้องการ
ALTER TABLE ชื่อตารางเดิม RENAME TO ชื่อตารางใหม่	เปลี่ยนชื่อตาราง
ALTER TABLE ชื่อตาราง DROP	ลบคอลัมน์
ALTER TABLE ชื่อตาราง ADD	เพิ่มคอลัมน์ใหม่ในตาราง
ORDER BY ASC	เรียงลำดับข้อมูล จากน้อยไปมาก
ORDER BY DESC	เรียงลำดับข้อมูล จากมากไปน้อย
LIMIT	
EXIT;	ออกจากการทำงาน

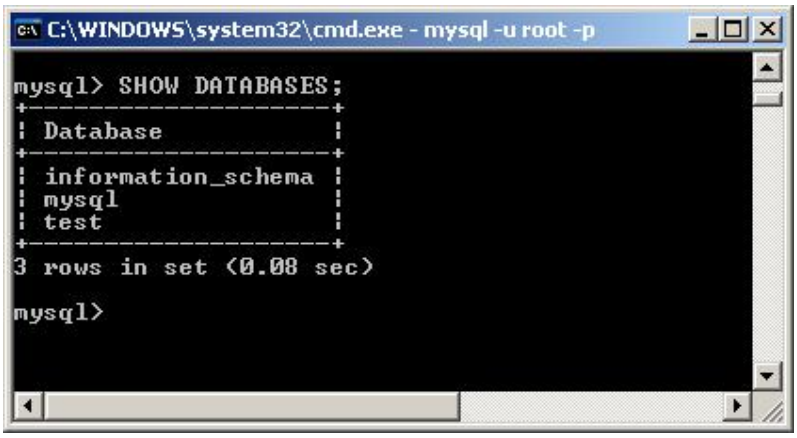
คำสั่ง Show Databases;

เป็นคำสั่งแสดงรายชื่อฐานข้อมูลทั้งหมดที่อยู่ใน MySQL

รูปแบบของคำสั่ง

```
SHOW DATABASES;
```

ผลลัพธ์ที่ได้



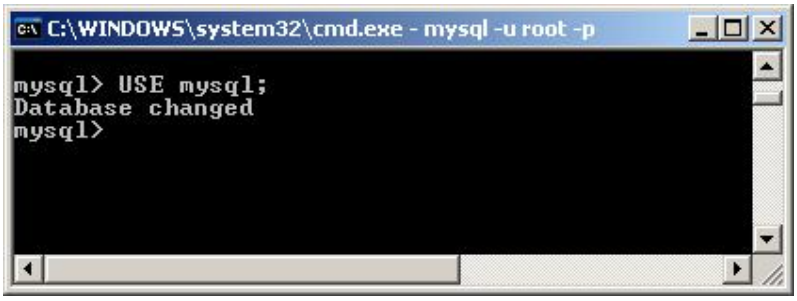
คำสั่ง Use;

ก่อนที่จะเริ่มต้นจัดการกับฐานข้อมูลใด ต้องใช้คำสั่ง Use เพื่อเลือกใช้ฐานข้อมูลที่จะทำงานด้วยก่อนเสมอ

รูปแบบของคำสั่ง

```
USE ชื่อฐานข้อมูล;
```

ผลลัพธ์ที่ได้



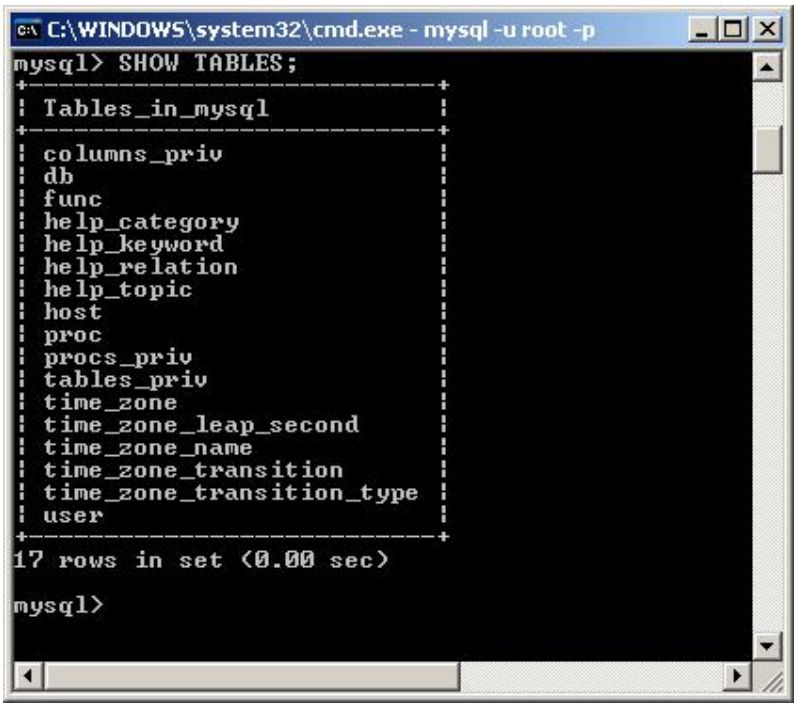
คำสั่ง Show Tables;

เป็นคำสั่งแสดงรายชื่อตารางทั้งหมดที่อยู่ในฐานข้อมูลที่เราเลือกใช้ จากคำสั่งที่ผ่านมาเราได้เลือกใช้ฐานข้อมูลชื่อ mysql ไปแล้วโดยใช้คำสั่ง USE mysql; ดังนั้นคำสั่งนี้จะแสดงรายชื่อตารางทั้งหมดของฐานข้อมูลชื่อ mysql

รูปแบบของคำสั่ง

```
SHOW TABLES;
```

ผลลัพธ์ที่ได้



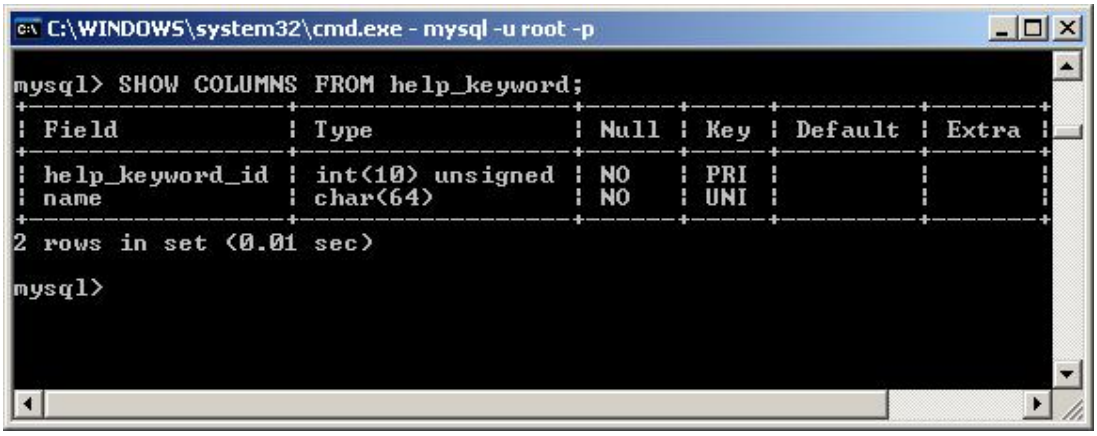
คำสั่ง Show Columns;

เป็นคำสั่งแสดงชื่อคอลัมน์หรือชื่อฟิลด์ พร้อมด้วยรายละเอียดอย่างอื่น เช่น ชนิดข้อมูล ความยาว คีย์หลัก เป็นต้น ทั้งนี้ต้องกำหนดด้วยว่าจะดูคอลัมน์ของตารางใด

รูปแบบของคำสั่ง

```
SHOW COLUMNS FROM ชื่อตาราง;
```

ผลลัพธ์ที่ได้



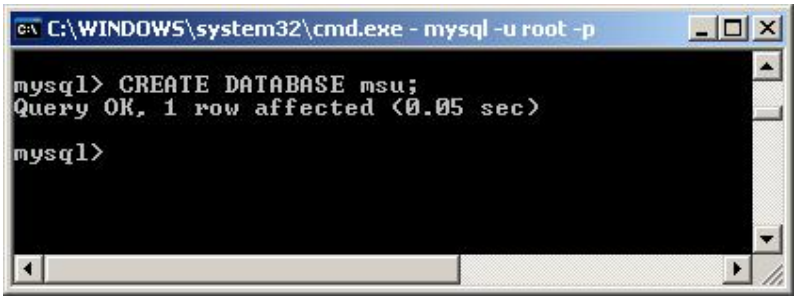
คำสั่ง Create Database;

เป็นคำสั่งในการสร้างฐานข้อมูลขึ้นมาใหม่

รูปแบบของคำสั่ง

```
CREATE DATABASE ชื่อฐานข้อมูล;
```

ตัวอย่างเช่น



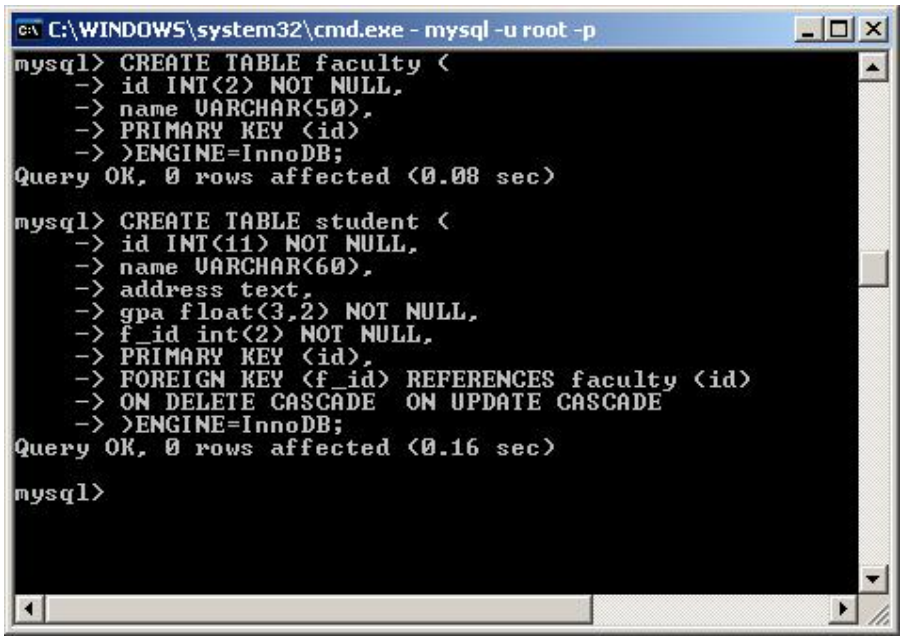
คำสั่ง Create Table;

เป็นคำสั่งในการสร้างตารางขึ้นมาใหม่ ทั้งนี้เราจะสร้างตารางในฐานข้อมูลใดต้องเข้าไปที่ฐานข้อมูลตัวนั้นก่อน โดยใช้คำสั่ง USE เช่น USE msu; แล้วจึงทำคำสั่งในการสร้างตาราง

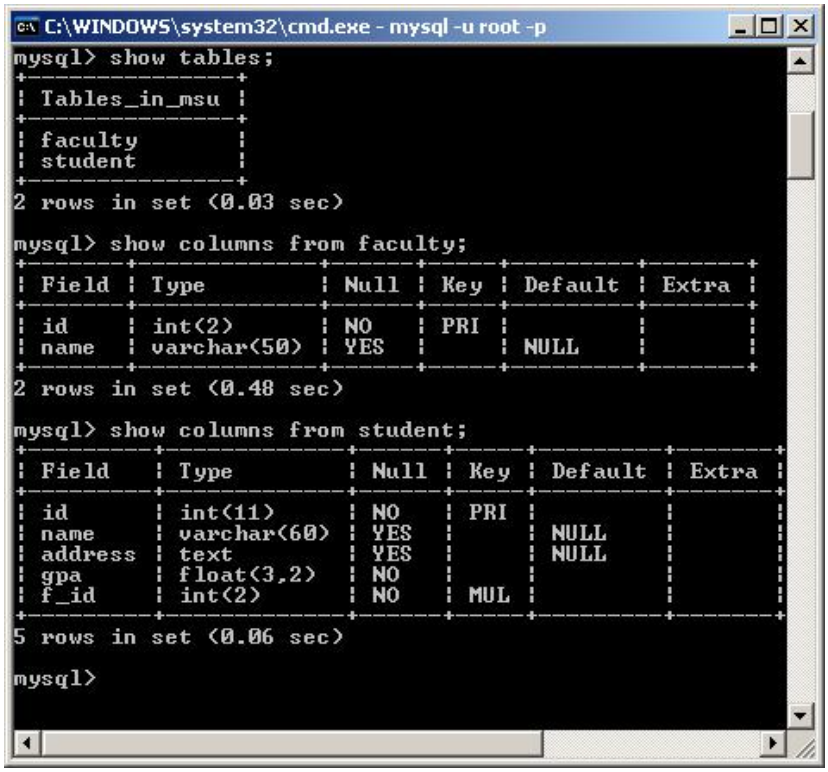
รูปแบบของคำสั่ง

```
CREATE TABLE ชื่อตาราง (
   ชื่อคอลัมน์ที่_1  ชนิดข้อมูล  [แอตทริบิวต์ต่าง ๆ] ,
   ชื่อคอลัมน์ที่_2  ชนิดข้อมูล  [แอตทริบิวต์ต่าง ๆ] ,
   ……..
   ชื่อคอลัมน์ที่_n  ชนิดข้อมูล  [แอตทริบิวต์ต่าง ๆ] ,
   PRIMARY KEY (ชื่อคอลัมน์) ,
   FOREIGN KEY (ชื่อคอลัมน์) REFERENCES ชื่อตารางที่อ้างอิง (ชื่อคอลัมน์ที่อ้างอิง)
   ON DELETE {CASCADE หรือ RESTRICT}
   ON UPDATE {CASCADE หรือ RESTRICT}
) ENGINE=InnoDB ;
```

ตัวอย่างเช่น



จากตัวอย่างนี้ทำการสร้างตาราง 2 ตาราง ชื่อ faculty เพื่อเก็บข้อมูลคณะ และชื่อ student เพื่อเก็บข้อมูลนิสิต ซึ่งทั้ง 2 ตารางมีความเกี่ยวข้องกัน บนพื้นฐานที่ว่านิสิตทุกคนต้องมีคณะที่ตนเองสังกัดอยู่ ดังนั้นในตาราง student จึงมีคอลัมน์หนึ่งชื่อ f_id เป็น Foreign Key (คีย์นอก) เชื่อมโยงกับคอลัมน์ id ในตาราง faculty เมื่อสร้างเสร็จอาจตรวจสอบโดยใช้คำสั่ง Show Tables; เพื่อดูชื่อตาราง และคำสั่ง Show Columns From faculty; หรือ Show Columns From student; เพื่อดูชื่อคอลัมน์หรือรายละเอียดของตาราง ดังรูป



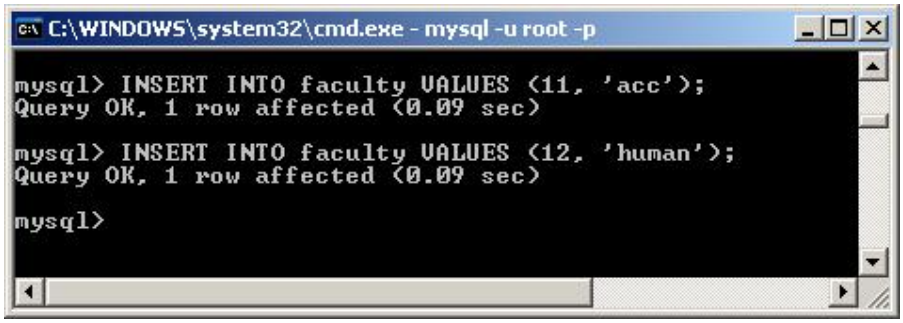
หมายเหตุ : MySQL เวอร์ชั่นปัจจุบัน (เวอร์ชั่น 5.0) ยังไม่สนับสนุนการทำ Foreign Key และ Transaction ดังนั้น MySQL จึงต้องใช้ ENGINE ตัวอื่นที่สนับสนุนการทำ Foreign Key และ Transaction เช่น InnoDB ดังตัวอย่างนี้ ได้เพิ่มคำสั่ง ENGINE=InnoDB ในส่วนท้ายของคำสั่ง Create Table

คำสั่ง Insert

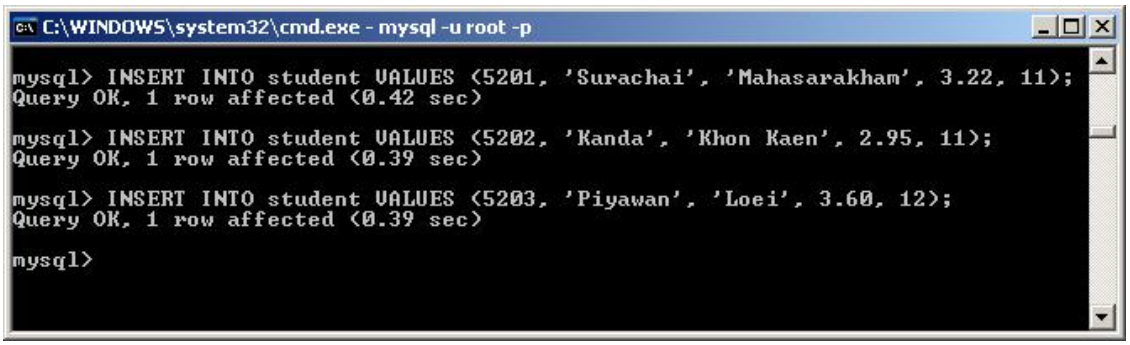
เป็นคำสั่งเพิ่มข้อมูลลงในตาราง ซึ่งมีรูปแบบดังนี้

```
INSERT INTO ชื่อตาราง VALUES (ค่าของคอลัมน์1, ค่าของคอลัมน์2, ..., ค่าของคอลัมน์N);
```

ตัวอย่างเช่น เพิ่มข้อมูลรหัสคณะ และชื่อคณะ ลงในตาราง faculty จำนวน 2 แถว



เพิ่มข้อมูลรหัสนิสิต ชื่อ ที่อยู่ เกรดเฉลี่ย และรหัสคณะ ลงในตาราง Student จำนวน 3 แถว

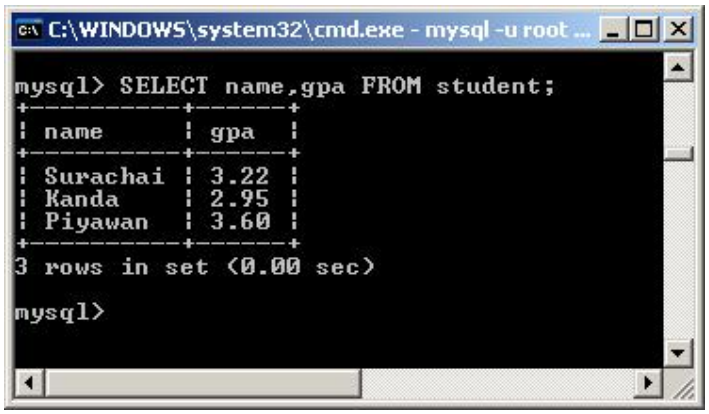


คำสั่ง Select

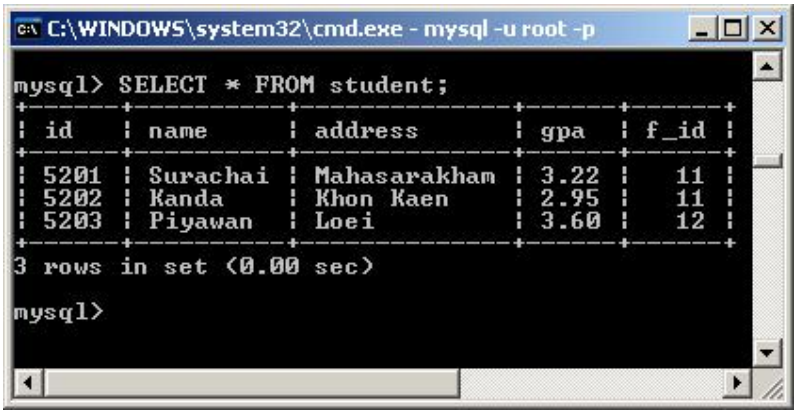
เป็นคำสั่งสำหรับเลือกหรือดึงข้อมูลในตารางขึ้นมาแสดง ซึ่งมีรูปแบบดังนี้

```
SELECT คอลัมน์1, คอลัมน์2, ... FROM ชื่อตาราง [WHERE เงื่อนไข] ;
```

ตัวอย่างเช่น ดึงข้อมูลในตาราง student ขึ้นมาแสดง โดยให้แสดงเฉพาะ ชื่อ (name) และเกรดเฉลี่ย (gpa)



กรณีที่ต้องการดูข้อมูลทุกคอลัมน์ให้ใช้เครื่องหมายดอกจัน (*) แทนการกำหนดชื่อคอลัมน์ทั้งหมด



ถ้าต้องการกำหนดเงื่อนไขในการดึงข้อมูลขึ้นมาแสดง ให้ใส่ WHERE ตามด้วยเงื่อนไข ร่วมกับคำสั่ง SELECT เช่น

SELECT * FROM student WHERE gpa > 3.00 ORDER BY gpa DESC ; **คำอธิบาย** แสดงข้อมูลทุกคอลัมน์ โดยดึงเฉพาะแถวที่มีเกรดเฉลี่ยมากกว่า 3.00 และให้แสดงโดยเรียงลำดับจากนิสิตที่มีเกรดเฉลี่ยสูงที่สุดก่อน
SELECT * FROM student WHERE f_id = 11 ; **คำอธิบาย** แสดงข้อมูลทุกคอลัมน์ โดยดึงเฉพาะแถวที่มีรหัสคณะคือ 11
SELECT * FROM student WHERE name LIKE '%n%' ; **คำอธิบาย** แสดงข้อมูลทุกคอลัมน์ โดยดึงเฉพาะแถวที่มีตัวอักษร n ในชื่อ (เครื่องหมาย % แทนตัวอักษรใด ๆ)
SELECT * FROM student WHERE id = 5203 ; **คำอธิบาย** แสดงข้อมูลทุกคอลัมน์ โดยดึงเฉพาะแถวที่มีรหัสนิสิตคือ 5203
SELECT id, name FROM student ORDER BY name ASC ; **คำอธิบาย** แสดงข้อมูลคอลัมน์รหัสนิสิต และชื่อนิสิต โดยเรียงลำดับจากชื่อนิสิต (จะเรียงจาก a ไปถึง z แต่ถ้าต้องการเรียงจาก z กลับมา a ให้เปลี่ยนจาก ASC เป็น DESC)

SELECT * FROM student WHERE address != 'Mahasarakham' ; **คำอธิบาย** แสดงข้อมูลทุกคอลัมน์ โดยดึงเฉพาะรายการของนิสิตที่ไม่ได้อยู่ในจังหวัดมหาสารคาม
SELECT * FROM student WHERE gpa > 2.50 AND f_id = 12 ; **คำอธิบาย** แสดงข้อมูลทุกคอลัมน์ โดยดึงเฉพาะแถวที่มีเกรดเฉลี่ยมากกว่า 3.00 และมีรหัสคณะคือ 12

การเชื่อมโยงข้อมูลจากหลายตาราง (Join Tables)

เนื่องจากการปฏิบัติงานจริง ข้อมูลที่เก็บในฐานข้อมูลจะประกอบไปด้วยตารางหลายตารางซึ่งส่วนใหญ่จะมีความสัมพันธ์กัน การดึงข้อมูลจากตารางเดียวบางครั้งอาจจะไม่ได้ข้อมูลตามที่ต้องการ ดังนั้นจึงต้องมีการเชื่อมโยงข้อมูลจากหลายตาราง เช่นตัวอย่างนี้มีการเชื่อมโยง 2 ตาราง โดยตาราง Student มีคอลัมน์ f_id เป็นคีย์นอก (Foreign Key) เก็บรหัสคณะ ซึ่งมีความสัมพันธ์กับตาราง Faculty ที่เก็บข้อมูลคณะ ดังนี้

Faculty

id	name
11	acc
12	human

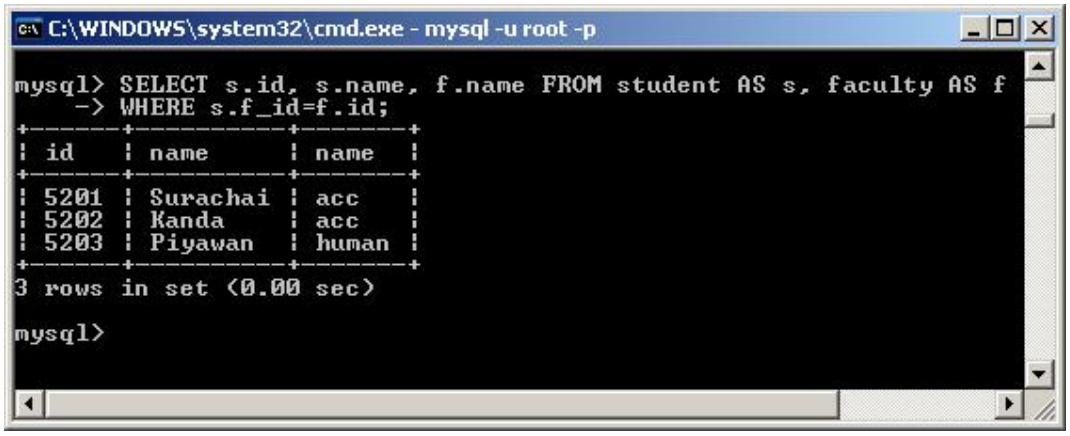
Student

id	name	address	gpa	f_id
5201	Surachai	Mahasarakham	3.22	11
5202	Kanda	Khon Kaen	2.95	11
5203	Piyawan	Loei	3.60	12

คำสั่ง SQL นี้จะทำการเชื่อมโยงข้อมูลจาก 2 ตาราง คือ Faculty และ Student โดยจะแสดงเฉพาะคอลัมน์รหัสนิสิต ชื่อนิสิต และชื่อคณะ ซึ่งข้อมูลรหัสนิสิต และชื่อนิสิตดึงมาจากตาราง Student ส่วนชื่อคณะจะดึงมาจากตาราง Faculty รูปแบบคำสั่งมีดังนี้

```
SELECT s.id, s.name, f.name FROM student AS s, faculty AS f  WHERE s.f_id = f.id ;
```

ผลลัพธ์ที่ได้



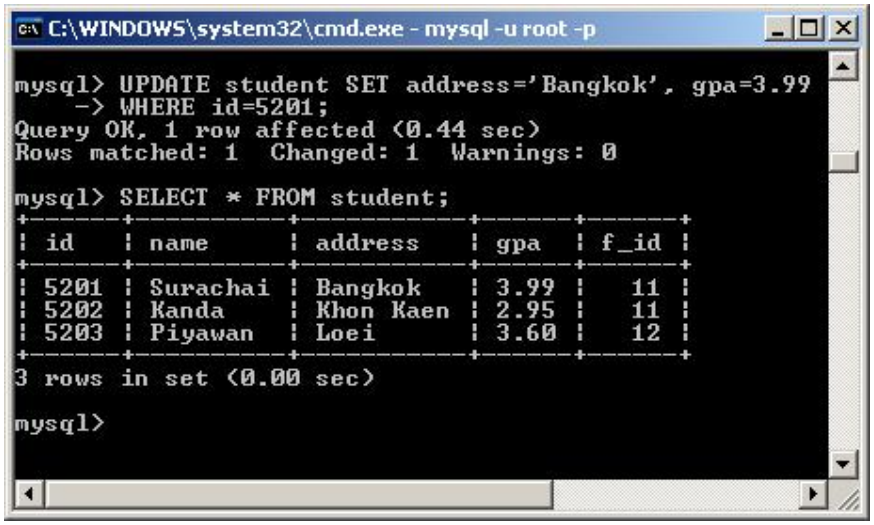
คำสั่ง Update

เป็นคำสั่งในการแก้ไขข้อมูลที่มีอยู่แล้วในตาราง ซึ่งมีรูปแบบดังนี้

```
UPDATE ชื่อตาราง SET ชื่อคอลัมน์1=ค่าใหม่, ชื่อคอลัมน์2=ค่าใหม่, ...N [WHERE เงื่อนไข] ;
```

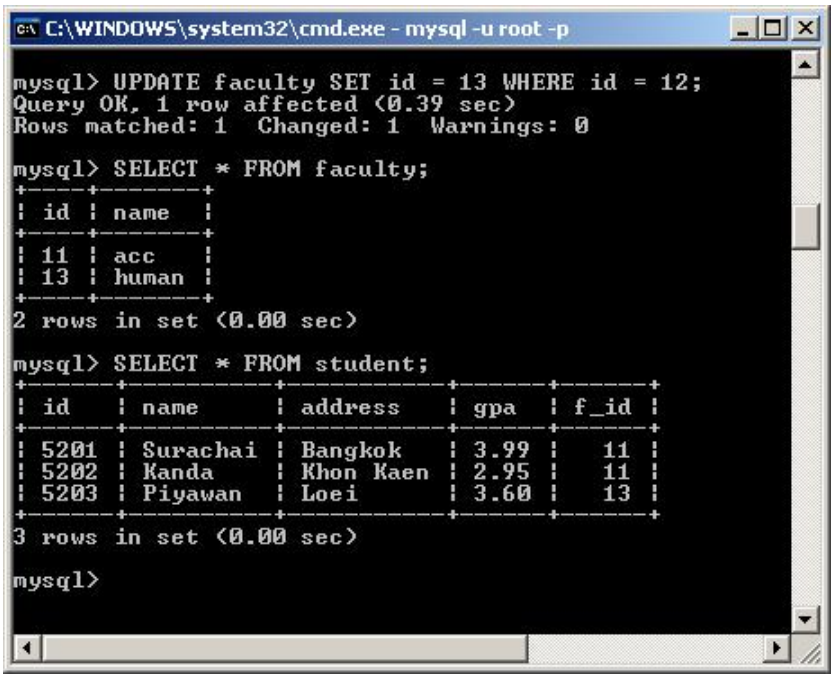
คำสั่ง SQL นี้จะทำการแก้ไขข้อมูลในตาราง Student โดยแก้ไขที่อยู่ (address) เป็น Bangkok และเกรดเฉลี่ย (gpa) เป็น 3.99 ของนิสิตที่มีรหัสคือ 5201 เท่านั้น จากนั้นตรวจสอบผลการแก้ไขได้ด้วยคำสั่ง SELECT * FROM student;

ข้อควรระวัง : ถ้าไม่ใส่ WHERE .. จะเป็นการแก้ไขข้อมูลของนิสิตทุกคน



คำสั่ง SQL นี้จะทำการแก้ไขข้อมูลในตาราง Faculty โดยแก้ไขรหัสคณะจาก 12 เป็น 13 และเกรดเฉลี่ย จากนั้นตรวจสอบผลการแก้ไขได้ด้วยคำสั่ง SELECT * FROM faculty; และ SELECT * FROM student;

ข้อสังเกต : หลังจากที่แก้ไขรหัสคณะเสร็จเรียบร้อยแล้ว ทำให้มีผลกับข้อมูลในตาราง Student ด้วย โดยทำให้รหัสคณะ (f_id) ของนิสิตรหัส 5203 เปลี่ยนไปเป็นรหัส 13 ไปด้วย (เดิมคือ 12) เนื่องจากการสร้างตาราง Student ในข้างต้น เราได้ทำการกำหนดคอลัมน์ f_id เป็นคีย์นอก (ทำเป็น Foreign Key และกำหนดเป็น ON UPDATE CASCADE) ซึ่งจะสัมพันธ์กับรหัสคณะ (id) ในตาราง Faculty จึงทำให้เมื่อมีการแก้ไขรหัสคณะในตาราง Faculty จะทำให้รหัสคณะ (f_id) ในตาราง Student เปลี่ยนแปลงไปด้วย

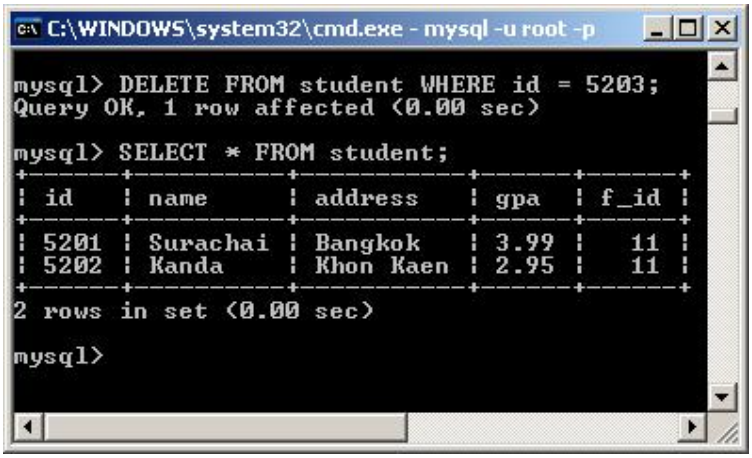


คำสั่ง Delete

เป็นคำสั่งในการลบแถวหรือลบข้อมูลในตาราง มีรูปแบบดังนี้

```
DELETE FROM ชื่อตาราง [WHERE เงื่อนไข] ;
```

ตัวอย่างคำสั่ง SQL นี้จะทำการลบข้อมูลนิสิตที่มีรหัสนิสิตคือ 5203 จากนั้นลองใช้คำสั่ง SELECT * FROM student; เพื่อตรวจสอบผลลัพธ์ ดังรูป



ถ้าต้องการลบข้อมูลนิสิตที่มีรหัสคณะคือ 11 ทั้งหมดออกจากตาราง ให้ใช้คำสั่ง DELETE FROM student WHERE f_id=11;

ข้อควรระวัง : ถ้าไม่ใส่ WHERE .. จะเป็นการลบข้อมูลนิสิตทุกคนออกจากตาราง

คำสั่ง Alter

เป็นคำสั่งในการแก้ไขโครงสร้าง และองค์ประกอบต่าง ๆ ของตาราง เช่น เปลี่ยนชื่อตาราง เพิ่ม-ลบคอลัมน์ หรือเปลี่ยนชนิดข้อมูลของคอลัมน์ เป็นต้น ดังตัวอย่างคำสั่งต่อไปนี้

```
ALTER TABLE student RENAME TO std ;
```
คำอธิบาย เปลี่ยนชื่อตารางจาก student เป็น std

```
ALTER TABLE std ADD phone VARCHAR(60), ADD email VARCHAR(70) ;
```
คำอธิบาย เพิ่มคอลัมน์ใหม่ในตาราง std จำนวน 2 คอลัมน์ คือคอลัมน์ phone ชนิดข้อมูล VARCHAR ขนาด 60 ไบต์ และคอลัมน์ email ชนิดข้อมูล VARCHAR ขนาด 70 ไบต์

```
ALTER TABLE std DROP email;
```
คำอธิบาย ลบคอลัมน์ชื่อ email ออกจากตาราง std

```
ALTER TABLE std CHANGE phone image VARCHAR(80) ;
```
คำอธิบาย เปลี่ยนชื่อคอลัมน์จาก phone เป็น image ชนิดข้อมูล VARCHAR ขนาด 80 ไบต์

```
ALTER TABLE std CHANGE id id INT( 11 ) NOT NULL AUTO_INCREMENT ;
```
คำอธิบาย เปลี่ยนแปลงคุณสมบัติของคอลัมน์ id โดยเพิ่มให้มีคุณสมบัติ AUTO_INCREMENT (คุณสมบัตินี้จะทำให้คอลัมน์ id ที่เก็บข้อมูลรหัสนิสิต รันรหัสต่อเนื่องอัตโนมัติ ในกรณีที่ INSERT ข้อมูลลงในคอลัมน์นี้เป็นค่าว่าง)

คำสั่ง Drop Table

เป็นคำสั่งในการลบตาราง มีรูปแบบดังนี้

```
DROP TABLE ชื่อตาราง ;
```

ตัวอย่างเช่น

```
DROP TABLE std ;
DROP TABLE faculty ;
```
หรือจะใช้คำสั่งเพียงบรรทัดเดียวก็ได้เช่น
```
DROP TABLE std, faculty ;
```
คำอธิบาย ลบตารางชื่อ std และ faculty ออกจากฐานข้อมูล

คำสั่ง Drop Database

เป็นคำสั่งในการลบฐานข้อมูล มีรูปแบบดังนี้

```
DROP DATABASE ชื่อฐานข้อมูล ;
```

ตัวอย่างเช่น

```
DROP DATABASE msu ;
```
คำอธิบาย ลบฐานข้อมูลชื่อ msu

คำสั่งในการทำ Transaction ใน MySQL

Transaction เป็นกระบวนการที่ระบบฐานข้อมูลนำมาใช้เพื่อป้องกันข้อผิดพลาดที่อาจจะเกิดขึ้นโดยไม่ได้ตั้งใจ เช่น การลบหรือแก้ไขข้อมูลผิดรายการ ทำให้ต้องเสียเวลาในการจัดการกับข้อมูลที่ผิดพลาดไป ยิ่งกับระบบฐานข้อมูลที่มีข้อมูลจำนวนมากแล้วก็ยิ่งจะเสียเวลามากขึ้น ดังนั้นการทำ Transaction จะช่วยให้เราสามารถยกเลิกการกระทำที่ก่อให้เกิดการเปลี่ยนแปลงกับตารางฐานข้อมูล เช่น การใช้คำสั่ง INSERT, UPDATE หรือ DELETE เป็นต้น แต่จะไม่มีผลกับคำสั่ง CREATE, SELECT, ALTER และ DROP เนื่องจากไม่ใช่คำสั่งในการเปลี่ยนแปลงข้อมูล นอกจากนี้แล้ว Transaction จะมีผลเฉพาะช่วงที่เรากำหนดขอบเขตการทำ Transaction เอาไว้เท่านั้น

กระบวนการทำ Transaction ใน MySQL สามารถทำได้ตั้งแต่ MySQL เวอร์ชั่น 3.24 แต่เอนจิ้นของ MySQL คือ ISAM และ MyISAM กลับไม่สนับสนุน ดังนั้นถ้าจะทำ Transaction ต้องอาศัยเอนจิ้นตัวอื่นแทน เช่น InnoDB หรือ BDB แต่ล่าสุดทาง MySQL ได้ประกาศว่าใน MySQL ตัวใหม่ที่กำลังจะออกมา เวอร์ชัน 5.1 นี้ MyISAM ซึ่งเป็นเอนจิ้นสำหรับ MySQL โดยตรง จะสนับสนุนการทำ Transaction และ Foreign Key ด้วย

ดังนั้นถ้าจะทำ Transaction ต้องกำหนดเอนจิ้นเป็น InnoDB แทน MyISAM โดยกำหนดต่อท้ายคำสั่งในการสร้างตาราง เช่น

```
CREATE TABLE ... (
  ……..
) ENGINE = InnoDB ;
```

ขั้นตอนในการทำ Transaction มีดังนี้

1. กำหนดจุดเริ่มต้นในการทำ Transaction ใช้คำสั่ง

```
START  TRANSACTION ;
```

2. ใช้คำสั่ง SQL ทำงานไปตามปกติ เช่น คำสั่ง INSERT, UPDATE หรือ DELETE

3. ถ้าต้องการยกเลิกการเปลี่ยนแปลงข้อมูลในข้อที่ 2 ให้ใช้คำสั่ง

```
ROLLBACK ;
```

4. ถ้าต้องการยืนยันการเปลี่ยนแปลงนั้นให้ใช้คำสั่ง

```
COMMIT ;
```

หลังการใช้คำสั่ง ROLLBACK หรือ COMMIT จะถือว่ากระบวนการทำ Transaction สิ้นสุด ถ้าต้องการทำ Transaction ใหม่ ก็ต้องเริ่มด้วยคำสั่ง START TRANSACTION เสมอ ในกรณีที่ใช้คำสั่ง START TRANSACTION ไปแล้ว แต่ไม่ได้ใช้คำสั่ง ROLLBACK หรือ COMMIT การเปลี่ยนแปลงนั้นจะส่งผลเพียงชั่วระยะเวลาที่เชื่อมต่อกับฐานข้อมูลอยู่เท่านั้น การเปลี่ยนแปลงนั้นจะถูกยกเลิกไป หรือเกิดการ ROLLBACK แบบอัตโนมัตินั่นเอง

แบบฝึกหัดท้ายบทที่ 19

1. ให้เขียนคำสั่ง SQL เพื่อสร้างฐานข้อมูล และตาราง ซึ่งมีรายละเอียด ดังนี้

ชื่อฐานข้อมูล : company

ชื่อตาราง : employee

ชื่อฟิลด์	ชนิด(ความยาว)	คำอธิบาย	หมายเหตุ
Eid	Int(4)	รหัสพนักงาน	Primary Key
Ename	VarChar(40)	ชื่อพนักงาน	
Ejob	VarChar(20)	ตำแหน่งงาน	
Estartdate	VarChar(20)	วันที่เริ่มงาน	
Esalary	Int(5)	เงินเดือน	
Ecomm	Int(4)	ค่าคอมมิชชัน	
Did	Int(2)	รหัสแผนก	* (Foreign Key)

2. จากตารางที่สร้างในข้อ 1. ให้ทำการเขียนคำสั่ง SQL เพื่อเพิ่มข้อมูลลงไปในตาราง employee ดังนี้

Eid	Ename	Ejob	Estartdate	Esalary	Ecomm	Did
1001	JOHN	CLERK	13 DEC 1986	6200	0	22
1002	OWEN	SALESMAN	24 JUN 1990	4900	470	33
1003	PETER	SALESMAN	15 JUN 1991	9650	800	33
1004	TIGER	PRESIDENT	26 OCT 2002	10400	0	11

3. จากตารางในข้อ 2. ให้ทำการเขียนคำสั่ง SQL เพื่อดึงข้อมูลขึ้นมาแสดงตามเงื่อนไขต่อไปนี้

3.1) ดึงข้อมูลพนักงานทุกคนจากตารางนี้ ขึ้นมาแสดง

3.2) ดึงข้อมูลของพนักงานที่เริ่มทำงานเดือน "JUN" เท่านั้น

3.3) ดึงข้อมูลโดยแสดงเฉพาะรหัสพนักงาน ชื่อพนักงาน และตำแหน่งงาน

3.4) ดึงข้อมูลของพนักงานคนที่ไม่มีค่าคอมมิชชัน

3.5) ดึงข้อมูลของพนักงานที่มีเงินเดือนมากกว่า 7000 และมีรหัสแผนกคือ 33

4. จากข้อมูลในข้อ 2. ให้ทำการเขียนคำสั่ง SQL เพื่อแก้ไขเรคคอร์ดของพนักงาน โดยทางบริษัทได้ขึ้นเงินเดือนขั้นต่ำให้พนักงานในตำแหน่ง SALESMAN เป็น 12500 ทุกคน

5. จากข้อมูลในข้อ 2. ให้ทำการเขียนคำสั่ง SQL เพื่อลบเรคอร์ดของพนักงานที่มีรหัส 1004

6. จากข้อมูลในข้อ 2. ให้ทำการเขียนคำสั่ง SQL เพื่อลบเรคอร์ดของพนักงานที่มีเงินเดือนต่ำกว่า 6500

7. เขียนคำสั่ง SQL เพื่อเพิ่มคอลัมน์ใหม่ชื่อว่า address เพื่อเก็บที่อยู่ โดยเก็บข้อมูลเป็นตัวอักษร ขนาด 50

8. เขียนคำสั่ง SQL เพื่อลบคอลัมน์ชื่อว่า Estartdate และ Did