บทที่ 6 ภาษาเอสคิวแอล (SQL)

เอสคิวแอล (SQL) คืออะไร

SQL ย่อมาจาก Structured Query Language เป็นภาษาที่ใช้ในการจัดการข้อมูลของฐานข้อมูลเชิงสัมพันธ์ โดยผู้คิดค้นภาษาเอสคิวแอลเป็นรายแรก คือ บริษัทไอบีเอ็มหลังจากนั้นผู้ผลิตซอฟท์แวร์ด้านระบบจัดการฐานข้อมูลเชิงสัมพันธ์ได้พัฒนาระบบที่สนับสนุนเอสคิวแอลมากขึ้นจนเป็นที่นิยมใช้กันอย่างแพร่หลาย ในปัจจุบันซึ่งผู้ผลิตแต่ละรายก็พยายามที่จะพัฒนาระบบจัดการฐานข้อมูลของตนให้มีลักษณะเด่นเฉพาะขึ้นมา ทำให้รูปแบบการใช้คำสั่งเอสคิวแอลมีรูปแบบที่แตกต่างกันไปบ้าง เช่น ORACLE, ACCESS, SQL Base ของ Sybase INGRES หรือ SQL Server ของ Microsoft เป็นต้น

เอสคิวแอลเป็นภาษาในยุคที่ 4 ซึ่งใกล้เคียงภาษาพูด (ภาษาอังกฤษ) ทำให้ง่ายต่อการใช้และทำความเข้าใจ ไม่ต้องสนใจขั้นตอนว่าต้องทำอย่างไรเพื่อให้ได้ข้อมูลมา แค่เพียงพิมพ์คำสั่งง่าย ๆ ลงไปเพื่อบอกว่าเราต้องการข้อมูลอะไร จากตารางไหน ระบบจัดการฐานข้อมูล (DBMS) จะทำการค้นหาข้อมูลให้ทันที เป็นภาษามาตราฐาน ซึ่งจะมีรูปแบบในการเขียนคำสั่งคล้าย ๆ กัน ไม่ว่าจะใช้งานบนเครื่องที่มีระบบปฏิบัติการใดๆ เอสคิวแอลเป็นตัวจักรสำคัญของระบบฐานข้อมูลเชิงสัมพันธ์ ทั้งที่ผู้ผลิตฮาร์ดแวร์ (hardware) และซอฟต์แวร์ (Software)รายใหญ่ต่างที่เข้ามามีส่วนร่วมในการพัฒนาผลิตภัณฑ์ให้ทำงานกับเอสคิวแอลอย่างมีประสิทธิภาพและสามารถ ทำงานร่วมกับภาษาโปรแกรมอื่นได้ เช่น ภาษาซี (C) ภาษาปาสคาล (PASCAL) ภาษาโคบอล (COBOL) เราสามารถจะใช้ภาษาเอสคิวแอลร่วมกับภาษเหล่านี้ให้สามารถทำงานร่วมกันได้โดยจะใช้ภาษาเหล่านี้ในการเขียนโปรแกรมของการคำนวณที่ซับซ้อน

รูปแบบการใช้งานคำสั่งเอสคิวแอล

รูปแบบของการใช้คำสั่ง SQL สามารถใช้ได้ 2 รูปแบบ ดังนี้

1. คำสั่ง SQL ใช้เรียกดูข้อมูลแบบตอบโต้ (Interactive SQL) เป็นการใช้คำสั่ง SQL สั่งงานบนจอภาพ เพื่อเรียกดูข้อมูลจากฐานข้อมูลได้โดยตรงในขณะที่ทำงาน
2. คำสั่ง SQL ที่ใช้เขียนร่วมกับโปรแกรมอื่น ๆ (Embedded QSL) เป็นการนำคำสั่ง SQL ไปใช้ร่วมกับชุดคำสั่งที่เขียนโดยภาษาต่าง ๆ เช่น COBOL PASCAL ACCESS ฯลฯ

การแบ่งกลุ่มคำสั่งเอสคิวแอล แบ่งได้เป็น 3 กลุ่ม

1. Data Definition Language (DDL) เป็นกลุ่มคำสั่งที่ใช้สำหรับกำหนดออบเจ็กต์ฐานข้อมูล เช่น create drop และ alter เป็นต้น

2. Data Manipulation Language (DML) เป็นกลุ่มคำสั่งที่ใช้สำหรับการเข้าถึงและแก้ไข ลบข้อมูล เช่น select insert delete และ update เป็นต้น

3. Data Control Language (DCL) เป็นกลุ่มคำสั่งที่ใช้ควบคุมความปลอดภัยข้อมูล เช่น grant และ revoke

1. กลุ่มคำสั่ง DDL

1.1 คำสั่งสร้างตาราง (Create)

คำสั่ง Create table เป็นคำสั่งที่ใช้สำหรับสร้างตาราง กำหนดชื่อคอลัมน์ และชนิดข้อมูลในแต่ละคอลัมน์ของตาราง

รูปแบบคำสั่ง :

```
CREATE TABLE ชื่อตาราง
(ชื่อคอลัมน์ที่ 1 ชนิดข้อมูล(ขนาดข้อมูล)  [เงื่อนไข],
 ชื่อคอลัมน์ที่ 2  ชนิดข้อมูล(ขนาดข้อมูล)  [เงื่อนไข],
                    .....,
 ชื่อคอลัมน์ที่ N  ชนิดข้อมูล(ขนาดข้อมูล)  [เงื่อนไข],
[FOREIGN KEY (ชื่อคอลัมน์ที่เป็น FK) REFERENCES ชื่อตารางที่สัมพันธ์ (ชื่อคอลัมน์ที่เป็น PK)]);
```

ตัวอย่าง การใช้คำสั่ง SQL ในการสร้างตาราง

1. **ตารางอาจารย์ (Teacher)**

ชื่อคอลัมน์	ความหมาย	ชนิดข้อมูล	ขนาด	คีย์	หมายเหตุ
teach_Id	รหัสอาจารย์	varchar	6	PK	not null
teach_name	ชื่อ	varchar	50		not null
teach_lname	นามสกุล	varchar	50		not null
teach_tel	เบอร์โทรศัพท์	varchar	10		

สามารถเขียนคำสั่ง SQL ในการสร้างตารางได้ดังนี้

```
CREATE TABLE teacher
(teach_Id  varchar(9)  PRIMARY KEY  NOT NULL,
 teach_name  varchar(50)  NOT NULL,
 teach_lname  varchar(50)  NOT NULL,
 teach_tel  varchar(10));
```

ได้ผลลัพธ์ ดังนี้

Teacher

Field	Type	Collation	Attributes	Null	Default	Extra
teach_Id	varchar(9)	utf8_general_ci		No		
teach_name	varchar(50)	utf8_general_ci		No		
teach_lname	varchar(50)	utf8_general_ci		No		
teach_tel	varchar(10)	utf8_general_ci		Yes	*NULL*	

2. ตารางนักศึกษา (student)

ชื่อคอลัมน์	ความหมาย	ชนิดข้อมูล	ขนาด	คีย์	หมายเหตุ
Std_Id	รหัสนักศึกษา	varchar	9	PK	not null
Std_name	ชื่อ	varchar	50		not null
Std_lname	นามสกุล	varchar	50		not null
Std_address	ที่อยู่	text			
Std_tel	เบอร์โทรศัพท์	varchar	10		
teach_Id	รหัสอาจารย์	varchar	6	FK(teacher)	

สามารถเขียนคำสั่ง SQL ในการสร้างตารางได้ดังนี้

```
CREATE TABLE student
(Std_Id  varchar(9)  PRIMARY KEY  NOT NULL,
 Std_name  varchar(50)  NOT NULL,
 Std_lname  varchar(50)  NOT NULL,
 Std_address  text,
 Std_tel  varchar(10),
 teach_Id varchar(6),
 FOREIGN KEY (teach_Id) REFERENCES teacher(teach_Id));
```

ได้ผลลัพธ์ ดังนี้

Student

Field	Type	Collation	Attributes	Null	Default	Extra
Std_Id	varchar(9)	utf8_general_ci		No		
Std_name	varchar(50)	utf8_general_ci		No		
Std_lname	varchar(50)	utf8_general_ci		No		
Std_address	text	utf8_general_ci		Yes	*NULL*	
Std_tel	varchar(10)	utf8_general_ci		Yes	*NULL*	
teach_Id	varchar(6)	utf8_general_ci		Yes	*NULL*	

1.2 คำสั่งแก้ไขตาราง (Alter)

เป็นคำสั่งที่ใช้สำหรับเปลี่ยนแปลงโครงสร้างของตารางที่มีอยู่แล้ว ให้เปลี่ยนไปตามความต้องการ เช่น การเปลี่ยนชนิดข้อมูล หรือขนาดของข้อมูล การเพิ่มคอลัมน์ในตาราง การลบคอลัมน์ หรือการลบเงื่อนไข เป็นต้น

1.2.1 คำสั่ง Alter สำหรับแก้ไขชนิดข้อมูลในคอลัมน์

รูปแบบคำสั่ง :

```
ALTER TABLE ชื่อตาราง
MODIFY ชื่อคอลัมน์  ชนิดข้อมูล  เงื่อนไข ;
```

ตัวอย่าง คำสั่ง Alter สำหรับแก้ไขชนิดข้อมูลในบางคอลัมน์ จากตาราง Teacher

Field	Type	Collation	Attributes	Null	Default	Extra
teach_Id	varchar(9)	utf8_general_ci		No		
teach_name	varchar(50)	utf8_general_ci		No		
teach_lname	varchar(50)	utf8_general_ci		No		
teach_tel	varchar(10)	utf8_general_ci		Yes	*NULL*	

หากต้องการเปลี่ยนขนาดของข้อมูลในส่วนของ teach_Id จาก varchar(9) เป็น varchar(13) สามารถเขียนคำสั่ง SQL ได้ ดังนี้

```
ALTER TABLE Teacher
MODIFY  teach_Id   varchar(13)  Not Null;
```

ได้ผลลัพธ์ ดังนี้

Field	Type	Collation	Attributes	Null	Default	Extra
teach_Id	varchar(13)	utf8_general_ci		No		
teach_name	varchar(50)	utf8_general_ci		No		
teach_lname	varchar(50)	utf8_general_ci		No		
teach_tel	varchar(10)	utf8_general_ci		Yes	*NULL*	

1.2.2 คำสั่ง Alter สำหรับเพิ่มคอลัมน์

รูปแบบคำสั่ง :

```
ALTER TABLE ชื่อตาราง
ADD  ชื่อคอลัมน์   ชนิดข้อมูล   เงื่อนไข ;
```

ตัวอย่าง คำสั่ง Alter สำหรับเพิ่มคอลัมน์ในตาราง จากตาราง Teacher

Field	Type	Collation	Attributes	Null	Default	Extra
teach_Id	varchar(13)	utf8_general_ci		No		
teach_name	varchar(50)	utf8_general_ci		No		
teach_lname	varchar(50)	utf8_general_ci		No		
teach_tel	varchar(10)	utf8_general_ci		Yes	*NULL*	

หากต้องการเพิ่มคอลัมน์ teach_email ชนิดข้อมูล varchar(30) สามารถเขียนคำสั่ง SQL ได้ ดังนี้

```
ALTER TABLE Teacher
ADD  teach_email varchar(30) ;
```

ได้ผลลัพธ์ ดังนี้

Field	Type	Collation	Attributes	Null	Default	Extra
teach_id	varchar(13)	utf8_general_ci		No		
teach_name	varchar(50)	utf8_general_ci		No		
teach_lname	varchar(50)	utf8_general_ci		No		
teach_tel	varchar(10)	utf8_general_ci		Yes	*NULL*	
teach_email	varchar(30)	utf8_general_ci		Yes	*NULL*	

1.2.3 คำสั่ง Alter สำหรับลบคอลัมน์

รูปแบบคำสั่ง :

```
ALTER TABLE ชื่อตาราง
DROP COLUMN ชื่อคอลัมน์ ;
```

ตัวอย่าง คำสั่ง Alter สำหรับลบคอลัมน์ในตาราง จากตาราง Teacher

Field	Type	Collation	Attributes	Null	Default	Extra
teach_id	varchar(13)	utf8_general_ci		No		
teach_name	varchar(50)	utf8_general_ci		No		
teach_lname	varchar(50)	utf8_general_ci		No		
teach_tel	varchar(10)	utf8_general_ci		Yes	*NULL*	
teach_email	varchar(30)	utf8_general_ci		Yes	*NULL*	

หากต้องการลบคอลัมน์ teach_tel สามารถเขียนคำสั่ง SQL ได้ ดังนี้

```
ALTER TABLE Teacher
DROP COLUMN  teach_tel;
```

ได้ผลลัพธ์ ดังนี้

Field	Type	Collation	Attributes	Null	Default	Extra
teach_id	varchar(13)	utf8_general_ci		No		
teach_name	varchar(50)	utf8_general_ci		No		
teach_lname	varchar(50)	utf8_general_ci		No		
teach_email	varchar(30)	utf8_general_ci		Yes	*NULL*	

1.2.4 คำสั่ง Alter สำหรับเปลี่ยนชื่อคอลัมน์

รูปแบบคำสั่ง :

```
ALTER TABLE ชื่อตาราง
RENAME COLUMN ชื่อคอลัมน์เดิม TO ชื่อคอลัมน์ใหม่;
```

ตัวอย่าง คำสั่ง Alter สำหรับเปลี่ยนชื่อคอลัมน์ จากตาราง Teacher

Field	Type	Collation	Attributes	Null	Default	Extra
teach_Id	varchar(13)	utf8_general_ci		No		
teach_name	varchar(50)	utf8_general_ci		No		
teach_lname	varchar(50)	utf8_general_ci		No		
teach_email	varchar(30)	utf8_general_ci		Yes	*NULL*	

หากต้องการเปลี่ยนชื่อคอลัมน์จาก teach_email ไปเป็น email สามารถเขียนคำสั่ง SQL ได้ดังนี้

```
ALTER TABLE Teacher RENAME COLUMN teach_email TO email;
```

1.3 คำสั่งลบตาราง (Drop)

เป็นคำสั่งที่ใช้สำหรับลบตารางที่ไม่ต้องการออก

รูปแบบคำสั่ง :

```
DROP TABLE ชื่อตาราง;
```

ตัวอย่าง คำสั่ง Drop Table เพื่อใช้ลบตาราง Teacher สามารถเขียนคำสั่ง SQL ได้

```
DROP TABLE Teacher;
```

ผลลัพธ์ที่ได้จากการใช้คำสั่งข้างต้น คือ ตาราง Teacher จะถูกลบออกจากฐานข้อมูล

2. กลุ่มคำสั่ง DML

2.1 คำสั่งเพิ่มข้อมูลลงในตาราง (Insert)

เป็นคำสั่งสำหรับเพิ่มข้อมูลที่ต้องการลงไปในตารางต่างๆ ในฐานข้อมูล ซึ่งมีรูปแบบคำสั่งดังนี้

รูปแบบคำสั่ง :

```
INSERT INTO ชื่อตาราง (ชื่อคอลัมน์ที่1, ชื่อคอลัมน์ที่2, ..., ชื่อคอลัมน์ที่N)
VALUES('ข้อมูลของคอลัมน์ที่1', 'ข้อมูลของคอลัมน์ที่2', ..., 'ข้อมูลของคอลัมน์ที่N');
```

ตัวอย่าง คำสั่ง Insert สำหรับเพิ่มข้อมูลลงในตาราง Teacher ในทุกคอลัมน์

จากตาราง Teacher

Field	Type	Collation	Attributes	Null	Default	Extra
teach_Id	varchar(13)	utf8_general_ci		No		
teach_name	varchar(50)	utf8_general_ci		No		
teach_lname	varchar(50)	utf8_general_ci		No		
teach_email	varchar(30)	utf8_general_ci		Yes	*NULL*	

สามารถเขียนคำสั่งเพิ่มข้อมูลได้ดังนี้

```
INSERT INTO Teacher(teach_Id, teach_name, teach_lname, teach_email)
VALUES ('001', 'สมชาย', 'เต็มร้อย', 'pppp@hotmail.com');
```

หรือ

```
INSERT INTO Teacher
VALUES ('001', 'สมชาย', 'เต็มร้อย', 'pppp@hotmail.com');
```

ได้ผลลัพธ์ ดังนี้

teach_Id	teach_name	teach_lname	teach_email
001	สมชาย	เต็มร้อย	pppp@hotmail.com

จากตัวอย่าง สามารถเขียนได้ทั้งสองแบบ ซึ่งจะได้ผลลัพท์ที่เหมือนกัน

ตัวอย่าง คำสั่ง Insert สำหรับเพิ่มข้อมูลลงในตาราง Teacher ในบางคอลัมน์

จากตาราง Teacher

Field	Type	Collation	Attributes	Null	Default	Extra
teach_Id	varchar(13)	utf8_general_ci		No		
teach_name	varchar(50)	utf8_general_ci		No		
teach_lname	varchar(50)	utf8_general_ci		No		
teach_email	varchar(30)	utf8_general_ci		Yes	*NULL*	

สามารถเขียนคำสั่งเพิ่มข้อมูลทุกคอลัมน์ยกเว้นคอลัมน์ tech_email ได้ดังนี้

```
INSERT INTO Teacher(teach_Id, teach_name, teach_lname)
VALUES ('002', 'สมหญิง', 'ไทยเดิม');
```

ได้ผลลัพธ์ ดังนี้

teach_Id	teach_name	teach_lname	teach_email
001	สมชาย	เต็มร้อย	pppp@hotmail.com
002	สมหญิง	ไทยเดิม	*NULL*

2.2 คำสั่งแก้ไขข้อมูลในตาราง (Update)

เป็นคำสั่งสำหรับแก้ไขข้อมูลจากข้อมูลที่มีอยู่เป็นข้อมูลอื่นๆ ตามที่ต้องการ

รูปแบบคำสั่ง :

```
UPDATE ชื่อตาราง
SET ชื่อคอลัมน์ที่1=ข้อมูลที่1, ชื่อคอลัมน์ที่2=ข้อมูลที่2, ... ,ชื่อคอลัมน์ที่N=ข้อมูลที่N
WHERE   เงื่อนไข;
```

ตัวอย่าง คำสั่ง Update สำหรับแก้ไขข้อมูลในตาราง

จากตาราง Teacher

teach_Id	teach_name	teach_lname	teach_email
001	สมชาย	เต็มร้อย	pppp@hotmail.com
002	สมหญิง	ไทยเดิม	*NULL*

สามารถเขียนคำสั่งแก้ไขข้อมูลในคอลัมน์ teach_email ของครูที่มีรหัส 002 ให้เป็น somying@yru.ac.th ได้ดังนี้

```
UPDATE  Teacher
SET tech_email= somying@yru.ac.th
WHERE   tech_Id= '002';
```

ได้ผลลัพธ์ ดังนี้

teach_Id	teach_name	teach_lname	teach_email
001	สมชาย	เต็มร้อย	pppp@hotmail.com
002	สมหญิง	ไทยเดิม	somying@yru.ac.th

2.3 คำสั่งลบข้อมูลออกจากตาราง (Delete)

เป็นคำสั่งสำหรับเพิ่มข้อมูลที่ต้องการลงไปในตารางต่างๆ ในฐานข้อมูล ซึ่งมีรูปแบบคำสั่งดังนี้

รูปแบบคำสั่ง :

```
DELETE FROM ชื่อตาราง  WHERE  เงื่อนไข;
```

ตัวอย่าง คำสั่ง Delete สำหรับการลบข้อมูลในตาราง Teacher โดยต้องการลบข้อมูลครูคนที่ 1 ที่มีรหัสครูเท่ากับ 001

จากตาราง Teacher

teach_Id	teach_name	teach_lname	teach_email
001	สมชาย	เต็มร้อย	pppp@hotmail.com
002	สมหญิง	ไทยเดิม	*NULL*

สามารถเขียนคำสั่งเพิ่มข้อมูลทุกคอลัมน์ยกเว้นคอลัมน์ tech_email ได้ดังนี้

```
DELETE FROM Teacher  WHERE teach_Id = '001';
```

ได้ผลลัพธ์ ดังนี้

teach_Id	teach_name	teach_lname	teach_email
002	สมหญิง	ไทยเดิม	*NULL*

2.4 คำสั่งเรียกดูข้อมูลจากฐานข้อมูล (Select)

คำสั่งเรียกดูข้อมูลจากฐานข้อมูลสามารถใช้ในการเรียกดูข้อมูลต่างๆ จากฐานข้อมูลตามเงื่อนไขที่ต้องการซึ่งมีรูปแบบคำสั่งที่ใช้ในการเรียกดูข้อมูลอยู่หลายรูปแบบดังนี้

2.4.1 การเรียกดูข้อมูลทั้งตาราง

รูปแบบของคำสั่ง SELECT เพื่อใช้เรียกดูข้อมูลทุกคอลัมน์จากตารางใดตารางหนึ่งเป็นดังนี้

รูปแบบคำสั่ง :

```
SELECT *
FROM ชื่อตาราง;
```

ในการเรียกดูข้อมูลทั้งตารางจะต้องใช้เครื่องหมาย * เพื่อแทนการเรียกดูข้อมูลจากทุกคอลัมน์ของตารางหนึ่ง

ตัวอย่าง ต้องการเรียกดูข้อมูลทั้งตารางของตาราง Student

คำสั่งที่ใช้คือ

```
SELECT  *
FROM Student;
```

ได้ผลลัพธ์ ดังนี้

stdCode	stdName	address	sex	facultyCode	majorCode	gpa
50420001	นงคราญ	กรุงเทพฯ	F	BA	ACC	3.1
50420002	สมควร	กรุงเทพฯ	M	IT	ICT	3.1
50420010	ศรีสมร	ปทุมธานี	F	IT	ICT	2.5
50420018	นิรุตน์	อยุธยา	M	BA	ACC	3.7
50420019	จงรัก	*NULL*	M	BA	MN	3.8
50420020	สมชาย	ระยอง	M	BA	MN	2.1
51410002	สมฤทัย	ชลบุรี	F	IT	ICT	2.0

2.4.2 การเรียกดูข้อมูลเพียงบางคอลัมน์

รูปแบบคำสั่ง SELECT เพื่อเรียกดูข้อมูลเพียงบางคอลัมน์จากตารางใดตารางหนึ่ง คือ

รูปแบบคำสั่ง :

```
SELECT ชื่อคอลัมน์ที่1, ชื่อคอลัมน์ที่2, .....
FROM ชื่อตาราง;
```

ในการเรียกดูข้อมูลเพียงบางคอลัมน์ จำเป็นต้องระบุชื่อคอลัมน์ให้ถูกต้อง ตามที่ระบุไว้ในโครงสร้าง หากเรียกดูมากกว่าหนึ่งคอลัมน์จะต้องมีเครื่องหมาย , คั่นระหว่างชื่อคอลัมน์

ตัวอย่าง ต้องการเรียกดู รหัสนักศึกษา ชื่อนักศึกษา และที่อยู่

คำสั่งที่ใช้คือ

```
SELECT  stdCode, stdName, address
FROM Student;
```

ได้ผลลัพธ์ ดังนี้

stdCode	stdName	address
50420001	นงคราญ	กรุงเทพฯ
50420002	สมควร	กรุงเทพฯ
50420010	ศรีสมร	ปทุมธานี
50420018	นิรุตน์	อยุธยา

2.4.2 การเรียกดูข้อมูลโดยเปลี่ยนชื่อคอลัมน์

ในบางครั้งชื่อคอลัมน์ที่เราตั้งขึ้นอาจจะไม่ตรงกับความต้องการ เราสามารถเปลี่ยนชื่อของคอลัมน์ใหม่ในการเรียกดูข้อมูลได้โดยใช้ AS หากต้องการเปลี่ยนชื่อของคอลัมน์ไหนก็ให้ใส่ AS ที่คอลัมน์นั้น

รูปแบบคำสั่ง :

```
SELECT ชื่อคอลัมน์ในตาราง AS ชื่อคอลัมน์ใหม่
FROM ชื่อตาราง;
```

ตัวอย่าง ต้องการแสดงชื่อฟิลด์เป็นภาษาไทย ได้แก่ stdCode เป็น รหัสนักศึกษา, stdName เป็น ชื่อนักศึกษา

คำสั่งที่ใช้คือ

```
SELECT  stdCode AS รหัสนักศึกษา, stdName AS ชื่อนักศึกษา
FROM Student;
```

ได้ผลลัพธ์ ดังนี้

รหัสนักศึกษา	ชื่อนักศึกษา
50420001	นงคราญ
50420002	สมควร
50420010	ศรีสมร
50420018	นิรุตน์
50420019	จงรัก

2.4.3 การเรียกดูข้อมูลโดยจัดเรียงลำดับผลลัพธ์

การจัดเรียงลำดับผลลัพธ์โดยใช้ ORDER BY ซึ่งเป็นคำสั่งที่ใช้สำหรับการเรียงลำดับข้อมูลตามอักษรหรือตัวเลขของคอลัมน์ที่ต้องการ สามารถกำหนดการเรียงจากมากไปหาน้อย (Descending order: DESC) หรือจากน้อยไปหามาก (Ascending order: ASC)

รูปแบบคำสั่ง :

```
SELECT ชื่อคอลัมน์ที่1, ชื่อคอลัมน์ที่2, ชื่อคอลัมน์ที่3, ...
FROM ชื่อตาราง
ORDER BY ชื่อคอลัมน์ที่ต้องการเรียงลำดับ DESC/ASC;
```

ตัวอย่าง ต้องการแสดงข้อมูล รหัสนักศึกษา ชื่อนักศึกษา และที่อยู่ โดยเรียงลำดับตามรหัสนักศึกษาจากมากไปน้อย

คำสั่งที่ใช้คือ

```
SELECT  stdCode, stdName, address
FROM Student
ORDER BY stdCode DESC;
```

ได้ผลลัพธ์ ดังนี้

stdCode	stdName	address
51410021	อุมาพร	เชียงใหม่
51410013	สมชาติ	ลำปาง
51410012	ลิขิต	กรุงเทพฯ
51410011	สัญญา	อุตรดิตย์
51410008	พงษ์ธร	ลำปาง

ตัวอย่าง ต้องการแสดงข้อมูล รหัสนักศึกษา ชื่อนักศึกษา และเกรดเฉลี่ย โดยเรียงลำดับตามชื่อนักศึกษาจากตัวอักษร ก ไป ฮ (หรือน้อยไปมาก)

คำสั่งที่ใช้คือ

```
SELECT  stdCode, stdName, gpa
FROM Student
ORDER BY stdName ASC;
```

ได้ผลลัพธ์ ดังนี้

stdCode	stdName	gpa
50420019	จงรัก	3.8
50420001	นงคราญ	3.1
50420018	นิรุตน์	3.7
51410005	ประภาคาร	2.8
51410006	ปิยะนุช	3.5
51410008	พงษ์ธร	2.9

2.4.4 การเรียกดูข้อมูลแบบมีเงื่อนไข

การเรียกดูข้อมูลแบบมีเงื่อนไข เป็นการระบุค่าเฉพาะของข้อมูลที่ต้องการเรียกดู อาจจะใช้เงื่อนไขเพื่อดึงดูข้อมูลของนักศึกษาที่มีเกรดเฉลี่ยมากกว่า 3.00 เป็นต้น ดังนั้นการเรียกดูข้อมูลแบบมีเงื่อนไขจะมีโอเปอร์เรเตอร์ทางตรรกะ หรือ โอเปอเรเตอร์ SQL เป็นตัวประกอบในการแสดงเงื่อนไข

การเรียกดูข้อมูลแบบมีเงื่อนไข จะใช้ WHERE ต่อท้าย FROM ดังนี้

รูปแบบคำสั่ง :

```
SELECT ชื่อคอลัมน์ที่1, ชื่อคอลัมน์ที่2, ชื่อคอลัมน์ที่3, ...
FROM ชื่อตาราง
WHERE เงื่อนไข;
```

ส่วนของ WHERE ประกอบด้วยส่วนที่สำคัญ 3 คือ

1. ชื่อคอลัมน์ เป็นชื่อคอลัมน์ที่จะนำมาใช้เป็นเงื่อนไข

2. โอเปอเรเตอร์การเปรียบเทียบ ซึ่งแบ่งออกเป็นโอเปอเรเตอร์ทางตรรกะ (LOGICAL OPERATER) โอเปอเรเตอร์ SQL รวมถึงการใช้โอเปอเรเตอร์ BOOLEAN

โอเปอเรเตอร์	ความหมาย
=	เท่ากับ
>	มากกว่า
>=	มากเท่ากับ
<	น้อยกว่า
<=	น้อยกว่าเท่ากับ
<>	ไม่เท่ากับ

3. ข้อมูลเฉพาะที่ต้องการแสดงเป็นเงื่อนไขของชื่อคอลัมน์ที่ระบุในข้อ 1 อาจเป็นค่าคงที่ (CONSTANT) กลุ่มของข้อมูลหรือ นิพจน์(EXPRESSON) หรือชื่อคอลัมน์อื่นที่ต้องการนำมาเปรียบเทียบข้อมูลเฉพาะที่เป็นประเภทตัวอักษรหรือ วัน เดือน ปี (DATE) เมื่อนำมาเป็นเงื่อนไขเฉพาะจะต้องมีเครื่องหมาย ‘ ’ กำกับ

ตัวอย่าง ต้องการแสดงข้อมูล รหัสนักศึกษา ชื่อนักศึกษา และเกรดเฉลี่ย ของนักศึกษาที่มีเกรดเฉลี่ยมากกว่า 3.00

คำสั่งที่ใช้คือ

```
SELECT  stdCode, stdName, gpa
FROM Student
WHERE gpa>3.00;
```

ได้ผลลัพธ์ ดังนี้

stdCode	stdName	gpa
50420001	นงคราญ	3.1
50420002	สมควร	3.1
50420018	นิรุตน์	3.7
50420019	จงรัก	3.8
51410006	ปิยะนุช	3.5
51410007	สัดดา	3.5
51410011	สัญญา	3.6
51410012	ลิขิต	3.5
51410021	อุมาพร	3.5

ตัวอย่าง ต้องการแสดงข้อมูล รหัสนักศึกษา ชื่อนักศึกษา ที่อยู่ และเกรดเฉลี่ย ของนักศึกษาที่อยู่จังหวัดกรุงเทพฯ

คำสั่งที่ใช้คือ

```
SELECT  stdCode, stdName, address, gpa
FROM Student
WHERE address = “กรุงเทพฯ”;
```

ได้ผลลัพธ์ ดังนี้

stdCode	stdName	address	gpa
50420001	นงคราญ	กรุงเทพฯ	3.1
50420002	สมควร	กรุงเทพฯ	3.1
51410006	ปิยะนุช	กรุงเทพฯ	3.5
51410012	ลิขิต	กรุงเทพฯ	3.5

2.4.5 การแสดงตามเงื่อนไขของคอลัมน์เป็นค่าระหว่างค่าสองค่า

การเรียกดูข้อมูลแบบมีเงื่อนไขโดยใช้คำสั่ง BETWEEN... AND.... ซึ่งเป็นโอเปอเรเตอร์ที่กำหนดเงื่อนไขของคอลัมน์เป็นค่าระหว่างค่าสองค่า การใช้ BETWEEN...AND จะแสดงต่อท้ายชื่อคอลัมน์ที่ถูกระบุให้เป็นเงื่อนไข

รูปแบบคำสั่ง :

```
SELECT ชื่อคอลัมน์ที่1, ชื่อคอลัมน์ที่2, ชื่อคอลัมน์ที่3, ...
FROM ชื่อตาราง
WHERE  ชื่อคอลัมน์ BETWEEN ค่าที่1 AND ค่าที่2 ;
```

ตัวอย่าง ต้องการแสดงข้อมูล รหัสนักศึกษา ชื่อนักศึกษา และเกรดเฉลี่ย ของนักศึกษาที่เกรดเฉลี่ยอยู่ระหว่าง 2.50 กับ 3.00

คำสั่งที่ใช้คือ

```
SELECT  stdCode, stdName, gpa
FROM Student
WHERE gpa BETWEEN 2.50 AND 3.00;
```

ได้ผลลัพธ์ ดังนี้

stdCode	stdName	gpa
50420010	ศรีสมร	2.5
51410004	มณีรัตน์	3.0
51410005	ประภาคาร	2.8
51410008	พงษ์ธร	2.9
51410013	สมชาติ	3.0

2.4.6 การเรียกดูข้อมูลแบบมีเงื่อนไขโดยใช้คำสั่ง IN

IN เป็นโอเปอรเรเตอร์ที่ใช้กับเงื่อนไขของคอลัมน์ที่ต้องการระบุเงื่อนไขเป็นกลุ่มของข้อมูลโดย IN จะแสดงต่อท้ายชื่อคอลัมน์ที่ถูกระบุเป็นเงื่อนไข และกลุ่มของข้อมูลที่เป็นข้อมูลเฉพาะของคอลัมน์ที่เป็นเงื่อนไขนี้ จะระบุในวงเล็บ และมีเครื่องหมาย , คั่น

รูปแบบคำสั่ง :

```
SELECT ชื่อคอลัมน์ที่1, ชื่อคอลัมน์ที่2, ชื่อคอลัมน์ที่3, ...
FROM ชื่อตาราง
WHERE  ชื่อคอลัมน์ IN('ข้อมูล1','ข้อมูล2','ข้อมูล3',...);
```

ตัวอย่าง ต้องการแสดงข้อมูลทุกคอลัมน์ของนักศึกษาที่มีรหัส 50420010, 51410004, 51410012 และ 51410021

คำสั่งที่ใช้คือ

```
SELECT  *
FROM Student
WHERE stdCode IN('50420010', '51410004', '51410012', '51410021');
```

ได้ผลลัพธ์ ดังนี้

stdCode	stdName	address	sex	facultyCode	majorCode	gpa
50420010	ศรีสมร	ปทุมธานี	F	IT	ICT	2.5
51410004	มณีรัตน์	เชียงราย	F	BA	MN	3.0
51410012	ลิขิต	กรุงเทพฯ	M	BA	ACC	3.5
51410021	อุมาพร	เชียงใหม่	F	BA	ACC	3.5

2.4.7 การเรียกดูข้อมูลแบบมีเงื่อนไขโดยใช้คำสั่ง LIKE

LIKE เป็นโอเปอรเรเตอร์ที่ใช้ในการค้นหาข้อมูลของคอลัมน์ที่เก็บข้อมูลประเภทตัวอักษรเท่านั้น โดยยังไม่ทราบค่าที่แน่นอนทั้งหมดของข้อมูลที่จะค้นหา หรือรู้เพียงบางตัวอักษรเท่านั้น โอเปอเรเตอร์ LIKE จะระบุต่อท้ายชื่อคอลัมน์ที่เป็นเงื่อนไข โดยจะใช้สัญญาลักษณ์ช่วยในการค้นหาข้อมูลเป็นตัวช่วยในการค้นหาข้อมูลที่เรียกว่า Wild Card (อักขระพิเศษ) ดังนี้

ตัวอักขระพิเศษ		ความหมาย
MS Access	phpMyAdmin	
*	%	แทนอักษรใดๆ กี่ตัวก็ได้
?	_	แทนอักษรใดๆ 1 ตัว

โดยใน DBMS ที่เป็น MS Access และ phpMyAdmin จะใช้อักขระพิเศษในการค้นหาข้อมูลที่ไม่เหมือนกันดังแสดงในตาราง

รูปแบบคำสั่ง :

```
SELECT ชื่อคอลัมน์ที่1, ชื่อคอลัมน์ที่2, ชื่อคอลัมน์ที่3, ...
FROM ชื่อตาราง
WHERE  ชื่อคอลัมน์ LIKE “ข้อมูลที่ต้องการค้นหา”;
```

ตัวอย่าง ต้องการแสดงข้อมูลทุกคอลัมน์ของนักศึกษาที่มีชื่อขึ้นต้นด้วยตัวอักษร ส

คำสั่งที่ใช้คือ

```
SELECT  *
FROM Student
WHERE stdName LIKE “ส%”;
```

ได้ผลลัพธ์ ดังนี้

stdCode	stdName	address	sex	facultyCode	majorCode	gpa
50420002	สมควร	กรุงเทพฯ	M	IT	ICT	3.1
50420020	สมชาย	ระยอง	M	BA	MN	2.1
51410002	สมฤทัย	ชลบุรี	F	IT	ICT	2.0
51410011	สัญญา	อุตรดิตย์	M	IT	ITM	3.6
51410013	สมชาติ	ลำปาง	M	IT	ITM	3.0

ตัวอย่าง ต้องการแสดงข้อมูลทุกคอลัมน์ของนักศึกษาที่มีชื่อเป็นตัวอักษรตัวที่สองเป็น ร

คำสั่งที่ใช้คือ

```
SELECT  *
FROM Student
WHERE stdName LIKE “_ร%”;
```

ได้ผลลัพธ์ ดังนี้

stdCode	stdName	address	sex	facultyCode	majorCode	gpa
50420010	ศรีสมร	ปทุมธานี	F	IT	ICT	2.5
51410005	ประภาคาร	เชียงใหม่	M	IT	ICT	2.8

ตัวอย่าง ต้องการแสดงข้อมูลทุกคอลัมน์ของนักศึกษาที่มีชื่อลงท้ายด้วย ย

คำสั่งที่ใช้คือ

```
SELECT  *
FROM Student
WHERE stdName LIKE “%ย”;
```

ได้ผลลัพธ์ ดังนี้

stdCode	stdName	address	sex	facultyCode	majorCode	gpa
50420020	สมชาย	ระยอง	M	BA	MN	2.1
51410002	สมฤทัย	ชลบุรี	F	IT	ICT	2.0

2.4.8 การเรียกดูข้อมูลแบบมีเงื่อนไขโดยใช้คำสั่ง IS NULL

IS NULL เป็นโอเปอเรเตอร์ที่ใช้ในแสดงค่าของคอลัมน์ที่มีค่าเป็นค่าว่าง หรือไม่มีค่า

รูปแบบคำสั่ง :

```
SELECT ชื่อคอลัมน์ที่1, ชื่อคอลัมน์ที่2, ชื่อคอลัมน์ที่3, ...
FROM ชื่อตาราง
WHERE  ชื่อคอลัมน์ IS NULL;
```

ตัวอย่าง ต้องการแสดงข้อมูลรหัสนักศึกษา ชื่อนักศึกษา ของนักศึกษาที่มีที่อยู่เป็นค่าว่าง

คำสั่งที่ใช้คือ

```
SELECT  stdCode, stdName
FROM Student
WHERE address IS NULL;
```

ได้ผลลัพธ์ ดังนี้

stdCode	stdName
50420019	จงรัก
51410007	ลัดดา

2.4.9 การเรียกดูข้อมูลแบบมีเงื่อนไขโดยใช้โอเปอเรเตอร์บูลีน

การเรียกดูข้อมูลที่มีเงื่อนไขมากกว่าหนึ่งเงื่อนไข สามารถใช้โอเปอเรเตอร์บูลีนเป็นตัวเชื่อมโยงเงื่อนไขดังกล่าว โอเปอเรเตอร์บูลีน ประกอบด้วย

AND ใช้เชื่อมเงื่อนไขสองเงื่อนไข โดยข้อมูลที่จะแสดงออกมาจะต้องเป็นจริงตามเงื่อนไขทั้งสอง

OR ใช้เชื่อมเงื่อนไขสองเงื่อนไข โดยข้อมูลที่จะแสดงออกมาจะต้องเป็นจริงตามเงื่อนไขใดเงื่อนไขหนึ่ง

NOT ใช้แสดงหน้าเงื่อนไขใดเงื่อนไขหนึ่ง เพื่อให้ได้ข้อมูลที่ไม่เป็นไปตามเงื่อนไขที่ระบุ นอกจากนี้ยังสามารถใช้ NOT เป็นเงื่อนไขในเชิงปฏิเสธ โดยใช้ NOT นำหน้าดังนี้

โอเปอเรเตอร์	**ความหมาย**
NOT BETWEEN ... AND	ไม่มีค่าอยู่ระหว่างค่าสองค่าที่ระบุ
NOT IN	ไม่มีค่าที่เป็นกลุ่มข้อมูลที่ระบุ
NOT LIKE	ไม่มีค่าตามตัวอักษรหรือสัญญาลักษณ์
IS NOT NULL	ไม่มีค่าเป็นค่าว่าง

ตัวอย่าง ต้องการแสดงข้อมูลทุกคอลัมน์ของนักศึกษาที่อยู่กรุงเทพฯ และเป็นเพศชาย

คำสั่งที่ใช้คือ

```
SELECT *
FROM Student
WHERE address = “กรุงเทพฯ” AND sex = “M” ;
```

ได้ผลลัพธ์ ดังนี้

stdCode	stdName	address	sex	facultyCode	majorCode	gpa
50420002	สมควร	กรุงเทพฯ	M	IT	ICT	3.1
51410012	ลิขิต	กรุงเทพฯ	M	BA	ACC	3.5

ตัวอย่าง ต้องการแสดงข้อมูลรหัสนักศึกษา ชื่อนักศึกษา รหัสคณะ และเกรดเฉลี่ยของนักศึกษาที่เป็นเพศหญิง อยู่คณะบริหารธุรกิจ และมีเกรดเฉลี่ยมากกว่า 3.00

คำสั่งที่ใช้คือ

```
SELECT  stdCode, stdName, facultyCode, gpa
FROM Student
WHERE sex = "F" AND facultyCode= "BA" AND gpa>3.00;
```

ได้ผลลัพธ์ ดังนี้

stdCode	stdName	facultyCode	gpa
50420001	นงคราญ	BA	3.1
51410007	สัดดา	BA	3.5
51410021	อุมาพร	BA	3.5

ตัวอย่าง ต้องการแสดงข้อมูลรหัสนักศึกษา ชื่อนักศึกษา รหัสคณะ และรหัสสาขาของนักศึกษาที่อยู่คณะบริหารธุรกิจ แต่ไม่อยู่สาขาวิชาการตลาด

คำสั่งที่ใช้คือ

```
SELECT  stdCode, stdName, facultyCode, majorCode
FROM Student
WHERE facultyCode= "BA" AND NOT majorCode = "MK";
```

ได้ผลลัพธ์ ดังนี้

stdCode	stdName	facultyCode	majorCode
50420001	นงคราญ	BA	ACC
50420018	นิรุตน์	BA	ACC
50420019	จงรัก	BA	MN
50420020	สมชาย	BA	MN
51410004	มณีรัตน์	BA	MN
51410012	ลิขิต	BA	ACC
51410021	อุมาพร	BA	ACC

2.4.10 การลำดับการทำงานก่อนหลังของโอเปอเรเตอร์

จะเห็นว่าโอเปอเรเตอร์ต่างๆ ที่ใช้ในการแสดงเงื่อนไขมีมากมาย ซึ่งโดยปกติโอเปอเรเตอร์จะมีลำดับการทำงานจากซ้ายไปขวา หากไม่ต้องการให้ทำงานตามลำดับตามที่กล่าวมาเราสามารถกำหนดลำดับการทำงานของโอเปอเรเตอร์ให้ทำงานก่อนหลังด้วยเครื่องหมายวงเล็บ ( ) เหมือนหลักการคำนวณทางคณิตศาสตร์

ตัวอย่าง แสดงข้อมูล ชื่อนักศึกษา เพศ รหัสสาขา และเกรดเฉลี่ย ของนักศึกษาที่อยู่สาขาวิชาการบัญชี มีเกรดเฉลี่ยน้อยกว่า 2.00 หรือ มากกว่า 3.50

คำสั่งที่ใช้คือ

```
SELECT  stdName, sex, majorCode, gpa
FROM Student
WHERE majorCode = "ACC" AND (gpa< 2.00 OR gpa>3.50);
```

ได้ผลลัพธ์ ดังนี้

stdName	sex	majorCode	gpa
นิรุตน์	M	ACC	3.7

จากตัวอย่างข้างต้นหากไม่ใส่เครื่องหมายวงเล็บ จะมีผลลัพธ์ที่แตกต่างออกไป เช่น

คำสั่งที่ใช้คือ

```
SELECT  stdName, sex, majorCode, gpa
FROM Student
WHERE majorCode = "ACC" AND gpa< 2.00 OR gpa>3.50;
```

ได้ผลลัพธ์ ดังนี้

stdName	sex	majorCode	gpa
นิรุตน์	M	ACC	3.7
จงรัก	M	MN	3.8
สัญญา	M	ITM	3.6

จะเห็นได้ว่าผลลัพธ์ที่ได้จะเป็นผลลัพธ์ที่ไม่ถูกต้องเนื่องจากว่า โจทย์ต้องการนักศึกษาที่อยู่สาขาวิชาบัญชี (ACC) เพียงอย่างเดียวแต่ในกรณีนี้จะแสดงข้อมูลทั้งสาขาวิชาการจัดการระบบสารสนเทศ (ITM) และสาขาวิชาการจัดการ (MN) อีกด้วย จึงจำเป็นต้องใส่วงเล็บเพื่อกำหนดลำดับการทำงานของโอเปอเรเตอร์

2.4.11 นิพจน์ทางคณิตศาสตร์

เราสามารถเรียกดูข้อมูลโดยแสดงผลข้อมูลทางคอลัมน์เป็นนิพจน์ทางคณิตศาสตร์ได้ โดยมีนิพจน์ทางคณิตศาสตร์ที่สามารถนำไปใช้ได้ ดังนี้

เครื่องหมาย	ความหมาย
+ (Add)	บวก
- (Subtract)	ลบ
* (Multiply)	คูณ
/ (Divide)	หาร

ตัวอย่าง แสดงข้อมูล รหัสอาจารย์ ชื่ออาจารย์ และเงินเดือนหลังหักภาษี 7 เปอร์เซ็นต์

คำสั่งที่ใช้คือ

```
SELECT  lectCode, name, salary-(salary*7/100) AS totalSalary
FROM Lecturer ;
```

ได้ผลลัพธ์ ดังนี้

lectCode	name	totalSalary
001	อ.ฟ้าใส	32550.0000
002	อ.งามแข	26040.0000
005	อ.โอรส	13950.0000
006	อ.ทรงศรี	8370.0000

2.4.12 การแสดงข้อมูลด้วยฟังก์ชั่นที่เกี่ยวกับการรวม

การเรียกดูข้อมูลอาจกระทำการสรุปค่าของข้อมูล โดยการรวม การหาค่าเฉลี่ย การนับหรือการหาค่าสูงสุด หรือต่ำสุด โดยฟังก์ชั่นที่เกี่ยวกับการรวมที่สามารถนำไปใช้ในการเรียกดูข้อมูลมีดังนี้

รูปแบบฟังก์ชั่น	ผลที่ได้จากการใช้ฟังก์ชั่น
AVG(ชื่อคอลัมน์)	เป็นฟังก์ชั่นที่ใช้ในการหาค่าเฉลี่ยของคอลัมน์หนึ่ง ๆ ที่เก็บข้อมูลประเภทตัวเลข
COUNT(ชื่อคอลัมน์)	เป็นฟังก์ชั่นที่ใช้ในการนับจำนวนแถว
MAX(ชื่อคอลัมน์)	เป็นฟังก์ชั่นที่ใช้ในการคำนวณหาค่าสูงสุด
MIN(ชื่อคอลัมน์)	เป็นฟังก์ชั่นที่ใช้ในการคำนวณหาค่าต่ำสุด
SUM(ชื่อคอลัมน์)	เป็นฟังก์ชั่นที่ใช้ในการคำนวณหาค่าผลรวม

ตัวอย่าง แสดงค่าเฉลี่ยของเกรดเฉลี่ยของนักศึกษาทั้งหมด

คำสั่งที่ใช้คือ

```
SELECT  AVG(gpa) AS 'ค่าเฉลี่ยของ GPA'
FROM Student ;
```

ได้ผลลัพธ์ ดังนี้

ค่าเฉลี่ยของ GPA
3.1

ตัวอย่าง แสดงข้อมูลจำนวนนักศึกษาทั้งหมด

คำสั่งที่ใช้คือ

```
SELECT  COUNT(stdCode) AS 'จำนวนนักศึกษา'
FROM Student ;
```

ได้ผลลัพธ์ ดังนี้

จำนวนนักศึกษา
16

ตัวอย่าง แสดงข้อมูลจำนวนนักศึกษาที่อยู่คณะบริหารธุรกิจ

คำสั่งที่ใช้คือ

```
SELECT  COUNT(*) AS 'จำนวนนักศึกษา'
FROM Student
WHERE facultyCode = 'BA';
```

ได้ผลลัพธ์ ดังนี้

จำนวนนักศึกษาคณะบริหารธุรกิจ
9

ตัวอย่าง หาค่าเกรดเฉลี่ยสูงสุด และค่าเกรดเฉลี่ยต่ำสุดของนักศึกษาทั้งหมด

คำสั่งที่ใช้คือ

```
SELECT  MAX(gpa) AS 'เกรดเฉลี่ยสูงสุด', MIN(gpa) AS 'เกรดเฉลี่ยต่ำสุด'
FROM Student;
```

ได้ผลลัพธ์ ดังนี้

เกรดเฉลี่ยสูงสุด	เกรดเฉลี่ยต่ำสุด
3.8	2.0

ตัวอย่าง หาค่าเกรดเฉลี่ยสูงสุด และค่าเกรดเฉลี่ยต่ำสุดของนักศึกษาคณะคณะเทคโนโลยีสารสนเทศ

คำสั่งที่ใช้คือ

```
SELECT  MAX(gpa) AS 'เกรดเฉลี่ยสูงสุด', MIN(gpa) AS 'เกรดเฉลี่ยต่ำสุด'
FROM Student
WHERE facultyCode = 'IT';
```

ได้ผลลัพธ์ ดังนี้

เกรดเฉลี่ยสูงสุด	เกรดเฉลี่ยต่ำสุด
3.6	2.0

ตัวอย่าง หาผลรวมทั้งหมดของเงินเดือนอาจารย์

คำสั่งที่ใช้คือ

```
SELECT  SUM(salary) AS 'เงินเดือนรวมของอาจารย์'
FROM Lecturer;
```

ได้ผลลัพธ์ ดังนี้

เงินเดือนรวมของอาจารย์
87000

2.4.13 การแสดงข้อมูลโดยการจัดกลุ่ม

การจัดกลุ่มจะใช้คำสั่ง GROUP BY เพื่อให้จัดกลุ่มตามคอลัมน์ที่กำหนดให้จัดกลุ่มข้อมูลเฉพาะย่อยลงไป เช่น ให้หาค่าเฉลี่ยของเกรดเฉลี่ยโดยจัดกลุ่มตามรหัสคณะ เป็นต้น

ในกรณีที่ใช้ GROUP BY การระบุชื่อของคอลัมน์ที่จะเรียกข้อมูลออกมาดูนั้น จะต้องเป็นข้อมูลของคอลัมน์ที่สามารถนำมาใช้จัดกลุ่มโดยใช้ GROUP BY ได้ เช่น ต้องการเรียกดูข้อมูล รหัสคณะ และค่าเฉลี่ยของเกรดเฉลี่ยของแต่ละคณะ การเรียกดูข้อมูลในกรณีนี้จะทำได้ เพราะทั้งรหัสคณะและเกรดเฉลี่ยสามารถหาค่าออกมาในลักษณะการจัดกลุ่มได้

แต่หากมีการเรียกดูข้อมูล ชื่อนักศึกษา รหัสคณะ และค่าเฉลี่ยของเกรดเฉลี่ยแต่ละคณะ การเรียกดูข้อมูลในลักษณะนี้จะไม่ถูกต้อง เพราะชื่อนักศึกษา จะแสดงเป็นรายละเอียดซึ่งไม่สามารถจัดกลุ่มได้ ดังนั้น จะต้องระมัดระวังเวลาเรียกดูข้อมูลจากคอลัมน์ต่าง ๆ เมื่อมีการคำสั่ง GROUP BY นั่นคือรายละเอียดข้อมูลที่เรียกดู ควรเป็นฟังก์ชั้นในการรวม หรือชื่อคอลัมน์ที่สามารถจัดกลุ่มโดยใช้คำสั่ง GROUP BY ได้

ตัวอย่าง แสดงข้อมูลรหัสคณะ และค่าเฉลี่ยของเกรดเฉลี่ยของคณะต่างๆ

คำสั่งที่ใช้คือ

```
SELECT  facultyCode, AVG(gpa) AS ‘ค่าเฉลี่ยของเกรดเฉลี่ย’
FROM Student
GROUP BY facultyCode ;
```

ได้ผลลัพธ์ ดังนี้

facultyCode	ค่าเฉลี่ยของเกรดเฉลี่ย
BA	3.2333333333333
IT	2.9285714285714

ตัวอย่าง แสดงข้อมูลรหัสคณะ เกรดเฉลี่ยสูงสุด และเกรดเฉลี่ยต่ำสุดของนักศึกษาแต่ละคณะ

คำสั่งที่ใช้คือ

```
SELECT  facultyCode, MAX(gpa) AS ‘เกรดเฉลี่ยสูงสุด’ , MIN(gpa) AS
‘เกรดเฉลี่ยต่ำสุด’
FROM Student
GROUP BY facultyCode ;
```

ได้ผลลัพธ์ ดังนี้

facultyCode	เกรดเฉลี่ยสูงสุด	เกรดเฉลี่ยต่ำสุด
BA	3.8	2.1
IT	3.6	2.0

ตัวอย่าง แสดงข้อมูลรหัสสาขา และค่าเฉลี่ยของเกรดเฉลี่ยของนักศึกษาแต่ละสาขาวิชา ยกเว้นสาขาวิชาการตลาด

คำสั่งที่ใช้คือ

```
SELECT  majorCode, AVG(gpa) AS 'ค่าเฉลี่ยของเกรดเฉลี่ย'
FROM Student
WHERE majorCode <> 'MK'
GROUP BY majorCode ;
```

ได้ผลลัพธ์ ดังนี้

majorCode	ค่าเฉลี่ยของเกรดเฉลี่ย
ACC	3.45
ICT	2.6
ITM	3.3666666666667
MN	2.9666666666667

2.4.14 การแสดงข้อมูลโดยมีเงื่อนไขจากการจัดกลุ่ม

คำสั่ง HAVING จะใช้ร่วมกับคำสั่ง GROUP BY เสมอ เพื่อต้องการให้แสดงข้อมูลที่ได้ผ่านการจัดกลุ่มโดย GROUP BY เพียงบางส่วนตามเงื่อนไขที่ระบุคำสั่ง HAVING ดังนั้น การเรียกดูข้อมูลโดยใช้คำสั่ง HAVING จะต้องมีคำสั่ง GROUP BY อยู่ด้วยเสมอ

ตัวอย่าง แสดงข้อมูลรหัสสาขา และค่าเฉลี่ยของเกรดเฉลี่ยของนักศึกษาแต่ละสาขาวิชา ที่มีค่าเฉลี่ยของเกรดเฉลี่ยมากกว่า 3.00

คำสั่งที่ใช้คือ

```
SELECT  majorCode, AVG(gpa) AS 'ค่าเฉลี่ยของเกรดเฉลี่ย'
FROM Student
GROUP BY majorCode
HAVING AVG(gpa) > 3.00;
```

ได้ผลลัพธ์ ดังนี้

majorCode	ค่าเฉลี่ยของเกรดเฉลี่ย
ACC	3.45
ITM	3.3666666666667
MK	3.2

ตัวอย่าง แสดงข้อมูลรหัสสาขา และจำนวนนักศึกษาแต่ละสาขาวิชาที่มีจำนวนนักศึกษามากกว่า 2 คน

คำสั่งที่ใช้คือ

```
SELECT  majorCode, COUNT(stdCode) AS 'จำนวนนักศึกษา'
FROM Student
GROUP BY majorCode
HAVING COUNT(stdCode) > 2;
```

ได้ผลลัพธ์ ดังนี้

majorCode	จำนวนนักศึกษา
ACC	4
ICT	4
ITM	3
MN	3

2.4.15 การใช้คำสั่ง GROUP BY และ HAVING ร่วมกับ WHERE

จากการที่คำสั่ง GROUP BY และ HAVING ใช้ในการเรียกดูข้อมูลบางแถวออกมาตามเงื่อนไขที่ระบุไว้ สำหรับการใช้ WHERE เพื่อแสดงเงื่อนไขนั้น จะใช้ฟังก์ชั่นที่เกี่ยวกับการรวมมาเป็นเงื่อนไขไม่ได้

ตัวอย่าง แสดงข้อมูลรหัสสาขา และค่าเฉลี่ยของเกรดเฉลี่ยของนักศึกษาแต่ละสาขาวิชา ที่มีค่าเฉลี่ยของเกรดเฉลี่ยมากกว่า 2.50

คำสั่งที่ใช้คือ

```
SELECT  majorCode, AVG(gpa) AS 'ค่าเฉลี่ยของเกรดเฉลี่ย'
FROM Student
WHERE AVG(gpa) > 2.50;
GROUP BY majorCode
```

ในกรณีนี้คำสั่งในการเรียกดูข้อมูลไม่ถูกต้อง โดยที่คำสั่ง WHERE สามารถใช้ในการกำหนดเงื่อนไขได้ แต่ไม่สามารถใช้ร่วมกับฟังก์ชั่นเกี่ยวกับการรวมเพื่อระบุเงื่อนไขในการดึงข้อมูลที่ถูกจัดกลุ่มบางข้อมูลออกมาได้ จากตัวอย่างข้างต้นจะต้องเขียนด้วยคำสั่ง HAVING ดังนี้

```
SELECT  majorCode, AVG(gpa) AS 'ค่าเฉลี่ยของเกรดเฉลี่ย'
FROM Student
GROUP BY majorCode
HAVING AVG(gpa) > 2.50 ;
```

ได้ผลลัพธ์ ดังนี้

majorCode	ค่าเฉลี่ยของเกรดเฉลี่ย
ACC	3.45
ICT	2.6
ITM	3.3666666666667
MK	3.2
MN	2.9666666666667

ตัวอย่าง แสดงข้อมูลรหัสสาขา และจำนวนนักศึกษาแต่ละสาขาวิชา ยกเว้นสาขาวิชาการบัญชี

คำสั่งที่ใช้คือ

```
SELECT  majorCode, COUNT(*) AS 'จำนวนนักศึกษา'
FROM Student
WHERE NOT majorCode = 'ACC'
GROUP BY majorCode ;
```

ได้ผลลัพธ์ ดังนี้

majorCode	จำนวนนักศึกษา
ICT	4
ITM	3
MK	2
MN	3

ตัวอย่าง แสดงข้อมูลรหัสสาขา และจำนวนนักศึกษาแต่ละสาขาวิชา ที่มีจำนวนนักศึกษามากกว่า 2 คน และไม่แสดงข้อมูลของสาขาวิชาการบัญชี

คำสั่งที่ใช้คือ

```
SELECT  majorCode, COUNT(*) AS 'จำนวนนักศึกษา'
FROM Student
WHERE majorCode <> 'ACC'
GROUP BY majorCode ;
HAVING COUNT(*) > 2 ;
```

ได้ผลลัพธ์ ดังนี้

majorCode	จำนวนนักศึกษา
ICT	4
ITM	3
MN	3

แบบฝึกหัด

ตอนที่ 1 จากตารางในภาคผนวก ก. จงเขียนคำสั่ง SQL เพื่อแสดงข้อมูลตามเงื่อนไขต่างๆ ดังนี้

1.1. แสดงข้อมูล รหัสนักศึกษา ชื่อนักศึกษา และเกรดเฉลี่ย ของนักศึกษาที่มีเกรดเฉลี่ยระหว่าง 2.00 ถึง 3.00

1.2. แสดงข้อมูล ชื่อนักศึกษา เพศ และเกรดเฉลี่ย ของนักศึกษาที่อยู่กรุงเทพ และเป็นเพศชาย

1.3. แสดงข้อมูล รหัสนักศึกษา ชื่อนักศึกษา และเกรดเฉลี่ย ของนักศึกษาทั้งหมด โดยเรียงลำดับเกรดจากน้อยไปหามาก

1.4. แสดงข้อมูล รหัสนักศึกษา ชื่อนักศึกษา และเกรดเฉลี่ย ของนักศึกษาที่อักษรของชื่อมีตัว “ค”เป็นตัวที่สาม

1.5. แสดงข้อมูล ชื่อนักศึกษา และเพศ ของนักศึกษาที่ไม่มีอักษร “ย”เป็นตัวสุดท้ายของชื่อ

1.6. แสดงข้อมูล รหัสนักศึกษา ชื่อนักศึกษา ที่อยู่ เพศ รหัสคณะ รหัสสาขา และเกรดเฉลี่ย ของนักศึกษาที่ไม่มีรหัสนักศึกษา 51410002, 51410011 และ 51410013 และเรียงลำดับรหัสนักศึกษาจากน้อยไปหามาก

1.7. แสดงข้อมูลรหัสนักศึกษา ชื่อนักศึกษา ของนักศึกษาที่มีที่อยู่ไม่เป็นค่าว่าง

1.8. แสดงข้อมูลรหัสอาจารย์ ชื่ออาจารย์ และเงินเดือนหลังหักค่าประกันสังคม 750 บาท

1.9. แสดงรหัสคณะ และค่าเฉลี่ยของเงินเดือนอาจารย์แต่ละคณะ

1.10. แสดงรหัสคณะ และจำนวนอาจารย์ผู้สอนของแต่ละคณะ

1.11. แสดงเกรดเฉลี่ยสูงสุด เกรดเฉลี่ยต่ำสุด ของแต่ละสาขา

1.12. แสดงรหัสสาขาวิชา เกรดเฉลี่ยสูงสุด และเกรดเฉลี่ยต่ำสุดของแต่ละสาขา แต่ไม่ต้องแสดงของสาขาบัญชี

1.13. แสดงรหัสสาขาวิชา และค่าเฉลี่ยของเกรดเฉลี่ยของแต่ละสาขา ที่มีเกรดเฉลี่ยมากกว่า 3.00

1.14. แสดงรหัสสาขาวิชา และจำนวนนักศึกษาแต่ละสาขาที่มีจำนวนนักศึกษามากกว่า 2 คน

1.15. แสดงชื่อจังหวัด และค่าเฉลี่ยของเกรดเฉลี่ยของแต่ละจังหวัดที่มีจำนวนนักศึกษามากกว่า 2 คน

ตอนที่ 2 จงเขียนผลลัพธ์ที่เกิดขึ้นกับตารางในภาคผนวก ก. ภายหลังจากเขียนคำสั่ง SQL ต่อไปนี้

2.1
```
SELECT fucultyCode, Count(*) AS "จำนวน นศ"
FROM Student
WHERE gpa>3.00
GROUP BY fucultyCode ;
```

2.2
```
SELECT majorCode AS รหัสสาขาวิชา , AVG(gpa) AS "เกรดเฉลี่ย"
FROM student
GROUP BY majorCode ;
HAVING AVG(gpa)>3.00 ;
```

2.3
```
SELECT stdName, sex, gpa
FROM student
WHERE address = ‘กรุงเทพฯ’ and sex = ‘M’;
```

2.4
```
SELECT stdCode, stdName, sex, gpa
FROM student
WHERE stdName LIKE “_ง%”
ORDER BY stdCode DESC;
```

2.5 **SELECT** stdCode, stdName, sex, gpa

FROM student

WHERE gpa NOT BETWEEN 2.5 AND 3.0

ORDER BY stdName ASC;

ตอนที่ 3 ให้นักศึกษาเขียนคำสั่ง SQL เพื่อสร้างตารางจากข้อมูล ดังต่อไปนี้

3.1 STUDENT

ชื่อคอลัมน์	คำอธิบาย	ชนิดข้อมูล	เงื่อนไข	ค่าเริ่มต้น
id	รหัสนักศึกษา	Varchar(9)	Primary key	
firstname	ชื่อ	Varchar(20)	Not null	
lastname	นามสกุล	Varchar(20)	Not null	
address	ที่อยู่	text		

3.2 SUBJECT

ชื่อคอลัมน์	คำอธิบาย	ชนิดข้อมูล	เงื่อนไข	ค่าเริ่มต้น
sub_id	รหัสวิชา	Varchar(8)	Primary key	
sub_name	ชื่อวิชา	Varchar(70)	Not null	
credit	จำนวนหน่วยกิจ	Int(1)	Not null	

3.3 REGISTER

ชื่อคอลัมน์	คำอธิบาย	ชนิดข้อมูล	เงื่อนไข	ค่าเริ่มต้น
semester	ภาคการศึกษา	Varchar(6)	Not null	
id	รหัสนักศึกษา	Varchar(9)	FK(student)	
sub_id	รหัสวิชา	Varchar(8)	FK(subject)	
section	กลุ่ม	Varchar(2)	Not null	

ตอนที่ 4 จากตารางที่กำหนดให้ จงเขียนคำสั่ง SQL ตามโจทย์ในแต่ละข้อ

STUDENT

stdCode	stdName	address	sex	gpa
50420001	นงคราญ	กรุงเทพฯ	F	3.1
50420002	สมควร	กรุงเทพฯ	M	3.1
50420010	ศรีสมร	ปทุมธานี	F	2.5
50420018	นิรุตน์	อยุธยา	M	3.7
50420019	จงรัก	*NULL*	M	3.8
50420020	สมชาย	ระยอง	M	2.1
51410002	สมฤทัย	ชลบุรี	F	2.0
51410004	มณีรัตน์	เชียงราย	F	3.0

4.1 แก้ไขข้อมูลชื่อนักศึกษาที่มีรหัส 50420019 จากเดิมชื่อ จงรัก ให้เป็น สมรักษ์ และ จากเพศหญิง (F) ให้เป็น เพศชาย (M)

4.2 แก้ไขข้อมูลเกรดเฉลี่ยของนักศึกษาที่มีรหัส 50420002 จากเดิม 3.1 ให้เป็น 3.5

4.3 เพิ่มข้อมูลนักศึกษาใหม่ลงไปในตาราง ซึ่งมีรหัสนักศึกษา 51410005 ชื่อนางสาวนงคราญ แก้วกล้า อยู่จังหวัดยะลา ยังไม่มีเกรดเฉลี่ย

4.4 ลบข้อมูลนักศึกษาที่มี รหัส50420020 เนื่องจากว่านักศึกษาคนนี้ลาออก

4.5 ลบตารางข้อมูลนักศึกษาออกจากระบบฐานข้อมูล

ภาคผนวก ก.

ตารางนักศึกษา (STUDENT)

stdCode	stdName	address	sex	fucultyCode	majorCode	gpa
50420001	นงคราญ	กรุงเทพฯ	F	BA	ACC	3.1
50420002	สมควร	กรุงเทพฯ	M	IT	ICT	3.1
50420010	ศรีสมร	ปทุมธานี	F	IT	ICT	2.5
50420018	นิรุตน์	อยุธยา	M	BA	ACC	3.7
50420019	จงรัก	*NULL*	M	BA	MN	3.8
50420020	สมชาย	ระยอง	M	BA	MN	2.1
51410002	สมฤทัย	ชลบุรี	F	IT	ICT	2.0
51410004	มณีรัตน์	เชียงราย	F	BA	MN	3.0
51410005	ประภาคาร	เชียงใหม่	M	IT	ICT	2.8
51410006	ปิยะนุช	กรุงเทพฯ	F	IT	ITM	3.5
51410007	สัดดา	*NULL*	F	BA	MK	3.5
51410008	พงษ์ธร	ลำปาง	M	BA	MK	2.9
51410011	สัญญา	อุตรดิตย์	M	IT	ITM	3.6
51410012	ลิขิต	กรุงเทพฯ	M	BA	ACC	3.5
51410013	สมชาติ	ลำปาง	M	IT	ITM	3.0
51410021	อุมาพร	เชียงใหม่	F	BA	ACC	3.5

ตารางสาขาวิชา (MAJOR)

facultyCode	majorCode	majorName
BA	ACC	สาขาวิชาการบัญชี
IT	ICT	สาขาวิชาเทคโนโลยีสารสนเทศและการสื่อสาร
IT	ITM	สาขาวิชาการจัดการระบบสารสนเทศ
BA	MK	สาขาวิชาการตลาด
BA	MN	สาขาวิชาการจัดการ

ตารางอาจารย์ผู้สอน (LECTURER)

lectCode	name	facultyCode	majorCode	salary
001	อ.ฟ้าใส	BA	ACC	35000
002	อ.งามแข	BA	ACC	28000
005	อ.โอรส	IT	ICT	15000
006	อ.ทรงศรี	IT	ITM	9000

ตารางรายวิชา (SUBJECT)

sbjCode	sbjName	credit
B10000	การบัญชีเบื้องต้น	3
B10001	การบัญชีชั้นกลาง	3
B10002	พฤติกรรมองค์การ	3
B10003	เศรษฐศาสตร์เบื้องต้น	3
B10004	เศรษฐศาสตร์มหภาค	3
B10005	การตลาดและการบริการ	3
B10006	การจัดการช่องทางจำหน่าย	3
B10007	ระบบบริหารงานบุคคล	3
IT0001	เทคโนโลยีสารสนเทศเบื้องต้น	3
IT0002	การโปรแกรมคอมพิวเตอร์ 1	3
IT0003	การโปรแกรมคอมพิวเตอร์ 2	3
IT0004	ระบบฐานข้อมูล	3
IT0005	การวิเคราะห์และออกแบบระบบ	3
IT0006	เครือข่ายคอมพิวเตอร์	3
IT0007	โครงสร้างข้อมูล	3
IT0008	การบริหารศูนย์คอมพิวเตอร์	3
IT0009	ระบบสารสนเทศเพื่อการจัดการ	3

ตารางการสอน (ITEM_TEACHING)

lectCode	sbjCode	section
001	B10000	1
001	B10001	1
005	IT0001	1
005	IT0001	2
006	IT0005	1
006	IT0009	1

ตารางคณะ (FACULTY)

facultyCode	facultyName
BA	คณะบริหารธุรกิจ
IT	คณะเทคโนโลยีสารสนเทศ

ตารางการลงทะเบียน (ITEM_REGISTERED)

semester	stdCode	sbjCode	section
2555/2	50420001	B10000	1
2555/2	50420001	IT0009	1
2555/2	50420010	IT0001	1
2555/2	50420010	IT0009	1
2555/2	50420018	B10000	1
2555/2	50420018	IT0009	1
2555/2	50420019	B10002	1
2555/2	50420019	B10003	2
2555/2	50420020	B10007	1
2555/2	50420002	B10007	2
2555/2	50420002	IT0004	1
2555/2	50420007	IT0007	1
2555/2	50420007	B10005	1
2555/2	50420007	B10006	1
2555/2	50420010	IT0005	1
2555/2	50420010	IT0001	1
2555/2	50420011	IT0002	1
2555/3	50420011	IT0004	1
2555/4	50420012	B10000	1
2555/5	50420012	B10001	1
2555/6	50420012	IT0009	1
2555/7	50420013	IT0005	1
2555/8	50420013	IT0006	2
2555/9	50420013	IT0009	1