ตัวอย่าง

สร้าง Query สืบค้นข้อมูล ด้วย SQL Server 2008

เอกรินทร์ วทัญญูเลิศสกุล

วท.ม. วิทยาการสารสนเทศ

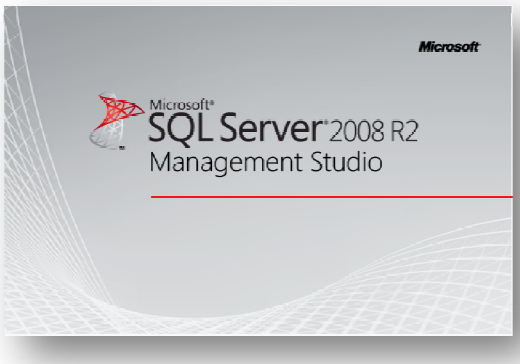
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

บทนำ

Microsoft SQL Server 2008

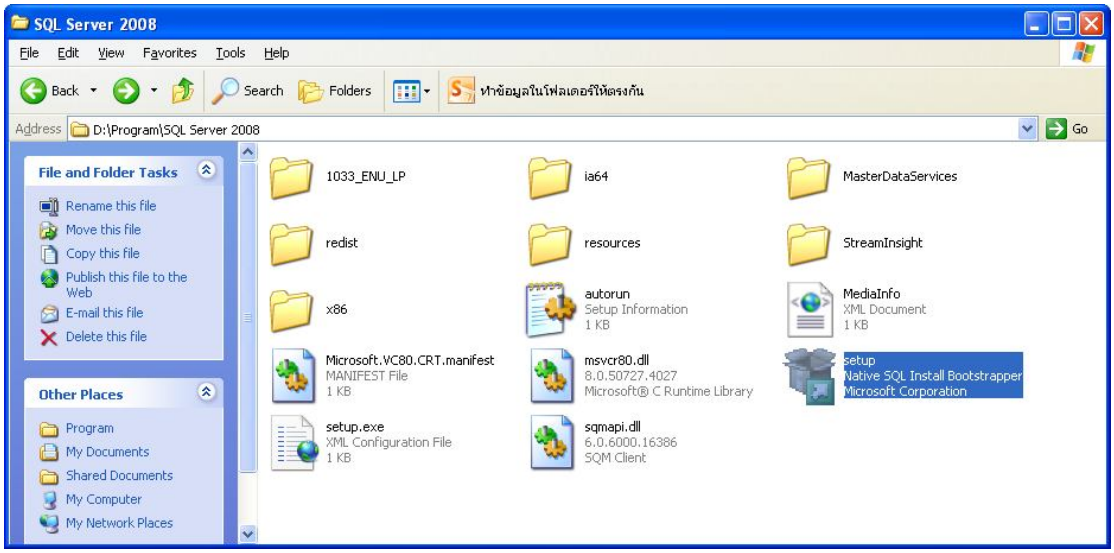
วัตถุประสงค์

1. เข้าใจรูปแบบคำสั่งและการใช้งาน Insert, Update, Delete และ Select
2. สามารถสร้างฐานข้อมูล และตาราง รวมทั้งการเขียนคำสั่ง SQL และวิธีการประมวลผลคำสั่งด้วยโปรแกรม SQL Server Management Studio (SSMS) ได้



การติดตั้งโปรแกรม SQL Server 2008

1. การติดตั้งโปรแกรม SQL Server 2008 ในกรณีที่ใช้แผ่นติดตั้งอัตโนมัติให้ข้ามไปขั้นตอนที่ 2 แต่สำหรับการติดตั้งในกรณีที่ไม่ได้ใช้แผ่นติดตั้งอัตโนมัติ ให้เปิดโฟลเดอร์ที่เก็บโปรแกรมติดตั้งไว้ และดับเบิ้ลคลิ้กที่ไฟล์Setup.Exe ดังรูป



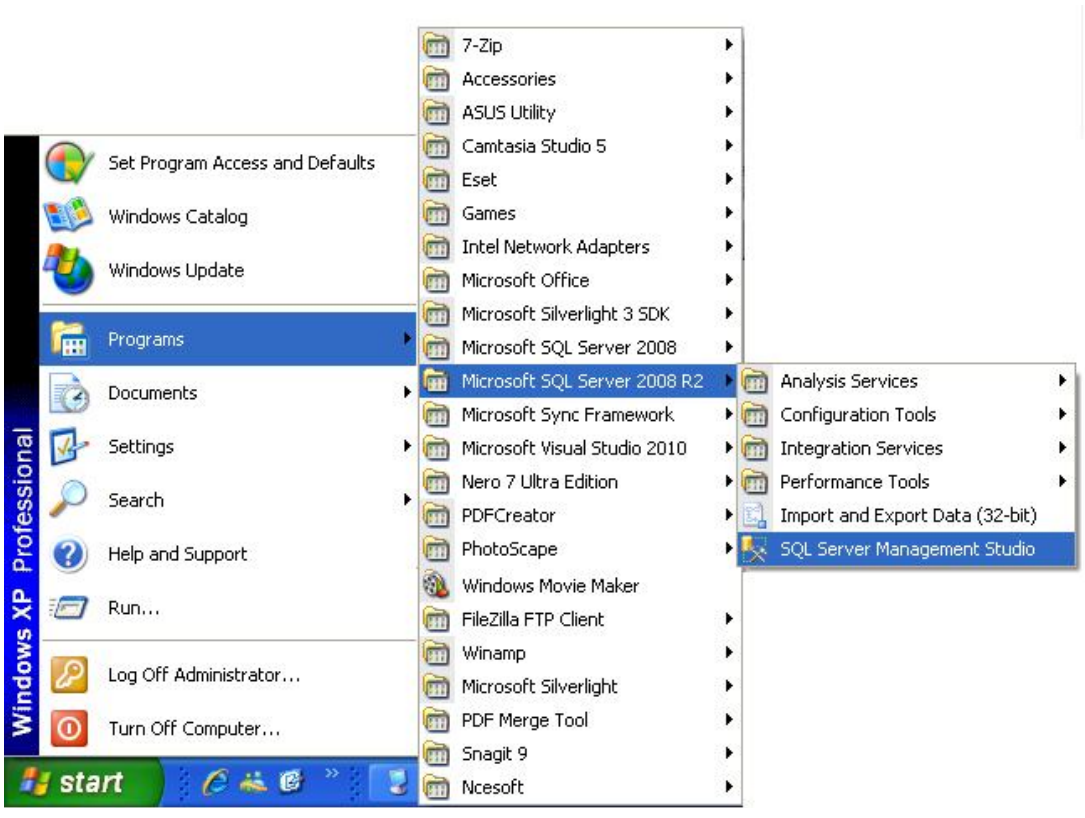
SQL Server Management Studio

SQL Server ในรุ่นเวอร์ชันก่อน 2005 จะมีโปรแกรม SQL Server Analyzer ทำหน้าที่ในการเขียนคำสั่งต่างๆ ของ SQL ซึ่งเครื่องมือที่สำคัญมากของโปรแกรมเมอร์หรือผู้พัฒนาโปรแกรม เนื่องจากในการเขียนโปรแกรมขั้นสูงที่เกี่ยวข้องกับการสืบค้นข้อมูล จำเป็นต้องทดสอบหรือเรียนรู้คำสั่งโปรแกรมผ่านเครื่องมือนี้ แต่สำหรับ SQL Server 2008 ได้รวมเครื่องมือดังกล่าวนี้ไว้ในชื่อผลิตภัณฑ์ใหม่คือ SQL Server Management Studio หรือ SSMS

SSMS เป็นสภาพแวดล้อมการทำงานสำหรับการสำหรับการเข้าถึงข้อมูล(accessing), การกำหนดค่า (configuring), การจัดการ (managing), การบริหาร (administering) และการพัฒนา (developing) ทุกองค์ประกอบของ SQL Server โดย SSMS ได้รวบรวมเครื่องมือต่างๆ ในรูปแบบของรูปภาพกราฟิก และนอกจาก SSMS จะมี Query Analyzer แล้วยังได้รวมโปรแกรม SQL Manager และจัดการการวิเคราะห์ต่างๆ ที่อยู่ในรุ่นก่อนหน้าของ SQL Server ไว้ในสภาพแวดล้อมเดียว

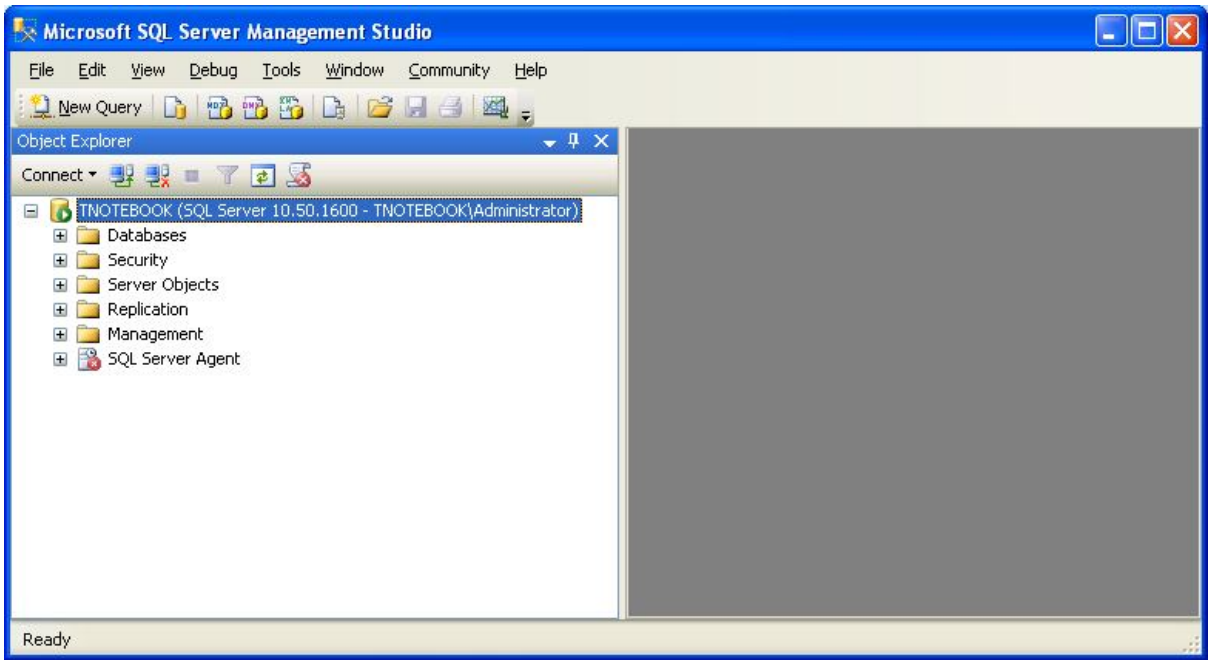
การเรียกใช้งาน SSMS สามารถดำเนินการได้ดังนี้

1. คลิ้กที่ปุ่ม Start\Programs\Microsoft SQL Server 2008 R2
2. คลิ้ก SQL Server Management Studio



3. พิมพ์ชื่อ Server Name (ปกติจะเป็นชื่อของเครื่องคอมพิวเตอร์) แล้วคลิ้กปุ่มConnect

4. หากชื่อ Server Name ถูกต้อง จะปรากฏหน้าจอดังรูป



ภาษา SQL

Standard relational database Query Language (SQL) เป็นภาษามาตรฐานสำหรับระบบฐานข้อมูลคือ ภาษา Standard relational database Query Language หรือเอสคิวแอล (SQL) หรือซีควอล (SE-QUEL) ซึ่งเป็นภาษาที่พัฒนาขึ้นมาโดยบริษัทไอบีเอ็ม

ภาษา SQL (Standard Query Language) เป็นส่วนหนึ่งของระบบฐานข้อมูลแบบรีเลชั่นเนล (Relational Database) ที่ได้รับความนิยมมากเพราะง่ายต่อความเข้าใจ และอยู่ในรูปภาษาอังกฤษ

ประเภทคำสั่งของ SQL

- คำสั่งกำหนดประเภทข้อมูล (Data Definition Language Command: DDL)

  เป็นกลุ่มคำสั่งที่ใช้สำหรับการปรับเปลี่ยนโครงสร้างของฐานข้อมูล ประกอบด้วยคำสั่ง Create, Replace, Alter, Truncate, Rename, Drop

- คำสั่งในการควบคุมโครงสร้างข้อมูล (Data Control Language Command: DCL)

  ประกอบด้วยคำสั่งที่ใช้ในการควบคุม การเกิดภาวะพร้อมกัน หรือการป้องกันการเกิดเหตุการณ์ที่ผู้ใช้หลายคนเรียกใช้ข้อมูลพร้อมกัน และคำสั่งที่เกี่ยวข้องกับการควบคุมความปลอดภัยของข้อมูลด้วยการกำหนดสิทธิของผู้ใช้ที่แตกต่างกัน เช่น คำสั่ง Grant และ Revoke

- คำสั่งในการปรับปรุงข้อมูล (Data Manipulation Language Command: DML)

  ประกอบด้วยคำสั่งที่ใช้ในการเรียกใช้ ข้อมูล การเปลี่ยนแปลงข้อมูล การเพิ่มหรือลบข้อมูล ซึ่งได้แก่คำสั่ง Insert, Delete และ Update

- คำสั่งที่ใช้ในการค้นหาข้อมูล (Data Retrieval Command)

  มีหน้าที่ในการค้นหาข้อมูล เพื่อแสดงรายการข้อมูล หรือคำนวณ โดยมีคำสั่งเพียงคำสั่งเดียวนั่นคือ คำสั่ง Select

- คำสั่งในการควบคุมการทำรายการข้อมูล (Transaction Control Command)

  เป็นคำสั่งที่ใช้ในการยืนยันรายการต่างๆ ที่ผู้ใช้งานได้กระทำกับฐานข้อมูล โดยคำสั่งในกลุ่มนี้จะประกอบด้วย “Commit” และ “Rollback”

คำสั่ง Insert

เราสามารถใช้คำสั่ง INSERT ในการเพิ่มข้อมูลต่อท้ายตารางได้โดยการใช้คำสั่งINSERT ร่วมกับวลี VALUEซึ่งมีรูปแบบดังนี้

```
INSERT INTO tablename (column1, column2, column..n)
VALUES (Value1, Value2, Value..n )
```

เมื่อ tablename คือ ชื่อของตารางที่จะทำการเพิ่มข้อมูล
column คือ ชื่อของคอลัมน์ที่จะทำการเพิ่มข้อมูล
value คือ ค่าต่างๆที่จะเพิ่มให้กับคอลัมน์นั้นๆ

ตัวอย่างเช่น

```
Insert Into Student (ID, Name, Age, Tel) Values ('1020', 'Egkarin', 28, '089001234')
```

คำสั่ง Update

คำสั่ง Update ใช้สำหรับการปรับปรุงข้อมูลเดิม โดยสามารถกำหนดขอบเขตในการแก้ไขข้อมูลได้โดยใช้คำสั่ง Where โดยกาไม่มีการกำหนดเงื่อนไขที่ใช้ในคำสั่ง Update จะหมายถึงการแก้ไขข้อมูลในทุกรายการ (Record) ซึ่งมีรูปแบบในการใช้งาน ดังนี้

```
Update tablename Set column1 = value1 [, column2 = value2,…] [Where condition]
```

เมื่อ tablename คือ ชื่อของตารางที่จะทำการเพิ่มข้อมูล
column คือ ชื่อของคอลัมน์ที่จะทำการเพิ่มข้อมูล
value คือ ค่าต่างๆที่จะเพิ่มให้กับคอลัมน์นั้นๆ
condition คือเงื่อนไขที่ระบุขอบเขตในการแก้ไขรายการในตาราง

ตัวอย่างเช่น

```
'แก้ไขชื่อและหมายเลขโทรศัพท์ของตาราง student ที่มีรหัส (ID) เท่ากับ '1020'
Update student  Set Name = 'เอกรินทร์' , Tel = '045123456' Where ID = '1020'

'แก้ไขอายุของทุกรายการในตาราง student ให้มีค่าเป็น 20
Update student Set Age = 20
```

คำสั่ง Delete

เป็นคำสั่งที่ใช้สำหรับลบรายการข้อมูลภายในตาราง สามารถกำหนดขอบเขตที่ใช้สำหรับลบรายการได้เช่นเดียวกับคำสั่ง Update โดยมีรูปแบบการใช้งานดังนี้

```
Delete From tablename [Where condition]
```

เมื่อ tablename คือ ชื่อของตารางที่จะทำการเพิ่มข้อมูล

condition คือเงื่อนไขที่ระบุขอบเขตในการแก้ไขรายการในตาราง

ตัวอย่างเช่น

```
Update student Set Name = 'เอกรินทร์' , Tel = '045123456' Where ID = '1020'
```

คำสั่ง Select

เป็นคำสั่งที่มีความสำคัญอย่างมากที่สุดในกลุ่มของคำสั่งSQL เนื่องจากจะเป็นเสมือนเครื่องมือของผู้ใช้งานระบบฐานข้อมูล สามารถแสดงผลลัพธ์ในรูปแบบรายการ ทำให้ทราบได้ว่าข้อมูลที่มีอยู่ในฐานข้อมูลมีอะไรบ้าง ข้อมูลที่ได้บันทึก แก้ไข หรือลบนั้นเป็นอย่างไร นอกจากนี้ยังสามารถใช้เพื่อการคำนวณ หรือประมวลผลในรูปแบบต่างๆ ได้อีกด้วย

รูปแบบของคำสั่ง Select มีดังนี้

```
SELECT select_list
FROM table_source
[ WHERE search_condition ]
[ GROUP BY group_by_expression ]
[ HAVING search_condition ]
[ ORDER BY order_expression [ ASC | DESC ] ]
```

เมื่อ select_list คือ รายการคอลัมน์ที่ต้องการแสดงผล

table_source คือ ชื่อตาราง

search_condition คือ เงื่อนไขที่ใช้กำหนดขอบเขตการค้นหารายการข้อมูล

group_by_expression คือ คอลัมน์ที่ใช้ในการจัดทำ group by

order_expression คือ คอลัมน์ที่ใช้ในการจัดเรียงข้อมูล

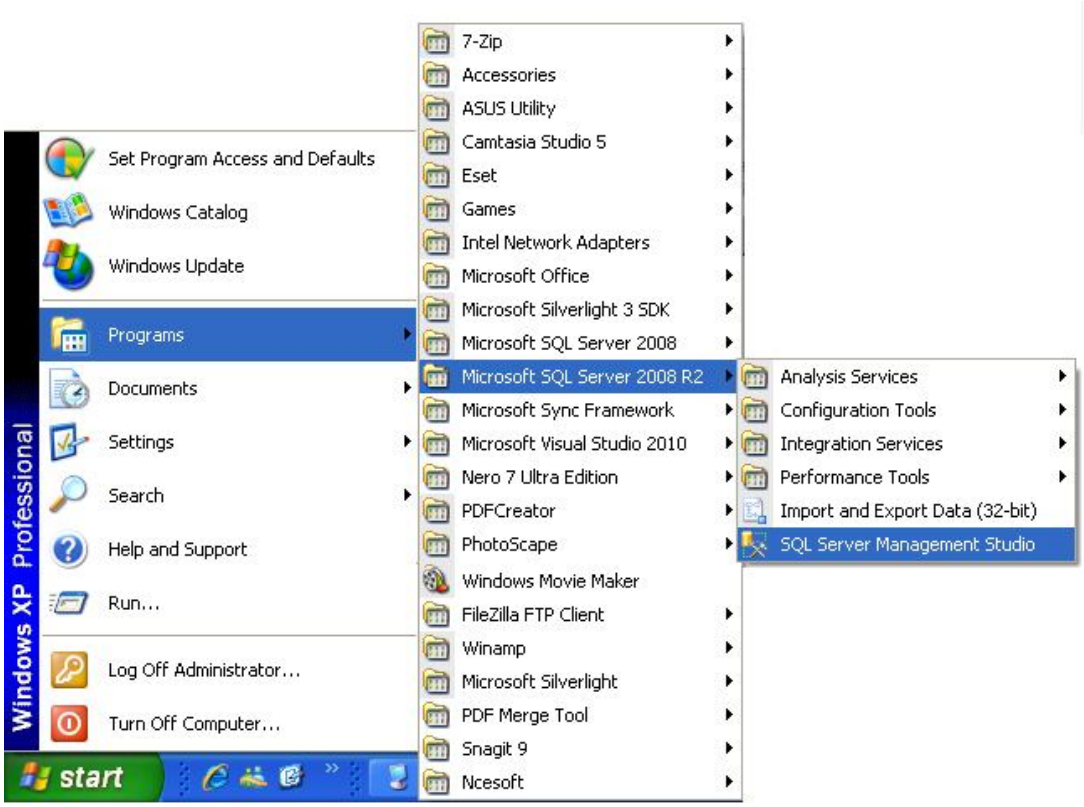
ตัวอย่างเช่น

```
Select * From student ;              'แสดงทุกรายการทุกคอลัมน์จากตาราง student
Select name, age From student Where age > 15   'แสดงคอลัมน์ name และ age ที่มีอายุน้อยกว่า 15
```

การสร้างฐานข้อมูล

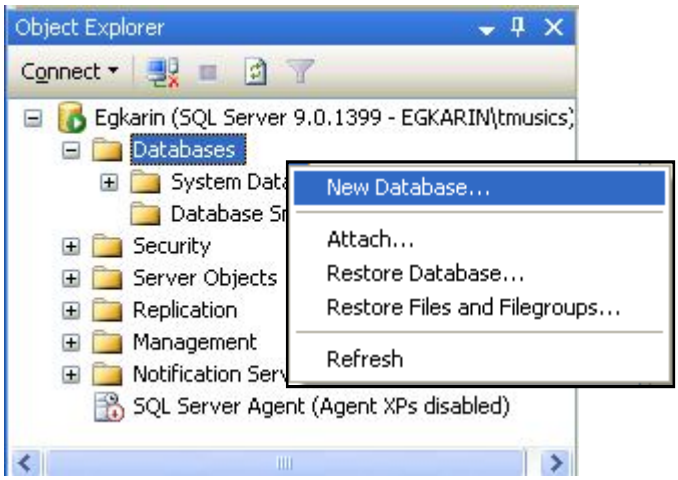
1. เปิดโปรแกรม **SQL Server Management Studio**

Start -> Programs -> Microsoft SQL Server 2008 R2 -> SQL Server Management Studio



2. คลิ้กปุ่มConnect

3. คลิ้กเมาส์ขวาที่Databases ที่หน้าต่าง Object Explorer แล้วเลือก New Database…



4. ตั้งชื่อของฐานข้อมูล แล้วคลิ้กปุ่มOK

การสืบค้นข้อมูล

Selecting Rows

วัตถุประสงค์

1. สามารถเขียนชุดคำสั่งสอบถามข้อมูลด้วยคำสั่ง Select
2. สามารถเขียนชุดคำสั่งเพื่อดำเนินการทางคณิตศาสตร์ได้
3. สามารถระบุชื่อคอลัมน์โดยใช้นามแฝง (Aliases) ได้
4. สามารถเชื่อมต่อคอลัมน์ได้

รูปแบบพื้นฐานคำสั่งสอบถามข้อมูล

คำสั่งสอบถามข้อมูลหรือคิวรี่ (Query) คือคำสั่งที่ใช้ในการสืบค้นผลลัพธ์จากฐานข้อมูล ซึ่งคำสั่งที่ใช้มีคำสั่งเดียวนั่คือคำสั่ง Select มีรูปแบบการใช้งานพื้นฐานดังนี้

SELECT	[*Distinct*]	{* , *column [alias]* , …}
FROM	*table*	

เมื่อ

SELECT	คำสั่งที่ใช้ในการสืบค้นข้อมูล ต้องกำหนดคอลัมน์อย่างน้อย 1 คอลัมน์
Distinct	ระงับการแสดงรายการที่มีข้อมูลซ้ำ
*	แสดงทุกคอลัมน์
column	แสดงข้อมูลตามชื่อของคอลัมน์
alias	ให้ชื่อที่ปรากฏในส่วนหัวของคอลัมน์เป็นไปตามกำหนด
table	ชื่อตารางที่ประกอบไปด้วยคอลัมน์ตามที่กำหนดให้แสดงผล

ข้อกำหนดการเขียนคำสั่ง SQL

การเขียนคำสั่ง SQL เพื่อสอบถามข้อมูลหรือจัดการข้อมูลในตารางสมารถทำได้ง่ายๆ โดยมีข้อกำหนดในการเขียนคำสั่งดังนี้

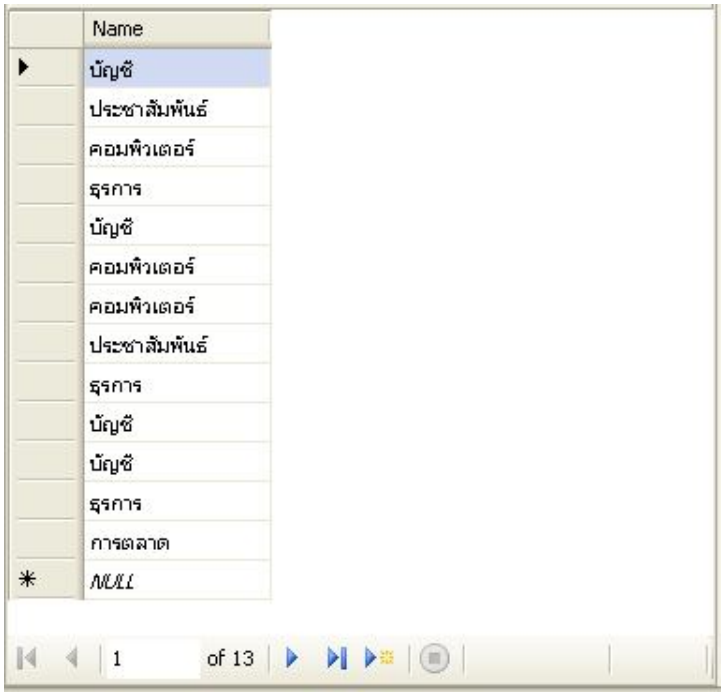
1. การเขียนคำสั่ง SQL อาจจะเขียนภายใน 1 บรรทัด หรือหลายบรรทัดได้
2. ส่วนของคำสั่ง SQL สามารถแยกเขียนอีกบรรทัดได้ เพื่อให้ง่ายต่อการอ่านและแก้ไข

3. การใช้ปุ่มแท็บ (Tab) ในการพิมพ์คำสั่งจะช่วยทำให้อ่านโค้ดได้ง่ายมากขึ้น
4. การขึ้นบรรทัดใหม่ไม่ถือเป็นการแบ่งคำสั่งหรือแยกชื่อย่อออกจากกัน
5. ตัวพิมพ์ใหญ่และตัวพิมพ์เล็กไม่มีผลต่อคำสั่ง SQL (Non Case Sensitive)
6. ใช้เครื่องหมายเซมิคอลอน (Semicolon) “ ; ” วางท้ายประโยคคำสั่ง เมื่อต้องการเขียนคำสั่ง SQL มากกว่ากว่า 1 ประโยค

ตัวอย่าง คำสั่งสอบถามข้อมูลเพื่อแสดงชื่อของแผนกทั้งหมดจากตาราง Depart

```
SELECT   Name
FROM    Depart
```

ผลลัพธ์ที่ได้



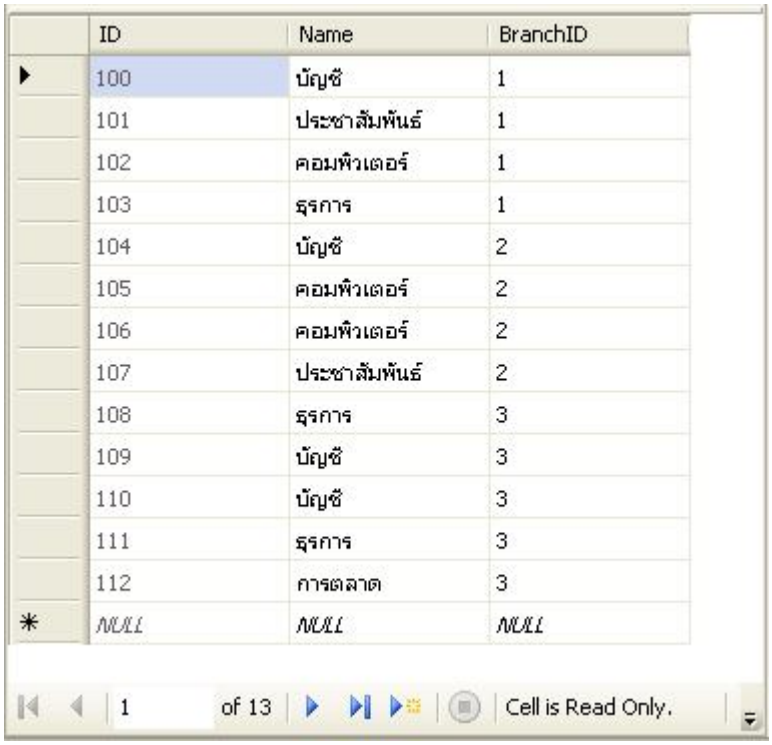
การแสดงข้อมูลทั้งตาราง (Selecting All Columns and Rows)

การสืบค้นเพื่อแสดงรายการทุกคอลัมน์และทุกแถว ใช้เครื่องหมายดอกจันทร์ “ * ” แทนคอลัมน์ทั้งหมดที่ต้องการแสดง

ตัวอย่าง การสืบค้นเพื่อแสดงรายการทั้งหมดทุกคอลัมน์และทุกแถวจากตาราง Depart

```
SELECT  *
FROM  Depart
```

ผลลัพธ์ที่ได้



	ID	Name	BranchID
▶	100	บัญชี	1
	101	ประชาสัมพันธ์	1
	102	คอมพิวเตอร์	1
	103	ธุรการ	1
	104	บัญชี	2
	105	คอมพิวเตอร์	2
	106	คอมพิวเตอร์	2
	107	ประชาสัมพันธ์	2
	108	ธุรการ	3
	109	บัญชี	3
	110	บัญชี	3
	111	ธุรการ	3
	112	การตลาด	3
*	NULL	NULL	NULL

1 of 13 Cell is Read Only.

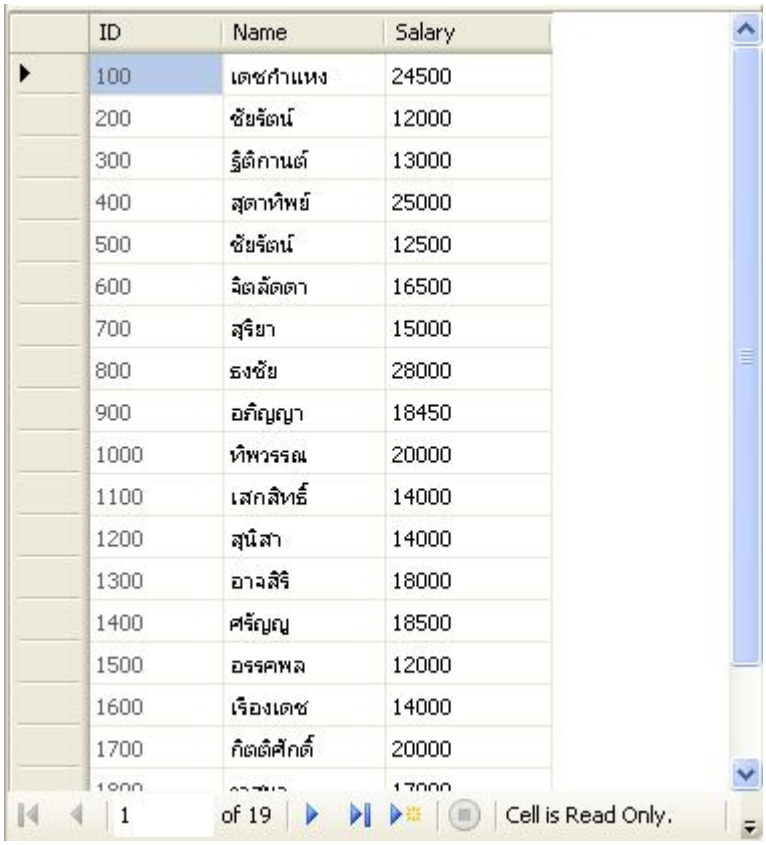
การกำหนดคอลัมน์ในการแสดงผล (Selecting Special Columns)

กำหนดคอลัมน์เพื่อการสืบค้นและแสดงผล สามารถกำหนดคอลัมน์ที่ต้องการแสดงผลจากตารางต่างๆ ได้ โดยระบุชื่อคอลัมน์ที่ต้องการหลังคำสั่ง Select และหากจำนวนคอลัมน์ที่ต้องการให้แสดงนั้นมีมากกว่า 1 คอลัมน์ สามารถเขียนชื่อคอลัมน์เหล่านั้นต่อกันได้โดยใช้เครื่องหมายจุลภาค “ , ” คั่นระหว่างชื่อคอลัมน์

ตัวอย่าง แสดงผล รหัส ชื่อ และเงินเดือน ของพนักงานทั้งหมดจากตาราง Employee

```
SELECT  ID , Name , Salary
FROM  Employee
```

ผลลัพธ์ที่ได้



ID	Name	Salary
100	เดชกำแหง	24500
200	ชัยรัตน์	12000
300	ฐิติกานต์	13000
400	สุดาทิพย์	25000
500	ชัยรัตน์	12500
600	จิตลัดดา	16500
700	สุริยา	15000
800	ธงชัย	28000
900	อภิญญา	18450
1000	ทิพวรรณ	20000
1100	เสกสิทธิ์	14000
1200	สุนิสา	14000
1300	อาจสิริ	18000
1400	ศรัญญู	18500
1500	อรรคพล	12000
1600	เรืองเดช	14000
1700	กิตติศักดิ์	20000

1 of 19 Cell is Read Only.

การคำนวณทางคณิตศาสตร์ (Arithmetic Expression)

ในการแสดงผลของข้อมูลบางครั้งอาจจำเป็นต้องคำนวณ หรือประมวลผลทางคณิตศาสตร์เพื่อได้คำตอบตามวัตถุประสงค์ที่ต้องการ ซึ่งภาษา SQL นั้นได้สนับสนุนการสืบค้นข้อมูลในลักษณะดังกล่าวได้เป็นอย่างดี โดยในการแสดงผลนั้นอาจประกอบด้วยชื่อของคอลัมน์ที่มีในตาราง ตัวเลข หรือเครื่องหมายหรือตัวดำเนินการทางคณิตศาสตร์ (Arithmetic Operation) ซึ่งเครื่องหมายการดำเนินการทางคณิตศาสตร์ได้แก่

ตัวดำเนินการ	คำอธิบาย
+	บวก
-	ลบ
*	คูณ
/	หาร

ตัวอย่าง แสดงผลรายได้ทั้งปีของพนักงาน โดยการคำนวณจากเงินเดือนจำนวนทั้งหมด 12 เดือน (เงินเดือน x 12) สามารถเขียนคำสั่งเพื่อแสดงผลได้ดังนี้

ตัวอย่าง แสดงการสืบค้นรายได้ทั้งปีของพนักงาน

```
SELECT    ID, Name, Salary, Salary * 12
FROM      Employee
```

ผลลัพธ์ที่ได้

	ID	Name	Salary	Expr1
▶	100	เดชคำแหง	24500	294000
	200	ชัยรัตน์	12000	144000
	300	ฐิติกานต์	13000	156000
	400	สุดาทิพย์	25000	300000
	500	ชัยรัตน์	12500	150000
	600	จิตลัดดา	16500	198000
	700	สุริยา	15000	180000
	800	ธงชัย	28000	336000
	900	อภิญญา	18450	221400
	1000	ทิพวรรณ	20000	240000
	1100	เสกสิทธิ์	14000	168000
	1200	สุนิสา	14000	168000
	1300	อาจสิริ	18000	216000
	1400	ศรัญญู	18500	222000
	1500	อรรคพล	12000	144000
	1600	เรืองเดช	14000	168000
	1700	กิตติศักดิ์	20000	240000

1 of 19 Cell is Read Only.

จากผลการรันคำสั่ง SQL ข้างต้น จะพบว่าคอลัมน์ Expr1 ไม่ใช่คอลัมน์ที่มีอยู่ในตาราง Employee แต่เป็นคอลัมน์ที่ถูกสร้างมาจากการคำนวณของคอลัมน์ Salary ที่คูณด้วย 12

ลำดับการคำนวณทางคณิตศาสตร์

ในการคำนวณทางคณิตศาสตร์นั้นหากประกอบด้วยตัวดำเนินการมากกว่า 1 ตัวขึ้นไป ให้ถือลำดับการ คูณ และ หาร เป็นเครื่องหมายที่ต้องกระทำการทางคณิตศาสตร์ก่อนเป็นลำดับแรก แต่

การจำกัดการสืบค้นข้อมูล

Limiting Selected Rows

วัตถุประสงค์

1. สามารถจัดเรียงข้อมูลโดยใช้ส่วนคำสั่ง Order By ได้
2. สามารถใช้เงื่อนไขในการสืบค้นข้อมูลได้โดยใช้ส่วนคำสั่ง Where ได้

การจัดเรียงข้อมูลโดยใช้ส่วนคำสั่ง Order By

ในการสืบค้นข้อมูล ผู้ใช้งานสามารถกำหนดการจัดเรียงข้อมูลได้เพื่อให้เหมาะสมต่อการนำไปใช้งาน โดยสามารถใช้ส่วนคำสั่ง Order By วางไว้ในส่วนท้ายของประโยคคำสั่งสอบถามข้อมูล (SQL Query) โดยรูปแบบของการจัดเรียงข้อมูลด้วย Order By แสดงได้ดังนี้

```
Select  expr
From  table
[ Order By {colum , expr} [ASC/DESC] ]
```

เมื่อ	expr	คือ นิพจน์ (Expression) อาจประกอบไปด้วยคอลัมน์ และค่าคงที่ต่าง ๆ (ตัวเลข, ข้อความหรือตัวอักษร, วันที่และเวลา), ค่าว่าง เพื่ออ้างอิงไปยังชื่อคอลัมน์ของตารางนั้น ๆ
	Order By	คือส่วนของคำสั่งที่ทำหน้าที่กำหนดรูปแบบการจัดเรียงข้อมูลเพื่อการแสดงผล
	ASC	รูปแบบการจัดเรียงข้อมูลแบบน้อยไปมาก
	DESC	รูปแบบการจัดเรียงข้อมูลแบบมากไปน้อย

ตัวอย่าง การสืบค้นข้อมูลจากตาราง Employee โดยกำหนดการแสดงคอลัมน์ นามสกุล, รหัสหน่วยงาน และรหัสหัวหน้างาน และจัดเรียงข้อมูลตามชื่อพนักงาน

```
SELECT    Name, Surname, Salary
FROM      Employee
ORDER BY Name
```

Name	Surname	Salary
กิตติศักดิ์	สายแวว	20000
จิตลัดดา	สายเสมอ	16500
ชัยรัตน์	มั่นชาติ	12000
ชัยรัตน์	อาจเอี่ยม	12500
ฐิติกานต์	จันธานี	13000
เดชคำแหง	จอมยุ	24500
ทิพวรรณ	บุญเพียง	20000
ธงชัย	มั่นชาติ	28000
เรืองเดช	บันไดทอง	14000
วาสนา	สีตางาม	17000
ศรัญญู	ทองคุณ	18500
ศิริวรรณ	เจริญชัย	16500
สุดาทิพย์	คำโพธิ์ทอง	25000

1 of 20

การจัดเรียงข้อมูลโดยเบื้องต้นจะกำหนดให้มีการจัดเรียงแบบ เรียงรายชื่อจากค่าน้อยไปมาก (Ascending : ASC) กล่าวคือ

- ค่าตัวเลข จะแสดงผลจากค่าน้อยไปหาค่ามาก
- วันที่ จะแสดงรายการเรียงจากวันที่แรกไปหาวันที่สุดท้าย เช่น วันที่ 11 ม.ค. 2552 มาก่อนวันที่ 5 ม.ค. 2554
- ตัวอักษรหรือข้อความ จะเรียงข้อมูลจากตัวพยัญชนะแรกไปหาพยัญชนะตัวทุดท้าย เช่น A-Z , ก-ฮ
- ค่าว่าง (Null) จะเป็นค่าที่ถูกเรียงไว้อันดับแรกของข้อมูลทั้งหมด ตัวอย่างเช่น Order By Salary

ตัวอย่าง การสืบค้นข้อมูลจากตาราง Employee โดยกำหนดการแสดงคอลัมน์ นามสกุล, รหัสหน่วยงาน และรหัสหัวหน้างาน และจัดเรียงข้อมูลตามเงินเดือนพนักงาน โดยมีการกำหนดรูปแบบการจัดเรียงแบบ ASC

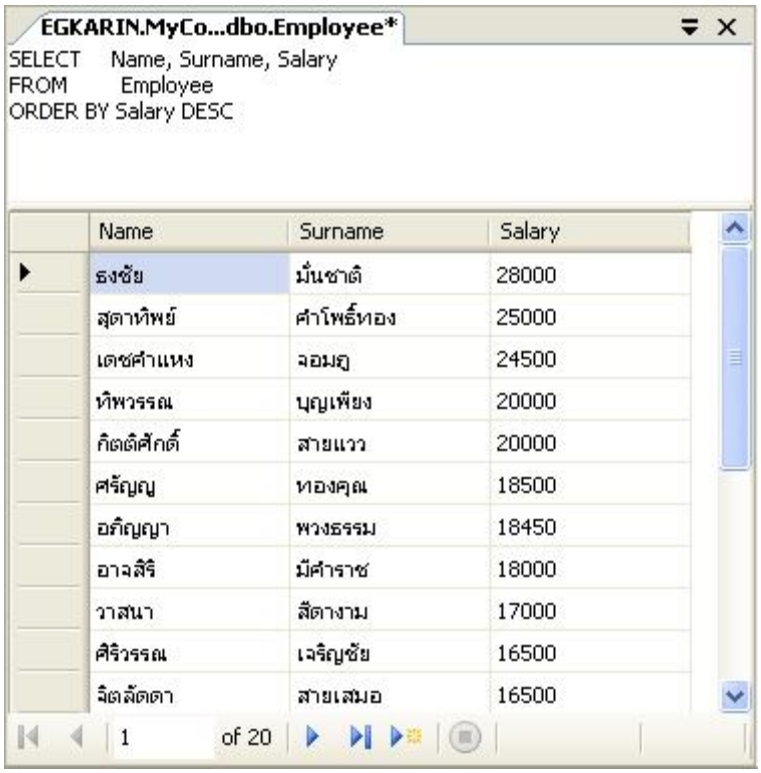
```
SELECT    Name, Surname, Salary
FROM      Employee
ORDER BY Salary ASC
```

Name	Surname	Salary
เอกรินทร์	วทัญญูเลิศสกุล	*NULL*
ชัยรัตน์	มั่นชาติ	12000
อรรถพล	พูลวัน	12000
ชัยรัตน์	อาจเอี่ยม	12500
ฐิติกานต์	จันธานี	13000
เรืองเดช	บันไดทอง	14000
เสกสิทธิ์	อารยมาตย์	14000
สุนิสา	สมบุญ	14000
สุริยา	กอแก้ว	15000
จิตลัดดา	สายเสมอ	16500
ศิริวรรณ	เจริญชัย	16500
วาสนา	สีดางาม	17000
อาจสิริ	มีศาราช	18000

1 of 20

จะพบว่าข้อมูลได้ถูกจัดเรียงการแสดงผลใหม่โดยเรียงจากพนักงานที่มีเงินเดือนน้อยที่สุดไปหาคนที่มีเงินเดือนมากที่สุด สำหรับพนักงานที่มีเงินเดือนเป็นค่าว่างจะถูกนำมาจัดเรียงไว้ในอับแรกสุด ดังนั้นการจัดเรียงลำดับข้อมูลจากค่าน้อยไปหามาก แม้จะไม่ได้เติมข้อความ ASC ตามท้ายคอลัมน์ที่จะจัดเรียง ข้อมูลก็จะถูกเรียงลำดับแบบน้อยไปหามากโดยอัตโนมัติ

ส่วนการจัดเรียงข้อมูลที่ต้องการสืบค้น โดยเรียงลำดับจากค่ามากไปน้อย จะกำหนดคำสั่ง DESC ต่อท้ายคอลัมน์ที่ต้องการจัดเรียง ดังตัวอย่าง



การจัดเรียงข้อมูลโดยใช้เลขลำดับ

การจัดเรียงข้อมูลในกรณีที่ชื่อของคอลัมน์มีความยาวหรือไม่สะดวกในการพิมพ์ สามารถใช้เลขลำดับของตำแหน่งการแสดงผลของคอลัมน์ เพื่อกำหนดการจัดเรียงตามรูปแบบที่ต้องการได้ เช่น

```
SELECT    Name,    Surname,    Salary
FROM      Employee
ORDER BY  2
```

ฟังก์ชัน

Function

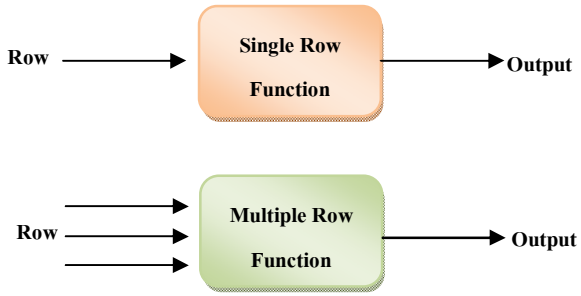
วัตถุประสงค์

1. เข้าใจรูปแบบการใช้งานฟังก์ชัน (Function)
2. สามารถนำฟังก์ชันมาประยุกต์ใช้งานในการสืบค้นข้อมูลได้อย่างถูกต้อง

ฟังก์ชัน

ฟังก์ชันหมายถึงกลุ่มคำสั่งหรือโปรแกรมย่อยที่เมื่อมีการเรียกใช้งานแล้วจะมีค่าคืนกลับมา โดยทั่วไปจะประโยชน์เพื่อการคำนวณ ซึ่งบางฟังก์ชันอาจต้องการค่าข้อมูลเพื่อใช้ในการประมวลผลของฟังก์ชัน เช่น ฟังก์ชันการหาผลรวม ฟังก์ชันเลขยกกำลัง เป็นต้น บางฟังก์ชันก็อาจไม่ต้องการข้อมูลในการประมวลผล เช่น ฟังก์ชัน PI (ค่าพายเป็นค่าคงตัวทางคณิตศาสตร์ ที่เกิดจากความยาวเส้นรอบวงหารด้วยเส้นผ่านศูนย์กลางของวงกลม มีค่าเท่ากับ 3.14159…) , ฟังก์ชัน Rand ซึ่งใช้สำหรับสุ่มค่าตัวเลข เป็นต้น

ฟังก์ชันในภาษา SQL แบ่งออกได้ 2 กลุ่ม ใหญ่คือ ฟังก์ชันแบบแถวเดียว (Single Row Function) และฟังก์ชันแบบหลายแถว (Multiple Row Function)



อาร์กิวเมนต์ (Argument)

อาร์กิวเมนต์ คือค่าที่ป้อนให้แก่ฟังก์ชันเพื่อใช้ในการประมวลผลหรือคำนวณของฟังก์ชัน ซึ่งข้อมูลที่เป็นอาร์กิวเมนต์จะถูกกำหนดตามคุณสมบัติของฟังก์ชันที่ต้องการ โดยบางฟังก์ชันอาจมีการกำหนดอาร์กิวเมนต์มากกว่า 1 ค่า หรืออาจไม่มีการกำหนดอาร์กิวเมนต์ก็ได้

รูปแบบ

Function_name (column | expression, [arg1, arg2, …])

เมื่อ Function_name คือ ชื่อของฟังก์ชัน

arg คือ อาร์กิวเมนต์

ตัวอย่างการใช้งานฟังก์ชัน Substring() เช่น Substring (PositionName , 3, 5)

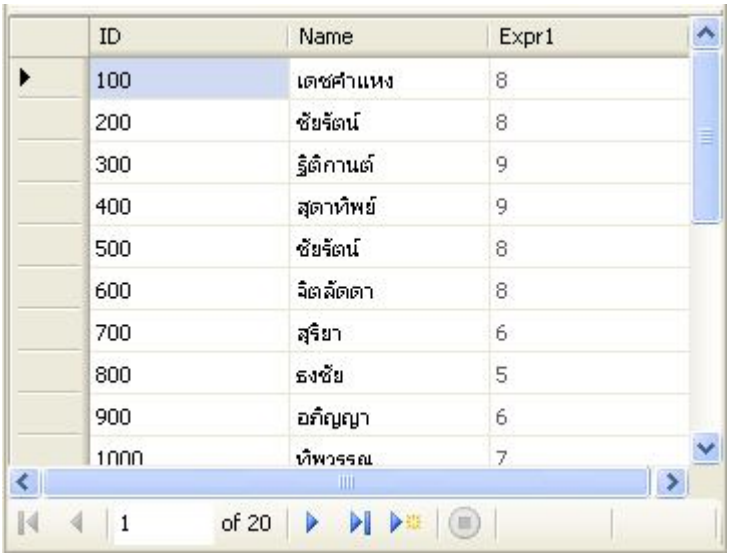
ฟังก์ชันแถวเดียว (Single Row Function)

คือฟังก์ชันที่มีคำนวณหรือประมวลผลข้อมูลทีละระเบียน (Record) หรือทีละแถว (Row) กล่าวคือฟังก์ชันแถวเดียวจะถูกประมวลผลเท่ากับจำนวนแถวของข้อมูลที่มีของตาราง โดยสามารถเขียนไว้เพื่อให้แสดงผลในรูปแบบของข้อมูลที่จะแสดงในการสืบค้นได้ ซึ่งจะต้องทำการเขียนไว้หลังคำสั่ง SELECT เช่น

ตัวอย่าง

```
SELECT    ID , NAME , LEN(NAME)
FROM      EMPLOYEE
```

ผลลัพธ์ที่ได้



	ID	Name	Expr1
▶	100	เตชคำแหง	8
	200	ชัยรัตน์	8
	300	ฐิติกานต์	9
	400	สุดาทิพย์	9
	500	ชัยรัตน์	8
	600	จิตลัดดา	8
	700	สุริยา	6
	800	ธงชัย	5
	900	อภิญญา	6
	1000	ทิพวรรณ	7

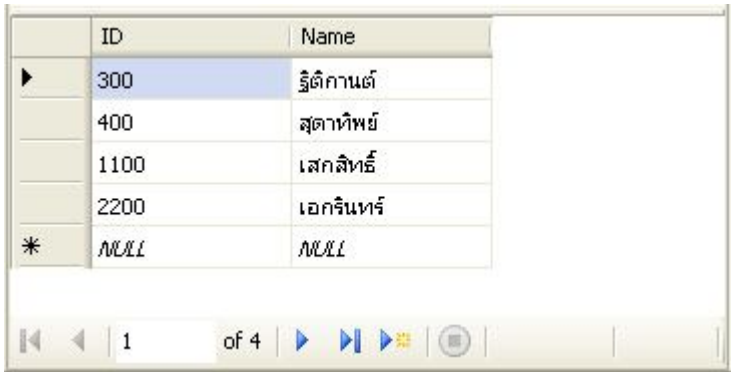
1 of 20

หรือยังอาจเขียนฟังก์ชันเพื่อใช้กำหนดเป็นเงื่อนไขในการสืบค้น โดยจะเขียนฟังก์ชันดังกล่าวไว้ในส่วนหลังของคำสั่ง Where เช่น

ตัวอย่าง

```
SELECT    ID , NAME
FROM      EMPLOYEE
WHERE     LEN(NAME) = 9
```

ผลลัพธ์ที่ได้



	ID	Name
▶	300	ฐิติกานต์
	400	สุดาทิพย์
	1100	เสกสิทธิ์
	2200	เอกรินทร์
*	NULL	NULL

1 of 4

ภาษา SQL มีฟังก์ชันแบบแถวเดียวเป็นจำนวนมาก แต่ในหนังสือเล่มนี้จะขอยกตัวอย่างการใช้งานฟังก์ชันเพียงบางส่วน เพื่อเป็นแนวทางในการศึกษาและการใช้งานการเขียนคำสั่งด้วยภาษา SQL โดยจะแบ่งประเภทของฟังก์ชันตามชนิดของข้อมูลที่ใช้ในการประมวลผล ดังนี้

- ฟังก์ชันข้อมูลชนิดตัวเลข
- ฟังก์ชันข้อมูลชนิดตัวอักษรหรือข้อความ
- ฟังก์ชันข้อมูลชนิดวันที่
- ฟังชันอื่นๆ

ฟังก์ชันข้อมูลชนิดตัวเลข

ฟังก์ชันที่ทำหน้าที่ในการจัดการข้อมูลชนิดตัวเลขได้แก่

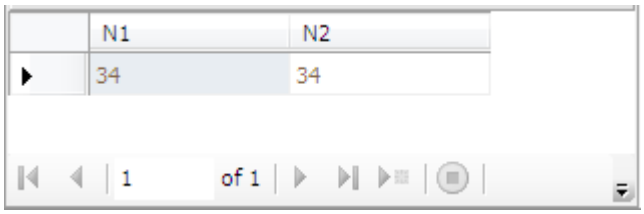
ABS	CEILING	FLOOR	RAND
ROUND	SQUARE	SQRT	PI
POWER			

ฟังก์ชัน ABS	
ความหมาย	คำนวณค่าสัมบูรณ์ (Absolute) คือผลต่างระหว่างจำนวนนั้นกับค่า 0
รูปแบบ	ABS (column | expression)
ตัวอย่าง	SELECT ABS (-34)

ตัวอย่าง

```
SELECT    ABS(-34) AS   N1 ,  ABS(34)  AS  N2
```

ผลลัพธ์ที่ได้

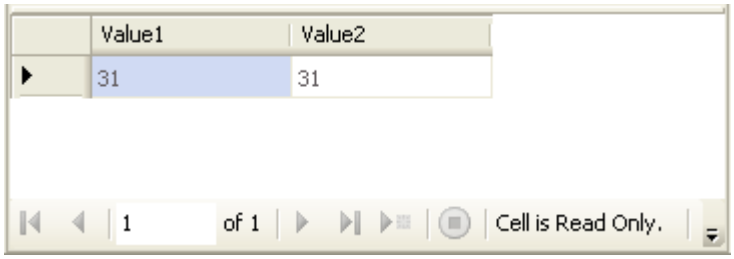


ฟังก์ชัน CEILING	
ความหมาย	คำนวณค่าแบบปัดเศษหลังจุดทศนิยมขึ้น
รูปแบบ	CEILING (column | expression)
ตัวอย่าง	CEILING (35.1)

ตัวอย่าง

```
SELECT    CEILING(30.01) AS Value1 , CEILING(30.95) AS Value2
```

ผลลัพธ์ที่ได้



การแสดงผลข้อมูลจากหลายตาราง

(Displaying Data from Multiple Tables)

วัตถุประสงค์

- สามารถเขียนคำสั่ง SELECT เพื่อเข้าถึงข้อมูลจากตารางได้มากกว่า 1 ตาราง แบบ Equity Join และ Non-Equality Join
- สามารถเรียกดูข้อมูลแบบ Outer Join ได้
- สามารถเชื่อมต่อตารางแบบ Self Join ได้

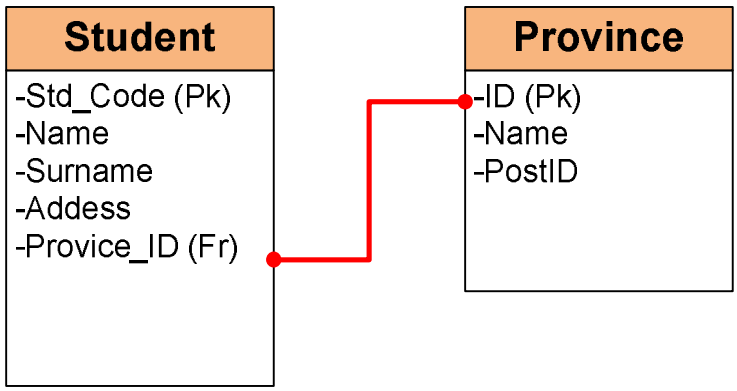
คีย์ (Key)

คีย์ คือ คอลัมน์ที่สามารถใช้ในการบ่งบอกความแตกต่างของข้อมูลแต่ละแถวหรือรายการได้ ซึ่งการกำหนดค่าของคีย์นั้นอาจเป็นได้ทั้งคอลัมน์เดี่ยวๆ หรือ กลุ่มของคอลัมน์ ตัวอย่างของคีย์ที่ใช้ในการกำหนดความสัมพันธ์ของตาราง ได้แก่

- คีย์หลัก (Primary Key) หมายถึง คอลัมน์ที่ทำหน้าที่แยกแยะความแตกต่างของข้อมูลในแต่ละรายการ เช่น เลขที่ รหัส โดยจะมีคุณสมบัติพิเศษคือ เป็นเป็นข้อมูลที่เป็นเอกลักษณ์เฉพาะ คือ เป็นคอลัมน์ที่มีค่าไม่ซ้ำกันในตาราง และไม่สามารถเป็นค่าว่าง (Null) ได้ โดยในแต่ละตารางอาจมีคอลัมน์ที่มีคุณสมบัติในลักษณ์ดังกล่าวนี้ได้มากกว่า 1 คอลัมน์ แต่จะมีเพียงคอลัมน์เดียวเท่านั้นที่จะถูกำหนดและทำหน้าที่เป็นคีย์หลัก
- คีย์คู่แข่ง(Candidate Key) คือ คอลัมน์ที่มีคุณสมบัติที่เหมือนกับคีย์หลักคือสามารถใช้แยกแยะข้อมูลแต่ละแถวได้ แต่ไม่ได้ถูกเลือกใช้เป็นคีย์หลัก
- คีย์นอก (Foreign Key) คือ คอลัมน์ที่เก็บค่าคีย์หลักจากตารางที่ต้องการเชื่อมโยง ใช้เพื่อเชื่อมโยงความสัมพันธ์ระหว่างตาราง

การเชื่อมโยงตาราง (Join)

การเชื่อมโยงตารางจะถูกนำมาใช้ในการสืบค้นข้อมูลจากตารางหลายๆ ตาราง โดยในแต่ละแถวของแต่ละตารางจะถูกเชื่อมโยงกันโดยใช้หลักการพื้นฐานของความสัมพันธ์ (Relation) ตามคุณสมบัติฐานข้อมูลเชิงสัมพันธ์ (Relational Database) คือ คีย์หลัก (Primary Key: Pk) และคีย์นอก (Foreign Key: Fr) ดังภาพ



จากภาพแสดงความสัมพันธ์ของตาราง 2 ตาราง คือตารางข้อมูลนักศึกษา (Student) และตารางจังหวัด (Province) ตารางทั้ง 2 มีความสัมพันธ์กันโดย ตารางจังหวัดมีคีย์หลักคือ ID สัมพันธ์กับตารางนักศึกษาที่คีย์นอกคือ Province_ID

ซึ่งลักษณะของการเขียนคำสั่ง SQL เพื่อเชื่อมโยงความสัมพันธ์ระหว่างตารางโดยทั่วไปจะสามารถแบ่งได้ 2 แบบคือ

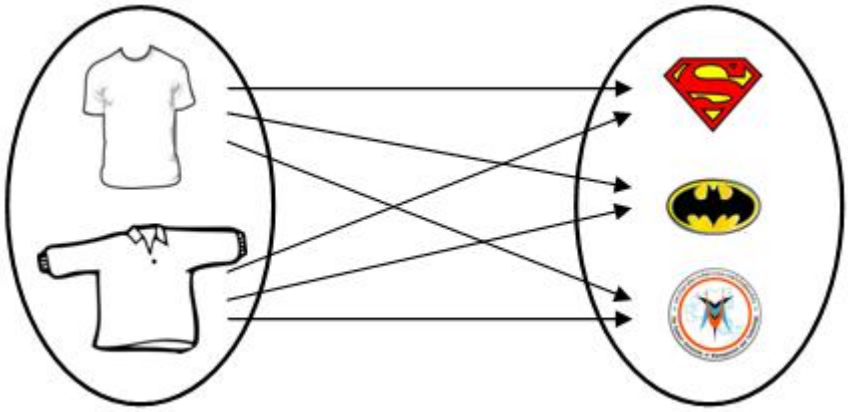
- Inner Join ได้แก่ Equijoin, Non-Equijoin, Cross Join, Self Join
- Outer Join ได้แก่ Left Outer Join, Right Outer Join, Full Outer Join

Cartesian Product (Cross Join)

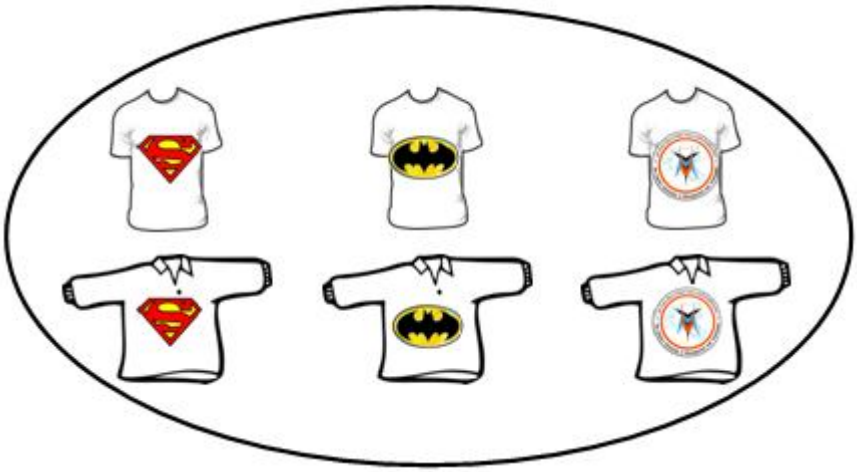
Cartesian Product คือ ข้อบังคับหรือข้อกำหนดในการอนุญาตให้เก็บเฉพาะข้อมูลที่เหมาะสมลงในฐานข้อมูล เพื่อให้การสืบค้นข้อมูลจากฐานข้อมูลอย่างมีความถูกต้อง

Cartesian Product จึงมีความสำคัญอย่างยิ่งในการสืบค้นข้อมูลที่มีความสัมพันธ์ของตารางตั้งแต่ 2 ตารางขึ้นไป ดังตัวอย่างการสืบค้นต่อไปนี้ เป็นการสืบค้นข้อมูลจาก 2 ตาราง โดยไม่ได้กำหนดลักษณะความสัมพันธ์ใดๆ อันส่งผลให้ข้อมูลซ้ำซ้อนกัน และขาดความน่าเชื่อถือ

ยกตัวอย่างเช่น เซตข้อมูลเสื้อ ประกอบด้วยข้อมูลจำนวน 2 ข้อมูล คือ เสื้อแขนสั้น และเสื้อแขนยาวส่วนเซตข้อมูลโลโก้สัญลักษณ์ ประกอบด้วยข้อมูลสัญลักษณ์ Super Man, Bat Man และ UMT เมื่อกำหนดให้กระทำการ Cartesian Product กับข้อมูลจากทั้ง 2 เซต ความสัมพันธ์ที่เกี่ยงข้องกันจะแสดงด้วยลูกศรดังรูป



ซึ่งจะได้ผลการดำเนินการ Cartesian Product จะทำให้ได้เซตข้อมูลที่มีจำนวนสมาชิกเท่ากับ 6 ข้อมูล ดังแสดงในรูป

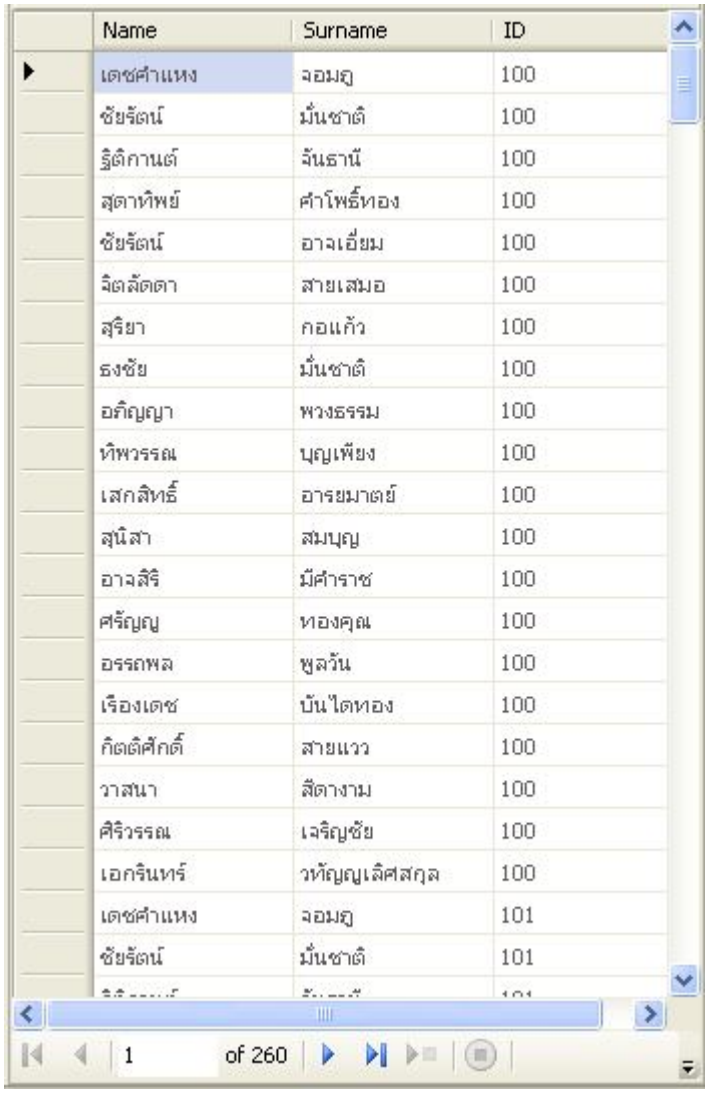


จากข้อมูลในตาราง Employee และตาราง Depart มีจำนวนรายการข้อมูลในตารางคือ 20 และ 13 รายการตามลำดับ สามารถสืบค้นข้อมูลจากความสัมพันธ์ของทั้ง 2 ตาราง ด้วยรูปแบบความสัมพันธ์ แบบ Cartesian Product ได้ดังนี้

ตัวอย่าง

```
SELECT    Name, Surname, DepartID
FROM      Employee ,  Depart
```

ผลลัพธ์ที่ได้



Name	Surname	ID
เตชคำแหง	จอมฐ	100
ชัยรัตน์	มั่นชาติ	100
ฐิติกานต์	จันธานี	100
สุดาทิพย์	คำโพธิ์ทอง	100
ชัยรัตน์	อาจเอี่ยม	100
จิตลัดดา	สายเสมอ	100
สุริยา	กอแก้ว	100
ธงชัย	มั่นชาติ	100
อภิญญา	พวงธรรม	100
ทิพวรรณ	บุญเพียง	100
เสกสิทธิ์	อารยมาตย์	100
สุนิสา	สมบุญ	100
อาจสิริ	มีคำราช	100
ศรัญญู	ทองคุณ	100
อรรถพล	พูลวัน	100
เรืองเดช	บันไดทอง	100
กิตติศักดิ์	สายแวว	100
วาสนา	สีดางาม	100
ศิริวรรณ	เจริญชัย	100
เอกรินทร์	วทัญญูเลิศสกุล	100
เตชคำแหง	จอมฐ	101
ชัยรัตน์	มั่นชาติ	101

1 of 260

จะพบว่าผลลัพธ์ที่ได้จากการสืบค้นมีจำนวนทั้งสิ้น260 รายการ ซึ่งจำนวนรายการที่ได้จากการสืบค้นนี้เกิดจากการนำข้อมูลจากทุกๆ รายการในตาราง Employee มาเชื่อมโยงเข้ากับข้อมูลทุกรายการจากตาราง Depart ด้วยความสัมพันธ์แบบ Cartesian Product ซึ่งหมายถึงข้อมูลจากทั้งสองตารางถูกนำมาจับคู่กันในทุกความเป็นไปได้ ดังนั้นจำนวนรายการที่ได้จากการสืบค้นจึงเกิดจากการนำจำนวนรายการจากทั้ง 2 ตารางมาคูณ (Product) กัน คือ 20 x 13 ซึ่งมีค่าเท่ากับ 260 รายการ

จากการสืบค้นด้วยวิธีการ Cartesian Product นี้จะพบว่าไม่สามารถนำข้อมูลที่ได้ไปใช้ประโยชน์ได้เนื่องจากข้อมูลที่ได้นั้นจะโยงความสัมพันธ์อย่างไม่มีหลักหรือกฎเกณฑ์ ด้วยเหตุนี้ในการสร้างตารางข้อมูลจึงจำเป็นต้องมีการจัดเก็บคีย์ไว้ในตาราง สำหรับกำหนดความเชื่อมโยง

ความสัมพันธ์ระหว่างตาราง ซึ่งคีย์ที่ใช้ในการกำหนดความเชื่อมโยงระหว่างตารางนี้ประกอบด้วยคีย์ 2 ประเภทคือ

การสืบค้นข้อมูลจากหลายตารางอย่างง่าย

การแสดงข้อมูลจากตารางที่มีความสัมพันธ์กันตั้งแต่ 2 ตารางขึ้นไปสามารถกำหนดเงื่อนไขเพื่อผูกความสัมพันธ์ของคอลัมน์หรือคีย์หลัก และคีย์นอกในแต่ละตารางได้ด้วย WHERE ดังรูปแบบ

รูปแบบ

SELECT *table.column , table.column ...*

FROM *table1 , table2*

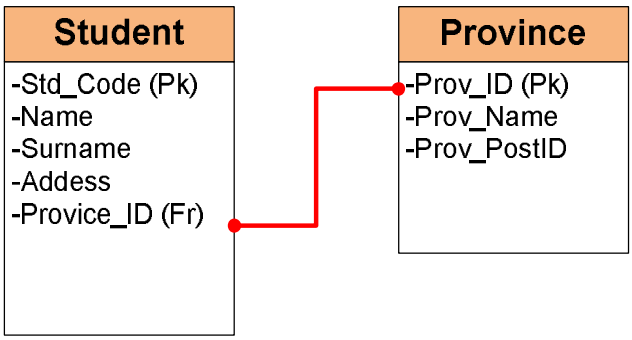
WHERE *table1.column_FR = table2.column_PK*

เมื่อ table.column คือ ชื่อตารางและชื่อคอลัมน์ ของข้อมูลที่ต้องการให้แสดงผล

table1.column_FR คือ ชื่อตารางและชื่อคอลัมน์คีย์นอก

table2.column_PK คือ ชื่อตารางและชื่อคอลัมน์คีย์หลัก

ยกตัวอย่างโครงสร้างฐานข้อมูลดังรูป



สามารถเขียนคำสั่ง SQL เพื่อแสดงข้อมูลของนักศึกษาพร้อมชื่อจังหวัดและรหัสไปรษณีย์ได้ดังนี้

ฟังก์ชันแบบกลุ่ม

Group Function

วัตถุประสงค์

1. เข้าใจรูปแบบการใช้งานฟังก์ชันแบบกลุ่ม (Group Function)
2. สามารถนำฟังก์ชันแบบกลุ่มมาประยุกต์ใช้งานในการสืบค้นข้อมูลได้อย่างถูกต้อง

ฟังก์ชันแบบกลุ่ม

ฟังก์ชันแบบกลุ่ม (Group Function) หรือ ฟังก์ชันแบบรวบรวม (Aggregation Function) หรือ ฟังก์ชันหลายแถว (Multi Row Function) คือฟังก์ชันที่เกิดจากการประมวลผลหรือคำนวณได้จากข้อมูลหลายๆ แถว เช่น การคำนวณยอดขายสินค้าประจำเดือน ราคาเฉลี่ยของสินค้าที่ซื้อมา (อาจมีการสั่งซื้อหลายครั้งมีราคาที่แตกต่างกัน) เป็นต้น โดยสามารถใช้งานฟังก์ชันเหล่านี้ร่วมกับส่วนของคำสั่ง Group By เพื่อให้เกิดผลลัพธ์แยกตามหมวดหมู่ หรือกลุ่มได้ ตัวอย่างเช่น การคำนวณหายอดรวมการขายสินค้าของพนักงานแต่ละคน คืออาจแสดงข้อมูลพนักงานทุกคนพร้อมผลรวมยอดขายที่สามารถขายได้ ซึ่งตัวอย่างของฟังก์ชันแบบหลายแถว มีดังนี้

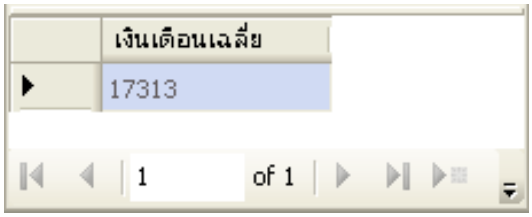
AVG	COUNT	MAX	MIN
SUM	STDEV		

ฟังก์ชัน AVG	
ความหมาย	คำนวณหาค่าเฉลี่ย
รูปแบบ	AVG (column | expression)
ตัวอย่าง	SELECT AVG (SALARY) FROM EMPOLYEE

ตัวอย่าง ใช้ฟังก์ชัน AVG เพื่อคำนวณหาค่าเงินเดือนเฉลี่ยของพนักงานทั้งหมด

```
SELECT   AVG (SALARY) AS  'เงินเดือนเฉลี่ย'
FROM  EMPOLYEE
```

ผลลัพธ์ที่ได้



ฟังก์ชัน COUNT	
ความหมาย	นับจำนวนรายการ
รูปแบบ	COUNT (column | expression)
ตัวอย่าง	SELECT Count(*) FROM Emp

ตัวอย่าง ใช้ฟังก์ชัน COUNT() เพื่อนับรายการทั้งหมดในตาราง

```
SELECT    COUNT(*)  AS   รายการทั้งหมด
FROM      Employee
```

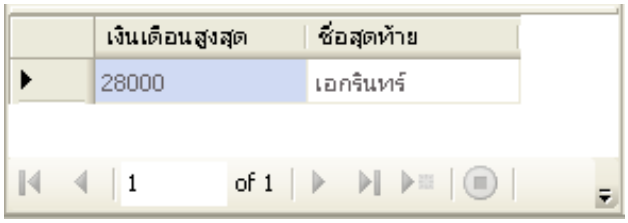
ผลลัพธ์ที่ได้

ฟังก์ชัน MAX	
ความหมาย	คำนวณหาค่าสูงสุดในแต่ละคอลัมน์
รูปแบบ	MAX (column | expression)
ตัวอย่าง	SELECT MAX(Salary) FROM Emp

ตัวอย่าง ใช้ฟังก์ชัน MAX() เพื่อหาค่าเงินเดือนสูงสุด และชื่อพนักงานคนสุดท้ายเมื่อเรียงตามลำดับตัวอักษร

```
SELECT   MAX (Salary) AS เงินเดือนสูงสุด , MAX (Name) AS ชื่อสุดท้าย
FROM     Employee
```

ผลลัพธ์ที่ได้

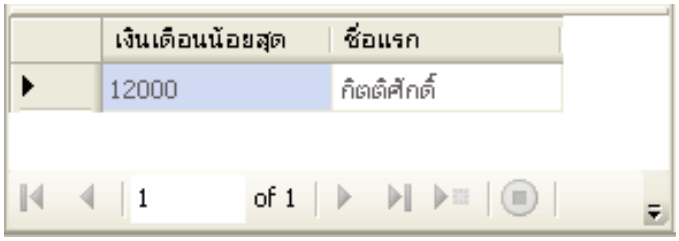


ฟังก์ชัน MIN	
ความหมาย	คำนวณหาค่าต่ำสุดในแต่ละคอลัมน์
รูปแบบ	MIN (column | expression)
ตัวอย่าง	SELECT MIN(Salary)

ตัวอย่าง ใช้ฟังก์ชัน MIN() เพื่อหาค่าเงินเดือนต่ำสุด และชื่อพนักงานคนแรกเมื่อเรียงตามลำดับตัวอักษร

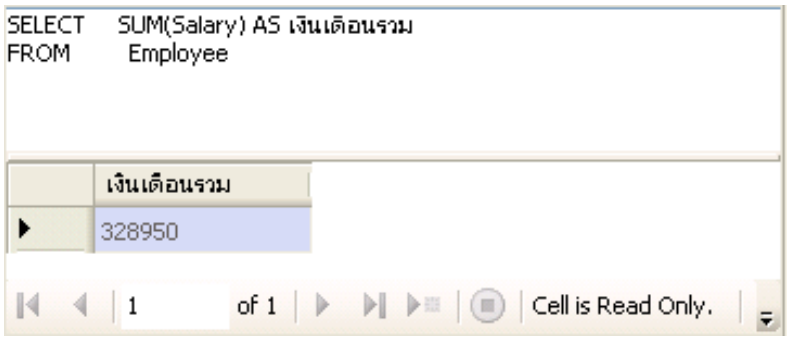
```
SELECT   MIN(Salary)  AS  เงินเดือนน้อยสุด , MIN(Name)  AS  ชื่อแรก
FROM     Employee
```

ผลลัพธ์ที่ได้



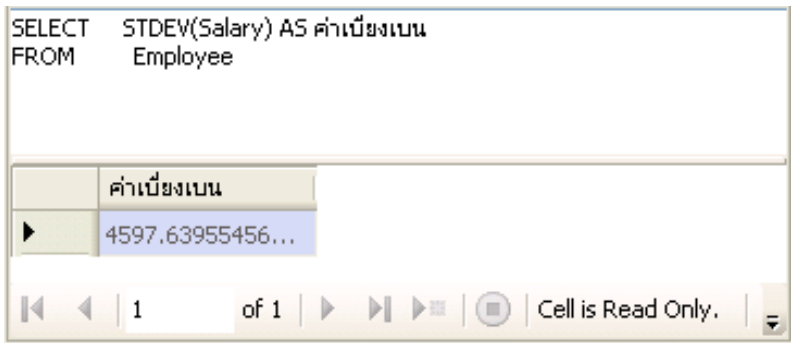
ฟังก์ชัน SUM	
ความหมาย	คำนวณหาผลรวมของคอลัมน์
รูปแบบ	SUM (column | expression)
ตัวอย่าง	SELECT SUM(Salary) FROM Emp

ตัวอย่าง



ฟังก์ชัน STDEV	
ความหมาย	คำนวณค่าเบี่ยงเบนมาตรฐาน
รูปแบบ	STDEV(column | expression)
ตัวอย่าง	SELECT STDEV(Salary) FROM EMP

ตัวอย่าง



คิวรี่ย่อย

Sub queries

วัตถุประสงค์

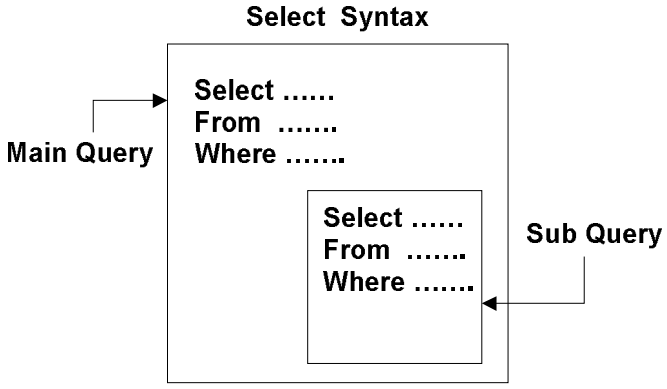
เข้าใจการเขียนคำสั่ง SQL แบบ Sub Queries

ประยุกต์ใช้คิวรี่ย่อยในคำสั่งที่ซับซ้อนได้

สามารถจัดเรียงข้อมูลด้วยคิวรี่ย่อยได้

คิวรี่ย่อย

คิวรี่ย่อย (Sub Query) คือ ประโยคคำสั่งสอบถามข้อมูล ซึ่งหมายถึงคำสั่ง Select และเป็นประโยคคำสั่งที่ซ้อนอยู่ภายใน หรือเป็นส่วนหนึ่งในประโยคสั่ง Select อีกชั้นหนึ่ง



การใช้คิวรี่ย่อยจะมีประโยชน์อย่างมากเมื่อต้องการสืบค้นข้อมูลจากตารางแบบ Self Join ที่จำเป็นต้องมีเงื่อนไขที่ซับซ้อน หรือการสืบค้นนั้นต้องการขั้นตอนการประมวลผลจากหลากหลายตาราง โดยสามารถเขียนคิวรี่ย่อยตามท้ายส่วนของคำสั่งต่างๆ ดังต่อไปนี้

- Where
- Having
- From (ที่เป็นส่วนของประโยคคำสั่ง Select และ Delete)

รูปแบบคิวรี่ย่อย

การสืบค้นผลลัพธ์จากฐานข้อมูลโดยใช้ คิวรี่ย่อยมีรูปแบบการใช้งานพื้นฐาน มีดังนี้

```
SELECT      [Distinct]      {* , column [alias] , …}
FROM        table
WHERE       expr    Operator
            ( SELECT        [Distinct]      {* , column [alias] , …}
            From  table )
```

เมื่อ SELECT คำสั่งที่ใช้ในการสืบคนข้อมูล ต้องกำหนดคอลัมน์อย่างน้อย 1 คอลัมน์

Distinct ระงับการแสดงรายการที่มีข้อมูลซ้ำ

* แสดงทุกคอลัมน์

column แสดงข้อมูลตามชื่อของคอลัมน์

alias ให้ชื่อที่ปรากฏในส่วนหัวของคอลัมน์เป็นไปตามกำหนด

table ชื่อตารางที่ประกอบไปด้วยคอลัมน์ตามที่กำหนดให้แสดงผล

expr คือ นิพจน์ (Expression) อาจประกอบไปด้วยคอลัมน์ และค่าคงที่ต่าง ๆ (ตัวเลข, ข้อความหรือตัวอักษร, วันที่และเวลา), ค่าว่าง เพื่ออ้างอิงไปยังชื่อคอลัมน์ของตารางนั้น ๆ

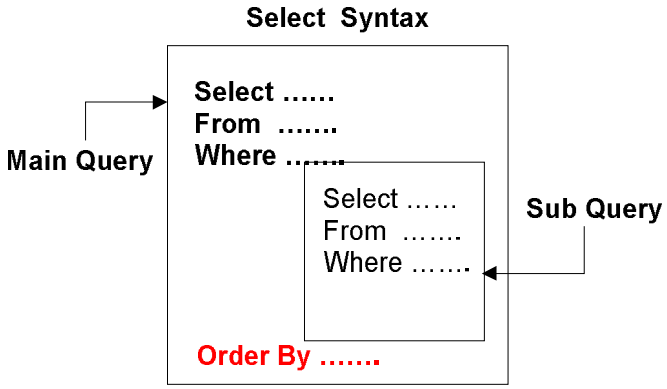
Operator คือตัวดำเนินการ เช่น > , = หรือ IN

ตัวดำเนินการที่ใช้ในการเขียนคำสั่ง SQL ที่มีคิวรี่ย่อยนี้ สามารถแบ่งตามลักษณะของผลลัพธ์ที่ได้จากคิวรี่ย่อย แยกเป็น 2 กลุ่มคือ

- ข้อมูลแบบแถวเดียว (Single Row) ได้แก่เครื่องหมาย > , = , >= , < , <= , <>
- ข้อมูลแบบหลายแถว (Multiple Row) ได้แก่เครื่องหมาย IN , NOT IN

ข้อพึงระวังในการใช้งานคิวรี่ย่อยที่นอกจากจะต้องเขียนคิวรี่ย่อยนี้ไหว้หลังตัวดำเนินการ คือ

1. ต้องเขียนคิวรี่ย่อยไว้ภายในเครื่องหมายวงเว็บ ( )
2. ส่วนคำสั่ง ORDER BY จะไม่สามารถวางไว้ในคิวรี่ย่อยได้ กล่าวคือทั้งประโยคคำสั่งจะสามารถมี ORDER BY ได้เพียงจุดเดียวคือในส่วนของ Select หลัก เท่านั้น



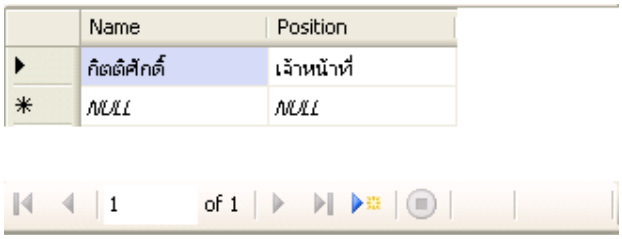
พื้นฐานการใช้งานคิวรี่ย่อย

จากโครงสร้างการใช้งานคิวรี่ย่อย จะประกอบด้วยชุดประโยคคำสั่ง Select จำนวน 2 ส่วน เพื่อให้ง่ายต่อความเข้าใจ จะขอยกตัวอย่างการสืบค้นแบบคิวรี่ย่อยในรูปแบบง่ายๆ ดังนี้

ตัวอย่าง

```
SELECT   Name ,  Position
FROM     Employee
WHERE   ID  =
         (SELECT  ID
         FROM    Employee
         WHERE   Surname = ‘สายแวว’)
```

ผลลัพธ์ที่ได้



	Name	Position
▶	กิตติศักดิ์	เจ้าหน้าที่
*	NULL	NULL

1 of 1

ขอแยกพิจารณาการทำงานของประโยคคำสั่งนี้เป็น 2 ส่วนคือ

1. ส่วนแรก คือส่วนสืบค้นย่อยหรือคิวรี่ที่ซ้อนอยู่ด้านใน ส่วนนี้จะถูกประมวลผลก่อน โดยจะพบว่าผลจากการทำงานของประโยคนี้คือการหารหัวของพนักงานที่มีนามกุล “สายแวว” ในตาราง Employee ซึ่งตรงกับพนักงานที่มีรหัส “1700”

```
(SELECT  ID
FROM    Employee
WHERE   Surname = ‘สายแวว’)
```

ผลลัพธ์ที่ได้

	ID
▶	1700
*	NULL

1 of 1

2. ส่วนที่ 2 คือส่วนที่ทำหน้าที่สืบค้นหลัก (Main Select) ผลจากการทำงานของคิวรี่ในส่วนนี้จะเป็นการแสดงชื่อ และตำแหน่งงาน โดยเลือกมาจากผู้ที่มีรหัสพนักงานตรงกับผลที่ได้จากคิวรี่ย่อย นั่นคือ พนักงานที่มีรหัสพนักงานเป็น “1700” ซึ่งก็คือพนักงานที่ชื่อ “กิตติศักดิ์” ตามผลลัพธ์ที่ได้จากการประมวลผลในข้างต้นนั่นเอง

ประวัติผู้เขียน

ชื่อ-สกุล อ.เอกรินทร์ วทัญญูเลิศสกุล (แซ่เฮ้ง)

การศึกษา - พ.ศ. 2546 วิทยาศาสตรมหาบัณฑิต

สาขาวิชาวิยาการสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

- พ.ศ. 2542 วิทยาศาสตรบัณฑิต เกียรตินิยมอันดับ 2

สาขาวิชาเทคโนโลยีอุตสาหกรรมอิเล็กทรอนิกส์

สถาบันราชภัฏอุบลราชธานี

- พ.ศ. 2539 ประกาศนียบัตรวิชาชีพชั้น

สาขาวิชาอิเล็กทรอนิกส์-คอมพิวเตอร์

วิทยาลัยเทคนิคอุบลราชธานี

พ.ศ. 2537 ประกาศนียบัตรวิชาชีพ

สาขาวิชาอิเล็กทรอนิกส์

วิทยาลัยเทคนิคอุบลราชธานี

การทำงาน พ.ศ. 2549-ปัจจุบัน

อาจารย์ประจำสาขาวิชาคอมพิวเตอร์ธุรกิจ

คณะบริหารธุรกิจ

มหาวิทยาลัยการจัดการและเทคโนโลยีอีสเทิร์น

พ.ศ. 2555-ปัจจุบัน

ผอ.สำนักเทคโนโลยีสารสนเทศและนวัตกรรม

พ.ศ. 2552-2554

หัวหน้าศูนย์ความเป็นเลิศด้านนวัตกรรม

วิชาที่สอน :

- Business Programming (JAVA)
- Business Programming (Visual Basic)
- Data Communication Network
- Data structure
- Decision Support System
- Visual Programming

พ.ศ. 2547-2549

หัวหน้าศูนย์สารสนเทศ

อาจารย์ประจำสาขาวิชาเทคโนโลยีสารสนเทศ

วิทยาลัยนอร์ทกรุงเทพ

วิชาที่สอน :

- Computer Programming
- Database Management System (Oracle)
- Special Topics in Business Computer I (Wireless Technology)

- Special Topics in Business Computer II (SQL Server)

พ.ศ. 2543-2549 System Analysis and Developer

บริษัท โกลเด้นไทย อินดรัสทรีย์ จำกัด (จ. สมุทรปราการ)