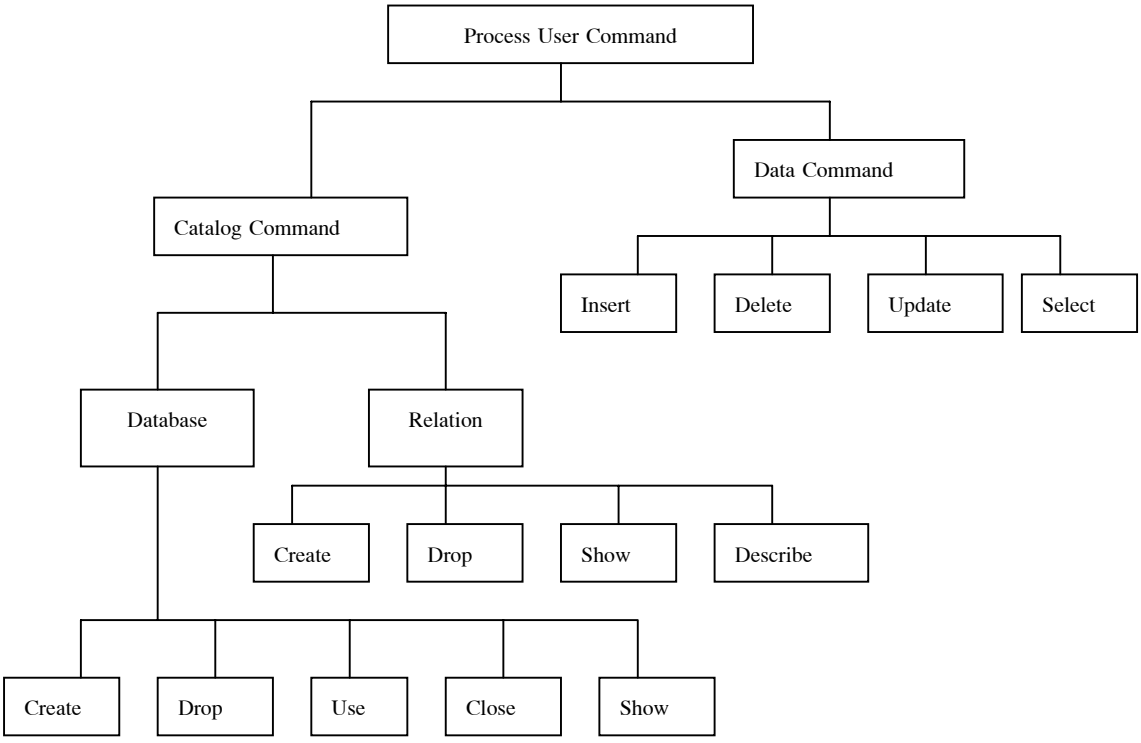
บทที่ 4

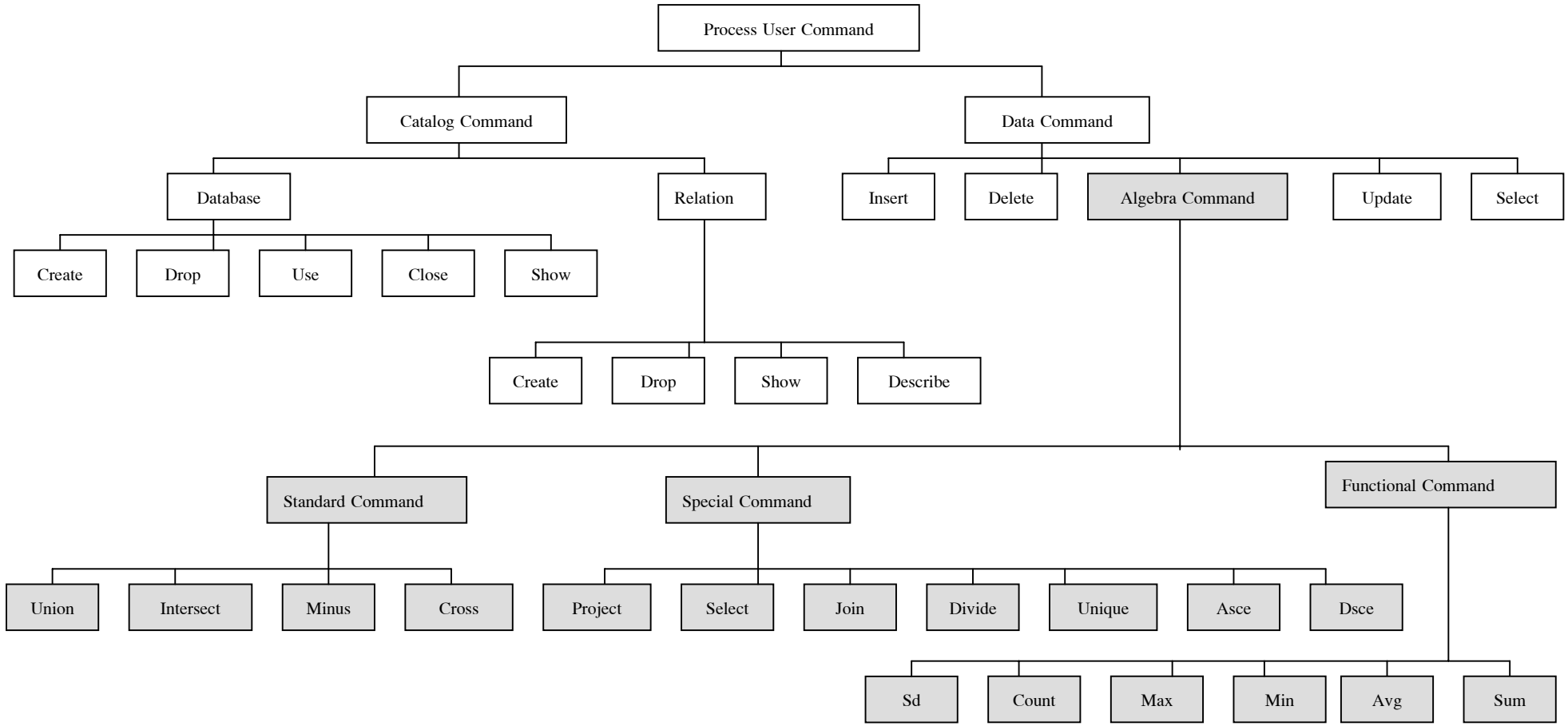
การออกแบบและพัฒนาระบบ

4.1 โครงสร้างระบบการดำเนินงานใหม่

การดำเนินงานในวิทยานิพนธ์นี้เป็นการเพิ่มการดำเนินงานในส่วนการประมวลผลคำสั่งในโครงสร้างระบบ MiniRDBMS เพื่อรองรับการประมวลผลคำสั่งปฏิบัติการพีชคณิตสัมพันธ์ 8 คำสั่งที่กล่าวถึงในบทที่ 2 และเพิ่มคำสั่งปฏิบัติการใหม่ขึ้นมา เพื่อการสอบถามข้อมูลหรือสารสนเทศได้ครอบคลุมมากยิ่งขึ้น ภาพประกอบ 4-1 แสดงโครงสร้างระบบการดำเนินงานประมวลผลคำสั่งเดิมและภาพประกอบ 4-2 แสดงโครงสร้างระบบการดำเนินงานประมวลผลคำสั่งใหม่



ภาพประกอบ 4-1 โครงสร้างระบบการดำเนินงานประมวลผลคำสั่งเดิม



ภาพประกอบ 4-2 โครงสร้างระบบดำเนินงานประมวลผลคำสั่งใหม่

4.2 คำสั่งปฏิบัติการในภาษาสอบถามพีชคณิตสัมพันธ์สำหรับ MiniRDBMS

วัตถุประสงค์ของงานวิทยานิพนธ์นี้ เพื่อพัฒนาตัวแปลภาษาสอบถามฐานข้อมูลพีชคณิตสัมพันธ์ที่ประกอบด้วยคำสั่งปฏิบัติการพื้นฐานที่ในอนาคตข้อความสั่งในภาษาแบบแคลคูลัสสัมพันธ์ เช่น SQL สามารถเรียกใช้คำสั่งปฏิบัติการพื้นฐานเหล่านี้ที่สมนัยกันมาดำเนินงานแทนได้

ผู้ดำเนินงานวิทยานิพนธ์ได้ศึกษารูปแบบและความหมายของคำสั่งปฏิบัติการมาตรฐานที่ C.J. Date เสนอ [Date, 1989] พร้อมทั้งศึกษาคำสั่งปฏิบัติการที่จำเป็นเพิ่มเติมเพื่อให้สามารถเขียนชุดคำสั่งปฏิบัติการที่สมนัยกับแต่ละข้อความสั่งในภาษาแบบแคลคูลัสสัมพันธ์ เช่น SQL ได้ โดยสรุปคำสั่งปฏิบัติการที่ได้ออกแบบและพัฒนาแบ่งออกเป็น 3 กลุ่มดังนี้

1. กลุ่มคำสั่งปฏิบัติการปกติกับเซต ประกอบด้วย
   - Union
   - Intersect
   - Minus
   - Cross

   การดำเนินงานของทุกคำสั่งในกลุ่มนี้เป็น Binary Operation
2. กลุ่มคำสั่งปฏิบัติการพิเศษกับเซต ประกอบด้วย
   - Project
   - Select
   - Join
   - Divide
   - Unique
   - Asce
   - Dsce

   การดำเนินงานของคำสั่งในกลุ่มนี้ส่วนใหญ่เป็น Unary Operation ยกเว้นคำสั่ง Join และ Divide เป็น Binary Operation
3. กลุ่มคำสั่งปฏิบัติการแบบฟังก์ชัน ประกอบด้วย
   - Count
   - Max
   - Min
   - Avg
   - Sum
   - Sd

   การดำเนินงานของทุกคำสั่งในกลุ่มนี้เป็น Unary Operation

4.3 รูปแบบและความหมายของคำสั่งปฏิบัติการ

รูปแบบแต่ละคำสั่งปฏิบัติการได้ถูกออกแบบให้เป็นแถวคำสั่งประกอบด้วย ชื่อตารางข้อมูลผลลัพธ์ สำหรับเก็บผลลัพธ์ที่ได้ ตามด้วยเครื่องหมาย “=” ชื่อคำสั่งปฏิบัติการและชื่อตารางข้อมูลสำหรับดำเนินงานตามคำสั่งปฏิบัติการซึ่งอาจเป็นตารางข้อมูลเดียว หรือสองตารางข้อมูล ดังรูปแบบต่อไปนี้

Unary operation :

<result> = <operator> <operand>

หรือ **Binary operation :**

<result> = <operator> <operand1> <operand2>

โดย <result> คือชื่อตารางข้อมูลสำหรับเก็บผลลัพธ์จากการดำเนินงานของคำสั่งปฏิบัติการกับตารางข้อมูล ซึ่งอยู่ทางขวามือของเครื่องหมาย “=”

<operator> คือคำสั่งปฏิบัติการใน 3 กลุ่มข้างต้น

<operand> , <operand1> และ <operand2> คือชื่อตารางข้อมูลสำหรับใช้เป็นข้อมูลในการดำเนินงาน

ความหมายของสัญลักษณ์ซึ่งใช้ในการเขียนรูปแบบแต่ละคำสั่งปฏิบัติการมีดังนี้

[ ] หมายถึง ข้อความภายในวงเล็บนี้จะเลือกใช้หรือไม่ใช้ก็ได้ เวลาใช้งานไม่ต้องพิมพ์เครื่องหมายนี้

< > หมายถึง ข้อความภายในสัญลักษณ์นี้บ่งบอกสิ่งที่ต้องการ เช่น ตารางข้อมูล ตัวแปร ค่าคงที่ หรือ เครื่องหมายที่ผู้ใช้เป็นผู้กำหนดสำหรับใช้แทนค่าข้อมูลจริง

... หมายถึง ยังมีต่อโดยละไว้ในฐานที่เข้าใจ

‘ ’ หมายถึง ต้องพิมพ์ข้อความหรือเครื่องหมายภายในสัญลักษณ์นี้ คำที่เป็นอักษรภาษาอังกฤษตัวพิมพ์ใหญ่ทั้งคำจะหมายถึงคำสงวนภายในคำสั่งปฏิบัติการต้องปรากฏคำสงวนนี้

4.3.1 กลุ่มคำสั่งปฏิบัติการปกติกับเซต

Union, Intersect, Minus และ Cross

รูปแบบ <result> ‘=’ UNION <table1> <table2> ‘;’

ความหมาย Union เป็นคำสั่งปฏิบัติการในการรวมแถวข้อมูลของตารางข้อมูล <table1> และ <table2> มาไว้ในตารางข้อมูล <result> แถวข้อมูลใดที่ซ้ำกันจากตารางข้อมูลทั้งสองจะถูกนำมาเพียงแถวเดียว

รูปแบบ <result> ‘=’ INTERSECT <table1> <table2> ‘;’
ความหมาย Intersect เป็นคำสั่งปฏิบัติการในการเลือกแถวข้อมูลที่เหมือนกันจากตารางข้อมูล <table1> และ <table2> มาไว้ในตารางข้อมูล <result>

รูปแบบ <result> ‘=’ MINUS <table1> <table2> ‘;’
ความหมาย Minus เป็นคำสั่งปฏิบัติการในการเลือกแถวข้อมูลที่ปรากฏในตารางข้อมูล <table1> แต่ไม่ปรากฏในตารางข้อมูล <table2> มาไว้ในตารางข้อมูล <result>

รูปแบบ <result> ‘=’ CROSS <table1> <table2> ‘;’
ความหมาย Cross เป็นคำสั่งปฏิบัติการในการจับคู่แถวข้อมูลทุกคู่ที่เป็นไปได้ของตารางข้อมูล <table1> และ <table2> มาไว้ในตารางข้อมูล <result> โดยแต่ละแถวข้อมูลในตารางข้อมูล <table1> จะถูกต่อท้ายด้วยแถวข้อมูลในตารางข้อมูล <table2>

ตารางข้อมูลสองตารางซึ่งนำมาดำเนินงานกับคำสั่ง Union, Intersect และ Minus ต้องมีคุณสมบัติ Union-Compatible และตารางข้อมูลผลลัพธ์ที่ได้จากคำสั่งปฏิบัติการเหล่านี้จะมีคุณสมบัติ Union-Compatible กับตารางข้อมูลทั้งสองด้วย

4.3.2 กลุ่มคำสั่งปฏิบัติการพิเศษกับเซต

Select

รูปแบบ <result> ‘=’ SELECT <table> ‘[’ <attribute> <operator> <constant> ‘]’ ‘;’
ความหมาย Select เป็นคำสั่งปฏิบัติการในการเลือกแถวข้อมูลจากตารางข้อมูล <table> โดยใช้ค่าของแอตทริบิว <attribute> เป็นเกณฑ์ในการเลือกแถวข้อมูล
โดย <operator> คือเครื่องหมายแทนการเปรียบเทียบค่า
<constant> คือค่าคงที่ซึ่งอาจเป็นชนิดเลขจำนวน หรือค่าคงที่ชนิดอักขระ

Project

รูปแบบ <result> ‘=’ PROJECT <table> ‘[’ <attribute1> [<attribute2>...] ‘]’ ‘;’
ความหมาย Project เป็นคำสั่งปฏิบัติการในการสกัดข้อมูลของแอตทริบิว <attributeN> โดยที่ N=1,2,3... จากตารางข้อมูล <table> อย่างน้อยต้องมีหนึ่งแอตทริบิว กรณีที่มีมากกว่าหนึ่งแอตทริบิว ให้แยกแอตทริบิวเหล่านั้นด้วยช่องว่างอย่างน้อยหนึ่งช่องว่าง

Join

รูปแบบ <result> ‘=’ JOIN <table1> <table2> ‘[’ <attribute1> ‘=’ <attribute2> ‘]’ ‘;’

ความหมาย Join เป็นคำสั่งปฏิบัติการในการจับคู่แถวข้อมูลของสองตารางข้อมูลทำนองเดียวกับคำสั่ง Cross ต่างกันที่แถวข้อมูลที่จะนำมาจับคู่กันเป็นแถวข้อมูลใหม่จะต้องมีค่าของแอตทริบิว <attribute1> ของ <table1> และ <attribute2> ของ <table2> ที่เท่ากัน และเรียกว่า Natural Join

Divide

รูปแบบ <result> ‘=’ DIVIDE <table1> <table2> ‘;’

ความหมาย Divide เป็นคำสั่งปฏิบัติการสำหรับดำเนินงานกับสองตารางข้อมูล โดยตารางข้อมูล <table1> ต้องเป็น Binary Relation ทำหน้าที่เป็นตัวตั้ง และตารางข้อมูล <table2> ต้องเป็น Unary Relation ทำหน้าที่เป็นตัวหาร แอตทริบิวเดียวของตารางข้อมูล <table2> ต้องใช้โดเมนเดียวกันกับหนึ่งแอตทริบิวในตารางข้อมูล <table1> ผลลัพธ์ที่ได้จะเป็น Unary Relation

Unique

รูปแบบ <result> ‘=’ UNIQUE <table> ‘;’

ความหมาย Unique เป็นคำสั่งปฏิบัติการสำหรับเลือกแถวข้อมูลแนวนอนที่ไม่ซ้ำกันจากตารางข้อมูล <table> และเก็บผลลัพธ์ใน <result>

Asce

รูปแบบ <result> ‘=’ ASCE <table> ‘[‘ <attribute> ‘]’ ‘;’

ความหมาย Asce เป็นคำสั่งปฏิบัติการสำหรับการดำเนินงานเรียงลำดับแถวข้อมูลแนวนอนจากค่าน้อยไปหาค่ามากตามค่าของแอตทริบิว <attribute> ที่ระบุในคำสั่งและเก็บผลลัพธ์ใน <result>

Dsce

รูปแบบ <result> ‘=’ DSEC <table> ‘[’ <attribute> ‘]’ ‘;’

ความหมาย Dsce เป็นคำสั่งปฏิบัติการสำหรับการดำเนินงานเรียงลำดับแถวข้อมูลแนวนอนจากค่ามากไปหาค่าน้อยตามค่าของแอตทริบิว <attribute> ที่ระบุในคำสั่งและเก็บผลลัพธ์ใน <result>

4.3.3 กลุ่มคำสั่งปฏิบัติการแบบฟังก์ชัน

Count

รูปแบบ <result> ‘=’ COUNT <table> ‘;’

ความหมาย Count เป็นคำสั่งปฏิบัติการสำหรับการดำเนินงานนับจำนวนแถวข้อมูลแนวนอนจากตารางข้อมูล <table> และเก็บผลลัพธ์ใน <result>

Max

รูปแบบ <result> ‘=’ MAX <table> ‘[’ <attribute> ‘]’ ‘;’

ความหมาย Max เป็นคำสั่งปฏิบัติการสำหรับการดำเนินงานหาค่าสูงสุดของค่าของแอตทริบิว <attribute> และเก็บผลลัพธ์ใน <result>

Min

รูปแบบ <result> ‘=’ MIN <table> ‘[’ <attribute> ‘]’ ‘;’

ความหมาย Min เป็นคำสั่งปฏิบัติการสำหรับการดำเนินงานการหาค่าต่ำสุดของค่าของแอตทริบิว <attribute> และเก็บผลลัพธ์ใน <result>

Avg

รูปแบบ <result> ‘=’ AVG <table> ‘[’ <attribute> ‘]’ ‘;’

ความหมาย Avg เป็นคำสั่งปฏิบัติการสำหรับการดำเนินงานคำนวณหาค่าเฉลี่ยของค่าของแอตทริบิว <attribute> และเก็บผลลัพธ์ใน <result>

Sum

รูปแบบ <result> ‘=’ SUM <table> ‘[’ <attribute> ‘]’ ‘;’

ความหมาย Sum เป็นคำสั่งปฏิบัติการสำหรับการดำเนินงานคำนวณผลรวมของค่าของแอตทริบิว <attribute> และเก็บผลลัพธ์ใน <result>

Sd

รูปแบบ <result> ‘=’ SD <table> ‘[’ <attribute> ‘]’ ‘;’

ความหมาย Sd เป็นคำสั่งปฏิบัติการสำหรับการดำเนินงานคำนวณส่วนเบี่ยงเบนมาตรฐานของค่าของแอตทริบิว <attribute> และเก็บผลลัพธ์ใน <result>

4.4 โครงสร้างของตัวแปลภาษาสอบถามฐานข้อมูลพีชคณิตสัมพันธ์

ตัวแปลภาษาสอบถามฐานข้อมูลพีชคณิตสัมพันธ์สำหรับ MiniRDBMS เป็นตัวแปลภาษาแบบ Interpreter มีองค์ประกอบหลักคือเซตของอักขระที่ใช้ คำสงวน ตัวแปร ค่าคงที่ และการดำเนินงานเชิงเปรียบเทียบ ส่วนขั้นตอนการดำเนินงานมี 3 ขั้นตอน ดังนี้

- ขั้นตอนการแยกและวิเคราะห์ศัพท์
- ขั้นตอนการวิเคราะห์วากยสัมพันธ์
- ขั้นตอนการวิเคราะห์ความหมายและตีความ

4.4.1 องค์ประกอบหลัก

เซตอักขระ

เซตของอักขระ (Character Sets) ประกอบด้วย

1) ตัวอักษร ประกอบด้วยตัวอักษร 52 ตัว คือ อักษรตัวใหญ่ A-Z และอักษรตัวเล็ก a-z
2) ตัวเลข ประกอบด้วยตัวเลข 10 จำนวน คือ 0-9
3) อักขระพิเศษ ประกอบด้วยอักขระทั้งหมด 12 ตัว คือ " > < = ! [ ] . _ * ; และช่องว่าง

คำสงวน

คำสงวน (Reserved Word) จะถือว่าอักษรตัวใหญ่หรืออักษรตัวเล็กไม่แตกต่างกัน คำสงวนทั้งหมดมีดังนี้

ASCE	AVG	COUNT	CROSS	DSCE
DIVIDE	HELP	INTERSECT	JOIN	MAX
MIN	MINUS	PROJECT	SD	SELECT
SUM	UNION	UNIQUE		

ตัวแปร

ชื่อตัวแปรประกอบด้วยอักขระในเซตอักขระตั้งแต่ 1 - 16 ตัว และต้องขึ้นต้นด้วยอักษร และอาจตามด้วยตัวอักษรหรือตัวเลขหรืออักษรพิเศษ "_" (Underscore) ก็ได้ โดยชื่อตัวแปรจะต้องไม่ตรงกับคำสงวน

ค่าคงที่

ค่าคงที่ (Constant) มี 2 ประเภท คือ

1) ค่าคงที่ชนิดเลขจำนวนเต็ม (Integer Constant) เป็นค่าคงที่ที่เป็นเลขจำนวนประกอบด้วยตัวเลขเท่านั้น ซึ่งขึ้นต้นด้วยตัวเลข 1-9 และตามด้วยตัวเลข 0-9
2) ค่าคงที่ชนิดอักขระ (Character Constant) จะได้แก่สายอักขระ (String) ใด ๆ ตั้งแต่หนึ่งอักขระขึ้นไป ซึ่งทุกอักขระในสายอักขระจะต้องเป็นอักขระที่อยู่ในเซตอักขระเท่านั้น

การดำเนินงานเชิงเปรียบเทียบ

การดำเนินงานเชิงเปรียบเทียบ (Comparison Operation) ซึ่งใช้ในบางคำสั่งปฏิบัติการพีชคณิตสัมพันธ์ ประกอบด้วยเครื่องหมายการเปรียบเทียบค่าสองค่า (Comparison Operator) คั่นระหว่างค่า สองค่า เครื่องหมายเปรียบเทียบค่าที่ใช้มีดังนี้

“>”	หมายถึง	มากกว่า
“<”	หมายถึง	น้อยกว่า
“>=”	หมายถึง	มากกว่าหรือเท่ากับ
“<=”	หมายถึง	น้อยกว่าหรือเท่ากับ
“=”	หมายถึง	เท่ากับ
“!=”	หมายถึง	ไม่เท่ากับ
“<>”	หมายถึง	ไม่เท่ากับ

4.4.2 ขั้นตอนการดำเนินงาน

ตัวแปลภาษาสอบถามแบบพีชคณิตสัมพันธ์ที่พัฒนาเป็นตัวแปลภาษาแบบ Interpreter มี 3 ขั้นตอนการดำเนินงานทั่วไป ดังนี้

ขั้นตอนการแยกและวิเคราะห์ศัพท์

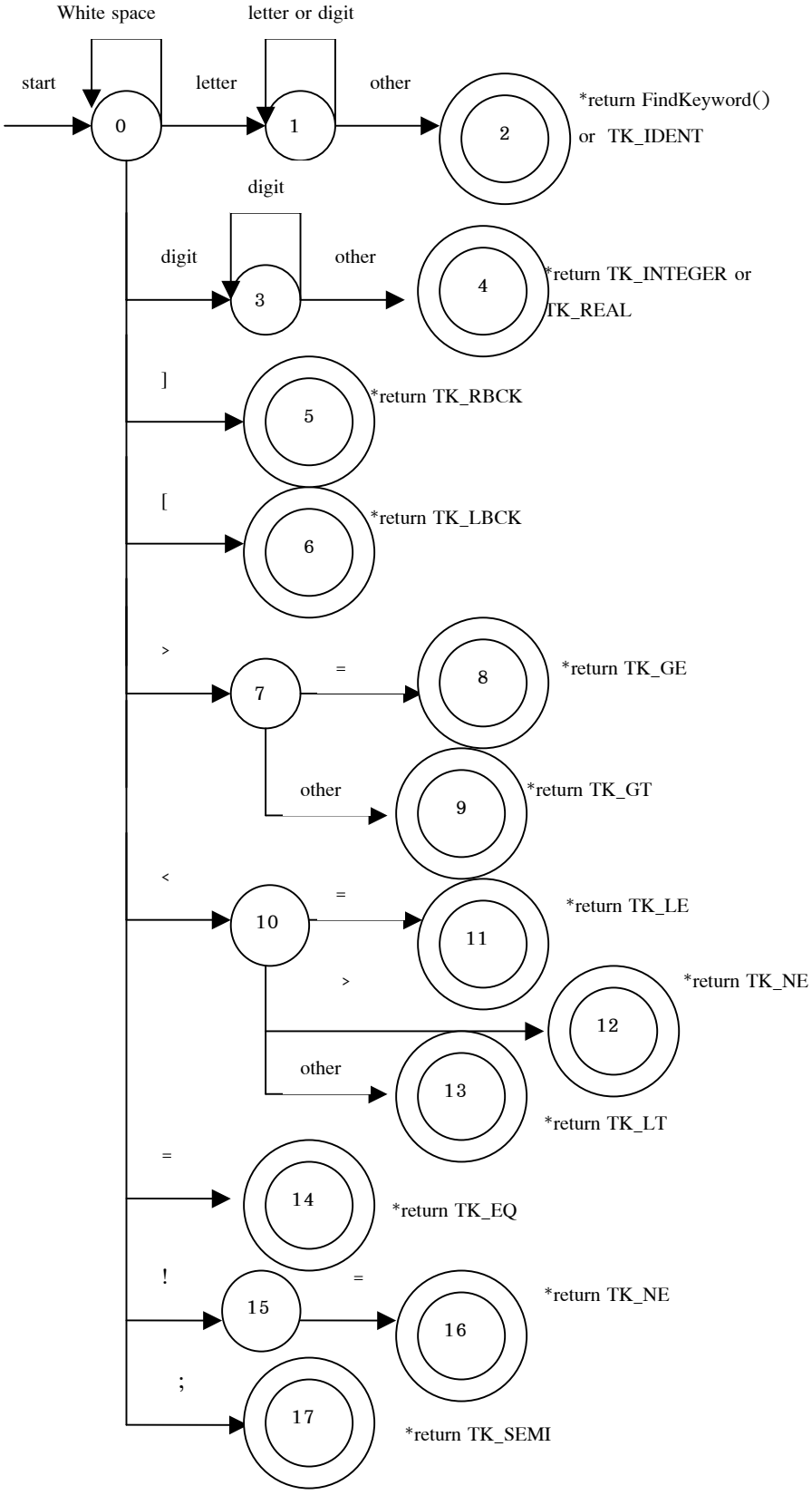
โดยทั่วไปขั้นตอนการวิเคราะห์ศัพท์จะเริ่มจากการทำงานของโปรแกรมที่เรียกว่าสแกนเนอร์ (Scanner) ทำหน้าที่อ่านแถวคำสั่งทีละอักขระ รวมกลุ่มอักขระเป็นคำและวิเคราะห์คำตามรูปแบบที่กำหนดในไวยากรณ์ของภาษา แต่ละกลุ่มของอักขระที่เป็นคำนี้เรียกว่า โทเคน (Token) และถ้ามีข้อผิดพลาดเกิดขึ้นจะให้แสดงข้อความบอกข้อผิดพลาดด้วย

การกำหนดโทเคน

โทเคนเป็นกลุ่มคำต่าง ๆ ในแถวคำสั่ง โดยแบ่งออกเป็นหลายประเภท เช่น Variable, Constant หรือ Reserved Word ต่างๆ ซึ่งแต่ละโทเคนจะมีการกำหนดรหัส ดังแสดงในตาราง 4-1

ตาราง 4-1 รหัสของโทเคน

โทเคน	กำหนดรหัสเป็น	โทเคน	กำหนดรหัสเป็น
>	TK_GT	cross	TK_CROSS
<	TK_LT	divide	TK_DIVIDE
>=	TK_GE	dsce	TK_DSCE
<=	TK_LE	help	TK_HELP
=	TK_EQ	intersect	TK_INTERSECT
<>	TK_NE	join	TK_JOIN
!=	TK_NE	max	TK_MAX
ตัวเลขจำนวนเต็ม	TK_INTEGER	min	TK_MIN
ตัวเลขจำนวนจริง	TK_REAL	minus	TK_MINUS
สายอักขระ	TK_IDENT	project	TK_PROJECT
]	TK_RBCK	quit	TK_QUIT
[	TK_LBCK	sd	TK_SD
;	TK_SEMICOLON	select	TK_SELECT
Asce	TK_ASCE	sum	TK_SUM
Avg	TK_AVG	union	TK_UNION
Count	TK_COUNT	unique	TK_UNIQUE



ภาพประกอบ 4-3 แผนภาพการแยกโทเคนที่ใช้ในระบบ

การแยกโทเคน

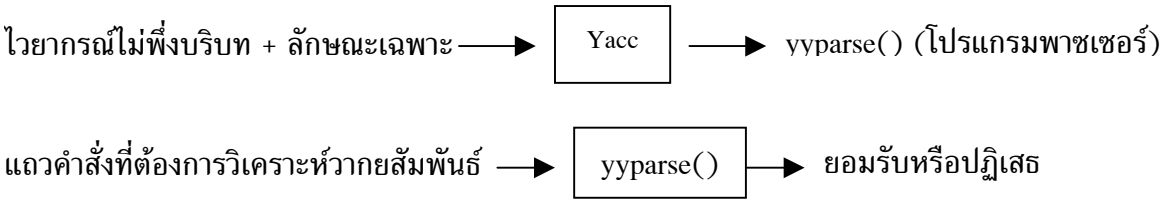
เนื่องจากในงาน MiniRDBMS เดิมได้มีการพัฒนาโปรแกรมแยกและวิเคราะห์คำศัพท์ชื่อ yylex() โดยใช้ภาษา C แล้ว ดังนั้นจึงเพิ่มโทเคนใหม่ในตาราง 4-1 ข้างต้น ทำให้โปรแกรมสามารถแยกทั้งโทเคนเดิมที่มีอยู่ และแยกโทเคนใหม่ที่เพิ่มเติมขึ้นได้อย่างมีประสิทธิภาพ ภาพประกอบ 4-3 แสดงวิธีการแยกโทเคนของโปรแกรม yylex() ที่ทำงานกับโทเคนใหม่สำหรับงานวิจัยนี้ โดยเครื่องหมายวงกลมและหมายเลขภายในจะแสดงถึงสถานะการทำงาน ซึ่งเริ่มจากสถานะการทำงานเริ่มต้นคือ start ถ้าเป็นเครื่องหมายวงกลมซ้อนกันจะหมายถึงสถานะสิ้นสุดการทำงานแยกโทเคนและสามารถทำการแยกโทเคนได้หนึ่งโทเคน ส่วนเครื่องหมาย * หมายถึง การคืน (Unget) อักขระที่รับเข้ามาหนึ่งตัว นั่นคือเมื่อรับอักขระเข้ามาแล้วพบสถานะที่มีเครื่องหมาย * จะต้อง Unget อักขระนั้นแล้วจึงทำการรับอักขระใหม่เพื่อเข้าสู่สถานะเริ่มต้นของการแยกโทเคนใหม่อีกครั้ง

ขั้นตอนการวิเคราะห์วากยสัมพันธ์

ขั้นตอนการวิเคราะห์วากยสัมพันธ์ (Syntax Analysis) เป็นการทำงานของโปรแกรมที่เรียกว่า พาซเซอร์ (Parser) เพื่อใช้วิเคราะห์ความถูกต้องของแถวคำสั่งที่ใช้โดยนำโทเคนที่ได้จากสแกนเนอร์มาตรวจสอบว่ามีการเรียงถูกต้องตามรูปแบบที่กำหนดไว้ในไวยากรณ์ของภาษาหรือไม่ ขั้นตอนการวิเคราะห์ศัพท์จะทำงานร่วมกับขั้นตอนการวิเคราะห์วากยสัมพันธ์ กล่าวคือสแกนเนอร์จะถูกเรียกใช้จากพาซเซอร์ที่ต้องการใช้โทเคนตัวถัดไป

ในการสร้างตัวแปลภาษานี้ได้ใช้โปรแกรมอรรถประโยชน์ชื่อ Yacc ของระบบปฏิบัติการ Linux เป็นโปรแกรมผลิตชุดคำสั่งย่อยภาษาซี เพื่อใช้วิเคราะห์วากยสัมพันธ์ (Syntax Analysis) ข้อมูลเข้าของ Yacc เป็นไวยากรณ์ไม่พึ่งบริบท (Context-Free Grammar) และลักษณะเฉพาะ (Attribute) รวมกันเรียกว่า ไวยากรณ์ลักษณะเฉพาะ (Attributed Grammar) ผลลัพธ์ที่ได้จาก Yacc เป็นชุดคำสั่งย่อยชื่อ yyparse() ซึ่งจะให้ค่าเป็น 0 เมื่อยอมรับว่าข้อมูลที่นำมาวิเคราะห์เขียนถูกต้องตามวากยสัมพันธ์ หรือให้ค่าเป็น 1 เมื่อปฏิเสธ

ดังนั้นจึงได้นำ Yacc มาช่วยในการสร้างพาซเซอร์ การเรียกใช้พาซเซอร์จะเรียกผ่านโปรแกรมย่อยชื่อ yyparse() โดยโปรแกรมแยกและวิเคราะห์คำศัพท์ yylex() ทำหน้าที่แยกโทเคนจากแถวคำสั่งที่ผู้ใช้พิมพ์เข้ามา และส่งให้โปรแกรม yyparse()



ภาพประกอบ 4-4 ข้อมูลเข้า/ออกและผลลัพธ์ที่ได้จากการใช้โปรแกรม Yacc และ yyparse ( )

ข้อมูลเข้าของ Yacc เป็นไวยากรณ์ไม่พึ่งบริบท (Context Free Grammar) และลักษณะเฉพาะ (Attribute) รวมกันเรียกว่า ไวยากรณ์ลักษณะเฉพาะ (Attribute Grammar) ไวยากรณ์ไม่พึ่งบริบทของแต่ละแถวคำสั่ง ซึ่งใช้ในโปรแกรม Yacc นี้จะเขียนอยู่ในรูปกฎ (Rule) แต่ละกฎประกอบด้วยสองส่วน แยกทั้งสองส่วนด้วยเครื่องหมาย ":" ส่วนแรกหน้าเครื่องหมาย ":" คือสัญลักษณ์ไม่สิ้นสุด (Nonterminal Symbol) ส่วนหลังเครื่องหมาย ":" คือสัญลักษณ์ไม่สิ้นสุดหรือสัญลักษณ์สิ้นสุด (Terminal Symbol) หรือสายอักขระว่าง (Empty String)

ตัวอย่างต่อไปนี้แสดงรูปแบบของข้อมูลเข้าของ Yacc เปรียบเทียบกับแผนภาพวากยสัมพันธ์ (Syntax Diagram) ที่สมนัยกัน

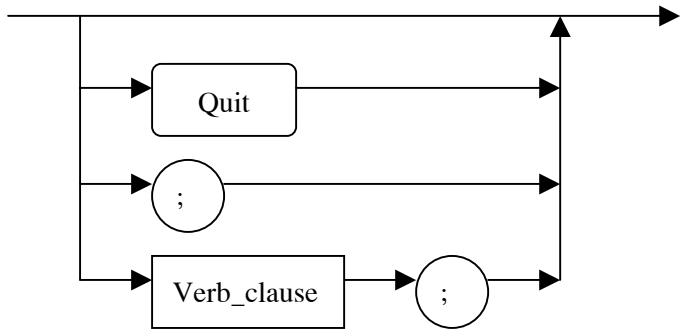
ตัวอย่างรูปแบบข้อมูลเข้าของ Yacc

```
query:  /* empty */ { YYABORT; }
        | TK_QUIT   { Lex->SqlCommand = SQLCOM_QUIT;}
        | TK_SEMICOLON  { YYABORT; }
        | verb_clause end_semi {}

end_semi:/* empty */
        {
           PrintErrPosition(COL);
          ShowErrMsg(ERR_SEMICOLON);
          YYABORT;
       }
        | TK_SEMICOLON {}

verb_clause : querycommand
            | helpcommand;

querycommand : relname
               {
               if (!(SetStrNameNew($1)))
                 YYABORT;
               }
                 TK_EQ  querystmt;
querystmt : s_operation
```

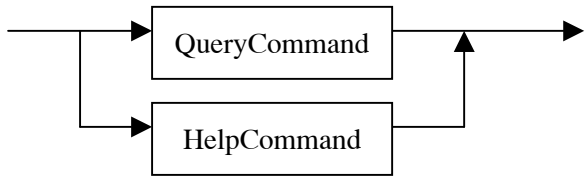
```
        | r_operation
        | p_operation;
s_operation : TK_INTERSECT
              {
                Lex->SqlCommand     = SQLCOM_INTERSECT;
              }
                r_get_rname
            | TK_UNION
              {
                Lex->SqlCommand     = SQLCOM_UNION;
              }
                r_get_rname
            | TK_MINUS
                {
                 Lex->SqlCommand     = SQLCOM_MINUS;
              }
                r_get_rname
            | TK_CROSS
              {
                Lex->SqlCommand     = SQLCOM_CROSS;
              }
                r_get_rname ;

r_get_rname : relname
                {
                  if (!(SetStrName($1)))
                YYABORT;
                 }
              relname
               {
                   if (!(SetStrNameLast($3)))
                YYABORT;
               }
```

แผนภาพวากยสัมพันธ์ที่สมนัยกัน

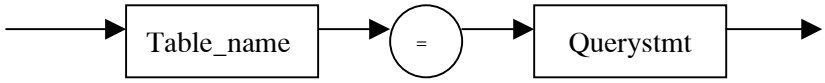
Query



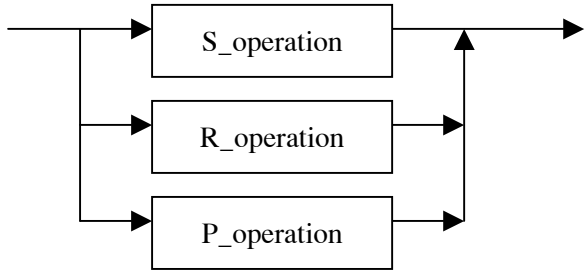
Verb_clause



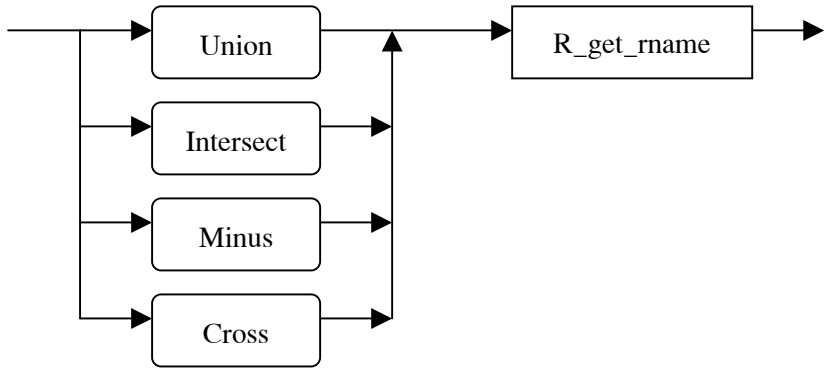
QueryCommand



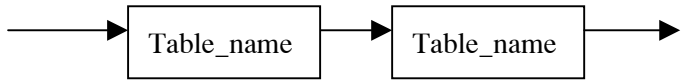
Querystmt



S_operation



R_get_rname



ขั้นตอนการวิเคราะห์ความหมายและตีความ

ขั้นตอนการวิเคราะห์ความหมายและตีความเป็นขั้นตอนการทำงานหลังจากที่มีขั้นตอนการแยกโทเคนและการตรวจสอบวากยสัมพันธ์จนถูกต้องแล้ว โดยเขียนโปรแกรมด้วยภาษาซี เพื่อตรวจสอบว่าข้อความสั่งนั้น ๆ สามารถใช้งานได้จริงหรือไม่ โดยขั้นตอนในส่วนนี้จะแยกไปดำเนินงานในแต่ละโมดูลของคำสั่งปฏิบัติการ

ขั้นตอนนี้จะรับโทเคนของแถวคำสั่งปฏิบัติการและตรวจสอบว่าคำสั่งปฏิบัติการนั้นคือคำสั่งใด จากนั้นจึงเข้าสู่โมดูลการทำงานของคำสั่งปฏิบัติการนั้น โดยทำการวิเคราะห์และตีความของแต่ละโทเคนที่รับมา เช่น ชื่อตารางข้อมูล ชื่อแอตทริบิว และคุณสมบัติต่าง ๆ ว่าถูกต้องหรือไม่และสามารถดำเนินงานได้จริงหรือไม่

4.5 ตัวอย่างข้อความผิดพลาดของระบบ

4.5.1 ขั้นตอนการวิเคราะห์วากยสัมพันธ์

กรณีที่ผู้ใช้พิมพ์แถวคำสั่งปฏิบัติการ โดยไม่ได้ใส่เครื่องหมาย “;” ท้ายแถวคำสั่ง เช่น

```
result = union dept1 dept2
```

ระบบจะแสดงข้อความผิดพลาดดังนี้

```
MYRDBMS>>result = union deptl dept2

result = union deptl dept2
                           ^
ERROR : ';' expected

MYRDBMS>>
```

กรณีผู้ใช้มีการพิมพ์แถวคำสั่งปฏิบัติการผิดไวยากรณ์ที่กำหนดไว้ เช่น ผู้ใช้พิมพ์คำสั่งปฏิบัติการจาก Asce เป็น Acse

```
result = acse student [std_id] ;
```

ระบบจะแสดงข้อความผิดพลาดดังนี้

```
MYRDBMS>>result = acse student [std_id] ;

result = acse student [std_id] ;
         ^
ERROR : Syntax error near 'acse stude...'

MYRDBMS>>
```

กรณีผู้ใช้พิมพ์ชื่อตารางข้อมูลที่ต้องการใช้ในแถวคำสั่งไม่ครบตามหลักไวยากรณ์ เช่น คำสั่งปฏิบัติการ Cross ต้องการตารางข้อมูลสองตาราง แต่ผู้ใช้พิมพ์แค่หนึ่งตารางเท่านั้น

```
result = cross student ;
```

ระบบจะแสดงข้อความผิดพลาดดังนี้

```
MYRDBMS>>result = cross student ;

result = cross student ;
                       ^
ERROR : Syntax error near ';...'

MYRDBMS>>
```

กรณีผู้ใช้พิมพ์เครื่องหมาย “[ ]” เป็น “( )”

```
result = select person (dept_id = 1) ;
```

ระบบจะแสดงข้อความผิดพลาดดังนี้

```
MYRDBMS>>result = select person (dept_id = 1) ;

result = select person (dept_id = 1) ;
                       ^
ERROR : Syntax error near '(dept_id =...'

MYRDBMS>>
```

แม้แต่กรณีที่ผู้ใช้ต้องการใช้คำสั่งปฏิบัติการ Project โดยภายในเครื่องหมาย “[]” จะต้องมีรายชื่อแอตทริบิวที่ต้องการคำตอบ ที่ถูกต้องก็คือแต่ละแอตทริบิวจะต้องคั่นด้วยช่องว่าง แต่ถ้าผู้ใช้คั่นด้วยเครื่องหมาย ‘,’ จะทำให้เกิดความผิดพลาดขึ้น

```
result = project  student [std_id , std_name] ;
```

ระบบจะแสดงข้อความผิดพลาดดังนี้

```
MYRDBMS>>result = project  student [std_id , std_name] ;

result = project  student [std_id , std_name] ;
                                  ^
ERROR : Syntax error near ', std_name...'

MYRDBMS>>
```

4.5.2 ขั้นตอนการวิเคราะห์ความหมายและตีความ

กรณีที่ผู้ใช้ต้องการดูข้อมูลทั้งหมดที่อยู่ในตารางข้อมูล student แต่พิมพ์ผิดเป็น student1 เมื่อผ่านขั้นตอนการวิเคราะห์วากยะสัมพันธ์จะทำงานถูกต้อง แต่เมื่อผ่านขั้นตอนการวิเคราะห์ความหมายและตีความแถวคำสั่ง

```
result = select student1 [*];
```

ระบบจะเข้าสู่โมดูลการทำงานของคำสั่งปฏิบัติการ Select จากนั้นจะตรวจสอบว่าตารางข้อมูล student1 มีอยู่ในฐานข้อมูลหรือไม่ ในกรณีนี้ student1 ไม่มีอยู่ในฐานข้อมูล ดังนั้นระบบจะแสดงข้อความผิดพลาดดังนี้

```
result = select studentl [*];
                 ^
ERROR : Relation name 'STUDENTl' does not exist in database 'PERSONAL'

MYRDBMS>>
```

นอกจากตรวจสอบว่าตารางข้อมูลที่ระบุนั้นอยู่ในฐานข้อมูลหรือไม่แล้ว ระบบยังต้องมีการตรวจสอบว่าแอตทริบิวที่ระบุนั้นอยู่ในฐานข้อมูลหรือไม่ เช่น

```
result = max student [grade];
```

ระบบจะทำการตรวจสอบว่าในตารางข้อมูล student นั้นมีแอตทริบิว grade หรือไม่ ในที่นี้ตารางข้อมูล student ไม่มีแอตทริบิว grade ระบบจะแสดงข้อผิดพลาดดังนี้

```
MYRDBMS>>result = max student [grade];

result = max student [grade];
                     ^
ERROR : Invalid attribute name in 'attribute list'

MYRDBMS>>
```

กรณีคำสั่งปฏิบัติการ Union, Intersect และ Minus ตารางข้อมูลสองตารางที่ใช้ดำเนินงานได้นั้นจะต้องมีคุณสมบัติ Union-Compatitble ระบบจะต้องทำการตรวจสอบคุณสมบัติก่อนที่จะดำเนินงาน

```
result = minus student dept1 ;
```

ระบบจะตรวจสอบได้ว่าตารางข้อมูล student และ dept1 ไม่มีคุณสมบัติ Union-Compatitble ระบบจะแสดงข้อความผิดพลาดดังนี้

```
MYRDBMS>>result = minus student deptl ;
ERROR:Two Relation are not union compatible
MYRDBMS>>
```

กรณีคำสั่งปฏิบัติการ Divide ตารางข้อมูลที่หนึ่งจะต้องมีคุณสมบัติ Binary Relation นั่นคือประกอบด้วยสองแอตทริบิว และตารางข้อมูลที่สองจะต้องมีคุณสมบัติ Unary Relation นั่นคือประกอบด้วยหนึ่งแอตทริบิว

```
result = divide student dept1;
```

จากตัวอย่างตารางข้อมูล student ไม่มีคุณสมบัติ Binary Relation และตารางข้อมูล dept1 ไม่มีคุณสมบัติ Unary Relation เช่นกัน ระบบจะแสดงข้อความผิดพลาดดังนี้

```
MYRDBMS>>result = divide student deptl;

ERROR:Two relation must be Binary Relation and Unary Relation

MYRDBMS>>
```

4.6 ขั้นตอนวิธี

ขั้นตอนวิธีของแต่ละคำสั่งปฏิบัติการเป็นขั้นตอนวิธีที่ใช้ภาษาธรรมชาติผสมกับภาษาที่มีโครงสร้างข้อความควบคุมคล้ายภาษาซี เช่น

ถ้า *นิพจน์ตรรกะเป็นจริง ดำเนินงานต่อไปนี้*	หมายถึง	if statement
ตราบเท่าที่ *นิพจน์ตรรกะเป็นจริง ดำเนินงานต่อไปนี้*	หมายถึง	while statement

4.6.1 กลุ่มคำสั่งปฏิบัติการปกติกับเซต

คำสั่งปฏิบัติการ Union

ตัวแปรในการเรียกใช้

กำหนดให้	R1	หมายถึง	ตารางข้อมูลที่ 1
	R2	หมายถึง	ตารางข้อมูลที่ 2
	Result	หมายถึง	ตารางข้อมูลผลลัพธ์ที่ได้จากการดำเนินงาน
	Tuple1	หมายถึง	ทุเปิลของตารางข้อมูลที่ 1
	Tuple2	หมายถึง	ทุเปิลของตารางข้อมูลที่ 2
	Flag	หมายถึง	ตัวแปรตรวจสอบ

ขั้นตอนวิธี

Union(Result, R1, R2)

1. รับชื่อ Result, R1 และ R2
2. ถ้า R1 ไม่อยู่ในฐานข้อมูล
   2.1 แสดงข้อความผิดพลาด
   2.2 ไปที่ข้อ 11
3. นำข้อมูล R1 เข้าสู่หน่วยความจำ
4. ถ้า R2 ไม่อยู่ในฐานข้อมูล
   4.1 แสดงข้อความผิดพลาด
   4.2 ไปที่ข้อ 11
5. นำข้อมูล R2 เข้าสู่หน่วยความจำ
6. เข้าสู่โมดูล CheckUnionCompatible(R1, R2)
   R1 และ R2 ไม่มีคุณสมบัติ Union-Compatible
   - แสดงข้อความผิดพลาด
   - ไปที่ข้อ 11
7. สำเนา Tuple1 ทั้งหมดลงใน Result
8. กำหนด Flag = 0
9. ตราบเท่าที่มีข้อมูลใน R2
   9.1 ตราบเท่าที่มีข้อมูลใน Result
   ถ้า Tuple1 เท่ากับ Result กำหนด Flag = 1
   9.2 ถ้า Flag เท่ากับ 0 สำเนา Tuple2 ลงใน Result
   9.3 กำหนด Flag = 0
10. พิมพ์ Result ออกทางจอภาพ
11. จบการทำงาน

<u>คำสั่งปฏิบัติการ Intersect</u>

ตัวแปรในการเรียกใช้

กำหนดให้	R1	หมายถึง	ตารางข้อมูลที่ 1
	R2	หมายถึง	ตารางข้อมูลที่ 2
	Result	หมายถึง	ตารางข้อมูลผลลัพธ์ที่ได้จากการดำเนินงาน
	Tuple1	หมายถึง	ทุเปิลของตารางข้อมูลที่ 1
	Tuple2	หมายถึง	ทุเปิลของตารางข้อมูลที่ 2
	Flag	หมายถึง	ตัวแปรตรวจสอบ

ขั้นตอนวิธี

Intersect(Result, R1, R2)

1. รับชื่อ Result, R1 และ R2
2. ถ้า R1 ไม่อยู่ในฐานข้อมูล
   2.1 แสดงข้อความผิดพลาด
   2.2 ไปที่ข้อ 10
3. นำข้อมูล R1 เข้าสู่หน่วยความจำ
4. ถ้า R2 ไม่อยู่ในฐานข้อมูล
   4.1 แสดงข้อความผิดพลาด
   4.2 ไปที่ข้อ 10
5. นำข้อมูล R2 เข้าสู่หน่วยความจำ
6. เข้าสู่โมดูล CheckUnionCompatible(R1, R2)
   R1 และ R2 ไม่มีคุณสมบัติ Union-Compatible
   - แสดงข้อความผิดพลาด
   - ไปที่ข้อ 10
7. กำหนด Flag = 0
8. ตราบเท่าที่มีข้อมูลใน R1
   8.1 ตราบเท่าที่มีข้อมูลใน R2
   ถ้า Tuple1 เท่ากับ Tuple2 กำหนด Flag = 1
   8.2 ถ้า Flag เท่ากับ 1 สำเนา Tuple1 ลงใน Result
   8.3 กำหนด Flag = 0
9. พิมพ์ Result ออกทางจอภาพ
10. ออกจากการทำงาน

<u>คำสั่งปฏิบัติการ Minus</u>

ตัวแปรในการเรียกใช้

กำหนดให้	R1	หมายถึง	ตารางข้อมูลที่ 1
	R2	หมายถึง	ตารางข้อมูลที่ 2
	Result	หมายถึง	ตารางข้อมูลผลลัพธ์ที่ได้จากการดำเนินงาน
	Tuple1	หมายถึง	ทุเปิลของตารางข้อมูลที่ 1
	Tuple2	หมายถึง	ทุเปิลของตารางข้อมูลที่ 2
	Flag	หมายถึง	ตัวแปรตรวจสอบ

ขั้นตอนวิธี

Minus(Result, R1, R2)

1. รับชื่อ Result, R1 และ R2
2. ถ้า R1 ไม่อยู่ในฐานข้อมูล
   2.1 แสดงข้อความผิดพลาด
   2.2 ไปที่ข้อ 10
3. นำข้อมูล R1 เข้าสู่หน่วยความจำ
4. ถ้า R2 ไม่อยู่ในฐานข้อมูล
   4.1 แสดงข้อความผิดพลาด
   4.2 ไปที่ข้อ 10
5. นำข้อมูลของ R2 เข้าสู่หน่วยความจำ
6. เข้าสู่โมดูล CheckUnionCompatible(R1, R2)
   R1 และ R2 ไม่มีคุณสมบัติ Union-Compatible
   - แสดงข้อความผิดพลาด
   - ไปที่ข้อ 10
7. กำหนด Flag = 0
8. ตราบเท่าที่มีข้อมูลใน R1
   8.1 ตราบเท่าที่มีข้อมูลใน R2
   ถ้า Tuple1 เท่ากับ Tuple2 กำหนด Flag = 1
   8.2 ถ้า Flag เท่ากับ 0 สำเนา Tuple1 ลงใน Result
   8.3 กำหนด Flag = 0
9. พิมพ์ Result ออกทางจอภาพ
10. ออกจากการทำงาน

โมดูล CheckUnionCompatible

โมดูลที่ดำเนินงานตรวจสอบคุณสมบัติ Union-Compatible ของคำสั่งปฏิบัติการ Union, Intersect และ Minus

ตัวแปรในการเรียกใช้

กำหนดให้	R1	หมายถึง	ตารางข้อมูลที่ 1
	R2	หมายถึง	ตารางข้อมูลที่ 2
	#Att1	หมายถึง	จำนวนแอตทริบิวของตารางข้อมูลที่ 1
	#Att2	หมายถึง	จำนวนแอคทริบิวของตารางข้อมูลที่ 2
	i	หมายถึง	ดัชนีควบคุมการทำงาน เริ่มต้นที่ 1
	Domain1[i]	หมายถึง	โดเมนของแอตทริบิว i ในตารางข้อมูลที่ 1
	Domain2[i]	หมายถึง	โดเมนของแอตทริบิว i ในตารางข้อมูลที่ 2
	Flag	หมายถึง	ตัวแปรตรวจสอบ

ขั้นตอนวิธี

CheckUnionCompatible(R1, R2)

1. ถ้า #Att1 ไม่เท่ากับ #Att2 ไปที่ข้อ 4
2. กำหนด i = 1 และ Flag = 1
3. ตราบเท่าที่ i < #Att1
   3.1 ถ้า Domain1[i] ไม่เท่ากับ Domain2[i] กำหนดให้ Flag = 0
   3.2 i = i + 1
4. ออกจากการทำงาน

หมายเหตุ ถ้า Flag = 1 หมายถึง R1 และ R2 มีคุณสมบัติ Union-Compatible
ถ้า Flag = 0 หมายถึง R1 และ R2 ไม่มีคุณสมบัติ Union-Compatible

<u>คำสั่งปฏิบัติการ Cross</u>

ตัวแปรในการเรียกใช้

กำหนดให้	R1	หมายถึง	ตารางข้อมูลที่ 1
	R2	หมายถึง	ตารางข้อมูลที่ 2
	Result	หมายถึง	ตารางข้อมูลผลลัพธ์ที่ได้จากการดำเนินงาน
	Tuple1	หมายถึง	ทุเปิลของตารางข้อมูลที่ 1
	Tuple2	หมายถึง	ทุเปิลของตารางข้อมูลที่ 2
	ConcatBuff	หมายถึง	ตัวแปรชนิด String

ขั้นตอนวิธี

Cross(Result, R1, R2)

1. รับชื่อ Result, R1 และ R2
2. ถ้า R1 ไม่อยู่ในฐานข้อมูล
   2.1 แสดงข้อความผิดพลาด
   2.2 ไปที่ข้อ 8
3. นำข้อมูล R1 เข้าสู่หน่วยความจำ
4. ถ้า R2 ไม่อยู่ในฐานข้อมูล
   4.1 แสดงข้อความผิดพลาด
   4.2 ไปที่ข้อ 8
5. นำข้อมูล R2 เข้าสู่หน่วยความจำ
6. ตราบเท่าที่มีข้อมูลใน R1
   ตราบเท่าที่มีข้อมูลใน R2
   - นำ Tuple1 เชื่อมต่อกับ Tuple2 สำเนาลงใน ConcatBuff
   - สำเนา ConcatBuff ลงใน Result
7. พิมพ์ Result ออกทางจอภาพ
8. ออกจากการทำงาน

4.6.2 กลุ่มคำสั่งปฏิบัติการพิเศษกับเซต

<u>คำสั่งปฏิบัติการ Project</u>

ตัวแปรในการเรียกใช้

กำหนดให้	R	หมายถึง	ตารางข้อมูล
	Result	หมายถึง	ตารางข้อมูลผลลัพธ์ที่ได้จากการดำเนินงาน
	Tuple	หมายถึง	ทุเปิลของตารางข้อมูล R
	AttList	หมายถึง	เซตของชื่อของแอตทริบิวที่ต้องการ
	Flag	หมายถึง	ตัวแปรตรวจสอบ
	Att	หมายถึง	แอตทริบิวใน AttList

ขั้นตอนวิธี

Project(Result, R, AttList)

1. รับชื่อ Result, R และ AttList
2. ถ้า R ไม่อยู่ในฐานข้อมูล
   2.1 แสดงข้อความผิดพลาด
   2.2 ไปที่ข้อ 9
3. นำข้อมูลของ R เข้าสู่หน่วยความจำ
4. กำหนดค่า Flag = 0
5. ตราบเท่าที่มีข้อมูลใน AttList
   ถ้า AttList ไม่อยู่ใน R กำหนดให้ Flag = 1
6. ถ้า Flag เท่ากับ 1
   6.1 แสดงข้อความผิดพลาด
   6.2 ไปที่ข้อ 9
7. ตราบเท่าที่มีข้อมูลใน R
   สำเนา Tuple เฉพาะข้อมูลที่อยู่ใน Att ลงใน Result
8. พิมพ์ Result ออกทางจอภาพ
9. ออกจากการทำงาน

<u>คำสั่งปฏิบัติการ Select</u>

ตัวแปรในการเรียกใช้

กำหนดให้	R	หมายถึง	ตารางข้อมูล
	Result	หมายถึง	ตารางข้อมูลผลลัพธ์ที่ได้จากการดำเนินงาน
	Tuple	หมายถึง	ทุเปิลของตารางข้อมูล R
	AttCond	หมายถึง	แอตทริบิวที่เป็นเงื่อนไข
	Condition	หมายถึง	เงื่อนไขในการสอบถามฐานข้อมูล
	Tuple.AttCond	หมายถึง	ข้อมูลที่อยู่ในแอตทริบิว AttCond ของ Tuple

ขั้นตอนวิธี

Select(Result, R, AttCond, Condition)

1. รับชื่อ Result, R, AttCond และ Condition
2. ถ้า R ไม่อยู่ในฐานข้อมูล
   2.1 แสดงข้อความผิดพลาด
   2.2 ไปที่ข้อ 8
3. นำข้อมูล R เข้าสู่หน่วยความจำ
4. ถ้า AttCond เท่ากับ '*'
   4.1 ตราบเท่าที่มีข้อมูลใน R
      สำเนา Tuple ลงใน Result
   4.2 พิมพ์ Result ออกทางจอภาพ
   4.3 ไปที่ข้อ 8
5. ถ้า AttCond ไม่อยู่ใน R
   5.1 แสดงข้อความผิดพลาด
   5.2 ไปที่ข้อ 8
6. ตราบเท่าที่มีข้อมูลใน R
   ถ้า Tuple.AttCond เป็นไปตามตาม Condition สำเนา Tuple ลงใน Result
7. พิมพ์ Result ออกทางจอภาพ
8. ออกจากการทำงาน

<u>คำสั่งปฏิบัติการ Join</u>

ตัวแปรในการเรียกใช้

กำหนดให้	R1	หมายถึง	ตารางข้อมูลที่ 1
	R2	หมายถึง	ตารางข้อมูลที่ 2
	Result	หมายถึง	ตารางข้อมูลผลลัพธ์ที่ได้จากการดำเนินงาน
	Tuple1	หมายถึง	ทุเปิลของตารางข้อมูลที่ 1
	Tuple2	หมายถึง	ทุเปิลของตารางข้อมูลที่ 2
	AttJoin1	หมายถึง	แอตทริบิวที่ต้องการนำมา Join ในตารางข้อมูลที่ 1
	AttJoin2	หมายถึง	แอตทริบิวที่ต้องการนำมา Join ในตารางข้อมูลที่ 2
	Tuple1.AttJoin1	หมายถึง	ข้อมูลที่อยู่ในแอตทริบิว AttJoin1 ของ Tuple1
	Tuple2.AttJoin2	หมายถึง	ข้อมูลที่อยู่ในแอตทริบิว AttJoin2 ของ Tuple2
	ConcatBuff	หมายถึง	ตัวแปรชนิด String

ขั้นตอนวิธี

Join(Result, R1, R2, AttJoin1, AttJoin2)

1. รับชื่อ Result, R1, R2, AttJoin1 และ AttJoin2
2. ถ้า R1 ไม่อยู่ในฐานข้อมูล
   2.1 แสดงข้อความผิดพลาด
   2.2 ไปที่ข้อ 10
3. นำข้อมูล R1 เข้าสู่หน่วยความจำ
4. ถ้า R2 ไม่อยู่ในฐานข้อมูล
   4.1 แสดงข้อความผิดพลาด
   4.2 ไปที่ข้อ 10
5. นำข้อมูล R2 เข้าสู่หน่วยความจำ
6. ถ้า AttJoin1 ไม่อยู่ใน R1
   6.1 แสดงข้อความผิดพลาด
   6.2 ไปที่ข้อ 10
7. ถ้า AttJoin2 ไม่อยู่ใน R2
   7.1 แสดงข้อความผิดพลาด
   7.2 ไปที่ข้อ 10
8. ตราบเท่าที่มีข้อมูลใน R1
   ตราบเท่าที่มีข้อมูลใน R2
   ถ้า Tuple1.AttJoin1 เท่ากับ Tuple2.AttJoin2

- สำเนา Tuple1 ลงใน ConcatBuff
- สำเนา Tuple2 เชื่อมต่อกับ Tuple1 ใน ConcatBuff
- สำเนา ConcatBuff ลงใน Result

9. พิมพ์ Result ออกทางจอภาพ
10. ออกจากการทำงาน

<u>คำสั่งปฏิบัติการ Divide</u>

ข้อกำหนดเบื้องต้น

1. ตารางข้อมูลตัวตั้งต้องเป็น Binary Relation และตารางข้อมูลตัวหารต้องเป็น Unary Relation
2. ข้อมูลในตารางตัวตั้งมีการเรียงลำดับทุเปิลตามค่าของแอตทริบิวตัวแรกและตัวที่สองตามลำดับ
3. ข้อมูลในตารางตัวหารมีการเรียงลำดับทุเปิลตามค่าของแอตทริบิวตัวแรกและตัวที่สองตามลำดับ

ตัวแปรในการเรียกใช้

กำหนดให้	R1	หมายถึง	ตารางข้อมูลที่ 1
	R2	หมายถึง	ตารางข้อมูลที่ 2
	Result	หมายถึง	ตารางผลลัพธ์ที่ได้จากการดำเนินงาน
	Att1	หมายถึง	แอตทริบิวของตารางข้อมูลที่ 1
	Att2	หมายถึง	แอตทริบิวของตารางข้อมูลที่ 1
	Att3	หมายถึง	แอตทริบิวของตารางข้อมูลที่ 2
	Tuple1	หมายถึง	ทุเปิลของตารางข้อมูลที่ 1
	Tuple2	หมายถึง	ทุเปิลของตารางข้อมูลที่ 2
	Tuple1.Att1	หมายถึง	ข้อมูลที่อยู่ในแอตทริบิว Att1 ของ Tuple1
	Tuple1.Att2	หมายถึง	ข้อมูลที่อยู่ในแอตทริบิว Att2 ของ Tuple1
	Tuple2.Att3	หมายถึง	ข้อมูลที่อยู่ในแอตทริบิว Att3 ของ Tuple2
	Temp	หมายถึง	ตัวแปรชนิด String
	First	หมายถึง	ตัวแปรตรวจสอบ
	Flag	หมายถึง	ตัวแปรตรวจสอบ
	Check	หมายถึง	ดัชนีควบคุมการทำงาน
	Done	หมายถึง	ดัชนีควบคุมการทำงาน

ขั้นตอนวิธี

Divide(Result, R1, R2)

1. รับชื่อ Result, R1 และ R2
2. ถ้า R1 ไม่อยู่ในฐานข้อมูล
   2.1 แสดงข้อความผิดพลาด
   2.2 ไปที่ข้อ 14
3. นำข้อมูล R1 เข้าสู่หน่วยความจำ
4. ถ้า R2 ไม่อยู่ในฐานข้อมูล
   4.1 แสดงข้อความผิดพลาด
   4.2 ไปที่ข้อ 14
5. นำข้อมูล R2 เข้าสู่หน่วยความจำ
6. ถ้าจำนวนแอตทริบิวของ R1 ไม่เท่ากับ 2
   6.1 แสดงข้อความผิดพลาด
   6.2 ไปที่ข้อ 14
7. ถ้าจำนวนแอตทริบิวของ R2 ไม่เท่ากับ 1
   7.1 แสดงข้อความผิดพลาด
   7.2 ไปที่ข้อ 14
8. กำหนดให้ Flag = 0
9. ถ้า Domain ของ Tuple1.Att1 หรือ Domain ของ Tuple1.Att2 เท่ากับ Domain ของ Tuple2.Att3 กำหนดให้ Flag = 1 (สมมติให้ Domain ของ Tuple1.Att1 เท่ากับ Domain ของ Tuple2.Att3)
10. ถ้า Flag ไม่เท่ากับ 1
    10.1 แสดงข้อความผิดพลาด
    10.2 ไปที่ข้อ 14
11. กำหนดให้ Check = 0 , First = 0 และ Done = 0
12. ตราบเท่าที่มีข้อมูลใน R2
    ตราบเท่าที่มีข้อมูลใน R1
    - Done = Done + 1
    - ถ้า Tuple1.Att1 เท่ากับ Tuple2.Att3 และ First เท่ากับ 0
      - Check = Check+1
      - ถ้า Check เท่ากับจำนวนทุเปิลทั้งหมดของ R2
        - สำเนา Tuple1.Att2 ใน Result
        - Check = 0

- ถ้า Check ไม่เท่ากับจำนวนทุเปิลทั้งหมดของ R2
  - Temp = Tuple1.Att2
  - First = 1
  - ไปที่ 12

- ถ้า Tuple1.Att1 เท่ากับ Tuple2.Att3 และ First ไม่เท่ากับ 0
  - ถ้า Temp เท่ากับ Tuple1.Att2 ให้ Check = Check + 1
  - ถ้า Check เท่ากับจำนวนทุเปิลทั้งหมดของ R2
    - สำเนา Temp ใน Result
    - Check = 0
    - ถ้า Done ไม่เท่ากับจำนวนทุเปิลทั้งหมดของ R1 ไปที่ข้อ 12
  - ถ้า Check ไม่เท่ากับจำนวนทุเปิลทั้งหมดของ R2 ไปที่ข้อ 12

13. พิมพ์ Result ออกทางจอภาพ

14. ออกจากการทำงาน

<u>คำสั่งปฏิบัติการ Unique</u>

ตัวแปรในการเรียกใช้

กำหนดให้	R	หมายถึง	ตารางข้อมูล
	Result	หมายถึง	ตารางข้อมูลผลลัพธ์ที่ได้จากการดำเนินงาน
	RD	หมายถึง	ตารางข้อมูล R ที่ลบข้อมูลใน Result ออก
	Tuple	หมายถึง	ทุเปิลของตารางข้อมูล R
	TupleD	หมายถึง	ทุเปิลของตารางข้อมูล RD
	P	หมายถึง	หมายเลขของ Tuple
	Q	หมายถึง	หมายเลขของ TupleD
	Flag	หมายถึง	ตัวแปรตรวจสอบ
	k	หมายถึง	ตัวแปร index ของ Array
	Dup[k]	หมายถึง	Array ที่เก็บหมายเลข tuple ที่ซ้ำกัน

ขั้นตอนวิธี

Unique(Result, R)

1. รับชื่อ Result และ R
2. ถ้า R ไม่อยู่ในฐานข้อมูล
   2.1 แสดงข้อความผิดพลาด
   2.2 ไปที่ข้อ 7

3. นำข้อมูล R เข้าสู่หน่วยความจำ
4. กำหนดให้ P = 0 และ Q = 1
5. ตราบเท่าที่มีข้อมูลใน R
   5.1 P = P + 1
   5.2 กำหนดให้ k = 0, Flag = 0
   5.3 ตราบเท่าที่ Dup[k] มีข้อมูล
      5.3.1 ถ้า P เท่ากับ Dup[k] กำหนดให้ Flag = 1
      5.3.2 k = k + 1
   5.4 ถ้า Flag เท่ากับ 1 ไปที่ข้อ 5
   5.5 สำเนา Tuple ลงใน Result
   5.6 ตราบเท่าที่มีข้อมูลใน RD
      5.6.1 Q = Q + 1
      5.6.2 ถ้า Tuple เท่ากับ TupleD
         5.6.2.1 กำหนด Dup[k] = Q
         5.6.2.2 k = k + 1
6. พิมพ์ Result ออกทางจอภาพ
7. ออกจากการทำงาน

<u>คำสั่งปฏิบัติการ Asce</u>

ตัวแปรในการเรียกใช้

กำหนดให้	R	หมายถึง ตารางข้อมูล
	Result	หมายถึง ตารางข้อมูลผลลัพธ์ที่ได้จากการดำเนินงาน
	AttCond	หมายถึง แอตทริบิวที่เป็นเงื่อนไข
	Tuple	หมายถึง ทุเปิลของตารางข้อมูล R
	Domain.AttCond	หมายถึง Domain ของแอตทริบิวของตารางข้อมูล R
	LinkList	หมายถึง โครงสร้างข้อมูลประเภท linklist
	Tuple.AttCond	หมายถึง ข้อมูลที่อยู่ในแอตทริบิวของ Tuple
	L	หมายถึง ข้อมูลที่อยู่ใน LinkList

ขั้นตอนวิธี

Asce(Result, R, AttCond)

1. รับชื่อ Result, R และ AttCond
2. ถ้า R ไม่อยู่ในฐานข้อมูล

2.1 แสดงข้อความผิดพลาด

2.2 ไปที่ข้อ 10

3. นำข้อมูล R เข้าสู่หน่วยความจำ
4. ถ้า AttCond ไม่อยู่ใน R

4.1 แสดงข้อความผิดพลาด

4.2 ไปที่ข้อ 10

5. ถ้า Domain.AttCond มีชนิดข้อมูลเป็น Date

5.1 แสดงข้อความผิดพลาด

5.2 ไปที่ข้อ 10

6. ตราบเท่าที่มีข้อมูลใน R

สำเนา Tuple.AttCond ลงใน LinkList

7. ทำการเรียงลำดับข้อมูลใน LinkList จากน้อยไปมาก โดยใช้ Bubble Sort Algorithm
8. ตราบเท่าที่ LinkList ไม่เท่ากับ NULL

8.1 ตราบเท่าที่มีข้อมูลใน R

ถ้า Tuple.AttCond เท่ากับ L ทำการสำเนา Tuple ลงใน Result

8.2 LinkList.Next = LinkList

9. พิมพ์ Result ออกทางจอภาพ
10. ออกจากการทำงาน

<u>คำสั่งปฏิบัติการ Dsce</u>

ตัวแปรในการเรียกใช้

กำหนดให้	R	หมายถึง	ตารางข้อมูล
	Result	หมายถึง	ตารางข้อมูลผลลัพธ์ที่ได้จากการดำเนินงาน
	AttCond	หมายถึง	แอตทริบิวที่เป็นเงื่อนไข
	Tuple	หมายถึง	ทุเปิลของตารางข้อมูล R
	Domain.AttCond	หมายถึง	Domain ของแอตทริบิวของตารางข้อมูล R
	LinkList	หมายถึง	โครงสร้างข้อมูลประเภท linklist
	Tuple.AttCond	หมายถึง	ข้อมูลที่อยู่ในแอตทริบิวของ Tuple
	L	หมายถึง	ข้อมูลที่อยู่ใน LinkList

ขั้นตอนวิธี

Dsce(Result, R, AttCond)

1. รับชื่อ Result, R และ AttCond
2. ถ้า R ไม่อยู่ในฐานข้อมูล
   2.1 แสดงข้อความผิดพลาด
   2.2 ไปที่ข้อ 10
3. นำข้อมูล R เข้าสู่หน่วยความจำ
4. ถ้า AttCond ไม่อยู่ใน R
   4.1 แสดงข้อความผิดพลาด
   4.2 ไปที่ข้อ 10
5. ถ้า Domain.AttCond มีชนิดข้อมูลเป็น Date
   5.1 แสดงข้อความผิดพลาด
   5.2 ไปที่ข้อ 10
6. ตราบเท่าที่มีข้อมูลใน R
   สำเนา Tuple.AttCond ลงใน LinkList
7. ทำการเรียงลำดับข้อมูลใน LinkList จากมากไปน้อย โดยใช้ Bubble Sort Algorithm
8. ตราบเท่าที่ LinkList ไม่เท่ากับ NULL
   8.1 ตราบเท่าที่มีข้อมูลใน R
      ถ้า Tuple.AttCond เท่ากับ L ทำการสำเนา Tuple ลงใน Result
   8.2 LinkList.Next = LinkList
9. พิมพ์ Result ออกทางจอภาพ
10. ออกจากการทำงาน

4.6.3 กลุ่มคำสั่งปฏิบัติการแบบฟังก์ชัน

<u>คำสั่งปฏิบัติการ Count</u>

ตัวแปรในการเรียกใช้

กำหนดให้	R	หมายถึง	ตารางข้อมูล
	Result	หมายถึง	ตารางข้อมูลผลลัพธ์ที่ได้จากการดำเนินงาน
	Count	หมายถึง	ดัชนีควบคุมการทำงาน

ขั้นตอนวิธี

Count(Result, R)

1. รับชื่อ Result และ R
2. ถ้า R ไม่อยู่ในฐานข้อมูล
   2.1 แสดงข้อความผิดพลาด
   2.2 ไปที่ข้อ 7
3. นำข้อมูลของ R เข้าสู่หน่วยความจำ
4. กำหนดให้ Count = 0
5. ตราบเท่าที่มีข้อมูลใน R กำหนดให้ Count = Count + 1
6. พิมพ์ค่า Count ออกทางจอภาพ
7. ออกจากการทำงาน

<u>คำสั่งปฏิบัติการ Max</u>

ตัวแปรในการเรียกใช้

กำหนดให้	R	หมายถึง	ตารางข้อมูล
	Result	หมายถึง	ตารางข้อมูลผลลัพธ์ที่ได้จากการดำเนินงาน
	AttCond	หมายถึง	แอตทริบิวที่เป็นเงื่อนไข
	Tuple.AttCond	หมายถึง	ข้อมูลที่อยู่ในแอตทริบิวของตารางข้อมูล R
	Domain.AttCond	หมายถึง	Domain ของแอตทริบิวของตารางข้อมูล R
	Max	หมายถึง	ตัวแปรชนิดตัวเลข หมายถึงค่าที่มากที่สุด

ขั้นตอนวิธี

Max(Result, R, AttCond)

1. รับชื่อ Result, R และ AttCond
2. ถ้า R ไม่อยู่ในฐานข้อมูล
   2.1 แสดงข้อความผิดพลาด
   2.2 ไปที่ข้อ 9

3. นำข้อมูลของ R เข้าสู่หน่วยความจำ
4. ถ้า AttCond ไม่อยู่ใน R
    4.1 แสดงข้อความผิดพลาด
    4.2 ไปที่ข้อ 9
5. ถ้า Domain.AttCond ไม่เป็นข้อมูลประเภท “Integer” หรือ “Real”
    5.1 แสดงข้อความผิดพลาด
    5.2 ไปที่ข้อ 9
6. กำหนดให้ Max = Tuple.AttCond
7. ตราบเท่าที่มีข้อมูลใน R
    ถ้า Max < Tuple.AttCond กำหนดให้ Max = Tuple.AttCond
8. พิมพ์ค่า Max ออกทางจอภาพ
9. ออกจากการทำงาน

<u>คำสั่งปฏิบัติการ Min</u>

ตัวแปรในการเรียกใช้

กำหนดให้	R	หมายถึง	ตารางข้อมูล
	Result	หมายถึง	ตารางข้อมูลผลลัพธ์ที่ได้จากการดำเนินงาน
	AttCond	หมายถึง	แอตทริบิวที่เป็นเงื่อนไข
	Tuple.AttCond	หมายถึง	ข้อมูลที่อยู่ในแอตทริบิวของตารางข้อมูล R
	Domain.AttCond	หมายถึง	Domain ของแอตทริบิวของตารางข้อมูล R
	Min	หมายถึง	ตัวแปรชนิดตัวเลข หมายถึงค่าที่น้อยที่สุด

ขั้นตอนวิธี

Min(Result, R, AttCond)

1. รับชื่อ Result, R และ AttCond
2. ถ้า R ไม่อยู่ในฐานข้อมูล
    2.1 แสดงข้อความผิดพลาด
    2.2 ไปที่ข้อ 9
3. นำข้อมูลของ R เข้าสู่หน่วยความจำ
4. ถ้า AttCond ไม่อยู่ใน R
    4.1 แสดงข้อความผิดพลาด
    4.2 ไปที่ข้อ 9
5. ถ้า Domain.AttCond ไม่เป็นข้อมูลประเภท “Integer” หรือ “Real”

5.1 แสดงข้อความผิดพลาด

5.2 ไปที่ข้อ 9

6. กำหนดให้ Min = Tuple.AttCond
7. ตราบเท่าที่มีข้อมูลใน R

   ถ้า Min > Tuple.AttCond กำหนดให้ Min = Tuple.AttCond
8. พิมพ์ค่า Min ออกทางจอภาพ
9. ออกจากการทำงาน

<u>คำสั่งปฏิบัติการ Sum</u>

ตัวแปรในการเรียกใช้

กำหนดให้	R	หมายถึง	ตารางข้อมูล
	Result	หมายถึง	ตารางข้อมูลผลลัพธ์ที่ได้จากการดำเนินงาน
	AttCond	หมายถึง	แอตทริบิวที่เป็นเงื่อนไข
	Tuple.AttCond	หมายถึง	ข้อมูลที่อยู่ในแอตทริบิวของตารางข้อมูล R
	Domain.AttCond	หมายถึง	Domain ของแอตทริบิวของตารางข้อมูล R
	Sum	หมายถึง	ตัวแปรชนิดตัวเลข หมายถึงผลรวมข้อมูล

ขั้นตอนวิธี

Sum(Result, R, AttCond)

1. รับชื่อ Result, R และ AttCond
2. ถ้า R ไม่อยู่ในฐานข้อมูล

   2.1 แสดงข้อความผิดพลาด

   2.2 ไปที่ข้อ 9
3. นำข้อมูล R เข้าสู่หน่วยความจำ
4. ถ้า AttCond ไม่อยู่ใน R

   4.1 แสดงข้อความผิดพลาด

   4.2 ไปที่ข้อ 9
5. ถ้า Domain.AttCond ไม่เป็นข้อมูลประเภท “Integer” หรือ “Real”

   5.1 แสดงข้อความผิดพลาด

   5.2 ไปที่ข้อ 9
6. กำหนดให้ Sum = 0
7. ตราบเท่าที่มีข้อมูลใน R

   Sum = Sum + Tuple.AttCond

8. พิมพ์ค่า Sum ออกทางจอภาพ
9. ออกจากการทำงาน

<u>คำสั่งปฏิบัติการ Avg</u>

ตัวแปรในการเรียกใช้

กำหนดให้	R	หมายถึง	ตารางข้อมูล
	Result	หมายถึง	ตารางข้อมูลผลลัพธ์ที่ได้จากการดำเนินงาน
	AttCond	หมายถึง	แอตทริบิวที่เป็นเงื่อนไข
	Tuple.AttCond	หมายถึง	ข้อมูลที่อยู่ในแอตทริบิวของตารางข้อมูล R
	Domain.AttCond	หมายถึง	Domain ของแอตทริบิวของตารางข้อมูล R
	Sum	หมายถึง	ตัวแปรชนิดตัวเลขเก็บค่าผลรวมข้อมูล
	i	หมายถึง	ดัชนีควบคุมการทำงาน
	Avg	หมายถึง	ตัวแปรชนิดตัวเลขเก็บค่าเฉลี่ยของข้อมูล

ขั้นตอนวิธี

Avg(Result, R, AttCond)

1. รับชื่อ Result, R และ AttCond
2. ถ้า R ไม่อยู่ในฐานข้อมูล
   2.1 แสดงข้อความผิดพลาด
   2.2 ไปที่ข้อ 10
3. นำข้อมูล R เข้าสู่หน่วยความจำ
4. ถ้า AttCond ไม่อยู่ใน R
   4.1 แสดงข้อความผิดพลาด
   4.2 ไปที่ข้อ 10
5. ถ้า Domain.AttCond ไม่เป็นข้อมูลประเภท “Integer” หรือ “Real”
   5.1 แสดงข้อความผิดพลาด
   5.2 ไปที่ข้อ 10
6. กำหนดให้ Sum = 0, i = 0 และ Avg = 0
7. ตราบเท่าที่มีข้อมูลใน R
   7.1 Sum = Sum + Tuple.AttCond
   7.2 i = i + 1
8. Avg = Sum / i

9. พิมพ์ค่า Avg ออกทางจอภาพ
10. ออกจากการทำงาน

<u>คำสั่งปฏิบัติการ Sd</u>

ตัวแปรในการเรียกใช้

กำหนดให้	R	หมายถึง	ตารางข้อมูล
	Result	หมายถึง	ตารางข้อมูลผลลัพธ์ที่ได้จากการดำเนินงาน
	AttCond	หมายถึง	แอตทริบิวที่เป็นเงื่อนไข
	Tuple.AttCond	หมายถึง	ข้อมูลที่อยู่ในแอตทริบิวของตารางข้อมูล R
	Domain.AttCond	หมายถึง	Domain ของแอตทริบิวของตารางข้อมูล R
	Sum	หมายถึง	ตัวแปรชนิดตัวเลข หมายถึงผลรวมข้อมูล
	i	หมายถึง	ดัชนีควบคุมการทำงาน
	Avg	หมายถึง	ตัวแปรชนิดตัวเลขเก็บค่าเฉลี่ยของข้อมูล
	Num1	หมายถึง	ตัวแปรชนิดตัวเลข
	Num2	หมายถึง	ตัวแปรชนิดตัวเลข
	Num3	หมายถึง	ตัวแปรชนิดตัวเลข
	Sd	หมายถึง	ตัวแปรชนิดตัวเลขเก็บค่าส่วนเบี่ยงเบนมาตรฐาน

ขั้นตอนวิธี

Sd(Result, R, AttCond)

1. รับชื่อ Result, R และ AttCond
2. ถ้า R ไม่อยู่ในฐานข้อมูล
   2.1 แสดงข้อความผิดพลาด
   2.2 ไปที่ข้อ 12
3. นำข้อมูลของ R เข้าสู่หน่วยความจำ
4. ถ้า AttCond ไม่อยู่ใน R
   4.1 แสดงข้อความผิดพลาด
   4.2 ไปที่ข้อ 12
5. ถ้า Domain.AttCond ไม่เป็นข้อมูลประเภท “Integer” หรือ “Real”
   5.1 แสดงข้อความผิดพลาด
   5.2 ไปที่ข้อ 12
6. กำหนดให้ Sum = 0, i = 0 และ Num3 = 0

7. ตราบเท่าที่มีข้อมูลใน R

   7.1 Sum = Sum + Tuple.AttCond

   7.2 i = i + 1

8. Avg = Sum / i

9. ตราบเท่าที่มีข้อมูลใน R

   9.1 Num1 = Tuple.AttCond − Avg

   9.2 Num2 = Num1*Num1

   9.3 Num3 = Num3 + Num2

10. $Sd = \sqrt{Num3 / (i - 1)}$

11. พิมพ์ค่า Sd ออกทางจอภาพ

12. ออกจากการทำงาน