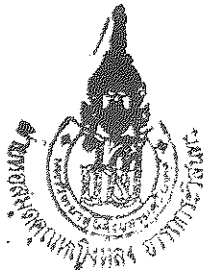


โปรแกรม คำสั่งปฏิบัติการฐานข้อมูลแบบพีชคณิตสัมพันธ์

The Relational Algebra Operation Package (RAO Package)

เกษมันทน์ แก้วอิ่ม

Gasamand Gaew-Im

วิทยานิพนธ์วิทยาศาสตรมหาบัณฑิต สาขาวิทยาการคอมพิวเตอร์

มหาวิทยาลัยสงขลานครินทร์

Master of Science Thesis in Computer Science

Prince of Songkla University

2531

ชื่อหัวข้อวิทยานิพนธ์	โปรแกรม คำสั่งปฏิบัติการฐานข้อมูลแบบพีชคณิตสัมพันธ์
ชื่อผู้เขียน	นาย เกษมัณฑน์ แก้วอิ่ม
สาขาวิชา	วิทยาการคอมพิวเตอร์

คณะกรรมการที่ปรึกษา

.....ประธานกรรมการ
(ผศ.ดร. อัชฌา ธีรเชษฐมงคล)

.....กรรมการ
(ผศ.ดร. พูลพงษ์ บุญพราหมณ์)

คณะกรรมการสอบ

.....ประธานกรรมการ
(ผศ.ดร. อัชฌา ธีรเชษฐมงคล)

.....กรรมการ
(ผศ.ดร. พูลพงษ์ บุญพราหมณ์)

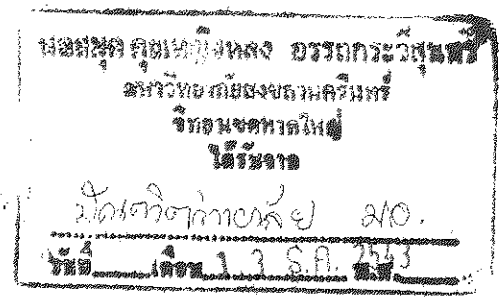
.....กรรมการ
(อาจารย์ วุฒิพงศ์ เตชะดำรงสิน)

.....กรรมการ
(รศ.ดร. ศิริพงษ์ ศรีนิพัฒน์)

บัณฑิตวิทยาลัย มหาวิทยาลัยสงขลานครินทร์ อนุมัติให้วิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตร์มหาบัณฑิต สาขาวิทยาการคอมพิวเตอร์

.....

(รศ.ดร. ก้าน จันทร์พรหมมา)
คณบดีบัณฑิตวิทยาลัย



หัวข้อวิทยานิพนธ์ โปรแกรม คำสั่งปฏิบัติการฐานข้อมูลแบบพีชคณิตสัมพันธ์

ผู้เขียน นาย เกษมัณฑน์ แก้วอิ่ม

สาขาวิชา วิทยาการคอมพิวเตอร์

ปีการศึกษา 2531

บทคัดย่อ

ระบบฐานข้อมูลนับว่าเป็นสิ่งที่มีความจำเป็นมาก สำหรับหน่วยงาน หรือองค์กรขนาดใหญ่ที่มีการใช้ข้อมูลเป็นจำนวนมาก ทั้งนี้เพราะระบบฐานข้อมูลทำให้การจัดเก็บข้อมูลและการใช้ข้อมูลเหล่านั้นเป็นไปอย่างมีระบบ และมีประสิทธิภาพ ส่วนที่มีความสำคัญและจำเป็นมากที่สุดในระบบฐานข้อมูล คือ ระบบจัดการฐานข้อมูลซึ่งเป็นโปรแกรมที่ใช้ในการจัดการ และอำนวยความสะดวกในการปฏิบัติการกับข้อมูลภายในฐานข้อมูล ไม่ว่าจะเป็นการเพิ่ม ลบ เปลี่ยนแปลงข้อมูล หรือการหยิบชิ้นข้อมูลที่ต้องการ

ในการจัดทำวิทยานิพนธ์ครั้งนี้ ได้ทำการพัฒนาโปรแกรมสำหรับคำสั่งปฏิบัติการฐานข้อมูลแบบพีชคณิตสัมพันธ์ในส่วนของ Data Manipulation Language โดยใช้กับฐานข้อมูลแบบรีเลชั่น คำสั่งปฏิบัติการต่าง ๆ แบ่งออกเป็น 2 กลุ่ม โดยในกลุ่มแรกจะเป็นคำสั่งปฏิบัติการพื้นฐานกับเซท คำสั่งในกลุ่มนี้ประกอบด้วย Intersection Union Difference และ Cross product ส่วนในกลุ่มหลังจะเป็นคำสั่งพิเศษซึ่งได้แก่ Select Project Join และ Divide

โปรแกรมทั้งหมดทำการพัฒนาบนเครื่องคอมพิวเตอร์ VAX-11/785 ซึ่งใช้ ULTRIX-32 เป็นระบบดำเนินงาน โดยใช้ภาษา C ในการพัฒนาโปรแกรมทั้งหมด

Thesis title The Relational Algebra Operation Package (RAO Package)

Author Mr. Gasamand Gaew-Im

Major program Computer Sciences

Academic year 1988

Abstract

Database system is highly required for the management of the large organization dealing with a great number of data. Database system enables the organization to store and utilize data systematically and efficiently. The most important part of a database system is the database management system (DBMS). It is the computer software that renders the users all facilities concerning the data operations such as the updating, the insertion, the deletion and the retrieval of data.

This thesis aims at developing an instructional relational algebra operation which are data manipulation language of a relational database management system. All operations are divided into two groups. The first group is the conventional set operations which consist of the intersection, the union, the difference and the cross product. The second group is the special operations operating on attribute(s) of relation(s). It consists of the selection, the projection, the joining and the dividing.

All programs have been developed successfully in C language on the ULTRIX-32 operating system of VAX-11/785.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้ประสบความสำเร็จลุล่วงได้ด้วยดี ด้วยความช่วยเหลือและสนับสนุนจากหลายฝ่าย

ขอกราบขอบพระคุณ ผศ.ดร. อัชฌา ธีรเชษฐมงคล ในฐานะอาจารย์ที่ปรึกษา ที่กรุณาให้คำปรึกษา แนะนำและแก้ไขปัญหาต่างๆ ตลอดจนการแก้ไขตรวจทานการจัดทำวิทยานิพนธ์ฉบับนี้จนสมบูรณ์

ขอกราบขอบพระคุณ ผศ.ดร. พูลพงษ์ บุญพราหมณ์ ในฐานะอาจารย์ที่ปรึกษาร่วม ที่กรุณาให้คำปรึกษา แนะนำและแก้ไขปัญหาต่างๆ

ขอกราบขอบพระคุณ คุณพ่อและคุณแม่ ที่ให้การสนับสนุนและเป็นกำลังใจ ด้วยดีตลอดมา

ขอขอบคุณ คุณนิษฐิดา นวลศรี ที่กรุณาให้คำปรึกษา แนะนำและแก้ไขปัญหาต่างๆ ที่เกิดขึ้นจากการใช้ระบบคอมพิวเตอร์ VAX-11/785 และระบบดำเนินงาน ULTRIX-32

ขอขอบคุณ เจ้าหน้าที่ประจำศูนย์คอมพิวเตอร์ที่ให้ความช่วยเหลือ และอำนวยความสะดวกในการใช้เครื่องคอมพิวเตอร์ ด้วยดีตลอดมา

ขอขอบคุณ คุณสุจิตรา โพธาวาณิช ที่เป็นกำลังใจในการทำวิทยานิพนธ์ ครั้งนี้

นอกจากนี้ขอขอบคุณ คณะกรรมการทุกท่าน ที่ช่วยตรวจและแก้ไขวิทยานิพนธ์

เกษมัณฑน์ แก้วอิ่ม

สารบัญ

รายการภาพ

บทที่ 1

บทนำ

1.1 ระบบฐานข้อมูล

ในปัจจุบันเราอาจกล่าวได้ว่า คอมพิวเตอร์ได้เข้ามามีบทบาทในทุกวงการอย่างกว้างขวาง อีกทั้งเทคโนโลยีและประสิทธิภาพของเครื่องคอมพิวเตอร์ในปัจจุบันก็ได้มีการพัฒนามาถึงจุดที่ค่อนข้างจะอิ่มตัว แนวโน้มในการพัฒนาของคอมพิวเตอร์ในปัจจุบันจึงหันมาให้ความสนใจกับการพัฒนา ซอฟแวร์ (software) กันมากขึ้น ระบบฐานข้อมูล (database system) และระบบจัดการฐานข้อมูล (database management system) เป็นปรากฏการใหม่ทางซอฟแวร์ที่มีคนให้ความสนใจกันมากในรอบสิบปีที่ผ่านมา

บ่อยครั้งที่เรามักจะเคยได้ยินคำว่า ฐานข้อมูล (database) แต่ไม่ทราบว่าคืออะไร หรือมีความหมายอย่างไร และมีประโยชน์อะไร โดยทั่วไปมักจะเข้าใจว่า ฐานข้อมูลก็คือ การนำเอาข้อมูลต่างๆ ที่ต้องการใช้งานมารวมกันไว้ ซึ่งความจริงแล้วความหมายของฐานข้อมูลไม่ใช่เพียงแต่นำข้อมูลมาสะสมหรือรวมกันไว้ ณ ที่แห่งเดียวกันเท่านั้น แต่ยังมีอีกหลายสิ่งที่เราจะต้องนำมาพิจารณา วัตถุประสงค์หลักของฐานข้อมูลก็คือ การจัดทำให้ข้อมูลหลักและความสัมพันธ์ของข้อมูลหลักเหล่านั้นเป็นอันหนึ่งอันเดียวกันอย่างเป็นระบบ ในระบบงานหรือองค์กรขนาดใหญ่ที่ไม่มีการนำเอาฐานข้อมูลมาใช้ มักจะเกิดปัญหาในการใช้งานของข้อมูลขึ้นอยู่เสมอ เช่น เกิดการซ้ำซ้อนของข้อมูล ข้อมูลขาดความอ่อนตัวไม่มีความยืดหยุ่น กล่าวคือ การซ้ำซ้อนของข้อมูลนั้นเป็นลักษณะของการจัดเก็บข้อมูลชนิดหรือประเภทเดียวกันในหลายหน่วยงานซ้ำกัน ซึ่งจะทำให้ข้อมูลเหล่านั้นมีความทันสมัยของข้อมูลที่แตกต่างกัน ส่วนการที่ข้อมูลขาดความอ่อนตัวไม่มีความยืดหยุ่นอาจเกิดจากความต้องการรายละเอียดของข้อมูลประเภทเดียวกัน แต่จะต้องนำมาจากหลายแห่ง

เป็นต้น โดยทั่วไปแล้วฐานข้อมูลจะเป็นส่วนที่ข้อมูลมีการใช้ร่วมกัน ดังนั้นวัตถุประสงค์ของการจัดทำฐานข้อมูลจะต้องมีการคำนึงถึงหลายสิ่งหลายอย่าง เช่น เมื่อมีการใช้ข้อมูลร่วมกันก็จะต้องมีผู้ที่ดูแลและรับผิดชอบข้อมูลทั้งหมด และในแต่ละประเภทขององค์กร ต้องมีการกำหนดมาตรฐานรูปแบบของข้อมูลให้เหมือนกัน ให้ผู้มีหน้าที่รับผิดชอบเท่านั้นที่มีสิทธิ์ใช้หรือเปลี่ยนแปลงข้อมูลได้ การปฏิบัติการกับฐานข้อมูลควรกระทำได้อย่างรวดเร็ว และมีประสิทธิภาพ เป็นต้น

วัตถุประสงค์ที่กล่าวมาทั้งหมด ไม่สามารถที่จะทำสำเร็จได้จากระบบใดระบบหนึ่งเพียงอย่างเดียวเท่านั้น จะต้องมีการนำวิธีการต่าง ๆ มาใช้เพื่อให้บรรลุถึงวัตถุประสงค์และความต้องการทั้งหลายที่ได้กล่าวมาข้างต้น การจัดการให้ได้มาซึ่งวัตถุประสงค์อาจรวม เรียกว่า ระบบฐานข้อมูล ซึ่งจะประกอบไปด้วย 3 ส่วนใหญ่ ๆ คือ

ก. ฐานข้อมูล

ฐานข้อมูล (database) คือข้อมูลของ สิ่งที่เราสนใจในหน่วยงาน หรือองค์กรหนึ่งๆ โดยจัดอยู่ในรูปลักษณะโครงสร้างของข้อมูลต่างๆ ของ สิ่งที่เราสนใจ (entity) และ ความสัมพันธ์ (relationship) ระหว่างสิ่งที่เราสนใจ เช่น ลูกค้า เจ้าหนี้ อาจารย์ และ นักศึกษา เหล่านี้จะคือสิ่งที่เราสนใจ ส่วนความสัมพันธ์ที่เกิดจากสิ่งสองสิ่ง (หรือมากกว่า) ที่มีความสัมพันธ์กัน เช่น ความสัมพันธ์ที่เกิดระหว่างอาจารย์และนักศึกษาอาจจะเป็น "การเป็นอาจารย์ที่ปรึกษา" เป็นต้น

ข. ระบบจัดการฐานข้อมูล

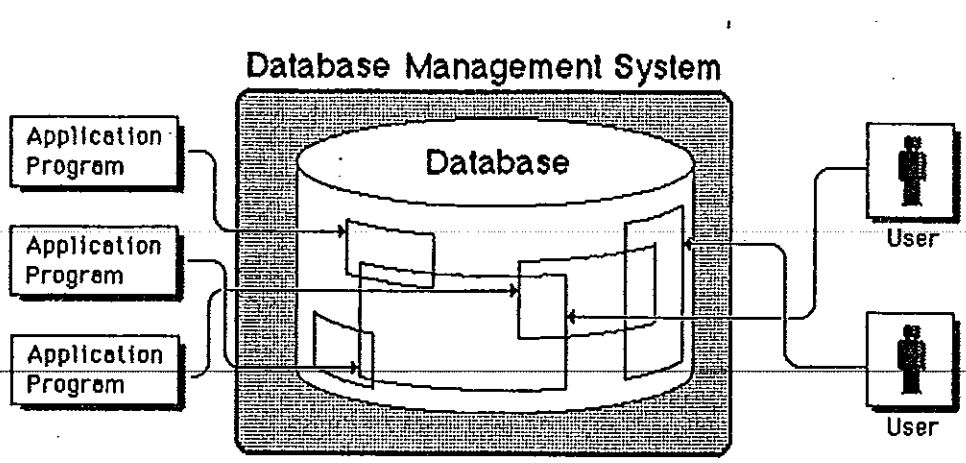
ระบบจัดการฐานข้อมูล (database management system - DBMS) เป็นโปรแกรมที่ใช้จัดการกับข้อมูลต่างๆ ที่อยู่ภายในฐานข้อมูล และเป็นสิ่งที่จำเป็นที่สุดในระบบฐานข้อมูลที่อำนวยความสะดวก ในการปฏิบัติการต่างๆ กับข้อมูลในฐานข้อมูล โดยจะทำหน้าที่นำข้อมูลมาไว้ร่วมกันในฐานข้อมูลอย่างมีระบบ รวมทั้งการกำหนดความ

หมาย และความสัมพันธ์ของข้อมูลในฐานข้อมูลของผู้ใช้ และสามารถนำเสนอข้อมูลที่มีทัศนะในการมองข้อมูลที่ต่างกันได้ เมื่อเกิดกรณีที่ผู้ใช้แต่ละคนเลือกใช้ข้อมูลเดียวกัน แต่ใช้ในงานที่แตกต่างกัน

ค. ผู้ใช้

เมื่อกล่าวถึงผู้ใช้ (user) ในระบบฐานข้อมูลจะแบ่งผู้ใช้ออกเป็น 3 ประเภทคือ ผู้ใช้ประเภทแรก ได้แก่ ผู้บริหารฐานข้อมูล (database administrator) ซึ่งเป็นบุคคลที่ใช้ฐานข้อมูลในลักษณะของการจัดการ การดูแลข้อมูล และการบริหารงานอื่นๆกับข้อมูลภายในฐานข้อมูล ซึ่งจะกล่าวโดยละเอียดในหัวข้อต่อไป ผู้ใช้ประเภทถัดมาได้แก่ ผู้ใช้ทั่วๆไปที่จะมาใช้ข้อมูลที่อยู่ภายในฐานข้อมูลนั้น โดยทำการติดต่อกับฐานข้อมูลโดยตรง ด้วยการป้อนคำถาม (query) จากแป้นพิมพ์และได้รับคำตอบทันทีทางจอภาพ สำหรับผู้ใช้ประเภทสุดท้ายได้แก่ โปรแกรมเมอร์ที่เขียนโปรแกรมประยุกต์ (application program) เรียกใช้ข้อมูลในฐานข้อมูล โดยข้อมูลเหล่านั้นได้มาจากการทำงานของ DBMS ร่วมด้วย

ในรูปที่ 1-1 เป็นแผนภาพของระบบฐานข้อมูลดังที่กล่าวมา จากแผนภาพแสดงให้เห็นว่าการที่ผู้ใช้ หรือโปรแกรมต่างๆ จะใช้ข้อมูลในฐานข้อมูลได้นั้นจะต้องผ่าน ระบบจัดการฐานข้อมูลก่อนเสมอ ดังนั้นระบบจัดการฐานข้อมูลจึงเปรียบเสมือนสื่อกลางที่เชื่อมโยงผู้ใช้กับข้อมูลในฐานข้อมูล



รูปที่ 1-1 แผนภาพแสดงส่วนประกอบของระบบฐานข้อมูล

1.2 ผู้บริหารฐานข้อมูล (Database Administrator - DBA)

โดยทั่วไปฐานข้อมูลขององค์กรหรือหน่วยงานหนึ่งๆ จะเกี่ยวข้องกับผู้ใช้หลายฝ่ายด้วยกัน จึงต้องมีปัจจัยที่ใช้ในการกำหนดความต้องการร่วมกัน เพื่อขจัดปัญหาในส่วนที่มีการเก็บข้อมูลชิ้นเดียวกันซ้ำๆ กัน และเพื่อที่จะประสานและกำหนดขั้นตอนในการออกแบบ วิธีการใช้งานและวิธีดูแลข้อมูลให้คงความถูกต้องอยู่เสมอ ผู้ที่ทำหน้าที่ในการจัดการและดูแลความเรียบร้อยต่างๆ ของฐานข้อมูลนี้จะเรียกว่า ผู้จัดการหรือผู้บริหารฐานข้อมูล (database administrator - DBA)

งานทางด้านจัดการฐานข้อมูลเป็นงานที่บริการผู้ใช้แต่ละคนในเรื่องของข้อมูล ดังนั้นผู้บริหารฐานข้อมูลจะต้องมีอำนาจหน้าที่สูงพอที่จะทำการควบคุมและ รับผิดชอบต่อโครงสร้างของข้อมูลทั้งหมดตลอดจนถึงวิธีการเข้าถึงข้อมูล ผู้ที่จะจัดการและบริหารฐานข้อมูลได้จะต้องเข้าใจระบบงานของ องค์กรหรือหน่วยงานนั้น ๆ เป็นอย่างดี ต้องทราบถึงลักษณะการใช้ข้อมูลของฝ่ายต่างๆ นอกจากนี้ยังต้องออกแบบการใช้งาน และดูแลรักษาตลอดจนการรักษาความปลอดภัยของข้อมูล งานหลักของผู้บริหารฐานข้อมูลที่สำคัญอีกสิ่งหนึ่งก็คือ จะต้องมองถึงความต้องการข้อมูลในอนาคตได้ไม่ยิ่งหย่อนไปกว่าความต้องการในปัจจุบัน ดังนั้นการออกแบบฐานข้อมูลจะต้องอ่อนตัวหรือมีความอิสระจากข้อมูลมากที่สุดเท่าที่จะทำได้

หน้าที่ของผู้บริหารฐานข้อมูลจึงอาจสรุปได้อย่างกว้าง ๆ [C.J. Date (1986)] ดังนี้คือ

1. รวบรวมสิ่งต่างๆ ที่หน่วยงานสนใจจะบันทึกข้อมูล พร้อมทั้งความสัมพันธ์ของสิ่งนั้นๆ

2. ออกแบบโครงสร้างของข้อมูลที่จะจัดเก็บในฐานข้อมูล รวมถึงวิธีการเข้าถึงข้อมูลเหล่านั้น

3. ติดต่อและให้บริการแก่ผู้ใช้ที่มีความประสงค์จะใช้ข้อมูลภายในฐานข้อมูล

4. กำหนดขั้นตอนเพื่อรักษาความปลอดภัยของข้อมูล โดยการกำหนดสิทธิในการใช้ข้อมูล เช่น อาจจะให้ฐานข้อมูลได้ในลักษณะที่เรียกข้อมูลมาดูได้เพียงอย่างเดียว หรืออาจจะมีสิทธิที่จะเปลี่ยนแปลงข้อมูลภายในฐานข้อมูลได้ เป็นต้น

5. กำหนดวิธีการในการสำรองข้อมูล และการกู้ข้อมูลเมื่อเกิดเหตุการณ์ที่ข้อมูลภายในฐานข้อมูลเสียหายขึ้น

6. ติดตาม ดูแลการใช้งานของข้อมูล ตลอดจนถึงความต้องการใช้งานใหม่ๆ ของผู้ใช้ เพื่อทำให้ฐานข้อมูลมีความทันสมัยอยู่เสมอ

1.3 สถาปัตยกรรมของระบบฐานข้อมูล

ในการใช้ระบบจัดการฐานข้อมูล โดยทั่วไปแบ่งรูปแบบของการมองภาพข้อมูลในฐานข้อมูลออกเป็น 3 ระดับ คือ ระดับ Internal ระดับ Conceptual และ ระดับ External [E.B Fernandez(1981)] ดัง รูปที่ 1-2

ก. Internal schema

ในระดับนี้ รูปแบบของข้อมูลจะมีความหมายที่ใกล้ชิดกับ การจัดเก็บทางกายภาพ (physical representation) มากที่สุด โดยจะเป็นการอธิบายถึงวิธีการ

จัดเก็บข้อมูล ในสื่อบันทึกข้อมูล เช่น เทปแม่เหล็ก และ จานแม่เหล็ก เป็นต้น

ข. External schema

ในระดับนี้จะเป็นรูปแบบข้อมูลส่วนที่ใกล้ชิดกับผู้ใช้มากที่สุด ในอีกความหมายหนึ่งก็คือการมองข้อมูลในฐานข้อมูลเฉพาะส่วนของผู้ใช้แต่ละคนนั่นเอง

ค. Conceptual schema

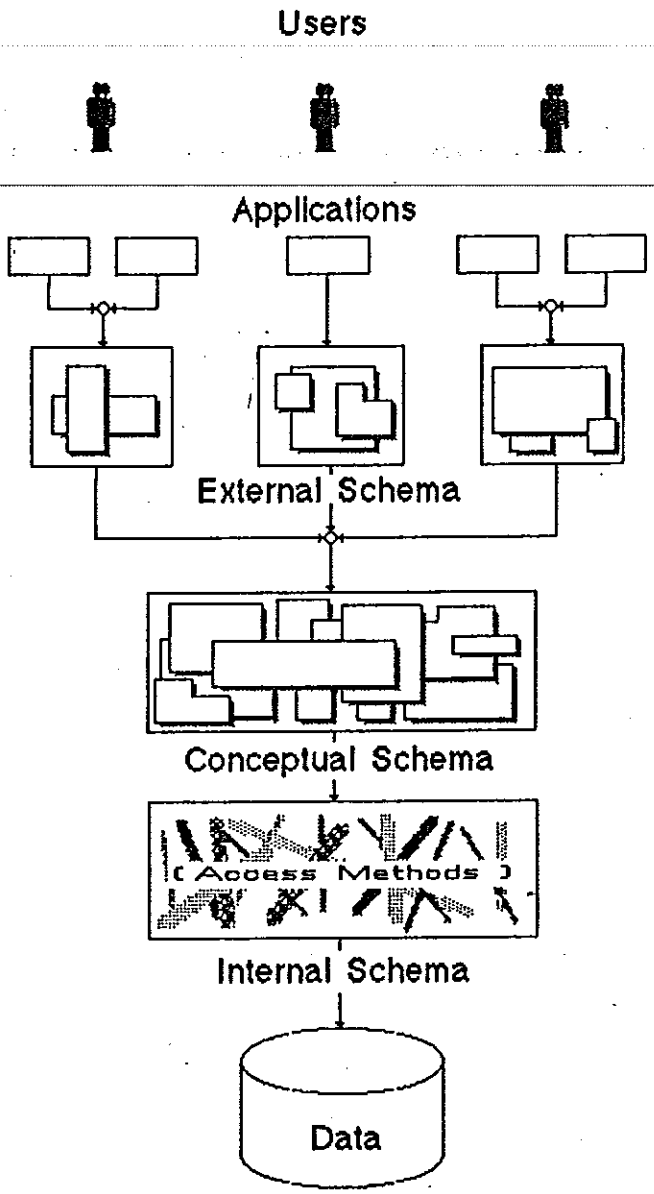
เป็นส่วนที่อยู่ระหว่าง Internal schema กับ External schema ในระดับนี้จะเป็นการมองข้อมูลภายในฐานข้อมูลในภาพรวม โดยผู้บริหารฐานข้อมูลจะเป็นผู้รับผิดชอบในการวิเคราะห์ และออกแบบให้สอดคล้องกับรูปแบบข้อมูลที่ระบบจัดการฐานข้อมูลใช้

1.4 รูปแบบข้อมูล

ระบบจัดการฐานข้อมูลทุกระบบ จะต้องอิงรูปแบบข้อมูล (data model) รูปแบบใดรูปแบบหนึ่ง เพื่อเป็นรูปแบบสำหรับการทำงานกับข้อมูลในฐานข้อมูลของผู้ใช้ ในระบบจัดการฐานข้อมูลที่มีใช้กันอยู่ในปัจจุบัน อาจแบ่งรูปแบบข้อมูลที่ใช้ออกเป็น 3 รูปแบบมาตรฐาน ดังนี้

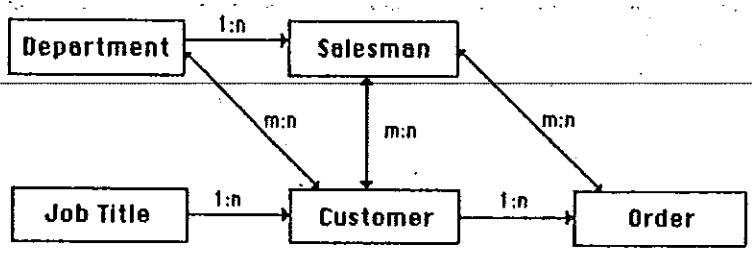
ก. Network data model [S. Atre (1980)]

รูปแบบของข้อมูลนี้ มีลักษณะความสัมพันธ์ของข้อมูลเป็น เครือข่าย (network) โดยที่ความสัมพันธ์ระหว่างสิ่งต่างๆ ที่สนใจจะบันทึกข้อมูลลงในฐานข้อมูลในรูปแบบเครือข่ายนี้ นับว่าเป็นเรื่องที่ค่อนข้างจะยุ่งยาก เนื่องจากรูปแบบของความสัมพันธ์สามารถที่จะมีได้ทั้ง 3 รูปแบบ คือ แบบ one-to-one (1:1) one-to-many (1:n) และ many-to-many (m:n) รูป 1-3 แสดงตัวอย่างของกลุ่มของสิ่งที่สนใจกลุ่มหนึ่งที่มีการนำมาจัดรูปแบบโครงสร้างความสัมพันธ์เป็นแบบเครือข่าย ซึ่งจะเห็นว่า



รูปที่ 1-2 ลำดับการของภาพในฐานข้อมูล

department มีความสัมพันธ์แบบ one-to-many กับsalesman และในขณะเดียวกันก็สามารถที่จะมีความสัมพันธ์แบบ many-to-many กับcustomer ได้ เป็นต้น



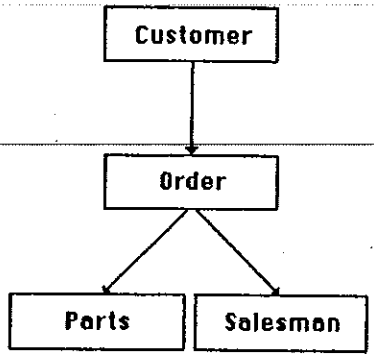
รูปที่ 1-3 ตัวอย่างการจัดข้อมูลในรูปแบบของเครือข่าย

ในปัจจุบัน ระบบจัดการกับฐานข้อมูลที่มีการจัดเก็บข้อมูลแบบเครือข่ายจะใช้มาตรฐานของ CODASYL (Conference On Data System Language) ซึ่งอิงรูปแบบข้อมูลซึ่งใช้กับภาษา COBOL เป็นหลักในการอธิบายรูปแบบข้อมูลและการปฏิบัติการ

ข. Hierachical data model หรือ Tree Structure [S. Atre (1980)]

รูปแบบของข้อมูลนี้ มีลักษณะความสัมพันธ์ของข้อมูลเป็นลำดับชั้นเหมือนกับโครงสร้างของต้นไม้ (tree structure) การจัดความสัมพันธ์ให้กับข้อมูลในรูปแบบข้อมูลชนิดนี้จะมีความยุ่งยากน้อยกว่าในแบบเครือข่าย ทั้งนี้เพราะมีรูปแบบของความสัมพันธ์เพียง 2 รูปแบบเท่านั้น คือ one-to-one(1:1) และ one-to-many (1:n) เท่านั้น รูป 1-4 แสดงตัวอย่างของโครงสร้างฐานข้อมูลแบบ hierachical data model จากรูปแสดงให้เห็นว่า ลูกค้า(customer) มีความสัมพันธ์แบบ one-to-many กับการสั่งซื้อ (order) และการสั่งซื้อมีความสัมพันธ์แบบ one-to-many กับชิ้นส่วน (parts) และ ผู้ขาย(salesman) ซึ่งหมายความว่า

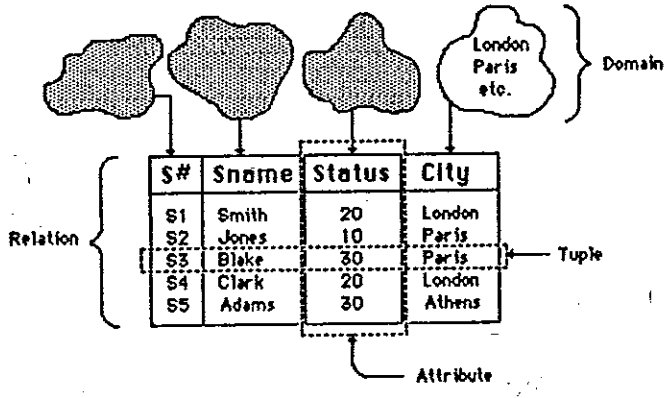
ลูกค้าหนึ่งคนสามารถที่จะสั่งสินค้าได้หลายครั้ง ในแต่ละครั้งของการสั่งสามารถที่จะสั่งสินค้าได้หลายชนิด และอาจสั่งจากผู้ขายหลายคน



รูปที่ 1-4 ตัวอย่างการจัดข้อมูลในรูปแบบของโครงสร้างต้นไม้

ค. Relational data model [C.J. Date (1986)]

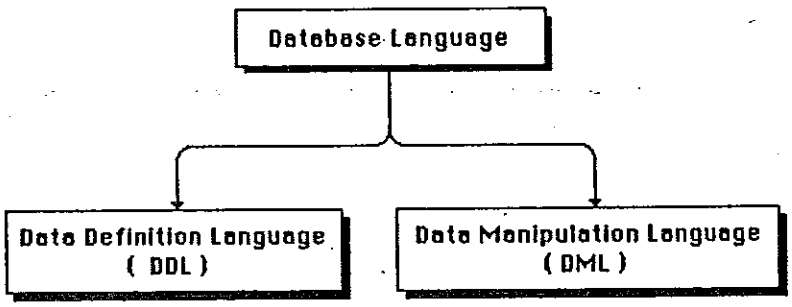
การจัดรูปแบบข้อมูลนี้เป็น รูปแบบที่มีการใช้กันมากในระบบจัดการฐานข้อมูลที่มีใช้อยู่กันในปัจจุบัน เป็นการมองข้อมูลทั้งหมดเก็บอยู่ในรูปของ ตารางแสดงความสัมพันธ์ หรือเรียกอีกชื่อหนึ่งว่า รีเลชัน (relation) ตามศัพท์ที่ใช้ในทางคณิตศาสตร์ ข้อมูลในแต่ละแถวจะเรียกว่า tuple และข้อมูลที่อยู่ในแต่ละสดมภ์จะเรียก attribute value จากรูป 1-5 เป็นตัวอย่างของข้อมูลที่มีโครงสร้างแบบรีเลชัน รายละเอียดและนิยามต่างๆ ที่เกี่ยวกับการจัดรูปแบบข้อมูล แบบรีเลชัน จะกล่าวถึงโดยละเอียดในหัวข้อต่อไป



รูปที่ 1-5 ตัวอย่างการจัดรูปแบบของข้อมูลแบบรีเลชัน

1.5 ภาษาฐานข้อมูล

ระบบจัดการฐานข้อมูล(DBMS) เป็นส่วนที่มีความสำคัญที่สุดในระบบฐานข้อมูลที่จะอำนวยความสะดวกในการกำหนดรูปแบบและลักษณะของข้อมูลที่จะเก็บในฐานข้อมูลและในการเพิ่ม ลบ เปลี่ยนแปลงและถามคำถามเกี่ยวกับข้อมูลภายในฐานข้อมูล ซึ่งผู้ใช้หรือโปรแกรมจะติดต่อกับฐานข้อมูลได้ต้องผ่านการทำงานของ DBMS เสมอ ในการที่จะเข้าถึง ข้อมูลต่าง ๆ ภายในฐานข้อมูลนั้นจำเป็นที่จะต้องมีภาษาเฉพาะใช้ ในการติดต่อกับระบบจัดการฐานข้อมูล ภาษาที่ใช้เรียกว่า **ภาษาฐานข้อมูล(database language)** ซึ่งโดยทั่วไปแบ่งออกเป็น 2 ส่วนคือ Data Definition Language (DDL) และ Data Manipulation Language (DML) [C.J. Date(1986)] ดังแสดงในรูปที่ 1-6



รูปที่ 1-6 ภาษาฐานข้อมูล

1.5.1 Data Definition Language - DDL

ข้อมูลที่จะทำการจัดเก็บในฐานข้อมูลนั้น จะต้องมีการกำหนดโครงสร้างรายละเอียดความหมาย และความสัมพันธ์ต่างๆ ของข้อมูลก่อน DDLเป็นภาษาฐานข้อมูลที่ใช้กำหนด สิ่งที่กล่าวมาข้างต้นให้กับข้อมูลก่อนทำการจัดเก็บไว้ในฐานข้อมูล รูปที่ 1-7 แสดงตัวอย่างของ DDL ที่ใช้กับระบบจัดการฐานข้อมูลชื่อ INGRES ซึ่งเป็นระบบจัดการฐานข้อมูลระบบหนึ่งที่มีการจัดรูปแบบของข้อมูลเป็น แบบรีเลชั่น

```
CREATE  SALE ( S#       = TEXT(5), SNAME    = TEXT(20),
               STATUS   = I2, CITY      = TEXT(15) )
```

หมายเหตุ : TEXT(n) เป็นการกำหนดว่าค่าของ attribute จะเป็นข้อมูลที่เป็นตัวอักษรขนาด n ไบท์

I2 เป็นข้อมูลที่เป็นเลขจำนวนเต็มขนาด 2 หลัก

รูปที่ 1-7 ตัวอย่าง DDL ในระบบจัดการฐานข้อมูล INGRES

จากรูปที่ 1-7 เป็นการกำหนดให้ระบบจัดการฐานข้อมูลสร้างรีเลชั่นใหม่ที่ชื่อ SALE โดยจะประกอบไปด้วยรายละเอียดคือ attribute ชื่อ "S#" มีค่าข้อมูลเป็นตัวอักษรขนาดไม่เกิน 5 ตัวอักษร attribute ชื่อ "SNAME" มีค่าข้อมูลเป็นตัวอักษรขนาดไม่เกิน 20 ตัวอักษร attribute ชื่อ "STATUS" เป็นข้อมูลจำนวนเต็มที่มีค่าไม่เกิน 2 หลัก และ attribute ชื่อ "CITY" มีค่าเป็นตัวอักษรขนาดไม่เกิน 15 ตัวอักษร

1.5.2 Data Manipulation Language - DML

เมื่อมีการกำหนดโครงสร้าง รายละเอียด และความหมายของข้อมูลเรียบร้อยแล้ว ก็จะต้องมีภาษาที่ใช้ในการปฏิบัติการกับข้อมูลเหล่านั้นได้ โดยการปฏิบัติเหล่านี้คือ การเพิ่มข้อมูลเข้าในฐานข้อมูล (insertion) การลบข้อมูลออกจากฐานข้อมูล (deletion) การปรับปรุงเปลี่ยนค่าข้อมูลในฐานข้อมูล (amendment) และการหยิบชิ้นข้อมูลที่ต้องการในฐานข้อมูล (retrieve) หรือการป้อนคำถามเพื่อถามหาชิ้นข้อมูลที่ต้องการ (query) ซึ่งในส่วนนี้ก็คือการใช้ DML เพื่อปฏิบัติการกับข้อมูลในฐานข้อมูลนั่นเอง วิธีการปฏิบัติการจะขึ้นอยู่กับลักษณะรูปแบบของข้อมูล(data model) ที่จัดเก็บ สำหรับการทำวิทยานิพนธ์นี้ใช้รูปแบบของการจัดเก็บข้อมูลแบบรีเลชัน (relation) ซึ่งลักษณะของ DML ที่จะใช้ในการปฏิบัติการกับข้อมูลเพื่อให้ได้ซึ่งรายละเอียดของข้อมูลที่มีการจัดโครงสร้างของข้อมูลแบบรีเลชันนี้มีอยู่ 2 แบบ [C.J. Date(1986)] ดังนี้คือ

ก. Relational calculus

การปฏิบัติการแบบนี้จะเป็นการได้มาซึ่งรายละเอียดของข้อมูลในฐานข้อมูล โดยการบ่งบอกถึงคุณลักษณะของข้อมูลที่ต้องการ ดังที่กล่าวมาแล้วว่ารูปแบบของข้อมูลที่จัดเก็บแบบรีเลชันนั้นอิงอยู่กับรูปแบบของเซท ดังนั้นอาจกล่าวได้ว่า relational calculus ก็คือ การบอกค่าสมาชิกในเซทด้วยการเขียนอธิบาย คุณลักษณะของสมาชิกในเซท เช่น จาก DDL ของรีเลชัน SALE ในรูปที่ 1-7 ถ้าเราต้องการค่า S# ของสมาชิก (ซึ่งในที่นี้คือ tuple) ของรีเลชัน SALE ที่มีค่า STATUS มากกว่า 15 เราก็อาจเขียนคำถามในรูป relational calculus ได้ดังนี้

TEMP(SALE.S#): (SALE.STATUS > 15)

ซึ่งหมายความว่าต้องการค่า S# จากรีเลชันชื่อ SALE ที่มีค่าของ STATUS มากกว่า 15 และนำผลลัพธ์ที่ได้จัดเก็บไว้ใน รีเลชันที่ชื่อ TEMP

ข. Relational algebga

การปฏิบัติการแบบเป็นการได้มาซึ่งข้อมูลในฐานข้อมูล โดยปฏิบัติการกับข้อมูลโดยตรง ด้วยการบอกลำดับและวิธีการได้มาซึ่งข้อมูลอย่างเป็นขั้นตอน ซึ่งคำสั่งที่ใช้ทั่วไปทั้งหมดสามารถแบ่งออกได้เป็น 2 กลุ่ม คือ ในกลุ่มแรกจะเป็นคำสั่งปฏิบัติการมาตรฐานที่ใช้กับเซท (set operation) คำสั่งในกลุ่มนี้ประกอบไปด้วยคำสั่ง intersect union difference และ cross product ส่วนกลุ่มคำสั่งที่เหลือจะเป็นคำสั่ง ปฏิบัติการพิเศษ นอกเหนือจากคำสั่งปฏิบัติการกับเซทดังกล่าว คำสั่งในชุดนี้ประกอบด้วยคำสั่ง select project join และ divide คำสั่งปฏิบัติการทั้งหมดนี้จะกล่าวถึงโดยละเอียดในบทต่อไป

1.6 ความเป็นมาในการจัดทำวิทยานิพนธ์

จากสิ่งที่กล่าวมาทั้งหมด พอที่จะสรุปได้ว่าสำหรับระบบฐานข้อมูลใดๆ นั้น สิ่งที่เป็นหัวใจของระบบฐานข้อมูล ซึ่งจะทำการดำเนินงานต่างๆ กับข้อมูลในฐานข้อมูลก็คือ ระบบจัดการฐานข้อมูล(DBMS) แต่ในการที่จะทำการพัฒนาระบบจัดฐานข้อมูลที่สมบูรณ์ได้นั้นจะต้องอาศัยปัจจัยอีกหลายประการ เช่น โปรแกรมสำหรับการกำหนดโครงสร้าง รายละเอียด และความหมายของข้อมูลที่จะทำการเก็บในฐานข้อมูล โปรแกรมในการปฏิบัติการกับฐานข้อมูลด้วยการป้อนคำถาม (query) เพื่อหยิบชิ้นข้อมูล (retrieve) ที่ต้องการในฐานข้อมูล และโปรแกรมคำสั่งปฏิบัติการพื้นฐานสำหรับเพิ่ม ลบ และเปลี่ยนแปลงแก้ไขข้อมูล ก็เป็นปัจจัยหนึ่งที่มีความสำคัญสำหรับระบบจัดการฐานข้อมูลด้วยเช่นกัน ในการทำวิทยานิพนธ์ครั้งนี้ มุ่งการพัฒนาโปรแกรมในการหยิบชิ้นข้อมูลที่ต้องการด้วยคำสั่งปฏิบัติการฐานข้อมูลแบบพีชคณิตสัมพันธ์ ซึ่งใช้กับฐานข้อมูลที่มีโครงสร้างแบบรีเลชัน

1.7 วัตถุประสงค์ในการทำวิทยานิพนธ์

ในการทำวิทยานิพนธ์นี้จะทำการศึกษาหลักการและเทคนิคต่างๆ ที่จะใช้ในการพัฒนาคำสั่งปฏิบัติการฐานข้อมูลแบบพีชคณิตสัมพันธ์ (Relational Algebra Operation) ของฐานข้อมูลซึ่งใช้ในรูปแบบรีเลชัน ซึ่งคำสั่งทั้งหมดนี้จัดว่าเป็นคำสั่งพื้นฐานในการปฏิบัติการฐานข้อมูล จากนั้นเป็นการพัฒนาโปรแกรมดำเนินงาน โปรแกรมที่ได้นี้สามารถที่จะนำไปใช้เป็นโปรแกรมพื้นฐาน สำหรับใช้ร่วมกับการพัฒนาโปรแกรมส่วนอื่นๆ ของ ระบบจัดการฐานข้อมูลต่อไป อาทิ เช่น โปรแกรมการกำหนดโครงสร้าง กำหนดความหมาย รายละเอียดของข้อมูลในฐานข้อมูล หรือ โปรแกรมการปฏิบัติการกับฐานข้อมูลด้วยการป้อนคำถาม เป็นต้น ซึ่งโปรแกรมเหล่านี้จะทำให้ได้ระบบจัดการฐานข้อมูลที่สมบูรณ์ขั้นต้นระบบหนึ่ง

1.8 เนื้อหาในการจัดทำวิทยานิพนธ์

ในการจัดทำวิทยานิพนธ์ฉบับนี้นั้น ได้ทำการแบ่งเนื้อหาของการจัดทำออกเป็น 5 บท ดังนี้

บทที่ 1 กล่าวถึง เรื่องทั่วๆ ไปเกี่ยวกับฐานข้อมูลและสิ่งที่เกี่ยวข้อง เช่นระบบฐานข้อมูล ผู้จัดการฐานข้อมูล สถาปัตยกรรมของระบบฐานข้อมูล รูปแบบข้อมูล ภาษาฐานข้อมูล เป็นต้น

บทที่ 2 กล่าวถึง รูปแบบข้อมูลแบบรีเลชัน นิยามเบื้องต้น และนิยามคำสั่งปฏิบัติการแบบพีชคณิตสัมพันธ์

บทที่ 3 กล่าวถึง ขอบเขตในการจัดทำวิทยานิพนธ์ ขั้นตอนการดำเนินงานและรายละเอียดอื่นๆ การออกแบบระบบ และการพัฒนาโปรแกรม

บทที่ 4 เป็นเอกสารสำหรับผู้ใช้ กล่าวถึง สิ่งที่ผู้ใช้ควรทราบก่อนการใช้งาน องค์ประกอบหลักของจอภาพในการใช้งาน แผนภาพสรุปโครงสร้างของรายการหลักและรายการย่อย

ข้อแนะนำในการใช้แป้นพิมพ์เพื่อควบคุมการทำงานต่างๆ ของโปรแกรม และการใช้โปรแกรมในรายการหลักและรายการย่อย

บทที่ 6 บทสรุป กล่าวถึง ผลการจัดทำวิทยานิพนธ์ ปัญหาและข้อเสนอแนะ ประโยชน์ที่ได้รับจากการจัดทำวิทยานิพนธ์

ภาคผนวก ประกอบด้วย โปรแกรมในส่วนของคำสั่งปฏิบัติการฐานข้อมูล ตัวอย่างข้อมูลในแฟ้มข้อมูล ตัวอย่างผลที่ได้จากการทำงานของโปรแกรม คำศัพท์ทั่วไปและความหมาย

บทที่ 2

คำสั่งปฏิบัติการแบบพีชคณิตสัมพันธ์

เนื่องจากรูปแบบข้อมูล แบบรีเลชั่นได้อาศัยพื้นฐานมาจากแนวความคิดทางคณิตศาสตร์ที่เกี่ยวข้องกับทฤษฎีทางเซท ดังนั้นนิยามต่างๆ ของรีเลชั่นจึงนิยามอยู่ในรูปของเซทเป็นส่วนใหญ่

นิยาม กำหนดให้ $D_1, D_2, \ldots, D_n$ เป็นเซท ถ้า R เป็นเซทของ n-tuple ที่มีค่า $(d_1, d_2, \ldots, d_n)$ โดยที่ $d_1 \in D_1, d_2 \in D_2, \ldots, d_n \in D_n$ ตามลำดับแล้ว เราจะเรียก R ว่าเป็น relation ที่มีดีกรี(degree)n D_j โดยที่ $j = 1,\ldots,n$ เรียกว่าเป็น โดเมน(Domain) ของ รีเลชั่น R และ d_i เรียกว่าเป็น attribute value ของ D_i โดยที่ $i = 1,\ldots,n$

ถ้า n = 1 จะเรียกว่า R เป็น unary รีเลชั่น

n = 2 จะเรียกว่า R เป็น binary รีเลชั่น

n = 3 จะเรียกว่า R เป็น ternary รีเลชั่น

n > 3 จะเรียกว่า R เป็น n-ary รีเลชั่น

ตัวอย่าง 2.1 จากรูปที่ 1-5 เราสามารถที่จะเขียน tuple ของรีเลชั่น Sale(S#,SNAME, STATUS,CITY) ให้อยู่ในรูปแบบของ set ได้ดังนี้ คือ

Sale = {(S1,Smith,20,London),(S2,Jones,10,Paris),
(S3,Blake,30,Paris),(S4,Clark,20,London),
(S5,Adams,30,Athens)}

ในบางครั้งเราสามารถที่จะให้ความหมายของ รีเลชั่นในอีกความหมายหนึ่งว่า **รีเลชั่นเป็นสับเซท(subset) ของผลคูณ(Cross Product) ของเซท 2 เซท ขึ้นไปนั่นเอง**

<u>ตัวอย่างที่ 2.2</u> ถ้ากำหนดให้ A,B เป็นเซทซึ่งมีสมาชิกดังนี้

$$A = \{a_1, a_2\} \quad \text{และ} \quad B = \{b_1, b_2, b_3\}$$

ดังนั้น cross product ของ A และ B คือ

$$A \otimes B = \{(a_1,b_1),(a_1,b_2),(a_1,b_3),(a_2,b_1),(a_2,b_2),(a_2,b_3)\}$$

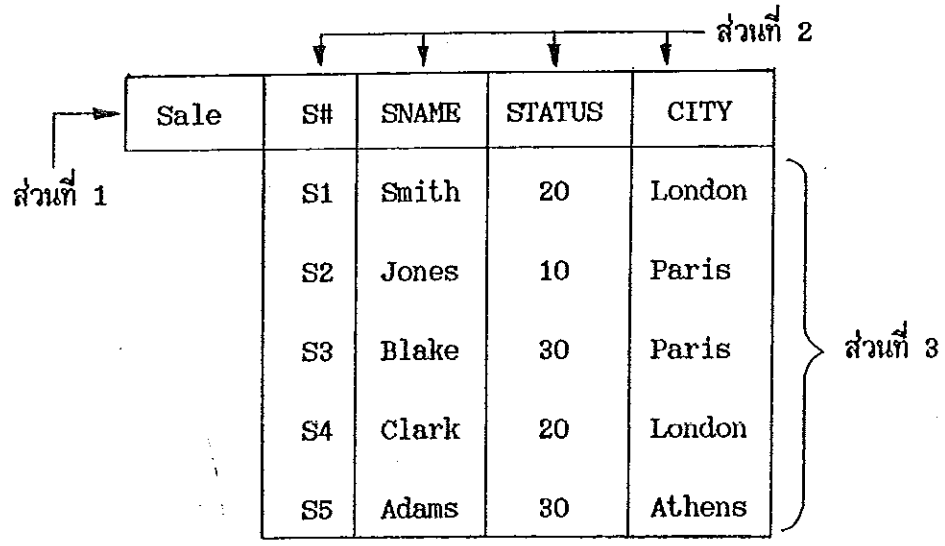
และถ้าให้ R เป็นเซทที่มีค่าดังนี้

$$R = \{(a_2,b_3),(a_1,b_1)\}$$

เราจะได้ว่า $R \subseteq A \otimes B$

นั่นคือ **R เป็นรีเลชั่น (relation) หนึ่งระหว่างเซท A และเซท B**

เพื่อความสะดวกในการทำความเข้าใจของผู้ใช้ เราจึงมักที่จะเขียนข้อมูลในรีเลชั่นต่างๆ ให้อยู่ในรูปของตาราง ดังตัวอย่างรีเลชั่น Sale ของตัวอย่างที่ 2.1 ดังแสดงในรูปที่ 2-1



<u>รูปที่ 2-1</u> รีเลชั่นในรูปตาราง

จากรูป 2.1 เป็นการเขียนตัวอย่างข้อมูลในรูปของตารางหรือรีเลชั่น โดยจากรูปประกอบด้วยรายละเอียด 3 ส่วน ดังนี้คือ

ส่วนที่ 1 : เป็นชื่อของรีเลชั่น

ส่วนที่ 2 : เป็นส่วนของ attribute-name ของรีเลชั่นเพื่อบอกว่ารีเลชั่นนั้นจะประกอบด้วยรายละเอียด (attribute) ชื่ออะไรบ้าง สมมุติว่าชื่อของ attribute แทนด้วย attribute-name จากรูปที่ 2-1 รีเลชั่น Sale ประกอบไปด้วย 4 attribute-name คือ S# SNAME STATUS และ CITY

ส่วนที่ 3 : ในส่วนนี้เราจะเรียกว่าเป็น occurrence หรือ instance ซึ่งหมายถึงตัวอย่างของข้อมูลในรีเลชั่นนั้น ในส่วนที่ 3 นี้จะมีลักษณะการเรียกข้อมูลอยู่ 2 ลักษณะ คือ

- ข้อมูลที่อยู่ในแนวแต่ละสดมม์(column) เรียกว่า attribute-value
- ข้อมูลที่อยู่ในแต่ละแถว(row) เรียกว่า tuple

จาก attribute-name (ในส่วนที่ 2) และ attribute-value (ในส่วนที่ 3) มักจะรวมเรียก ว่าเป็น attribute

คุณสมบัติของรีเลชั่น

1. ข้อมูลในแต่ละรีเลชั่นจะไม่มี tuple ที่ซ้ำกันเลย ผลจากคุณสมบัติข้อนี้ทำให้เราสามารถกำหนด key เพื่อใช้เป็นตัวแทนของ tuple สำหรับเรียกหรืออ้างอิงถึงและ key ที่กำหนดขึ้นจะประกอบด้วย attribute เดียว (single key) หรือซึ่งอาจจะเป็นกลุ่มของ attribute (compound key) ก็ได้

2. tuple ที่อยู่ภายในรีเลชั่นไม่จำเป็นต้องมีการเรียงค่าตาม key จากมากไปหาน้อยหรือน้อยไปหามาก

3. attribute ในแต่ละรีเลชั่นไม่จำเป็นต้องมีการเรียงลำดับจากขวาไปซ้ายหรือซ้ายไปขวา

4. attribute-value จะต้องเป็นค่าโดด (atomic value)

2.1 นิยามเบื้องต้น

นิยาม กำหนดให้ m-tuple $r=(r_1,\ldots,r_m)$ และ n-tuple $s=(s_1,\ldots,s_n)$

concatenation ของ r กับ s $(r^\frown s)$ คือ (m+n)-tuple ซึ่งถูกกำหนดโดย

$$r^\frown s = (r_1,\ldots,r_m,s_1,\ldots,s_n)$$

ตัวอย่าง

ถ้า r = (1,2,x) และ s = (a,z,3) ดังนั้น $r^\frown s$ = (1,2,x,a,z,3)

นิยาม ให้ R เป็น n-ary รีเลชั่น, r เป็น tuple หนึ่งของ R และ $\{D_1,\ldots,D_n\}$ เป็นโดเมนของ R ดังนั้น

1. $r[D_i]$ จะหมายถึงค่า attribute ลำดับที่ i ของ tuple r
2. ถ้า $A \subseteq \{D_1,\ldots,D_n\}$ แล้ว
   (ก) r[A] คือ tuple ใน R ซึ่งประกอบด้วยค่าของข้อมูลตามลำดับโดเมนในเซท A
   (ข) R(A) = { r[A] }

ตัวอย่าง

ถ้า r = (a,2,f) เป็น tuple หนึ่งใน R(D) เมื่อ $D = \{D_1,D_2,D_3\}$ และ ถ้า $A = \{D_1,D_3\}$

(ก) r[A] = (a,f)

(ข) $R(D_1,D_2,D_3)$ =

a	2	f
b	1	g
c	3	f
d	3	g

$R(D_1)$ = a, b, c, d

$R(D_3,D_2)$ =

f	2
g	1
f	3
g	3

นิยาม ให้ T(x,y) เป็น binary รีเลชั่น, เซทจินตภาพ(image set)ของ x ภายใต้ T กำหนดโดย

$$g_T(x) = \{y:(x,y) \in T\}$$

ตัวอย่าง

ถ้า $R(D_1,D_2) = \{(1,a),(1,b),(2,c),(1,d)\}$

ดังนั้น $g_R(D_1=1)=\{a,b,d\}$, $g_R(D_1=2)=\{c\}$, $g_R(D_1=3)=\{\}$

นิยาม ให้ R เป็น n-ary รีเลชั่น $\{D_1,\ldots,D_n\}$ เป็นโดเมนของ R, $A=\{D_1,\ldots,D_k\}$ ($k <= n$) ดังนั้น

1. $\bar{A} = \{D_1,\ldots,D_n\} - A$

2. ถ้า r เป็น tuple ของ R แล้ว

$$g_R(r[\bar{A}]) = \{s : s \in R(A) \wedge (r[\bar{A}], s) \in R(\bar{A} \frown A)\}$$

<u>ตัวอย่าง</u> กำหนดให้ R คือ $R(D_1, D_2, D_3, D_4, D_5)$

1	a	x	f	2
2	a	y	g	3
1	b	x	f	2
2	c	y	b	3

และ $A = \{D_3, D_2, D_4\}$ ดังนั้น $\bar{A} = \{D_1, D_5\}$, ให้ r = (1,a,x,f,2)

เพราะฉะนั้น r[A] = (x,a,f) , $r[\bar{A}]$ = (1,2) และ

$$g_R(r[\bar{A}]) = g_R((1,2)) = \{(x,a,f),(x,b,f)\}$$

<u>นิยาม</u> เซทของ attribute รีเลชัน A และ B จะเรียกว่า <u>เข้ากันได้</u>(Compatible) ถ้า A และ B มีขนาด(degree)ที่เท่ากัน และ มีโดเมนที่ตำแหน่งสมนัยกันมีค่าข้อมูลชนิดเดียวกัน

2.2 <u>นิยามคำสั่งปฏิบัติการข้อมูลแบบพีชคณิตสัมพันธ์</u>

สำหรับตัวอย่างของคำสั่งปฏิบัติการในหัวข้อนี้จะใช้รีเลชั่นในรูปที่ 2-2 เป็นตัวอย่างข้อมูลในฐานข้อมูลที่ใช้

PRODUCT (<u>CODE</u>, COST, PRICE)

A	5	8
B	4	4
C	6	9

PRODUCT1 (CODE)

A
B

BUYER (NAME, ITEM)		BUYER1 (NAME, ITEM)	
SMITH	A	JUDY	C
JONES	B	ROBERT	A
ADAMS	A	SMITH	A
SMITH	B	JACK	B
JONES	A	ADAMS	A
SMITH	C		

รูปที่ 2-2 ตัวอย่างข้อมูลแบบรีเลชั่น

2.2.1 Intersection

นิยาม ให้ R และ S เป็น n-ary รีเลชั่นที่ เข้ากันได้, r และ s เป็น tuple ของ R และ S ตามลำดับ, intersection (แทนด้วยสัญลักษณ์ $\cap$) ของ R กับ S สามารถกำหนดได้ ดังนี้

$$R \cap S = \{t : t \in R \wedge t \in S\}$$

ผลลัพธ์จากการ intersection ระหว่างรีเลชั่น R และ S จะได้รีเลชั่นใหม่ซึ่งมีดีกรีเท่ากับดีกรีของ R หรือ S พร้อมทั้ง attribute ชุดเดียวกันกับของ R และ S ด้วย

ตัวอย่าง

จากรูปที่ 2-2 BUYER และ BUYER1 เป็นรีเลชั่นที่เข้ากันได้

ผลลัพธ์จากการ Intersection ของ BUYER กับ BUYER1 คือ รีเลชั่น RESULT ซึ่งมีค่าดังต่อไปนี้

BUYER ∩ BUYER1 = RESULT (NAME, ITEM)

SMITH	A
ADAMS	A

2.2.2 Union

นิยาม ให้ R และ S เป็น n-ary รีเลชั่นที่ เข้ากันได้, r และ s เป็น tuple ของ R หรือ S ตามลำดับ, union (แทนด้วยสัญลักษณ์ ∪) ของ R กับ S สามารถกำหนดได้ ดังนี้

$$R \cup S = \{t : t \in R \vee t \in S\}$$

ผลลัพธ์จากการ union ระหว่างรีเลชั่น R และ S จะได้รีเลชั่นใหม่ซึ่งมีดีกรีเท่ากับดีกรีของ R หรือ S พร้อมทั้ง attribute ชุดเดียวกันกับของ R และ S ด้วย

ตัวอย่าง

จากรูปที่ 2-2 ผลลัพธ์จากการ Union ของ BUYER กับ BUYER1 คือ รีเลชั่น RESULT ซึ่งมีค่าดังต่อไปนี้

BUYER ∪ BUYER1 = RESULT (NAME, ITEM)

SMITH	A
JONES	B
ADAMS	A
SMITH	B
JONES	A
SMITH	C
JUDY	C
ROBERT	A
JACK	B

2.2.3 Difference

นิยาม ให้ R และ S เป็น n-ary รีเลชั่นที่ เข้ากันได้(compatible), r และ s เป็น tuple ของ R และ S ตามลำดับ, difference (แทนด้วยสัญลักษณ์ -) ของ R กับ S สามารถกำหนดได้ ดังนี้

$$R - S = \{t : t \in R \wedge t \notin S\}$$

ผลลัพธ์จากการ difference ระหว่างรีเลชั่น R และ S จะได้รีเลชั่นใหม่ซึ่งมีดีกรีเท่ากับดีกรีของ R หรือ S พร้อมทั้ง attribute ชุดเดียวกันกับของ R และ S ด้วย

ตัวอย่าง

จากรูปที่ 2-2 ผลจากการ Difference ของ BUYER กับ BUYER1 คือ รีเลชั่น RESULT ซึ่งมีค่าดังต่อไปนี้

BUYER - BUYER1 = RESULT (NAME, ITEM)

JONES	B
SMITH	B
JONES	A
SMITH	C

2.2.4 Cross Product

นิยาม ให้ R และ S เป็น m-ary รีเลชั่นและ n-ary รีเลชั่น ตามลำดับ, r เป็น tuple ของ R, s เป็น tuple ของ S, Cross product (แทนด้วยสัญลักษณ์ ⊗) ของ R กับ S สามารถกำหนดได้ ดังนี้

$$R \otimes S = \{r^\frown s : r \in R \wedge s \in S\}$$

ผลลัพธ์จากการ cross product ระหว่างรีเลชั่น R และ S จะได้รีเลชั่นใหม่ซึ่งมีดีกรีเท่ากับดีกรีของ R และ S รวมกัน

ตัวอย่าง

จากรูปที่ 2-2 ผลลัพธ์จาก Cross Product ของ BUYER กับ PRODUCT คือ รีเลชั่น RESULT ซึ่งมีค่าดังต่อไปนี้

BUYER ⊗ PRODUCT = RESULT (NAME, ITEM, CODE, COST, PRICE)

SMITH	A	A	5	8
JONES	B	A	5	8
ADAMS	A	A	5	8
SMITH	B	A	5	8
JONES	A	A	5	8
SMITH	C	A	5	8
SMITH	A	B	4	4
JONES	B	B	4	4
ADAMS	A	B	4	4
SMITH	B	B	4	4
JONES	A	B	4	4
SMITH	C	B	4	4
SMITH	A	C	6	9
JONES	B	C	6	9
ADAMS	A	C	6	9
SMITH	B	C	6	9
JONES	A	C	6	9
SMITH	C	C	6	9

2.2.5 Selection

นิยาม ให้ R เป็น n-ary รีเลชั่น, $D = \{D_1,\ldots,D_n\}$ เป็นโดเมนเซทของ R,

$A \in \{D_1,\ldots,D_n\}$, v เป็นค่าคงที่, $\theta \in \{<,\leq,>,\geq,=,\neq\}$

<u>selection ของ R</u> สามารถกำหนดได้ ดังนี้

$$R(A\ \theta\ v) = \{r : r \in R \wedge (A\ \theta\ v)\}$$

ผลลัพธ์จากการทำ selection กับรีเลชั่น R จะได้รีเลชั่นใหม่ซึ่งมี tuple เป็นส่วนหนึ่งของรีเลชั่น R

<u>ตัวอย่าง</u>

จากรูปที่ 2-2 ผลลัพธ์จาก Selection ของรีเลชั่น PRODUCT ซึ่งมีค่า PRICE < 8 คือ รีเลชั่น S1 ซึ่งมีค่าดังต่อไปนี้

PRODUCT (PRICE < 8) = S1(CODE, COST, PRICE)

A	5	8
B	4	4

2.2.6 <u>Projection</u>

<u>นิยาม</u> ให้ R เป็น n-ary รีเลชั่น, $\{D_1,\ldots,D_n\}$ เป็นโดเมนเซทของ R, A เป็นสับเซทของ $\{D_1,\ldots,D_n\}$, <u>projection ของ R ภายใต้เซท A</u> สามารถกำหนดได้ดังนี้

$$R(A) = \{\, r[A] : r \in R \,\}$$

ผลลัพธ์จากการทำ projection กับรีเลชั่น R จะได้รีเลชันซึ่งประกอบด้วยค่าทั้งหมดของแต่ละสดมภ์ที่ต้องการ (ค่าซ้ำกันอาจแสดงเพียงครั้งเดียว)

ตัวอย่าง

จากรูปที่ 2-2 ผลลัพธ์จาก Projection ของรีเลชั่น BUYER ภายใต้เซทของ attribute (NAME) คือ รีเลชั่น P ซึ่งมีค่าดังต่อไปนี้

BUYER (NAME) = P (NAME)

SMITH

JONES

ADAMS

2.2.7 Join

นิยาม ให้ R เป็น m-ary รีเลชั่นที่มี $\{D_1,\ldots,D_m\}$ เป็นโดเมนเซท และ r เป็น tuple ใน R, S เป็น n-ary รีเลชั่นที่มี $\{E_1,\ldots,E_n\}$ เป็นโดเมนเซท และ s เป็น tuple ใน S, $A \in \{D_1,\ldots,D_m\}$, $B \in \{E_1,\ldots,E_n\}$ ซึ่งA และ B เป็นโดเมนที่มีค่าประเภทเดียวกัน ดังนั้น Join ระหว่าง R กับ S ภายใต้เงื่อนไข $A\ \theta\ B$ เมื่อ $\theta \in \{<,\leq,>,\geq,=,\neq\}$ คือ

$$R(A\ \theta\ B)S = \{(r^\frown s): r \in R \wedge s \in S \wedge (r[A]\ \theta\ s[B])\}$$

ตัวอย่าง

จากรูปที่ 2-2 ผลลัพธ์จาก Join ระหว่าง BUYER และ PRODUCT

ภายใต้เงื่อนไข ITEM = COST คือ รีเลชั่น J ซึ่งมีค่าดังต่อไปนี้

BUYER (ITEM=CODE) PRODUCT = J (NAME ITEM CODE COST PRICE)

SMITH	A	A	5	8
JONES	B	B	4	4
ADAMS	A	A	5	8
SMITH	B	B	4	4
JONES	A	A	5	8
SMITH	C	C	6	9

2.2.8 Divide

นิยาม ให้ R เป็น m-ary รีเลชั่นที่มี $A=\{D_1,\ldots,D_m\}$ เป็นโดเมนเซท และ r เป็น tuple ของ R และ S เป็น k-ary รีเลชั่นที่มี $B \subseteq A$ เป็นโดเมนเซท และ s เป็น tuple ของ S และ $\bar{A} = A - B$

$$R(A) \div S(B) = \{ r[\bar{A}] : g_R(r[\bar{A}])=S \}$$

ตัวอย่าง

จากรูปที่ 2-2 ผลลัพธ์จากการ Divide รีเลชั่น BUYER ด้วยรีเลชั่น PRODUCT1 คือรีเลชั่น D ซึ่งมีค่าดังต่อไปนี้

BUYER + PRODUCT1 = D (NAME)

SMITH

JONES

สำหรับโปรแกรมปฏิบัติการ di vide ซึ่งพัฒนาในวิทยานิพนธ์นี้ จะจำกัดให้ R เป็น binary relation และ S เป็น unary relation ซึ่งมี attribute ที่ใช้โดเมนเซทเดียวกันกับ attribute หนึ่งใน R

บทที่ 3

การวิเคราะห์ ออกแบบและการพัฒนาโปรแกรม

3.1 ขอบเขตในการทำวิทยานิพนธ์

การที่จะทำการพัฒนาโปรแกรมสำหรับระบบจัดการฐานข้อมูลที่สมบูรณ์นั้น เป็นเรื่องที่ยาก เพราะจะต้องมีการวิเคราะห์ถึงผลที่จะได้จากการทำงาน ในแต่ละขั้นตอนอย่างละเอียดเพื่อให้การทำงานเป็นไปอย่างมีประสิทธิภาพมากที่สุด เช่นการนำเสนอข้อมูลจะต้องใช้เวลาในการทำงานให้น้อยที่สุด ซึ่งอาจจะต้องมีการนำเอาทฤษฎีหลายๆทฤษฎีมาช่วยในการทำงาน หรือแม้แต่การเก็บข้อมูลลงในฐานข้อมูลก็จะต้องมีวิธีที่มีประสิทธิภาพ ไม่มีการใช้เนื้อที่ในหน่วยความจำโดยเปล่าประโยชน์ เป็นต้น การทำวิจัยในครั้งนี้เป็นเพียงการวางแผนและพัฒนาโปรแกรมสำหรับคำสั่งพื้นฐานในการปฏิบัติการกับฐานข้อมูล ซึ่งพอจะสรุปขอบเขตได้ ดังต่อไปนี้

1. ทำการศึกษา และ ออกแบบปทานุกรมข้อมูล (data dictionary) เพื่อใช้ในการดำเนินงาน รวมทั้งออกแบบโครงสร้างข้อมูล (data structure) เพื่อเก็บรายละเอียดของข้อมูลภายในฐานข้อมูลขณะดำเนินงาน

2. พัฒนาโปรแกรมที่สามารถดำเนินงานต่างๆ กับข้อมูล ในฐานข้อมูลแบบรีเลชั่นได้ ด้วยคำสั่งปฏิบัติการแบบพีชคณิตสัมพันธ์ซึ่งมีความหมายการดำเนินงานตามนิยามที่ให้ในบทที่ 2 ดังต่อไปนี้

- INTERSECT
- UNION
- DIFFERENCE
- CROSS PRODUCT
- SELECT
- PROJECT
- JOIN
- DIVIDE

3. ผลลัพท์ที่ได้จากการดำเนินงาน จะต้องสามารถแสดงผลทางจอภาพ และทางเครื่องพิมพ์ (ไม่จำกัดประเภท) ที่ใช้อยู่กับระบบคอมพิวเตอร์ VAX-11/785

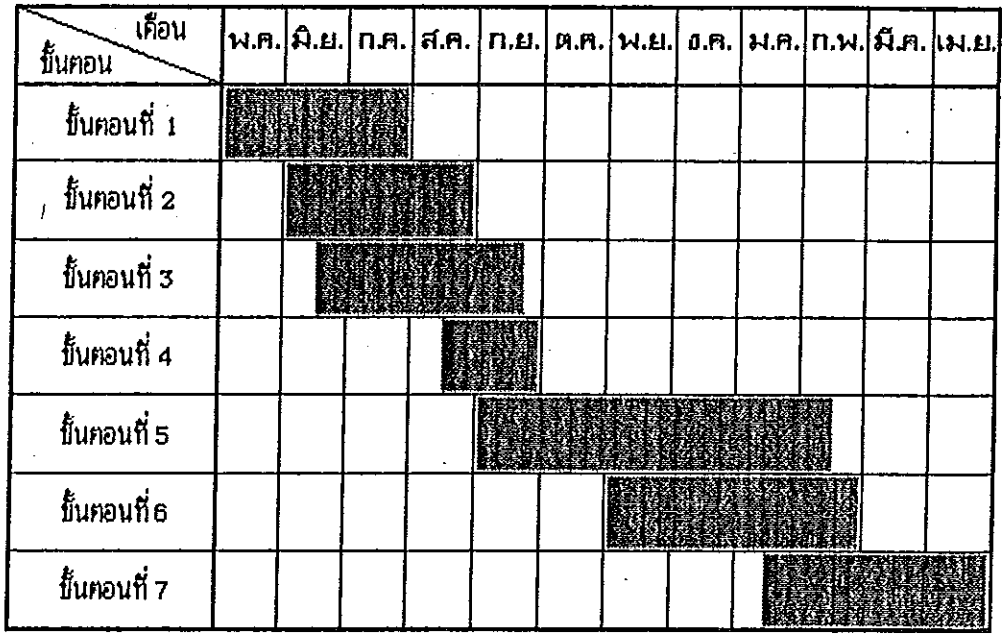
3.2 ขั้นตอนในการดำเนินงานและรายละเอียดอื่นๆ

3.2.1 ขั้นตอนการดำเนินการวิจัย

การดำเนินการวิจัยสามารถแบ่งออกได้เป็น 7 ขั้นตอนใหญ่ๆ ดังนี้คือ

1. ศึกษา ทำความเข้าใจกับนิยามและความหมาย รวมทั้งขั้นตอนการทำงานของคำสั่งปฏิบัติการฐานข้อมูลแบบพีชคณิตสัมพันธ์ทั้งหมด
2. ออกแบบระบบ (system design) โดยพิจารณาปทานุกรมข้อมูล สำหรับเก็บข้อมูลเกี่ยวกับการ กำหนดโครงสร้างของฐานข้อมูลและรายละเอียดของข้อมูลเป็นอันดับแรกตามด้วย การออกแบบโครงสร้างข้อมูล เพื่อเก็บรายละเอียดเกี่ยวกับโครงสร้างของฐานข้อมูลที่ใช้ขณะดำเนินงานและออกแบบโครงสร้างของโปรแกรมที่จะทำการพัฒนา
3. กำหนดและออกแบบรูปแบบของการแสดงผลจากการทำงานต่างๆ บนจอภาพ
4. เลือกภาษาที่เหมาะสมในการพัฒนาโปรแกรม
5. พัฒนาโปรแกรมให้ทำงานได้ตามระบบที่ได้ออกแบบไว้
6. ทดสอบโปรแกรมและเพิ่มเติมรายละเอียดปลีกย่อยต่างๆ เพื่อให้ได้โปรแกรมที่สมบูรณ์
7. จัดทำเอกสาร (documentation) และพิมพ์รายงานวิทยานิพนธ์

ช่วงเวลาในการดำเนินงานทั้งหมดแสดงโดย Gantt chart ในรูปที่ 3-1



รูปที่ 3-1 Gantt chart ของขั้นตอนในการวิจัย

3.2.2 เครื่องมือและอุปกรณ์ที่ใช้ในการทำวิจัย

เครื่องคอมพิวเตอร์ที่ใช้ในการพัฒนาโปรแกรมเป็นเครื่อง VAX-11/785 ซึ่งเป็น Superminicomputer ที่มีหน่วยความจำหลัก 8 MB ใช้ระบบดำเนินงาน ULTRIX-32 และมีภาษาคอมพิวเตอร์ระดับสูงให้ใช้งานได้ดังนี้ คือ C FORTRAN77 RM/COBOL Pascal LISP และ Modula2

3.2.3 สถานที่ทำการวิจัย

ศูนย์คอมพิวเตอร์ มหาวิทยาลัยสงขลานครินทร์

3.2.4 ระยะเวลาทำการวิจัย

1 ปีการศึกษา (ประมาณ 12 เดือน)

3.2.5 ภาษาที่ใช้ในการทำการวิจัย

ภาษาที่ใช้ คือ ภาษา C (C language) สาเหตุที่เลือกใช้ภาษา C ในการทำวิจัยนั้น เนื่องจาก

1. ลักษณะของภาษา C เป็น "ภาษาโครงสร้าง (structure language)" ซึ่งนั่นจะหมายถึง การพัฒนาโปรแกรมจะเป็นไปอย่างมีประสิทธิภาพ เช่น เราสามารถพัฒนาโปรแกรมที่เป็น โมดูล (module) ได้ หรือ การตรวจสอบขั้นตอนการทำงานก็สามารถทำได้สะดวก เป็นต้น

2. ลักษณะของรูปแบบการใช้ภาษามีความยืดหยุ่นสูงมาก เมื่อเปรียบเทียบกับภาษาโครงสร้างอื่นๆ เช่น Pascal หรือ Modula2

3. เนื่องจากระบบดำเนินงาน(operating system) ของเครื่องคอมพิวเตอร์ที่ใช้เป็น ULTRIX-32 ซึ่งพัฒนาขึ้นโดยใช้ภาษา C ดังนั้นการเลือกใช้ภาษา C พัฒนาโปรแกรมทำให้สามารถที่จะเข้าถึงระบบการทำงานของเครื่องได้รวดเร็วกว่าภาษาอื่น

4. มีโปรแกรมที่ช่วยในการค้นหาจุดที่เกิดการทำงานผิดพลาดได้ โปรแกรมเหล่านั้นได้แก่ adb debugger และ dbx debugger

5. ในการพัฒนาโปรแกรมได้มีการนำ โปรแกรมสำเร็จรูป(program package) ที่ชื่อ "ncurses" ซึ่งเป็นโปรแกรมสำเร็จรูปในการจัดการทำงานของจอภาพในลักษณะที่เป็น window โปรแกรมนี้ถูกพัฒนาขึ้นโดยใช้ภาษา C ดังนั้นโปรแกรมสำหรับงานวิจัยจึงสามารถที่จะเรียก ฟังก์ชัน(function) ต่างๆ มาใช้งานได้ทันทีโดยไม่ต้องผ่านโปรแกรมเชื่อมโยง(linkage program) ซึ่งทำให้ลดความยุ่งยากในการพัฒนาโปรแกรม

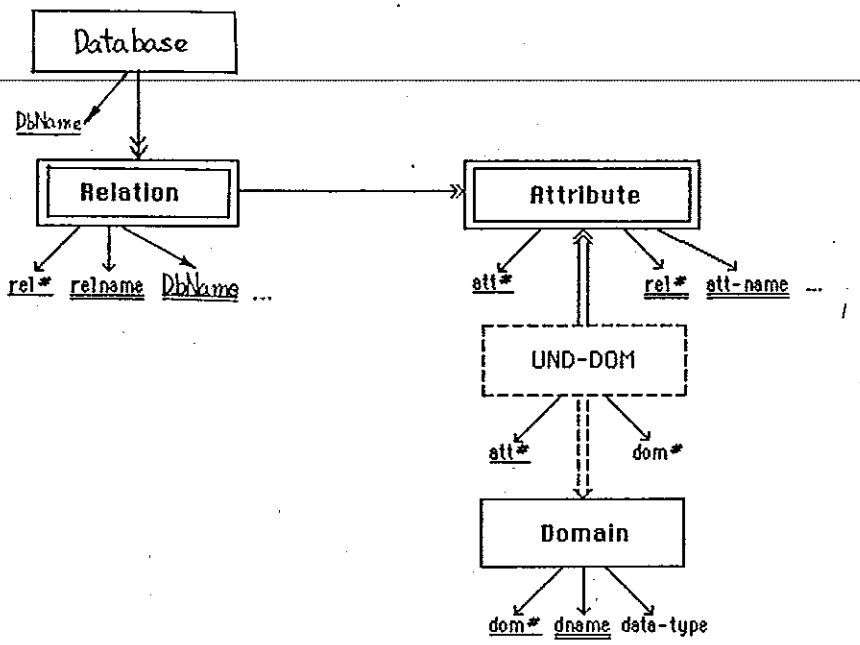
3.3 การออกแบบระบบ

3.3.1 ปทานุกรมข้อมูล

ในการทำวิทยานิพนธ์นี้ โปรแกรมที่พัฒนาเป็นการปฏิบัติการกับข้อมูลในฐานข้อมูลโดยสมมุติว่าได้มีการเก็บข้อมูลในฐานข้อมูลเรียบร้อยแล้ว และในการปฏิบัติการ จะต้องมีข้อมูลเกี่ยวกับรายละเอียดทั้งหลายของโครงสร้างฐานข้อมูลและชนิดของข้อมูลที่เก็บอยู่แล้ว ซึ่งรายละเอียดที่ต้องการเหล่านี้จะถูกเก็บรวบรวมในส่วนที่เรียกกันว่า ปทานุกรมข้อมูล แต่เนื่องจากในขณะที่ทำวิทยานิพนธ์นี้ ยังไม่มีส่วนที่เป็นปทานุกรมข้อมูล และยังไม่มีข้อมูลในฐานข้อมูลให้ใช้ ผู้ทำวิทยานิพนธ์นี้จึงได้ออกแบบปทานุกรมข้อมูลที่จะใช้เอง และทำการสร้างข้อมูลในฐานข้อมูลเพื่อทดลองใช้กับโปรแกรมที่พัฒนาเอง

ในการออกแบบ ปทานุกรมข้อมูลนั้นถือว่าเป็นจุดเริ่มต้นของการทำงานทั้งหมด ซึ่งปทานุกรมข้อมูล ที่ได้ออกแบบนี้จะใช้สำหรับเก็บรายละเอียดโครงสร้างและ ความสัมพันธ์

ของข้อมูล โดยใช้ O-R diagram (Object-Relation ship diagram) เป็นอุปกรณ์ช่วย ดังแสดงใน รูปที่ 3-2



รูปที่ 3-2 ปทานุกรมข้อมูลของระบบ

จากรูป 3-2 จะประกอบไปด้วย สิ่งที่สนใจ (object type) ดังนี้คือ

Relation เป็น object type ซึ่งแทนรายละเอียดของแต่ละรีเลชั่นในฐานข้อมูล ประกอบไปด้วยรายละเอียด คือ รหัสรีเลชั่น ชื่อรีเลชั่น จำนวน attribute จำนวนtuple ขนาดของtuple วันที่ที่ทำการสร้าง และชนิดของรีเลชั่น

Domain เป็น object type ซึ่งแทนรายละเอียดของชนิดข้อมูลซึ่งในแต่ละ attribute ของแต่ละรีเลชั่น ประกอบด้วยรายละเอียด คือ รหัสของโดเมน ชื่อโดเมน ชนิด และขนาด

Attribute เป็น object type ซึ่งแทนรายละเอียดของ attribute ต่างๆ ภายในรีเลชั่น แต่ละรีเลชั่นประกอบด้วย รหัส attribute รหัสรีเลชั่น ชื่อattribute ตำแหน่งที่เริ่มต้นในtuple ขนาดของattribute และชนิดของ attribute

UND-DOM เป็น object type ซึ่งแทนรายละเอียดของความสัมพันธ์ระหว่าง object type Attribute กับ Domain ในลักษณะที่ว่า ค่าที่เป็นไปได้ของ attribute หนึ่งๆ ต้องเป็นค่าในโดเมนใดโดเมนหนึ่ง ประกอบด้วยรายละเอียด คือ รหัส attriubute และ รหัสโดเมน

จาก object type ที่กล่าวมาทั้งหมดสามารถสร้างรีเลชั่นที่สมนัยกับแผนภาพ ได้ดังนี้

RELATION(rel#,relname, ...)

ATTRIBUTE(att#,rel#,attr-name, ...)

DOMAIN(dom#,dname,data-type, ...)

UND-DOM(att#,dom#)

3.3.2 โครงสร้างแฟ้มข้อมูล

จากปทานุกรมข้อมูล ที่ได้ทำการออกแบบ (ด้วย O-R diagram) ไว้แล้วนั้น ทำให้เราสามารถที่จะกำหนดถึง โครงสร้างของแฟ้มข้อมูล (file structure) ต่างๆ ที่สมนัยกันเพื่อใช้ในระบบได้ แฟ้มข้อมูลหลักที่จัดเก็บข้อมูลตามโครงสร้างปทานุกรมจะมีอยู่ด้วยกันทั้งหมด 3 แฟ้ม ดังนี้คือ

ก. <Database-Name>.rel

วัตถุประสงค์ : สำหรับเก็บรายละเอียดของรีเลชั่นต่างๆ ที่อยู่ในฐานข้อมูลชื่อ <Database-Name>

รายละเอียด :

ชื่อ field	ชนิด	รูปแบบ	วัตถุประสงค์ในการใช้งาน
Relation-ID	P-Key	Integer	รหัสของรีเลชั่น
Relation-Name	S-Key	Char(10)	ชื่อของรีเลชั่น
#Attribute	Non-Key	Integer	จำนวน attribute
#Tuple	Non-Key	Integer	จำนวน tuple
Tuple length	Non-Key	Integer	เก็บขนาดของ tuple
Create date	Non-Key	Char(8)	วันที่ ที่สร้าง
Status	Non-Key	Char(1)	ชนิดของรีเลชั่น - Base relation (B) - Keep relation (K) - Temp. relation (T)

หมายเหตุ : 1. ตัวอย่างของข้อมูลในแฟ้มข้อมูลนี้ ดูได้จาก ภาคผนวก ข.

2. P-Key คือ primary key

3. S-Key คือ secondary key

ข. <Database-Name>.att

วัตถุประสงค์ : เก็บรายละเอียดของ attribute ของทุกๆ รีเลชั่นที่อยู่ในฐานข้อมูลชื่อ <Database-Name>

รายละเอียด :

ชื่อ field	ชนิด	รูปแบบ	วัตถุประสงค์ในการใช้งาน
Attribute-ID	P-Key	Integer	รหัสของ attribute
Relation-ID	S-Key	Integer	รหัสของรีเลชั่น
Attribute-Name	S-Key	Char(15)	ชื่อของ attribute
Start position	Non-Key	Integer	Column ที่เริ่มต้น
End position	Non-Key	Integer	Column สุดท้าย
Domain-ID	Non-Key	Integer	รหัสของ domain
Status	Non-Key	Char(1)	ชนิดของ attribute - Prime-key (P) - Non-key (N)

หมายเหตุ : 1. ตัวอย่างของข้อมูลในแฟ้มข้อมูลนี้ ดูได้จาก ภาคผนวก ข.

2. P-Key คือ primary key

3. S-Key คือ secondary key

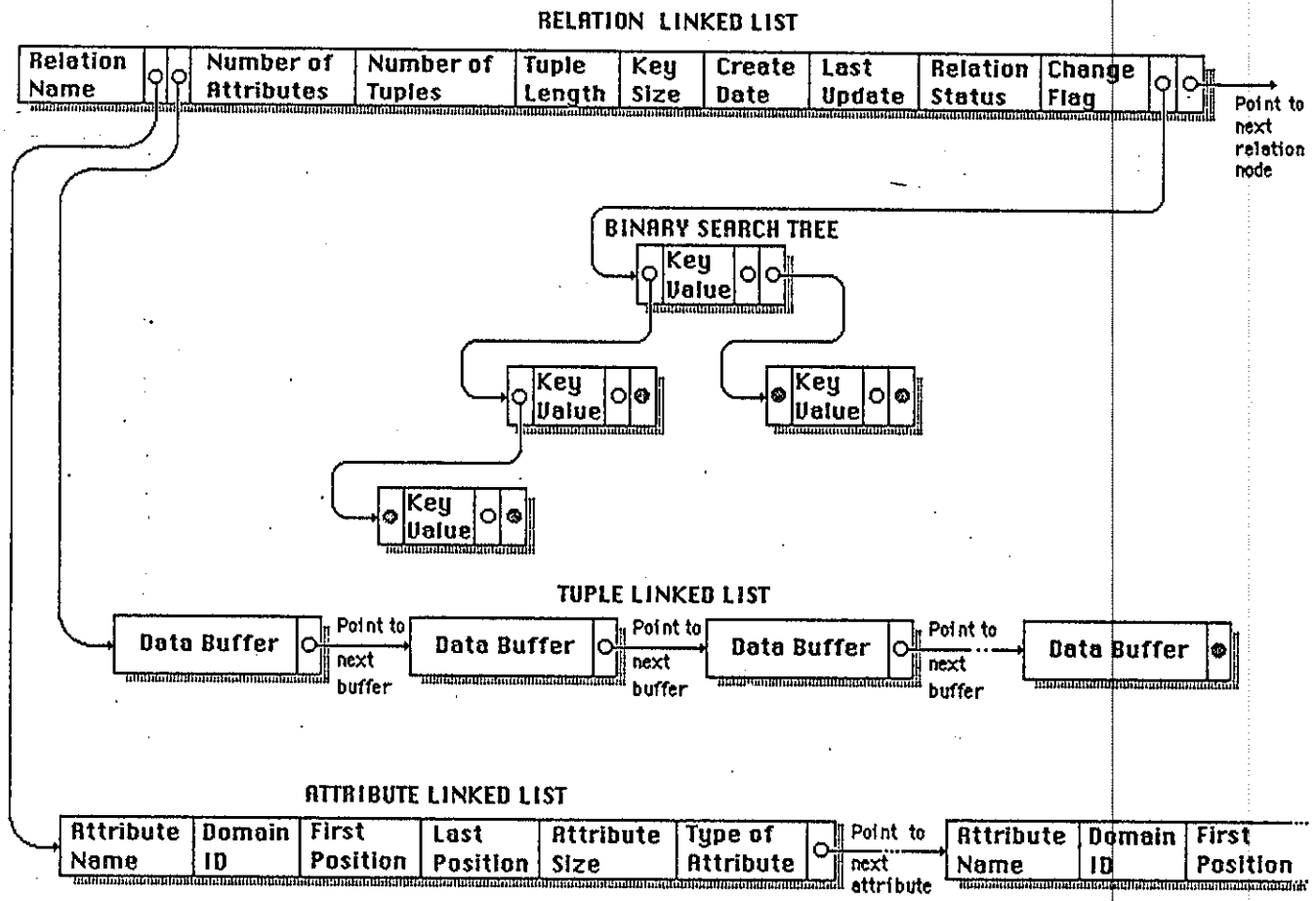
ค. <Relation-Name>.dat

วัตถุประสงค์ : เป็นแฟ้มข้อมูลเก็บข้อมูล (data) ของรีเลชั่นต่างๆ ในฐานข้อมูล

รายละเอียด : ไม่แน่นอนขึ้นอยู่กับโครงสร้างของรีเลชั่น ที่เก็บอยู่ในแฟ้มข้อมูลในข้อ ข.

หมายเหตุ : - ตัวอย่างของข้อมูลในแฟ้มข้อมูลประเภทนี้ ดูได้จาก ภาคผนวก ข.

- สามารถนำแฟ้มข้อมูลที่ใช้กับโปรแกรมภาษาโคบอลมาใช้ได้ทันที



รูปที่ 3-3 โครงสร้างข้อมูล (data structure)

นอกจากแฟ้มข้อมูลหลักทั้ง 3 แล้วยังมีแฟ้มข้อมูลทั่วไปอีก 2 แฟ้มข้อมูล คือ

README ใช้เก็บ Help Message ต่างๆ

SYSTEM ใช้เก็บชื่อของฐานข้อมูลทั้งหมดที่มีการจัดเก็บไว้ในระบบ

3.3.3 โครงสร้างข้อมูล

โครงสร้างข้อมูล (data structure) ที่ใช้เป็นโครงสร้างข้อมูลแบบ linked list ซึ่งประกอบไปด้วยสอง singly linked list หนึ่ง multi-linked list และหนึ่ง binary search tree ดังแสดงในรูปที่ 3-3 โดยมีรายละเอียด ดังต่อไปนี้

ก. Relation linked list

linked list นี้ใช้เก็บรายชื่อและรายละเอียดต่างๆ ของแต่ละรีเลชั่น โดยที่แต่ละ โหนด (node) ใน list จะได้มาจากการอ่านข้อมูลหนึ่ง record จากแฟ้มข้อมูล <Database-Name>.rel หรืออีกความหมายหนึ่งคือ หนึ่งโหนดแทนข้อมูลหนึ่งรีเลชั่นนั่นเอง

ข. Attribute linked list

linked list นี้ใช้เก็บรายละเอียดของ attribute ในแต่ละรีเลชั่น โดยที่แต่ละโหนดใน list จะได้มาจากการอ่านข้อมูลหนึ่ง record จากแฟ้มข้อมูล <Database-Name>.att ซึ่งหนึ่งโหนดจะแทนหนึ่ง attribute ของรีเลชั่น ดังนั้นในระบบจะมี attribute linked list ได้มากกว่าหนึ่ง list โดยที่แต่ละ list จะขึ้นกับหนึ่ง node ใน relation linked list เสมอ นั่นคือหนึ่งแต่ละโหนดใน relation linked list จะประกอบด้วยหนึ่ง attribute linked list เสมอ

ค. Tuple linked list

linked list นี้จัดเก็บข้อมูลจริงของแต่ละ tuple ของรีเลชั่นหนึ่งๆ โดยที่หนึ่งโหนดได้มาจากการอ่านแฟ้มข้อมูล <Relation-Name>.dat หนึ่ง record ขนาดของโหนดของแต่ละ linked list จะขึ้นอยู่กับค่าของ tuple length ของแต่ละรีเลชั่น (ในแฟ้มข้อมูล <Database-Name>.rel) ในระบบสามารถที่จะมี tuple linked list ได้มากกว่าหนึ่ง list โดยที่จำนวน tuple linked list นี้จะขึ้นกับจำนวนโหนด ใน relation linked list เสมอ นั่นคือหนึ่งโหนด relation linked list จะมีข้อมูลเก็บอยู่ในหนึ่ง tuple linked list เสมอ

ง. Binary search tree

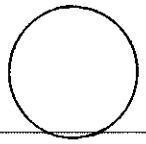
แต่ละรีเลชั่นนอกจากจะมี tuple linked list ซึ่งบรรจุข้อมูลในแต่ละ tuple ของรีเลชั่นแล้ว ยังอาจจะมีหนึ่ง binary search tree สำหรับใช้จัดเก็บค่าของดัชนี (index) โดยในแต่โหนดจะเก็บค่าของ key ของtuple และเลขที่(address)ที่ tuple ได้ถูกจัดเก็บไว้ใน tuple linked list รูปแบบของ binary search tree ที่ใช้นี้ เป็นแบบที่โหนดลูกทางซ้าย (left child node) จะต้องมีค่าน้อยกว่าหรือเท่ากับโหนดแม่ (parent node) และโหนดลูกทางขวา (right child node) จะต้องมีค่ามากกว่าโหนดแม่เสมอ

ตัวอย่างของโครงสร้างข้อมูลที่ได้กล่าวมาทั้งหมด สามารถที่จะดูตัวอย่างเพิ่มเติมได้จาก ภาคผนวก ข.

3.3.4 แผนภาพการไหลของข้อมูล

- แผนภาพการไหลของข้อมูล (Data Flow Diagram - DFD) ใช้สำหรับอธิบายการไหลของข้อมูลต่างๆ ที่เกิดขึ้นในระบบเพื่อทำให้ทราบว่าขบวนการ (process)

ต่างๆ ภายในระบบนั้นจะทำการรับและส่งข้อมูลอะไร สัญลักษณ์ที่ใช้ใน DFD จะมีอยู่ด้วยกัน 6 แบบ ดังนี้คือ



แทน ขบวนการ (process) ดำเนินงานต่างๆ

แทน แฟ้มข้อมูล ที่ใช้ในการดำเนินงาน

แทน ส่วนการทำงานที่ไม่รวมอยู่ในการออกแบบ (external agent)

แทน การไหลของข้อมูล (data flow)

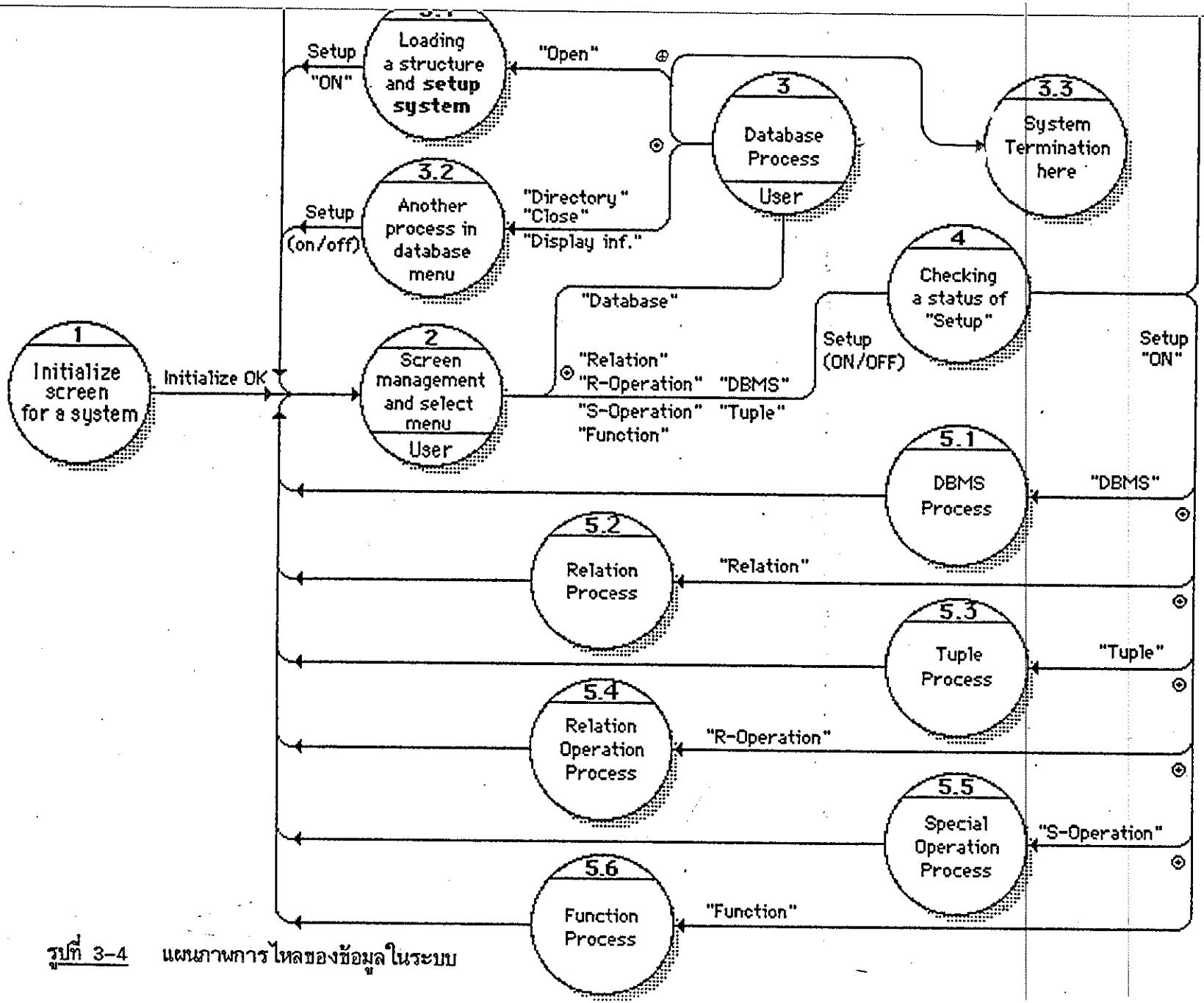
แทน การไหลของข้อมูลในทางใดทางหนึ่ง

แทน การไหลของข้อมูลทั้ง 2 ทาง (หรือมากกว่า)

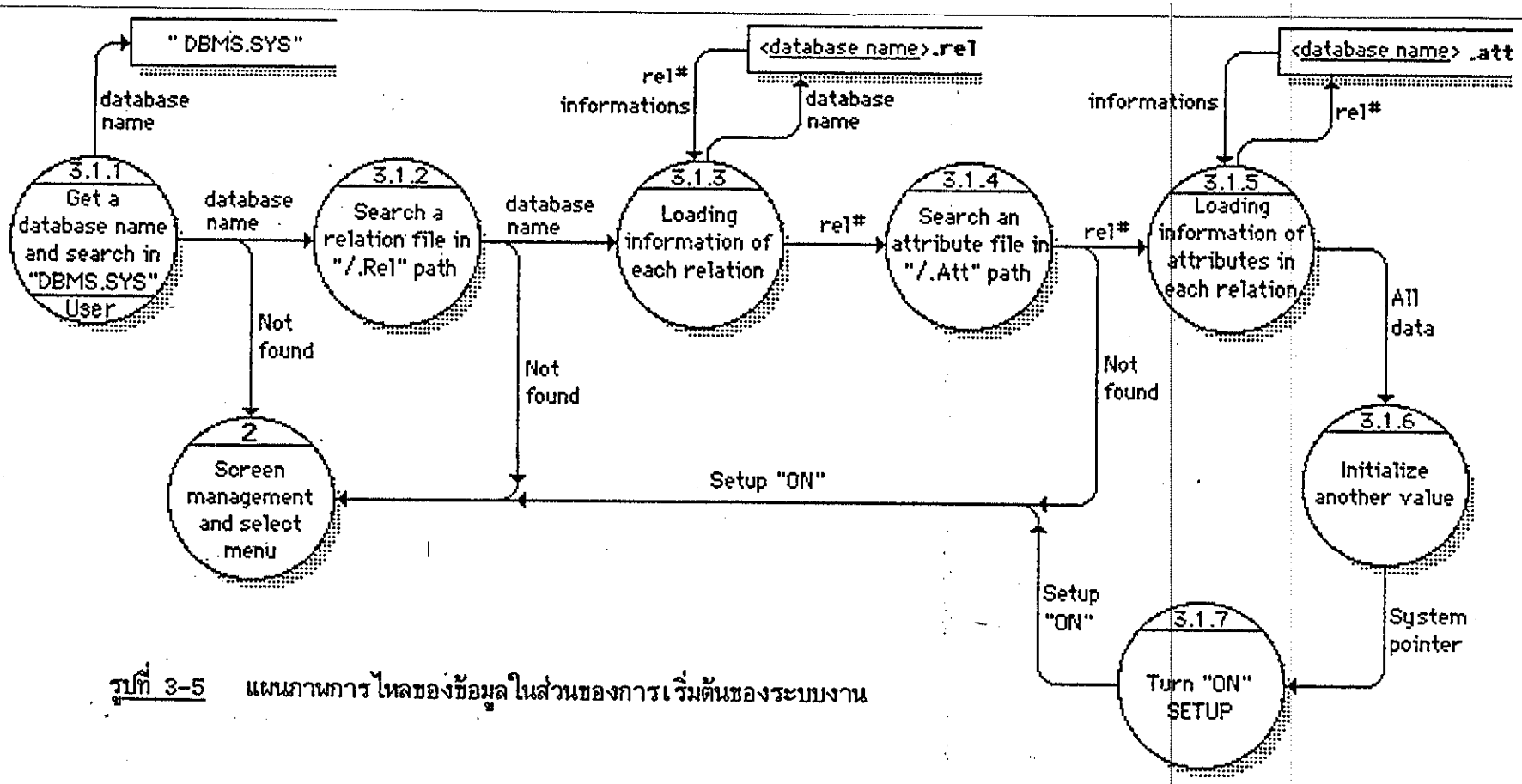
สำหรับ DFD ของระบบ แสดงดังรูป 3-4 ถึง 3-11

จากรูปที่ 3-4 เป็นแผนภาพการไหลของข้อมูลในส่วนเริ่มต้นของการทำงาน จะประกอบไปด้วยขบวนการต่างๆ 13 ขบวนการดังต่อไปนี้

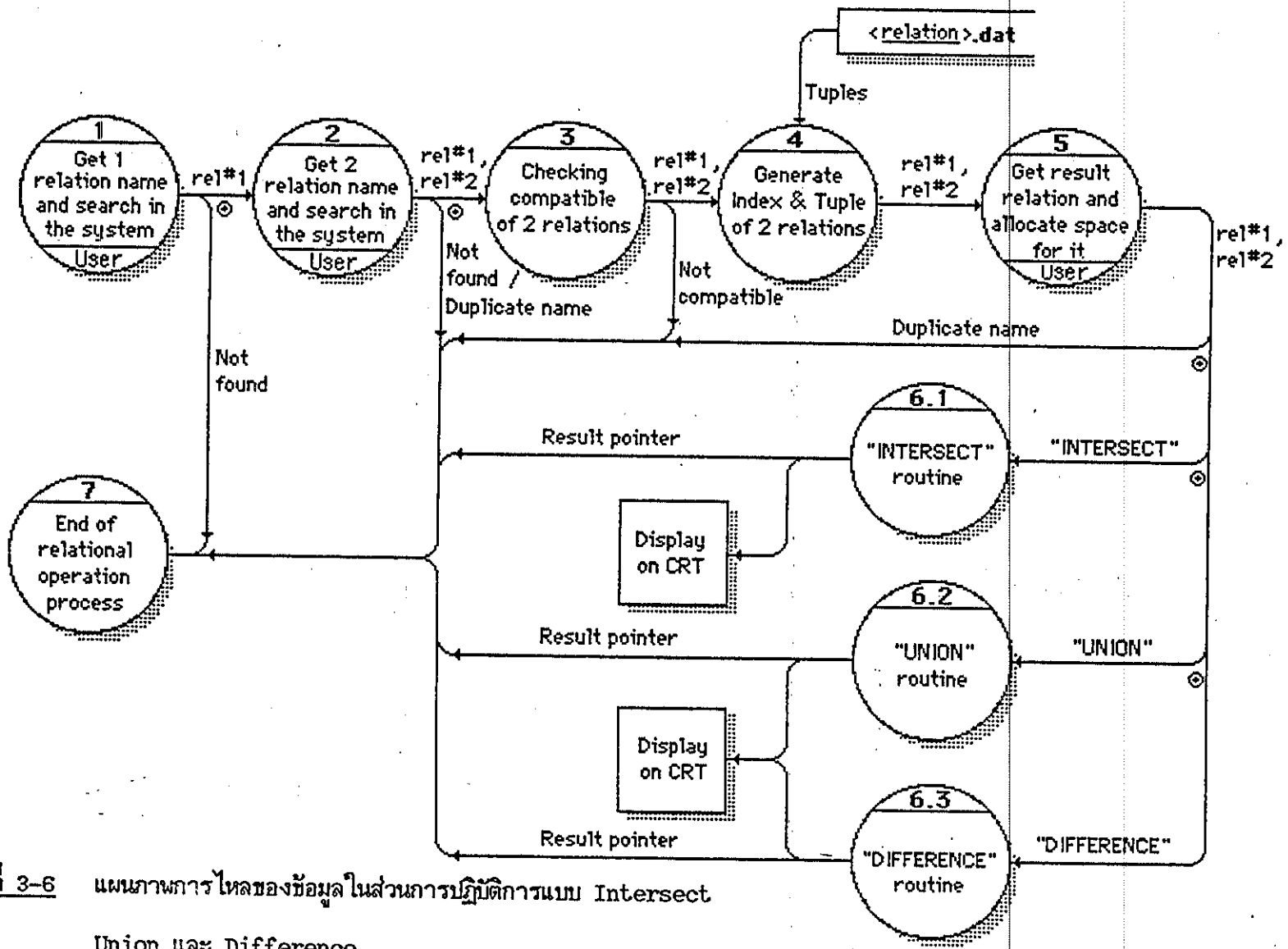
<u>ขบวนการที่ 1</u> เป็นการรับข้อมูลจากผู้ใช้ เพื่อนำไปใช้ในการปรับสภาพของจอภาพให้เหมาะสมกับการแสดงผลต่างๆ โดยสามารถที่จะเลือกการแสดงผลการทำงานได้กับจอภาพ 2



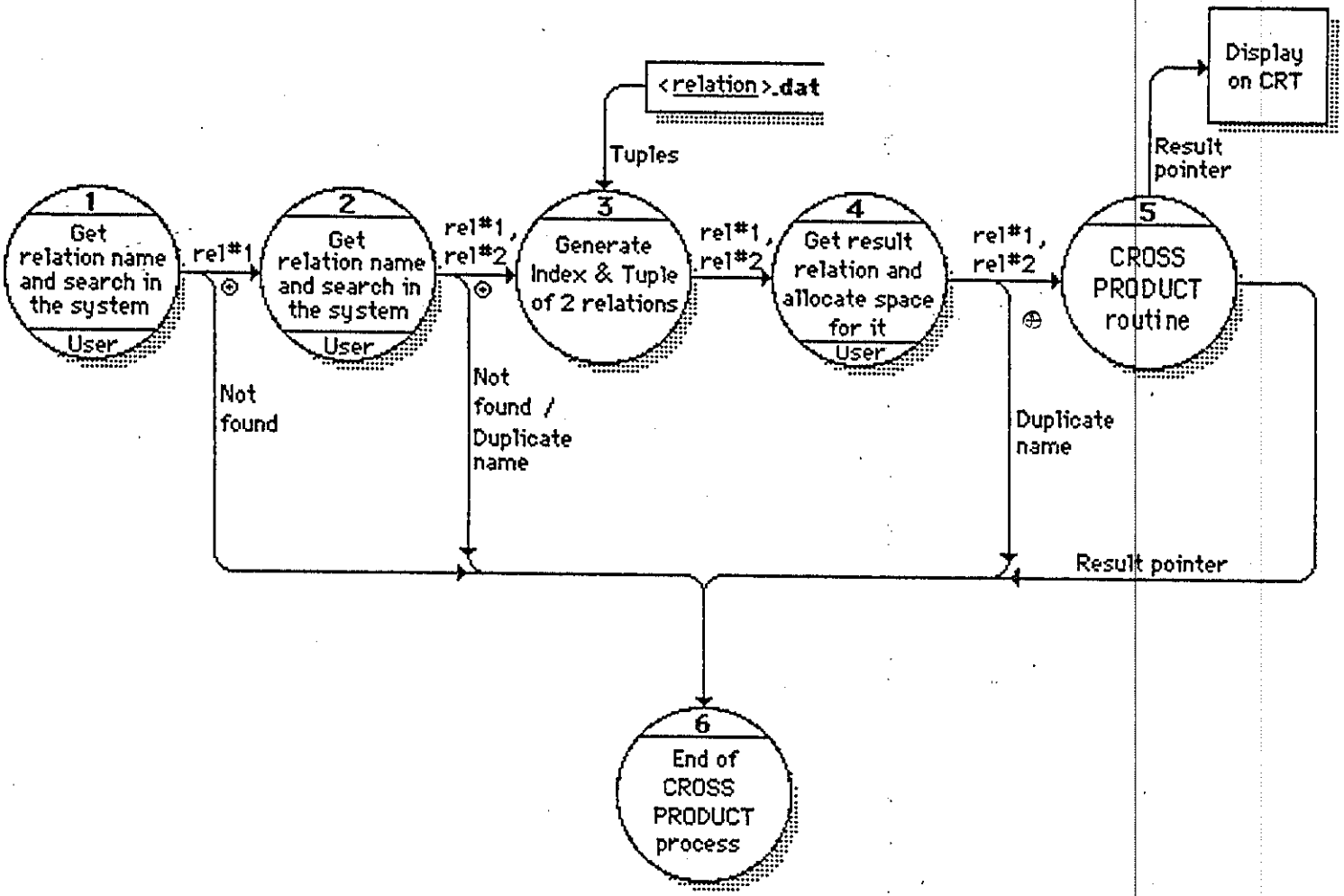
รูปที่ 3-4 แผนภาพการไหลของข้อมูลในระบบ



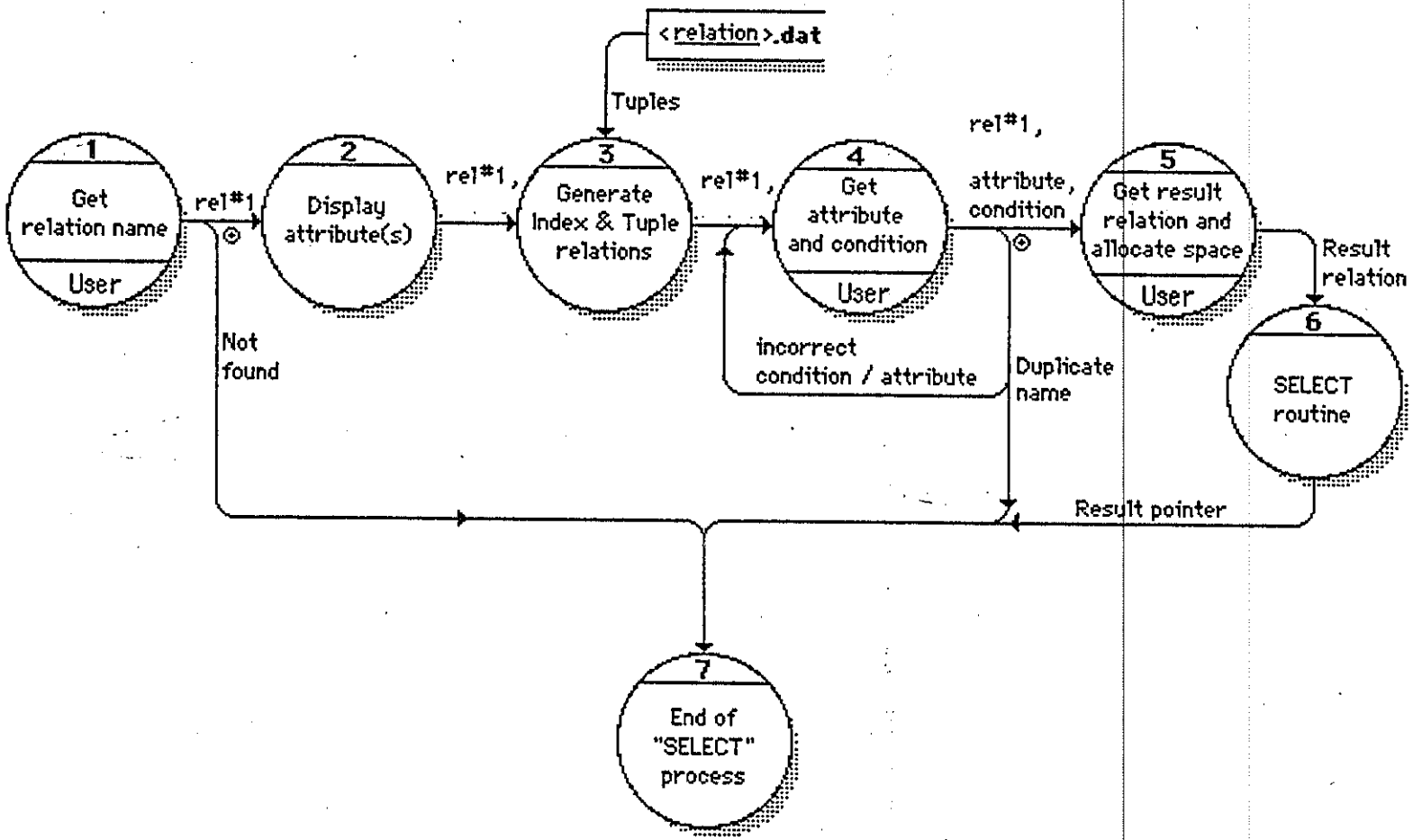
รูปที่ 3-5 แผนภาพการไหลของข้อมูลในส่วนของการเริ่มต้นของระบบงาน



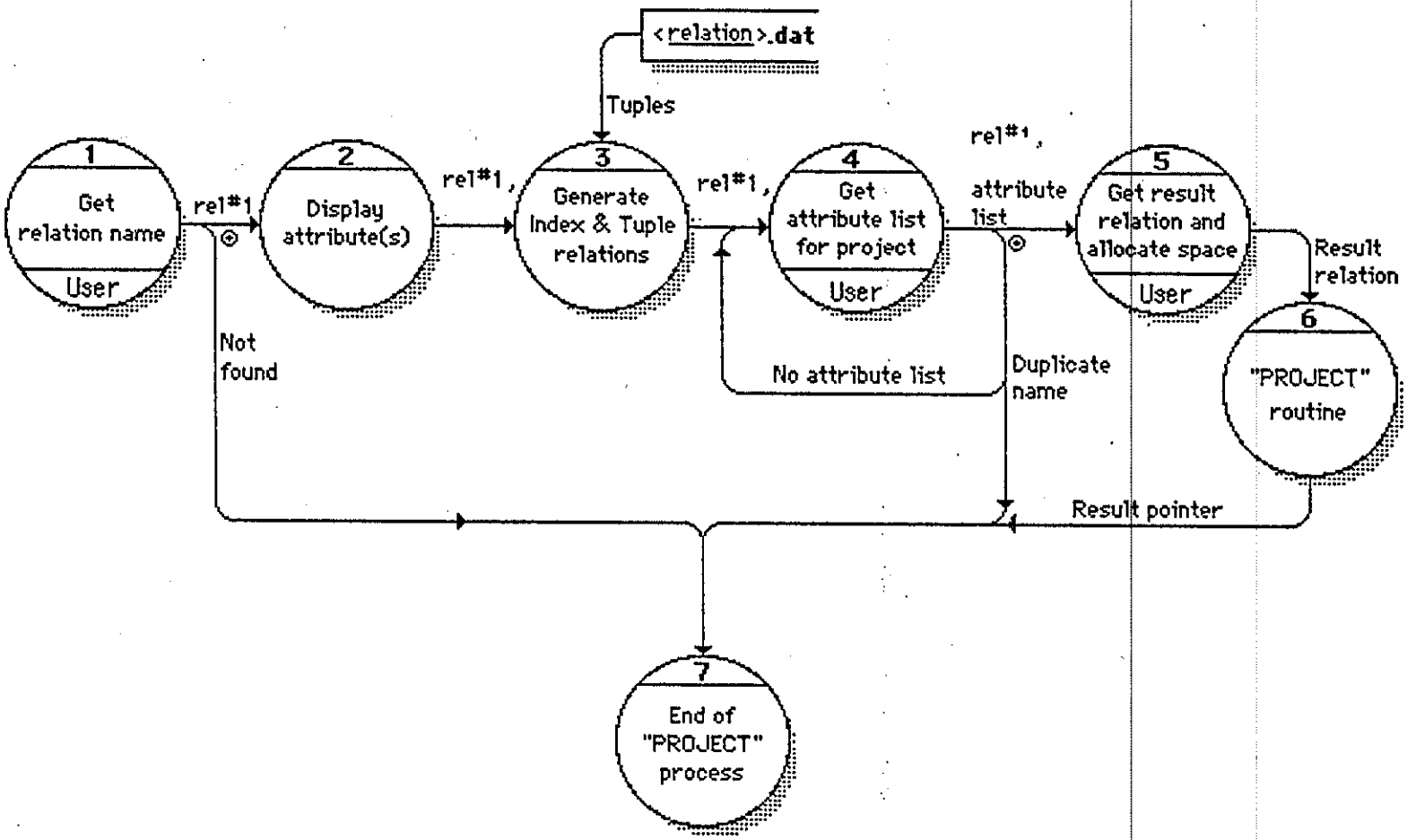
รูปที่ 3-6 แผนภาพการไหลของข้อมูลในส่วนการปฏิบัติการแบบ Intersect Union และ Difference



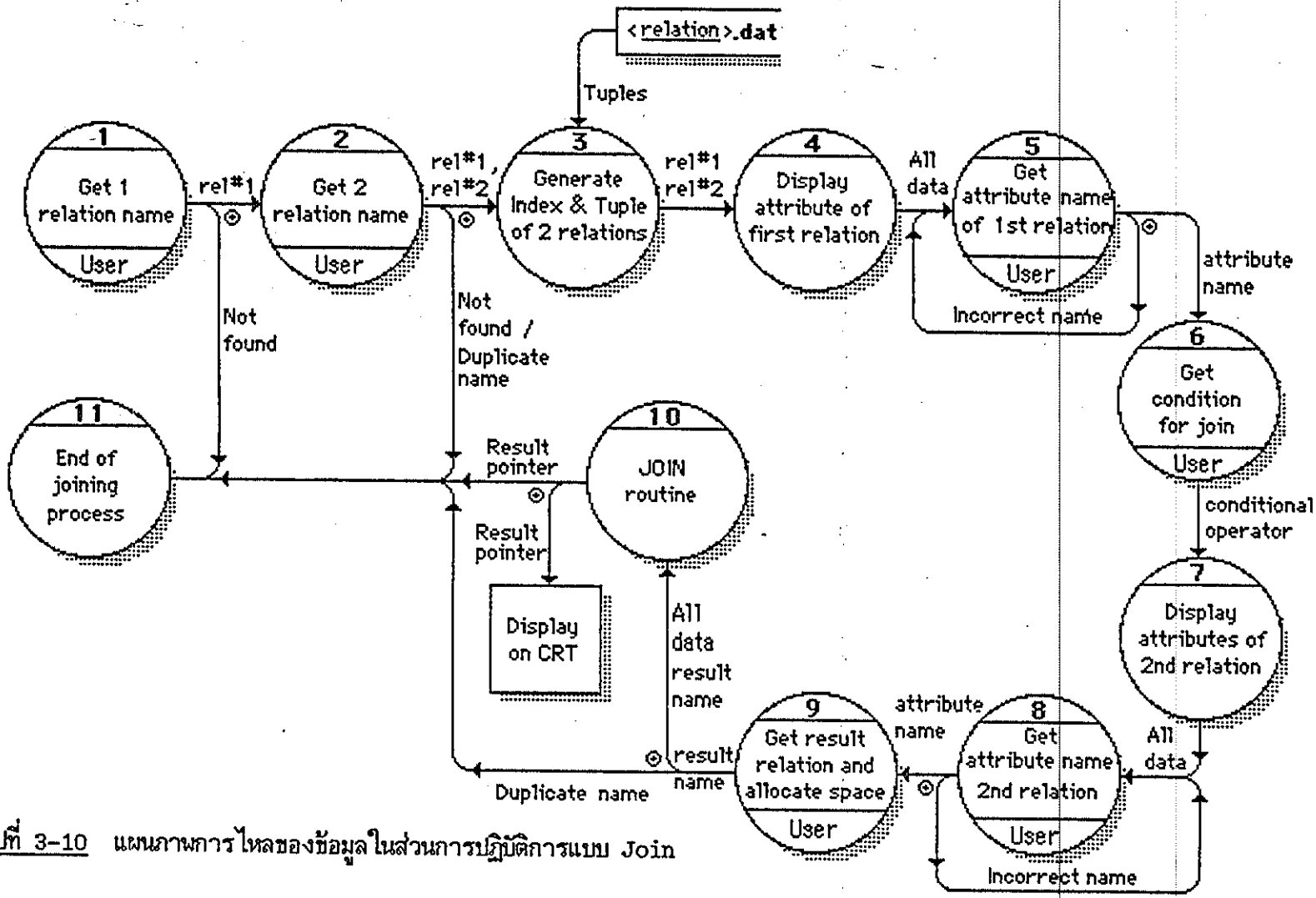
รูปที่ 3-7 แผนภาพการไหลของข้อมูลในส่วนการปฏิบัติการแบบ Cross product



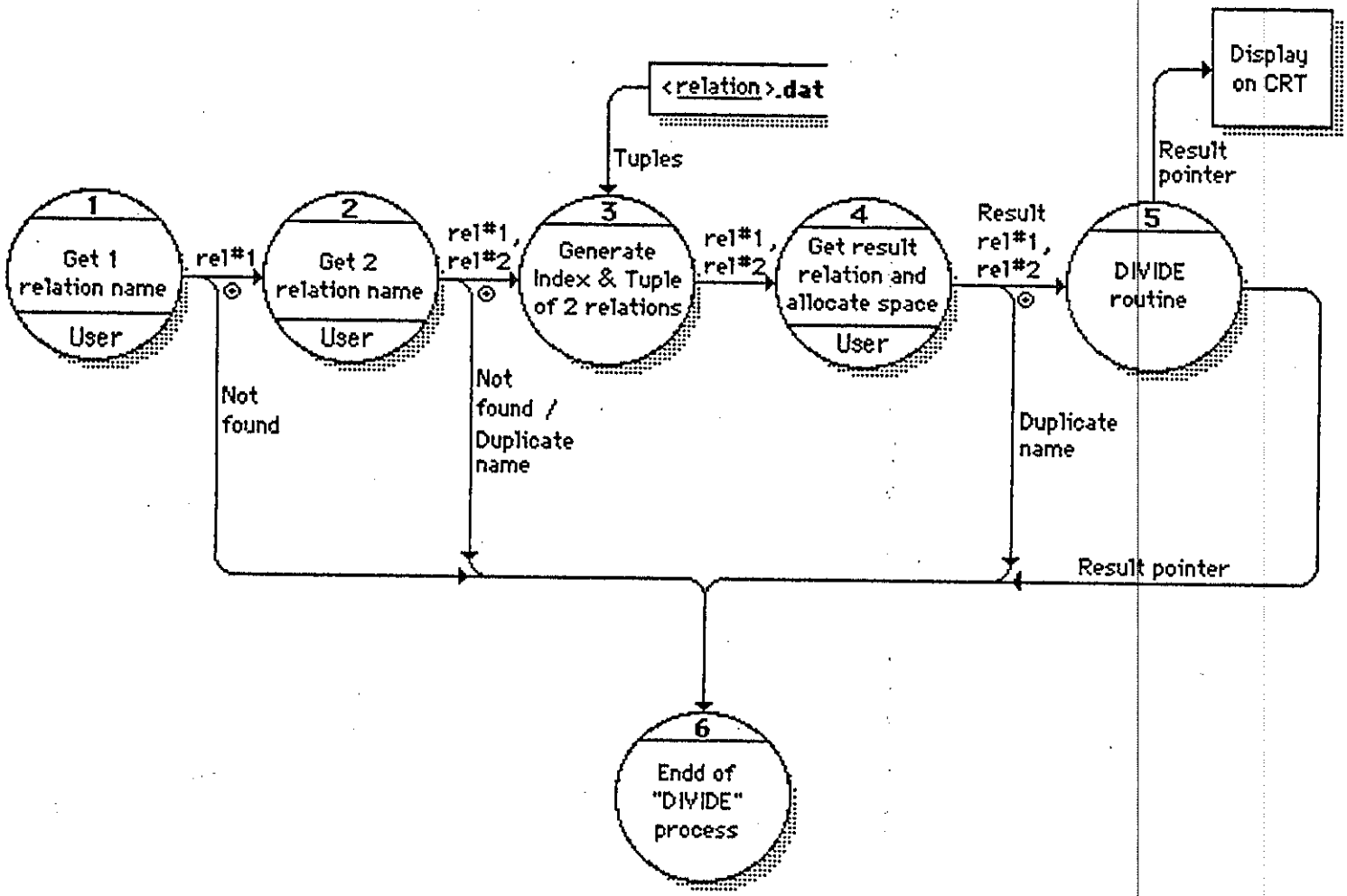
รูปที่ 3-8 แผนภาพการไหลของข้อมูลในส่วนการปฏิบัติการแบบ Select



รูปที่ 3-9 แผนภาพการไหลของข้อมูลในส่วนการปฏิบัติการแบบ Project



รูปที่ 3-10 แผนภาพการไหลของข้อมูลในส่วนการปฏิบัติการแบบ Join



รูปที่ 3-11 แผนภาพการไหลของข้อมูลในส่วนการปฏิบัติการแบบ Divide

ชนิดคือ จอภาพแบบ VT ซึ่งเป็นเทอร์มินัลของ VAX-11/785 และจอภาพ Monochrome ของเครื่องไมโครคอมพิวเตอร์ที่นำมาใช้เป็นเทอร์มินัลของ VAX-11/785

ขบวนการที่ 2 เป็นการทำงานที่ควบคุมการเลือกรายการหลัก การควบคุมการทำงานเป็นแบบ pull down menu ซึ่งผลที่ได้จากการทำงานในขบวนการนี้ทำให้ทราบว่าผู้ใช้เลือกรายการทำงานใด

ขบวนการที่ 3 เป็นการทำงานที่จัดการเกี่ยวกับฐานข้อมูลต่างๆ ที่มีอยู่ เช่น การนำฐานข้อมูลเข้า/ออกจากระบบ การขอดูรายชื่อของฐานข้อมูลที่มีอยู่ทั้งหมด การแสดงรายละเอียดของฐานข้อมูลที่กำลังใช้งานอยู่ หรือ การยกเลิกการทำงาน เป็นต้น

ขบวนการที่ 3.1 เป็นการทำงานของรายการย่อย open ซึ่งจะทำการนำฐานข้อมูลเข้าสู่ระบบ และนำโครงสร้างต่างๆ ของรีเลชั่นทั้งหมดในฐานข้อมูลนั้น ไปเก็บยังโครงสร้างข้อมูลที่ได้ออกแบบไว้ จากนั้นจึงทำการกำหนดค่าของ set up ให้มีค่าเป็น ON

ขบวนการที่ 3.2 เป็นการทำงานของรายการย่อย close directory หรือ display infor. โดยที่การทำงานจะเกี่ยวข้องกับ การนำฐานข้อมูลออกจากระบบ การขอดูรายชื่อของฐานข้อมูลที่มีอยู่ทั้งหมด การแสดงรายละเอียดของฐานข้อมูลที่กำลังใช้งานอยู่ตามลำดับ และการทำงานของรายการย่อย close จะมีผลทำให้ค่าของ set up มีค่าเป็น OFF

ขบวนการที่ 3.3 เป็นการยกเลิกการทำงานของระบบทั้งหมดแล้วกลับสู่ shell ของระบบ UNIX

ขบวนการที่ 4 เป็นขบวนการในการทำงานที่ทำการตรวจสถานะของค่า set up ว่ามีค่าเป็น ON หรือ OFF ก่อนที่จะทำงานตามรายการย่อย DBMS relation tuple R-Operation S-Operation หรือ Function ต่อไป

ขบวนการที่ 5.1 เป็นส่วนการทำงานทั่วๆ ไปภายในระบบจัดการฐานข้อมูลที่ได้พัฒนาขึ้น เช่น ส่วนของ help menu ส่วนของ option setup หรือส่วนของ exit to shell เป็นต้น

ขบวนการที่ 5.2 เป็นการทำงานในส่วนที่เกี่ยวข้องกับ การเก็บรีเลชั่นลงฐานข้อมูล การแสดงข้อมูลที่เก็บอยู่ในรีเลชั่น และการลบรีเลชั่นออกจากฐานข้อมูล เป็นต้น

ขบวนการที่ 5.3 เป็นการทำงานที่เกี่ยวข้องกับ tuple ที่อยู่ภายในรีเลชั่น เช่น การเพิ่ม ลบ และเปลี่ยนแปลงข้อมูล เป็นต้น

ขบวนการที่ 5.4 เป็นส่วนการทำงานที่เกี่ยวข้องกับ คำสั่งปฏิบัติการฐานข้อมูลที่ใช้กันทั่วๆ ไป ได้แก่ intersection union cross-product และ difference

ขบวนการที่ 5.5 เป็นส่วนของการทำงานที่เกี่ยวข้องกับคำสั่งปฏิบัติการฐานข้อมูลแบบพิเศษ เช่น select project join และ divide

ขบวนการที่ 5.6 เป็นส่วนของการทำงานที่เกี่ยวข้องกับ ฟังก์ชั่นอำนวยความสะดวกต่างๆ เช่นการหาค่าสูงสุด ต่ำสุด ผลรวม และ ค่าเฉลี่ยของ attribute ในรีเลชั่น

จากรูปที่ 3-5 เป็นแผนภาพการไหลของข้อมูลที่ทำการขยายมาจากรูปที่ 3-6 โดยแบ่งออกเป็นขบวนการต่าง ๆ 8 ขบวนการ ดังนี้คือ

ขบวนการที่ 3.3.1 รับชื่อฐานข้อมูลจากผู้ใช้และทำการค้นหาในแฟ้มข้อมูล DBMS.SYS ซึ่งเป็นแฟ้มข้อมูลที่เก็บรวบรวมรายชื่อของฐานข้อมูลทั้งหมดไว้

ขบวนการที่ 3.3.2 ทำการค้นหาแฟ้มข้อมูลที่มีชื่อเดียวกันกับชื่อฐานข้อมูลที่ระบุในขบวนการที่ 3.3.1 จาก directory "/.Rel"

ขบวนการที่ 3.3.3 ทำการอ่านชื่อ และรายละเอียดของรีเลชั่นจากแฟ้มข้อมูล <database>.rel แล้วนำรายละเอียดมาเก็บไว้ในโครงสร้างข้อมูล relation linked list

ขบวนการที่ 3.3.4 นำชื่อของรีเลชั่น ไปทำการค้นหาแฟ้มข้อมูลใน directory "/.Att" เพื่อนำรายละเอียดของ attribute มาใช้งาน

ขบวนการที่ 3.3.5 ทำการอ่านรายละเอียดของ attribute จากแฟ้มข้อมูล <database>.att แล้วนำรายละเอียดของ attribute ต่าง ๆ มาเก็บไว้ในโครงสร้างข้อมูล attribute linked list

ขบวนการที่ 3.3.6 ทำการกำหนดค่าพอยเตอร์ของระบบให้กับโครงสร้างข้อมูล

ขบวนการที่ 3.3.7 กำหนดค่าของสถานะ setup ให้มีค่าเป็น ON

จากรูปที่ 3-6 เป็นแผนภาพการไหลของข้อมูลส่วนหนึ่งที่ต่อจากขบวนการที่ 5.4 ในส่วนปฏิบัติการแบบ intersection union และ cross-product โดยแบ่งออกเป็นขบวนการต่าง ๆ 9 ขบวนการ ดังนี้คือ

ขบวนการที่ 1 ทำการรับชื่อรีเลชั่นแรกจากผู้ใช้

ขบวนการที่ 2 ทำการรับชื่อรีเลชั่นที่สองจากผู้ใช้

ขบวนการที่ 3 ทำการตรวจสอบโครงสร้างของรีเลชั่นทั้งสองว่า เข้ากันได้หรือไม่

ขบวนการที่ 4 อ่านข้อมูลของรีเลชั่นทั้งสองจากแฟ้มข้อมูล <relation>.dat แล้วนำไปเก็บในโครงสร้างข้อมูลของ tuple linked list และ binary search tree

ขบวนการที่ 5 ทำการรับชื่อรีเลชั่นที่ต้องการจะนำผลลัพธ์ไปเก็บ

ขบวนการที่ 6.1 เป็นส่วนการทำงานของคำสั่งปฏิบัติการแบบ intersection

ขบวนการที่ 6.2 เป็นส่วนการทำงานของคำสั่งปฏิบัติการแบบ union

ขบวนการที่ 6.3 เป็นส่วนการทำงานของคำสั่งปฏิบัติการแบบ difference

ขบวนการที่ 7 เป็นส่วนสิ้นสุดการทำงานในส่วนคำสั่งปฏิบัติการ intersection union และ difference

จากรูปที่ 3-7 เป็นแผนภาพการไหลของข้อมูลส่วนหนึ่งที่ต่อจากขบวนการที่ 5.4 ในส่วนปฏิบัติการแบบ cross product โดยแบ่งออกเป็นขบวนการต่าง ๆ 6 ขบวนการ ดังนี้คือ

ขบวนการที่ 1 ทำการรับชื่อรีเลชั่นแรกจากผู้ใช้

ขบวนการที่ 2 ทำการรับชื่อรีเลชั่นที่สองจากผู้ใช้

ขบวนการที่ 3 อ่านข้อมูลของรีเลชั่นทั้งสองจากแฟ้มข้อมูล <relation>.dat แล้วนำไปเก็บในโครงสร้างข้อมูลของ tuple linked list และ binary search tree

ขบวนการที่ 4 ทำการรับชื่อรีเลชั่นที่ต้องการจะนำผลลัพธ์ไปเก็บ

ขบวนการที่ 5 เป็นส่วนการทำงานของคำสั่งปฏิบัติการแบบ cross product

ขบวนการที่ 6 เป็นส่วนสิ้นสุดการทำงานในส่วนคำสั่งปฏิบัติการ cross product

จากรูปที่ 3-8 เป็นแผนภาพการไหลของข้อมูลส่วนหนึ่งที่ต่อจากขบวนการที่ 5.5 ในส่วนปฏิบัติการแบบ select โดย แบ่งออกเป็นขบวนการต่าง ๆ 7 ขบวนการ ดังนี้คือ

ขบวนการที่ 1 ทำการรับชื่อรีเลชั่นจากผู้ใช้

ขบวนการที่ 2 แสดงชื่อ attribute ทั้งหมดของรีเลชั่น

ขบวนการที่ 3 อ่านข้อมูลของรีเลชั่นจากแฟ้มข้อมูล <relation>.dat แล้วนำไปเก็บในโครงสร้างข้อมูลของ tuple linked list และ binary search tree

ขบวนการที่ 4 รับชื่อของ attribute (เพียง 1 ชื่อ) และเงื่อนไขในการ select

ขบวนการที่ 5 ทำการรับชื่อรีเลชั่นที่ต้องการจะนำผลลัพธ์ไปเก็บ

ขบวนการที่ 6 เป็นส่วนการทำงานของคำสั่งปฏิบัติการแบบ select

ขบวนการที่ 7 เป็นส่วนสิ้นสุดการทำงานในส่วนคำสั่งปฏิบัติการ select

จากรูปที่ 3-9 เป็นแผนภาพการไหลของข้อมูลส่วนหนึ่งที่ต่อจากขบวนการที่ 5.5 ในส่วนปฏิบัติการแบบ project โดยแบ่งออกเป็นขบวนการต่าง ๆ 7 ขบวนการ ดังนี้คือ

ขบวนการที่ 1 ทำการรับชื่อรีเลชั่นจากผู้ใช้

ขบวนการที่ 2 แสดงชื่อ attribute ทั้งหมดของรีเลชั่น

ขบวนการที่ 3 อ่านข้อมูลของรีเลชั่นจากแฟ้มข้อมูล <relation>.dat แล้วนำไปเก็บในโครงสร้างข้อมูลของ tuple linked list และ binary search tree

ขบวนการที่ 4 รับรายชื่อของ attribute ที่ต้องการ ซึ่งสามารถที่จะมีจำนวน attribute ที่ต้องการ project ได้มากกว่าหนึ่งชื่อ

ขบวนการที่ 5 ทำการรับชื่อรีเลชั่นที่ต้องการจะนำผลลัพธ์ไปเก็บ

ขบวนการที่ 6 เป็นส่วนการทำงานของคำสั่งปฏิบัติการแบบ project

ขบวนการที่ 7 เป็นส่วนสิ้นสุดการทำงานในส่วนคำสั่งปฏิบัติการ project

จากรูปที่ 3-10 เป็นแผนภาพการไหลของข้อมูลส่วนหนึ่งที่ต่อจากขบวนการที่ 5.5 ในส่วนปฏิบัติการแบบ join โดย แบ่งออกเป็นขบวนการต่าง ๆ 11 ขบวนการ ดังนี้คือ

ขบวนการที่ 1 ทำการรับชื่อรีเลชั่นแรกจากผู้ใช้

ขบวนการที่ 2 ทำการรับชื่อรีเลชั่นที่สองจากผู้ใช้

ขบวนการที่ 3 อ่านข้อมูลของรีเลชั่นทั้งสองจากแฟ้มข้อมูล <relation>.dat แล้วนำไปเก็บในโครงสร้างข้อมูลของ tuple linked list และ binary search tree

ขบวนการที่ 4 แสดงรายชื่อของ attribute ทั้งหมดของรีเลชั่นที่ 1

ขบวนการที่ 5 ทำการรับชื่อของ attribute ของรีเลชั่นที่ 1

ขบวนการที่ 6 รับเงื่อนไขในการ join

ขบวนการที่ 7 แสดงรายชื่อของ attribute ทั้งหมดของรีเลชั่นที่ 2

ขบวนการที่ 8 ทำการรับชื่อของ attribute ของรีเลชั่นที่ 2

ขบวนการที่ 9 ทำการรับชื่อรีเลชั่นที่ต้องการจะนำผลลัพธ์ไปเก็บ

ขบวนการที่ 10 เป็นส่วนการทำงานของคำสั่งปฏิบัติการแบบ join

ขบวนการที่ 11 เป็นส่วนสิ้นสุดการทำงานในส่วนคำสั่งปฏิบัติการ join

จากรูปที่ 3-11 เป็นแผนภาพการไหลของข้อมูลส่วนหนึ่งที่ต่อจากขบวนการที่ 5.5 ในส่วนปฏิบัติการแบบ divide โดยแบ่งออกเป็นขบวนการต่าง ๆ 6 ขบวนการ ดังนี้คือ

ขบวนการที่ 1 ทำการรับชื่อรีเลชั่นแรกจากผู้ใช้

ขบวนการที่ 2 ทำการรับชื่อรีเลชั่นที่สองจากผู้ใช้

ขบวนการที่ 3 อ่านข้อมูลของรีเลชั่นทั้งสองจากแฟ้มข้อมูล <relation>.dat แล้วนำไปเก็บในโครงสร้างข้อมูลของ tuple linked list และ binary search tree

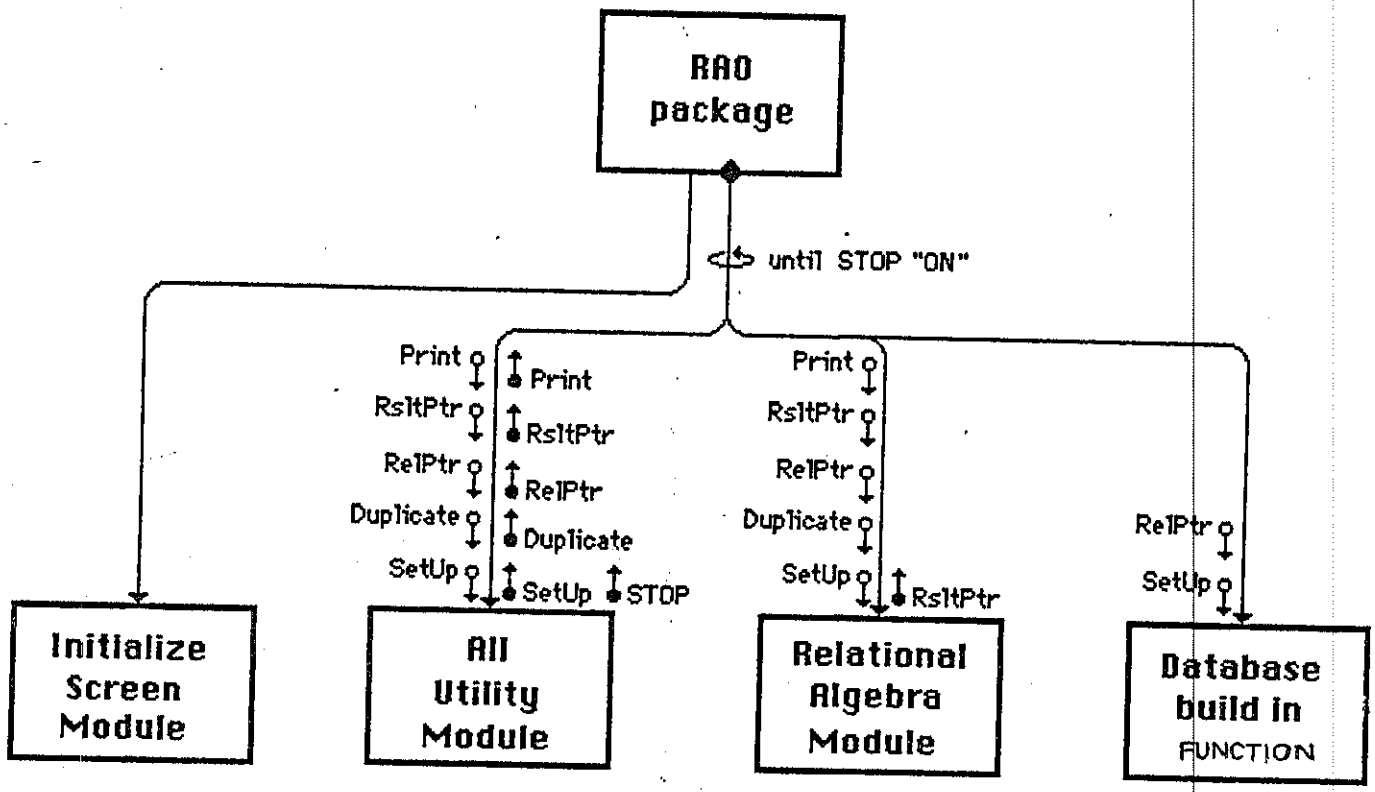
ขบวนการที่ 4 ทำการรับชื่อรีเลชั่นที่ต้องการจะนำผลลัพธ์ไปเก็บ

ขบวนการที่ 5 เป็นส่วนการทำงานของคำสั่งปฏิบัติการแบบ divide

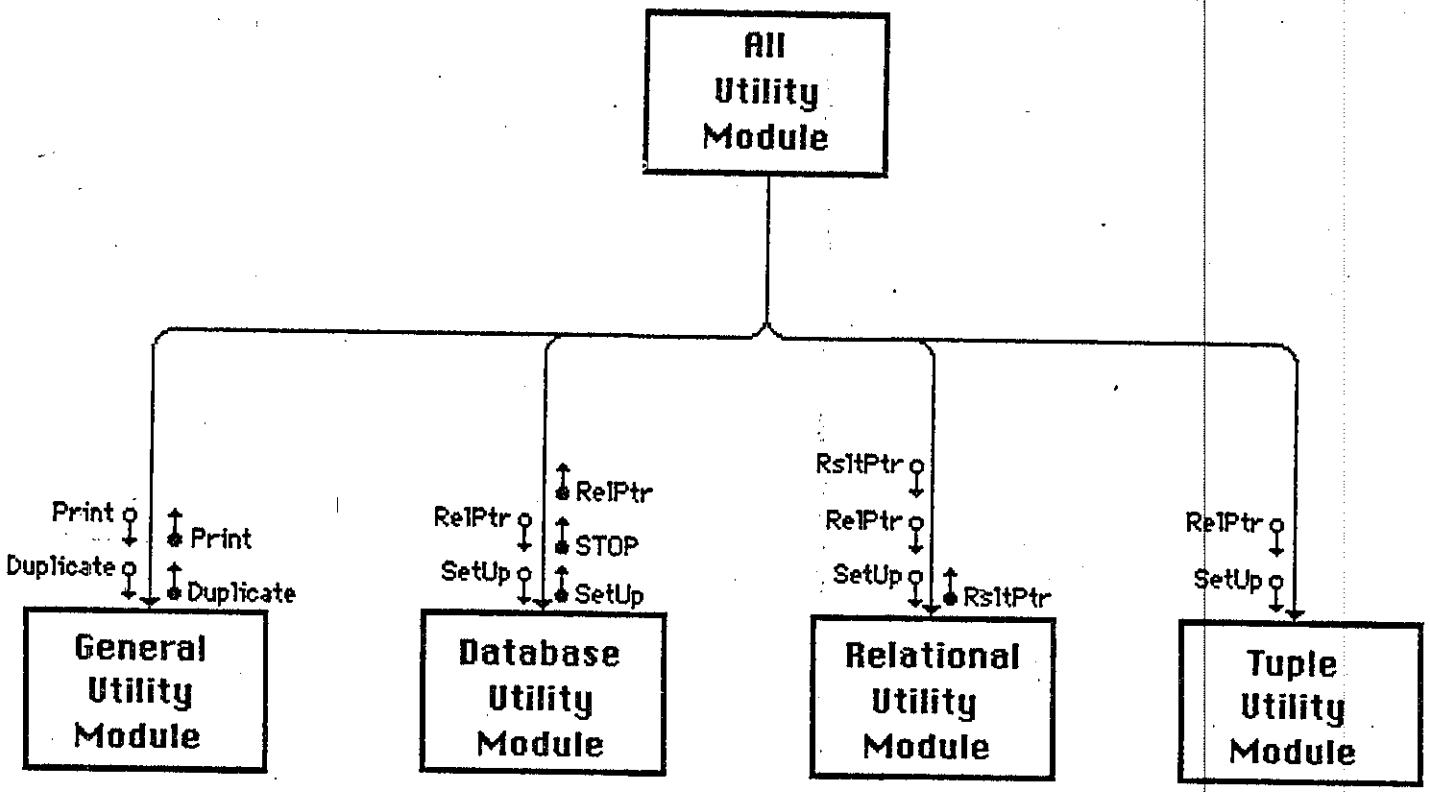
ขบวนการที่ 6 เป็นส่วนสิ้นสุดการทำงานในส่วนคำสั่งปฏิบัติการ divide

3.3.5 แผนภาพโครงสร้างโปรแกรม

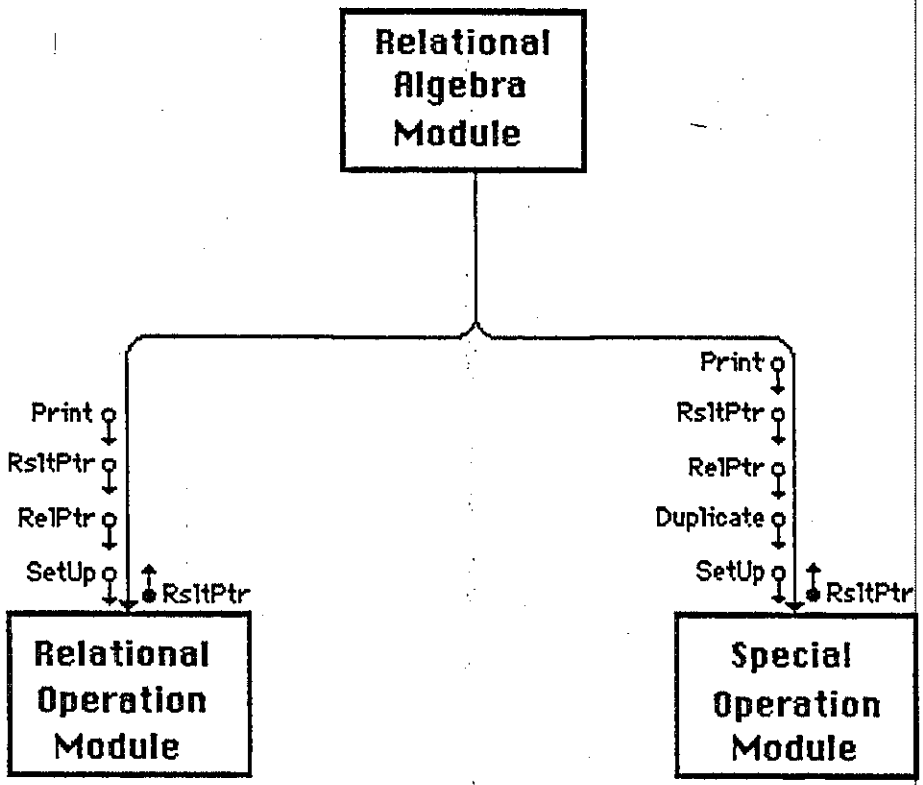
ขั้นตอนต่อมาหลังจากที่ได้การไหลของข้อมูลแล้ว เราสามารถที่จะมองระบบ ออกเป็น โมดูล (module) ได้ จากนั้นจึงนำโมดูลเหล่านั้นมาพัฒนาเป็นโปรแกรมย่อยต่อไป สำหรับแผนภาพโครงสร้างโปรแกรม (structure diagram) ของระบบแสดง ดังรูป 3-12 ถึง 3-14



รูปที่ 3-12 แผนภาพ structure diagram ของระบบ



รูปที่ 3-13 แผนภาพ structure diagram ของการทำงานในส่วน utility



รูปที่ 3-14 แผนภาพ structure diagram ของคำสั่งปฏิบัติการแบบพีชคณิตสัมพันธ์

จากรูปที่ 3-12 : เป็นการมองและแบ่งระบบทั้งหมดออกเป็น 4 ส่วนใหญ่ๆ คือ

- Initialize screen module จะเป็นส่วนการทำงานที่จัดการกับจอภาพทั้งหมดก่อนการใช้งาน ในลักษณะที่เป็น หน้าต่าง (window)

- All utility module เป็นส่วนการทำงานที่เก็บระบบอำนวยความสะดวกต่างๆ เพื่อช่วยในการใช้งานของโปรแกรม เช่น Help menu และ Option setup เป็นต้น

- Relational algebra module เป็นส่วนการทำงานที่เกี่ยวข้องกับคำสั่งปฏิบัติการกับข้อมูลแบบพีชคณิตสัมพันธ์ทั้งหมดซึ่ง ได้แก่ Intersect Union Difference Cross-product Select Project Join และ Divide

- Database built-in module เป็นส่วนการทำงานที่เกี่ยวข้องกับฟังก์ชั่นพื้นฐานในการปฏิบัติการกับค่าต่างๆ ของ attribute ของรีเลชั่น เช่น คำสั่ง Count Maximum Minimum Sum และ Average เป็นต้น

จากรูปที่ 3-13 : เป็นการแบ่งการทำงานในส่วนของ All utility module ออกเป็น 4 ส่วนใหญ่ๆ ดังนี้คือ

- General utility module เป็นส่วนของคำสั่งในการอำนวยความสะดวกทั่วๆ ไป เช่น Help-menu Exit-to-shell Option-setup เป็นต้น

- Database utility module เป็นส่วนของคำสั่งในการอำนวยความสะดวกที่เกี่ยวข้องเฉพาะกับฐานข้อมูล เช่น คำสั่ง Open Close Directory หรือ Display-information เป็นต้น

- Relation utility module เป็นส่วนของคำสั่งในการอำนวยความสะดวกที่เกี่ยวข้องเฉพาะรีเลชั่น ในฐานข้อมูลที่กำลังใช้งานอยู่ในระบบเท่านั้น เช่น คำสั่ง

Save Display หรือ Remove เป็นต้น

- Tuple utility module เป็นส่วนของคำสั่งในการอำนวยความสะดวกที่เกี่ยวข้องเฉพาะ tuple ของรีเลชั่นเท่านั้น เช่น คำสั่ง Insert delete หรือ update เป็นต้น

จากรูปที่ 3-14 : เป็นการแบ่งการทำงานในส่วนของ Relational algebra module ออกเป็น 2 ส่วน คือ

- Relational operation module เป็นส่วนของคำสั่งปฏิบัติการแบบเซท ซึ่งได้แก่ คำสั่ง Intersect Union Difference และ Cross-product

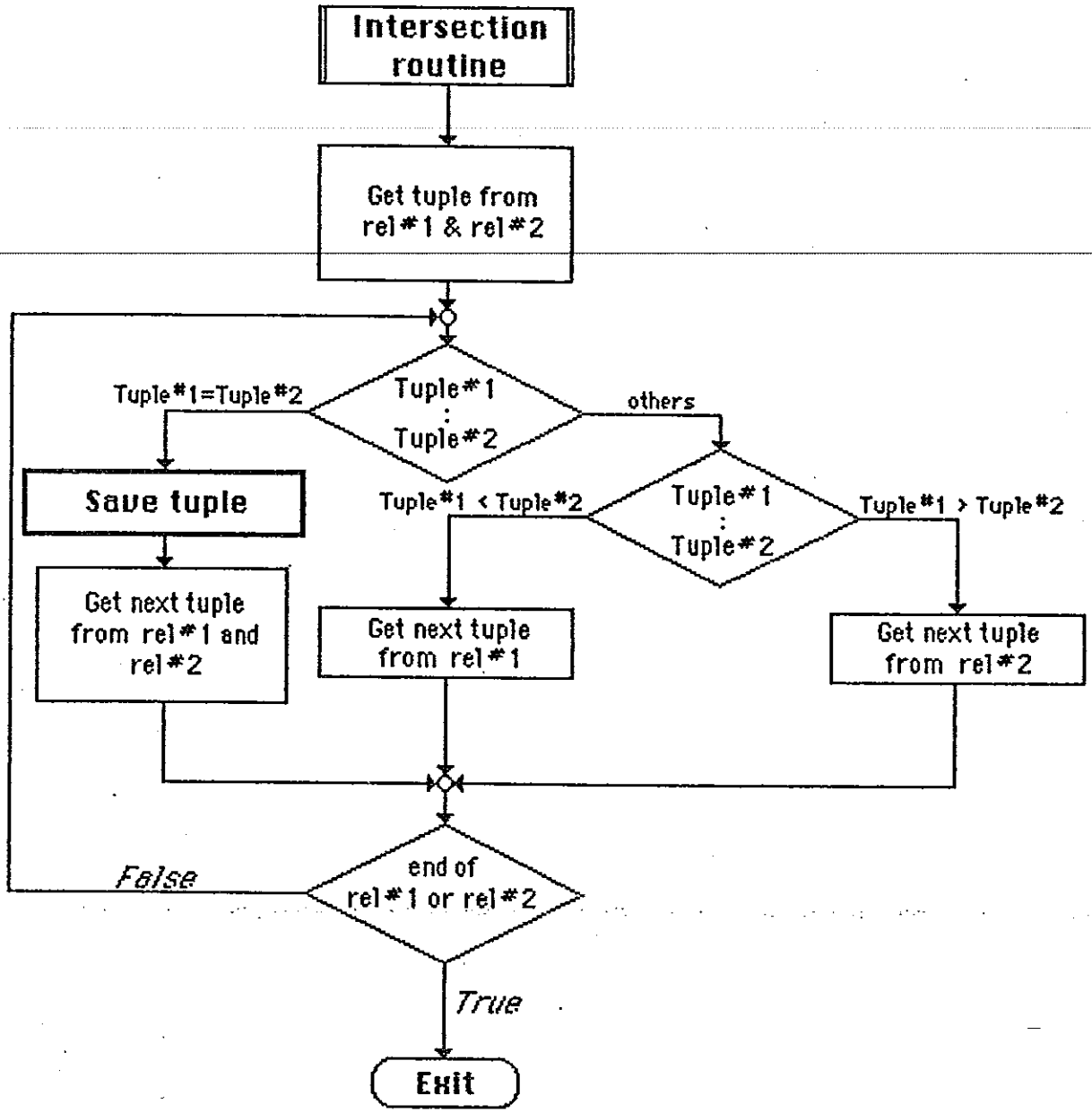
- Sepecial operation module เป็นส่วนของคำสั่งปฏิบัติการนิเศษที่นอกเหนือจากในส่วนแรก คำสั่งในส่วนนี้ได้แก่ คำสั่ง Select Project Join และ Divide

3.3.6 แผนภูมิแสดงขั้นตอนวิธีการทำงานของคำสั่งปฏิบัติการแบบพีชคณิตสัมพันธ์

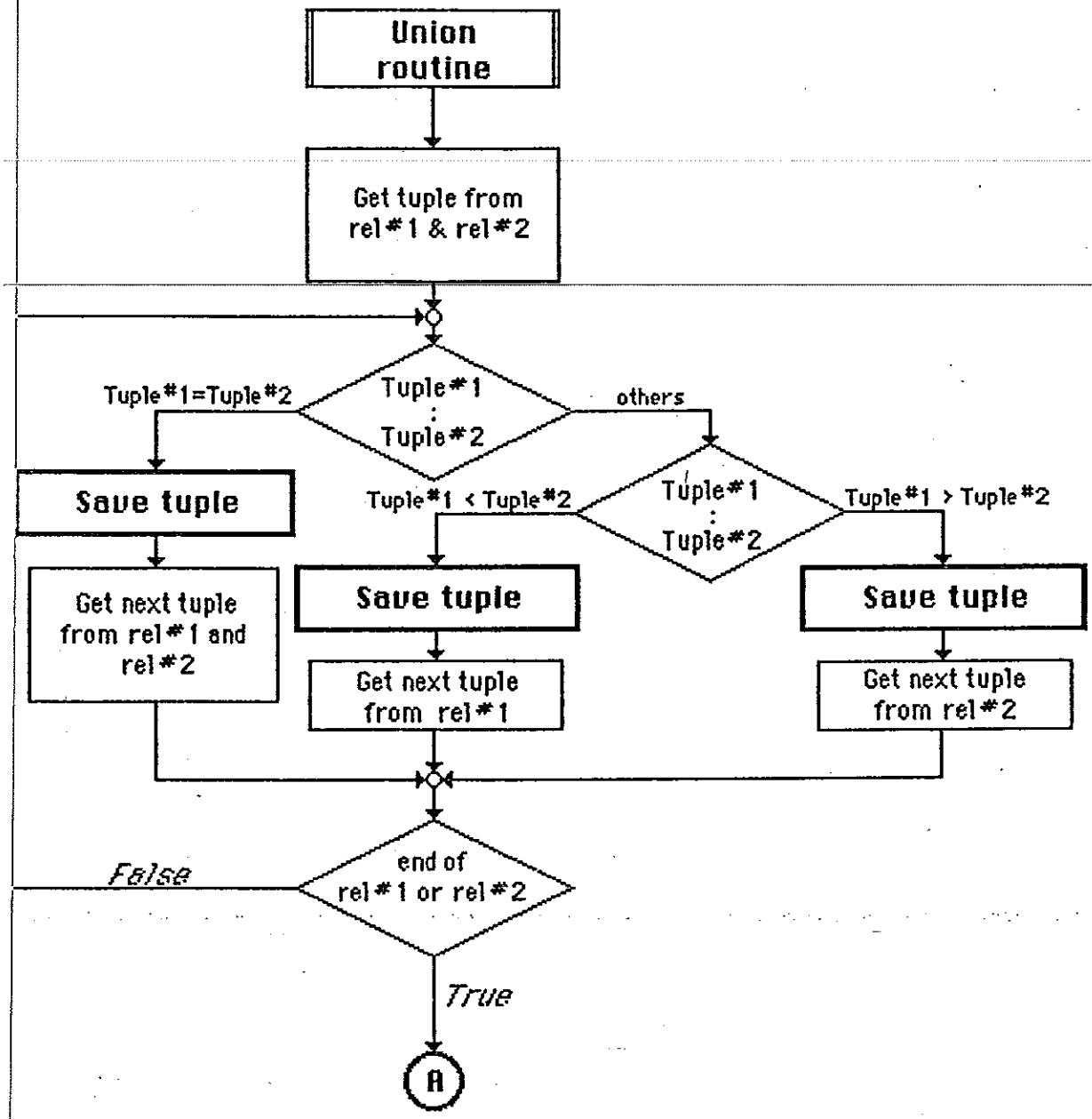
แผนภาพ (flowchart) แสดงขั้นตอนวิธี (algorithm) การทำงานของ คำสั่งปฏิบัติการแบบพีชคณิตสัมพันธ์ทั้งหมดแสดงในรูป 3-15 ถึงรูปที่ 3-22

3.4 การพัฒนาโปรแกรม

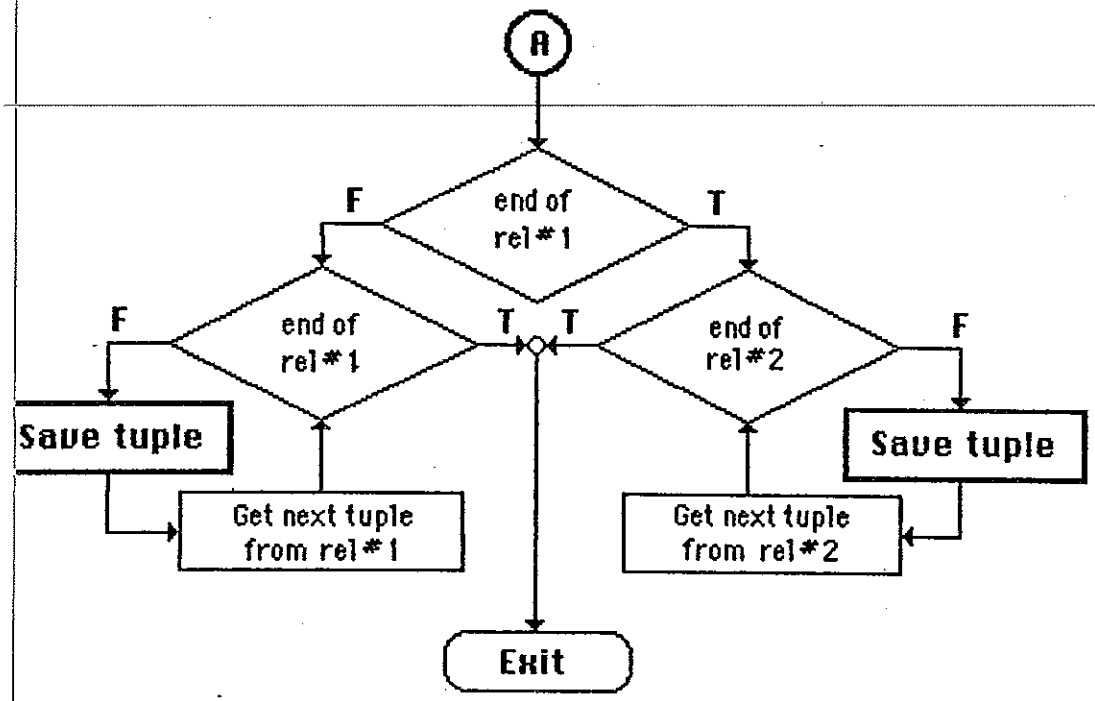
ในการพัฒนาโปรแกรมสำหรับการวิจัยครั้งนี้ ได้พัฒนาโปรแกรมด้วยภาษา C ทั้งหมด ดังนั้นรูปแบบของข้อมูล หรือการกำหนดต่างๆ ที่จะกล่าวต่อไป จึงยกมาจากรูปแบบของภาษา C เลย โดยที่ข้อความที่อยู่ระหว่างเครื่องหมาย /* กับ */ ถือว่าเป็นส่วนของการอธิบายสิ่งที่ได้กำหนดขึ้น



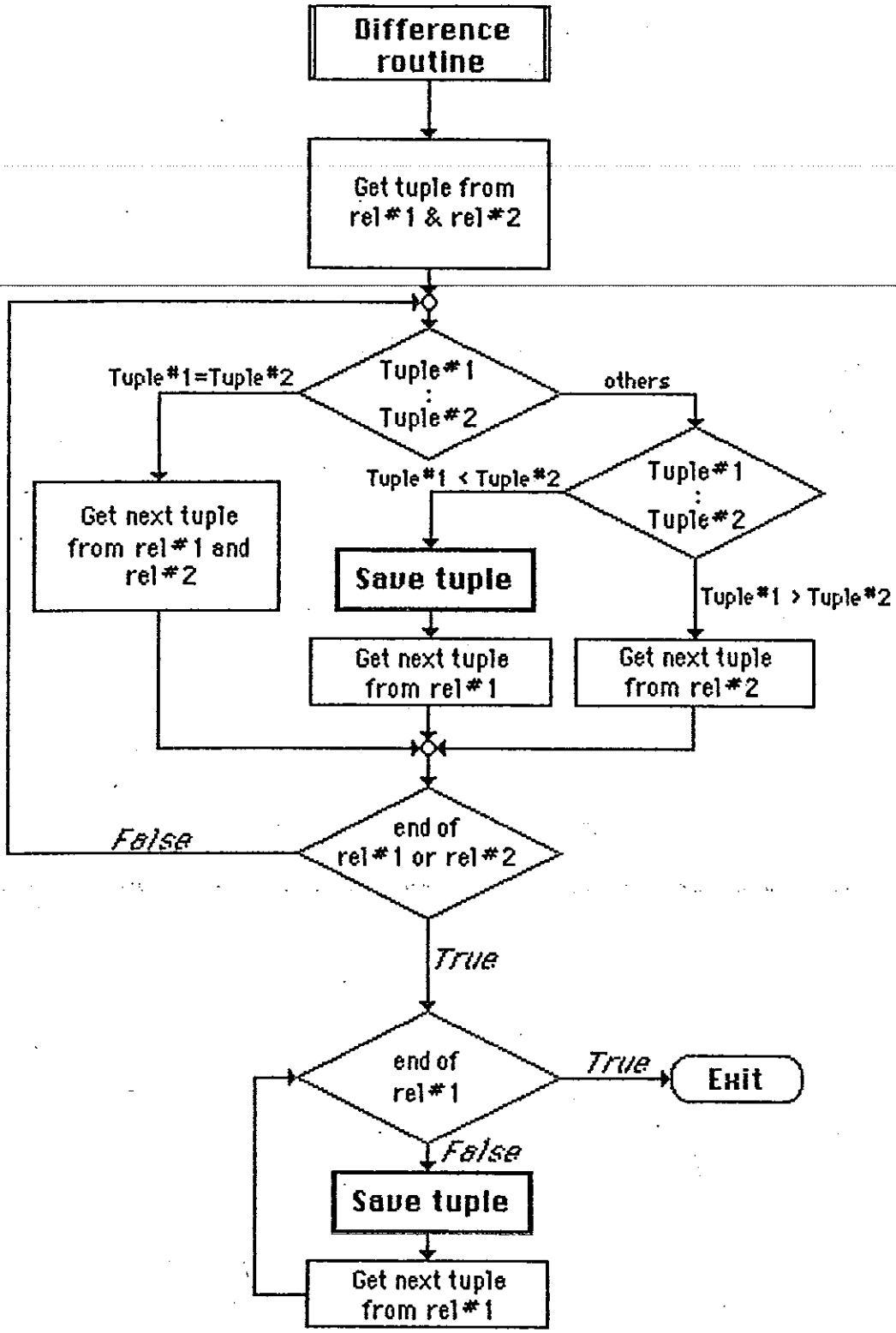
รูปที่ 3-15 flow chart ของคำสั่งปฏิบัติการ Intersection



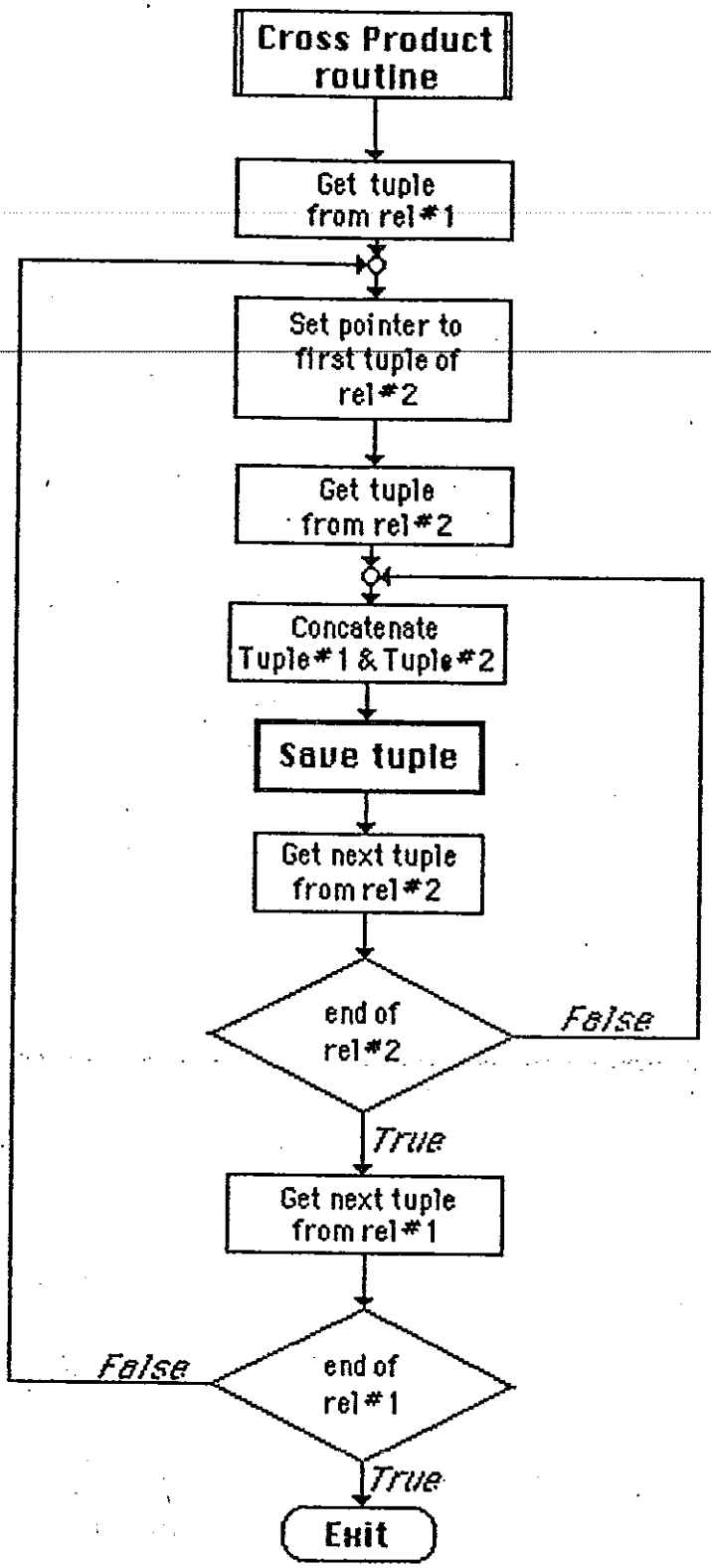
รูปที่ 3-16 flow chart ของคำสั่งปฏิบัติการ Union



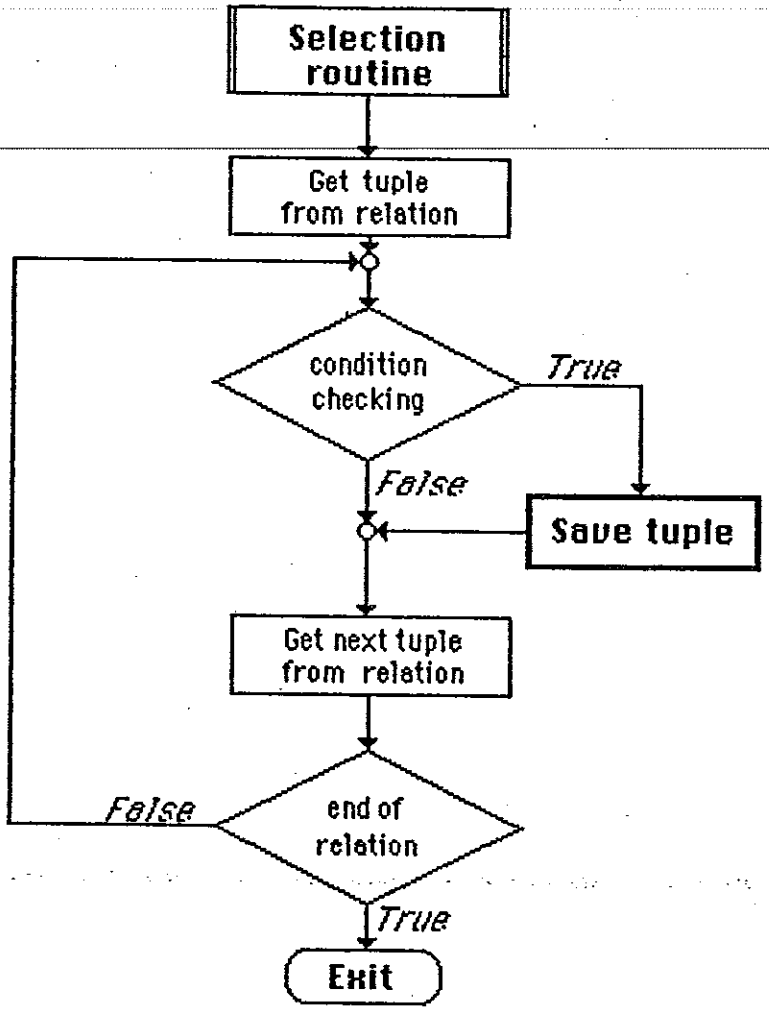
รูปที่ 3-17 flow chart ของคำสั่งปฏิบัติการ Union (ต่อ)



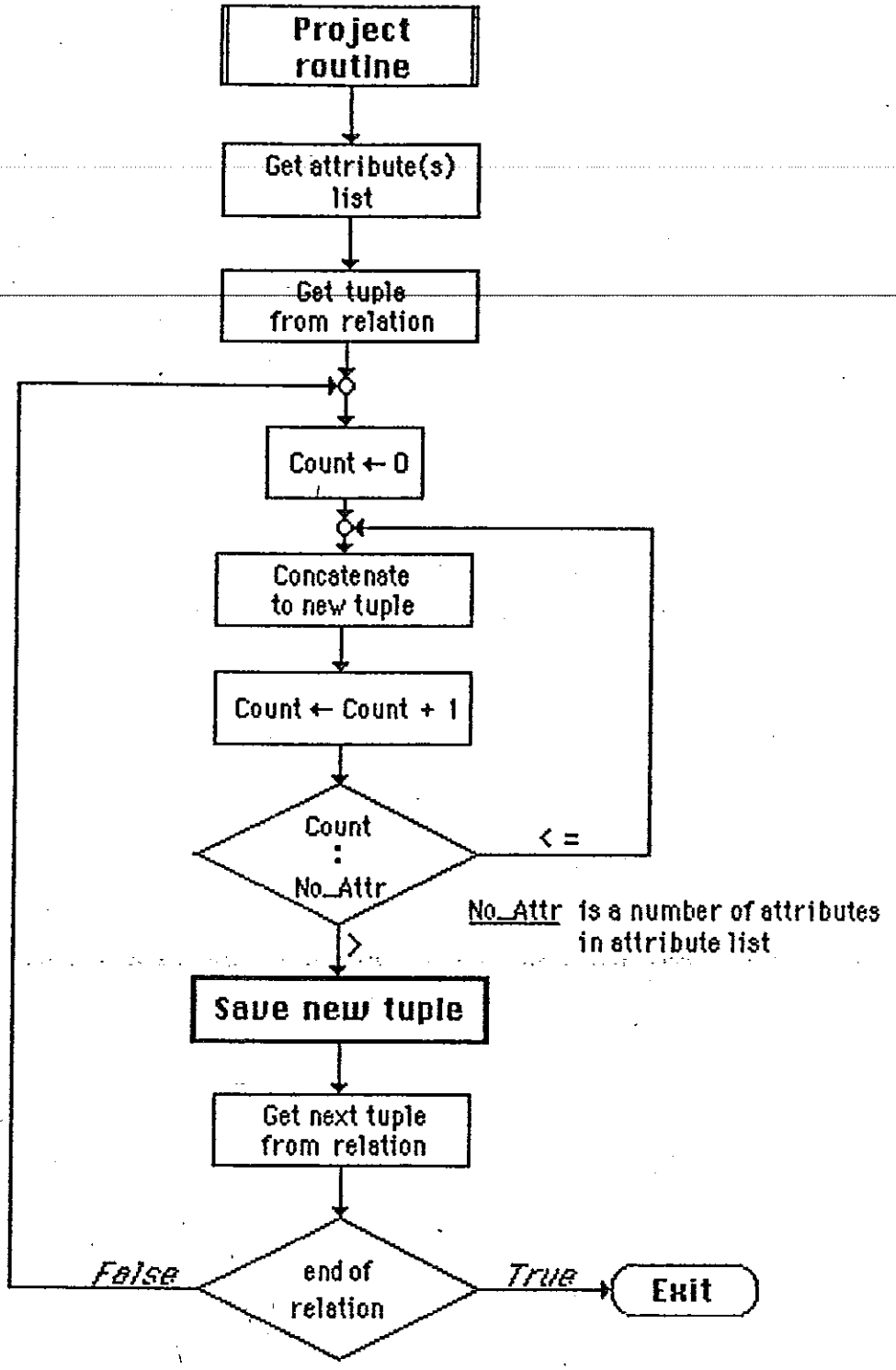
รูปที่ 3-18 flow chart ของคำสั่งปฏิบัติการ Difference



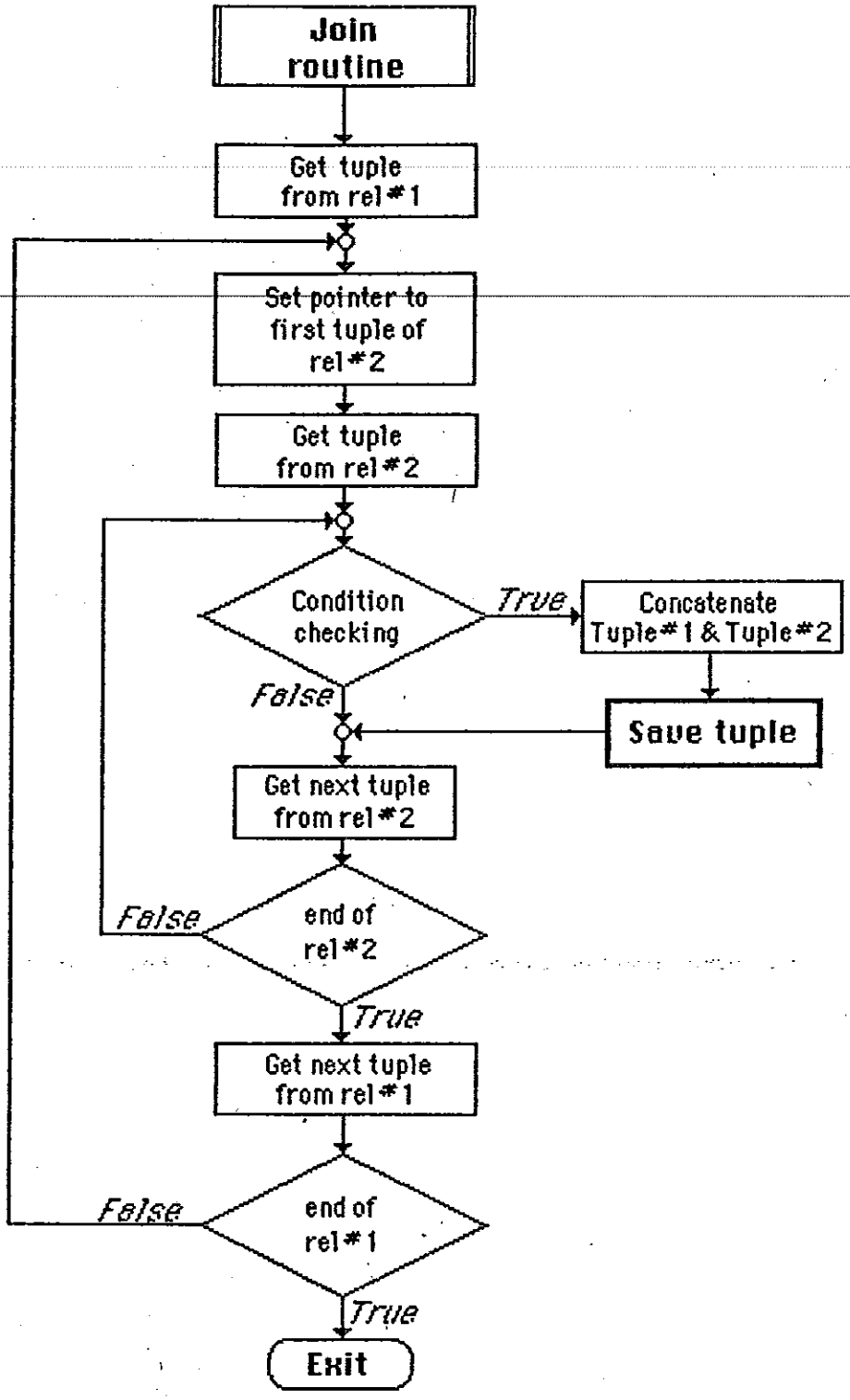
รูปที่ 3-19 flow chart ของคำสั่งปฏิบัติการ Cross product



รูปที่ 3-20 flow chart ของคำสั่งปฏิบัติการ Select



รูปที่ 3-21 flow chart ของคำสั่งปฏิบัติการ Project



รูปที่ 3-22 flow chart ของคำสั่งปฏิบัติการ Join

3.4.1 โครงสร้างข้อมูลที่ใช้ในการพัฒนาโปรแกรม

จากโครงสร้างข้อมูลในรูปที่ 3-2 สามารถเขียนโครงสร้างทั้งหมดด้วยภาษา C โดยแบ่งเป็นหัวข้อได้ ดังนี้คือ

- โครงสร้างของ Tuple node

```
#define    TSIZE     256       /* Maximum length of tuple */


/*--- Tuple Catalog ---*/
struct tupl {
   char           TupBuf[TSIZE+1];  /* Data area */
   struct tupl    *TupNext;         /* Pointer to next tuple node */
};
typedef struct tupl  TUPNODE;    /* Tuple node for tuple linked list */
```

- โครงสร้างของ Attribute node

```
#define    MAXATT    20   /* No. of attribute per relation */
#define    ANAME     15   /* Length of attribute's name */
struct attr {
   char          AttName[ANAME+1]; /* Name of attribute */
   int           AttDom;           /* Domain type */
   int           AttFstPos;        /* 1st position in tuple */
   int           AttLstPos;        /* End position in tuple */
   int           AttLngth;         /* Size of attribute */
```

```
    char          AttStatus;        /* Type of attribute */
    struct attr   *AttNext;         /* Pointer to next
                                       attribute node */
};
typedef  struct attr  ATTNODE;      /* Attribute node for
                                       attribute linked list */
```

ค่าของ AttStatus ในโครงสร้าง attr จะมีค่าที่เป็นไปได้ 3 ค่า ตามการกำหนดค่า ต่อไปนี้

```
#define    PRIME          'P'      /* Primary key */
#define    ALTERNATE      'A'      /* Alternate key */
#define    NON            'N'      /* Non-key attrinute */
```

- โครงสร้างของ Binary search tree node

```
struct tree {
    char          TreItem[TSIZE+1]; /* Key value */
    TUPNODE       *TreTupPtr;       /* Point to tuple node */
    struct tree   *TreLeftPtr;      /* Left pointer */
    struct tree   *TreRghtPtr;      /* Right pointer */
};
typedef struct tree  TRENODE;       /* Tree node for binary
                                       search tree */
```

- โครงสร้างของ Relation node

```
struct rel {
    char        RelName[RNAME+1]; /* Relation's name */
    ATTNODE     *RelAttPtr;     /* Pointer to attribute linked list */
    TUPNODE     *RelTupPtr;     /* Pointer to tuple linkedlist */
    int         RelNoAtt;       /* Number of attribute(s) */
    int         RelNoTup;       /* Number of tuple(s) */
    int         RelTupLngth;    /* Tuple length */
    int         RelKeySize;     /* Size of primary key */
    char        RelCreate[9];   /* Create date MM/DD/YY */
    char        RelStatus;      /* Type of relation */
    TRENODE     *RelIndxPtr;    /* Pointer to binary search tree */
    struct rel  *RelNext;       /* Pointer to next relation node */
};
typedef struct rel  RELNODE;    /* Relation node for list */
```

ค่าของ RelStatus จะมีค่าที่เป็นไปได้ 3 ค่า ตามการกำหนดค่า ดังนี้คือ

```
#define   BASE      'B'       /* Base relation */
#define   KEEP      'K'       /* Keep relation */
#define   TEMP      'T'       /* Temporary relation */
```

3.4.2 การกำหนดค่าอื่นๆ

- การกำหนดค่า directory path สำหรับแฟ้มข้อมูลของระบบ

```
#define RELPATH  "/grad/g292403/DML/.Rel"  /* Relation path */
#define ATTPATH  "/grad/g292403/DML/.Att"  /* Attribute path */
#define DATPATH  "/grad/g292403/DML/.Dat"  /* Data path */
```

- การกำหนดค่าฟังก์ชั่นเพื่อจองเนื้อในหน่วยความจำ

```
#define  MALLOC    ( (x *) malloc(sizeof(x) )
```

- การกำหนดค่าสำหรับชนิดของจอภาพ

```
#define  MICRO     0     /* Terminal of PC system */
#define  VT        1     /* Terminal of VAX system */
```

- การกำหนดค่าสำหรับการเปรียบเทียบทางตรรกศาสตร์

```
#define  GT        30    /* Greater than */
#define  LT        31    /* Less than */
#define  GE        32    /* Greater than or equal to */
#define  LE        33    /* Less than or equal to */
#define  EQ        34    /* Equal to */
#define  NE        35    /* Not equal to */
```

- การกำหนดค่าสำหรับคำสั่งการทำงานต่างๆ

```
#define  NO_OP          0
#define  ABOUT_DBMS     1
#define  HELP           2
#define  CREATE         3
#define  OPEN           4
#define  CLOSE          5
#define  PRINT_DBS      6
#define  DBMS_INFOR     7
#define  DIRECTORY      8
#define  QUIT           9
#define  SAVE           10
#define  RENAME         11
#define  REL_DISPLAY    12
#define  PRINT_REL      13
#define  PRINT_ALL      14
#define  MODIFY         15
#define  INSERT         16
#define  DELETE         17
#define  UPDATE         18
#define  INTERSECT      20
#define  UNION          21
#define  DIFF           22
```

```
#define  CROSS       23
#define  SELECT      24
#define  PROJECT     25
#define  JOIN        26
#define  DIVIDE      27
#define  EXT_SHELL   40
#define  SET_OPTION  41
```

- <u>การกำหนดค่าการควบคุมการแสดงผลทางจอภาพ</u>

```
/*--- Graphic mode for VT terminal ---*/
#define  EnGph   printf("^[(0")   /* Enter graphic mode */
#define  ExGph   printf("^[(B")   /* Exit from graphic mode */

/*--- Control intensity display ---*/
#define  NORMODE     printf("^[[1m")  /* Normal intensity */
#define  HGHMODE     printf("^[[0m")  /* High intensity */

/*--- Control cursor display ---*/
#define  CURS      printf("^[[?25h")  /* Visible cursor */
#define  NCURS     printf("^[[?25l")  /* Invisible cursor */
```

```
/*--- Control scren display ---*/
#define SMOOTH      printf("^[[?4h")    /* Smooth scrolling */
#define JUMP        printf("^[[?4l")    /* Jump scrolling */
```

- <u>การกำหนดค่าคงที่สำหรับ window ต่างๆ ในโปรแกรม</u>

```
#define  MAXMENU     6      /* No. of element in main menu */
#define  MaxDbms     4      /* No. of item in DBMS menu */
#define  MaxDbse     7      /* No. of item in Database menu */
#define  MaxRela     8      /* No. of item in Relation menu */
#define  MaxTupl     3      /* No. of item in Tuple menu */
#define  MaxRelO     4      /* No. of item in R-Operation menu */
#define  MaxSpeO     4      /* No. of item in S-Operation menu */
#define  MaxFunc     5      /* No. of item in Function menu */
#define  Size_Dbms   19     /* Size of DBMS element */
#define  Size_Dbse   17     /* Size of Database element */
#define  Size_Rela   13     /* Size of Relation element */
#define  Size_Tupl   10     /* Size of Tuple element */
#define  Size_RelO   17     /* Size of R-Operation element */
#define  Size_SpeO   11     /* Size of S-Operation element */
#define  Size_Func   11     /* Size of Function element */
```

3.4.3 ตัวแปรที่สำคัญ

- ตัวแปรที่ใช้งานทั่วไปในระบบ

```
RELNODE  *RelPtr;        /* System pointer */
RELNODE  *RsltPtr;       /* Pointer for temperary relation */
char     DbaseName[RNAME+1]; /* Database name for system */
int      SETUP;          /* Loading database status */
int      TERM_TYPE;      /* Save a terminal type (VT/MICRO) */
int      OP_CODE;        /* An operation code */
int      SET_PRINT;      /* Status of print output to file */
int      DUPLICATE;      /* Elimination of duplicate status */
char     Cmd[BUFSIZ+1]; /* For commanf in process */
```

- ตัวแปรสำหรับเก็บรายชื่อของ menu

```
/*---- Define Main Menu ----*/
static char *menu[] = {" DBMS "," Database "," Relation  ",
                       " Tuple  "," R-Operation  ",
                       " S-Operation  "," Function  "};
/*---- Define DBMS menu ----*/
static char *Dbms[]  =  {"  About DBMS       ",
                         "  Help menu        ",
                         "  Exit to C-shell  ",
                         "  Option setup     "};
```

```
/*---- Define Database menu ----*/

static char *Dbse[]  =  {"  Create          ",
                         "  Open            ",
                         "  Close           ",
                         "* Print           ",
                         "  Display Infor.  ",
                         "  Directory       ",
                         "  Quit            "};

/*---- Define Relation menu ----*/

static char *Rela[]  =  {"  Save        ",
                         "* Rename      ",
                         "  Display     ",
                         "* Print       ",
                         "* Print all   ",
                         "* Modify      ",
                         "* Structure   ",
                         "  Remove      "};

/*---- Define Tuple menu ----*/

static char *Tupl[]  =  {"* Insert  ",
                         "* Delete  ",
                         "* Update  "};
```

```
/*---- Define R-Operation menu ----*/
static char *RelO[] = {" Intersection   ",
                       " Union          ",
                       " Difference     ",
                       " Cross product  "};

/*---- Define S-Operation menu ----*/
static char *SprO[] = {" Select   ",
                       " Project  ",
                       " Join     ",
                       " Divide   "};

/*---- Define Function menu ----*/
static char *Func[] = {" Count  ",
                       " Maximun  ",
                       " Minimum  ",
                       " Sum      ",
                       " Average  "};
```

3.4.4 ฟังก์ชันและ การทำงาน

Close_Database()

เป็นการทำงานในส่วนของการลบฐานข้อมูลออกจากระบบ เพื่อให้ระบบว่างพร้อมที่

จะนำฐานข้อมูลใหม่เข้ามาในระบบต่อไป ฟังก์ชั่นนี้จะเปลี่ยนสถานะของตัวแปร SETUP ให้มีค่าเป็น OFF ซึ่งจะหมายความว่าในระบบไม่มีฐานข้อมูล

Directory()

ทำการแสดงรายชื่อของฐานข้อมูลที่สามารถนำมาใช้ในระบบได้ โดยจะแสดงอยู่ภายใน directory window ซึ่งขนาดของ window จะแปรเปลี่ยนตามจำนวนรายชื่อของฐานข้อมูล รายชื่อของฐานข้อมูลทั้งหมดจะเก็บไว้แฟ้มข้อมูลชื่อ /grad/g292403/DML/prog/.dir

```
Print_to_file(ptr)
RELNODE   *ptr;          /* Relation view pointer */
```

ทำการเขียน (write) รายละเอียดของรีเลชั่นที่ถูกชี้โดย ptr ลงในแฟ้มข้อมูลซึ่งในโปรแกรมจะทำการเขียนลงในแฟ้มข้อมูลชื่อ /grad/g292403/DML/prog/.Out เสมอ ฟังก์ชั่นนี้จะมีการทำงานก็ต่อเมื่อค่าของตัวแปร SET_PRINT มีค่าเป็น ON

```
display(ptr)
RELNODE   *ptr;          /* Relation view pointer */
```

ทำการแสดงรายละเอียดของรีเลชั่นทางจอภาพ

Exit_to_Shell()

เป็นฟังก์ชั่นที่จะหยุดการทำงานของโปรแกรมไว้ชั่วคราว เมื่อมีความต้องการที่จะใช้คำสั่ง shell การกลับเข้ามาทำงานในโปรแกรมต่อจะใช้คำสั่ง Ctrl D (^D)

```
Free_Attribute(Aptr)
ATTNODE   *Aptr;          /* Attribute pointer */
```

ทำการกำหนดให้เนื้อที่ในหน่วยความจำที่ถูกชี้โดยค่าของ Aptr ว่างลง เพื่อคืนเนื้อที่ในส่วนนี้ให้กับหน่วยความจำหลัก

```
Free_Relation(Ptr)
RELNODE   *Ptr;           /* Relation view pointer */
```

ทำการกำหนดให้เนื้อที่ในหน่วยความจำที่ถูกชี้โดยค่าของ Ptr ว่างลง เพื่อให้ฟังก์ชั่นอื่นสามารถนำไปใช้งานได้

```
Free_Tree(TPtr)
TRENODE   *TPtr;          /* Binary tree pointer */
```

เป็นการกำหนดให้เนื้อที่ในหน่วยความจำที่ถูกชี้โดยค่าของ TPtr ว่างลง เพื่อให้ฟังก์ชั่นอื่นสามารถนำไปใช้งานได้

```
GphBox(win)
WINDOW    *win;           /* Window pointer */
```

ทำการวาดกรอบของ window ซึ่งกำหนดขนาดจากตัวแปร win ในฟังก์ชั่นนี้การวาดกรอบจะขึ้นอยู่กับชนิดของจอภาพที่กำลังใช้งานด้วย โดยถ้าเป็นจอภาพของเครื่อง Microcomputer ก็จะวาดกรอบด้วย Text character แต่ถ้าเป็นจอภาพแบบ VT ก็จะวาดกรอบด้วย Graphic character

```
VTbox(win)
WINDOW    *win;          /* Window pointer */
```

เป็นฟังก์ชั่นการวาดกรอบของ window ด้วย graphic character

```
Help_Item(StrCmd)
char      StrCmd[];      /* String command */
```

ทำการแสดงไวยกรณ์ (syntax) ของคำสั่งปฏิบัติการต่างๆ โดยขึ้นกับค่าของ StrCmd

```
setup_win()
```

ทำการกำหนดโครงสร้างของ window หลักซึ่งได้แก่ Menu Line Area MsgL และ MsgA

```
Displaymenu()
```

ทำการแสดงรายการหลักทั้งหมดใน window menu

```
Open_Database()
```

ทำการรับชื่อฐานข้อมูล เพื่อนำเข้าสู่ระบบ

```
Operate_depend_on_selection()
```

ทำการเรียกใช้ฟังก์ชั่นการทำงานของคำสั่งปฏิบัติการต่างๆ โดยขึ้นกับค่า OP_CODE ที่มีการกำหนดค่ามาจากฟังก์ชั่นนั้นๆ

Save_Relation()

ทำการนำรีเลชั่นแบบ temporary ที่ถูกชี้โดยพอยเตอร์ RsltPtr มาเพิ่มไว้ในฐานข้อมูลที่อยู่ในระบบขณะนั้น

Set_Option()

ทำการเปลี่ยนสถานะของตัวแปร SET_PRINT และ DUPLICATE ให้มีค่าเป็น ON หรือ OFF

```
start_up(name)
char        *name;          /* Database name */
```

ทำการค้นหาชื่อของฐานข้อมูลซึ่งกำหนดโดยค่าของตัวแปร name เพื่อนำฐานข้อมูลนั้นเข้าสู่ระบบ

```
RELNODE    *Create_RelList(Rfp)
FILE       *Rfp;           /* Relation file pointer */
```

ทำการอ่านแฟ้มข้อมูลที่ถูกชี้โดยค่าของ Rfp เพื่อทำการสร้าง linked list ของรีเลชั่น

```
ATTNODE    *Create_AttList(RKey,Enf_KeyPos)
char       RKey[LNGTH+1];      /* Value of relation key */
int        *End_KeyPos;        /* Return key length */
```

ทำการสร้าง linked list ของ attribute โดยรับค่าคีย์ของรีเลชั่นมากับตัวแปร RKey เพื่อนำไปค้นหาในแฟ้มข้อมูล ฟังก์ชั่นนี้จะส่งค่าของ End_KeyPos กลับเพื่อเก็บไว้ใช้ในการสร้าง Binary search tree ต่อไป

```
TUPNODE   *Create_DatList(RName,TSize)
char      RName[RNAME+1];     /* Relation name */
int       TSize;              /* Size of tuple for allocate */
```

ทำการสร้าง linked list ของ tuple โดยโปรแกรมจะทำการอ่านข้อมูลจากแฟ้มข้อมูลชื่อ RName.dat

```
TRENODE   *Create_Tree(TuPtr,Lngth)
TUPNODE   *TuPtr;         /* Tuple pointer */
int       Lngth;          /* Length of key */
```

ทำการสร้าง binary search tree จากข้อมูลที่ถูกชี้โดยตัวแปร TuPtr ซึ่ง index แต่ละค่าจะมีขนาดเท่ากับ lngth

```
PrintAllStructure(HPtr)
RELNODE   *HPtr;          /* System pointer */
```

ทำการแสดงรายละเอียดเกี่ยวกับฐานข้อมูลและรีเลชั่นที่อยู่ในระบบขณะนั้น

Terminal_Type()

ทำการรับชนิดของจอภาพที่ใช้งาน เพื่อกำหนดค่าให้กับตัวแปร TERM_ TYPE ซึ่งชนิดของจอภาพจะมีอยู่ 2 ชนิด คือ VT และ MICRO

Compatible(ptr1,ptr2)

```
RELNODE   *ptr1;          /* Relation view pointer #1 */
RELNODE   *ptr2;          /* Relation view pointer #2 */
```

ทำการเปรียบเทียบโครงสร้างทั้งหมดของรีเลชั่นที่ถูกชี้โดยตัวแปร ptr1 และ ptr2 ว่ามีโครงสร้างและส่วนประกอบที่เหมือนกันหรือไม่

Check_Setup()

ทำการตรวจสอบสถานะของตัวแปร SETUP ว่ามีค่าเป็น ON หรือ OFF

Clear_Area()

ทำการลบจอภาพในส่วนของ window area

Clear_Result()

ทำการลบค่าต่างๆ ที่ถูกชี้โดยตัวแปร RsltPtr ซึ่งผลที่ได้จะทำให้ภายในระบบถือว่าไม่มี temporary relation

display_head(win,lngth,str)

```
WINDOW    *win;           /* Window for display heading */
```

```
int     lngth;          /* Size of heading line */
char    *str;           /* String heading */
```

ทำการแสดงชื่อของ window ที่ถูกชี้โดยตัวแปร win ซึ่งข้อความที่จะนำมาแสดงจะเก็บไว้ในตัวแปร str โดยมีขนาดของบรรทัดที่จะทำการแสดงเท่ากับ lngth

Display_Infor()

ทำการแสดงรายละเอียดของฐานข้อมูลที่อยู่ในระบบขณะนั้น

Display_msg()

ทำการแสดงข้อความที่บอกความผิดพลาดในการทำงาน หรือข้อความแนะนำการทำงานในขั้นตอนต่อไป ใน MsgA window

```
RELNODE   *Search_relation(ptr,name)
RELNODE   *ptr;           /* Relation linked list */
char      *name;          /* Relation name for search */
```

ทำการค้นหารีเลชั่นที่ถูกกำหนดค่าไว้ในตัวแปร name จาก relation linked list ที่ถูกชี้โดยตัวแปร ptr

TupleCnt(ptr)

```
TUPNODE   *ptr;           /* Tuple pointer */
```

ทำการนับจำนวนของ tuple ที่อยู่ใน tuple linked list ซึ่งถูกชี้โดยตัวแปร ptr

```
Count(Rptr)
RELNODE   *Rptr;          /* Relation view pointer */
```

ทำการนับจำนวนของ tuple ในรีเลชั่นที่ถูกที่โดยพอยเตอร์ Rptr

```
Maximum(Aptr,Tptr)
ATTNODE   *Aptr;          /* Attribute pointer */
TUPNODE   *Tptr;          /* Tuple pointer */
```

ทำการหาค่าสูงสุดของ attribute ที่ถูกชี้โดยตัวแปร Aptr ค่าที่ได้จะมาจาก tuple ที่ถูกชี้โดยตัวแปร Tptr

```
Minimum(Aptr,Tptr)
ATTNODE   *Aptr;          /* Attribute pointer */
TUPNODE   *Tptr;          /* Tuple pointer */
```

ทำการหาค่าต่ำสุดของ attribute ที่ถูกชี้โดยตัวแปร Aptr ค่าที่ได้จะมาจาก tuple ที่ถูกชี้โดยตัวแปร Tptr

```
Sum(Aptr,Tptr)
ATTNODE   *Aptr;          /* Attribute pointer */
TUPNODE   *Tptr;          /* Tuple pointer */
```

ทำการหาผลรวมของ attribute ที่ถูกชี้โดยตัวแปร Aptr ค่าที่ได้จะมาจาก tuple ที่ถูกชี้โดยตัวแปร Tptr

```
Average(Aptr,Tptr)
ATTNODE  *Aptr;          /* Attribute pointer */
TUPNODE   *Tptr;          /* Tuple pointer */
```

ทำการหาค่าเฉลี่ยของค่า attribute ที่ถูกชี้โดยตัวแปร Aptr ค่าที่ได้จะมาจาก tuple ที่ถูกชี้โดยตัวแปร Tptr

```
Wait()
```

สำหรับรอให้ผู้ใช้กดแป้นพิมพ์ใดๆ เพื่อทำงานต่อไป

```
Up_Case(str)
char      str[];          /* Input/Output string */
```

ทำการเปลี่ยนข้อความในตัวแปร str ที่มีตัวอักษรเล็กปนอยู่ ให้เป็นตัวอักษรตัวใหญ่

3.4.5. ฟังก์ชั่นคำสั่งปฏิบัติการแบบทีชคณิตสัมพันธ์

```
Intersection(R1_ptr,R2_ptr,rslt_name)
RELNODE   *R1_ptr;        /* First relation view pointer */
RELNODE   *R2_ptr;        /* Second relation view pointer */
char      *rslt_name;     /* Result relation name */
```

ทำปฏิบัติการกับข้อมูลด้วยคำสั่ง intersect โดยที่รีเลชั่นทั้งสองถูกชี้ด้วยพอยเตอร์ R1_ptr และ R2_ptr ตามลำดับ ชื่อของรีเลชั่นที่เก็บผลลัพท์จะถูกกำหนดโดยตัวแปร rslt_name โครงสร้างของรีเลชั่นซึ่งเก็บผลลัพท์จะถูกชี้ด้วยตัวแปรชื่อ RsltPtr

```
Union(R1_ptr,R2_ptr,rslt_name)
RELNODE   *R1_ptr;        /* First relation view pointer */
RELNODE   *R2_ptr;        /* Second relation view pointer */
char      *rslt_name;     /* Result relation name */
```

ทำปฏิบัติการกับข้อมูลด้วยคำสั่ง union โดยที่รีเลชั่นทั้งสองถูกชี้ด้วยพอยเตอร์ R1_ptr และ R2_ptr ตามลำดับ ชื่อของรีเลชั่นที่เก็บผลลัพท์จะถูกกำหนดโดยตัวแปร rslt_name โครงสร้างของรีเลชั่นซึ่งเก็บผลลัพท์จะถูกชี้ด้วยตัวแปรชื่อ RsltPtr

```
Difference(R1_ptr,R2_ptr,rslt_name)
RELNODE   *R1_ptr;        /* First relation view pointer */
RELNODE   *R2_ptr;        /* Second relation view pointer */
char      *rslt_name;     /* Result relation name */
```

ทำปฏิบัติการกับข้อมูลด้วยคำสั่ง difference โดยที่รีเลชั่นทั้งสองถูกชี้ด้วยพอยเตอร์ R1_ptr และ R2_ptr ตามลำดับ ชื่อของรีเลชั่นที่เก็บผลลัพท์จะถูกกำหนดโดยตัวแปร rslt_name โครงสร้างของรีเลชั่นซึ่งเก็บผลลัพท์จะถูกชี้ด้วยตัวแปรชื่อ RsltPtr

```
Cross_Product(R1_ptr,R2_ptr,rslt_name)
RELNODE   *R1_ptr;        /* First relation view pointer */
RELNODE   *R2_ptr;        /* Second relation view pointer */
char      *rslt_name;     /* Result relation name */
```

ทำปฏิบัติการกับข้อมูลด้วยคำสั่ง cross product โดยที่รีเลชั่นทั้งสองถูกชี้ด้วยพอยเตอร์ R1_ptr และ R2_ptr ตามลำดับ ชื่อของรีเลชั่นที่เก็บผลลัพท์จะถูกกำหนด

โดยตัวแปร rslt_name โครงสร้างของรีเลชั่นซึ่งเก็บผลลัพท์จะถูกชี้ด้วยตัวแปรชื่อ RsltPtr

```
Select(S_ptr,S_apt,S_op,S_val,S_rslt)
RELNODE   *S_ptr;        /* Relation view pointer to select */
ATTNODE   *S_apt;        /* Attribute to select */
int       S_op;          /* Select on what condition */
char      S_val[];       /* Value to select */
char      S_rslt[];      /* Result relation name */
```

ทำปฏิบัติการกับข้อมูลด้วยคำสั่ง select โดยที่ตัวแปรพอยเตอร์ S_ptr จะชี้ไปยังรีเลชั่นและ พอยเตอร์ S_apt จะชี้ที่ attribute ที่จะทำการ select เงื่อนไขจะถูกกำหนดโดยค่าของ S_op, ค่าของ attribute ที่ต้องการจะถูกกำหนดโดยตัวแปร S_val และ ชื่อของรีเลชั่นที่เก็บผลลัพท์จะถูกกำหนดโดยตัวแปร S_rslt โครงสร้างของรีเลชั่นซึ่งเก็บผลลัพท์จะถูกชี้ด้วยตัวแปรชื่อ RsltPtr

```
Project(Ptr,Att,result_name,NoAtt)
RELNODE   *Ptr;                 /* Relation view pointer to project */
char      Att[MAXATT][ANAME+1];  /* Attribute name list to project */
char      result_name;          /* Result relation name */
int       NoAtt;                /* Number of attribute to project */
```

ทำปฏิบัติการกับข้อมูลด้วยคำสั่ง project โดยที่พอยเตอร์ Ptr จะชี้ไปยังรีเลชั่น อะเรย์ Att จะเก็บรายชื่อของ attribute ที่จะทำการ projectและตัวแปร NoAtt

จะเก็บจำนวนของ attribute ชื่อของรีเลชั่นที่เก็บผลลัพท์จะถูกกำหนดโดยตัวแปร result_name โครงสร้างของรีเลชั่นซึ่งเก็บผลลัพท์จะถูกชี้ด้วยตัวแปรชื่อ RsltPtr

```
Join(R1,R2,Aname1,Aname2,rslt,J_cond)

RELNODE   *R1;              /* First relation view pointer  */
RELNODE   *R2;              /* Second relation view pointer */
char      Aname1;           /* Attribute name of 1st relation */
char      Aname2;           /* Attribute name of 2nd relation */
char      rslt;             /* Result relation name */
int       J_cond;           /* Condition for join */
```

ทำปฏิบัติการกับข้อมูลด้วยคำสั่ง Join โดยมีตัวแปรพอยเตอร์ R1 และ R2 ชี้ไปยังรีเลชั่นทั้งสองที่ต้องการนำข้อมูลมาเชื่อมกัน ส่วนตัวแปร Aname1 จะเก็บชื่อของ attribute ในรีเลชั่นแรกและตัวแปร Aname2 จะเก็บชื่อของ attribute ในรีเลชั่นที่สอง เงื่อนไขในการ Join กำหนดโดยค่าของตัวแปร J_cond ชื่อของรีเลชั่นที่เก็บผลลัพท์จะถูกกำหนดโดยตัวแปร rslt โครงสร้างของรีเลชั่นซึ่งเก็บผลลัพท์จะถูกชี้ด้วยตัวแปรชื่อ RsltPtr

```
Divide(R1,R2,rslt)

RELNODE   *R1;              /* First relation view pointer */
RELNODE   *R2;              /* Second relation view pointer */
char      *rslt;            /* Result relation name */
```

ทำปฏิบัติการกับข้อมูลด้วยคำสั่ง Divide โดยที่รีเลชั่นทั้งสองถูกชี้ด้วยพอย เตอร์ R1 และ R2 ตามลำดับ ชื่อของรีเลชั่นที่เก็บผลลัพท์จะถูกกำหนดโดยตัวแปร rslt โครงสร้างของรีเลชั่นซึ่งเก็บผลลัพท์จะถูกชี้ด้วยตัวแปรชื่อ RsltPtr

บทที่ 4

เอกสารสำหรับผู้ใช้

4.1 สิ่งที่ผู้ใช้ควรทราบก่อนใช้งาน

1. สำหรับผลลัพท์ที่ผู้ใช้ต้องการเก็บลงแฟ้มข้อมูล โปรแกรมจะนำผลลัพท์เก็บไว้ในแฟ้มข้อมูลที่ชื่อ '.Out' ต่อท้าย ซึ่งทุกๆครั้งที่มีการเรียกใช้โปรแกรม โปรแกรมจะทำการลบข้อมูลที่มีอยู่เดิมในแฟ้มข้อมูลนี้เสมอ

2. ในกรณีที่ผู้ใช้ทำการหยุดการทำงานของโปรแกรมด้วย Terminate control code เช่น Ctrl-C (^C) หรือ Ctrl-Z (^Z) โปรแกรมจะหยุดการทำงานทันทีและจะมีผลกระทบต่อการทำงานของจอภาพเมื่อผู้ใช้ต้องการทำงานใน shell การแก้ปัญหาในส่วนนี้ คือ ให้ผู้ใช้พิมพ์คำสั่ง reset แล้วจอภาพจึงจะกลับสู่สภาพการทำงานปกติ

```
psuvax > reset
```

3. เมื่อเริ่มใช้งานโปรแกรมจะทำการถามผู้ใช้ถึงชนิดของเครื่องที่ใช้งาน โดยจะแสดงข้อความ

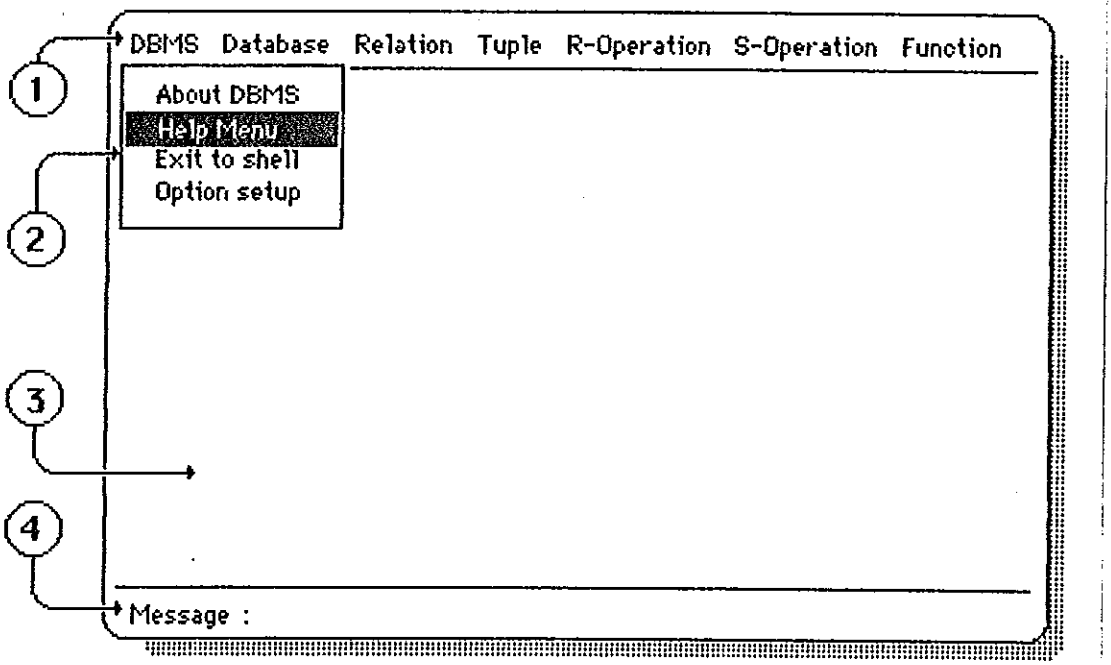
```
What a terminal type ?

VT terminal

Micro computer

select []
```

แล้วให้ผู้ใช้ระบุโดยกดอักษร 'V' หรือ 'M' เพื่อทำการปรับจอภาพสำหรับการแสดงผลที่ได้จากการทำงาน

4.2 องค์ประกอบหลักของจอภาพสำหรับการใช้งาน

การทำงานของโปรแกรมทั้งหมดจะเป็น "การจัดการแบบหน้าต่าง (window manipulation)" ซึ่งจะอำนวยความสะดวกในการแสดงผลมาก เนื่องจากสามารถที่จะมีปริมาณการแสดงผลได้มากกว่าการใช้การแสดงผลในจอภาพเดียว

ลักษณะของการแสดงผลทางจอภาพจะประกอบด้วยส่วนประกอบหลัก ดังต่อไปนี้



รูปที่ 4-1 องค์ประกอบหลักของจอภาพในการใช้งาน

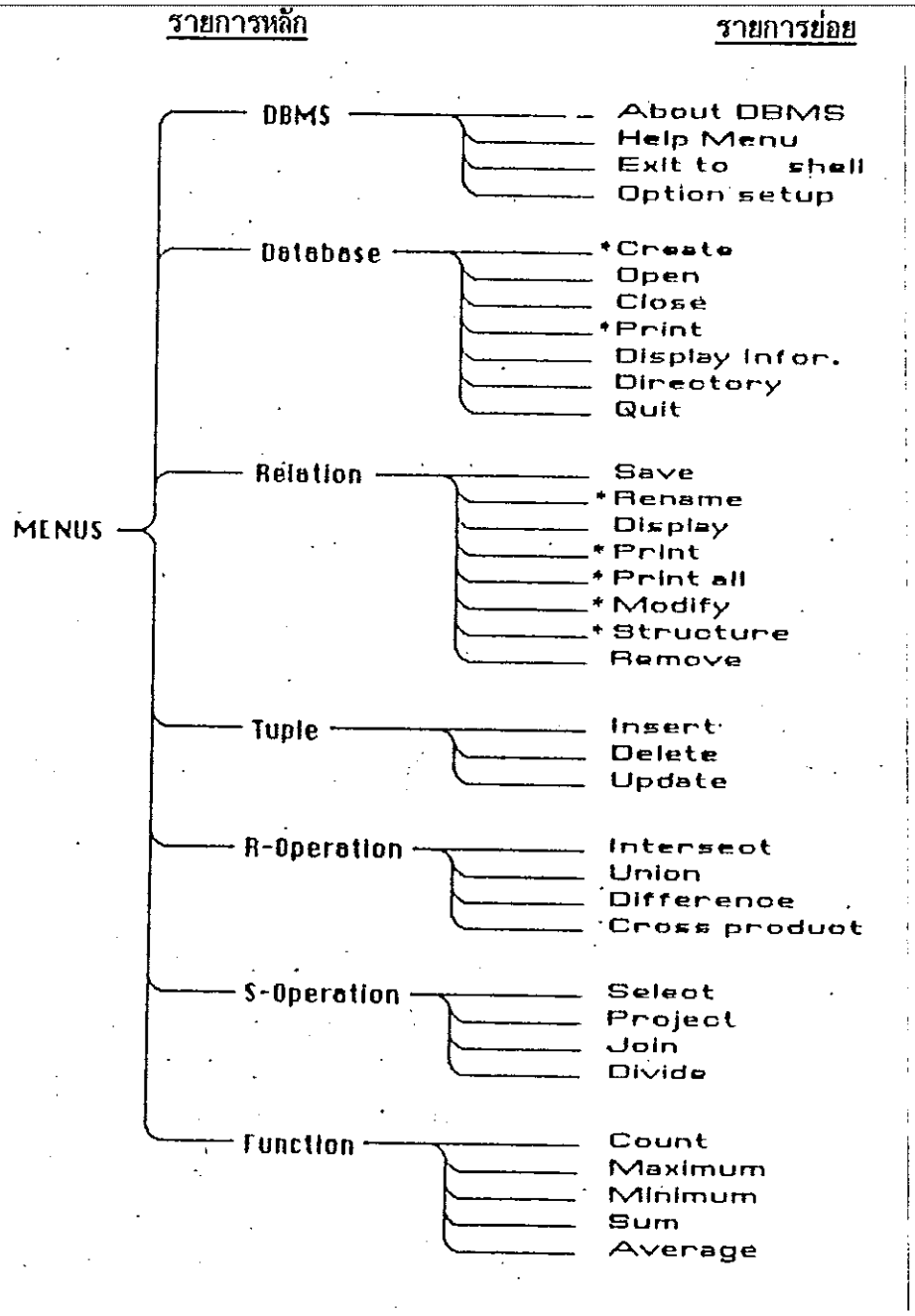
ส่วนที่ 1 เป็นส่วนของรายการหลัก (Main Menu) ซึ่งมีทั้งหมด 7 รายการ ซึ่งจะได้กล่าวถึงรายละเอียดในการใช้ต่อไป ในภายหลัง

ส่วนที่ 2 เป็นส่วนของรายการย่อย (Sub menu) ซึ่งจะอยู่ภายใต้รายการหลัก

ส่วนที่ 3 เป็นส่วนของบริเวณการแสดงผล (Display area) ซึ่งจะใช้สำหรับการแสดงผลที่ได้จากการใช้คำสั่งปฏิบัติการฐานแบบต่างๆ

ส่วนที่ 4 เป็นเนื้อที่สำหรับการแสดงข้อความของความผิดพลาด (Error message) เพื่อให้ผู้ใช้ได้ทราบถึงข้อผิดพลาดที่เกิดขึ้นในการทำงาน หรือใช้แสดงข้อความเพื่อแนะนำ (Guide message) ผู้ใช้ในการทำงานขั้นต่อไป

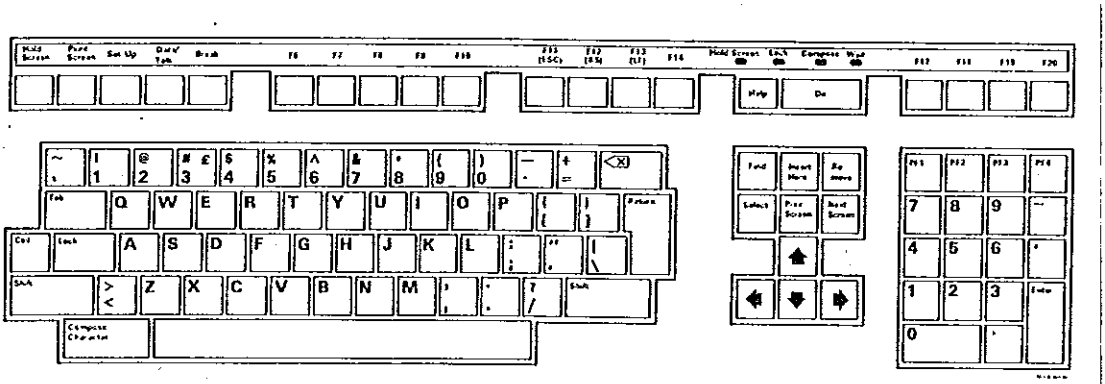
4.3 แผนภาพสรุปโครงสร้างของรายการหลักและรายการย่อย



รูปที่ 4-2 แผนภาพสรุปรายการหลักและรายการย่อย

4.4 ข้อแนะนำการใช้แป้นพิมพ์สำหรับควบคุมการทำงานต่างๆ ของโปรแกรม

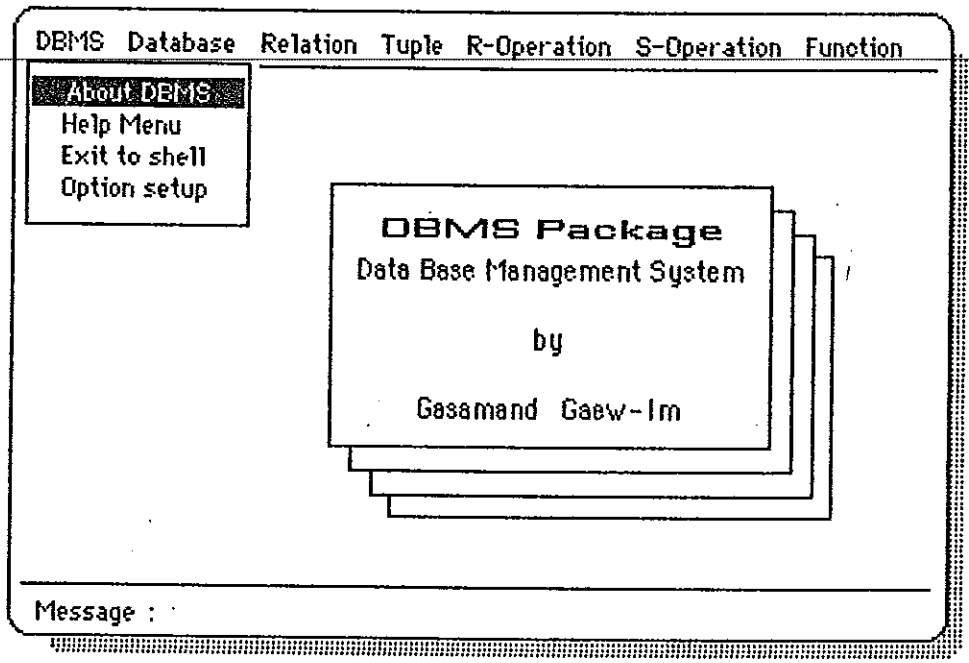
แป้นพิมพ์	การทำงาน
l หรือ L หรือ →	ใช้เลือกรายการหลักในแนวราบไปทางขวา
h หรือ H หรือ ←	ใช้เลือกรายการหลักในแนวราบไปทางซ้าย
j หรือ J หรือ ↓	ใช้เลือกรายการย่อยลงในแนวระดับ
k หรือ K หรือ ↑	ใช้เลือกรายการย่อยขึ้นในแนวระดับ
Return หรือ Enter	ใช้ตอบตกลงเมื่อเลือกการทำงานนั้น
Esc	ใช้ยกเลิกการทำงานบางอย่างทันที
Space bar	ใช้เปลี่ยนแปลงสถานะของการทำงาน หรือในบางกรณีก็จะใช้เลือกค่า ในการทำงานของคำสั่งปฏิบัติการฐานข้อมูลแบบ projection



รูปที่ 4.3 แป้นพิมพ์สำหรับควบคุมการทำงาน

4.5 การใช้โปรแกรมในรายการหลักและรายการย่อย

4.5.1. รายการหลัก DBMS



รูปที่ 4.4 จอภาพรายการหลัก DBMS

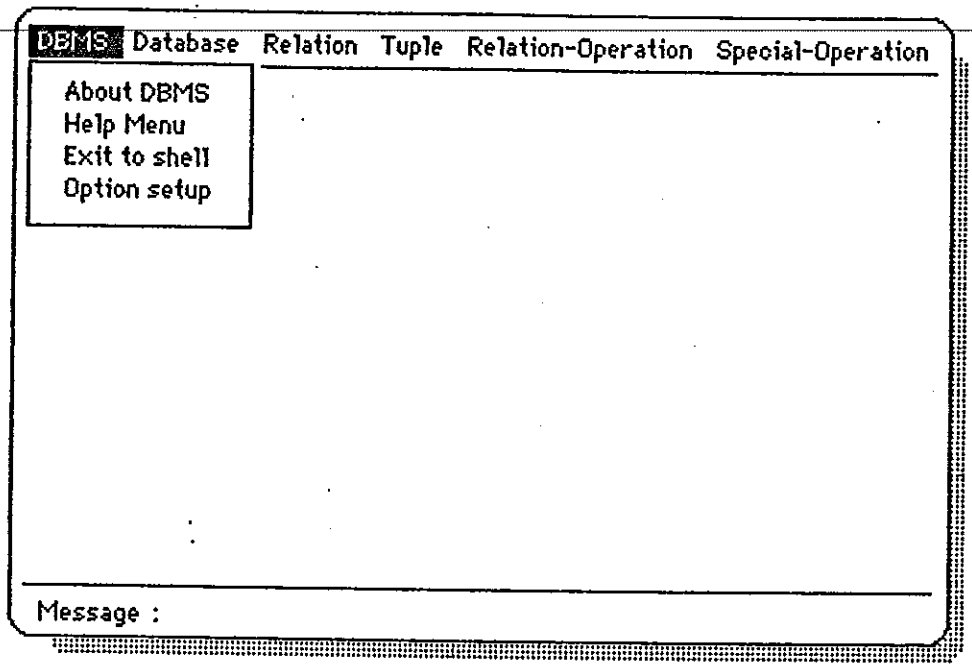
ในรายการหลัก DBMS การทำงานส่วนใหญ่ของโปรแกรมเป็นการทำงานที่เกี่ยวข้องกับงานทั่วไป รายการหลักนี้ประกอบไปด้วยรายการย่อย 4 รายการ คือ

- About DBMS
- Help Menu
- Exit to shell
- Option setup

ชื่อรายการ About DBMS

การทำงาน แสดงชื่อโปรแกรมระบบจัดการฐานข้อมูล

จอภาพขณะใช้งาน



รูปที่ 4-5 จอภาพขณะใช้งานของรายการย่อย About DBMS

ข้อจำกัดและคำแนะนำเพิ่มเติม

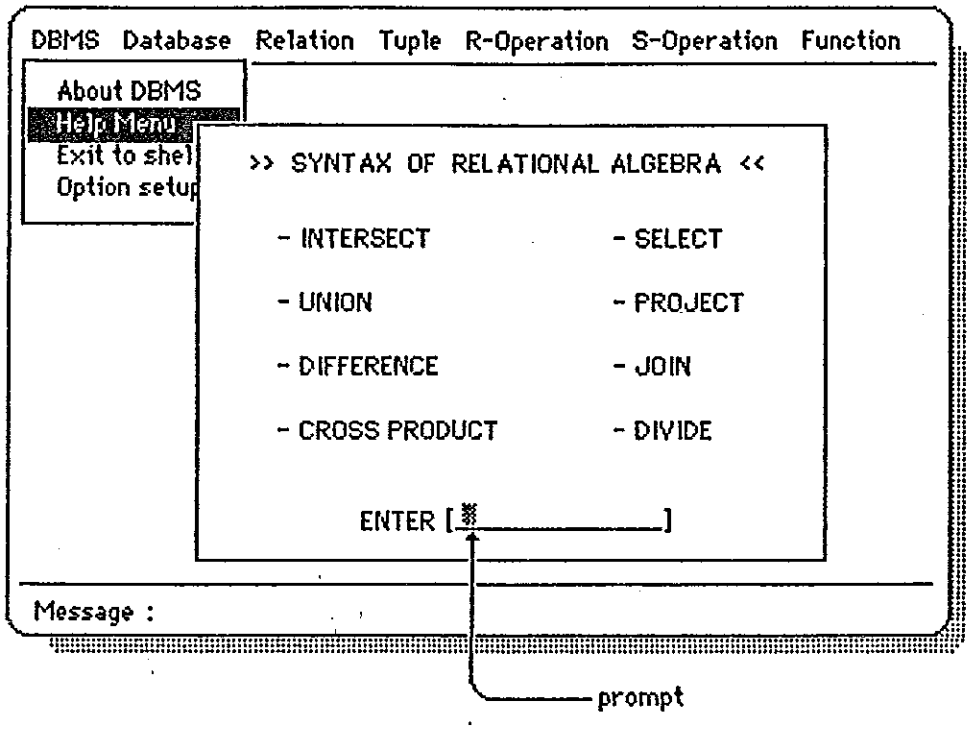
โปรแกรมการทำงานในส่วนนี้มีการหน่วงเวลาประมาณ 4 วินาที แล้วจะกลับเข้าสู่การทำงานหลักเอง

ชื่อรายการ Help Menu

การทำงาน แสดงไวยกรณ์ของคำสั่งปฏิบัติการฐานข้อมูลแบบต่างๆ ที่มีอยู่

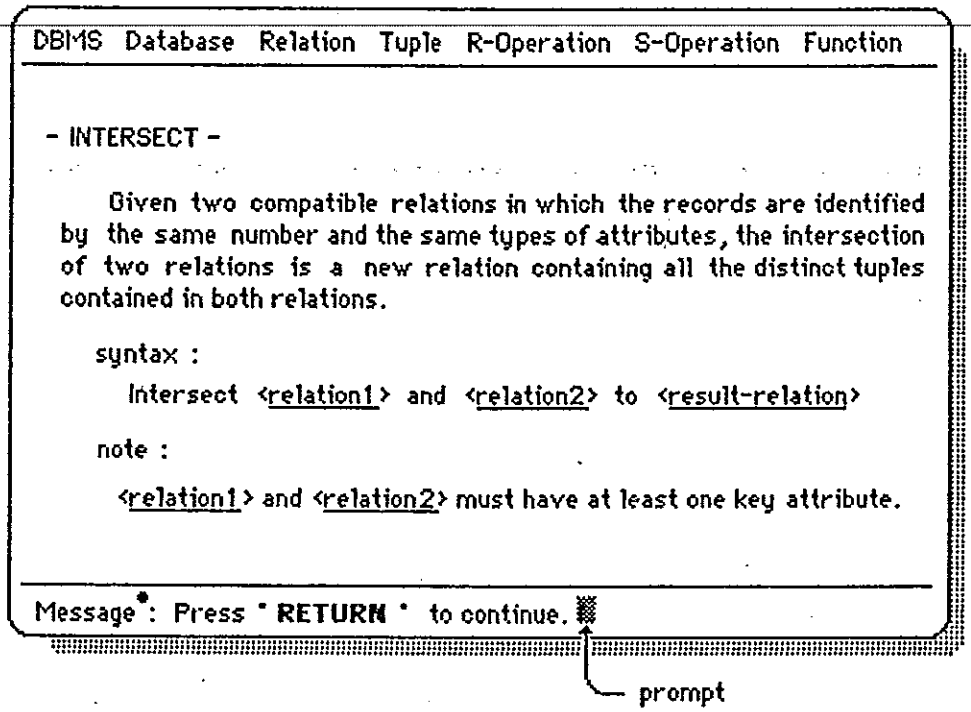
การใช้งาน โปรแกรมจะแสดงหน้าต่าง (window) ซึ่งบรรจุคำสั่งปฏิบัติการฐานข้อมูลแบบพีชคณิตสัมพันธ์ทั้ง 8 คำสั่ง (ดังรูป 4-6) เมื่อผู้ใช้ต้องการทราบรูปแบบไวยากรณ์ของคำสั่งใดก็พิมพ์คำสั่งนั้น ในส่วนของ ENTER [_____] โปรแกรมก็จะแสดงรูปแบบของไวยกรณ์นั้นออกมาผู้ใช้จะกลับสู่การทำงานหลักได้โดยกดอักษรใด ก็ได้บนแป้นพิมพ์

จอภาพขณะใช้งาน



รูปที่ 4-6 จอภาพขณะใช้งานของรายการย่อย Help Menu

ถ้าคำสั่งที่พิมพ์ใน ENTER [_____] เป็น intersection ก็จะได้จอภาพดังแสดงในรูปที่ 4-7



รูปที่ 4-7 จอภาพขณะใช้งานของรายการย่อย Help Menu

ข้อจำกัดและคำแนะนำเพิ่มเติม

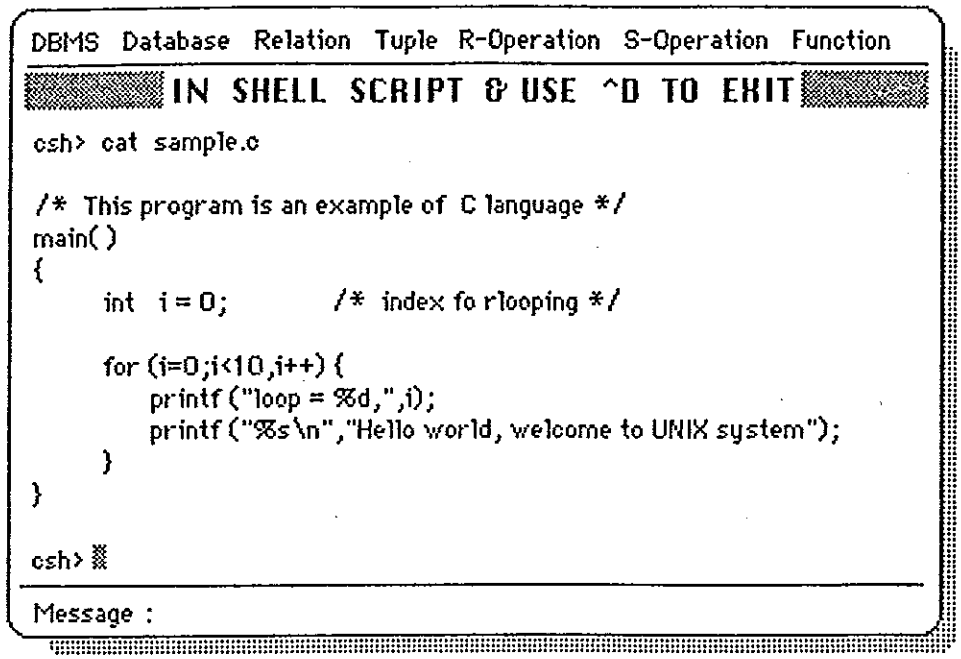
1. การรับข้อมูลสามารถใช้ได้ทั้งอักษรตัวเล็กและตัวใหญ่
2. ในกรณีที่ผู้ใช้พิมพ์ข้อความที่ไม่ได้ปรากฏอยู่ใน Help Menu โปรแกรมจะแสดง ข้อความแสดงการทำงานผิดพลาด และกลับเข้าสู่การทำงานหลัก

ชื่อรายการ Exit to shell

การทำงาน โปรแกรมจะหยุดทำงานชั่วคราวเพื่อให้ผู้ใช้สามารถใช้คำสั่งต่างๆ ใน shell ของ UNIX ได้ และเมื่อเสร็จการทำงานใน shell ผู้ใช้ก็สามารถที่จะกลับเข้าไปทำงานกับโปรแกรมต่อไปได้

การใช้งาน ผู้ใช้สามารถจะใช้คำสั่งในของ UNIX ได้ทุกคำสั่ง และจะกลับเข้าทำงานกับโปรแกรมต่อไปได้ โดยใช้คำสั่ง ^D

จอภาพขณะใช้งาน



รูปที่ 4-8 จอภาพขณะใช้งานของรายการย่อย Exit to shell

ข้อจำกัดและคำแนะนำเพิ่มเติม

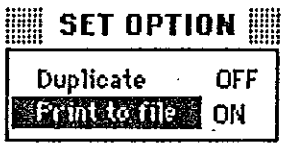
คำสั่งของ UNIX บางคำสั่งอาจจะมีผลทำให้รูปแบบของจอภาพเปลี่ยนไป เช่น คำสั่ง clear

ชื่อรายการ Option setup

การทำงาน ใช้เปลี่ยนแปลงสถานะของตัวแปรที่ใช้ควบคุมการทำงานบางอย่างภายในโปรแกรม ซึ่งได้แก่

- การเปลี่ยนสถานะเกี่ยวกับการตัดข้อมูลซ้ำ
- การเปลี่ยนสถานะการแสดงผล ให้สามารถนำผลลัพธ์เก็บไว้ในแฟ้มข้อมูลสำหรับนำออกเครื่องพิมพ์ต่อไป

การใช้งาน โปรแกรมจะแสดงหน้าต่าง set option (ดังรูป)

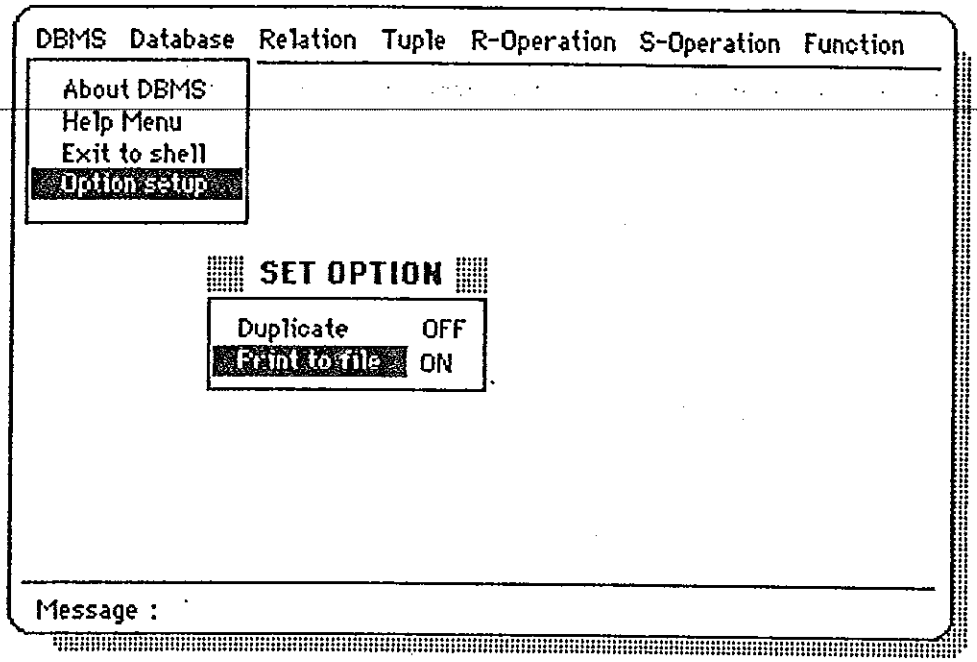


ซึ่งประกอบด้วย สถานะ 2 สถานะที่ผู้ใช้สามารถเปลี่ยนแปลงค่าได้ คือ

1. Duplicate เป็นการอนุญาตให้ผู้ใช้กำหนดได้ว่าผลที่จะได้จากการดำเนินงานนั้นต้องการให้ตัดค่าที่ซ้ำออกหรือไม่
2. Print to file เป็นการอนุญาตให้ผู้ใช้กำหนดได้ว่าผลที่จะได้จากการดำเนินงานต้องการเก็บไว้ในแฟ้มข้อมูลเพื่อพิมพ์หรือไม่

ทุกครั้งที่โปรแกรมเริ่มใช้งาน สถานะทั้ง 2 นี้จะถูกกำหนดค่าให้เป็น "OFF" การเปลี่ยนแปลงสถานะขณะที่โปรแกรมดำเนินงานจะขึ้นกับผู้ใช้ ในการเปลี่ยนสถานะ(ON/OFF)ใช้แป้นพิมพ์ space bar การเลือกจะใช้แป้นพิมพ์ ↑ ↓ j J k K และใช้แป้นพิมพ์ Return เพื่อรับสถานะที่มีการเปลี่ยนแปลงใหม่

จอภาพขณะใช้งาน

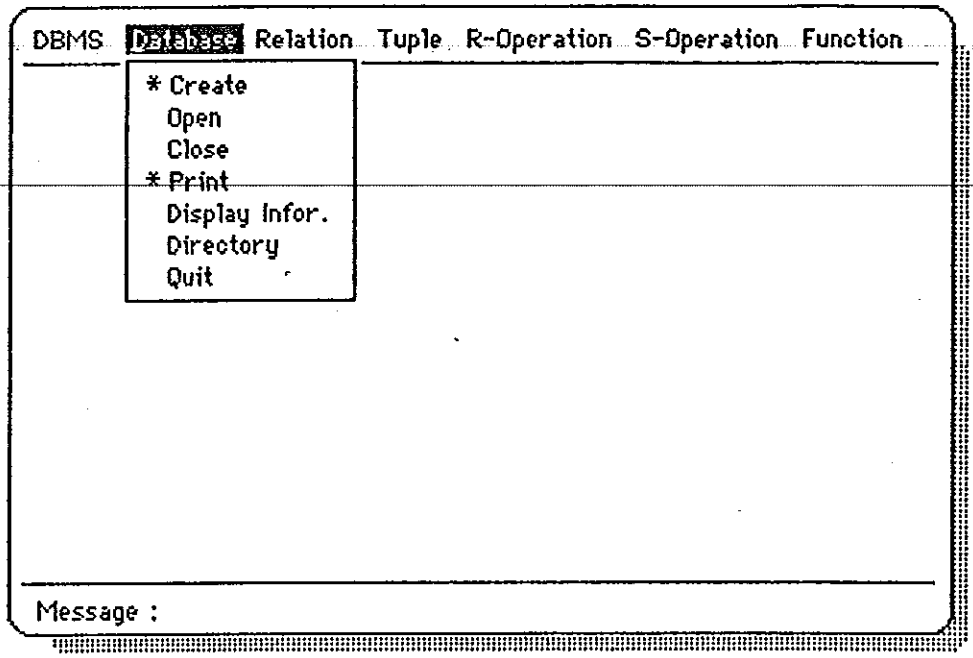


รูปที่ 4-9 จอภาพขณะใช้งานของรายการย่อย Option setup

ข้อจำกัดและคำแนะนำเพิ่มเติม

1. การเปลี่ยนสถานะ สามารถที่จะทำขณะใดขณะหนึ่งที่โปรแกรมทำงานอยู่ก็ได้
2. การเปลี่ยนสถานะของ duplicate จะมีผลต่อการทำงานของคำสั่ง Project

4.5.2 รายการหลัก Database



รูปที่ 4-10 จอภาพรายการหลัก Database

รายการหลักส่วนนี้จะทำงานเกี่ยวข้องกับฐานข้อมูล เช่น การนำฐานข้อมูลเข้า/ออกในระบบ การแสดงรายชื่อของฐานข้อมูลที่สามารถนำมาใช้ในระบบได้ การแสดงรายละเอียดของฐานข้อมูลที่อยู่ในระบบขณะดำเนินงานว่าประกอบไปด้วย relation ใดบ้าง เป็นต้น

ในรายการหลักส่วนนี้ประกอบไปด้วยรายการย่อยจำนวน 7 รายการ ดังนี้คือ

- Create*
- Display Infor.
- Open
- Directory
- Close
- Print*
- Quit

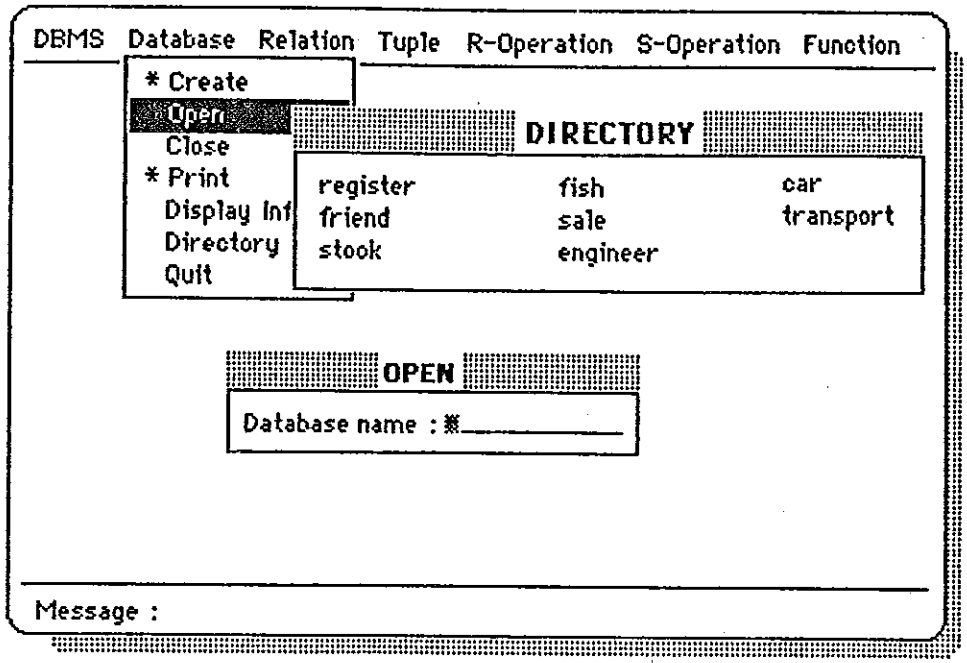
* เป็นการทำงานส่วนที่ยังไม่ได้ทำการพัฒนาโปรแกรม

ชื่อรายการ Open

การทำงาน นำฐานข้อมูลที่ผู้ใช้กำหนดมาไว้ในระบบ

การใช้งาน สามารถแบ่งออกได้เป็น 2 ขั้นตอน ดังนี้คือ

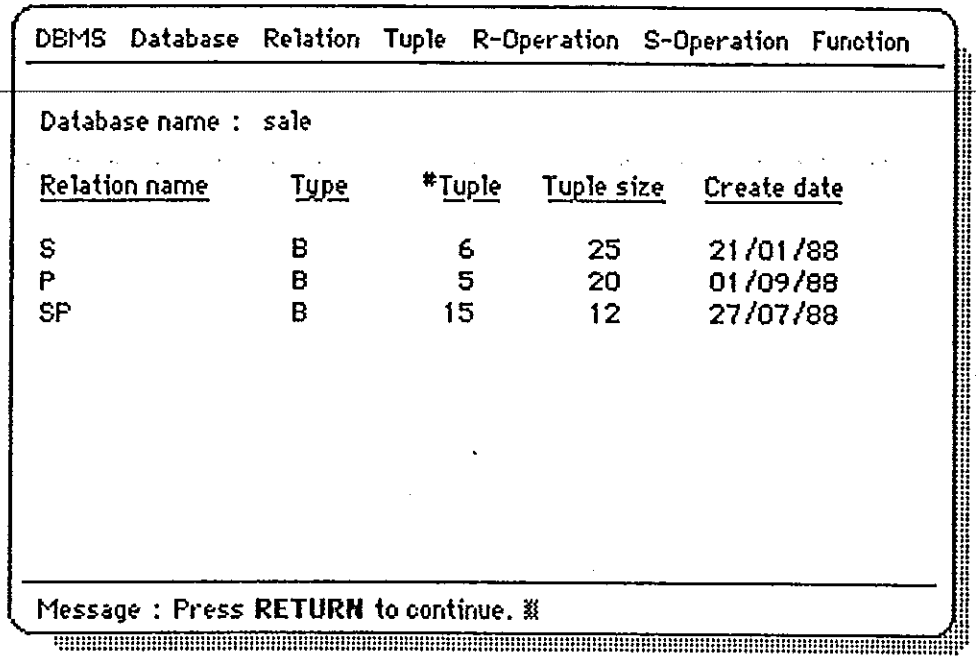
1. รับข้อมูล ซึ่งจะหมายถึงการให้ผู้ใช้ป้อนชื่อฐานข้อมูลที่ต้องการนำเข้ามาในระบบ โปรแกรมจะแสดงหน้าต่าง Directory เพื่อให้ผู้ใช้ทราบว่ามีฐานข้อมูลใดบ้าง ที่สามารถนำเข้ามาในระบบได้ หลังจากนั้นโปรแกรมจะแสดงหน้าต่าง Open (ดูรูปที่ 4-11ประกอบ) เพื่อให้ผู้ใช้ป้อนชื่อฐานข้อมูลที่ต้องการ



รูปที่ 4-11 จอภาพขณะใช้งานของรายการย่อย Open

2. การแสดงผล เป็นการแสดงรายละเอียดอย่างคร่าวๆ ของแต่ละ relation ที่อยู่ในฐานข้อมูลที่ได้นำเข้าสู่ระบบเรียบร้อยแล้ว

รูปที่ 4-12 แสดงจอภาพซึ่งแสดงรายชื่อรีเลชั่นต่างๆ พร้อมทั้งรายละเอียดอื่นๆ เมื่อผู้ใช้เลือกฐานข้อมูลที่ต้องการ



รูปที่ 4-12 จอภาพขณะใช้งานของรายการย่อย Open

ข้อจำกัดและคำแนะนำเพิ่มเติม

1. ถ้าผู้ใช้ป้อนชื่อของฐานข้อมูลที่ไม่ได้ปรากฏอยู่ใน directory window โปรแกรมจะข้อความบอกความผิดพลาดที่เกิดขึ้นแล้วจะกลับสู่การทำงานหลัก

2. ในกรณีที่ผู้ใช้ต้องการนำฐานข้อมูลใหม่เข้ามาไว้ในระบบ จะต้องมีการนำฐานข้อมูลเดิมที่อยู่ในระบบขณะนั้น ออกจากระบบก่อนโดยใช้ รายการย่อย close เพื่อให้ระบบว่างพร้อมที่จะรับฐานข้อมูลใหม่เข้ามาในระบบ สำหรับทำงานต่อไป

3. ในชั่วขณะเวลาหนึ่งๆ จะมีฐานข้อมูลซึ่งถูกเรียกใช้งานในระบบได้เพียงฐานข้อมูลเดียว

ชื่อรายการ Close

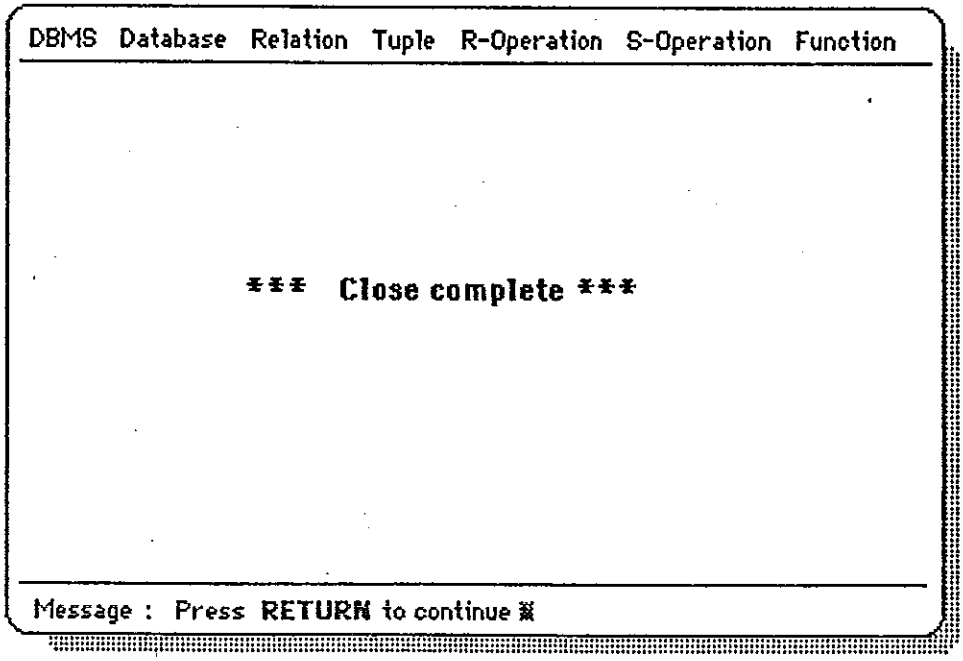
การทำงาน เป็นการนำฐานข้อมูลออกจากระบบ เพื่อให้ระบบว่างพร้อมที่จะนำฐานข้อมูลใหม่เข้ามาในระบบเพื่อปฏิบัติงานต่อไป

การใช้งาน เมื่อผู้ใช้เลือกรายการย่อยนี้แล้ว โปรแกรมจะทำการ Clear เนื้อที่ในส่วน Display area และจะแสดงข้อความ

*** Close database complete ***

เพื่อให้ทราบว่าระบบว่าง และพร้อมที่จะรับฐานข้อมูลใหม่เข้ามาในระบบได้ ถ้าผู้ใช้ต้องการ

จอภาพขณะใช้งาน



รูปที่ 4-13 จอภาพขณะใช้งานของรายการย่อย Close

ข้อจำกัดและคำแนะนำเพิ่มเติม

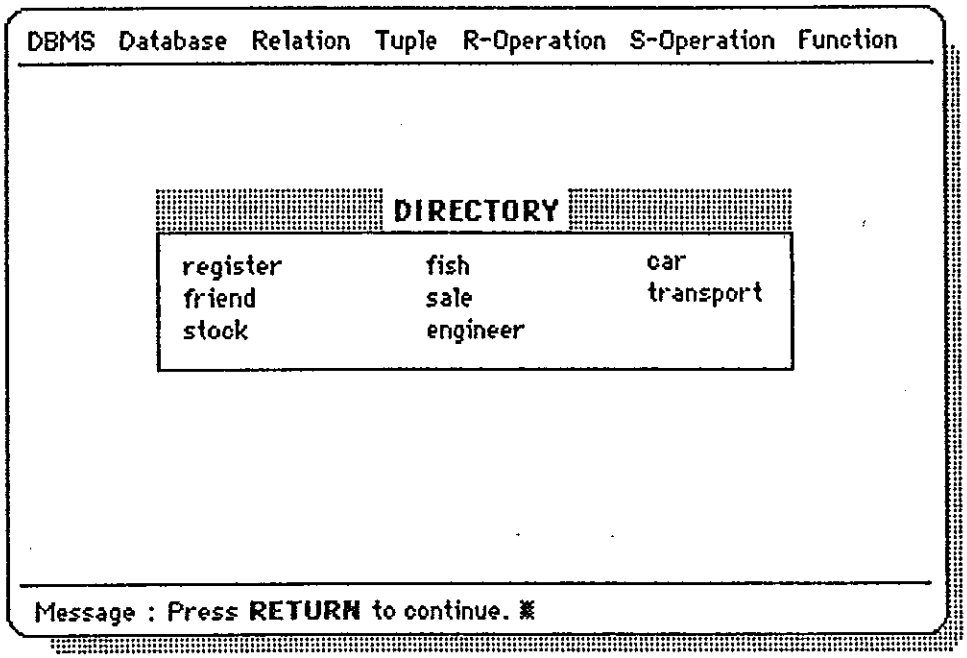
1. ในกรณีที่ผู้ใช้เลือกรายการย่อยนี้ โดยที่ในระบบไม่มีฐานข้อมูลใดอยู่เลย โปรแกรมจะแสดงข้อความเพื่อบอกให้ผู้ใช้ทราบ

ชื่อรายการ Directory

การทำงาน เพื่อแสดงรายชื่อของฐานข้อมูลที่สามารถนำมาใช้ในระบบได้

การใช้งาน โปรแกรมจะแสดงหน้าต่าง directory ซึ่งเป็นเก็บรายชื่อของฐานข้อมูลทั้งหมด ต่อจากนั้นโปรแกรมจะรอให้ผู้ใช้กดแป้นพิมพ์เพื่อกลับสู่การทำงานหลัก

จอภาพขณะใช้งาน



รูปที่ 4-14 จอภาพขณะใช้งานของรายการย่อย Directory

ข้อจำกัดและคำแนะนำเพิ่มเติม

ทุกครั้งที่ผู้ใช้เลือกรายการย่อย Open โปรแกรมจะมาเรียกการทำงานในส่วนของ directory เพื่อให้ผู้ใช้ได้ทราบถึงฐานข้อมูลที่มีอยู่เสมอ

ชื่อรายการ Display Infor.

การทำงาน เป็นการแสดงรายละเอียดของฐานข้อมูลที่อยู่ในระบบ ว่าประกอบไปด้วย relation ใดบ้าง แต่ละรีเลชั่นมีจำนวน tuple เท่าใด ขนาดของ tuple และอื่นๆ

การใช้งาน เมื่อผู้ใช้เลือกรายการย่อยนี้แล้ว โปรแกรมจะแสดงผลที่ได้ใน Display area แล้วจะรอให้ผู้ใช้กดแป้นพิมพ์เพื่อกลับสู่การทำงานหลัก

จอภาพขณะใช้งาน

DBMS Database Relation Tuple R-Operation S-Operation Function

Database name : sale

Relation name	Type	*Tuple	Tuple size	Create date
S	B	6	25	21/01/88
P	B	5	20	01/09/88
SP	B	15	12	27/07/88
SP1	K	13	7	22/01/88

Message : Press **RETURN** to continue. ※

รูปที่ 4-15 ตัวอย่างผลลัพธ์จากการใช้งานของรายการย่อย Display Infor.

ข้อจำกัดและคำแนะนำเพิ่มเติม

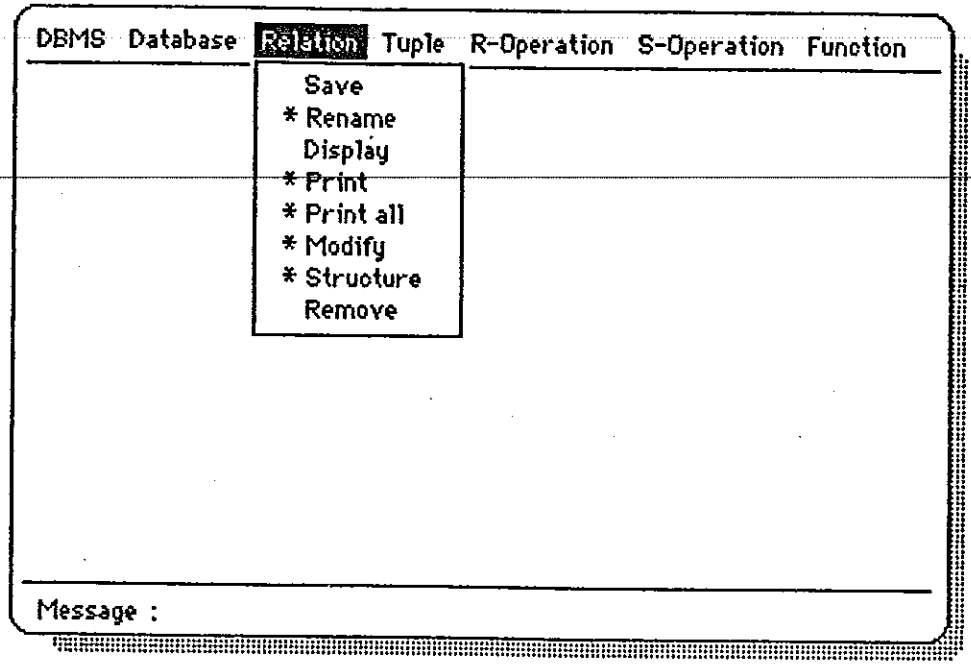
ทุกครั้งที่ผู้ใช้เลือกรายการย่อย Open และโปรแกรมได้นำฐานข้อมูลเข้าสู่ระบบเรียบร้อยแล้ว การทำงานจะผ่านส่วนของการ Display infor.

ชื่อรายการ Quit

การทำงาน เป็นการเลิกการทำงานทั้งหมดของโปรแกรม

การใช้งาน โปรแกรมจะคืนสถานะของจอภาพ ก่อนจะแสดง shell prompt

4.5.3 รายการหลัก Relation



รูปที่ 4-16 จอภาพรายการหลัก Relation

ในรายการหลักส่วนนี้จะเป็นการทำงานเกี่ยวรีเลชั่นทั้งหมด เช่นการเก็บรีเลชั่นที่ได้จากการทำงานไว้ในฐานข้อมูลเพื่อใช้งานต่อไป การแสดงข้อมูลที่อยู่ในแต่ละรีเลชั่นเป็นต้น รายการหลักนี้ประกอบไปด้วยรายการย่อย 8 รายการ ดังนี้คือ

- Save
- Rename*
- Display
- Print*
- Print all*
- Modify*
- Structure*
- Remove

* ยังไม่มีการพัฒนาโปรแกรมสำหรับการทำงานในรายการย่อยนี้

ชื่อรายการ Save

การทำงาน เป็นการเก็บ temporary relation เพิ่มเข้าไปในฐานข้อมูลที่อยู่ในระบบขณะนั้น เพื่อใช้ในการทำงานอื่นๆ ต่อไป

การใช้งาน เมื่อผู้ใช้เลือกรายการย่อยนี้โปรแกรมจะนำ temporary relation ไปเพิ่มไว้ในฐานข้อมูล ต่อจากนั้นโปรแกรมจะไปเรียกใช้การทำงานของรายการย่อย Display infor. เพื่อแสดงให้ผู้ใช้ได้ทราบว่า temporary relation นั้นๆ ได้ถูกนำเข้าไปเพิ่มในฐานข้อมูลเรียบร้อยแล้ว โดยจะเปลี่ยนสถานะจาก temporary relation ไปเป็น keep relation (K) หลังจากนั้นโปรแกรมจะรอให้ผู้ใช้กดแป้นพิมพ์เพื่อกลับสู่การทำงานหลัก

จอภาพขณะใช้งาน

DBMS Database Relation Tuple R-Operation S-Operation Function

Database name : sale

Relation name	Type	#Tuple	Tuple size	Create date
S	B	6	25	21/01/88
P	B	5	20	01/09/88
SP	B	15	12	27/07/88
SP1	K	13	7	22/01/88

Message : Press **RETURN** to continue. ▒

รูปที่ 4-17 ตัวอย่างผลลัพธ์จากการใช้งานของรายการย่อย Save

ข้อจำกัดและคำแนะนำเพิ่มเติม

1. เมื่อ temporary relation ได้ถูกนำไปเพิ่มไว้ในฐานข้อมูลเรียบร้อยแล้ว โปรแกรมจะถือว่าไม่มี temporary relation อยู่ในระบบแล้ว ดังนั้นถ้าผู้ใช้เลือกรายการย่อยนี้ซ้ำอีก โปรแกรมจะแสดงข้อความแสดงการผิดพลาดในการทำงานให้ผู้ใช้ทราบ แล้วจะกลับสู่การทำงานหลักต่อไป

2. temporary relation ที่จะนำไปเก็บในฐานข้อมูลได้นั้นจะต้องมีจำนวน tuple อย่างน้อย 1 tuple

ชื่อรายการ Display

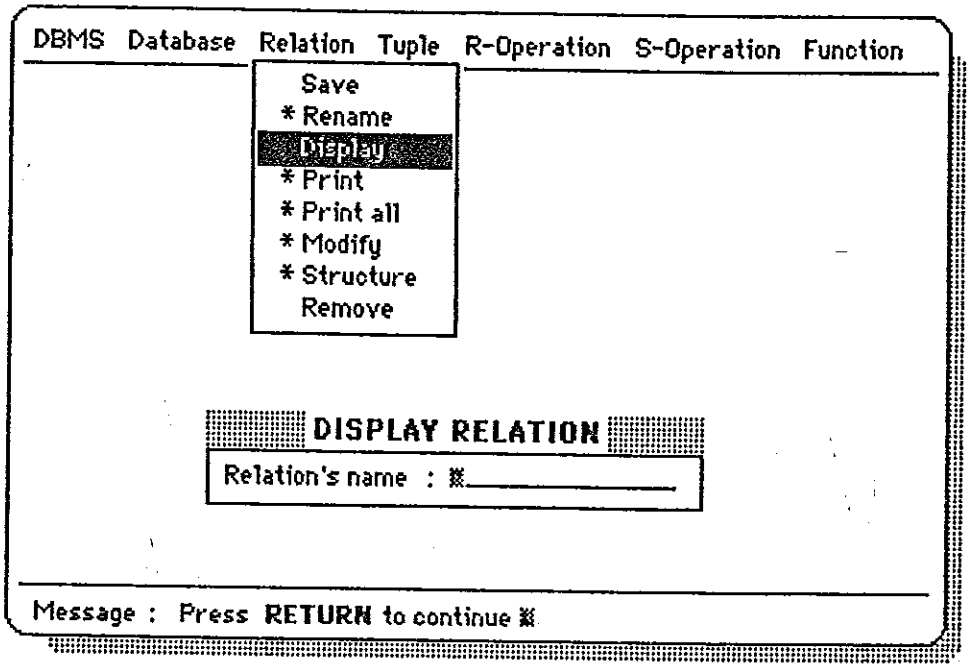
การทำงาน เป็นการแสดงรายละเอียดของข้อมูลที่อยู่ในรีเลชั่นที่กำหนด

การใช้งาน จะแบ่งการใช้งานออกเป็น 2 ขั้นตอน คือ

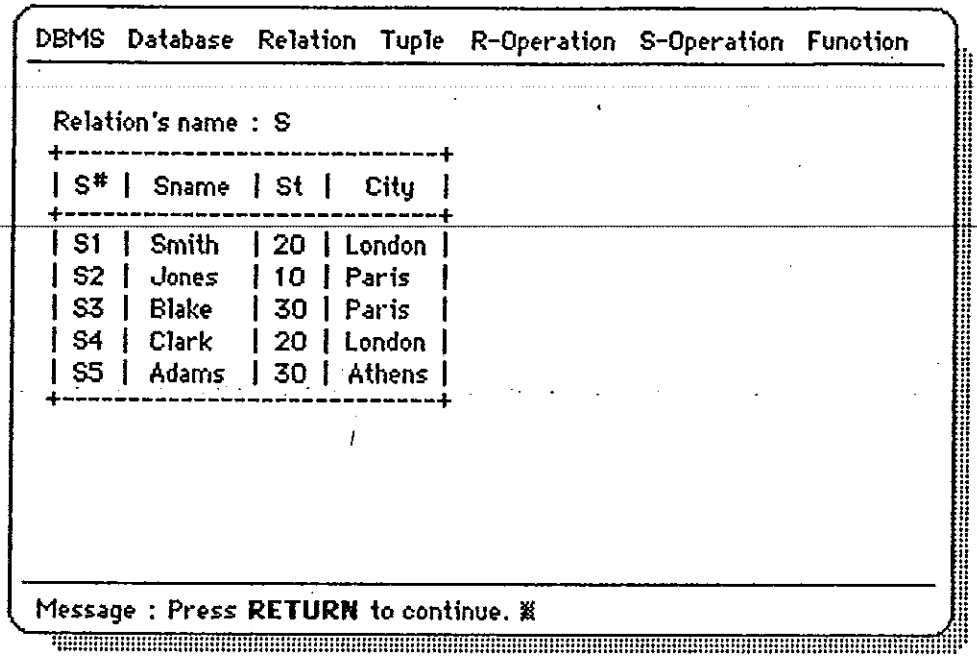
1. การรับข้อมูล โปรแกรมจะแสดงหน้าต่างเพื่อให้ผู้ใช้ป้อนชื่อของรีเลชั่นที่ต้องการ ดังรูป

2. การแสดงผล โปรแกรมจะแสดงรายละเอียดของข้อมูลที่มีอยู่ในรีเลชั่นนั้น แล้วให้ผู้ใช้กดแป้นพิมพ์เพื่อกลับสู่การทำงานหลัก

จอภาพขณะใช้งาน



รูปที่ 4-18 จอภาพขณะใช้งานของรายการย่อย Display



รูปที่ 4-19 ตัวอย่างของผลลัพธ์จากการใช้งานรายการย่อย Display

ข้อจำกัดและคำแนะนำเพิ่มเติม

1. ในกรณีที่ผู้ใช้ป้อนชื่อของรีเลชั่น ที่ไม่มีอยู่ในฐานข้อมูลที่อยู่ในระบบขณะนั้น โปรแกรมจะแสดงข้อความบอกความผิดพลาด แล้วกลับสู่การทำงานหลัก

2. ถ้าผู้ใช้จำไม่ได้ว่าในฐานข้อมูลที่อยู่ในระบบขณะนี้นั้น ประกอบด้วยรีเลชั่นอะไรบ้าง ให้ผู้ใช้เลือกการทำงานของรายการย่อย Display infor. เพื่อช่วยในการทำงาน

3. ในกรณีที่ tuple ของรีเลชั่นมีมากเกินกว่าที่จะแสดงในจอภาพ โปรแกรมจะแสดงจำนวนของ tuple ทั้งหมดได้จาก set option Print to file ให้มีสถานะเป็น ON เพื่อเก็บทุกอย่างลงในแฟ้มข้อมูล แล้วจึงสั่งพิมพ์ในตอนหลัง

ชื่อรายการ Remove

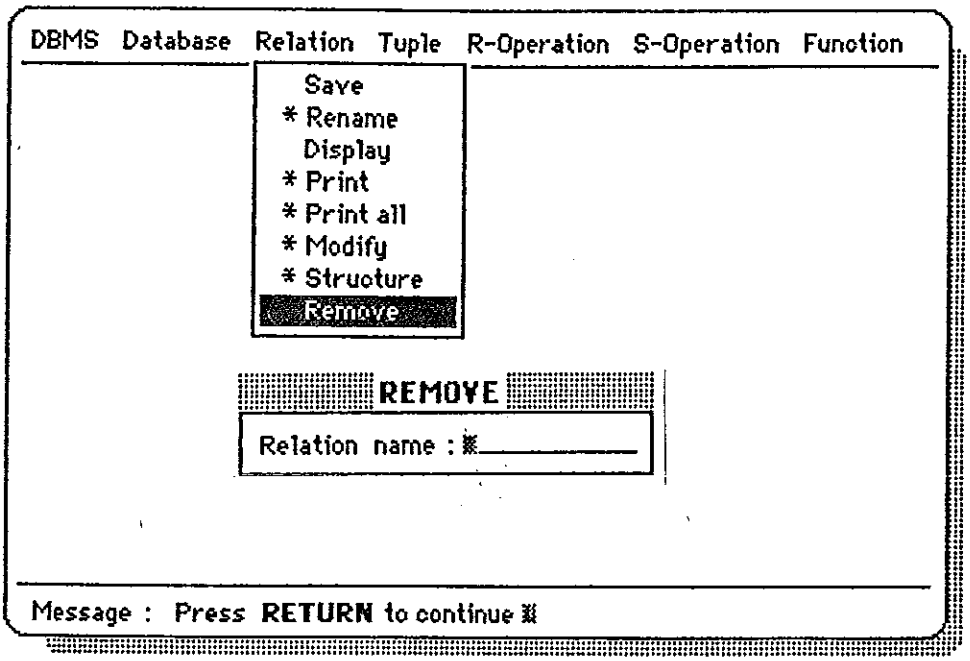
การทำงาน เป็นการนำ keep relation ออกจากฐานข้อมูล

การใช้งาน จะแบ่งการใช้งานออกเป็น 2 ขั้นตอน คือ

1. **การรับข้อมูล** โปรแกรมจะแสดงหน้าต่างเพื่อให้ผู้ใช้ป้อนชื่อของรีเลชั่นที่ต้องการจะนำออกจากฐานข้อมูล ดังรูป

2. **การแสดงผล** เมื่อโปรแกรมทำการนำรีเลชั่นที่ต้องการออกจากฐานข้อมูลเรียบร้อยแล้ว โปรแกรมจะไปเรียกใช้การทำงานของรายการย่อย Display infor. เพื่อแสดงให้ผู้ใช้ได้ทราบถึงรีเลชั่นที่เหลืออยู่

จอภาพขณะใช้งาน



รูปที่ 4-20 จอภาพขณะใช้งานของรายการย่อย Remove

DBMS Database Relation Tuple R-Operation S-Operation Function

Database name : sale

Relation name	Type	#Tuple	Tuple size	Create date
S	B	6	25	21/01/88
P	B	5	20	01/09/88
SP	B	15	12	27/07/88

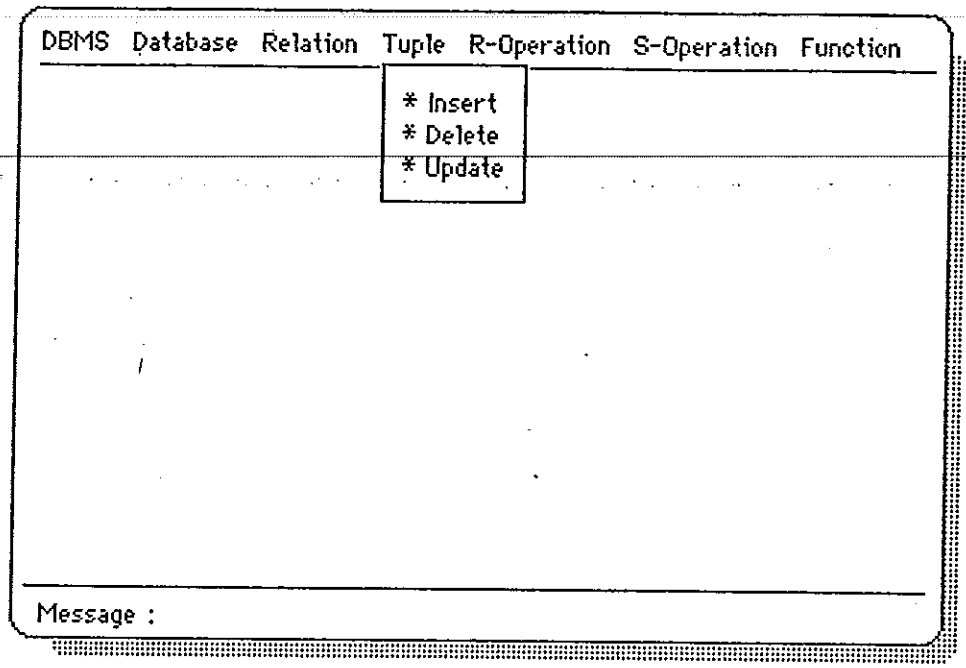
Message : Press **RETURN** to continue. ※

รูปที่ 4-21 ผลลัพธ์จากการใช้งานของรายการย่อย Remove

ข้อจำกัดและคำแนะนำเพิ่มเติม

รีเลชั่นที่จะนำออกจากฐานข้อมูลได้นั้น จะต้องเป็นรีเลชั่นประเภท keep relation เท่านั้น

4.5.4 รายการหลัก Tuple



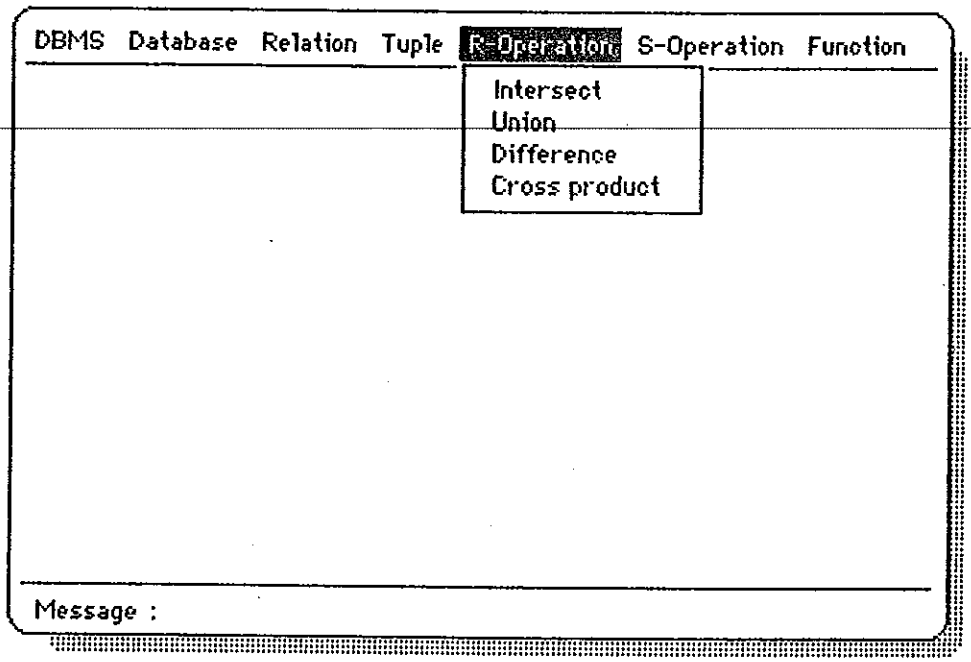
รูปที่ 4-22 จอภาพรายการหลัก Tuple

ในรายการหลักส่วนนี้จะเป็นการทำงานเกี่ยวกับ tuple ในรีเลชั่น เช่นการเพิ่ม ลบ หรือเปลี่ยนแปลงข้อมูลภายใน tuple เป็นต้น รายการหลักนี้ประกอบไปด้วยรายการย่อย 3 รายการ ดังนี้คือ

- Insert *
- Delete *
- Update *

* ยังไม่มีการพัฒนาโปรแกรมสำหรับการทำงานในรายการย่อยนี้

4.5.5 รายการหลัก R-Operation



รูปที่ 4-23 จอภาพรายการหลัก R-Operation

ในรายการหลักนี้เป็นคำสั่งปฏิบัติการกับฐานข้อมูล แบบรีเลชั่น โดยลักษณะของคำสั่งจะเป็นคำสั่งแบบ set operation ซึ่งได้แก่

- Intersection
- Union
- Difference
- Cross product

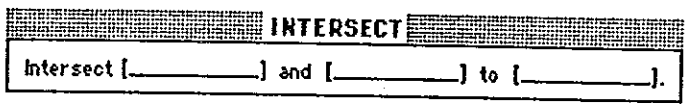
สำหรับคำสั่งเหล่านี้ สามารถดูรายละเอียดการปฏิบัติการได้ใน บทที่ 2

ชื่อรายการ Intersection

การทำงาน ดูนิยาม รายละเอียด และตัวอย่างในบทที่ 2

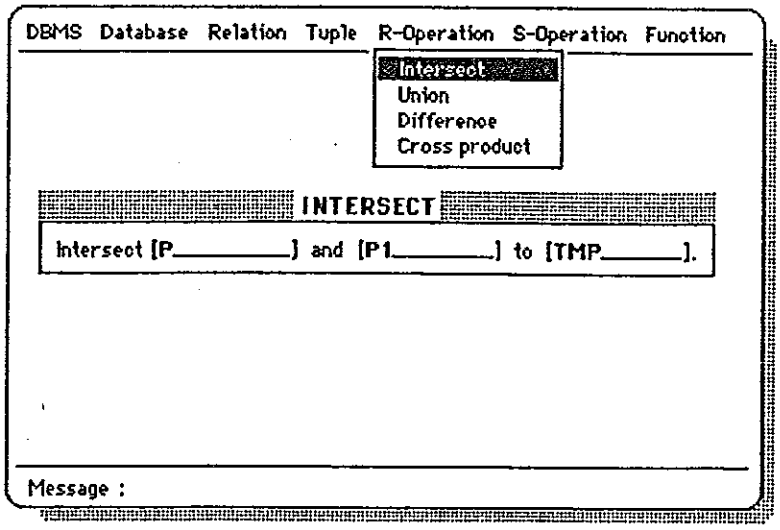
การใช้งาน จะแบ่งออกเป็น 2 ขั้นตอน ดังนี้คือ

1. การรับข้อมูล โปรแกรมจะแสดงหน้าต่าง intersection เพื่อให้ผู้ใช้ป้อนชื่อของรีเลชั่นทั้งสองที่จะ intersection กันและชื่อของรีเลชั่นที่ต้องการนำผลที่ได้จากการทำงานมาเก็บไว้ ดังรูป



2. การแสดงผล เมื่อผลที่ได้จากการทำงานถูกเก็บไว้ในรีเลชั่นเรียบร้อยแล้ว โปรแกรมจะไปเรียกรายการย่อย display มาใช้งานเพื่อให้ผู้ใช้ทราบถึงผลที่ได้จากการทำงาน

จอภาพขณะใช้งาน



รูปที่ 4-24 จอภาพขณะใช้งานของรายการย่อย Intersection

ข้อจำกัดและคำแนะนำเพิ่มเติม

1. ชื่อของรีเลชั่นทั้งสองที่จะนำมาทำการ intersect จะต้องปรากฏอยู่ในฐานข้อมูลที่อยู่ในระบบ และ ชื่อทั้งสองจะต้องไม่ซ้ำกัน

2. การที่จะนำรีเลชั่นทั้งสองมา intersection กันได้นั้น รีเลชั่นทั้งสองจะต้องมีขนาดของ tuple ที่เท่ากัน (จำนวน attribute ที่เท่ากัน) และมี domain type ของ attribute ที่สมนัยกัน เหมือนกัน

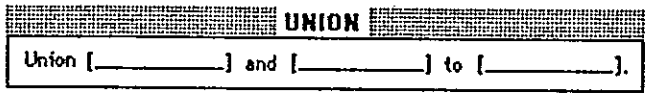
3. ในกรณีที่ไม่มีฐานข้อมูลอยู่ในระบบ โปรแกรมจะเรียกการทำงานของรายการย่อย open เพื่อทำการนำฐานข้อมูลมาไว้ในระบบเสียก่อน แล้วจึงจะทำงานต่อไป

ชื่อรายการ Union

การทำงาน ดูนิยาม รายละเอียด และตัวอย่างในบทที่ 2

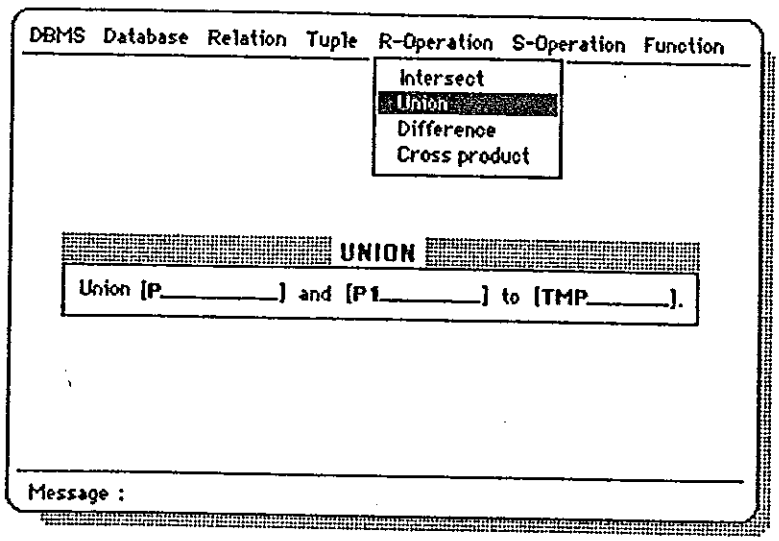
การใช้งาน จะแบ่งออกเป็น 2 ขั้นตอน ดังนี้คือ

1. การรับข้อมูล โปรแกรมจะแสดงหน้าต่าง union เพื่อให้ผู้ใช้ป้อนชื่อของรีเลชั่นทั้งสองที่จะทำการ union และชื่อของรีเลชั่นที่ต้องการนำผลที่ได้จากการทำงานมาเก็บไว้ ดังรูป



2. การแสดงผล เมื่อผลที่ได้จากการทำงานถูกเก็บไว้ในรีเลชั่นเรียบร้อยแล้ว โปรแกรมจะไปเรียกรายการย่อย display มาใช้งานเพื่อให้ผู้ใช้ทราบถึงผลที่ได้จากการทำงาน

จอภาพขณะใช้งาน



รูปที่ 4-25 ตัวอย่างการใช้งานของรายการย่อย Union

ข้อจำกัดและคำแนะนำเพิ่มเติม

1. ชื่อของรีเลชั่นทั้งสองที่จะนำมาทำการ union จะต้องปรากฏอยู่ในฐานข้อมูลที่อยู่ในระบบ และ ชื่อทั้งสองจะต้องไม่ซ้ำกัน

2. การที่จะนำรีเลชั่นทั้งสองมาทำการ union ได้นั้น รีเลชั่นทั้งสองจะต้องมีขนาดของ tuple ที่เท่ากัน (จำนวน attribute ที่เท่ากัน) และมี domain type ของ attribute ที่สมนัยกัน เหมือนกัน

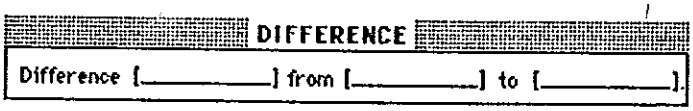
3. ในกรณีที่ไม่มีฐานข้อมูลอยู่ในระบบ โปรแกรมจะเรียกการทำงานของรายการย่อย open เพื่อทำการนำฐานข้อมูลมาไว้ในระบบเสียก่อน แล้วจึงจะทำงานต่อไป

ชื่อรายการ Difference

การทำงาน ดูนิยาม รายละเอียด และตัวอย่างในบทที่ 2

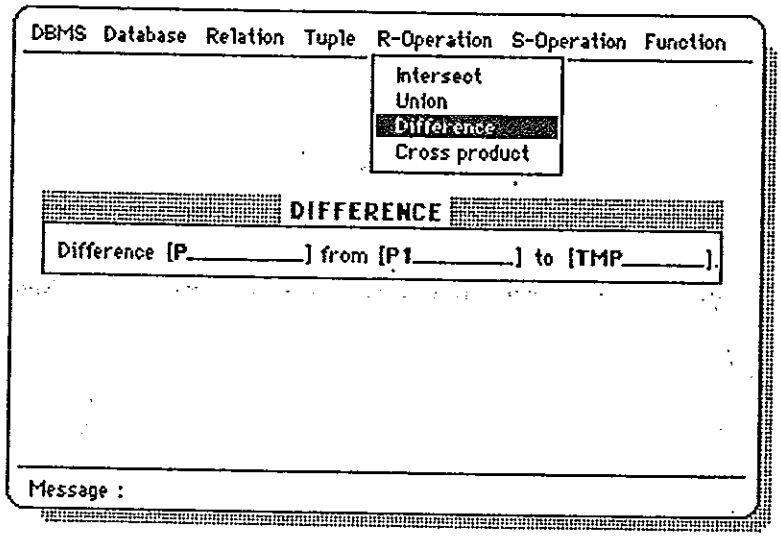
การใช้งาน จะแบ่งออกเป็น 2 ขั้นตอน ดังนี้คือ

1. การรับข้อมูล โปรแกรมจะแสดงหน้าต่าง difference เพื่อให้ผู้ใช้ป้อนชื่อของรีเลชั่นทั้งสองที่จะนำมา difference และชื่อของรีเลชั่นที่ต้องการนำผลที่ได้จากการทำงานมาเก็บไว้ ดังรูป



2. การแสดงผล เมื่อผลที่ได้จากการทำงานถูกเก็บไว้ในรีเลชั่นเรียบร้อยแล้ว โปรแกรมจะไปเรียกรายการย่อย display มาใช้งานเพื่อให้ผู้ใช้ทราบถึงผลที่ได้จากการทำงาน

จอภาพขณะใช้งาน



รูปที่ 4-26 จอภาพขณะใช้งานของรายการย่อย Difference

ข้อจำกัดและคำแนะนำเพิ่มเติม

1. ชื่อของรีเลชั่นทั้งสองที่จะนำมาทำการ difference จะต้องปรากฏอยู่ในฐานข้อมูลที่อยู่ในระบบ และ ชื่อทั้งสองจะต้องไม่ซ้ำกัน

2. การที่จะนำรีเลชั่นทั้งสองมาทำการ difference ได้นั้น รีเลชั่นทั้งสองจะต้องมีขนาดของ tuple ที่เท่ากัน (จำนวน attribute ที่เท่ากัน) และมี domain type ของ attribute ที่สมนัยกัน เหมือนกัน

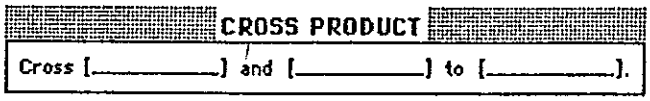
3. ในกรณีที่ไม่มีฐานข้อมูลอยู่ในระบบ โปรแกรมจะเรียกการทำงานของรายการย่อย open เพื่อทำการนำฐานข้อมูลมาไว้ในระบบเสียก่อน แล้วจึงจะทำงานต่อไป

ชื่อรายการ Cross product

การทำงาน ดูนิยาม รายละเอียด และตัวอย่างในบทที่ 2

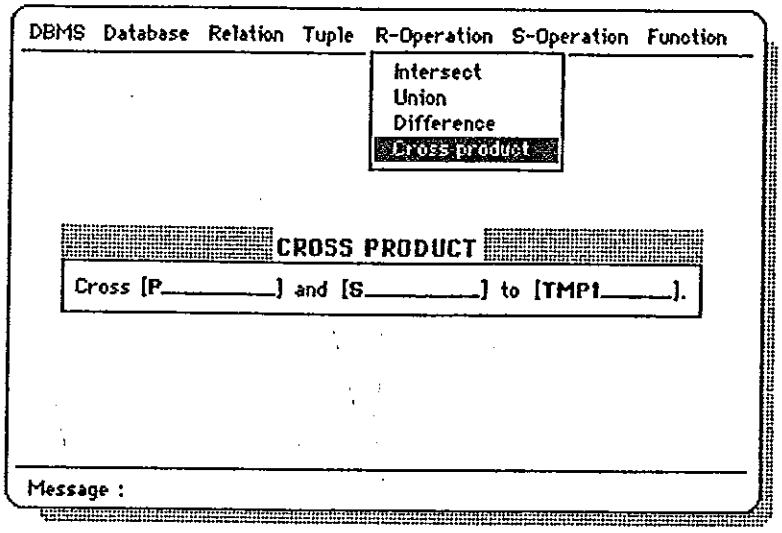
การใช้งาน จะแบ่งออกเป็น 2 ขั้นตอน ดังนี้คือ

1. การรับข้อมูล โปรแกรมแสดงหน้าต่าง Cross-product เพื่อให้ผู้ใช้ป้อนชื่อของรีเลชั่นทั้ง 2 ที่จะนำมา Cross-product และชื่อของรีเลชั่นที่ต้องการนำผลที่ได้จากการทำงานมาเก็บไว้



2. การแสดงผล เมื่อผลที่ได้จากการทำงานถูกเก็บไว้ในรีเลชั่นเรียบร้อยแล้ว โปรแกรมจะไปเรียกรายการย่อย display มาใช้งานเพื่อให้ผู้ใช้ทราบถึงผลที่ได้จากการทำงาน

จอภาพขณะใช้งาน



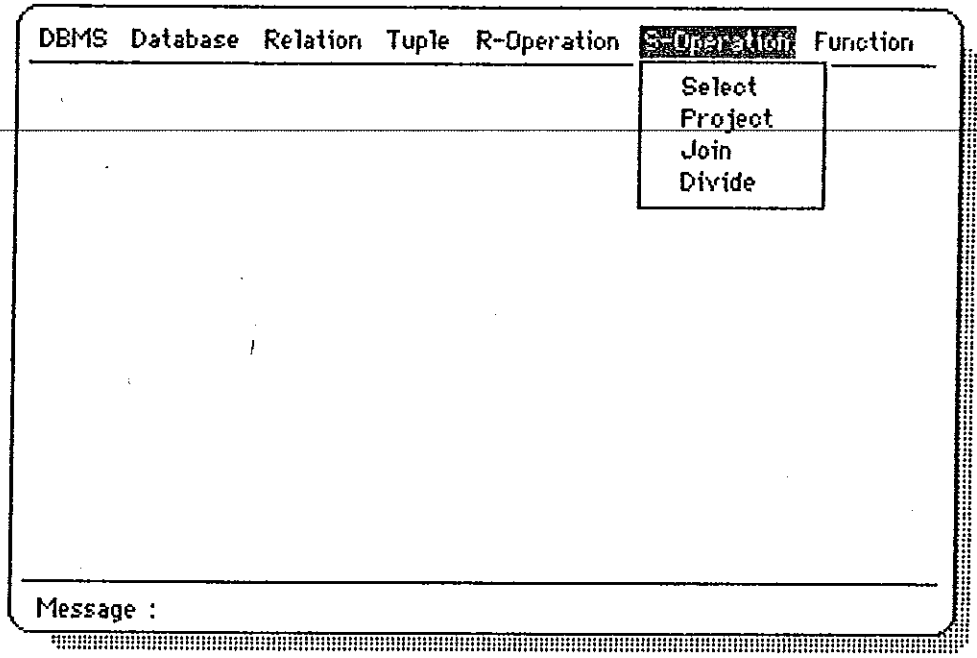
รูปที่ 4-27 ตัวอย่างการใช้งานของรายการย่อย Cross product

ข้อจำกัดและคำแนะนำเพิ่มเติม

1. ชื่อของรีเลชั่นทั้งสองที่จะนำมาทำการ Cross product จะต้องปรากฏอยู่ในฐานข้อมูลที่อยู่ในระบบ และ ชื่อทั้งสองจะต้องไม่ซ้ำกัน

2. ในกรณีที่ไม่มีฐานข้อมูลอยู่ในระบบ โปรแกรมจะเรียกการทำงานของรายการย่อย open เพื่อทำการนำฐานข้อมูลมาไว้ในระบบเสียก่อน แล้วจึงจะทำงานต่อไป

4.5.6 รายการหลัก S-Operation



รูปที่ 4-28 จอภาพรายการหลัก S-Operation

คำสั่งปฏิบัติการในหัวข้อนี้เป็นคำสั่งพิเศษที่ใช้ปฏิบัติการกับข้อมูล ซึ่งประกอบด้วยรายการย่อย 4 รายการ ดังนี้คือ

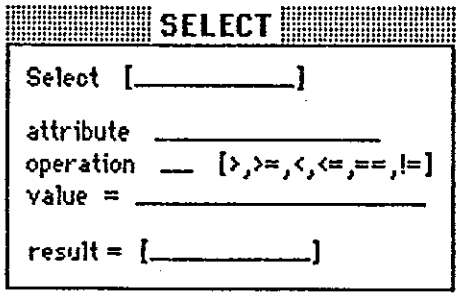
- Select
- Project
- Join
- Divide

ชื่อรายการ Select

การทำงาน ดูนิยาม รายละเอียด และตัวอย่างได้ในบทที่ 2

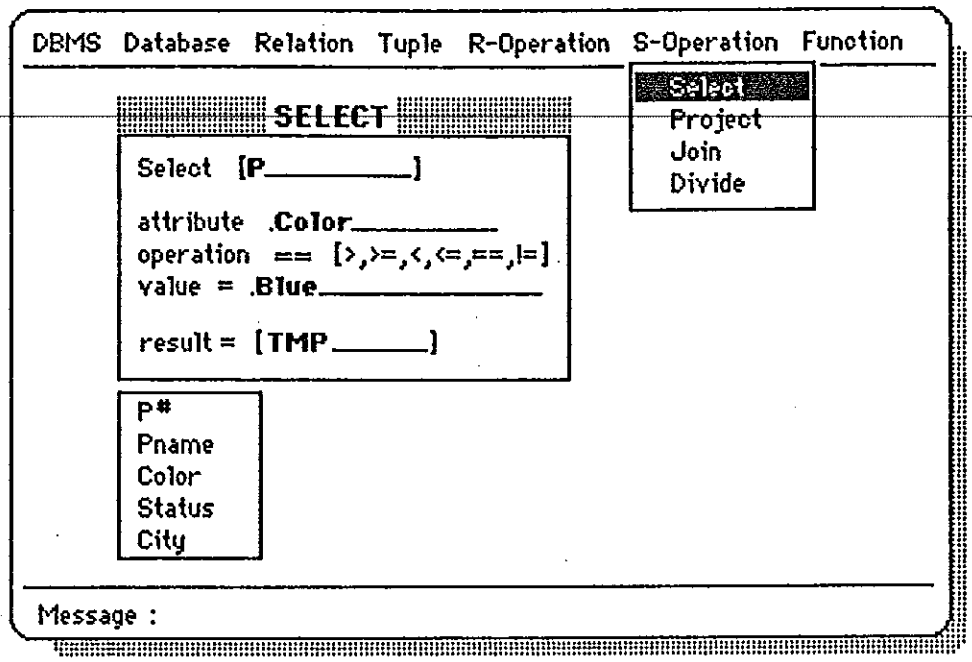
การใช้งาน จะแบ่งการทำงานออกเป็น 2 ขั้นตอน คือ

1. การรับข้อมูล เมื่อผู้ใช้เลือกรายการย่อยนี้โปรแกรมจะแสดง window ของ select เพื่อให้ผู้ใช้ป้อนชื่อรีเลชั่นและเงื่อนไขซึ่งประกอบด้วยชื่อ attribute เครื่องหมายของความสัมพันธ์ (relational operator) ค่าของ attribute ที่ต้องการให้เป็นไปตามเงื่อนไข และชื่อรีเลชั่นที่ต้องการนำผลลัพธ์ไปเก็บ ตามลำดับ



2. การแสดงผล การทำงานในรายการย่อยส่วนนี้ จะทำการเรียกการทำในรายการย่อย display เพื่อแสดงผลที่ได้จากการทำงาน

จอภาพขณะใช้งาน



รูปที่ 4-29 จอภาพขณะใช้งานของรายการย่อย Select

ข้อจำกัดและคำแนะนำเพิ่มเติม

1. ในกรณีที่ไม่มีฐานข้อมูลอยู่ในระบบ โปรแกรมจะเรียกการทำงานของรายการย่อย open เพื่อทำการนำฐานข้อมูลมาไว้ในระบบเสียก่อน แล้วจึงจะทำงานต่อไป

2. ก่อนที่ผู้ใช้จะทำการป้อนชื่อของ attribute โปรแกรมจะแสดง attribute window เพื่อให้ผู้ใช้ทราบว่ารีเลชั่นที่จะทำการ select นั้นประกอบด้วย attribute ที่ชื่ออะไรบ้าง ดังตัวอย่างแสดงในรูป 4-29

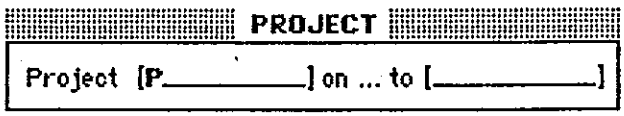
ชื่อรายการ Project

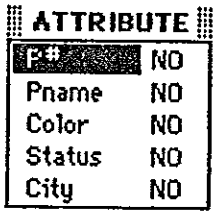
การทำงาน ดูนิยาม รายละเอียด และตัวอย่างได้ในบทที่ 2

การใช้งาน การใช้งานแบ่งออกเป็น 2 ขั้นตอน คือ

1. การรับข้อมูล เมื่อผู้ใช้เลือกรายการย่อยนี้ โปรแกรมจะแสดงหน้าต่างของ project เพื่อให้ผู้ใช้ป้อน
   - ป้อนชื่อรีเลชั่นที่ทำการ project
   - เลือก attribute(s) ที่ต้องการ โดยโปรแกรมจะแสดง attribute window เพื่อให้ผู้ใช้ทำการเลือก ซึ่งสามารถที่จะทำการเลือกได้มากกว่า 1 attribute แป้นพิมพ์ใช้สำหรับเลื่อนขึ้น/ลง space bar สำหรับเลือก และแป้นพิมพ์ return ใช้ตอบตกลง
   - ป้อนชื่อรีเลชั่นที่ต้องการนำผลไปเก็บ

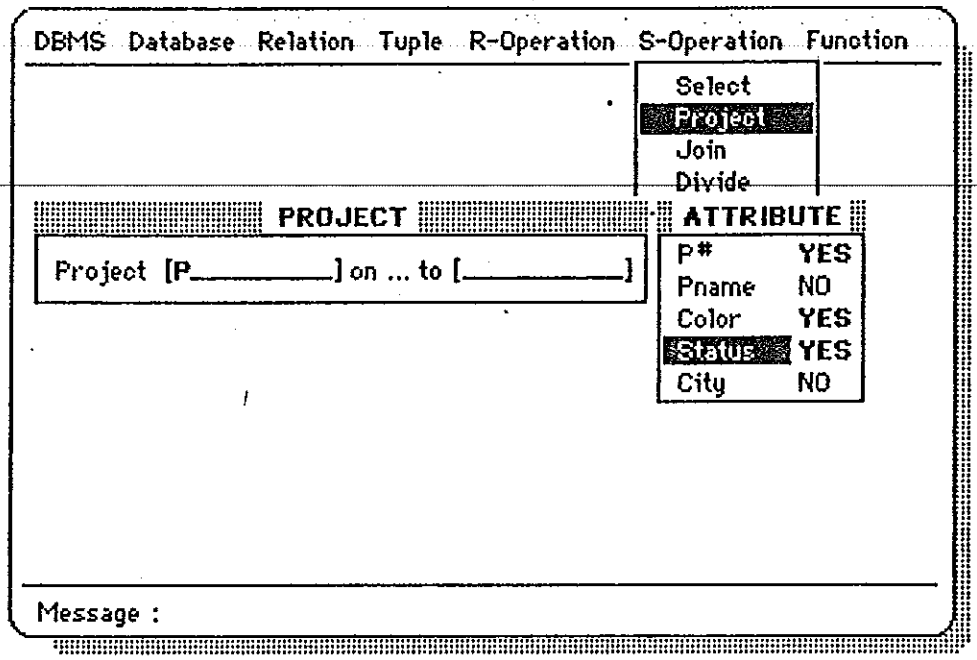
จากรูปเป็นแสดงขั้นตอนในส่วนของการรับข้อมูล





2. การแสดงผล โปรแกรมจะทำการเรียกการทำงานของ รายการย่อย display เพื่อทำการแสดงผลที่ได้จากการทำงาน

จอภาพขณะใช้งาน



รูปที่ 4-30 จอภาพขณะใช้งานของรายการย่อย Project

ข้อจำกัดและคำแนะนำเพิ่มเติม

1. ในกรณีที่ไม่มีฐานข้อมูลอยู่ในระบบ โปรแกรมจะเรียกการทำงานของรายการย่อย open เพื่อทำการนำฐานข้อมูลมาไว้ในระบบเสียก่อน แล้วจึงจะทำงานต่อไป

2. ในกรณีที่ผู้ใช้ไม่ทำการเลือก attribute โปรแกรมจะแสดงข้อความบอกความผิดพลาด แล้วให้ผู้ใช้ทำการเลือกใหม่

3. ผลจากการปฏิบัติการในคำสั่งนี้สามารถที่จะมี tuple ที่มีค่าเหมือนกันได้ ซึ่งถ้าผู้ใช้ไม่ต้องการให้เกิดกรณีนี้ขึ้น ให้ผู้ใช้ทำการเปลี่ยนสถานะของ DUPLICATE ให้มีค่าเป็น OFF ก่อนที่จะทำการเรียกใช้รายการย่อยนี้ (ดูรายละเอียดในรายการหลัก DBMS ที่รายการย่อย Option setup)

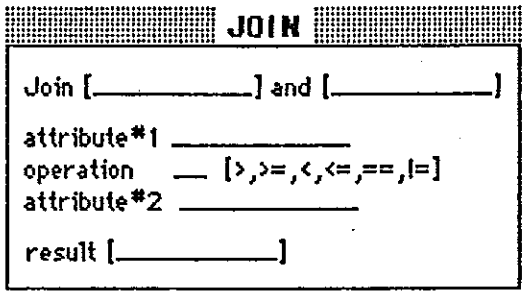
ชื่อรายการ Join

การทำงาน ดูนิยาม รายละเอียด และตัวอย่างได้ในบทที่ 2

การใช้งาน จะแบ่งออกเป็น 3 ขั้นตอน คือ

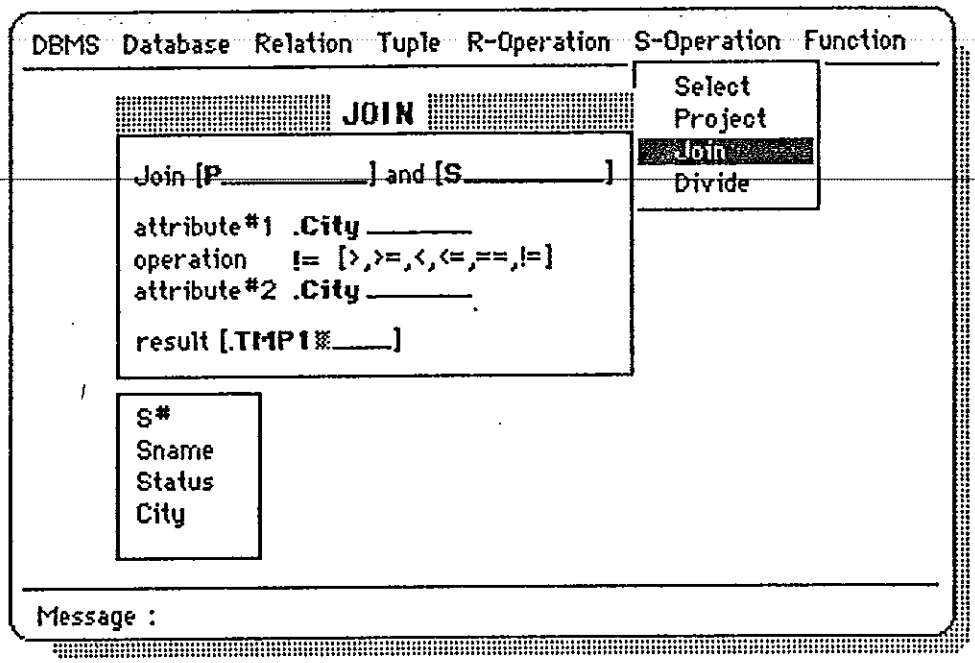
1. การรับข้อมูล เมื่อผู้ใช้เลือกรายการย่อยนี้ โปรแกรมจะแสดงหน้าต่างของ join เพื่อให้ผู้ใช้
   - ป้อนชื่อรีเลชั่นแรก
   - ป้อนชื่อรีเลชั่นที่สอง
   - ป้อนชื่อ attribute ของรีเลชั่นแรก
   - ป้อนเงื่อนไขสำหรับการ join
   - ป้อนชื่อ attribute ของรีเลชั่นที่สอง
   - ป้อนชื่อ รีเลชั่นที่ต้องการนำผลลัพธ์ไปเก็บ

จากรูปเป็นแสดงขั้นตอนในส่วนของการรับข้อมูล



2. การแสดงผล โปรแกรมจะทำการเรียกการทำงานของ รายการย่อย display เพื่อทำการแสดงผลที่ได้จากการทำงาน

จอภาพขณะใช้งาน



รูปที่ 4-31 จอภาพขณะใช้งานของรายการย่อย Join

ข้อจำกัดและคำแนะนำเพิ่มเติม

1. ในกรณีที่ไม่มีฐานข้อมูลอยู่ในระบบ โปรแกรมจะเรียกการทำงานของรายการย่อย open เพื่อทำการนำฐานข้อมูลมาไว้ในระบบเสียก่อน แล้วจึงจะทำงานต่อไป

2. ในกรณีที่โดเมนของ attribute ทั้ง 2 มีค่า domain type ที่ไม่เหมือนกันโปรแกรมจะไม่แสดงผลการทำงาน และจะแสดงข้อความเพื่อบอกความผิดพลาดที่เกิดขึ้น

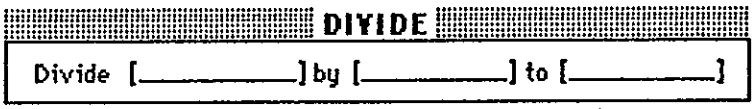
ชื่อรายการ Divide

การทำงาน ดูนิยาม รายละเอียด และตัวอย่างได้ในบทที่ 2

การใช้งาน จะแบ่งออกเป็น 2 ขั้นตอน คือ

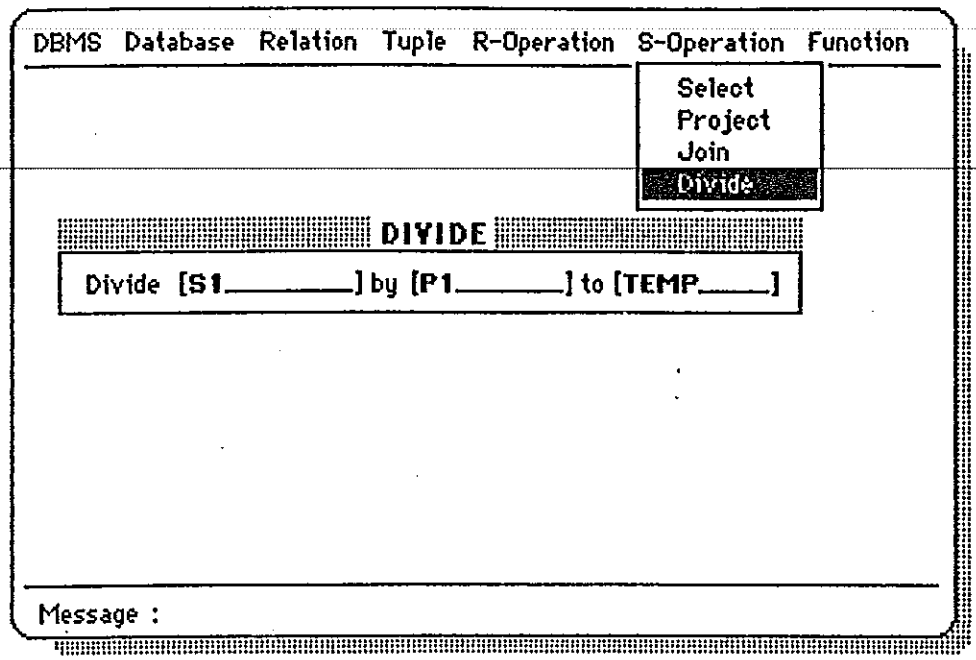
1. การรับข้อมูล เมื่อผู้ใช้เลือกรายการย่อยนี้ โปรแกรมจะแสดงหน้าต่างของ divide เพื่อให้ผู้ใช้ป้อน
   - ชื่อรีเลชั่นที่เป็นตัวตั้ง
   - ชื่อรีเลชั่นที่เป็นตัวหาร
   - ชื่อรีเลชั่นที่ใช้เก็บผลลัพธ์

   จากรูปแสดงการป้อนข้อมูลต่างๆ



2. การแสดงผล โปรแกรมจะทำการเรียกการทำงานของ รายการย่อย display เพื่อทำการแสดงผลที่ได้จากการทำงาน

จอภาพขณะใช้งาน



รูปที่ 4-32 จอภาพขณะใช้งานของรายการย่อย Divide

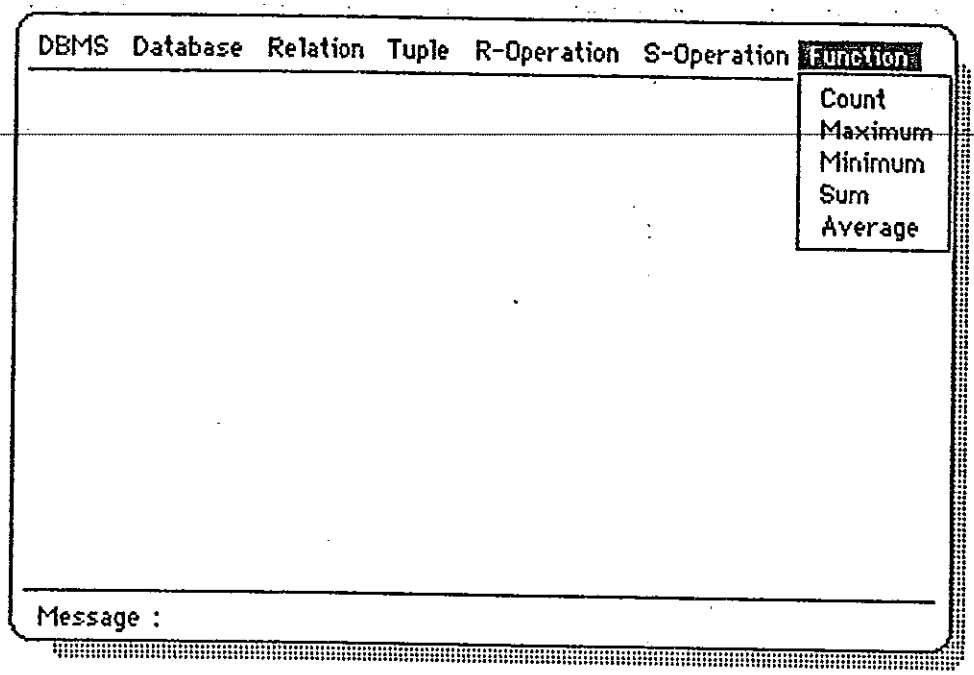
ข้อจำกัดและคำแนะนำเพิ่มเติม

1. ในกรณีที่ไม่มีฐานข้อมูลอยู่ในระบบ โปรแกรมจะเรียกการทำงานของรายการย่อย open เพื่อทำการนำฐานข้อมูลมาไว้ในระบบเสียก่อน แล้วจึงจะทำงานต่อไป

2. รีเลชั่นที่เป็นตัวตั้งต้องเป็น binary relation และรีเลชั่นที่เป็นตัวหารต้องเป็น unary relation

3. attribute หนึ่งของรีเลชั่นที่เป็นตัวตั้งจะต้องมี domain type ที่เหมือนกันกับ attribute ของรีเลชั่นที่เป็นตัวหาร

4.5.7 รายการหลัก Function



รูปที่ 4-33 จอภาพรายการหลัก Function

คำสั่งปฏิบัติการในหัวข้อนี้เป็นคำสั่งพิเศษที่ใช้ปฏิบัติการกับข้อมูล ซึ่งประกอบด้วยรายการย่อย 5 รายการ ดังนี้คือ

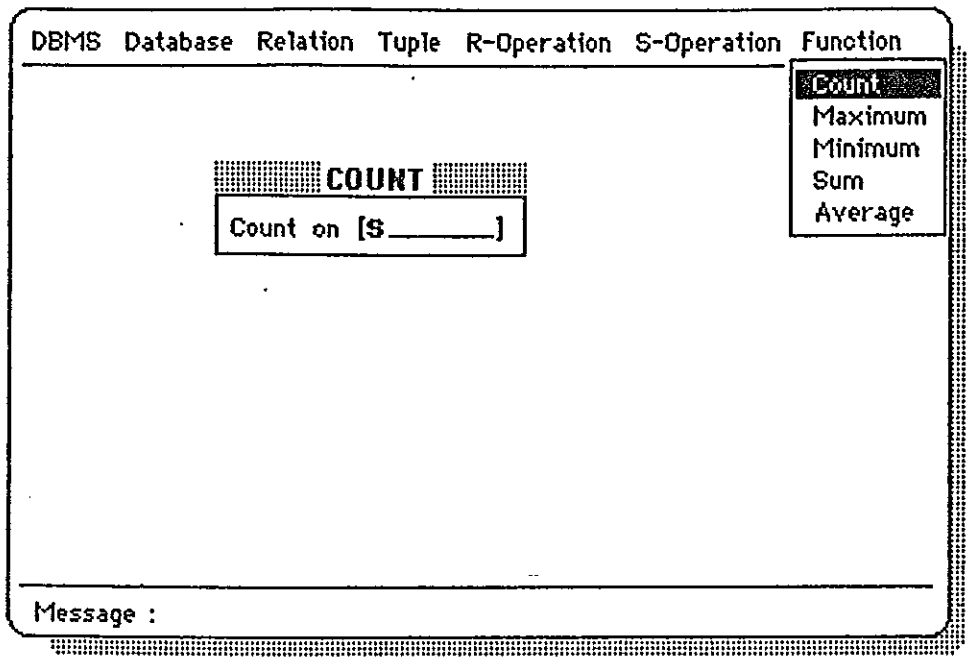
- Count
- Maximum
- Minimum
- Sum
- Average

ชื่อรายการ Count

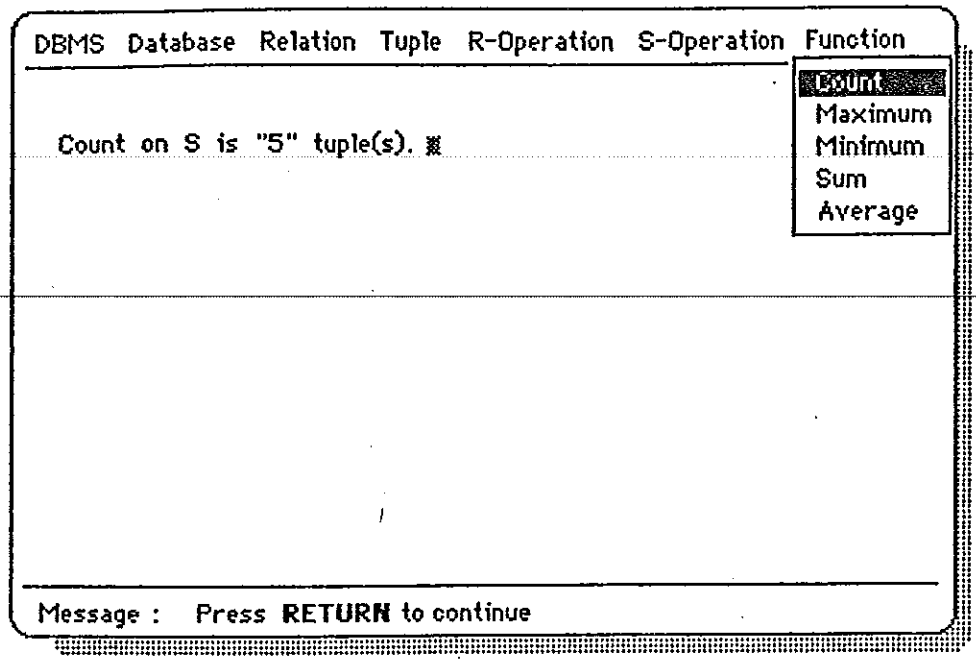
การทำงาน เป็นการนับจำนวน tuple ของรีเลชั่นที่กำหนด

การใช้งาน เมื่อผู้ใช้เลือกรายการย่อยนี้ โปรแกรมจะแสดงหน้าต่างของ Count (ดังรูปที่ 4-34) เพื่อให้ผู้ใช้ป้อนชื่อรีเลชั่นที่ต้องการนับจำนวนของ tuple จากนั้นจึงแสดงผลการทำงานดังรูปที่ 4-35

จอภาพขณะใช้งาน



รูปที่ 4-34 จอภาพขณะใช้งานของรายการย่อย Count



รูปที่ 4-35 จอภาพขณะใช้งานของรายการย่อย Count

ข้อจำกัดและคำแนะนำเพิ่มเติม

ในกรณีที่ไม่มีฐานข้อมูลอยู่ในระบบ โปรแกรมจะเรียกการทำงานของรายการย่อย open เพื่อทำการนำฐานข้อมูลมาไว้ในระบบเสียก่อน แล้วจึงจะทำงานต่อไป

ชื่อรายการ Maximum

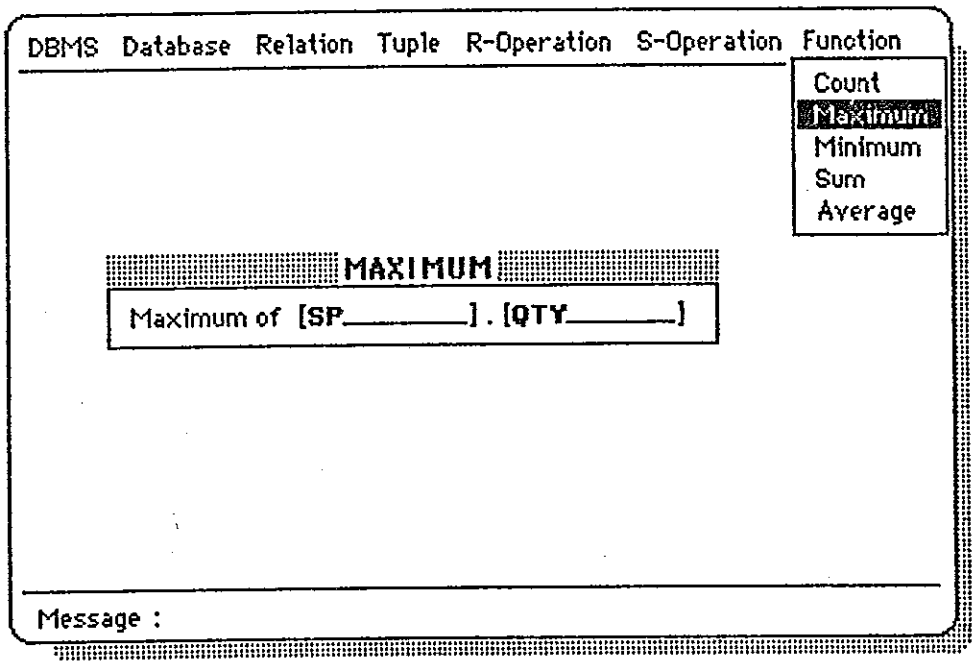
การทำงาน ทำการหาค่าที่สูงสุดจาก attribute ที่กำหนด

การใช้งาน การใช้งานแบ่งออกเป็น 2 ขั้นตอน คือ

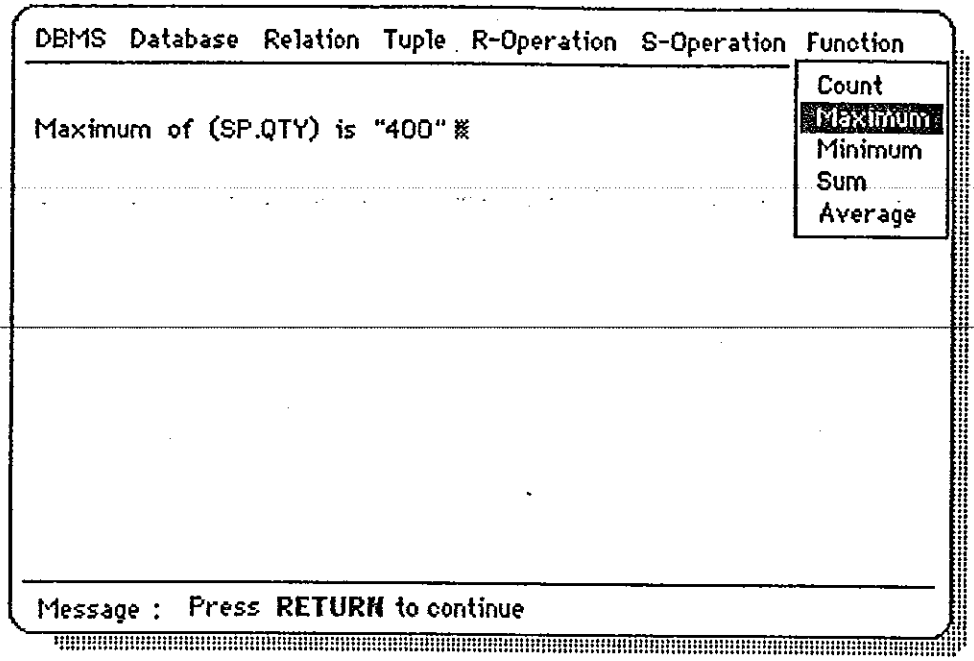
1. การรับข้อมูล เมื่อผู้ใช้เลือกรายการย่อยนี้ โปรแกรมจะแสดงหน้าต่างของ Maximum เพื่อให้ผู้ใช้ป้อน
   - ชื่อรีเลชั่น
   - ชื่อ attribute ที่ต้องการ โดยโปรแกรมจะแสดงหน้าต่าง attribute เพื่อให้ผู้ใช้ทำการเลือก

   จากรูปที่ 4-36 เป็นจอภาพในส่วนของการรับข้อมูล
2. การแสดงผล โปรแกรมจะแสดงผลการทำงาน ดังรูปที่ 4-37

จอภาพขณะใช้งาน



รูปที่ 4-36 จอภาพขณะใช้งานของรายการย่อย Maximum



รูปที่ 4-37 จอภาพขณะใช้งานของรายการย่อย Maximum

ข้อจำกัดและคำแนะนำเพิ่มเติม

1. ในกรณีที่ไม่มีฐานข้อมูลอยู่ในระบบ โปรแกรมจะเรียกการทำงานของรายการย่อย open เพื่อทำการนำฐานข้อมูลมาไว้ในระบบเสียก่อน แล้วจึงจะทำงานต่อไป

2. ในกรณีที่ผู้ใช้ไม่ทำการเลือก attribute โปรแกรมจะแสดงข้อความบอกความผิดพลาด แล้วให้ผู้ใช้ทำการเลือกใหม่

3. ค่าของ attribute ที่จะทำการหาค่าสูงสุดจะต้องไม่มีค่า domain type เป็นตัวอักษร

ชื่อรายการ Minimum

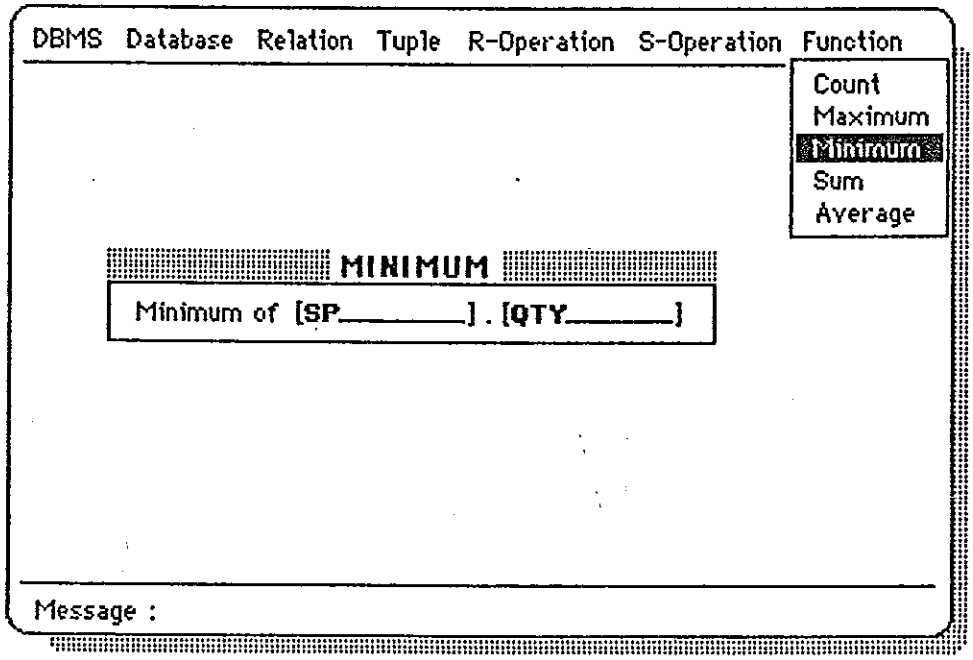
การทำงาน ทำการหาค่าที่ต่ำสุดจาก attribute ที่กำหนด

การใช้งาน การใช้งานแบ่งออกเป็น 2 ขั้นตอน คือ

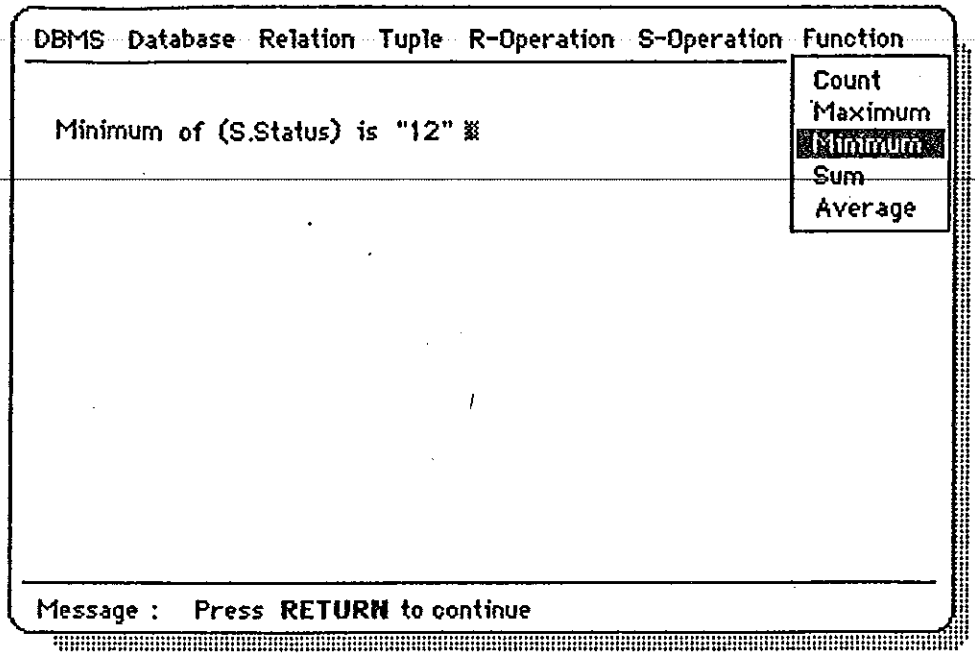
1. การรับข้อมูล เมื่อผู้ใช้เลือกรายการย่อยนี้ โปรแกรมจะแสดงหน้าต่างของ Minimum เพื่อให้ผู้ใช้
   - ป้อนชื่อรีเลชั่น
   - ป้อนชื่อ attribute ที่ต้องการ โดยโปรแกรมจะแสดงหน้าต่าง attribute เพื่อให้ผู้ใช้ทำการเลือก

   จากรูปที่ 4-38 เป็นจอภาพในส่วนของการรับข้อมูล
2. การแสดงผล โปรแกรมจะแสดงผลการทำงาน ดังรูปที่ 4-39

จอภาพขณะใช้งาน



รูปที่ 4-38 จอภาพขณะใช้งานของรายการย่อย Minimum



รูปที่ 4-39 จอภาพขณะใช้งานของรายการย่อย Minimum

ข้อจำกัดและคำแนะนำเพิ่มเติม

1. ในกรณีที่ไม่มีฐานข้อมูลอยู่ในระบบ โปรแกรมจะเรียกการทำงานของรายการย่อย open เพื่อทำการนำฐานข้อมูลมาไว้ในระบบเสียก่อน แล้วจึงจะทำงานต่อไป

2. ในกรณีที่ผู้ใช้ไม่ทำการเลือก attribute โปรแกรมจะแสดงข้อความบอกความผิดพลาด แล้วให้ผู้ใช้ทำการเลือกใหม่

3. ค่าของ attribute ที่จะทำการหาค่าสูงสุดจะต้องไม่มีค่า domain type เป็นตัวอักษร

ชื่อรายการ Sum

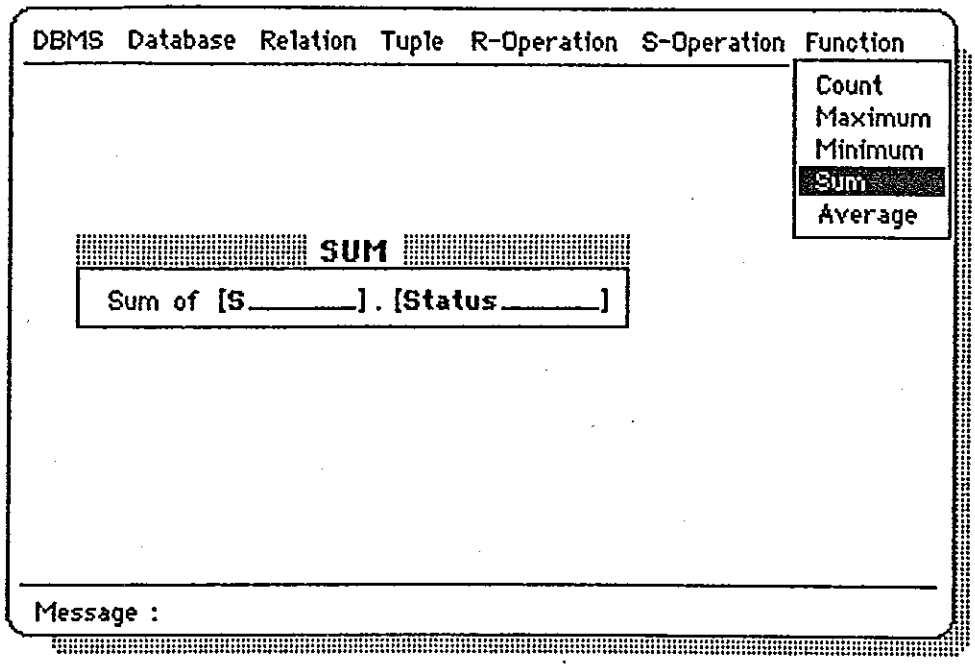
การทำงาน ทำการคำนวณหาผลรวมของค่าของ attribute ที่กำหนด

การใช้งาน จะแบ่งออกเป็น 2 ขั้นตอน คือ

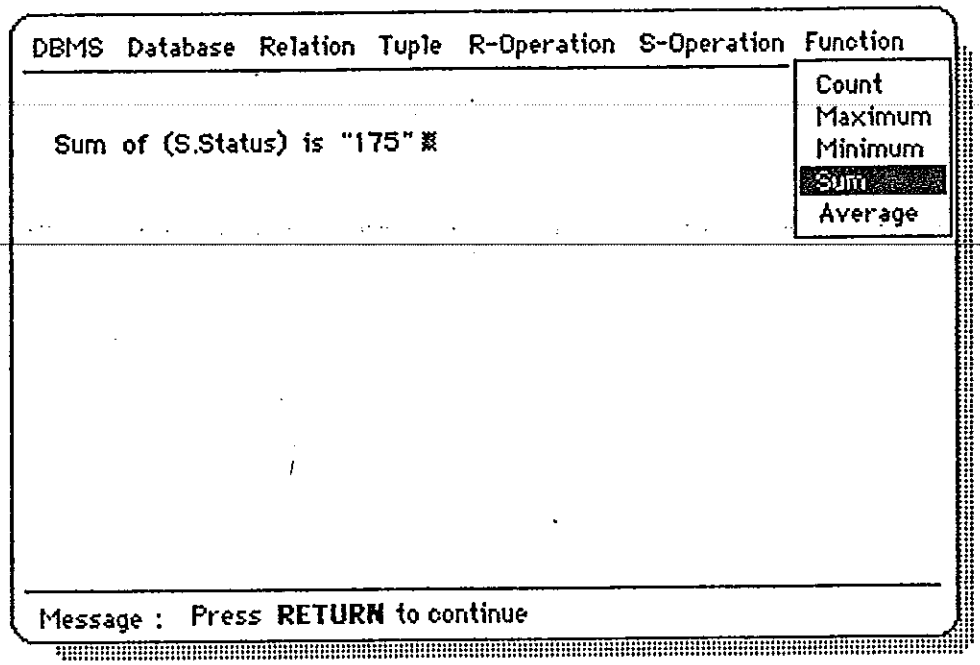
1. การรับข้อมูล เมื่อผู้ใช้เลือกรายการย่อยนี้ โปรแกรมจะแสดงหน้าต่างของ Sum เพื่อให้ผู้ใช้
   - ป้อนชื่อรีเลชั่น
   - ป้อนชื่อ attribute ที่ต้องการ โดยโปรแกรมจะแสดงหน้าต่าง attribute เพื่อให้ผู้ใช้ทำการเลือก

   จากรูปที่ 4-40 เป็นจอภาพในส่วนของการรับข้อมูล
2. การแสดงผล โปรแกรมจะแสดงผลการทำงาน ดังรูปที่ 4-41

จอภาพขณะใช้งาน



รูปที่ 4-40 จอภาพขณะใช้งานของรายการย่อย Sum



รูปที่ 4-41 จอภาพขณะใช้งานของรายการย่อย Sum

ข้อจำกัดและคำแนะนำเพิ่มเติม

1. ในกรณีที่ไม่มีฐานข้อมูลอยู่ในระบบ โปรแกรมจะเรียกการทำงานของรายการย่อย open เพื่อทำการนำฐานข้อมูลมาไว้ในระบบเสียก่อน แล้วจึงจะทำงานต่อไป

2. ในกรณีที่ผู้ใช้ไม่ทำการเลือก attribute โปรแกรมจะแสดงข้อความบอกความผิดพลาด แล้วให้ผู้ใช้ทำการเลือกใหม่

3. ค่าของ attribute ที่จะทำการหาผลรวมจะต้องเป็นเลขจำนวน

ชื่อรายการ Average

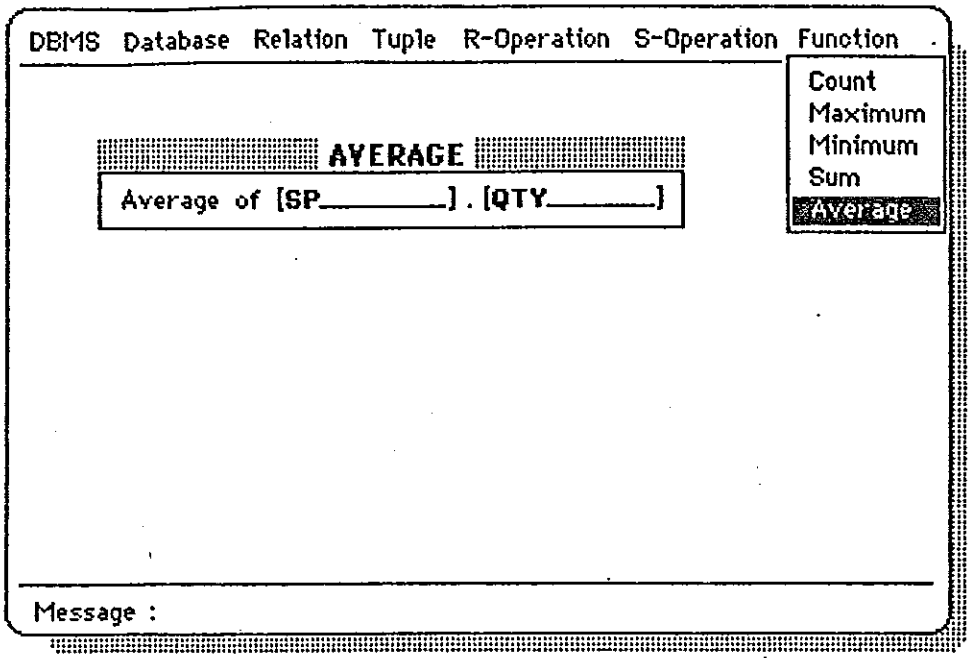
การทำงาน ทำการคำนวณค่าเฉลี่ยของ attribute ที่กำหนด

การใช้งาน จะแบ่งออกเป็น 2 ขั้นตอน คือ

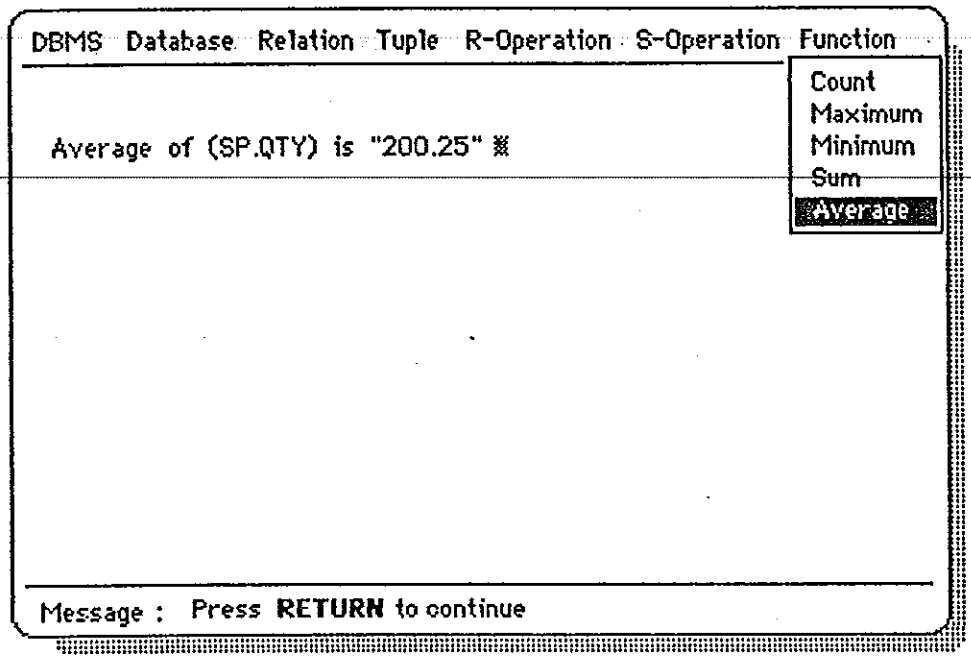
1. การรับข้อมูล เมื่อผู้ใช้เลือกรายการย่อยนี้ โปรแกรมจะแสดงหน้าต่างของ Average เพื่อให้ผู้ใช้
   - ป้อนชื่อรีเลชั่น
   - ป้อนชื่อ attribute ที่ต้องการ โดยโปรแกรมจะแสดงหน้าต่าง attribute เพื่อให้ผู้ใช้ทำการเลือก

   จากรูปที่ 4-42 เป็นจอภาพในส่วนของการรับข้อมูล
2. การแสดงผล โปรแกรมจะแสดงผลการทำงาน ดังรูปที่ 4-43

จอภาพขณะใช้งาน



รูปที่ 4-42 จอภาพขณะใช้งานของรายการย่อย Average



รูปที่ 4-43 จอภาพขณะใช้งานของรายการย่อย Average

ข้อจำกัดและคำแนะนำเพิ่มเติม

1. ในกรณีที่ไม่มีฐานข้อมูลอยู่ในระบบ โปรแกรมจะเรียกการทำงานของรายการย่อย open เพื่อทำการนำฐานข้อมูลมาไว้ในระบบเสียก่อน แล้วจึงจะทำงานต่อไป

2. ในกรณีที่ผู้ใช้ไม่ทำการเลือก attribute โปรแกรมจะแสดงข้อความบอกความผิดพลาด แล้วให้ผู้ใช้ทำการเลือกใหม่

3. ค่าของ attribute ที่จะทำการหาค่าเฉลี่ยจะต้องเป็นเลขจำนวน

บทที่ 5

สรุป

5.1 สรุปผลการวิจัย

ในการจัดทำวิทยานิพนธ์ครั้งนี้ ได้ทำการพัฒนาโปรแกรมคำสั่งปฏิบัติการฐานข้อมูลแบบมีช คณิตสัมพันธ์ ซึ่งโปรแกรมที่ได้ทำการพัฒนาขึ้นนี้ถือว่าเป็นโปรแกรมพื้นฐานส่วนหนึ่งในระบบจัดการฐานข้อมูล จากการออกแบบปทานุกรมข้อมูล โครงสร้างข้อมูล และการทำงานของโปรแกรมส่วนต่างๆ ได้เปิดโอกาสให้ผู้ที่สนใจสามารถนำไปเป็นโปรแกรมร่วมในการพัฒนาส่วนอื่นๆ ของระบบจัดการฐานข้อมูลได้ อาทิเช่น โปรแกรมสำหรับกำหนดโครงสร้าง รายละเอียด ความหมาย และความสัมพันธ์ของข้อมูลที่จะทำการจัดเก็บในฐานข้อมูล หรืออาจจะเป็นโปรแกรมสำหรับการปฏิบัติการกับข้อมูลโดยการป้อนคำถาม เป็นต้น

จากการทำวิทยานิพนธ์ครั้งนี้ พอจะสรุปผลได้ ดังนี้คือ

1. สามารถใช้โปรแกรมที่ได้เป็นโปรแกรมร่วม ในการพัฒนาส่วนอื่นๆ ของระบบจัดการฐานข้อมูลต่อไป

2. ได้โปรแกรมที่ง่าย และสะดวกแก่การใช้งานของผู้ใช้ เนื่องจากได้ใช้การจัดการแสดงผลทางจอภาพแบบหน้าต่าง (window manipulation)

3. สามารถนำโปรแกรมที่ได้ ไปใช้ในการเสริมทักษะสำหรับการเรียนการสอนวิชา Database Management Systems

4. สามารถนำแฟ้มข้อมูลที่ใช้อยู่กับโปรแกรมภาษา COBOL มาจัดเป็นระบบฐานข้อมูลได้ทันที โดยไม่ต้องพัฒนาโปรแกรมด้วยภาษา COBOL

5.2 ปัญหาในการจัดทำวิทยานิพนธ์

ปัญหาที่เกิดขึ้นในการทำวิทยานิพนธ์ครั้งนี้ โดยส่วนใหญ่จะเกิดจากการใช้โปรแกรมสำเร็จรูป ncurses และการแสดงผลการทำงานที่มีการหยุดชงักในบางครั้งทั้งนี้เนื่องจากคอมพิวเตอร์ VAX-11/785 ได้ใช้ระบบดำเนินงานที่มีการทำงานเป็นแบบ time sharing

5.3 ข้อเสนอแนะ

สำหรับโปรแกรมที่ได้ทำการพัฒนาขึ้นมานี้สามารถแสดงผลได้ดี บนจอภาพแบบ VT แต่สำหรับจอภาพที่ใช้กับเครื่อง microcomputer การแสดงผลจะไม่ดีเท่ากับจอภาพแบบ VT ทั้งนี้เนื่องจากจอภาพทั้ง 2 มีการกำหนดชุดของอักษรพิเศษและคำสั่งในการควบคุมจอภาพที่ต่างกัน ซึ่งผู้ที่สนใจอาจทำการกำหนดชุดของอักษรพิเศษขึ้นใหม่ได้ เพื่อให้การแสดงผลไม่มีความแตกต่างกัน

ภาคผนวก

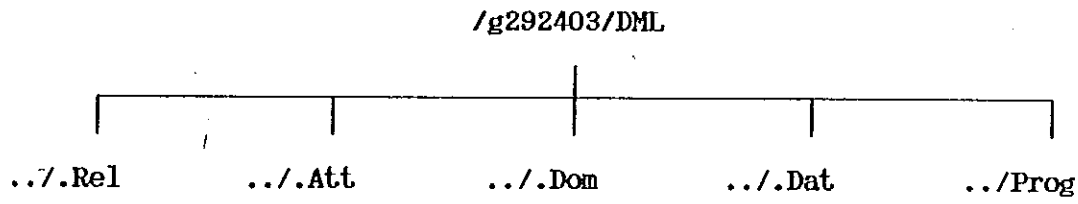
ภาคผนวก ก

Program listing

ภาคผนวก ข

ระบบการจัดเก็บในแฟ้มข้อมูลที่ใช้

สำหรับระบบที่ได้ทำการออกแบบ ได้จัดการเก็บแฟ้มข้อมูลใน directory ต่างๆ ซึ่งอาจแสดงได้ดังแผนภาพ ต่อไปนี้



Directory "/g292403/DML/.Rel"

directory นี้จะมีแฟ้มข้อมูลอยู่หลายแฟ้มข้อมูล ในแต่ละแฟ้มข้อมูลจะจัดเก็บข้อมูลเกี่ยวกับรายละเอียดต่าง ๆ ของฐานข้อมูลหนึ่ง ๆ (ชื่อแฟ้มข้อมูลก็คือชื่อฐานข้อมูล) ว่าประกอบด้วยรีเลชั่นอะไรบ้าง ชื่อแฟ้มข้อมูลจะลงท้ายด้วย ".rel" เสมอ

Directory "/g292403/DML/.Att"

directory นี้จะมีแฟ้มข้อมูลอยู่หลายแฟ้มข้อมูล ในแต่ละแฟ้มข้อมูลจะจัดเก็บข้อมูลเกี่ยวกับรายละเอียดของทุกรีเลชั่น (ชื่อแฟ้มข้อมูลระบุด้วยชื่อรีเลชั่น) ชื่อแฟ้มข้อมูลจะลงท้ายด้วย ".att" เสมอ

Directory "/g292403/DML/.Dom"

directory นี้จะมีแฟ้มข้อมูลอยู่หลายแฟ้มข้อมูล ซึ่งในแต่ละแฟ้มข้อมูลจะจัดเก็บรายละเอียดของโดเมนสำหรับ attribute (ชื่อแฟ้มข้อมูลระบุด้วยชื่อรีเลชั่น) ชื่อแฟ้มข้อมูลจะลงท้ายด้วย ".dom" เสมอ

ภาคผนวก ค

ตัวอย่างของผลที่ได้จากการทำงานของคำสั่งปฏิบัติการแบบต่างๆ

```
Command  : Display S
Relation : S
```

S#	Sname	St	City
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

```
Command  : Display P
Relation : P
```

P#	Pname	Color	St	City
P1	Nut	Red	12	Japan
P2	Bolt	Green	17	Paris
P3	Screw	Blue	17	Rome
P4	Screw	Red	14	London
P5	Cam	Blue	12	Paris
P6	Cog	Red	19	London

```
Command  : Display SP
Relation : SP
```

S#	P#	QTY
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400

Command : Cross S and P to TMP
Relation : TMP

S#	Sname	St	City	P#	Pname	Color	St	City
S1	Smith	20	London	P1	Nut	Red	12	Japan
S1	Smith	20	London	P2	Bolt	Green	17	Paris
S1	Smith	20	London	P3	Screw	Blue	17	Rome
S1	Smith	20	London	P4	Screw	Red	14	London
S1	Smith	20	London	P5	Cam	Blue	12	Paris
S1	Smith	20	London	P6	Cog	Red	19	London
S2	Jones	10	Paris	P1	Nut	Red	12	Japan
S2	Jones	10	Paris	P2	Bolt	Green	17	Paris
S2	Jones	10	Paris	P3	Screw	Blue	17	Rome
S2	Jones	10	Paris	P4	Screw	Red	14	London
S2	Jones	10	Paris	P5	Cam	Blue	12	Paris
S2	Jones	10	Paris	P6	Cog	Red	19	London
S3	Blake	30	Paris	P1	Nut	Red	12	Japan
S3	Blake	30	Paris	P2	Bolt	Green	17	Paris
S3	Blake	30	Paris	P3	Screw	Blue	17	Rome
S3	Blake	30	Paris	P4	Screw	Red	14	London
S3	Blake	30	Paris	P5	Cam	Blue	12	Paris
S3	Blake	30	Paris	P6	Cog	Red	19	London
S4	Clark	20	London	P1	Nut	Red	12	Japan
S4	Clark	20	London	P2	Bolt	Green	17	Paris
S4	Clark	20	London	P3	Screw	Blue	17	Rome
S4	Clark	20	London	P4	Screw	Red	14	London
S4	Clark	20	London	P5	Cam	Blue	12	Paris
S4	Clark	20	London	P6	Cog	Red	19	London
S5	Adams	30	Athens	P1	Nut	Red	12	Japan
S5	Adams	30	Athens	P2	Bolt	Green	17	Paris
S5	Adams	30	Athens	P3	Screw	Blue	17	Rome
S5	Adams	30	Athens	P4	Screw	Red	14	London
S5	Adams	30	Athens	P5	Cam	Blue	12	Paris
S5	Adams	30	Athens	P6	Cog	Red	19	London

Command : Select S on City == "London" to TMP
Relation : TMP

S#	Sname	St	City
S1	Smith	20	London
S4	Clark	20	London

Command : Join S and SP on S# >= S# to TMP
Relation : TMP

S#	Sname	St	City	S#	P#	QTY
S1	Smith	20	London	S1	P1	300
S1	Smith	20	London	S1	P2	200
S1	Smith	20	London	S1	P3	400
S1	Smith	20	London	S1	P4	200
S1	Smith	20	London	S1	P5	100
S1	Smith	20	London	S1	P6	100
S2	Jones	10	Paris	S1	P1	300
S2	Jones	10	Paris	S1	P2	200
S2	Jones	10	Paris	S1	P3	400
S2	Jones	10	Paris	S1	P4	200
S2	Jones	10	Paris	S1	P5	100
S2	Jones	10	Paris	S1	P6	100
S2	Jones	10	Paris	S2	P1	300
S2	Jones	10	Paris	S2	P2	400
S3	Blake	30	Paris	S1	P1	300
S3	Blake	30	Paris	S1	P2	200
S3	Blake	30	Paris	S1	P3	400
S3	Blake	30	Paris	S1	P4	200
S3	Blake	30	Paris	S1	P5	100
S3	Blake	30	Paris	S1	P6	100
S3	Blake	30	Paris	S2	P1	300
S3	Blake	30	Paris	S2	P2	400
S3	Blake	30	Paris	S3	P2	200
S4	Clark	20	London	S1	P1	300
S4	Clark	20	London	S1	P2	200
S4	Clark	20	London	S1	P3	400
S4	Clark	20	London	S1	P4	200
S4	Clark	20	London	S1	P5	100
S4	Clark	20	London	S1	P6	100
S4	Clark	20	London	S2	P1	300
S4	Clark	20	London	S2	P2	400
S4	Clark	20	London	S3	P2	200
S4	Clark	20	London	S4	P2	200
S4	Clark	20	London	S4	P4	300
S4	Clark	20	London	S4	P5	400
S5	Adams	30	Athens	S1	P1	300
S5	Adams	30	Athens	S1	P2	200
S5	Adams	30	Athens	S1	P3	400
S5	Adams	30	Athens	S1	P4	200
S5	Adams	30	Athens	S1	P5	100
S5	Adams	30	Athens	S1	P6	100
S5	Adams	30	Athens	S2	P1	300
S5	Adams	30	Athens	S2	P2	400
S5	Adams	30	Athens	S3	P2	200
S5	Adams	30	Athens	S4	P2	200
S5	Adams	30	Athens	S4	P4	300
S5	Adams	30	Athens	S4	P5	400

Command : Project P on P#,Color,Status, to TMP
Relation : TMP

P#	Color	St
P1	Red	12
P2	Green	17
P3	Blue	17
P4	Red	14
P5	Blue	12
P6	Red	19

Command : Project S on Sname,City to TMP
Relation : TMP

Sname	City
Smith	London
Jones	Paris
Blake	Paris
Clark	London
Adams	Athens

Command : Project SP on S#,P#, to TMP
Relation : TMP

S#	P#
S1	P1
S1	P2
S1	P3
S1	P4
S1	P5
S1	P6
S2	P1
S2	P2
S3	P2
S4	P2
S4	P4
S4	P5

Command : Project SP on S#,QTY to TMP
Relation : TMP

S#	QTY
S1	300
S1	200
S1	400
S1	100
S2	300
S2	400
S3	200
S4	200
S4	300
S4	400

Command : Project SP on S#,QTY to TMP
Relation : TMP

S#	QTY
S1	300
S1	200
S1	400
S1	200
S1	100
S1	100
S2	300
S2	400
S3	200
S4	200
S4	300
S4	400

เอกสารอ้างอิง

วัชรี พรหมทัตตเวที สมชาย หมื่นณรินทร์ และ ทวีศักดิ์ จันทร์วิทยานุชิต (2529), คัมภีร์ภาษา C, อิเลคทรอนิคส์ เวิร์ล จำกัด

A.F. Cardenas (1979), Data Base Management Systems, Allyn & Bacon, Inc., ISBN 0-205-19106-0

A.R. Kinderd (1980), Data Systems and management (An introduction to system analysis and design), 2nd Prentice-Hall, Inc., ISBN 0-13-19642-X

B.W. Kernighan and D.M. Ritchie (1987), The C programming language, Prentice-Hall, Inc.

C.J. Date (1986), An introduction to database system, 4th edition vol.1, Addison-Wesley Publishing Company, London, ISBN 0-201-19215-2

D.C. Tsichritzis and F.H. Lochovsky (1982), Data Model, Prentice-Hall, Inc., ISBN 0-13-196428-3

D. Maier (1983), The theory of relational database, Computer science press, Inc., ISBN 0-914894-42-0

E.B. Fernandez, R.C. Summers and C. Wood (1981), Database Security and Integrity, Addison-Wesley Publishing Company, ISBN 0-201-14467-0

G. Salton and M,J. McGill (1983), Introduction to modern information retrival, McGraw-Hill Book Company, ISBN 0-07-054484-0

J.D. Ullman (1980), Principle of database system, Computer Scince Press, ISBN 0-914894-13-7

J.S. Rohl (1984), Recursion via Pascal, Cambridge University Press, ISBN 0 521 269342

J.V. Duyn (1982), Developing a data dictionary system, Prentice-Hall, Inc., ISBN 0-13-204289-4

R. Krajewski (1984), 'Database Types', Byte Magazine, October, pp. 137-142

S. Atre (1980), Data Base: Structured Techniques for Design, Performance, and Management, John Wiley & Sons, Inc., ISBN 0193-9734

S. Leri and G. Pelagatti (1985), Distributed database principles and systems, McGraw-Hill Book Company, ISBN 0-07-0108293

S.R. Bourne (1983), The UNIX System, Addison-Wesley Publishing Company, ISBN 0-201-13791-7

T.J. Teorey and J.P. Fry (1982), Design of database structures, Prentice-Hall, Inc., ISBN 0-13-200097-0

Beckenbach and Drooyan (1969), Modern college algebra and Trigonometry, Wadsworth Publishing Company, Inc.

Holt, Rinehart and Winston (1983), UNIX programmer's manual, vol1, Bell laboratories, ISBN 0-03-061742-1

Directory "/g292403/DML/.Dat"

directory นี้จะมีแฟ้มข้อมูลอยู่หลายแฟ้มข้อมูล ซึ่งในแต่ละแฟ้มข้อมูลจะจัดเก็บข้อมูลจริงที่ใช้ในการปฏิบัติการ (ชื่อแฟ้มข้อมูลระบุด้วยชื่อรีเลชั่น) ชื่อแฟ้มข้อมูลจะลงท้ายด้วย ".dat" เสมอ

Directory "/g292403/DML/.Prog"

directory นี้ จะเก็บโปรแกรมที่ได้ทำการพัฒนาขึ้นมาทั้งหมด