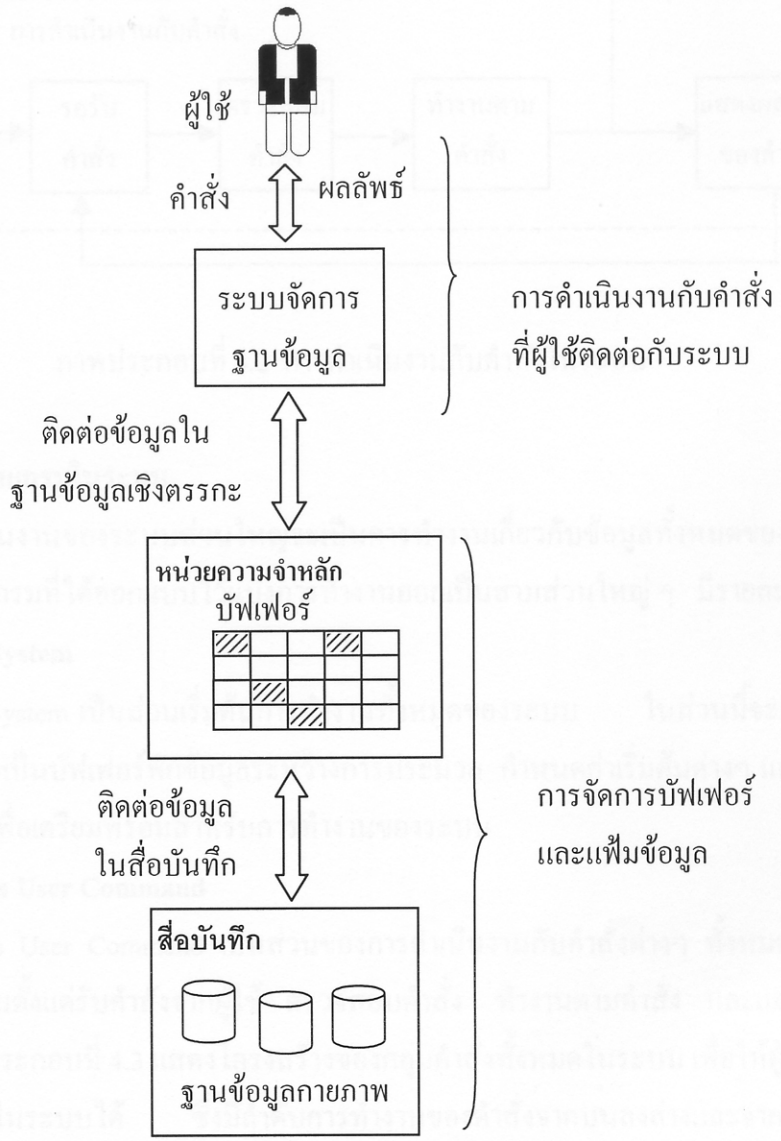
บทที่ 4

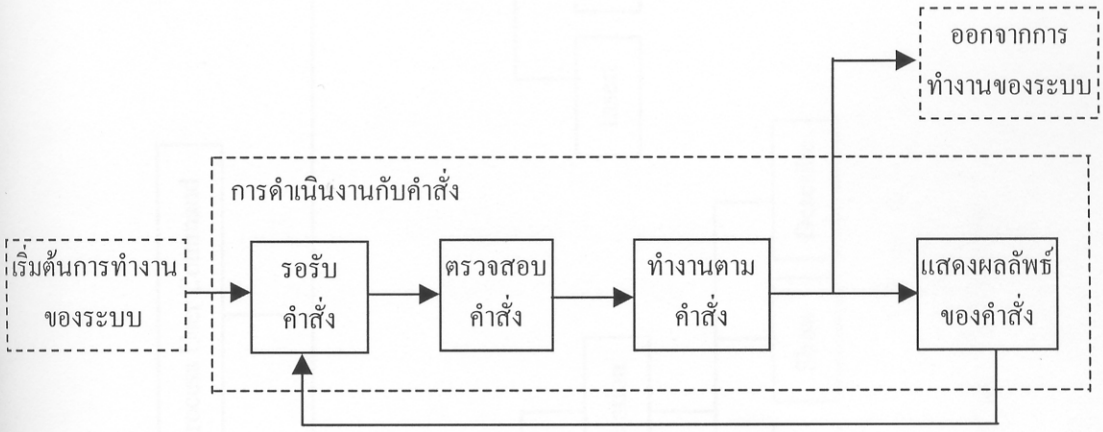
การดำเนินงานของโปรแกรมในระบบ

ระบบที่พัฒนาขึ้นเป็นระบบฐานข้อมูลเชิงสัมพันธ์ในส่วนการกำหนดโครงสร้างฐานข้อมูล และการดำเนินงานพื้นฐานกับฐานข้อมูล โดยระบบได้ออกแบบการจัดเก็บข้อมูลในแฟ้มข้อมูล และกำหนดบัฟเฟอร์ไว้สำหรับเป็นที่พักข้อมูลที่ต้องถูกเรียกใช้บ่อยๆ มาจัดเก็บไว้ในหน่วยความจำหลัก เพื่อลดการติดต่อกับแฟ้มข้อมูลบนสื่อบันทึกข้อมูล การทำงานในภาพรวมของระบบแสดงได้ดังภาพประกอบที่ 4.1



ภาพประกอบที่ 4.1 การทำงานของระบบที่พัฒนา

ระบบฐานข้อมูลส่วนจะต้องมีส่วนสำหรับการติดต่อใช้งานฐานข้อมูลจากผู้ใช้ทั่วไป คือ ส่วนของการกำหนดโครงสร้างฐานข้อมูลและชนิดข้อมูลที่จัดเก็บในฐานข้อมูล และการดำเนินงานพื้นฐานในการรับข้อมูลเข้า การลบข้อมูลออก หรือการปรับปรุงเปลี่ยนแปลงแก้ไขข้อมูลในฐานข้อมูล และส่วนประมวลผลข้อมูล ดังนั้นการดำเนินงานของระบบส่วนใหญ่จะเป็นการติดต่อกับผู้ใช้เพื่อดำเนินงานกับคำสั่งที่รับจากผู้ใช้ แสดงดังภาพประกอบที่ 4.2 เพื่อนำคำสั่งที่รับจากผู้ใช้มาตรวจสอบและทำงานตามคำสั่ง และแสดงผลลัพธ์ของคำสั่งให้ผู้ใช้ทราบ



ภาพประกอบที่ 4.2 การดำเนินงานกับคำสั่งในระบบ

4.1 โครงสร้างโปรแกรมในระบบ

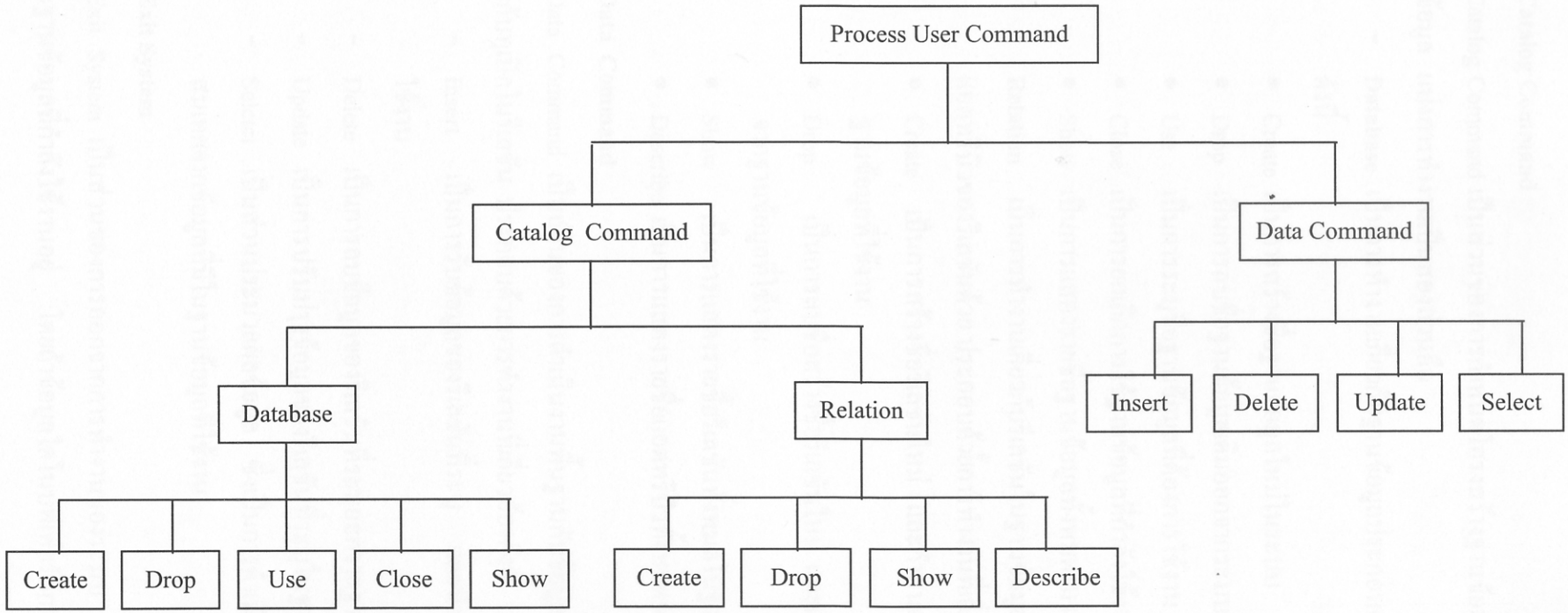
การดำเนินงานของระบบส่วนใหญ่จะเป็นการทำงานเกี่ยวกับข้อมูลทั้งหมดของฐานข้อมูลในระบบ โปรแกรมที่ได้ออกแบบไว้แบ่งการทำงานออกเป็นสามส่วนใหญ่ ๆ มีรายละเอียดดังนี้

1. Load System

Load System เป็นส่วนเริ่มต้นการทำงานทั้งหมดของระบบ ในส่วนนี้จะจัดเนื้อที่ในหน่วยความจำเพื่อเป็นบัฟเฟอร์พักข้อมูลระหว่างการประมวล กำหนดค่าเริ่มต้นต่างๆ และเปิดแฟ้มปทานุกรมข้อมูลเพื่อเตรียมพร้อมสำหรับการทำงานของระบบ

2. Process User Command

Process User Command เป็นส่วนของการดำเนินงานกับคำสั่งต่างๆ ทั้งหมดของระบบ ในส่วนนี้จะเริ่มต้นตั้งแต่รับคำสั่งจากผู้ใช้ ตรวจสอบคำสั่ง ทำงานตามคำสั่ง และแสดงผลลัพธ์ของคำสั่ง ภาพประกอบที่ 4.3 แสดงโครงสร้างของกลุ่มคำสั่งทั้งหมดในระบบ เพื่อให้ผู้ใช้สามารถใช้งานฐานข้อมูลในระบบได้ ซึ่งมีลำดับการทำงานของคำสั่งจากบนลงล่างและจากซ้ายไปขวา โดยแบ่งการทำงานออกเป็นสองส่วนดังนี้



ภาพประกอบที่ 4.3 โครงสร้างของกลุ่มคำสั่งในระบบ

Catalog Command

Catalog Command เป็นส่วนของการกำหนดโครงสร้างฐานข้อมูล และชนิดข้อมูลที่จัดเก็บในฐานข้อมูล แบ่งการทำงานเป็นสองส่วนคือ

- **Database** เป็นการทำงานเกี่ยวกับฐานข้อมูลประกอบด้วยการทำงานที่เกี่ยวข้องดังนี้
  - Create เป็นการสร้างชื่อฐานข้อมูลใหม่ในระบบ
  - Drop เป็นการลบชื่อฐานข้อมูลเดิมออกจากระบบ
  - Use เป็นการระบุชื่อฐานข้อมูลที่ต้องการใช้งาน
  - Close เป็นการยกเลิกการใช้ฐานข้อมูลที่กำลังใช้งาน
  - Show เป็นการแสดงรายชื่อฐานข้อมูลทั้งหมดในระบบ
- **Relation** เป็นการทำงานเกี่ยวกับรีเลชันในฐานข้อมูล รวมทั้งการทำงานกับแอตทริบิวของรีเลชันด้วย ประกอบด้วยการทำงานที่เกี่ยวข้องดังนี้
  - Create เป็นการสร้างชื่อรีเลชันใหม่ และกำหนดแอตทริบิวของรีเลชันในฐานข้อมูลที่ใช้งาน
  - Drop เป็นการลบชื่อสร้างชื่อรีเลชันใหม่ และแอตทริบิวของรีเลชันออกจากฐานข้อมูลที่ใช้งาน
  - Show เป็นการแสดงรายชื่อรีเลชันทั้งหมดในฐานข้อมูลที่ใช้งาน
  - Describe เป็นการแสดงรายชื่อแอตทริบิวทั้งหมดของรีเลชันที่ระบุ

Data Command

Data Command เป็นส่วนของการดำเนินงานพื้นฐานกับข้อมูลในฐานข้อมูล ซึ่งเป็นการทำงานเกี่ยวกับทุเปิลในรีเลชัน ประกอบด้วยการทำงานที่เกี่ยวข้องดังนี้

- Insert เป็นการรับข้อมูลของรีเลชันที่ระบุ และนำไปจัดเก็บในฐานข้อมูลที่ใช้งาน
- Delete เป็นการลบข้อมูลของรีเลชันที่ระบุออกจากฐานข้อมูลที่ใช้งาน
- Update เป็นการปรับปรุงข้อมูลของรีเลชันที่ระบุในฐานข้อมูลที่ใช้งาน
- Seletct เป็นส่วนประมวลผลข้อมูล ซึ่งเป็นการดำเนินงานเพื่อให้ได้มาซึ่งสารสนเทศจากข้อมูลที่มีในฐานข้อมูลที่ใช้งาน

3. Exit System

Exit System เป็นส่วนของการออกจากการทำงานของระบบ โดยจะปิดแฟ้มปทานุกรมข้อมูล และฐานข้อมูลที่กำลังใช้งานอยู่ โดยถ้าข้อมูลใดในบัฟเฟอร์มีการเปลี่ยนแปลงก็จะบันทึก

กลับลงดิสก์ให้ก่อนจะหยุดการทำงานของโปรแกรม พร้อมทั้งยกเลิกการจองเนื้อที่ของบัฟเฟอร์ในหน่วยความจำหลักให้ด้วย

4.2 มอดูลและการทำงาน

จากโครงสร้างการดำเนินงานของระบบ จึงได้พัฒนาโปรแกรมโดยแบ่งการทำงานออกเป็นงานย่อย ๆ เรียกว่า มอดูล (module) โดยแต่ละมอดูลจะทำงานเฉพาะอย่าง และนำมอดูลทั้งหมดมาเชื่อมโยงให้ทำงานร่วมกัน ดังนั้นการทำงานของระบบทั้งหมดจะประกอบด้วยมอดูลจำนวนมาก แต่สามารถจัดประเภทของมอดูลทั้งหมดที่มีออกเป็นสองประเภทใหญ่คือ

1. **Function Module** เป็นมอดูลที่ใช้ในการทำงานหลักของแต่ละคำสั่งของโปรแกรม มีหลายมอดูล ดังแสดงในตารางที่ 4.1
2. **Utility Module** เป็นมอดูลที่ใช้เพื่ออำนวยความสะดวกในการพัฒนาโปรแกรม มอดูลเหล่านี้จะถูกเรียกใช้โดยมอดูลหลักของการทำงาน มอดูลที่พัฒนาขึ้นเพื่ออำนวยความสะดวกในการพัฒนาโปรแกรมประกอบด้วย

มอดูลดำเนินงานกับแฟ้มข้อมูล

การจัดเก็บข้อมูลทั้งหมดในระบบจะจัดเก็บในแฟ้มข้อมูลต่างๆ ภายในระบบฐานข้อมูล โดยมีโครงสร้างแฟ้มข้อมูลตามที่ได้กล่าวไว้แล้วในหัวข้อ 3.1 พร้อมทั้งมีการจัดเตรียมบัฟเฟอร์ในหน่วยความจำสำหรับจัดเก็บข้อมูลระหว่างการประมวลผล มอดูลที่เกี่ยวข้องกับการทำงานในการจัดเก็บข้อมูลในแฟ้มข้อมูลมีจำนวนมาก โดยแบ่งออกเป็นส่วนๆ ดังนี้

- มอดูลเกี่ยวกับบัฟเฟอร์ ดังแสดงในตารางที่ 4.2
- มอดูลเกี่ยวกับแฟ้มข้อมูลในระบบ ดังแสดงในตารางที่ 4.3
- มอดูลเกี่ยวกับบล็อกข้อมูลในแฟ้มข้อมูล ดังแสดงในตารางที่ 4.4
- มอดูลเกี่ยวกับรายชื่อรีเลชันในแฟ้มข้อมูล ดังแสดงในตารางที่ 4.5
- มอดูลเกี่ยวกับข้อมูลของรีเลชันในแฟ้มข้อมูล ดังแสดงในตารางที่ 4.6

มอดูลดำเนินงานกับโครงสร้างข้อมูล $B_m$tree

การจัดเก็บค่าคีย์ที่กำหนดขึ้นในรีเลชันต่างๆ จะใช้โครงสร้างข้อมูล $B_m$tree ในระบบอาจมีโครงสร้างข้อมูล $B_m$tree จำนวนวนมาก แต่โครงสร้างข้อมูล $B_m$tree หนึ่งๆ จะจัดเก็บค่าคีย์ชุดเดียวกันจากรีเลชันเดียวกันเท่านั้น ตารางที่ 4.7 แสดงมอดูลเกี่ยวกับโครงสร้างข้อมูล $B_m$tree

มอดูลดำเนินงานกับปทานุกรมข้อมูล

มอดูลเกี่ยวกับฐานข้อมูล และรีเลชันที่ผู้ใช้สร้างขึ้น ประกอบด้วย

- มอดูลเกี่ยวกับรายชื่อฐานข้อมูลที่จัดเก็บไว้ในรีเลชันชื่อ db.cat ดังแสดงในตารางที่ 4.8
- มอดูลเกี่ยวกับรายชื่อรีเลชันที่จัดเก็บไว้ในรีเลชันชื่อ rel.cat ดังแสดงในตารางที่ 4.9
- มอดูลเกี่ยวกับรายชื่อแอตทริบิวที่จัดเก็บไว้ในรีเลชันชื่อ att.cat ดังแสดงในตารางที่ 4.10
- มอดูลเกี่ยวกับการกำหนดเงื่อนไขบังคับ ได้แก่ กำหนดคีย์หลัก คีย์รอง และเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิวของคีย์นอกจากค่าคีย์หลักในอีกรีเลชันหนึ่ง ซึ่งจะจัดเก็บไว้ในรีเลชันชื่อ cons.cat colcons.cat และ refcons.cat ดังแสดงในตารางที่ 4.11

มอดูลดำเนินงานกับข้อมูลในฐานข้อมูล

ในฐานข้อมูลหนึ่งๆ ผู้ใช้สามารถสร้างรีเลชันขึ้นได้จำนวนมาก ข้อมูลของแต่ละรีเลชันจะจัดเก็บเป็นทุเปิล การเพิ่มทุเปิลใหม่ของรีเลชัน และปรับปรุงแก้ไขหรือลบทุเปิลของรีเลชันสามารถทำได้โดยผ่านการทำงานของมอดูลต่อไปนี้

- มอดูลเกี่ยวกับการเพิ่มทุเปิลของรีเลชัน ดังแสดงในตารางที่ 4.12
- มอดูลเกี่ยวกับการปรับปรุงแก้ไขหรือลบทุเปิลของรีเลชัน ดังแสดงในตารางที่ 4.13

มอดูลดำเนินงานกับค่าแอตทริบิว

ข้อมูลที่ถูกจัดเก็บในแต่ละรีเลชัน ประกอบด้วยข้อมูลของแอตทริบิวจำนวนหนึ่ง ซึ่งสามารถจัดเก็บค่าของแอตทริบิวได้หลายชนิดขึ้นอยู่กับการสร้างรีเลชันของผู้ใช้ ข้อมูลชนิดต่างๆ จะจัดเก็บเป็นชุดเลขฐานสองตามรหัสแอสกีต่างๆ ไว้ในสายอักขระขนาดความยาวจำนวนหนึ่ง ดังนั้นข้อมูลวันที่และข้อมูลตัวเลขจะอยู่ในรูปของข้อมูลที่ผู้ใช้ไม่สามารถอ่านเข้าใจได้ ซึ่งการเข้าถึงการจัดเก็บ และการเรียกค้นข้อมูลชนิดต่างๆ จะต้องอาศัยการทำงานของมอดูลต่อไปนี้

- มอดูลเกี่ยวกับการจัดการค่าแอตทริบิว ดังแสดงในตารางที่ 4.14
- มอดูลเกี่ยวกับการจัดการข้อมูลตัวเลข ดังแสดงในตารางที่ 4.15
- มอดูลเกี่ยวกับการจัดการข้อมูลวันที่ ดังแสดงในตารางที่ 4.16
- มอดูลเกี่ยวกับการจัดการสายอักขระ ดังแสดงในตารางที่ 4.17

- มอดูลเกี่ยวกับการเปรียบเทียบค่าแอตทริบิว ดังแสดงในตารางที่ 4.18
- มอดูลเกี่ยวกับค่าแอตทริบิวที่เป็นคีย์ ดังแสดงในตารางที่ 4.19

มอดูลดำเนินงานกับทุเปิลของรีเลชัน

ข้อมูลของแต่ละรีเลชันจะจัดเก็บเป็นทุเปิล ซึ่งการเข้าถึง เพิ่ม แก้ไข หรือลบทุเปิลของรีเลชันจะต้องอาศัยการทำงานของมอดูลต่อไปนี้

- มอดูลเกี่ยวกับการทำงานพื้นฐานกับทุเปิลของรีเลชัน ดังแสดงในตารางที่ 4.20
- มอดูลเกี่ยวกับการเข้าถึงทุเปิลของรีเลชันแบบเรียงลำดับ ดังแสดงในตารางที่ 4.21
- มอดูลเกี่ยวกับการปรับปรุงแก้ไขหรือลบทุเปิลของรีเลชันแบบเรียงลำดับ ดังแสดงในตารางที่ 4.22

มอดูลดำเนินงานกับการรับและวิเคราะห์คำสั่ง

เนื่องจากการใช้งานโปรแกรมผู้ใช้ต้องพิมพ์คำสั่ง เพื่อให้โปรแกรมทำงานตามความต้องการของผู้ใช้ ซึ่งคำสั่งที่ผู้ใช้พิมพ์จะต้องถูกต้องตามลักษณะคำสั่งที่ได้ออกแบบไว้ในระบบ มอดูลที่เกี่ยวข้องกับการตรวจสอบและแปลความหมายของแต่ละคำสั่งที่รับจากผู้ใช้ ประกอบด้วย

- มอดูลเกี่ยวกับการวิเคราะห์คำสั่งเพื่อแยกเป็นโทเคน ดังแสดงในตารางที่ 4.23
- มอดูลเกี่ยวกับจัดเก็บข้อมูลในระหว่างรับคำสั่ง ดังแสดงในตารางที่ 4.24 โดยจะทำงานร่วมกับชุดคำสั่งย่อยชื่อ yyparse() ที่ได้จากการใช้โปรแกรม YACC

มอดูลดำเนินงานกับลิงสิสต์

ข้อมูลต่างที่ใช้ในโปรแกรมจะถูกจัดเก็บไว้ในโครงสร้างข้อมูลประเภทต่างๆ เพื่อความสะดวกในการเขียนโปรแกรมและการส่งผ่านข้อมูลระหว่างมอดูลต่างๆ ภายในโปรแกรม ข้อมูลที่จัดเก็บในโครงสร้างข้อมูลแต่ละประเภทมีจำนวนมากจึงมีการจัดเก็บเป็นลิงค์ลิสต์ไว้ในโครงสร้างข้อมูล LIST ซึ่งจะเก็บจำนวนโหนดที่จัดเก็บในลิสต์ไว้ด้วย มอดูลที่เกี่ยวข้องประกอบด้วย

- มอดูลเกี่ยวกับการจองเนื้อที่ในหน่วยความจำสำหรับลิงค์ลิสต์ ดังแสดงในตารางที่ 4.25
- มอดูลเกี่ยวกับการนำโหนดใหม่ไปต่อท้ายลิงค์ลิสต์ ดังแสดงในตารางที่ 4.26
- มอดูลเกี่ยวกับการคืนเนื้อที่ในหน่วยความจำสำหรับลิงค์ลิสต์ ดังแสดงในตารางที่ 4.27

มอดูลเกี่ยวกับการแสดงผลบนจอภาพ

การแสดงผลบนจอภาพเป็นการแสดงผลการทำงาน ซึ่งอาจเป็นเพียงการแสดงข้อความเพื่อแจ้งข้อผิดพลาดหรือผลการทำงาน หรืออาจเป็นการแสดงข้อมูลแต่ละทุเปิลของรีเลชันในรูปของตาราง ซึ่งมีมอดูลที่เกี่ยวข้องดังนี้

- มอดูลเกี่ยวกับแสดงข้อความบนจอภาพ ดังแสดงในตารางที่ 4.28
- มอดูลเกี่ยวกับการแสดงรายการทุเปิลของรีเลชันในรูปของตาราง ดังแสดงในตารางที่ 4.29

มอดูลเกี่ยวกับการเริ่มต้นเข้าสู่การทำงานของโปรแกรมหรือออกจากโปรแกรม

ในการทำงานของโปรแกรมจะต้องมีการเตรียมข้อมูลบางอย่างที่เกี่ยวข้องกับการใช้งานภายในโปรแกรม ดังแสดงในตารางที่ 4.30

มอดูลเกี่ยวกับการตั้งชื่อต่าง ๆ ในโปรแกรม

ในการทำงานของโปรแกรมจะต้องมีมอดูลสำหรับการกำหนดชื่อขึ้นในโปรแกรมเพื่อความสะดวกในการใช้งาน ดังแสดงในตารางที่ 4.31

ตารางที่ 4.1 มอดูลการทำงานหลักของระบบ

ชื่อมอดูล	**หน้าที่การทำงาน**
LoadSystem()	จัดเตรียมบัฟเฟอร์สำหรับปทานุกรมข้อมูล และฐานข้อมูล เปิดแฟ้มปทานุกรมข้อมูล และเข้าสู่การทำงานของระบบ
DbUse() STRNAME *dbname	เปิดใช้งานฐานข้อมูลที่ต้องการ ซึ่งในขณะใดขณะหนึ่งระบบจะอนุญาตให้เปิดใช้งานฐานข้อมูลได้เพียงฐานข้อมูลเดียวเท่านั้น ถ้ามีฐานข้อมูลเดิมเปิดอยู่จะทำการปิดฐานข้อมูลเดิมก่อน พร้อมทั้งปรับปรุงข้อมูลของฐานข้อมูลเดิมที่มีการเปลี่ยนแปลงในบัฟเฟอร์บันทึกลงดิสก์ให้โดยอัตโนมัติ
DbClose()	ปิดฐานข้อมูลที่กำลังใช้งานอยู่ พร้อมทั้งปรับปรุงข้อมูลที่มีการเปลี่ยนแปลงในบัฟเฟอร์บันทึกลงดิสก์ให้โดยอัตโนมัติ
DbCreate() STRNAME dbname	สร้างฐานข้อมูลใหม่ในระบบ

ตารางที่ 4.1 มอดูลการทำงานหลักของระบบ (ต่อ)

ชื่อมอดูล	หน้าที่การทำงาน
DbDrop() STRNAME dbname	ลบรายชื่อฐานข้อมูลเก่า และรายละเอียดทั้งหมดที่เกี่ยวข้องออกจากระบบ
DbShow()	แสดงรายชื่อฐานข้อมูลที่สร้างขึ้นในระบบ
RelCreate() STRNAME relname LIST listatt,lpkey,lskey,lref	สร้างรีเลชันใหม่ พร้อมข้อมูลแอตทริบิว และรายละเอียดที่เกี่ยวข้องไว้ในฐานข้อมูลที่กำลังใช้งานอยู่
RelDrop() STRNAME relname	ลบรีเลชันเก่า พร้อมทั้งข้อมูลแอตทริบิว และรายละเอียดที่เกี่ยวข้องกับรีเลชันนั้นออกจากฐานข้อมูลที่กำลังใช้งานอยู่
RelShow()	แสดงรายชื่อรีเลชันที่สร้างขึ้นในฐานข้อมูลที่กำลังใช้งานอยู่
AttShow() STRNAME relname	แสดงรายชื่อแอตทริบิวของรีเลชันที่ระบุ
InsertTuple() STRNAME relname LIST lattname,lvalue	เพิ่มทุเปิลใหม่ในรีเลชัน
UpdateTuple() STRNAME relname LIST lattname,lvalue,lcond	ปรับปรุงข้อมูลทุเปิลบางส่วนของรีเลชัน ตามชื่อแอตทริบิวและค่าที่ระบุ อาจปรับปรุงครั้งละมากกว่าหนึ่งทุเปิลขึ้นอยู่กับเงื่อนไขที่ระบุ
DeleteTuple() STRNAME relname LIST lattname,lcond	ลบทุเปิลออกจากรีเลชัน อาจลบครั้งละมากกว่าหนึ่งทุเปิลขึ้นอยู่กับเงื่อนไขที่ระบุ
SelectTuple() STRNAME relname LIST lattname,lcond	แสดงรายการทุเปิลในรีเลชันตามชื่อแอตทริบิวที่ระบุ อาจแสดงมากกว่าหนึ่งทุเปิลขึ้นอยู่กับเงื่อนไขที่ระบุ
ExitSystem()	ปิดแฟ้มปทานุกรมข้อมูล และปิดฐานข้อมูลที่กำลังใช้งานอยู่ด้วย โดยจะปรับปรุงข้อมูลที่มีการเปลี่ยนแปลงในบัฟเฟอร์บันทึกลงดิสก์ให้โดยอัตโนมัติ พร้อมทั้งยกเลิกการจองเนื้อที่บัฟเฟอร์ในหน่วยความจำหลัก และออกจากระบบ

ตารางที่ 4.2 มอดูลเกี่ยวกับบัฟเฟอร์

ชื่อมอดูล	หน้าที่การทำงาน
UpdateInfoLRU() REPLACE *replace int pageno	ปรับปรุงข้อมูลที่ใช้ในการคัดเลือกเพจเพื่อแทนที่ด้วยข้อมูลในบล็อกใหม่ที่จะอ่านจากดิสก์ในกรณีบัฟเฟอร์เต็มด้วยวิธีการแทนที่เพจแบบ LRU
PickPage() REPLACE *replace int numpage	หาตำแหน่งเพจบนบัฟเฟอร์เพื่อเตรียมไว้สำหรับข้อมูลใหม่ที่จะนำมาจัดเก็บในบัฟเฟอร์ กรณีบล็อกข้อมูลที่ต้องการไม่ได้อยู่บนบัฟเฟอร์
PinPage() REPLACE *replace int pageno ,numpage	กำหนดสถานะของเพจว่าอยู่ในระหว่างใช้งาน โดยกำหนดสถานะเป็นสถานะ pin
UnPinPage() REPLACE *replace int pageno, numpage	ยกเลิกสถานะการใช้งานเพจข้อมูลบนบัฟเฟอร์ โดยเปลี่ยนสถานะเพจจากสถานะ pin เป็น referenced
AddPageInBuffer() int numpage, blockid, pageno PMT *pgmaptbl	จัดเก็บข้อมูลบล็อกใหม่ในบัฟเฟอร์
FindPageInBuffer() PMT *pgmaptbl int blockid	ค้นหาบล็อกข้อมูลที่ต้องการว่าจัดเก็บอยู่บนบัฟเฟอร์หรือไม่
RemovePageFromBuffer() int numpage, pageno PMT *pgmaptbl	เคลื่อนย้ายเพจออกจากบัฟเฟอร์
BufMgrAllocateBuf() BUFMGR *bufmgr int numpages	จัดเตรียมเนื้อที่ในหน่วยความจำหลักสำหรับใช้เป็นบัฟเฟอร์ไว้พักข้อมูลระหว่างประมวลผล
BufMgrInit() BUFMGR *bufmgr int numpages	กำหนดค่าเริ่มต้นสำหรับการทำงานในส่วนการจัดการบัฟเฟอร์
BufMgrFreeBuf() BUFMGR *bufmgr	คืนเนื้อที่หน่วยความจำหลักที่จัดเตรียมไว้ใช้เป็นบัฟเฟอร์พักข้อมูลระหว่างประมวลผล
BufMgrNewPage() BUFMGR *bufmgr int *firstblockid, numblock PAGE** firstblock	ขอใช้ข้อมูลบล็อกใหม่ในแฟ้มข้อมูลโดยและกำหนดบิตสถานะของบล็อกเป็น 1 และกำหนดสถานะของเพจข้อมูลของบล็อกนั้นเป็นสถานะ pin ด้วย

ตารางที่ 4.2 มอดูลเกี่ยวกับบัฟเฟอร์ (ต่อ)

ชื่อมอดูล	หน้าที่การทำงาน
BufMgrPinPage() BUFMGR *bufmgr int pinblockid, emptypage PAGE** page	เมื่อต้องการใช้ข้อมูลบล็อกใดบนดิสก์ จะตรวจสอบก่อนว่าบล็อกข้อมูลที่ต้องการอยู่บนบัฟเฟอร์ไหม ถ้าไม่อยู่จะหาตำแหน่งเพจบนบัฟเฟอร์เพื่อจัดเก็บข้อมูลใหม่ที่จะอ่านจากดิสก์ และกำหนดสถานะของเพจเป็นสถานะ pin ไว้
BufMgrUnpinPage() BUFMGR *bufmgr int blockid,dirty	ยกเลิกการใช้งานเพจบนบัฟเฟอร์
BufMgrFreePage() BUFMGR *bufmgr int blockid	คืนหรือยกเลิกการใช้บล็อกข้อมูลในแฟ้มข้อมูลและกำหนดบิตสถานะของบล็อกเป็น 0 และถ้าข้อมูลบล็อกนั้นอยู่ในบัฟเฟอร์ด้วยจะต้องย้ายเพจข้อมูลบของบล็อกนั้นออกจากบัฟเฟอร์ด้วย
BufMgrFlushAllPage() BUFMGR *bufmgr	เคลื่อนย้ายเพจทั้งหมดออกจากบัฟเฟอร์ และถ้ามีข้อมูลใดในบัฟเฟอร์มีการเปลี่ยนแปลงจะบันทึกกลับลงดิสก์ให้ด้วย

ตารางที่ 4.3 มอดูลเกี่ยวกับแฟ้มข้อมูล

ชื่อมอดูล	หน้าที่การทำงาน
DbFileNew() char* fname int numblock DBFILE *dbfile	สร้างแฟ้มข้อมูลใหม่บนดิสก์
DbFileOpen() char* fname DBFILE *dbfile	เปิดแฟ้มข้อมูลเก่าบนดิสก์
DbFileClose() DBFILE *dbfile	ปิดแฟ้มข้อมูลบนดิสก์ ถ้ามีข้อมูลใดจัดเก็บอยู่ในบัฟเฟอร์และมีการเปลี่ยนแปลงจะทำการบันทึกกลับลงดิสก์ด้วย
DbFileInitBuffer() BUFMGR *bufmgr int numpage	จัดเตรียมเนื้อที่บนหน่วยความจำหลักสำหรับใช้เป็นบัฟเฟอร์พักข้อมูลของแฟ้มข้อมูลบนดิสก์ และกำหนดค่าเริ่มต้นสำหรับการทำงานในส่วนการจัดการบัฟเฟอร์
DbFileFreeBuffer() BUFMGR *bufmgr	คืนเนื้อที่บนหน่วยความจำหลักที่จัดเตรียมไว้เป็นบัฟเฟอร์พักข้อมูลของแฟ้มข้อมูลบนดิสก์

ตารางที่ 4.4 มอดูลเกี่ยวกับบล็อกข้อมูลในแฟ้มข้อมูล

ชื่อมอดูล	หน้าที่การทำงาน
DataBlockInit() DBLOCK *datablock BLOCKID blockid	กำหนดค่าเริ่มต้นของบล็อกข้อมูลใหม่
DataBlockAvailableSpace() DBLOCK *datablock	คำนวณหาเนื้อที่ว่างบนบล็อกข้อมูล
DataBlockFstRec() DBLOCK *datablock RID* fstrid	ค้นหาเรคอร์ดแรกในบล็อกข้อมูล
DataBlockNextRec () DBLOCK* datablock RID currid, *nextrid	ค้นหาเรคอร์ดถัดไปในบล็อกข้อมูล
DataBlockGetRec() DBLOCK *datablock RID rid char* recptr int* reclen	สำเนาข้อมูลของเรคอร์ดจากบล็อกข้อมูล ตามหมายเลขเรคอร์ดที่ต้องการ
DataBlockGetRecPtr() DBLOCK *datablock RID rid char** recptr int* reclen	หาตำแหน่งข้อมูลที่จัดเก็บในบัฟเฟอร์ตามหมายเลขเรคอร์ดที่ต้องการ
DataBlockInsertRec() DBLOCK *datablock char* recptr int reclen RID* rid	เพิ่มเรคอร์ดใหม่ในบล็อกข้อมูล
DataBlockDelRec() DBLOCK *datablock RID rid	ลบเรคอร์ดเก่าออกจากบล็อกข้อมูล
DataBlockUpdRec() DBLOCK *datablock RID rid char* newrecptr int newlen	ปรับปรุงแก้ไขข้อมูลของเรคอร์ดเก่าในบล็อกข้อมูล กรณีขนาดความยาวของคอร์ดเปลี่ยนไป แต่จะต้องไม่มีการเปลี่ยนแปลงหมายเรคอร์ด

ตารางที่ 4.4 มอดูลเกี่ยวกับบล็อกข้อมูลในแฟ้มข้อมูล (ต่อ)

ชื่อมอดูล	หน้าที่การทำงาน
ReadBlock() int fd, max_blockid BLOCKID blockid BLOCK *blockptr	อ่านบล็อกข้อมูลจากแฟ้มข้อมูลบนดิสก์
WriteBlock() int fd, max_blockid BLOCKID blockid BLOCK *blockptr	บันทึกบล็อกข้อมูลจัดเก็บลงในแฟ้มข้อมูลบนดิสก์
AllocateBlock() BUFMGR *bufmgr BLOCKID* startblockid int numblk	ขอใช้บล็อกใหม่ของแฟ้มข้อมูลบนดิสก์ โดยกำหนดบิตสถานะของบล็อกเป็น 1
DeallocateBlock() BUFMGR *bufmgr BLOCKID startblockid int numblock	ขอคืนหรือยกเลิกการใช้บล็อกข้อมูลของแฟ้มข้อมูลบนดิสก์ โดยกำหนดบิตสถานะของบล็อกเป็น 0
SetBit() BUFMGR *bufmgr BLOCKID* startblockid int numblock,bit	กำหนดบิตสถานะของบล็อกเป็น 0 หรือ 1

ตารางที่ 4.5 มอดูลเกี่ยวกับรายชื่อรีเลชันในแฟ้มข้อมูล

ชื่อมอดูล	หน้าที่การทำงาน
DeleteNameInDirBlk() char* name BUFMGR *bufmgr	ลบรายชื่อรีเลชันออกจากส่วนรายชื่อรีเลชันในแฟ้มข้อมูล
GetNameInDirBlk() char* name BLOCKID* fstblkid BUFMGR *bufmgr	ค้นรายชื่อรีเลชัน และหมายเลขบล็อกข้อมูลแรกที่จัดเก็บข้อมูลของรีเลชัน จากส่วนรายชื่อรีเลชันในแฟ้มข้อมูล
AddNameInDirBlk() char* name BLOCKID* fstblkid BUFMGR *bufmgr	เพิ่มรายชื่อรีเลชันใหม่และหมายเลขบล็อกข้อมูลแรกที่จัดเก็บข้อมูลของรีเลชัน ในส่วนรายชื่อรีเลชันในแฟ้มข้อมูล

ตาราง 4.6 มอดูลเกี่ยวกับข้อมูลของรีเลชันในแฟ้มข้อมูล

ชื่อมอดูล	หน้าที่การทำงาน
DataSpaceInit() DATASPACE *dataspace char* name BUFMGR *bufmgr	กำหนดค่าเริ่มต้นในการเข้าถึงข้อมูลของรีเลชัน โดยจะค้นหาชื่อรีเลชันจากข้อมูลในส่วนรายชื่อรีเลชัน ถ้าไม่พบชื่อรีเลชันที่ต้องการจะเพิ่มชื่อรีเลชันใหม่ให้ด้วย
DataSpaceDrop() DATASPACE *dataspace	ลบข้อมูลทุกทุเปิลของรีเลชันออกจากระบบ
DataSpaceInsertRec() DATASPACE *dataspace char* recptr int reclen RID* outrid	เพิ่มทุเปิลใหม่ของรีเลชัน
DataSpaceUpdRec() DATASPACE *dataspace char* recptr int reclen RID* rid	ปรับปรุงข้อมูลทุเปิลของรีเลชัน ในตำแหน่งหมาเลขเรคอร์ด rid
DataSpaceDelRec() DATASPACE *dataspace RID rid	ลบทุเปิลของรีเลชัน ในตำแหน่งหมาเลขเรคอร์ด rid
DataSpaceGetRec() DATASPACE *dataspace char* recptr int *reclen RID rid	เข้าถึงข้อมูลทุเปิลของรีเลชัน ในตำแหน่งหมาเลขเรคอร์ด rid
DataSpaceRecCt() DATASPACE *dataspace int *recct	นับจำนวนทุเปิลทั้งหมดของรีเลชัน
DataSpaceNewDataBlock() DATASPACE *dataspace HBLKINFO *hblockinfo	เพิ่มบล็อกข้อมูลใหม่ในการจัดเก็บข้อมูลของรีเลชัน
DataSpaceFindDataBlock () DATASPACE *dataspace RID rid, * Dblockrid DBLOCK**Hblock,**Dblock BLOCKID *Hblockid,Dblockid	ค้นหาบล็อกข้อมูลใหม่ในการจัดเก็บข้อมูลของรีเลชันที่มีข้อมูลของหมายเลขเรคอร์ดตามค่า rid ที่จัดเก็บอยู่ในบล็อกนั้น

ตาราง 4.6 มอดูลเกี่ยวกับข้อมูลของรีเลชันในแฟ้มข้อมูล (ต่อ)

ชื่อมอดูล	หน้าที่การทำงาน
SeqAccessInit() SEQACCESS *seqacc DATASPACE *dataspace BUFMGR *bufmgr	กำหนดค่าเริ่มต้นในการเข้าถึงข้อมูลของรีเลชันแบบตามลำดับเก็บ โดยจะค้นหาข้อมูลบล็อกข้อมูลแรกที่ใช้จัดเก็บข้อมูลของรีเลชันให้ด้วย เพื่อใช้ในการค้นหาทุเปิลของรีเลชัน
SeqAccessNextRec() SEQACCESS *seqacc RID *rid char* recptr int* reclen	ค้นหาข้อมูลทุเปิลถัดไปของรีเลชันแบบตามลำดับการจัดเก็บ
SeqAccessClose() SEQACCESS *seqacc	ยกเลิกการเข้าถึงข้อมูลของรีเลชันแบบตามลำดับ

ตารางที่ 4.7 มอดูลเกี่ยวกับโครงสร้างข้อมูล $B_m$tree

ชื่อมอดูล	หน้าที่การทำงาน
BtreeBuild() char *indexname int nokey KTYPE *ktype BUFMGR*bufmgr	สร้างโครงสร้างข้อมูลในการจัดเก็บดัชนีใหม่โดยกำหนดชนิดและขนาดของข้อมูลที่จะนำมาจัดเก็บเป็นดัชนีในโครงสร้างข้อมูล $B_m$tree พร้อมทั้งจำนวนคีย์กรณีเป็นคีย์ร่วมด้วย
BtreeDropIndex() char *indexname BUFMGR*bufmgr	ลบข้อมูลดัชนีทั้งหมดในกลุ่มดัชนีเดียวกันตามค่า indexname
BtreeSearchKey() char * indexname, *keyvalue RID *recrid BUFMGR *bufmgr	ค้นหาค่าคีย์และตำแหน่งหมายเลขเรคอร์ดของทุเปิลในรีเลชันที่มีค่าคีย์ที่สัมพันธ์กัน จากโครงสร้างข้อมูล $B_m$tree
BtreeInsertKey() char *indexname, *newkey RID recrid BUFMGR *bufmgr	เพิ่มค่าคีย์ใหม่ในโครงสร้างข้อมูล $B_m$tree
BtreeDeleteKey() char *indexname, *keyvalue RID *recrid BUFMGR *bufmgr	ลบค่าคีย์เก่าออกจากโครงสร้างข้อมูล $B_m$tree

ตารางที่ 4.7 มอดูลเกี่ยวกับโครงสร้างข้อมูล $B_m$tree (ต่อ)

ชื่อมอดูล	หน้าที่การทำงาน
BtreeSetKeyStr() char * keyptr, *value long len	กำหนดค่าคีย์ชนิดอักขระ
BtreeSetKeyInt() char *keyptr long value	กำหนดค่าคีย์ชนิดเลขจำนวนเต็ม
BtreeSetKeyReal() char *keyptr double value	กำหนดค่าคีย์ชนิดเลขทศนิยม
CompareKeys() char *newkey,*kvalue KTYPE *ktype int nokey	เปรียบเทียบค่าคีย์ทั้งสองตามชนิดข้อมูลของค่าคีย์
NodeScan () char *keyvalue BTREE *bthdr BTNODE *trnode int *p	ค้นหาตำแหน่งค่าคีย์ที่ต้องการจากโหนดปัจจุบัน
Search() DATASPACE *dataspace BTREE *bthdr BTNODE **bt char *keyvalue int *p RID*rid	ค้นหาตำแหน่งโหนดและตำแหน่งที่จัดเก็บค่าคีย์ต้องการค้นหาในโหนดจากโครงสร้างข้อมูล $B_m$tree
GetBtreeHeader() DATASPACE *dataspace char *indexname,*bthdr int *len RID*rid BUFMGR *bufmgr	อ่านโหนดที่ใช้เก็บข้อกำหนดของโครงสร้างข้อมูล $B_m$tree และชนิดและขนาดของค่าคีย์ที่จัดเก็บ เพื่อเป็นข้อมูลในการเข้าถึงคีย์ต่าง ๆ
PushInNode() BTNODE *bt char *key RID rbt int p, ksize	เพิ่มค่าคีย์ใหม่ในโหนดปัจจุบัน

ตารางที่ 4.7 มอดูลเกี่ยวกับโครงสร้างข้อมูล $B_m$tree (ต่อ)

ชื่อมอดูล	หน้าที่การทำงาน
SetNewParentToChild() DATASPACE *dataspace BTNODE *bt,*tmp int size,nodelen RID rid	ปรับปรุงค่าตัวชี้ไปยังโหนดพ่อของโหนดปัจจุบัน
SplitNode() DATASPACE *dataspace BTREE *bthdr BTNODE *bt, *tmp, *yp char **key int p RID *rbt, rid	แตกโหนดออกเป็นสองโหนดและกำหนดค่าคีย์ให้แต่ละโหนดๆ ละครึ่งหนึ่งของค่าคีย์เดิมทั้งหมด และใส่ค่าคีย์ใหม่ให้กับโหนดในตำแหน่งเหมาะสม
NodeInit() BTNODE **bt int nodesize	กำหนดค่าเริ่มต้นของโหนดใหม่
PtrPositionInParent() RID btrid BTNODE *parent int size	ค้นหาตำแหน่งตัวชี้ในโหนดพ่อที่ชี้มายังโหนดปัจจุบัน
MoveFromParentToLeft() DATASPACE *dataspace BTNODE *parent, *l, *r, *tmp int p, ksize,nodelen	*ย้ายค่าคีย์จากโหนดพ่อไปยังโหนดลูกทางซ้าย*
MoveFromParentToRight() DATASPACE *dataspace BTNODE *parent, *r, *tmp int p, ksize,nodelen RID ptr	ย้ายค่าคีย์จากโหนดพ่อไปยังโหนดลูกทางขวา
Remove() BTNODE *bt int p,ksize	ลบค่าคีย์ออกจากโหนด
Successor() DATASPACE *dataspace BTNODE *bt,*ptr int p, ksize,nodesize RID btrid,*rid	ค้นหาค่าคีย์ในที่มีค่าน้อยที่สุดจากโหนดลูกทางซ้ายทั้งหมดของโหนดปัจจุบันและย้ายค่าคีย์มาจัดเก็บไว้ในโหนดปัจจุบัน

ตารางที่ 4.7 มอดูลเกี่ยวกับโครงสร้างข้อมูล $B_m$tree (ต่อ)

ชื่อมอดูล	หน้าที่การทำงาน
MoveFromRight() DATASPACE *dataspace BTNODE *parent, *l, *r, *tmp int p, ksize,nodesize	ย้ายค่าคีย์จากโหนดทางซ้ายมาจัดเก็บไว้ในโหนดพ่อของโหนดปัจจุบัน
MoveFromLeft() DATASPACE *dataspace BTNODE *parent,*l,*bt,*tmp int p, ksize,nodesize	ย้ายค่าคีย์จากโหนดทางขวามาจัดเก็บไว้ในโหนดพ่อของโหนดปัจจุบัน
Combine() DATASPACE *dataspace BTNODE *parent, *l, *r, *tmp int p,bthdrlen BTREE *bthdr RID bthdrrid	รวมค่าคีย์ของโหนดปัจจุบันเข้ากับค่าคีย์ของโหนดทางซ้ายหรือขวาตามความเหมาะสม
Restore() DATASPACE *dataspace BTREE *bthdr BTNODE **bt char *btnode int p,bthdrlen RID *btrid,bthdrrid	ปรับปรุงค่าของคีย์ในโหนด เพื่อให้เป็นเป็นไปข้อกำหนดของโครงสร้างข้อมูล $B_m$tree เนื่องจากจำนวนค่าคีย์ของโหนดน้อยกว่าจำนวนคีย์ที่น้อยที่สุดที่แต่ละโหนดในโครงสร้างข้อมูล $B_m$tree ต้องมี
ReadNode() DATASPACE *dataspace char *data RID rid int len	อ่านโหนดจากโครงสร้างข้อมูล $B_m$tree
WriteNode() DATASPACE *dataspace char *data , int len RID rid	บันทึกโหนดเก่าในโครงสร้างข้อมูล $B_m$tree
GetNewNode() DATASPACE *dataspace char *data , int len RID *rid	บันทึกโหนดใหม่ในโครงสร้างข้อมูล $B_m$tree
DeleteNode() DATASPACE *dataspace RID delrid	ลบโหนดเก่าออกจากโครงสร้างข้อมูล $B_m$tree

ตาราง 4.8 มอดูลเกี่ยวกับรายชื่อฐานข้อมูลในปทานุกรมข้อมูล

ชื่อมอดูล	หน้าที่การทำงาน
DbSearch() char *dbname DBNODE *dbnode	ค้นหามีฐานข้อมูลที่ต้องการใช้งานอยู่ในระบบหรือไม่
SearchUsedDb()	ค้นหาชื่อฐานข้อมูลที่กำลังเปิดให้งานอยู่
IsUsedDb() char *dbname	ตรวจสอบว่าฐานข้อมูลที่ต้องการใช้งานเป็นฐานข้อมูลที่กำลังเปิดใช้งานอยู่หรือไม่
MakeDbTuple() char *data, db int norel long dbcreate	กำหนดค่าแอตทริบิวให้กับทุเปิลเพื่อจัดเก็บในรีเลชันที่จัดเก็บรายชื่อฐานข้อมูล
MakeDbNode() char *data DBNODE *dbnode	กำหนดรายละเอียดข้อมูลในโครงสร้างข้อมูล dbnode จากข้อมูลในทุเปิลของฐานข้อมูล
UpdateDbNode() DBNODE *dbnode	ปรับปรุงทุเปิลของรีเลชันที่จัดเก็บรายชื่อฐานข้อมูลทั้งหมดในระบบ โดยใช้รายละเอียดข้อมูลจากโครงสร้างข้อมูล dbnode
DbDropAllData() DBNODE * dbnode	ลบข้อมูลทั้งหมดที่เกี่ยวข้องกับฐานข้อมูลที่ต้องการ

ตาราง 4.9 มอดูลเกี่ยวกับรายชื่อรีเลชันในปทานุกรมข้อมูล

ชื่อมอดูล	หน้าที่การทำงาน
ValidAttName() COLNODE *ChkList ATTNODE *ListAtt int nullflag	ตรวจสอบรายชื่อแอตทริบิวต่าง ๆ ที่อ้างถึงว่ามีอยู่ในรายชื่อแอตทริบิวของรีเลชันที่ต้องการสร้างขึ้นหรือไม่
CheckValidAtt() ATTNODE *ListAtt COLNODE *pkey, *skey REFNODE *refnode	ตรวจสอบว่าแอตทริบิวต่างๆ ที่ถูกอ้างถึงในการกำหนดคีย์หลัก คีย์รอง และคีย์นอก มีอยู่ในรายชื่แอตทริบิวของรีเลชันที่ต้องการสร้างขึ้นหรือไม่ ซึ่งจะตรวจสอบก่อนส่งข้อมูลไปทำงานส่วนของการสร้างรีเลชันใหม่ในฐานข้อมูล
DupAttKey() COLNODE * key, *chk char keytype	ตราวจสอบว่าแอตทริบิวที่กำหนดเป็นคีย์รองเป็นแอตทริบิวที่กำหนดอยู่แล้วในคีย์หลักหรือไม่ ซึ่งจะตรวจสอบก่อนส่งข้อมูลไปทำงานส่วนของการสร้างรีเลชันใหม่ในฐานข้อมูล

ตาราง 4.9 มอดูลเกี่ยวกับรายชื่อรีเลชันของปทานุกรมข้อมูล (ต่อ)

ชื่อมอดูล	หน้าที่การทำงาน
RelSearch() char *rel,*key RID dbid,*relrid RELNODE *relnode	ค้นหาชื่อรีเลชันมีอยู่ฐานข้อมูล
MakeTupleRel() char *data,rel RID dbid int noatt, tuplesize,notuple long relcreate, rellstupd	ใส่ค่าสำหรับทุเปิลของรีเลชัน
MakeRelNode() char *data RELNODE *relnode RID *relid	กำหนดรายละเอียดข้อมูลในโครงสร้างข้อมูล relnode จากข้อมูลในทุเปิลของรีเลชัน
UpdateRelNode() RELNODE *rel	ปรับปรุงทุเปิลของรีเลชันที่จัดเก็บรายชื่อรีเลชันของฐานข้อมูลจากรายละเอียดข้อมูลในโครงสร้างข้อมูล relnode
SetRelKey() char *db,rel,key	กำหนดค่าคีย์สำหรับการค้นหาชื่อรีเลชัน
DropAllRelation() char *dbname	ลบรีเลชันทั้งหมดภายในฐานข้อมูล
DropAllKeyInRel () RID *relid	ลบข้อมูลดัชนีดัชนีทั้งหมดของรีเลชัน
DropAllAttInRel () RID *relid	ลบรายชื่อแอตทริบิวทั้งหมดของรีเลชัน

ตาราง 4.10 มอดูลเกี่ยวกับรายชื่อแอตทริบิวในปทานุกรมข้อมูล

ชื่อมอดูล	หน้าที่การทำงาน
AttSearch() char *db,*rel, *att,*key RID *attrid ATTNODE *attnode	ค้นชื่อแอตทริบิวอยู่ในรีเลชัน
MakeAttNode() char *data ATTNODE *attnode *RID *attid*	กำหนดรายละเอียดของโครงสร้างข้อมูล attnode จากข้อมูลในทุเปิลของแอตทริบิว

ตาราง 4.10 มอดูลเกี่ยวกับรายชื่อแอตทริบิวในปทานุกรมข้อมูล (ต่อ)

ชื่อมอดูล	หน้าที่การทำงาน
MakeAttInfoNode() char *data ATTINFO *attinfonode RID *attid	กำหนดรายละเอียดของโครงสร้างข้อมูล attinfo จากข้อมูลในทุเปิลของแอตทริบิว
SetAttKey() char *db,rel,att,key	กำหนดค่าคีย์สำหรับการค้นหาชื่อแอตทริบิว
AttStoreValue() char*attvalue,attdomain,*value int attlen	จัดเก็บค่าแอตทริบิวตามชนิดและขนาดความยาวข้อมูลของแอตทริบิว
AttStoreValueStr() char*attvalue,attdomain,*value int attlen	จัดเก็บค่าแอตทริบิว ถ้าข้อมูลที่จัดเก็บเป็นชนิดตัวเลขหรือวันที่มีการแปลงค่าแอตทริบิวที่เป็นข้อความให้เป็นค่าข้อมูลที่ถูกต้องตามชนิดของข้อมูลของแอตทริบิวด้วย
AttValueGet() char *tuple ATTINFO *attinfo int *len	อ่านค่าแอตทริบิวจากทุเปิลตามชนิดและขนาดของแอตทริบิว
AttValueGetStr() char *tuple ATTINFO *attinfo	อ่านค่าแอตทริบิวจากทุเปิลตามชนิดและขนาดของแอตทริบิว ถ้าข้อมูลเป็นค่าตัวเลขหรือวันที่จะแปลงเป็นข้อมูลชนิดอักขระที่ผู้ใช้สามารถอ่านเข้าใจได้ด้วย
GetLenOfStr() char *str unsigned int *l	อ่านค่าความยาวของแอตทริบิวที่จัดเก็บในเรคอร์ดความยาวแปรได้
StoreLenOfStr() char *str unsigned int l	บันทึกค่าความยาวของแอตทริบิวจัดเก็บไว้ในเรคอร์ดความยาวแปรได้
StoreValueandLen() Unsigned char *ptr char *value int *datalen	บันทึกค่าแอตทริบิวและค่าความยาวของแอตทริบิวจัดเก็บไว้ในเรคอร์ดความยาวแปรได้

ตาราง 4.11 มอดูลเกี่ยวกับเงื่อนไขบังคับในปทานุกรมข้อมูล

ชื่อมอดูล	หน้าที่การทำงาน
ConsRelGetConsId( ) RID *rel char constype EXPRNODE *cond	ค้นหาชื่อเงื่อนไขบังคับของรีเลชัน ตามชนิดเงื่อนไขบังคับที่ต้องการ และจัดเก็บชื่อเงื่อนไขบังคับทั้งหมดไว้ใน cond
ConsSave() RID *relid char type LIST *listcol	บันทึกเงื่อนไขบังคับ เมื่อมีการสร้างรีเลชันใหม่
ConsRelInsert() RID *relid, *consid char type COLNODE *col	บันทึกทุเปิลลงในรีเลชัน cons.cat
ColConsSave() RID *consid char type COLNODE *col	บันทึกทุเปิลลงในรีเลชัน colcons.cat
RefConsSave() RID *consid, *refcons char ondelete	บันทึกทุเปิลลงในรีเลชัน refcons.cat
ConsBuildKey() char *indexname LIST *lattkey	สร้างข้อมูลดัชนีเมื่อมีการกำหนดคีย์ของรีเลชัน
ConsRelDrop() RID *relid	ลบข้อมูลการกำหนดเงื่อนไขบังคับทั้งหมดของรีเลชัน
CheckRefAttKey() RID *relid REFNODE *refnode ATTNODE *attnode	ตรวจสอบชื่อแอตทริบิวทั้งหมดที่อ้างถึงในการกำหนดเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิวว่าแอตทริบิวดังกล่าวเป็นคีย์หลักของรีเลชันที่อ้างถึงหรือไม่ พร้อมกับตรวจสอบชนิดข้อมูลของคีย์หลักในรีเลชันที่อ้างถึงตรงกับชนิดข้อมูลของแอตทริบิวที่กำหนดเป็นคีย์นอกหรือไม่
CheckReference() DBNODE *dbnode REFNODE *RefNode ATTNODE *attnode	ตรวจสอบความถูกต้องของการกำหนดเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิวทั้งหมดของรีเลชันที่สร้างขึ้นใหม่ โดยจะตรวจสอบรายชื่อแอตทริบิวว่าอยู่ในรีเลชันที่อ้างถึงหรือไม่ พร้อมทั้งตรวจสอบชนิดของข้อมูลของแอตทริบิวที่อ้างอิงกันด้วย

ตารางที่ 4.12 มอดูลเกี่ยวกับการเพิ่มทุเปิลของรีเลชัน

ชื่อมอดูล	หน้าที่การทำงาน
CheckLengthInteger() int prec char *str	ตรวจสอบสายอักขระของข้อมูลเลขจำนวนเต็มใน str ว่ามีจำนวนตำแหน่งตัวเลขทั้งหมด อยู่ในช่วงที่ถูกต้องหรือไม่
CheckLengthFloat() int prec, scale char *str	ตรวจสอบสายอักขระของข้อมูลเลขทศนิยมใน str ว่ามีจำนวนตำแหน่งตัวเลขทั้งหมด และจำนวนตำแหน่งตัวเลขหลังจุดทศนิยมอยู่ในช่วงที่ถูกต้องหรือไม่
CheckAttDomain() ATTINFO *att LIST *litem int noatt	ตรวจสอบชนิดข้อมูลและขนาดข้อมูลค่าในโครงสร้างข้อมูล itemnode ทุกรายการ ว่าถูกต้องตามที่กำหนดในโครงสร้างข้อมูล att แต่ละตัวหรือไม่
CheckDomainRange() ATTINFO *att ITEMNODE *itemnode	ตรวจสอบค่าแอตทริบิวในโครงสร้างข้อมูล itemnode ว่าชนิดข้อมูลและขนาดข้อมูลถูกต้องตามที่กำหนดใน att หรือไม่
CheckAttNullable() ATTNODE *attnode	ตรวจสอบว่าแอตทริบิวอนุญาตให้ใส่ค่าว่างหรือไม่
CheckConsRef() RELNODE *relnode char *tuple LIST *latt	ตรวจสอบการอ้างอิงค่าแอตทริบิว
ConsRefChild() char *tuple RELNODE *relnod LIST *latt RID *consid, *refcons	ตรวจสอบเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิว(ถ้ามี) โดยตรวจสอบว่าค่าแอตทริบิวที่เป็นคีย์นอกของรีเลชันมีค่าอยู่ในคีย์หลักของรีเลชันที่อ้างถึงหรือไม่
CheckKeyCons() char * tuple RELNDOE *relnode LIST *latt RID *rid	ตรวจสอบทุกๆ ค่าคีย์หลักและคีย์รอง (ถ้ามี) ของทุเปิลที่จะเพิ่มว่าซ้ำกับค่าคีย์ของทุเปิลเก่าในรีเลชันหรือไม่

ตารางที่ 4.12 มอดูลเกี่ยวกับการเพิ่มทุเปิลในรีเลชันของฐานข้อมูล(ต่อ)

ชื่อมอดูล	หน้าที่การทำงาน
ConsKeyValue() LIST *latt RID *rid EXPRNODE *condconsid char *indexname,*key	ค้นหาค่าคีย์เพื่อเปรียบเทียบค่าคีย์ว่าซ้ำกับค่าคีย์ของทุเปิลเก่าในรีเลชันหรือไม่
MakeAttNodeAll() char *db,*rel ATTNODE *attnode	ค้นหาแอตทริบิวทั้งหมดของรีเลชันในฐานข้อมูลเพื่อจัดเก็บในโครงสร้างข้อมูล attnode
MakeAttInfoAll() char *db,*rel ATTINFO *attinfo	ค้นหาแอตทริบิวทั้งหมดของรีเลชันในฐานข้อมูลเพื่อจัดเก็บในโครงสร้างข้อมูล attinfo
SetListAttInsert() ATTNODE *attnode ATTINFO *attinfo int noatt	สำเนาข้อมูลจากโครงสร้างข้อมูล attnode จัดเก็บในโครงสร้างข้อมูล attinfo
CheckAttExist() LIST *ListAtt ATTNODE **lattnode int noatt	ตรวจสอบราชื่อแอตทริบิวของค่าข้อมูลในทุเปิลที่ต้องการเพิ่มว่ามีอยู่ในรายการแอตทริบิวของรีเลชันหรือไม่
StoreTuple() char *tu ple ITEMNODE *itemptr ATTNODE *att ATTINFO *attinfo int noatt	นำข้อมูลในโครงสร้างข้อมูล itemptr ทุกรายการ มาจัดเก็บเป็นทุเปิลข้อมูลตามลักษณะข้อมูลที่กำหนดใน attinfo ตามลำดับ ถ้าแอทริบิวใดไม่มีค่าใน itemptr จะกำหนดค่าจาก default ของ att หรือกำหนดเป็นค่าว่างให้
SetAttList() LIST *ListAtt ATTINFO*attinfo,*newattinfo int noatt	จัดลำดับโครงสร้างข้อมูล attinfo ใหม่ให้เป็นไปตามรายชื่อแอตทริบิวใน listatt ที่ผู้ใช้ระบุในคำสั่ง select

ตารางที่ 13 มอดูลเกี่ยวกับการปรับปรุงแก้ไขหรือลบทุเปิลของรีเลชัน

ชื่อมอดูล	หน้าที่การทำงาน
DeleteTupleAll() DBFILE *dbfile RELNODE *relnode	ลบทุกทุเปิลของรีเลชัน
DeleteTupleCond() RELNODE *relnode LIST *lcond	ลบทุเปิลในรีเลชันตามเงื่อนไขที่ต้องการ
RefConsDeleteAll() DBFILE *dbfile RID *relid char *relname LIST lrelcons int chkondelete	ค้นหารีเลชันอื่นทั้งหมดในฐานข้อมูลที่มีแอตทริบิวเป็นคีย์นอกและอ้างถึงคีย์หลักของรรีเลชันที่ต้องการลบแบบต่อเนื่องกันในกรณีลบทุกทุเปิลในรีเลชัน
RefConsGetChild() RELNODE *lrel, *relnode KEYCONS **keycons	ค้นหารีเลชันอื่นทั้งหมดในฐานข้อมูลที่มีแอตทริบิวเป็นคีย์นอกและอ้างถึงคีย์หลักของรรีเลชันที่ต้องการลบแบบต่อเนื่องกัน
FindRelAndAtt() RELNODE *lrel RID *relid	ค้นหารายชื่อรีเลชันที่ต้องการว่ามีในลิสต์รายชื่อรีเลชันในโครงสร้างข้อมูล RELNODE ของ lrel หรือไม่ ถ้าไม่มีให้ค้นหารายละเอียดทั้งหมดของรีเลชันมาจัดเก็บใน lrel ด้วย
RelGetAttInfo() RELNODE *relnode RID *relid	ค้นหารายละเอียดทั้งหมดของรีเลชัน รวมทั้งรายชื่อแอตทริบิวของรีเลชันตามค่ารหัสรีเลชันใน relid
GetKeyIdConsCol() LIST *latt RID *consid unsigned char *nokey	กำหนดรหัสแอตทริบบิวที่เป็นคีย์ของรีเลชันตามตำแหน่งของรายชื่อแอตทริบิวในโครงสร้างข้อมูล latt
SetKeyConsAttId() KEYCONS *pkeycons LIST *latt	กำหนดรหัสแอตทริบิวที่เป็นคีย์ของรีเลชันตามตำแหน่งของชื่อแอตทริบิวในโครงสร้างข้อมูล latt เพื่อตรวจสอบค่าคีย์ในขณะปรับปรุงทุเปิล
ExprGetAttId() EXPRNODE *ChkList ATTINFO *att int noatt	ตรวจสอบแอตทริบิวที่กำหนดในเงื่อนไข ว่ามีในรีเลชันหรือไม่ พร้อมกับกำหนดรหัสแอตทริบิวที่กำหนดเป็นค่าคีย์ตามตำแหน่งของรายชื่อแอตทริบิวของโครงสร้างข้อมูล att ให้ด้วย

ตารางที่ 4.13 มอดูลเกี่ยวกับการปรับปรุงแก้ไขหรือลบทุเปิลของรีเลชัน (ต่อ)

ชื่อมอดูล	หน้าที่การทำงาน
SecondaryKeyDeleteAll() RID *relid	ลบข้อมูลดัชนีของคีย์รองทั้งหมดในรีเลชันของรหัสรีเลชันใน relid ในกรณีลบทุเปิลของรีเลชัน
UpdAttIsKey() unsigned char *updattid,*keyid unsigned char *upd int n1, n2,*n3	ตรวจสอบแอตทริบิวที่เป็นคีย์ตรงกับแอตทริบิวที่ต้องการปรับปรุงทุเปิลของรีเลชันหรือไม่
RefConsGetChildUpd() RELNODE *lrel , *relnode unsigned char *updattid int noid KEYCONS **keycons RID *refrel	ค้นหารีเลชันอื่นทั้งหมดในฐานข้อมูลที่มีแอตทริบิวเป็นคีย์นอกและอ้างถึงคีย์หลักของรรีเลชันที่ต้องการปรับปรุงแก้ไข
RefConsGetChildUpd() RELNODE *lrel , *relnode unsigned char *updattid int noid PARENTCONS **keycons RID *refrel	ค้นหารีเลชันอื่นทั้งหมดในฐานข้อมูลที่คีย์นอกของรีเลชันที่ต้องการปรับปรุงทุเปิลอ้างค่าคีย์นอกจากคีย์หลักของรีเลชันนั้น

ตารางที่ 4.14 มอดูลเกี่ยวกับการจัดการค่าของแอตทริบิว

ชื่อมอดูล	หน้าที่การทำงาน
DStoreCharStr() char *ptr,*data int len	จัดเก็บข้อมูลชนิดอักขระ data ไว้ในสายอักขระ ptr ด้วยจำนวนไบต์เท่ากับ len
DstoreIntStr() char *ptr,*data int size	จัดเก็บข้อมูลชนิดตัวเลขจำนวนเต็มไว้ในสายอักขระ ptrโดยจะแปลงค่าจากสายอักขระ data ให้เป็นค่าเลขจำนวนเต็มแล้วจัดเก็บด้วยขนาดความยาว 1 2 หรือ 4 ไบต์ตามค่า size
DStoreRealStr() char *ptr,*data	จัดเก็บข้อมูลชนิดตัวเลขทศนิยมไว้ในสายอักขระ ptr โดยจะแปลงค่าจากสายอักขระ data ให้เป็นค่าเลขทศนิยม แล้วจัดเก็บด้วยขนาดความยาว 8 ไบต์
DStoreDateStr() char *ptr,*data	จัดเก็บข้อมูลชนิดวันที่ไว้ในสายอักขระ ptr โดยจะแปลงค่าจากสายอักขระ data ที่เป็นวันที่ในรูปแบบ วัน/เดือน/ปี ให้เป็นค่าเลขจำนวนเต็มขนาดความยาว 4 ไบต์ โดยสมการ ค่าวันที่เลขจำนวนเต็ม = ปี*10000 + เดือน * 100 + วัน

ตารางที่ 4.14 มอดูลเกี่ยวกับการจัดการค่าของแอตทริบิว (ต่อ)

ชื่อมอดูล	หน้าที่การทำงาน
DStoreInt() char *ptr long data int size	จัดเก็บข้อมูลชนิดตัวเลขไว้ในสายอักขระ ptr ด้วยขนาดความยาว 1 2 หรือ 4 ไบต์ขึ้นอยู่กับค่า size
DStoreReal() char *ptr double data	จัดเก็บข้อมูลชนิดตัวเลขทศนิยมไว้ในสายอักขระ ptr ด้วยขนาดความยาว 8 ไบต์
DStoreDate() char *ptr long ldate	จัดเก็บข้อมูลชนิดวันที่ที่อยู่ในรูปแบบเลขจำนวนเต็มไว้ในสายอักขระ ptr ด้วยขนาดความยาว 4 ไบต์
DgetCharStr() char *ptr int len	อ่านข้อมูลในรูปของสายอักขระขนาดความยาวเท่ากับ len ไบต์จากสายอักขระ ptr
DGetInt() char *ptr int size	อ่านค่าข้อมูลในรูปตัวเลขจำนวนเต็มขนาดความยาว 4 ไบต์จากสายอักขระptr จำนวนไบต์เท่ากับ size
DGetReal() char *ptr	อ่านค่าข้อมูลในรูปตัวเลขทศนิยมขนาดความยาว 8 ไบต์ จากสายอักขระ ptr จำนวน 8 ไบต์
DGetDate() char *ptr	อ่านค่าข้อมูลวันที่ในรูปค่าตัวเลขจำนวนเต็ม ขนาดความยาว 4 ไบต์ จากสายอักขระ ptr
DGetIntStr() char *ptr int size	อ่านค่าข้อมูลชนิดตัวเลขจำนวนเต็มจากสายอักขระ ptr จำนวน 1 2 หรือ 4 ไบต์ ขึ้นอยู่กับค่า size และแปลงเป็นสายอักขระของเลขจำนวนเต็มที่ผู้ใช้สามารถอ่านเข้าใจได้
DGetRealStr() char *ptr int prec,scale	อ่านค่าข้อมูลชนิดตัวเลขทศนิยมจากสายอักขระ ptr และแปลงเป็นสายอักขระของตัวเลขทศนิยมจำนวนเต็มที่ผู้ใช้สามารถอ่านเข้าใจได้ โดยมีจำนวนเลขหน้าจุดทศนิยมเท่ากับ prec และจำนวนตำแหน่งหลังจุดทศนิยมเท่ากับ scale
DGetDateStr() char *ptr	อ่านค่าข้อมูลชนิดวันที่ที่จัดเก็บเป็นตัวเลขจำนวนเต็มจากสายอักขระ ptr จำนวน 4 ไบต์ ไปเป็นสายอักขระของวันที่ในรูปแบบ วัน/เดือน/ปี ด้วยสมการ ค่าวันที่เลขจำนวนเต็ม = ปี*10000 + เดือน * 100 + วัน

ตารางที่ 4.15 มอดูลเกี่ยวกับการจัดการข้อมูลตัวเลข

ชื่อมอดูล	**หน้าที่การทำงาน**
IsInteger() char *str int length	ตรวจสอบสายอักขระว่าเป็นสายอักขระของเลขจำนวนเต็มหรือไม่
IsReal() char *str int length	ตรวจสอบสายอักขระว่าเป็นสายอักขระของเลขทศนิยมหรือไม่
StrToLong() char *str	แปลงสายอักขระของเลขจำนวนเต็ม ให้เป็นค่าเลขจำนวนเต็มขนาด 4 ไบต์
StrToReal() char*str	แปลงสายอักขระของเลขจำนวนทศนิยมให้เป็นค่าเลขทศนิยมขนาด 8 ไบต์

ตารางที่ 4.16 มอดูลเกี่ยวกับการจัดการข้อมูลวันที่

ชื่อมอดูล	**หน้าที่การทำงาน**
GetSysLongDate()	แปลงวันที่ปัจจุบันของระบบเป็นค่าเลขจำนวนเต็มขนาด 4 ไบต์ โดยใช้สมการ วันที่ = ปี *10000+ เดือน*100+ วัน
CheckDateStr() char *str	ตรวจสอบสายอักขระว่าเป็นสายอักขระของวันที่หรือไม่ โดยวันที่ต้องในรูปแบบ วัน/เดือน/ปี (dd/mm/yyyy)
StrToLongDate() char *strdate	แปลงค่าสายอักขระของวันที่ที่อยู่ในรูปแบบ วัน/เดือน/ปี ให้เป็นเลขจำนวนเต็มขนาด 4 ไบต์ จากการใช้สมการ วันที่ = ปี *10000+ เดือน*100+ วัน
LongDateToStr() Long longdate	แปลงค่าจากเลขจำนวนเต็มขนาด 4 ไบต์กลับไปเป็นสายอักขระของวันที่ที่อยู่ในรูปแบบ วัน/เดือน/ปี จากการใช้สมการ วันที่=ปี *10000+ เดือน*100+ วัน

ตารางที่ 4.17 มอดูลเกี่ยวกับการจัดการสายอักขระ

ชื่อมอดูล	หน้าที่การทำงาน
StrBlank() char *dst int n	ใส่ค่าว่างให้กับข้อมูล
StrUpr() char *str	แปลงสายอักขระเป็นอักษรตัวพิมพ์ใหญ่
StrLwr() char *str	แปลงสายอักขระเป็นอักษรตัวพิมพ์เล็ก
StrDup () char *str	จองเนื้อที่หน่วยความจำและสำเนาสายอักขระ
StrDupN() char *str int n	จองเนื้อที่หน่วยความจำและสำเนาสายอักขระขนาดความยาว n ไบต์
StrEql() char *str1 char *str2	เปรียบเทียบสายอักขระทั้งสองว่าเท่ากันหรือไม่
StrEqlI(s) char *str1 char *str2	เปรียบเทียบสายอักขระทั้งสองว่าเท่ากันหรือไม่ โดยไม่สนใจอักษรตัวพิมพ์ใหญ่หรือเล็ก
StrCmp () char *str1 char *str2	เปรียบเทียบสายอักขระทั้งสอง
StrCmpI () char *str1 char *str2	เปรียบเทียบสายอักขระทั้งสอง โดยไม่สนใจอักษรตัวพิมพ์ใหญ่หรือเล็ก
FindSubStr() char *str1 char *str2	ค้นหาสายอักขระที่สองในสายอักขระที่หนึ่ง
FindSubStrI() char *str1 char *str2	ค้นหาสายอักขระที่สองในสายอักขระที่หนึ่ง โดยไม่สนใจอักษรตัวพิมพ์ใหญ่หรือเล็ก
CharToStr() char ch	จัดเก็บอักขระไว้ในสายอักขระ

ตารางที่ 4.18 มอดูลเกี่ยวกับการเปรียบเทียบค่าแอตทริบิว

ชื่อมอดูล	หน้าที่การทำงาน
CompareStr () char *str1,*str2 int len	เปรียบเทียบข้อมูลจากสายอักขระทั้งสอง
CompareInt() long a,b	เปรียบเทียบข้อมูลจากค่าเลขจำนวนเต็มของเลขสองจำนวน
CompareReal() double a,b	เปรียบเทียบข้อมูลจากค่าเลขทศนิยมของเลขสองจำนวน
CompareRid() RID a,b	เปรียบเทียบข้อมูลจากค่าหมายเลขเรคอร์ดทั้งสอง
AttValueCompare() char *attvalue, *cmpvalue ATTINFO *att	เปรียบเทียบค่าแอตทริบิวทั้งสอง ตามชนิดและขนาดของแอตทริบิวที่กำหนดใน att
CondCompare() int funcid,attid char *tuple LIST *lattinfo ITEMNODE *item	ตรวจสอบค่าแอตทริบิวโดยใช้ฟังก์ชันในการเปรียบเทียบ
CondBetween() char *tuple LIST *lattinfo int attid ITEMNODE *item	ตรวจสอบค่าแอตทริบิวโดยใช้ฟังก์ชัน Between
CondIn() char *tuple LIST *lattinfo int attid ITEMNODE *item	ตรวจสอบค่าแอตทริบิวโดยใช้ฟังก์ชัน IN
CondCheck() EXPRNODE *expr char *tuple LIST *lattinfo	ตรวจสอบค่าแอตทริบิวตามชนิดฟังก์ชันที่กำหนดใน expr
CondNodeSet() EXPRNODE *cond int attid, funcid, len char attdomain, *value	กำหนดค่าเริ่มต้นสำหรับโครงสร้างข้อมูล EXPRNODE

ตารางที่ 4.19 มอดูลเกี่ยวกับค่าแอตทริบิวที่เป็นคีย์

ชื่อมอดูล	หน้าที่การทำงาน
AddValueInExpr() EXPRNODE *expr char *value, itemtype int len	เพิ่มข้อมูลไว้ในโครงสร้างข้อมูล EXPRNODE ในส่วนของ value อีกหนึ่งค่า
KeyBuild () char *indexname int nokey KTYPE *ktype DBFILE *dbfile	สร้างรีเลชันใหม่เพื่อจัดเก็บค่าคีย์ และหมายเลขเรคอร์ดของข้อมูลที่สัมพันธ์กับค่าคีย์นั้น โดยใช้โครงสร้างข้อมูล $B_m$tree ซึ่งค่าคีย์ที่จัดเก็บจะเป็นคีย์เดี่ยวหรือคีย์ร่วมก็ได้ โดยจะต้องระบุจำนวนคีย์ และประเภทและความยาวของแต่ละคีย์ด้วย
KeySearch() char *indexname,*key RID *recrid DBFILE *dbfile	ค้นหาค่าคีย์ และตำแหน่งหมายเลขเรคอร์ดของข้อมูลที่สัมพันธ์กับค่าคีย์นั้น
KeyInsert() char *indexname,*key RID rid DBFILE *dbfile	เพิ่มค่าคีย์ใหม่ของรีเลชัน โดยจัดเก็บตำแหน่งหมายเลขเรคอร์ดของข้อมูลที่สัมพันธ์กับค่าคีย์นั้นไว้ด้วย
KeyUpdate() char*indexname,*key,*newkey RID rid DBFILE *dbfile	เปลี่ยนแปลงค่าคีย์เก่ารีเลชันให้เป็นค่าคีย์ใหม่
KeyDelete() char *indexname,*key RID *rid DBFILE *dbfile	ลบค่าคีย์เก่าของของรีเลชัน
KeyDeleteAll() char *indexname DBFILE *dbfile	ลบค่าคีย์ทั้งหมดของรีเลชัน
KeySet() char *keyptr,value,attdomain int len,lkey	กำหนดค่าคีย์หรือส่วนหนึ่งของคีย์

ตารางที่ 4.20 มอดูลเกี่ยวกับการทำงานพื้นฐานกับทุเปิลของรีเลชัน

ชื่อมอดูล	หน้าที่การทำงาน
TupleRelCreate () DBFILE *dbfile char *relname	สร้างรีเลชันใหม่ในระบบ
TupleRelDrop() DBFILE *dbfile char *relname	ลบรีเลชันเก่าในระบบและข้อมูลที่จัดเก็บในรีเลชันทั้งหมดด้วย
TupleInsert() DBFILE *dbfile char *name,*tuple int len RID *rid	เพิ่มทุเปิลใหม่ของรีเลชัน
TupleUpdate() DBFILE *dbfile char *name, *tuple int len RID rid	ปรับปรุงทุเปิลของรีเลชัน ณ หมายเลขเรคอร์ดที่ต้องการ
TupleDelete() DBFILE *dbfile char *relname RID rid	ลบทุเปิลของรีเลชัน ณ หมายเลขเรคอร์ดที่ต้องการ
TupleDeleteAll() DBFILE *dbfile char *relname	ลบทุกทุเปิลของรีเลชัน
TupleGetRecCt() DBFILE *dbfile char *name	นับจำนวนทุเปิลทั้งหมดของรีเลชัน
TupleGet() DBFILE *dbfile char *relname,*tuple int *len	อ่านทุเปิลของรีเลชัน ณ หมายเลขเรคอร์ดที่ต้องการ

ตารางที่ 4.21 มอดูลเกี่ยวกับการเข้าถึงทุเปิลของรีเลชันแบบเรียงลำดับ

ชื่อมอดูล	หน้าที่การทำงาน
TupleSelect() DBFILE *dbfile char *relname LIST *lattinfo RID rid	อ่านทุเปิลของรีเลชันทั้งหมดเพื่อส่งไปจัดเก็บในรีเลชัน show.data ทีละทุเปิล จนครบทุกทุเปิลในรีเลชัน
TupleCondSelectFirst() DBFILE *dbfile char *relname,returntuple LIST *lattinfo EXPRNODE *condition RID *outrid	อ่านทุเปิลของรีเลชันทีละทุเปิล เมื่อพบทุเปิลแรกในรีเลชันตามเงื่อนไขที่กำหนดใน cond จะส่งข้อมูลทุเปิลแรกพร้อมกับหมายเลขเรคอร์ดกลับไปยังโปรแกรมที่เรียกใช้มอดูลนี้
TupleCondSelectReturn() DBFILE *dbfile char *relname LIST *lattinfo EXPRNODE*cond,*returncond	อ่านทุเปิลของรีเลชันทีละทุเปิล เพื่อนำค่าของแอตทริบิวที่ระบุใน returncond ไปเก็บในลิงค์ลิสต์ของ returncond เพื่อส่งกลับไปยังโปรแกรมที่เรียกใช้มอดูลนี้ โดยจะเก็บค่าแอตทริบิวของทุกทุเปิลที่ตรงตามเงื่อนไขที่กำหนดใน cond
TupleCondSelectReturnFirst() DBFILE *dbfile char *relname LIST *lattinfo EXPRNODE*cond,*returncond	อ่านทุเปิลของรีเลชันทีละทุเปิล เมื่อพบทุเปิลแรกในรีเลชันตามเงื่อนไขที่กำหนดใน cond จะนำค่าของแอตทริบิวที่ระบุใน returncond ไปเก็บไนลิงค์ลิสต์ของ returncond และส่งกลับไปยังโปรแกรมที่เรียกใช้มอดูลนี้
TupleSelectReturnRID() DBFILE *dbfile char *relname LIST *lattinfo EXPRNODE*cond,*returncond	อ่านทุเปิลของรีเลชันทีละทุเปิล เพื่อนำหมายเลขเรคอร์ดไปเก็บในลิงค์ลิสต์ของ returncond เพื่อส่งกลับไปยังโปรแกรมที่เรียกใช้มอดูลนี้ โดยจะเก็บเฉพาะหมายเลขเรคอร์ดที่มีค่าแอตทริบิวของทุเปิลที่ตรงตามเงื่อนไขที่กำหนดใน cond
TupleReturnRIDFirst() DBFILE *dbfile char *relname LIST *lattinfo EXPRNODE*cond,*returncond	อ่านทุเปิลของรีเลชันทีละทุเปิล เมื่อพบทุเปิลแรกในรีเลชันตามเงื่อนไขที่กำหนดใน cond จะนำหมายเลขเรคอร์ดไปเก็บในลิงค์ลิสต์ของ returncond เพื่อส่งกลับไปยังโปรแกรมที่เรียกใช้มอดูลนี้

ตารางที่ 4.22 มอดูลเกี่ยวกับการปรับปรุงแก้ไขหรือลบทุเปิลของรีเลชันแบบเรียงลำดับ

ชื่อมอดูล	หน้าที่การทำงาน
TupleDelCond() DBFILE *dbfile char *relname LIST *lattinfo EXPRNODE *cond,*returncond	อ่านทุเปิลของรีเลชันทั้งหมดเพื่อลบเฉพาะทุเปิลที่ตรงตามเงื่อนไขที่กำหนดใน cond โดยอาจมีการส่งค่าแอตทริบิวของทุเปิลที่ลบตามที่ระบุไว้ใน returncond
TupleDelCondCons() RELNODE *relnode KEYCONS *keycons EXPRNODE *cond	อ่านทุเปิลของรีเลชันทั้งหมดเพื่อลบเฉพาะทุเปิลที่ตรงตามเงื่อนไขที่กำหนดใน cond โดยจะต้องลบทุเปิลของรีเลชันอื่นที่มีค่าคีย์นอกสัมพันธ์กับค่าคีย์หลักของทุเปิลที่จะถูกลบด้วย
DeleteChild() DBFILE *dbfile char *db,*tuple ATTINFO *attinfo KEYCONS *keycons	ลบทุเปิลของรีเลชันอื่นที่ค่าคีย์นอกอ้างอิงค่าคีย์หลักของทุเปิลที่จะถูกลบ
SetKeyFromKeyid() char *key, * tuple int nokey nsigned char *keyid ATTINFO *att	กำหนดค่าคีย์ของทุเปิลที่จะถูกลบ
TupleUpdCondCons() RELNODE *relnode unsigned char*updid int no, *found, child ITEMNODE *upditem KEYCONS *keycons EXPRNODE *cond	อ่านทุเปิลของรีเลชันทั้งหมด เพื่อปรับปรุงค่าแอตทริบิวของทุเปิลของรีเลชันตามที่กำหนดใน updid โดยจะปรับปรุงเฉพาะทุเปิลที่ตรงตามเงื่อนไขที่ระบุใน cond (ถ้าไม่ได้ระบุ cond จะปรับปรุงทุกทุเปิล) พร้อมทั้งปรับปรุงค่าคีย์นอกของรีเลชันอื่นที่อ้างอิงค่าแอตทริบิวจากค่าคีย์หลักที่ถูกปรับปรุงด้วย นอกจากนี้ยังนับจำนวนทุเปิลที่ตรงตามเงื่อนไข และจำนวนทุเปิลที่ถูกปรับปรุงด้วย
CondCompareListItem() int funcid,noupdid char *tuple LIST *lattinfo unsigned char *updid ITEMNODE *litem	ตรวจสอบว่าทุเปิลที่จะถูกปรับปรุงมีค่าของทุกแอตทริบิวตามรหัสแอตทริบิวใน updid ตรงตามค่าแอตทริบิวที่กำหนดอยู่ใน item แล้วหรือไม่ ถ้าใช่จะไม่ต้องปรับปรุงทุเปิลนี้

ตารางที่ 4.19 มอดูลเกี่ยวกับการปรับปรุงแก้ไขหรือลบทุเปิลของรีเลชันแบบเรียงลำดับ (ต่อ)

ชื่อมอดูล	หน้าที่การทำงาน
UpdChild() DBFILE *dbfile char *tuple RID *rid ATTINFO *att unsigned char*updid int noupdid ITEMNODE *upditem KEYCONS *keycons	ปรับปรุงทุเปิลของรีเลชันอื่นที่มีการกำหนดเงื่อนไขบังคับการอ้างอิงค่าคีย์นอกจากค่าคีย์หลักของทุเปิลที่ถูกปรับปรุง
UpdConsKeyDup() char *tuple RID *rid ATTINFO *attinfo unsigned char *updid int noupdid ITEMNODE *upditem KEYCONS *keycons	ตรวจสอบว่าคีย์ของทุเปิลที่ถูกปรับปรุงด้วยหลังปรับปรุงไปซ้ำกับค่าคีย์เดิมในรีเลชันหรือไม่ โดยจะตรวจสอบทั้งคีย์หลักและคีย์รอง และวนทำซ้ำสำหรับรีเลชันที่สัมพันธ์กันทั้งหมดด้วย
UpdConsKey() char *tuple RID *rid ATTINFO *attinfo unsigned char *updid int noupdid ITEMNODE *upditem KEYCONS *keycons	ปรับปรุงค่าคีย์ทั้งคีย์หลักและคีย์รองของทุเปิลที่ถูกปรับปรุง
UpdConsKeyParentExist() char *tuple ATTINFO *attinfo unsigned char *updid int noupdid ITEMNODE *upditem PARENTCONS *keycons	ตรวจสอบว่า ค่าคีย์ที่ต้องการเปลี่ยนแปลงแก้ไขเป็นค่าใหม่มีในคีย์หลักของรีเลชันที่คีย์นอกของรีเลชันที่ถูกปรับปรุงอ้างถึงหรือไม่
UpdTuple() char* tuple ATTINFO *attinfo unsigned char *updid int noupdid ITEMNODE *item	ปรับปรุงค่าแอตทริบิวของทุเปิลตามรหัสแอตทริบิวที่กำหนดใน updid และ ค่าแอตทริบิวที่กำหนดใน item

ตารางที่ 4.23 มอดูลเกี่ยวกับการวิเคราะห์คำสั่งเพื่อแยกเป็นโทเคน

ชื่อมอดูล	หน้าที่การทำงาน
LexInitialize()	กำหนดตารางสถานะของอักขระตามรหัสแอสกีทั้ง 255 ตัว เพื่อใช้ในการวิเคราะห์สายอักขระของคำสั่งและแยกโทเคน
LexStart() unsigned char *str unsigned int length	กำหนดค่าเริ่มต้นสำหรับการวิเคราะห์สายอักขระของคำสั่งในแต่ละครั้ง
FindKeyword() LEX *lex unsigned int length	ค้นหาคำหลักจากตารางเก็บคำหลักทั้งหมด
GetToken() LEX *lex unsigned int length	ตัดข้อความและความยาวของโทเคนที่ได้
GetTextStr() LEX *lex	ตัดข้อความภายในเครื่องหมายอัญประกาศ
yylex()	วิเคราะห์สถานะของอักขระต่างๆ ในสายอักขระของคำสั่ง เพื่อตัดเป็นให้ได้โทเคนที่ถูกต้องตามความต้องการ
GetCh()	อ่านอักขระตำแหน่งปัจจุบันจากสายอักขระของคำสั่ง
GetChBack()	อ่านอักขระตำแหน่งก่อนหน้าอักขระในตำแหน่งปัจจุบันหนึ่งตำแหน่ง
GetChFirst()	อ่านอักขระตัวแรกในสายอักขระของคำสั่ง
UngetCh()	ยกเลิกการอ่านอักขระปัจจุบันในสายอักขระของคำสั่ง โดยย้อนกลับไปหนึ่งอักขระ
SkipCh()	อ่านอักขระถัดไปจากสายอักขระของคำสั่งโดยไม่สนใจอักขระตำแหน่งปัจจุบัน
Length()	คำนวณความยาวของสายอักขระของโทเคนที่ได้

ตารางที่ 4.24 มอดูลเกี่ยวกับจัดเก็บข้อมูลในระหว่างรับคำสั่ง

ชื่อมอดูล	หน้าที่การทำงาน
CheckIdentName() char *name int col	ตรวจสอบว่าชื่อตัวชี้เฉพาะได้แก่ ชื่อฐานข้อมูล ชื่อรีเลชัน และชื่อแอตทริบิวที่ผู้ใช้กำหนดขึ้น ว่าถูกต้องตามข้อกำหนดการตั้งชื่อตัวชี้เฉพาะที่โปรแกรมกำหนดไว้หรือไม่
SetStrName() STRING strname	จัดเก็บชื่อชี้เฉพาะต่าง ๆ ได้แก่ ชื่อฐานข้อมูล ชื่อรีเลชัน ชื่อแอตทริบิว ไว้ในโครงสร้างข้อมูล STRNAME
SaveKeyList() LIST *listsave LIST *listkey	ย้ายโครงสร้างข้อมูลที่จัดเก็บในลิงค์ลิสต์ของ listkey มาจัดเก็บไว้ในลิงค์ลิสต์ของ listsave
SaveRefList() LIST *listfkey,LIST *refatt STRING refrel int ondelete,short col	จัดเก็บข้อมูลเกี่ยวกับเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิวจากค่าคีย์หลักในอีกรีเลชันหนึ่ง ได้แก่ ชื่อรีเลชันที่อ้างถึง รายชื่อแอตทริบิวที่กำหนดคีย์นอก ข้อกำหนดของการลบทุเปิลอย่างต่อเนื่อง
SetItemUpd() LIST *lattname, *litem	จัดเก็บข้อมูลที่เกี่ยวข้องกับการปรับปรุงค่าทุเปิล ได้แก่ รายชื่อแอตทริบิวที่ต้องการปรับปรุง และ รายการค่าแอตทริบิวที่ต้องการกำหนดให้กับแอตทริบิว

ตารางที่ 4.25 มอดูลเกี่ยวกับการจองเนื้อที่ในหน่วยความจำสำหรับลิงค์ลิสต์

ชื่อมอดูล	หน้าที่การทำงาน
AddAttToList() char *attname,attdomain int attlen, attdecimal,attflag ITEMNODE *attdefaval	จองเนื้อที่ในหน่วยความจำ และจัดเก็บข้อมูลต่างๆในโครงสร้างข้อมูล ATTNODE เพื่อนำไปต่อท้ายลิงค์ลิสต์
AddColToList() STRING attname int nodup	จองเนื้อที่ในหน่วยความจำ และจัดเก็บข้อมูลต่างๆในโครงสร้างข้อมูล COLNODE เพื่อนำไปต่อท้ายลิงค์ลิสต์ โดยอาจกำหนดให้มีการตรวจสอบว่ามีค่าซ้ำกับค่าเดิมที่จัดเก็บอยู่ในลิสต์แล้วหรือไม่

ตารางที่ 4.25 มอดูลเกี่ยวกับการจองเนื้อที่ในหน่วยความจำสำหรับลิงค์ลิสต์ (ต่อ)

ชื่อมอดูล	หน้าที่การทำงาน
AddItemToList() STRING value char itemtype	จองเนื้อที่ในหน่วยความจำ และจัดเก็บข้อมูลต่างๆในโครง สร้างข้อมูล ITEMNODE เพื่อนำไปต่อท้ายลิงค์ลิสต์
AddExprToList() STRING attname int FuncId LIST *Item	จองเนื้อที่ในหน่วยความจำ และจัดเก็บข้อมูลต่างๆในโครง สร้างข้อมูล EXPRNODE เพื่อนำไปต่อท้ายลิงค์ลิสต์
AddRelConsToList() LIST *list char *rel,*cons	จองเนื้อที่ในหน่วยความจำ และจัดเก็บข้อมูลต่างๆในโครง สร้างข้อมูล RELCONS เพื่อนำไปต่อท้ายลิงค์ลิสต์
AddRefConsToList() LIST *list,*latt char *cons,*rel unsigned char *keyid,*updid int nokey KEYCONS *keycons	จองเนื้อที่ในหน่วยความจำ และจัดเก็บข้อมูลต่างๆในโครง สร้างข้อมูล REFCONS เพื่อนำไปต่อท้ายลิงค์ลิสต์
AddKeyConsToList() LIST *list char *cons unsigned char *keyid,*updpos int nokey,noupd REFCONS *refcons PARENTCONS *parent	จองเนื้อที่ในหน่วยความจำ และจัดเก็บข้อมูลต่างๆในโครง สร้างข้อมูล KEYCONS เพื่อนำไปต่อท้ายลิงค์ลิสต์
AddParentConsToList() LIST *list char *cons unsigned char *keyid,*updpos int nokey,noupd ATTINFO *attinfo unsigned char *refkeyid	จองเนื้อที่ในหน่วยความจำ และจัดเก็บข้อมูลต่างๆในโครง สร้างข้อมูล PARENTCONS เพื่อนำไปต่อท้ายลิงค์ลิสต์

ตารางที่ 4.26 มอดูลเกี่ยวกับการนำโหนดใหม่ไปต่อท้ายลิงค์ลิสต์

ชื่อมอดูล	หน้าที่การทำงาน
AppendAttList() LIST *list ATTNODE *node	นำโหนดใหม่ของโครงสร้างข้อมูล ATTNODE ไปต่อท้ายในลิงค์ลิสต์ในโครงสร้างข้อมูล LIST
AppendColList() LIST *list COLNODE *node int nodup	นำโหนดใหม่ของโครงสร้างข้อมูล COLNODE ไปต่อท้ายในลิงค์ลิสต์ในโครงสร้างข้อมูล LIST โดยอาจมีการตรวจสอบว่ามีค่าซ้ำกับค่าเดิมที่จัดเก็บอยู่ในลิสต์แล้วหรือไม่
AppendItemList() LIST *list ITEMNODE *node	นำโหนดใหม่ของโครงสร้างข้อมูล ITEMNODE ไปต่อท้ายในลิงค์ลิสต์ ในโครงสร้างข้อมูล LIST
AppendRefList() LIST *list REFNODE *node	นำโหนดใหม่ของโครงสร้างข้อมูล REFNODE ไปต่อท้ายในลิงค์ลิสต์ในโครงสร้างข้อมูล LIST
AppendExprList() LIST *list EXPRNODE *node	นำโหนดใหม่ของโครงสร้างข้อมูล EXPRNODE ไปต่อท้ายในลิงค์ลิสต์ในโครงสร้างข้อมูล LIST
AppendRelConsList() LIST *list RELCONS *node	นำโหนดใหม่ของโครงสร้างข้อมูล RELCONS ไปต่อท้ายในลิงค์ลิสต์ในโครงสร้างข้อมูล LIST
AppendRefConsList() LIST *list REFCONS *node	นำโหนดใหม่ของโครงสร้างข้อมูล REFCONS ไปต่อท้ายในลิงค์ลิสต์ในโครงสร้างข้อมูล LIST
AppendKeyConsList() LIST *list KEYCONS *node	นำโหนดใหม่ของโครงสร้างข้อมูล KEYCONS ไปต่อท้ายในลิงค์ลิสต์ในโครงสร้างข้อมูล LIST
AppendParentConsList() LIST *list PARENTCONS *node	นำโหนดใหม่ของโครงสร้างข้อมูล PARENTCONS ไปต่อท้ายในลิงค์ลิสต์ในโครงสร้างข้อมูล LIST
ListEmpty() LIST *list	กำหนดค่าเริ่มต้นสำหรับโครงสร้างข้อมูล LIST เพื่อเตรียมพร้อมสำหรับการเพิ่มโหนดในลิงค์สิสต์ของโครงสร้างข้อมูล LIST และการนับจำนวนโหนดที่เพิ่มเข้าไปในลิงค์ลิสต์ด้วย

ตารางที่ 4.27 มอดูลเกี่ยวกับการคืนเนื้อที่ในหน่วยความจำสำหรับลิงค์ลิสต์

ชื่อมอดูล	หน้าที่การทำงาน
FreeAttNodeList() LIST *list	คืนเนื้อที่ในหน่วยความจำที่จองไว้สำหรับโครงสร้างข้อมูล ATTNODE ของทุกโหนดในลิงค์ลิสต์ในโครงสร้างข้อมูล LIST
FreeColNodeList()	คืนเนื้อที่ในหน่วยความจำที่จองไว้สำหรับโครงสร้างข้อมูล COLNODE ของทุกโหนดในลิงค์ลิสต์ในโครงสร้างข้อมูล LIST
FreeColNodes() COLNODE *list	คืนเนื้อที่ในหน่วยความจำที่จองไว้สำหรับโครงสร้างข้อมูล COLNODE ของทุกโหนดในลิงค์ลิสต์ของ COLNODE
FreeItemNodeList()	คืนเนื้อที่ในหน่วยความจำที่จองไว้สำหรับโครงสร้างข้อมูล ITEMNODE ของทุกโหนดในลิงค์ลิสต์ในโครงสร้างข้อมูล LIST
FreeItemNodes() ITEMNODE *list	คืนเนื้อที่ในหน่วยความจำที่จองไว้สำหรับโครงสร้างข้อมูล ITEMNODE ของทุกโหนดในลิงค์ลิสต์ของ ITEMNODE
FreeRefNodeList()	คืนเนื้อที่ในหน่วยความจำที่จองไว้สำหรับโครงสร้างข้อมูล REFNODE ของทุกโหนดในลิงค์ลิสต์ในโครงสร้างข้อมูล LIST
FreeExprNodeList()	คืนเนื้อที่ในหน่วยความจำที่จองไว้สำหรับโครงสร้างข้อมูล EXPRNODE ของทุกโหนดในลิงค์ลิสต์ในโครงสร้างข้อมูล LIST
FreeExprNodes() EXPRNODE *list	คืนเนื้อที่ในหน่วยความจำที่จองไว้สำหรับโครงสร้างข้อมูล EXPRNODE ของทุกโหนดในลิงค์ลิสต์ของ EXPRNODE
FreeRelConsNodes() RELCONS *list	คืนเนื้อที่ในหน่วยความจำที่จองไว้สำหรับโครงสร้างข้อมูล RELCONS ของทุกโหนดในลิงค์ลิสต์ของ RELCONS
FreeRefConsNodes() REFCONS *list	คืนเนื้อที่ในหน่วยความจำที่จองไว้สำหรับโครงสร้างข้อมูล REFCONS ของทุกโหนดในลิงค์ลิสต์ของ REFCONS
FreeKeyConsNodes() KEYCONS *list	คืนเนื้อที่ในหน่วยความจำที่จองไว้สำหรับโครงสร้างข้อมูล REFCONS ของทุกโหนดในลิงค์ลิสต์ของ REFCONS

ตารางที่ 4.28 มอดูลเกี่ยวกับแสดงข้อความบนจอภาพ

ชื่อมอดูล	หน้าที่การทำงาน
ShowMsg() int msgcode	แสดงข้อความให้กับผู้ใช้ทางจอภาพ
ShowErrMsg() int errcode	แสดงข้อความแจ้งข้อผิดพลาดให้กับผู้ใช้ทางจอภาพ
PrintMessage() char *msg	พิมพ์ข้อความให้กับผู้ใช้ทางจอภาพ
PrintErrPosition() short col	พิมพ์ประโยคคำสั่งที่ผู้ใช้พิมพ์เข้ามา พร้อมทั้งแจ้งตำแหน่งที่ผิดพลาดของคำสั่งให้ด้วย

ตารางที่ 4.28 มอดูลเกี่ยวกับการแสดงรายการทุเปิลของรีเลชันในรูปของตาราง

ชื่อมอดูล	หน้าที่การทำงาน
WriteColsName() LIST *latt	จัดเก็บรายชื่อแอตทริบิว พร้อมทั้งขนาดและชนิดข้อมูลของค่าแอตทริบิวในรีเลชันที่ต้องการแสดงผลบนจอภาพไว้ในรีเลชัน show.col ที่เตรียมไว้
RealWriteTuple () char *data int len	จัดเก็บทุเปิลของรีเลชันที่ต้องการแสดงผลบนจอภาพในรูปของตารางไว้ในรีเลชัน show.data ที่เตรียมไว้
WriteTuple() char *tuple LIST *latt	จัดเตรียมทุเปิลของรีเลชันที่ต้องการแสดงผลบนจอภาพในรูปของตาราง โดยจัดเก็บในโครงสร้างข้อมูล PACKET
PacketNew() int l	การกำหนดโครงสร้างข้อมูล PACKET ใหม่เพื่อจัดเก็บเรคอร์ดแบบความยาวไม่แน่นอน
PacketStoreData() PACKET *packet char *from int datalen	จัดเก็บข้อมูลไว้ในโครงสร้างข้อมูล PACKET

ตารางที่ 4.28 มอดูลเกี่ยวกับการแสดงรายการทุเปิลของรีเลชันในรูปของตาราง (ต่อ)

ชื่อมอดูล	หน้าที่การทำงาน
GetRowBuffer() SEQACCESS *seqacc DATABUFF *data	อ่านทุเปิลจากรีเลชัน show.col เพื่อนำไปแสดงผลบนจอภาพในรูปของตาราง
PrintDataRow() char *NumberFlag char *Sep DATABUFF *data	แสดงทุเปิลบนจอภาพในรูปของตาราง
PrintHeader() DATABUFF *databuff char *NumberFlag, *SepStr	แสดงส่วนหัวของตารางพร้อมทั้งความนวณความกว้างสูงสุดของแต่ละสดมภ์ด้วย
GetHeader() DATABUFF *databuff	อ่านรายชื่อแอตทริบิวจากรีเลชัน show.col เพื่อจะนำไปแสดงเป็นส่วนหัวของตาราง
Result()	นำทุเปิลจากรีเลชัน show.col และ show.dat มาแสดงผลบนจอภาพในรูปของตาราง
ShowTupleRel() DBFILE *dbfile char *relname LIST *lattinfo	แสดงทุเปิลทั้งหมดของรีเลชันบนจอภาพในรูปของตาราง อาจแสดงทุกแอตริบิวหรือบางแอตทริบิวตามที่ระบุใน lattinfo
ShowTupleRelCond() DBFILE *dbfile char *relname LIST lattinfo,lselect EXPRNODE *cond	แสดงทุเปิลของรีเลชันบนจอภาพในรูปของตาราง อาจแสดงทุกแอตริบิวหรือบางแอตทริบิวตามที่ระบุใน lselect และจะแสดงเฉพาะทุเปิลที่ต้องการตามเงื่อนไขใน cond

ตารางที่ 4.30 มอดูลเกี่ยวกับการเริ่มต้นเข้าสู่การทำงานของโปรแกรมหรือออกจากโปรแกรม

ชื่อมอดูล	หน้าที่การทำงาน
DataBufferInit() BUFMGR *bufmgr int noframes	จัดเตรียมเนื้อที่ในหน่วยความจำสำหรับเป็นบัฟเฟอร์ เพื่อจัดเก็บข้อมูลจากดิสก์ระหว่างประมวลผล พร้อมทั้งกำหนดค่าเริ่มต้นสำหรับโปรแกรมส่วนการจัดการบัฟเฟอร์
DataBufferFree() BUFMGR *bufmgr	ยกเลิกการจองเนื้อที่บัฟเฟอร์ในหน่วยความจำ
OpenDbFile() DBFILE *dbfile BUFMGR *bufmgr char *filename int nopages	เปิดแฟ้มข้อมูลสำหรับจัดเก็บข้อมูลบนดิสก์ ถ้าไม่มีแฟ้มข้อมูลเดิมอยู่ในระบบจะสร้างแฟ้มข้อมูลใหม่ให้
CloseDbFile() DBFILE *dbfile	ปิดแฟ้มข้อมูล ถ้าข้อมูลบางส่วนที่จัดเก็บในบัฟเฟอร์ขณะประมวลผลมีการเปลี่ยน จะบันทึกกลับลงดิสก์ให้ด้วย
IndexBuildRelandAtt()	สร้างการจัดเก็บข้อมูลดัชนีสำหรับการค้นหาชื่อรีเลชัน และชื่อแอตทริบิว
SetListAttDb()	กำหนดโครงสร้างรีเลชันที่จัดเก็บรายชื่อฐานข้อมูล ซึ่งเป็นตัวแปรที่ใช้งานร่วมกัน (LISTATTDB)
SetListAttRel()	กำหนดโครงสร้างรีเลชันที่จัดเก็บรายชื่อรีเลชันของ ซึ่งเป็นตัวแปรที่ใช้งานร่วมกัน (LISTATTREL)
SetListAttAtt()	กำหนดโครงสร้างรีเลชันที่จัดเก็บรายชื่อแอตทริบิว ซึ่งเป็นตัวแปรที่ใช้งานร่วมกัน (LISTATTATT)
SetListAttCons()	กำหนดโครงสร้างรีเลชันที่จัดเก็บรายชื่อรีเลชันที่มีการกำหนดคีย์หลัก คีย์รอง และคีย์นอก ซึ่งเป็นตัวแปรที่ใช้งานร่วมกัน (LISTATTCONS)
SetListAttColCons()	กำหนดโครงสร้างรีเลชันที่จัดเก็บรายชื่อแอตทริบิวที่มีการกำหนดคีย์หลัก คีย์รอง และคีย์นอก ซึ่งเป็นตัวแปรที่ใช้งานร่วมกัน (LISTATTCOLCONS)

ตารางที่ 4.31 มอดูลเกี่ยวกับการตั้งชื่อต่าง ๆ ในโปรแกรม

ชื่อมอดูล	หน้าที่การทำงาน
CatalogFileName() char *filename	ชื่อแฟ้มข้อมูลสำหรับจัดเก็บปทานุกรมข้อมูลในระบบ (system.cat)
RelIndexName()	ชื่อดัชนีของรายชื่อรีเลชันทั้งหมดในระบบ
AttIndexName()	ชื่อดัชนีของรายชื่อแอตทริบิวทั้งหมดในระบบ
DbRelName()	ชื่อรีเลชันที่จัดเก็บรายชื่อฐานข้อมูลทั้งหมดในระบบ (db.cat)
RelRelName()	ชื่อรีเลชันที่จัดเก็บรายชื่อรีเลชันทั้งหมดในระบบ (rel.cat)
AttRelName()	ชื่อรีเลชันที่จัดเก็บรายชื่อแอตทริบิวทั้งหมดในระบบ (att.cat)
ConsRelName()	ชื่อรีเลชันที่จัดเก็บการกำหนดเงื่อนไขบังคับทั้งหมดในระบบ (cons.cat) ประกอบด้วยรหัสเงื่อนไขบังคับ ชื่อฐานข้อมูล ชื่อรีเลชัน ชื่อเงื่อนไขบังคับ และประเภทของเงื่อนไขบังคับว่ากำหนดเป็นคีย์หลัก คีย์รอง หรือการอ้างอิงค่าแอตทริบิว
ColConsRelName()	ชื่อรีเลชันที่จัดเก็บรายชื่อแอตทริบิวที่ถูกกำหนดเงื่อนไขบังคับทั้งหมดในระบบ (colcons.cat)
RefConsRelName()	ชื่อรีเลชันที่จัดเก็บรายละเอียดของการกำหนดเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิวทั้งหมดในระบบ (refcons.cat)
IndexName() char *dbname,*consname char *indexname	ชื่อดัชนีสำหรับการกำหนดคีย์ของรีเลชัน
ConsName() char *relname,*consname char constype int lstconsid	ชื่อเงื่อนไขบังคับ
DataFileName() char *db char *filename	ชื่อแฟ้มข้อมูลสำหรับจัดเก็บข้อมูลของฐานข้อมูล (<dbname>)
TupleRelName() char *db char *relname	ชื่อรีเลชันสำหรับจัดเก็บข้อมูลของรีเลชันในฐานข้อมูล (<relname>) ซึ่งแต่ละรีเลชันในฐานข้อมูลเดียวกันจะจัดเก็บในแฟ้มข้อมูลเดียวกัน

4.3 ขั้นตอนวิธีของมอดูลการทำงานหลักของระบบ

มอดูลทั้งหมดในระบบมีเป็นจำนวนมาก ดังที่ได้กล่าวถึงหน้าที่ไปแล้วในหัวข้อ 4.2 ในหัวข้อนี้จะกล่าวถึงเฉพาะขั้นตอนวิธีของมอดูลการทำงานหลักของระบบเท่านั้น ซึ่งการทำงานของแต่ละมอดูลได้มีการใช้ตัวแปรบางอย่างร่วมกัน เพื่ออ้างถึงข้อมูลเดียวกันทั้งระบบ ตัวแปรที่ใช้งานร่วมกันในแต่ละมอดูลมีดังนี้

- **CATALOGFILE** เป็น แฟ้มปทานุกรมข้อมูล
- **LISTATTDB** เป็นโครงสร้างปทานุกรมข้อมูล Database ที่จัดเก็บรายชื่อฐานข้อมูล
- **LISTATTREL** เป็นโครงสร้างปทานุกรมข้อมูล Relation ที่จัดเก็บรายชื่อรีเลชัน
- **LISTATTATT** เป็นโครงสร้างปทานุกรมข้อมูล Attribute ที่จัดเก็บรายชื่อแอตทริบิว
- **LISTATTCONS** เป็นโครงสร้างปทานุกรมข้อมูล Constraint ที่จัดเก็บรายละเอียดของการกำหนดเงื่อนไขบังคับในรีเลชัน ประกอบด้วย การกำหนดคีย์หลัก คีย์รอง และคีย์นอกหรือเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิว
- **LISTATTCOLCONS** เป็นโครงสร้างปทานุกรมข้อมูล ColConstraint ที่จัดเก็บรายชื่อแอตทริบิวที่มีการกำหนดคีย์หลัก คีย์รอง และคีย์นอก
- **LISTATTREFCONS** เป็นโครงสร้างปทานุกรมข้อมูล RefConstraint ที่จัดเก็บรายละเอียดการกำหนดคีย์นอก

มอดูลการทำงานหลักของระบบแต่ละมอดูลมีดังนี้

1. LoadSystem เป็นทำงานในส่วนของการจัดเตรียมบัฟเฟอร์ และค่าเริ่มต้นต่างๆ เพื่อเข้าสู่การทำงานของระบบ

ตัวแปรในการเรียกใช้

-

ขั้นตอนวิธี

1) จัดเตรียมบัฟเฟอร์สำหรับจัดเก็บปทานุกรมข้อมูล และข้อมูลในฐานข้อมูลระหว่างการประมวลผล
2) กำหนดค่าเริ่มต้นต่าง ๆ สำหรับการทำงานในระบบ
3) เปิดแฟ้มปทานุกรมข้อมูลเพื่อเตรียมพร้อมสำหรับการทำงานในระบบ ถ้าแฟ้มปทานุกรมข้อมูลเดิมไม่ได้มีอยู่ในระบบ จะทำการสร้างแฟ้มปทานุกรมข้อมูลใหม่ ซึ่งกำหนดเป็นตัวแปรร่วม ชื่อ CATALOGFILE

4) กำหนดโครงสร้างรีเลชันต่างๆ ที่จัดเก็บในแฟ้มปทานุกรมข้อมูลไว้ในตัวแปรที่ใช้ร่วมกัน ได้แก่ LISTATTDB LISTATTREL LISTATTATT LISTATTCONS LISTATTCOLCONS LISTATTREFCONS
5) เข้าสู่การทำงานของระบบเพื่อรอรับคำสั่งจากผู้ใช้
6) จบการทำงาน

2. **ExitSystem** เป็นทำงานในส่วนของการออกจากการทำงานของระบบ

ตัวแปรในการเรียกใช้

-

ขั้นตอนวิธี

1) ปิดแฟ้มปทานุกรมข้อมูลและแฟ้มข้อมูลที่จัดเก็บข้อมูลของฐานข้อมูลที่ใช้งาน (ถ้าเปิดอยู่) และถ้าข้อมูลในในบัฟเฟอร์มีการเปลี่ยนแปลงภายก็จะบันทึกการเปลี่ยนแปลงนั้นกลับลงดิสก์ให้โดยอัตโนมัติ
2) คืนบัฟเฟอร์ที่ใช้จัดเก็บปทานุกรมข้อมูล และข้อมูลของฐานข้อมูลในระหว่างการประมวลผล
3) ออกจากการทำงานของระบบ
4) จบการทำงาน

3. **DbUse** เป็นทำงานในส่วนของการเปิดฐานข้อมูลที่ต้องการใช้งาน

ตัวแปรในการเรียกใช้

- bname ชื่อฐานข้อมูลที่ต้องการใช้งาน

ขั้นตอนวิธี

1) ตรวจสอบว่ามีชื่อฐานข้อมูลที่ต้องการใช้งานอยู่ในระบบหรือไม่
2) ถ้าไม่มีชื่อฐานข้อมูลที่ต้องการใช้งานอยู่ในระบบ ให้แสดงข้อความแจ้งผู้ใช้ทราบและจบการทำงาน
3) ตรวจสอบว่าฐานข้อมูลใดเปิดใช้งานอยู่แล้วหรือไม่
4) ถ้ามีฐานข้อมูลเปิดอยู่ และไม่ใช่ฐานข้อมูลที่กำลังต้องการเปิดใช้งาน จะทำการปิดฐานข้อมูลเดิมที่เปิดอยู่ และถ้าข้อมูลในฐานข้อมูลที่จะปิดมีการเปลี่ยนแปลงภายในบัฟเฟอร์ก็จะบันทึกการเปลี่ยนแปลงนั้นกลับลงดิสก์ให้โดยอัตโนมัติ ซึ่งในขณะใดขณะหนึ่ง ระบบจะอนุญาตให้เปิดใช้งานฐานข้อมูลได้เพียงฐานข้อมูลเดียวเท่านั้น

5) เปิดฐานข้อมูลที่ต้องการใช้งาน
6) จบการทำงาน

4. **DbClose** เป็นทำงานในส่วนของการปิดฐานข้อมูลที่กำลังเปิดใช้งาน

ตัวแปรในการเรียกใช้

-

ขั้นตอนวิธี

1) ตรวจสอบว่ามีฐานข้อมูลเปิดใช้งานอยู่ในระบบหรือไม่
2) ถ้ามีฐานข้อมูลเปิดใช้งานอยู่ในระบบ ปิดฐานข้อมูลที่กำลังใช้งานอยู่ที่เปิดอยู่ ถ้าข้อมูลในฐานข้อมูลมีการเปลี่ยนแปลงภายในบัฟเฟอร์ก็จะบันทึกการเปลี่ยนแปลงนั้นกลับลงดิสก์ให้โดยอัตโนมัติ
3) จบการทำงาน

5. **DbCreate** เป็นทำงานในส่วนของการสร้างฐานข้อมูลใหม่ในระบบ

ตัวแปรในการเรียกใช้

- dbname ชื่อฐานข้อมูลใหม่ที่ต้องการสร้าง

ขั้นตอนวิธี

1) ตรวจสอบว่ามีชื่อฐานข้อมูลที่ต้องการสร้างอยู่ในระบบแล้วหรือไม่
2) ถ้าไม่มีชื่อฐานข้อมูลที่ต้องการสร้างอยู่ในระบบแล้วจะแสดงข้อความแจ้งผู้ใช้ทราบว่าชื่อฐานข้อมูลซ้ำกับชื่อฐานข้อมูลเดิมที่มีอยู่ในระบบแล้ว และจบการทำงาน
3) บันทึกรายละเอียดเกี่ยวกับฐานข้อมูลใหม่ที่สร้าง โดยจะจัดเก็บไว้ในรีเลชันที่จัดเก็บรายชื่อฐานข้อมูล
4) จบการทำงาน

6. **DbDrop** เป็นทำงานในส่วนของการลบฐานข้อมูลออกจากระบบ

ตัวแปรในการเรียกใช้

- dbname ชื่อฐานข้อมูลที่ต้องการลบ

ขั้นตอนวิธี

1) ตรวจสอบว่าฐานข้อมูลที่ต้องการลบมีอยู่ระบบหรือไม่

2) ถ้าไม่มีฐานข้อมูลที่ต้องการลบอยู่ในระบบ ให้แสดงข้อความแจ้งผู้ใช้ทราบว่าไม่มีชื่อฐานข้อมูลนี้ในระบบ และจบการทำงาน
3) ตรวจสอบว่าฐานข้อมูลที่ต้องการลบเปิดใช้งานอยู่หรือไม่
4) ถ้าฐานข้อมูลที่ต้องการลบเป็นฐานข้อมูลที่กำลังเปิดใช้งาน จะแสดงข้อความแจ้งผู้ใช้ทราบว่าไม่สามารถลบฐานข้อมูลที่กำลังเปิดใช้งานได้ และจบการทำงาน
5) ลบข้อมูลต่าง ๆ ที่เกี่ยวข้องกับฐานข้อมูลที่ต้องการลบ
   - ข้อมูลทุเปิลทั้งหมดของทุกรีเลชันในฐานข้อมูล
   - รายชื่อแอตทริบิวทั้งหมดของทุกรีเลชันในฐานข้อมูล
   - รายชื่อรีเลชันทั้งหมดในฐานข้อมูล
   - การกำหนดคีย์หลัก คีย์รอง และเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิว
   - ค่าคีย์ทั้งหมดของทุกรีเลชันในฐานข้อมูล
6) ลบรายชื่อฐานข้อมูลออกจากรีเลชันที่จัดเก็บรายชื่อฐานข้อมูล
7) จบการทำงาน

7. **DbShow** เป็นทำงานในส่วนของการแสดงรายชื่อฐานข้อมูลทั้งหมดที่จัดเก็บในระบบ

ตัวแปรในการเรียกใช้

-

ขั้นตอนวิธี

1) แสดงรายชื่อฐานข้อมูลทั้งหมดในระบบจากรีเลชันที่จัดเก็บรายชื่อฐานข้อมูล
2) จบการทำงาน

8. **RelCreate** เป็นทำงานในส่วนของการสร้างรีเลชันใหม่ในฐานข้อมูลที่กำลังเปิดใช้งาน

ตัวแปรในการเรียกใช้

- relname ชื่อรีเลชันที่ต้องการสร้างไว้ฐานข้อมูลที่กำลังใช้งาน
- lattname รายชื่อแอตทริบิว และรายละเอียดของแอตทริบิวที่ต้องการกำหนดไว้ในรีเลชัน
- lpkey รายชื่อแอตทริบิวที่ต้องการกำหนดเป็นคีย์หลักของรีเลชัน
- lskey รายชื่อแอตทริบิวที่ต้องการกำหนดเป็นคีย์รองของรีเลชัน
- lref รายชื่อแอตทริบิว และรายละเอียดของการกำหนดคีย์นอก หรือเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิว

ขั้นตอนวิธี

1) ตรวจสอบว่ามีฐานข้อมูลเปิดใช้งานอยู่ในระบบหรือไม่
2) ถ้าไม่มีฐานข้อมูลใดเปิดใช้งานอยู่ในระบบ จะแสดงข้อความแจ้งผู้ใช้ทราบว่าผู้ใช้ไม่ได้เปิดใช้งานฐานข้อมูลใดอยู่ และจบการทำงาน
3) ตรวจสอบว่ามีชื่อรีเลชันที่ต้องการสร้างอยู่ในฐานข้อมูลที่กำลังใช้งานอยู่แล้วหรือไม่
4) ถ้ามีชื่อรีเลชันที่ต้องการสร้างใหม่จัดเก็บในฐานข้อมูลอยู่แล้ว จะแสดงข้อความแจ้งผู้ใช้ทราบว่าชื่อรีเลชันซ้ำกับชื่อรีเลชันเดิมในฐานข้อมูลแล้ว และจบการทำงาน
5) ตรวจสอบความถูกต้องข้อมูลต่าง ๆ ที่เกี่ยวข้อง (ถ้ามี) ดังนี้
   - ตรวจสอบการกำหนดคีย์หลัก
   - ตรวจสอบการกำหนดคีย์รอง
   - ตรวจสอบการกำหนดเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิว
6) บันทึกรายละเอียดเกี่ยวกับรีเลชันใหม่ที่สร้าง (ถ้ามี) โดยจัดเก็บไว้ในรีเลชันต่างๆ ที่เกี่ยวข้องดังนี้
   - รีเลชันที่จัดเก็บการกำหนดคีย์หลัก คีย์รอง และเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิว
   - รีเลชันที่จัดเก็บรายชื่อแอตทริบิวที่มีการกำหนดคีย์หลัก คีย์รอง และเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิวในรีเลชัน
   - รีเลชันที่จัดเก็บรายละเอียดเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิว
7) เพิ่มรายชื่อแอตทริบิวของรีเลชันที่สร้างขึ้นใหม่ในฐานข้อมูลที่กำลังใช้งานอยู่
8) เพิ่มรายชื่อรีเลชันในฐานข้อมูลที่กำลังใช้งานอยู่
9) ปรับปรุงรีเลชันที่จัดเก็บรายชื่อฐานข้อมูล
10) จบการทำงาน

9. RelDrop เป็นทำงานในส่วนของการลบรีเลชันออกจากฐานข้อมูล

ตัวแปรในการเรียกใช้

- relname ชื่อรีเลชันที่ต้องการลบ

ขั้นตอนวิธี

1) ตรวจสอบว่ามีฐานข้อมูลเปิดใช้งานอยู่หรือไม่

2) ถ้าไม่มีฐานข้อมูลใดเปิดใช้งานอยู่ในระบบ จะแสดงข้อความแจ้งผู้ใช้ทราบว่าผู้ใช้ไม่ได้เปิดใช้งานฐานข้อมูลใดอยู่ และจบการทำงาน
3) ตรวจสอบว่ามีชื่อรีเลชันที่ต้องการลบอยู่ในฐานข้อมูลที่กำลังใช้งานอยู่หรือไม่
4) ถ้าไม่มีชื่อรีเลชันที่ต้องการในฐานข้อมูลที่กำลังใช้งานอยู่ จะแสดงข้อความแจ้งผู้ใช้ทราบว่าไม่พบชื่อรีเลชันในฐานข้อมูล และจบการทำงาน
5) ตรวจสอบว่ารีเลชันที่ต้องการลบมีคีย์นอกของรีเลชันอื่นในฐานข้อมูลเดียวกันอ้างถึงค่าของคีย์หลักในรีเลชันที่ต้องการลบหรือไม่
6) ถ้ามีคีย์นอกของรีเลชันอื่นในฐานข้อมูลเดียวกัน อ้างถึงค่าของคีย์หลักในรีเลชันที่ต้องการลบ จะแสดงข้อความแจ้งให้ผู้ใช้ทราบว่าไม่สามารถลบรีเลชันได้ และจบการทำงาน
7) ลบข้อมูลต่าง ๆ ที่เกี่ยวข้องกับรีเลชันที่ต้องการลบ
   - ข้อมูลทูเปิลทั้งหมดของรีเลชันที่ต้องการลบ
   - รายชื่อแอตทริบิวทั้งหมดของรีเลชันที่ต้องการลบ
   - การกำหนดคีย์หลัก คีย์รอง และเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิวของรีเลชันที่ต้องการลบ
   - ข้อมูลดัชนีของรีเลชันที่ต้องการลบ
8) ลบรายชื่อรีเลชันออกจากรีเลชันที่จัดเก็บรายชื่อรีเลชัน
9) จบการทำงาน

10. RelShow เป็นทำงานในส่วนของการแสดงรายชื่อรีเลชันทั้งหมดในฐานข้อมูลที่กำลังใช้งาน

ตัวแปรในการเรียกใช้

-

ขั้นตอนวิธี

1) ตรวจสอบว่ามีฐานข้อมูลเปิดใช้งานอยู่ในระบบหรือไม่
2) ถ้าไม่มีฐานข้อมูลใดเปิดใช้งานอยู่ในระบบ จะแสดงข้อความแจ้งผู้ใช้ทราบว่าผู้ใช้ไม่ได้เปิดใช้งานฐานข้อมูลใดอยู่ และจบการทำงาน
3) แสดงรายชื่อรีเลชันทั้งหมดจากรีเลชันที่จัดเก็บรายชื่อรีเลชันของฐานข้อมูลที่กำลังใช้อยู่
4) จบการทำงาน

11. AttShow เป็นทำงานในส่วนของการ

ตัวแปรในการเรียกใช้

- relname ชื่อรีเลชันที่ต้องการเพิ่มทุเปิล

ขั้นตอนวิธี

1) ตรวจสอบว่ามีฐานข้อมูลเปิดใช้งานอยู่หรือไม่
2) ถ้าไม่มีฐานข้อมูลใดเปิดใช้งานอยู่ในระบบ จะแสดงข้อความแจ้งผู้ใช้ทราบว่าผู้ใช้ไม่ได้เปิดใช้งานฐานข้อมูลใดอยู่ และจบการทำงาน
3) ตรวจสอบว่ามีชื่อรีเลชันที่ต้องการเพิ่มทุเปิลอยู่ในฐานข้อมูลที่กำลังใช้งานอยู่หรือไม่
4) ถ้าไม่มีชื่อรีเลชันที่ต้องการในฐานข้อมูลที่กำลังใช้งานอยู่ จะแสดงข้อความแจ้งผู้ใช้ทราบว่าไม่พบชื่อรีเลชันในฐานข้อมูล และจบการทำงาน
5) แสดงรายชื่อแอตทริบิวทั้งหมดของรีเลชันจากรีเลชันที่จัดเก็บรายชื่อแอตทริบิว
6) จบการทำงาน

12. InsertTuple เป็นทำงานในส่วนของการเพิ่มทุเปิลใหม่ในรีเลชันที่ระบุ

ตัวแปรในการเรียกใช้

- relname ชื่อรีเลชันที่ต้องการเพิ่มทุเปิล
- lattname รายชื่อแอตทริบิวที่ต้องการกำหนดค่าของทุเปิล
- lvalue ค่าของแอตทริบิวที่ผู้ใช้กำหนดให้กับทุเปิล

ขั้นตอนวิธี

1) ตรวจสอบว่ามีฐานข้อมูลเปิดใช้งานอยู่หรือไม่
2) ถ้าไม่มีฐานข้อมูลใดเปิดใช้งานอยู่ในระบบ จะแสดงข้อความแจ้งผู้ใช้ทราบว่าผู้ใช้ไม่ได้เปิดใช้งานฐานข้อมูลใดอยู่ และจบการทำงาน
3) ตรวจสอบว่ามีชื่อรีเลชันที่ต้องการเพิ่มทุเปิลอยู่ในฐานข้อมูลที่กำลังใช้งานอยู่หรือไม่
4) ถ้าไม่มีชื่อรีเลชันที่ต้องการในฐานข้อมูลที่กำลังใช้งานอยู่ จะแสดงข้อความแจ้งผู้ใช้ทราบว่าไม่พบชื่อรีเลชันในฐานข้อมูล และจบการทำงาน
5) ตรวจสอบจำนวนแอตทริบิวที่ผู้ใช้ระบุ กรณีผู้ใช้ไม่ได้ระบุรายชื่อแอตทริบิว จะหมายถึงจำนวนแอตทริบิวทั้งหมดในรีเลชัน ถ้าจำนวนแอตทริบิวกับจำนวน

ค่าที่ผู้ใช้กำหนดไม่เท่ากัน ก็จะแสดงข้อความแจ้งให้ผู้ใช้ทราบว่าจำนวนค่าแอตทริบิวไม่ถูกต้อง และจบการทำงาน

6) ถ้าผู้ใช้ระบุชื่อรายชื่อแอตทริบิวที่ต้องการกำหนดค่าของทุเปิล จะตรวจสอบว่ามีชื่อแอตทริบิวใดไม่ได้อยู่ในรีเลชันที่ต้องการจะเพิ่มทุเปิล จะแสดงข้อความแจ้งให้ผู้ใช้ทราบว่ามีชื่อแอตทริบิวไม่ได้อยู่ในรีเลชันที่ต้องการเพิ่มทุเปิล และจบการทำงาน
7) ตรวจค่าแอตทริบิวต่างๆ ที่ผู้ใช้กำหนดให้กับทุเปิล ว่าชนิดข้อมูลและขนาดความยาวของข้อมูลที่จะจัดเก็บถูกต้องตามที่กำหนดไว้ในโครงสร้างรีเลชันหรือไม่
8) ถ้าชนิดข้อมูลและขนาดความยาวของข้อมูลที่จะจัดเก็บไม่ถูกต้องตามที่กำหนดไว้ในโครงสร้างรีเลชัน จะแสดงข้อความแจ้งให้ผู้ใช้ทราบว่าค่าของแอตทริบิวไม่ถูกต้อง และจบการทำงาน
9) ถ้ามีบางแอตทริบิวของทุเปิลที่ผู้ใช้ไม่ได้กำหนดค่า จะใส่เป็นค่าว่างให้อัตโนมัติยกเว้นว่ามีการค่าปริยายสำหรับแอตทริบิวนั้น
10) ตรวจสอบการใส่ค่าว่างของแอตทริบิว (ถ้ามี) ถ้ามีค่าแอตทริบิวใดเป็นค่าว่างสำหรับค่าแอตทริบิวที่เป็นค่าว่างไม่ได้ จะแสดงข้อความแจ้งให้ผู้ใช้ทราบว่าค่าแอตทริบิวเป็นค่าว่างไม่ได้ และจบการทำงาน
11) ตรวจสอบการค่าแอตทริบิวในส่วนของการกำหนดเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิว(ถ้ามี) ถ้ามีค่าแอตทริบิวใดที่เป็นคีย์นอก และไม่ปรากฏในคีย์หลักของอีกรีเลชันหนึ่งที่อ้างถึง จะแสดงข้อความแจ้งให้ผู้ใช้ทราบว่าไม่พบค่าแอตทริบิวที่เป็นคีย์นอกเป็นค่าคีย์หลักของอีกรีเลชันหนึ่งที่อ้างถึง และจบการทำงาน
12) ตรวจสอบค่าคีย์ในข้อมูลดัชนีของคีย์ทั้งหมดในรีเลชัน (ถ้ามี) ถ้ามีค่าคีย์ซ้ำจะแสดงข้อความแจ้งให้ผู้ใช้ทราบว่ามีการกำหนดค่าคีย์ซ้ำ และจบการทำงาน
13) เพิ่มทุเปิลในรีเลชัน
14) เพิ่มค่าคีย์ในข้อมูลดัชนีของคีย์ทั้งหมดในรีเลชัน
15) ปรับปรุงจำนวนทุเปิลในรีเลชันที่จัดเก็บรายชื่อรีเลชัน
16) จบการทำงาน

13. UpdateTuple เป็นทำงานในส่วนของการปรับปรุงทุเปิลเดิมในรีเลชันที่ระบุ

ตัวแปรในการเรียกใช้

- relname ชื่อรีเลชันที่ต้องการปรับปรุงทุเปิล
- lattname รายชื่อแอตทริบิวที่ต้องการปรับปรุงค่าของทุเปิล
- lvalue ค่าของแอตทริบิวที่ผู้ใช้ใส่ค่าใหม่ให้กับทุเปิล
- lcond เงื่อนไขในการค้นหาทุเปิลที่ต้องการปรับปรุง

ขั้นตอนวิธี

1) ตรวจสอบว่ามีฐานข้อมูลเปิดใช้งานอยู่หรือไม่
2) ถ้าไม่มีฐานข้อมูลใดเปิดใช้งานอยู่ในระบบ จะแสดงข้อความแจ้งผู้ใช้ทราบว่าผู้ใช้ไม่ได้เปิดใช้งานฐานข้อมูลใดอยู่ และจบการทำงาน
3) ตรวจสอบว่ามีชื่อรีเลชันที่ต้องการปรับปรุงอยู่ในฐานข้อมูลที่กำลังใช้งานอยู่หรือไม่
4) ถ้าไม่มีชื่อรีเลชันที่ต้องการในฐานข้อมูลที่กำลังใช้งานอยู่ จะแสดงข้อความแจ้งผู้ใช้ทราบว่าไม่พบชื่อรีเลชันในฐานข้อมูล และจบการทำงาน
5) ตรวจสอบรายชื่อแอตทริบิวที่ต้องการปรับปรุงค่าของทุเปิล ถ้ามีชื่อแอตทริบิวใดไม่ได้อยู่ในรีเลชัน จะแสดงข้อความแจ้งให้ผู้ใช้ทราบว่ามีชื่อแอตทริบิวไม่ได้อยู่ในรีเลชัน และจบการทำงาน
6) ตรวจค่าแอตทริบิวต่างๆ ที่ผู้ใช้ใส่ค่าใหม่ให้กับทุเปิล ว่าชนิดข้อมูลและขนาดความยาวของข้อมูลที่จะจัดเก็บถูกต้องตามที่กำหนดไว้ในโครงสร้างรีเลชันหรือไม่
7) ถ้าชนิดข้อมูลและขนาดความยาวของข้อมูลที่จะจัดเก็บไม่ถูกต้องตามที่กำหนดไว้ในโครงสร้างรีเลชัน จะแสดงข้อความแจ้งให้ผู้ใช้ทราบว่าค่าของแอตทริบิวไม่ถูกต้อง และจบการทำงาน
8) ตรวจสอบการใส่ค่าว่างของแอตทริบิว (ถ้ามี) ถ้ามีค่าแอตทริบิวใดเป็นค่าว่างสำหรับค่าแอตทริบิวที่เป็นค่าว่างไม่ได้ จะแสดงข้อความแจ้งให้ผู้ใช้ทราบว่าค่าแอตทริบิวเป็นค่าว่างไม่ได้ และจบการทำงาน
9) ตรวจสอบรายชื่อแอตทริบิวที่กำหนดเป็นเงื่อนไขในการค้นหาทุเปิลที่ต้องการปรับปรุง (ถ้ามี) ถ้ามีชื่อแอตทริบิวใดไม่ได้อยู่ในรีเลชัน จะแสดงข้อความแจ้งให้ผู้ใช้ทราบว่ามีชื่อแอตทริบิวไม่ได้อยู่ในรีเลชัน และจบการทำงาน

10) ปรับปรุงแต่ละทุเปิลของรีเลชัน โดยอาจจะปรับปรุงทุกทุเปิลในรีเลชัน หรือบางทุเปิลที่เป็นไปตามเงื่อนไขที่ระบุ (ถ้ามี) โดยในการปรับปรุงแต่ละทุเปิลจะต้องตรวจสอบก่อนว่าสามารถปรับปรุงค่าของทุเปิลนั้นได้หรือไม่
   - ตรวจสอบค่าคีย์ในข้อมูลดัชนีของคีย์ทั้งหมดในรีเลชัน (ถ้ามี) ถ้ามีค่าใหม่ของแอตทริบิวที่เป็นคีย์มีค่าซ้ำกับค่าคีย์เดิมของทุเปิลอื่นในรีเลชัน จะไม่ปรับปรุงทุเปิลนี้ และไปทำงานข้อ 11)
   - ตรวจสอบการค่าแอตทริบิวในส่วนของการกำหนดเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิว(ถ้ามี)
     - กรณีถ้าค่าใหม่ของแอตทริบิวที่เป็นคีย์นอก ไม่ปรากฏในคีย์หลักของอีกรีเลชันหนึ่งที่อ้างถึง จะไม่ปรับปรุงทุเปิลนี้ และไปทำงานข้อ 11)
     - กรณีถ้าแอตทริบิวที่ต้องการปรับปรุงเป็นคีย์หลัก และมีคีย์นอกของอีกเลชันหนึ่งที่อ้างถึงคีย์หลักนี้จะทำการปรับปรุงทุกทุเปิลของรีเลชันอื่นๆ ที่มีค่าคีย์นอกสัมพันธ์กับค่าคีย์หลักที่ต้องการปรับปรุงด้วย โดยการปรับปรุงทุเปิลของรีเลชันอื่นๆ ก็จะทำการตรวจสอบในส่วนของการกำหนดค่าคีย์ เงื่อนไขบังคับการอ้างอิงค่าแอตทริบิว และเงื่อนไขบังคับการตรวจสอบค่าแอตทริบิว เช่นเดียวกับการปรับปรุงทุเปิลของรีเลชันนี้ด้วย ถ้ามีทุเปิลของรีเลชันอื่นไม่สามารถปรับปรุงได้ เนื่องจากไม่ตรงตามเงื่อนไขที่ตรวจสอบดังกล่าว จะส่งผลให้ไม่สามารถปรับปรุงทุเปิลของรีเลชันนี้ได้เช่นเดียวกัน และไปทำงานข้อ 11)
   - ปรับปรุงทุเปิลในรีเลชัน
   - ปรับปรุงค่าคีย์ในข้อมูลดัชนีของคีย์ทั้งหมดในรีเลชัน
11) กลับไปทำข้อ 10) จนกระทั่งอ่านครบทุกทุเปิลในรีเลชันที่ต้องการปรับปรุง
12) แสดงจำนวนทุเปิลที่ค้นพบทั้งหมดตามเงื่อนไข และจำนวนทุเปิลที่สามารถปรับปรุงค่าของแอตทริบิวในทุเปิลของรีเลชันที่ต้องการปรับปรุงทุเปิล
13) จบการทำงาน

14. DeleteTuple เป็นทำงานในส่วนของการลบทุเปิลเดิมในรีเลชันที่ระบุ

ตัวแปรในการเรียกใช้

- relname ชื่อรีเลชันที่ต้องการลบทุเปิล
- lcond เงื่อนไขในการค้นหาทุเปิลที่ต้องการลบ

ขั้นตอนวิธี

1) ตรวจสอบว่ามีฐานข้อมูลเปิดใช้งานอยู่หรือไม่
2) ถ้าไม่มีฐานข้อมูลใดเปิดใช้งานอยู่ในระบบ จะแสดงข้อความแจ้งผู้ใช้ทราบว่าผู้ใช้ไม่ได้เปิดใช้งานฐานข้อมูลใดอยู่ และจบการทำงาน
3) ตรวจสอบว่ามีชื่อรีเลชันที่ต้องการลบทุเปิล อยู่ในฐานข้อมูลที่กำลังใช้งานอยู่หรือไม่
4) ถ้าไม่มีชื่อรีเลชันที่ต้องการในฐานข้อมูลที่กำลังใช้งานอยู่ จะแสดงข้อความแจ้งผู้ใช้ทราบว่าไม่พบชื่อรีเลชันในฐานข้อมูล และจบการทำงาน
5) ตรวจสอบรายชื่อแอตทริบิวที่กำหนดเป็นเงื่อนไข (ถ้ามี) ในการค้นหาทุเปิลที่ต้องการลบออกจากรีเลชัน ถ้ามีชื่อแอตทริบิวใดไม่ได้อยู่ในรีเลชัน จะแสดงข้อความแจ้งให้ผู้ใช้ทราบว่ามีชื่อแอตทริบิวไม่ได้อยู่ในรีเลชัน และจบการทำงาน
6) ตรวจสอบการค่าแอตทริบิวในส่วนของการกำหนดเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิว(ถ้ามี) โดยถ้ามีแอตทริบิวในรีเลชันที่ต้องการลบทุเปิลเป็นคีย์หลักและมีคีย์นอกของอีกเลชันหนึ่งที่อ้างถึงคีย์หลักของรีเลชันนี้จะตรวจสอบว่า มีกำหนดเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิวแบบลบต่อเนื่องกันหรือไม่
   - ถ้ามีรีเลชันที่อ้างถึงรีเลชันนี้ แต่ไม่ได้กำหนดเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิวแบบลบต่อเนื่องกันก็แสดงว่าไม่สามารถลบทุเปิลใดๆของรีเลชันนี้ได้ จะแสดงข้อความแจ้งให้ผู้ใช้ทราบว่าไม่สามารถลบทุเปิลในรีเลชันได้ และจบการทำงาน
   - ถ้ามีรีเลชันที่อ้างถึงรีเลชันที่ต้องการลบทุเปิลนี้ และกำหนดเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิวเป็นแบบลบต่อเนื่องกัน แสดงว่าจะต้องลบทุเปิลทั้งหมดของเรีเลชันที่อ้างถึงรีเลชันที่ต้องการลบด้วย ดังนั้นจะจัดเก็บรายชื่อของทุกรีเลชันที่อ้างถึงรีเลชันที่ต้องการลบทุเปิล ซึ่งรีเลชันที่จัดเก็บทั้งหมดนี้ก็จะถูกตรวจสอบในลักษณะเดียวกับรีเลชันที่ต้องการลบทุเปิลในข้อ 6) วนทำซ้ำไปเรื่อยๆ จนกระทั่งได้รายชื่อรีเลชันทั้งหมดที่ต้องการลบทุเปิลแบบต่อเนื่อง
7) ลบทุเปิลในรีเลชัน แยกเป็นสองกรณี
   - กรณีลบทุกทุเปิลของรีเลชันทั้งหมดที่จัดเก็บไว้ในข้อ 6) จะลบทุกทุเปิลพร้อมกับค่าคีย์ทั้งหมดที่มีการอ้างอิงค่าแอตทริบิวระหว่างรีเลชันด้วย

- กรณีลบบางทุเปิลในรีเลชัน เมื่อมีการกำหนดเงื่อนไขในการค้นหาทุเปิลที่ต้องการลบ โดยจะทำการค้นหาทีละทุเปิลในรีเลชันจนครบทุกทุเปิลในรีเลชัน ซึ่งจะลบเฉพาะทุเปิลที่ตรงตามเงื่อนไขที่กำหนดเท่านั้น และในการลบแต่ละทุเปิลที่จะต้องลบข้อมูลที่เกี่ยวข้องด้วยดังนี้
  - ลบค่าคีย์ทั้งหมดของทุเปิลที่ต้องการลบออกจากข้อมูลดัชนีที่เกี่ยวข้อง
  - นำค่าคีย์หลักของทุเปิลที่ต้องการลบ ไปค้นหาทุเปิลในรีเลชันอื่นที่มีค่าคีย์นอกอ้างถึงคีย์หลักของทุเปิลนี้ (ถ้ามี) เพื่อทำการลบทุเปิลของรีเลชันเหล่านั้นด้วย และวนทำซ้ำจนกระทั่งลบทุเปิลของรีเลชันที่มีเงื่อนไขบังคับการอ้างอิงค่าแอตทริบิวกันตามที่จัดเก็บไว้ในข้อ 6)

8) แสดงจำนวนทุเปิลที่ลบออกจากรีเลชันที่ระบุ
9) จบการทำงาน

15. SelectTuple เป็นทำงานในส่วนของการแสดงรายการทุเปิลในรีเลชันที่ระบุ

ตัวแปรในการเรียกใช้

- relname ชื่อรีเลชันที่ต้องการปรับปรุงทุเปิล
- lattname รายชื่อแอตทริบิวที่ต้องการแสดงค่าของทุเปิล
- lcond เงื่อนไขในการค้นหาทุเปิลที่ต้องการแสดงค่าทุเปิล

ขั้นตอนวิธี

1) ตรวจสอบว่ามีฐานข้อมูลเปิดใช้งานอยู่หรือไม่
2) ถ้าไม่มีฐานข้อมูลใดเปิดใช้งานอยู่ในระบบ จะแสดงข้อความแจ้งผู้ใช้ทราบว่าผู้ใช้ไม่ได้เปิดใช้งานฐานข้อมูลใดอยู่ และจบการทำงาน
3) ตรวจสอบว่ามีชื่อรีเลชันที่ต้องการแสดงรายการทุเปิลอยู่ในฐานข้อมูลที่กำลังใช้งานอยู่หรือไม่
4) ถ้าไม่มีชื่อรีเลชันที่ต้องการในฐานข้อมูลที่กำลังใช้งานอยู่ จะแสดงข้อความแจ้งผู้ใช้ทราบว่าไม่พบชื่อรีเลชันในฐานข้อมูล และจบการทำงาน
5) ตรวจสอบรายชื่อแอตทริบิวที่ต้องการแสดงค่าของทุเปิล ถ้ามีชื่อแอตทริบิวใดไม่ได้อยู่ในรีเลชัน จะแสดงข้อความแจ้งให้ผู้ใช้ทราบว่ามีชื่อแอตทริบิวไม่ได้อยู่ในรีเลชัน และจบการทำงาน

6) ตรวจสอบรายชื่อแอตทริบิวที่กำหนดเป็นเงื่อนไขในการค้นหาทุเปิลที่ต้องการแสดงค่าทุเปิล (ถ้ามี) ถ้ามีชื่อแอตทริบิวใดไม่ได้อยู่ในรีเลชัน จะแสดงข้อความแจ้งให้ผู้ใช้ทราบว่ามีชื่อแอตทริบิวไม่ได้อยู่ในรีเลชัน และจบการทำงาน
7) แสดงรายการทุเปิลในรีเลชันที่ต้องการตามเงื่อนไขในการค้นหาทุเปิล (ถ้ามี)
8) จบการทำงาน