บทที่ 3

แนวคิดและทฤษฎี

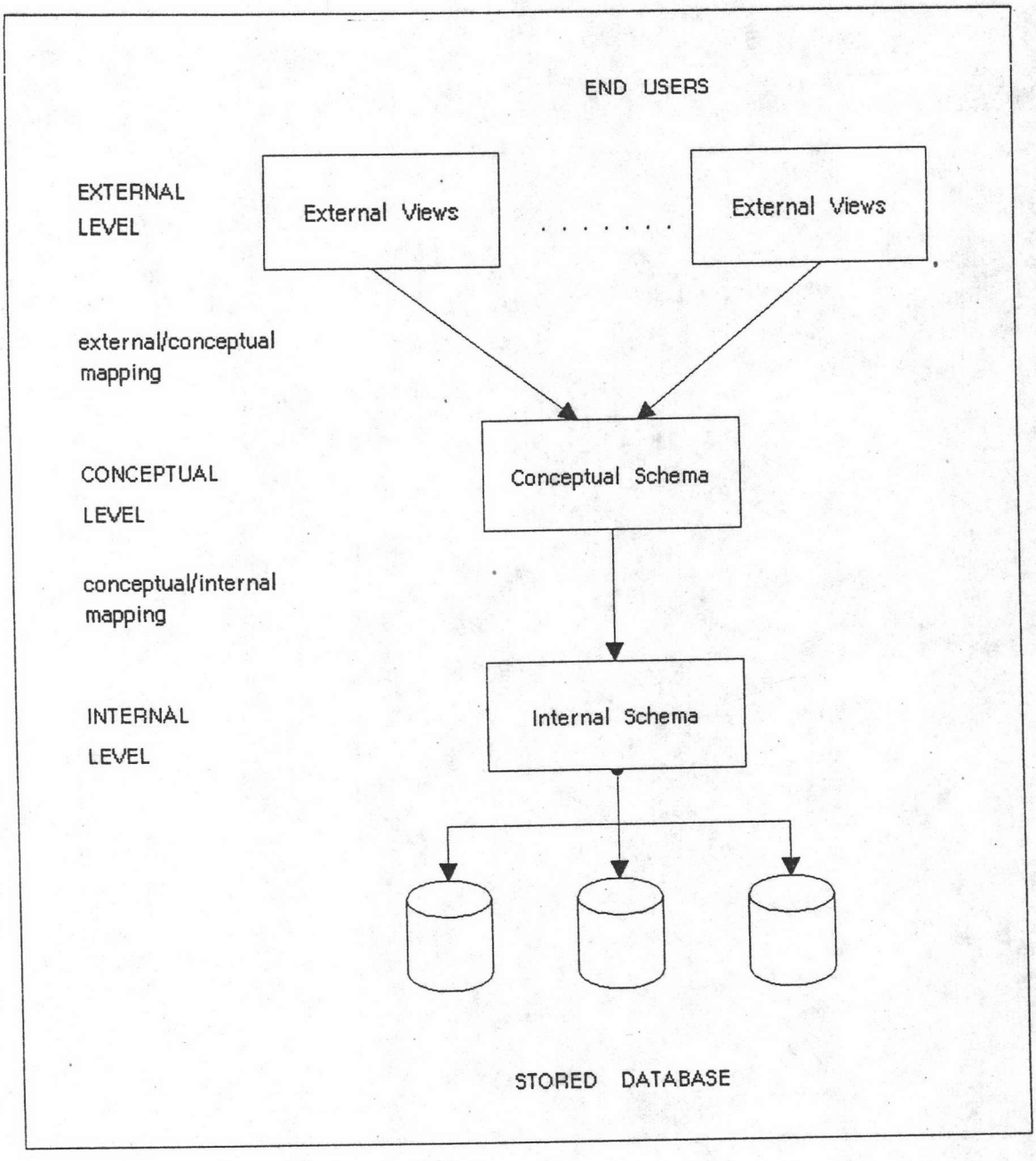
ฐานข้อมูล (Database)

ฐานข้อมูลเป็นที่เก็บรวบรวมข้อมูลของระบบเข้าไว้ด้วยกัน เพื่อให้ผู้ใช้และงานประยุกต์ต่าง ๆ สามารถดำเนินการ (เรียกใช้, เพิ่ม, ลบ หรือแก้ไข) กับข้อมูลนั้น ๆ ได้ โดยอาศัยระบบจัดการฐานข้อมูล (Database Management System-DBMS) เป็นตัวควบคุม วัตถุประสงค์หลักของระบบจัดการฐานข้อมูลคือ การจัดหามุมมองของข้อมูลให้กับผู้ใช้ โดยระบบจะซ่อนรายละเอียดต่าง ๆ ของข้อมูลเหล่านั้นว่าถูกเก็บและถูกบำรุงรักษาอย่างไร เพื่อให้ข้อมูลสามารถถูกดำเนินการได้อย่างมีประสิทธิภาพมากที่สุด

สถาปัตยกรรมของฐานข้อมูล

ในปี 1975 ANSI (American National Standard Institute) ได้กำหนดมาตรฐานที่เรียกว่า สถาปัตยกรรมสามแบบแผน (Three-Schema Architecture) ขึ้น ซึ่งต่อมารู้จักกันในนามของ "สถาปัตยกรรมของ ANSI/SPARC" โดยทำการอธิบายโครงสร้างของฐานข้อมูลโดยรวม เพื่อให้เห็นความแตกต่างระหว่างฐานข้อมูลเชิงกายภาพ และสิ่งที่ผู้ใช้มองเห็น ทั้งนี้เพราะผู้ใช้มีมุมมอง และวิธีการมองข้อมูลต่างกัน นอกจากนี้ผู้ใช้ก็ไม่ควรจะยุ่งยากกับความซับซ้อนของโครงสร้างที่เก็บฐานข้อมูล และเพื่อให้มีความยืดหยุ่นต่อการเปลี่ยนแปลง ไม่ว่าจะเป็นการเปลี่ยนที่โครงสร้างแฟ้มข้อมูล หรือเปลี่ยนแปลงอุปกรณ์ต่าง ๆ สถาปัตยกรรมสามแบบแผนดังกล่าว คือ [16]

1. แบบแผนฐานข้อมูลภายนอก (External schemas) เป็นระดับที่ใกล้กับผู้ใช้มากที่สุด และเป็นสิ่งที่ผู้ใช้คิดเกี่ยวกับข้อมูล โดยจะอธิบายถึงวิว (view) ที่ผู้ใช้สนใจ ซึ่งข้อมูลที่เก็บจริงอาจมีมากกว่าที่ผู้ใช้ต้องการหรือข้อมูลตัวเดียวกันผู้ใช้อาจมองต่างกัน แบบแผนนี้สามารถอธิบายได้ โดยใช้โมเดลข้อมูลเชิงตรรก (Logical data model หรือ High level data model) หรือโมเดลข้อมูลตามโครงสร้าง ที่ใช้ในการติดตั้งระบบ (Implementation data model)



รูปที่ 3.1 แสดงสถาปัตยกรรมของฐานข้อมูล

2. แบบแผนฐานข้อมูลภายใน (Internal schemas) เป็นการอธิบายการเก็บข้อมูลทางกายภาพจริงๆ มองข้อมูลโดยมุมมองของระบบจัดการฐานข้อมูล (Database Management System - DBMS) โดยใช้โครงสร้างข้อมูล (data structure) และการจัดการแฟ้มข้อมูล (file organization) เป็นตัวอธิบาย และจะทำงานร่วมกับระบบปฏิบัติการ (operating system) ในการเก็บข้อมูลลงที่เก็บข้อมูลสำรอง แบบแผนระดับนี้ สามารถอธิบายได้ด้วยโมเดลข้อมูลเชิงกายภาพ (Physical data model)

3. แบบแผนฐานข้อมูลเชิงมโนภาพ (Conceptual schemas) เป็นตัวที่ใช้เชื่อมระหว่างแบบแผนฐานข้อมูลภายนอกกับภายใน อธิบายรายละเอียดของฐานข้อมูลโดยรวม โดยซ่อนรายละเอียดของโครงสร้างการเก็บข้อมูลภายใน เพื่อเชื่อมกับสิ่งที่ผู้ใช้มองเห็น รูปแบบข้อมูล ความสัมพันธ์ เงื่อนไขต่างๆ รวมถึงความปลอดภัย และความถูกต้องของข้อมูล แบบแผนนี้สามารถอธิบายได้โดยใช้โมเดลข้อมูลเชิงตรรก หรือโมเดลข้อมูลตามโครงสร้าง ที่ใช้ในการติดตั้งระบบ

อย่างไรก็ตามในความเป็นจริง ระบบจัดการฐานข้อมูลส่วนใหญ่ไม่ได้แบ่งสถาปัตยกรรมทั้ง 3 แบบแผนออกจากกันอย่างเด็ดขาด แต่สำหรับระบบจัดการฐานข้อมูลที่มีลักษณะตามแบบแผนของสถาปัตยกรรมนี้ การทำงานส่วนมากจะให้ผู้ใช้อ้างถึงเพียงระดับภายนอกเท่านั้น ระบบจัดการฐานข้อมูลจะเป็นตัวเปลี่ยนการร้องขอ หรือการระบุนั้น ๆ ให้เป็นระดับมโนภาพและระดับภายในตามลำดับ แล้วจึงทำการประมวลผลข้อมูลเพื่อให้ได้ผลลัพธ์ออกมา กระบวนการแปลงการร้องขอให้เป็นผลลัพธ์ระหว่างระดับนี้ เรียกว่าการทำแมปปิง (Mapping) กระบวนการนี้ต้องใช้เวลา ดังนั้นระบบจัดการฐานข้อมูลที่ใช้ได้กับระบบฐานข้อมูลเล็ก ๆ จะไม่สนับสนุนระดับ หรือแบบแผนฐานข้อมูลภายนอก

<u>ความเป็นอิสระของข้อมูล</u>

สถาปัตยกรรมทั้งสามแบบแผนดังกล่าว สามารถใช้อธิบายแนวคิดเกี่ยวกับความเป็นอิสระของข้อมูล (Data Independence) ซึ่งหมายถึงความสามารถในการเปลี่ยนแปลงแบบแผนฐานข้อมูลที่ระดับใดก็ได้ โดยไม่มีผลกระทบกับระดับเหนือขึ้นไปแต่อย่างใด เราสามารถแบ่งความเป็นอิสระของข้อมูล ออกเป็น 2 ชนิด คือ

1. ความเป็นอิสระทางตรรก (Logical data independence) เป็นความสามารถที่จะเปลี่ยนแบบแผนฐานข้อมูลเชิงมโนภาพได้ โดยไม่ต้องเปลี่ยนแบบแผนฐานข้อมูลภายนอก หรือโปรแกรมงานประยุกต์แต่อย่างใด เช่นเราอาจต้องการขยาย หรือลดขนาดฐานข้อมูล โดยขยายหรือลดชนิดของระเบียนเก่าที่มีอยู่ ซึ่งในกรณีลดขนาดฐานข้อมูลเท่านั้น ที่จะส่งผลกระทบต่อแบบแผนฐานข้อมูลภายนอกที่มีการอ้างถึงข้อมูลดังกล่าวเท่านั้น

2. ความเป็นอิสระทางกายภาพ (Physical data independence) หมายถึงความสามารถในการเปลี่ยนแบบแผนฐานข้อมูลภายในได้ โดยไม่จำเป็นต้องทำการเปลี่ยนแปลงแบบแผนฐานข้อมูลเชิงมโนภาพ หรือแบบแผนฐานข้อมูลภายนอก บางครั้งเราอาจมีการเปลี่ยนแบบแผนฐานข้อมูลภายใน เพราะต้องการปรับโครงสร้างแฟ้มข้อมูลทางกายภาพและประสิทธิภาพการทำงานใหม่ เช่น ต้องการเปลี่ยนลำดับที่ของข้อมูลที่เก็บอยู่จริงหรือเปลี่ยนวิธีการเข้าถึงข้อมูล

(access method) ซึ่งหลังจากเปลี่ยนแล้วหากข้อมูลเดิมยังคงอยู่ ก็ไม่จำเป็นต้องเปลี่ยนแบบแผนฐานข้อมูลเชิงมโนภาพใหม่

การออกแบบฐานข้อมูล

ประสิทธิภาพของการดำเนินการข้อมูล นำไปสู่การออกแบบโครงสร้างข้อมูลที่สลับซับซ้อน เพื่อเป็นตัวแทนของข้อมูลในฐานข้อมูล การออกแบบฐานข้อมูลที่ดี จะต้องครอบคลุมสารสนเทศทั้งในปัจจุบัน และในอนาคต นอกจากนี้ยังต้องออกแบบให้เป็นอิสระจากการติดตั้งทางกายภาพอีกด้วย เพื่อว่าหากมีการเปลี่ยนแปลงความต้องการข้อมูล เปลี่ยนสภาพแวดล้อมทางกายภาพ เช่น เปลี่ยนฮาร์ดแวร์ หรือเปลี่ยนระบบจัดการฐานข้อมูล ก็ไม่สามารถส่งผลกระทบต่อโมเดลข้อมูลเชิงตรรกแต่อย่างใด

กระบวนการในการออกแบบฐานข้อมูลทั้งทางตรรก และทางกายภาพ สามารถแบ่งออกได้ 2 ขั้นตอน คือ การออกแบบเนื้อหาและโครงสร้างฐานข้อมูล กับ การออกแบบในเรื่องการประมวลผลฐานข้อมูล และซอฟต์แวร์งานประยุกต์ โดยการออกแบบทั้ง 2 ขั้นตอนนี้ มักจะต้องทำไปด้วยกัน เพราะบางครั้งเราต้องอาศัยผลจากการศึกษางานประยุกต์ จึงจะสามารถกำหนดเนื้อหาของข้อมูลได้ หรือการกำหนดงานประยุกต์ ต้องอาศัยโครงสร้างของฐานข้อมูล

ขั้นตอนการออกแบบฐานข้อมูลสามารถแบ่งออกได้หลายขั้นตอน ดังนี้

1. การเก็บรวบรวมและวิเคราะห์ข้อมูล (Requirements collection and analysis) การเก็บรวบรวมและวิเคราะห์ข้อมูลนี้ จะต้องพิจารณาทั้งข้อมูลที่มีใช้อยู่ในปัจจุบันและข้อมูลที่ต้องการใช้เพิ่มเติม รวมถึงข้อมูลที่จะเกิดขึ้นได้ในอนาคต โดยผู้ออกแบบควรศึกษา และตรวจสอบโปรแกรมประยุกต์ รวมถึงรายงาน แบบฟอร์ม หรือผังงานต่าง ๆ อย่างละเอียด วิเคราะห์สิ่งแวดล้อมที่ต้องจัดการ รูปแบบการประมวลผล แผนงานทั้งในปัจจุบันและอนาคต รวมถึงชนิด และความถี่ของรายการเปลี่ยนแปลง ทิศทางของข่าวสารที่เข้า และออกจากระบบ เพื่อให้ทราบถึงความถี่และปริมาณของข้อมูลในฐานข้อมูลที่จะถูกเรียกใช้หรือเปลี่ยนแปลง นอกจากนี้ยังต้องสอบถามและสัมภาษณ์บุคลากรทุกระดับ เพื่อให้ทราบถึงระบบงานเดิม และระบบงานใหม่ที่อยากได้ ผลลัพธ์ที่ได้ในขั้นนี้ คือ โมเดลของสภาพแวดล้อม และความต้องการของผู้ใช้

2. การออกแบบฐานข้อมูลเชิงมโนภาพ (Conceptual database design) ผู้ออกแบบสามารถพัฒนารายละเอียดของข้อมูล และผลิตแบบแผนฐานข้อมูลเชิงมโนภาพ โดยใช้โมเดลข้อมูล ที่เป็นอิสระจากระบบจัดการฐานข้อมูล เช่น โมเดลข้อมูลเชิงตรรก (logical

model) โดยมีการกำหนดเอนติตี (entity) รีเลชันชิป (relationship) และลักษณะประจำ (attribute) นอกจากนี้ยังต้องทำการออกแบบรายการเปลี่ยนแปลง (transaction design) ชนิดของการเข้าถึง ปริมาณของรายการเปลี่ยนแปลงและข้อมูล ความถี่ในการเข้าถึง

3. การเลือกระบบจัดการฐานข้อมูล (Choice of the DBMS) ควรเลือกระบบจัดการฐานข้อมูลที่เข้าได้กับฮาร์ดแวร์ (Hardware) และซอฟแวร์ (Software) ที่มี พิจารณาโครงสร้างในการเก็บข้อมูล ทิศทางการเข้าถึงข้อมูลที่ฐานข้อมูลชนิดนั้น ๆ ทำได้ ภาษาระดับสูงที่ใช้ได้ และค่าใช้จ่ายที่เกิดขึ้นจากการเลือกระบบจัดการฐานข้อมูลดังกล่าว

4. การแปลงให้อยู่ในโมเดลของระบบจัดการฐานข้อมูล (Data model mapping or logical database design) เป็นการสร้างแบบแผนฐานข้อมูลเชิงมโนภาพในรูปโมเดลของระบบจัดการฐานข้อมูลที่เลือกแล้ว โมเดลข้อมูลชนิดนี้สามารถแบ่งออกเป็น 3 ชนิดดังนี้ โมเดลข้อมูลแบบรีเลชันนัล (Relational Data Model) โมเดลข้อมูลแบบเครือข่าย (Network Data Model) และโมเดลข้อมูลแบบลำดับขั้น (Hierarchical Data Model)

5. การออกแบบฐานข้อมูลทางกายภาพ (Physical database design) เป็นการกำหนดโครงสร้างการเก็บข้อมูล และทิศทางการเข้าถึงข้อมูล เพื่อให้ได้มาซึ่งฐานข้อมูลที่มีประสิทธิภาพมากที่สุด ระบบจัดการฐานข้อมูลต่าง ๆ จะมีระบบจัดการแฟ้มข้อมูล ทิศทางการเข้าถึงข้อมูล ดัชนีตัวชี้ และอื่น ๆ ที่ต่างกันออกไป สิ่งที่เราจะพิจารณาเพื่อการออกแบบทางกายภาพ คือ

5.1 เวลาในการตอบสนอง (response time) คือช่วงเวลาตั้งแต่การส่งงานเข้าไปถึงเมื่อได้รับผลลัพธ์ที่ต้องการออกมา

5.2 การใช้ที่ว่าง (Space utilization) จำนวนที่ว่างของหน่วยเก็บข้อมูลที่จะถูกใช้โดยแฟ้มข้อมูลของฐานข้อมูล และโครงสร้างทิศทางการเข้าถึงข้อมูล

5.3 จำนวนงานที่ได้ (transaction throughput) จะคิดเป็นค่าเฉลี่ย โดยคำนวณจากจำนวนงานที่ระบบจัดการฐานข้อมูลสามารถประมวลผลได้ต่อหนึ่งหน่วยเวลา

6. การติดตั้งและนำฐานข้อมูลมาใช้จริง (Database implementation)

ความปลอดภัยของฐานข้อมูล (Database Security)

การรักษาความปลอดภัยของฐานข้อมูล สามารถทำได้ทั้งทางกายภาพและทางซอฟแวร์ ดังนี้

1. ทางกายภาพ ได้แก่การติดตั้งระบบรักษาความปลอดภัยต่างๆ เช่น การมียามรักษาการณ์ มีกุญแจล็อคเครื่อง ให้เซ็นชื่อหรือพิมพ์ลายนิ้วมือ แม้ว่าสิ่งเหล่านี้จะไม่อยู่ในขอบเขตของผู้ออกแบบ แต่ผู้ออกแบบก็ควรได้แนะนำไว้

2. ทางซอฟแวร์ สามารถทำได้หลายวิธี เช่น

2.1 การให้อำนาจ (Authorization) เป็นการตรวจสอบหลักฐานผู้ใช้ว่าเป็นตัวจริงที่มีสิทธิมาร้องขอข้อมูลหรือไม่ มักจะติดตั้งการรักษาความปลอดภัยไว้ที่ระดับระบบปฏิบัติการ วิธีที่นิยมมากที่สุดคือให้ใส่รหัสผ่าน (password) แต่ถึงอย่างไรวิธีนี้ก็ไม่ปลอดภัยมากนัก บางแห่งอาจมีการเพิ่มเติมให้ใช้บัตร หรือกุญแจเฉพาะ เมื่อจะเข้าสู่ระบบ (login) ในระบบจัดการฐานข้อมูลส่วนมากจะมีระบบการรักษาความปลอดภัยนี้ให้สำหรับผู้ใช้แต่ละคน เพื่อกำหนดผู้มีสิทธิใช้ข้อมูลของตนได้เองเรียกว่า ภาษาในการให้สิทธิ (authorization language) เช่น ใน ORACLE มีคำสั่ง grant มีรูปแบบคือ

GRANT { privilege,privilege,...|ALL} ON table

TO {user,user,...|PUBLIC} [WITH GRANT OPTION];

ซึ่งเป็นการให้สิทธิในการกระทำกับตารางนั้นกับผู้ใช้อื่นๆ

2.2 การควบคุมการเข้าถึง (Access Control) เป็นการทำให้แน่ใจว่าข้อมูลหรือสิ่งต่างๆในระบบจะถูกเข้าถึงได้ เมื่อผ่านการตรวจสอบการให้สิทธิแล้ว วิธีนี้จะทำได้เมื่อมีการติดตั้งการให้สิทธิแล้วเท่านั้น เราอาจทำโดยสร้างตารางแมททริกซ์ของการเข้าถึงข้อมูล (access control matrix) โดยแต่ละคอลัมภ์แทนเป้าหมายในระบบ (objects) เช่นชื่อตารางหรือวิว ส่วนแถวต่าง ๆ แทนรหัสประจำตัว หรือกลุ่มของผู้ใช้ และค่าในแต่ละช่องที่สัมพันธ์กัน ก็คือสิทธิในการเข้าถึงที่กำหนดให้ผู้ใช้สำหรับตาราง หรือวิวนั้น ๆ ได้แก่ สิทธิการลบ เพิ่ม เปลี่ยนแปลง อ่านข้อมูล เป็นต้น

2.3 วิว (Views) การสร้างวิวในระบบจัดการฐานข้อมูลโดยทั่วไปจะมีคำสั่งที่ใช้ในการสร้างวิวจากตารางที่มีในระบบ เพื่อให้ได้ข้อมูลเฉพาะในส่วนที่ผู้ใช้จะมีสิทธิ โดยผู้จัดการฐานข้อมูลจะเป็นคนทำ และให้สิทธิในการเข้าถึงวิวนั้น ๆ ให้แก่ผู้ใช้

2.4 การเข้ารหัส (Data Encryption) [21] เป็นการนำข้อมูลปกติที่สามารถอ่านได้มาแปลงให้เป็นข้อมูลที่ไม่สามารถอ่านได้ โดยผ่านกระบวนการเข้ารหัส พร้อมกับกำหนดคีย์สำหรับการเข้ารหัส ผลลัพธ์ที่ได้จะถูกเก็บอยู่ในฐานข้อมูล เราเรียกผลลัพธ์นี้ว่า ciper text เมื่อข้อมูลถูกร้องขอ และผ่านกระบวนการตรวจสอบสิทธิในการเข้าถึงข้อมูลแล้ว ข้อมูลดังกล่าวจะถูกแปลงกลับมาให้อยู่ในรูปที่สามารถอ่านได้อีกครั้ง กระบวนการเข้ารหัสและการถอดรหัส ซึ่งเป็นที่นิยม และเป็นมาตรฐาน คือ Data Encription Standard (DESC) ซึ่งสามารถสร้างผลลัพธ์ได้แตกต่างกันกว่า 2^{56} รูปแบบ

2.5 การลงบันทึกความปลอดภัย (Security logs) เป็นการบันทึกเกี่ยวกับผู้ที่ตั้งใจจะฝ่าฝืนเข้าสู่ระบบ โดยจะบันทึกข้อมูลเหล่านั้นไว้ในแฟ้มลงบันทึก หรือมีการส่งข่าวไปให้ผู้ควบคุมเครื่องหรือผู้บริหารฐานข้อมูลทราบ

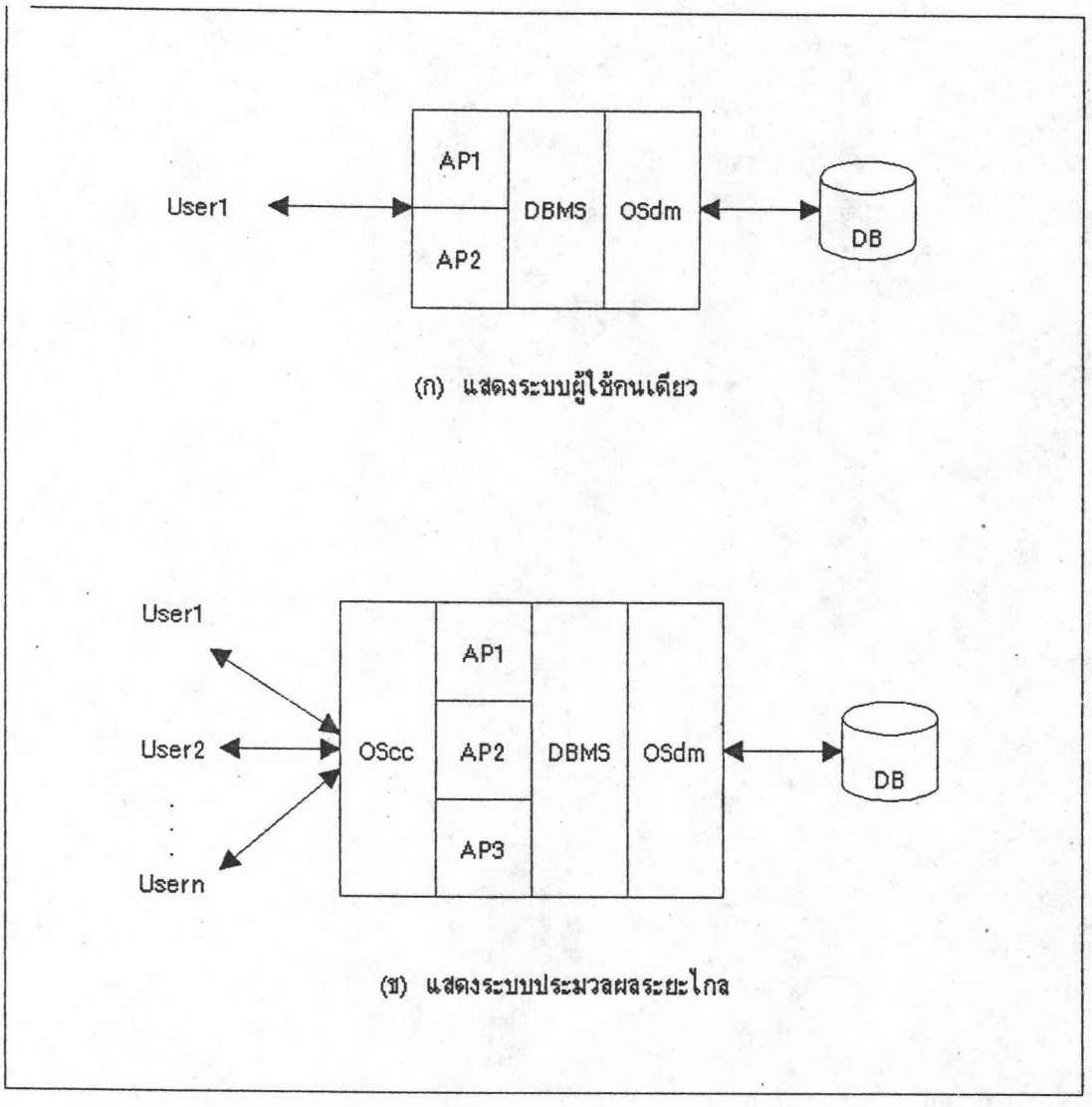
2.6 การทำหลักฐานการตรวจสอบ (Audit Trail) เป็นการเก็บข่าวสารเกี่ยวกับเรื่องใครมาดึงข้อมูล มาใช้เครื่องเทอร์มินอล และใช้เมื่อเวลาใด

ประเภทของระบบฐานข้อมูล

เราสามารถแบ่งระบบฐานข้อมูลตามลักษณะการเก็บข้อมูลได้เป็น 2 ประเภท ดังนี้

1. ระบบฐานข้อมูลแบบรวมศูนย์ (Centralized database system) [18]

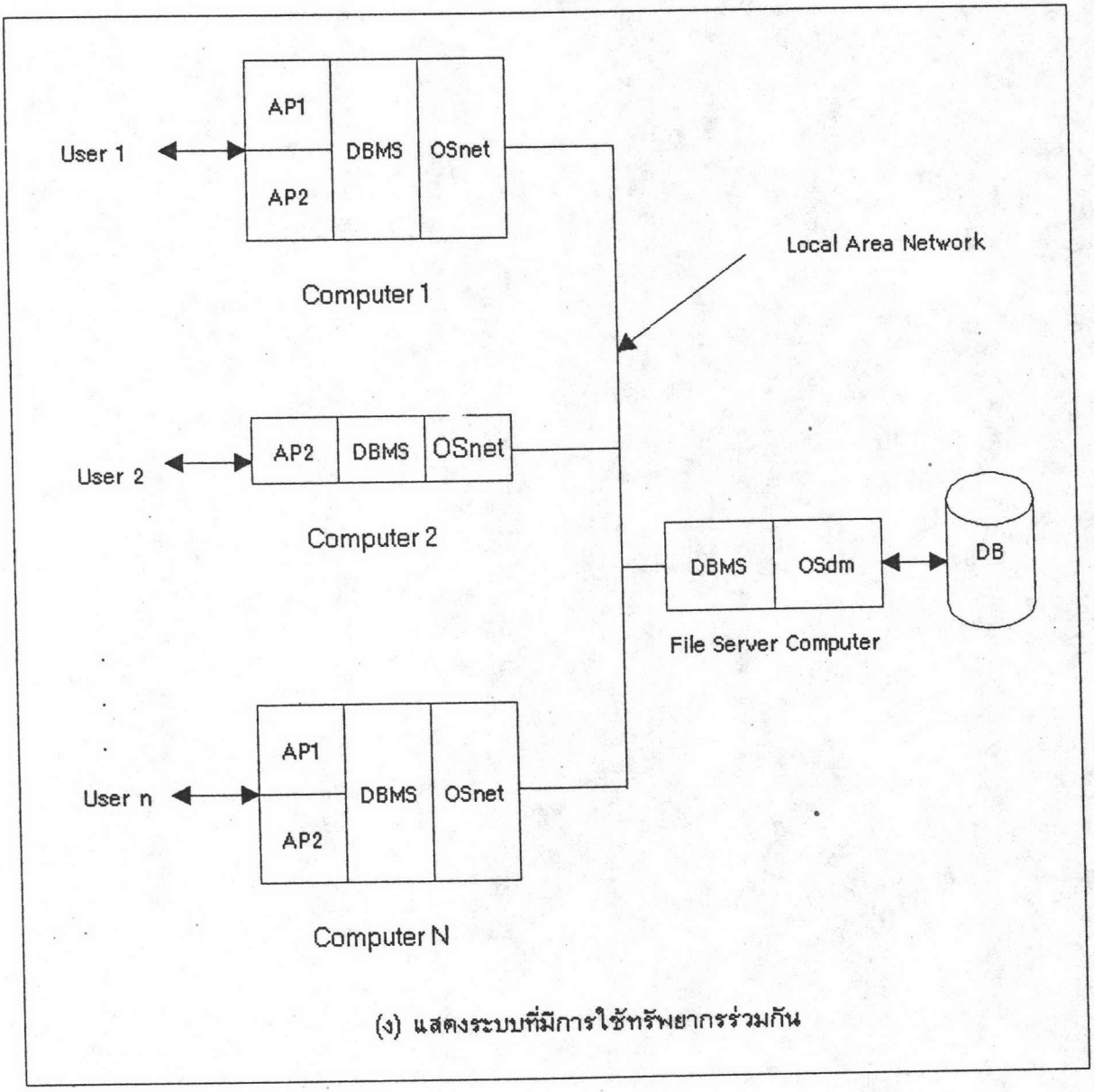
ระบบฐานข้อมูลแบบรวมศูนย์ หมายถึง ระบบฐานข้อมูลหนึ่ง ซึ่งประกอบด้วยข้อมูลและระบบจัดการฐานข้อมูล อยู่บนเครื่องฮาร์ดแวร์เดียว (Single hardware machine) หรือสถานีเดียว (single site) แม้ว่าอาจมีการเข้าถึงข้อมูลแบบระยะไกล (remote access) ก็ตาม ระบบฐานข้อมูลนี้ก็ยังเป็นระบบฐานข้อมูลแบบรวมศูนย์อยู่ ทั้งนี้เพราะส่วนประกอบหลักของระบบทั้งหมด (ข้อมูล และระบบจัดการฐานข้อมูล) และการประมวลผลที่เกี่ยวข้องกับฐานข้อมูลยังคงอยู่ที่คอมพิวเตอร์เครื่องเดียว ตัวอย่างระบบฐานข้อมูลแบบรวมศูนย์ คือ ระบบผู้ใช้คนเดียว (Single user system) ระบบประมวลผลระยะไกล (Teleprocessing system) ระบบผู้ใช้-ผู้ให้บริการ (Client-server system) ที่มีผู้ให้บริการเดียว และมีฐานข้อมูลเดียว หรือระบบที่มีผู้ให้บริการ และมีฐานข้อมูลมากกว่าหนึ่งตัว โดยไม่มีผู้ให้บริการตัวใดใช้ฐานข้อมูลร่วมกัน หรือ ระบบที่ใช้ทรัพยากรร่วมกัน (Resource sharing system) ดังปรากฏในรูป 3.2



หมายเหตุ : OScc = Communication Controlportion of operation system

OSdm = Data management portion of operating system

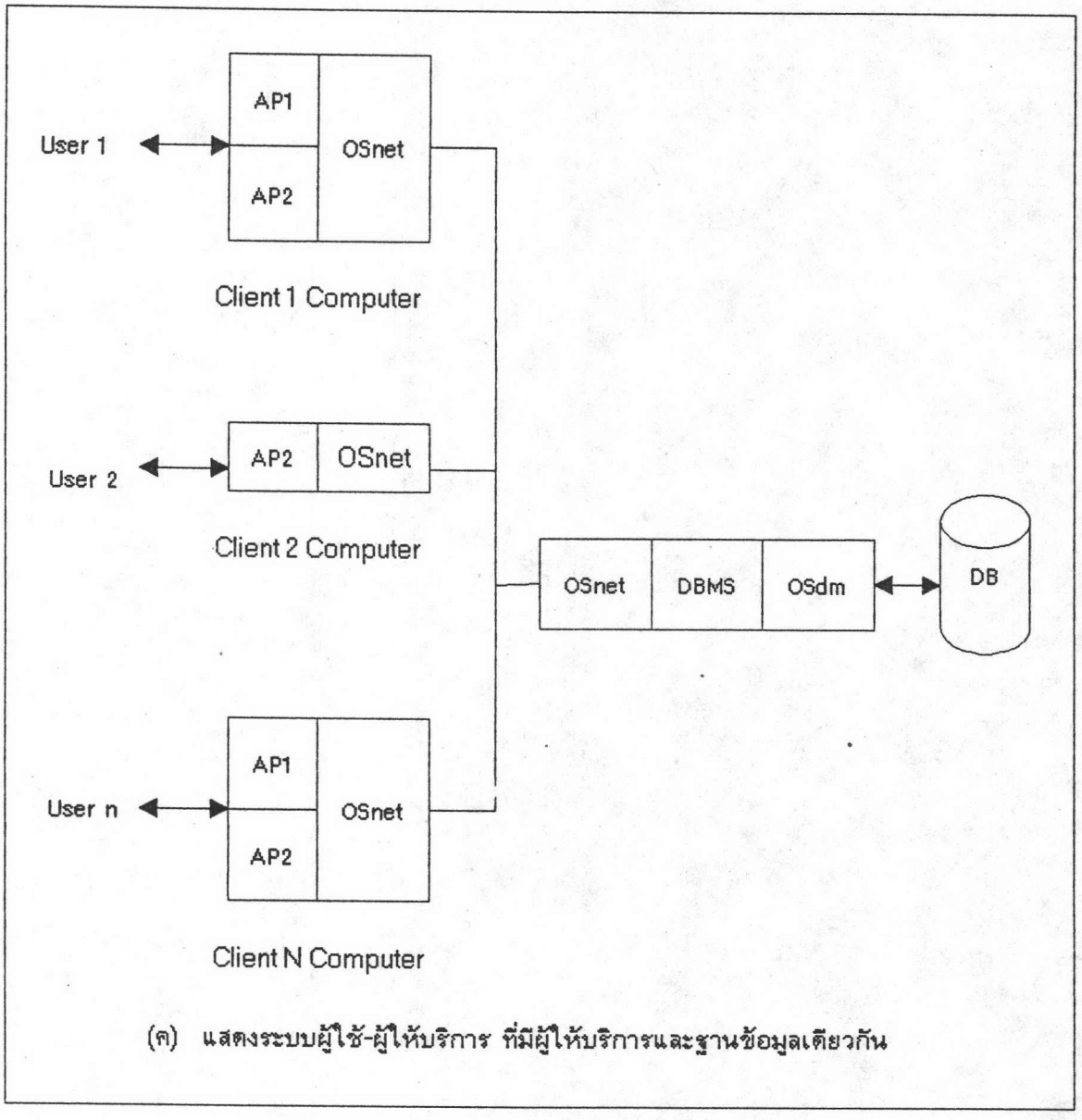
รูปที่ 3.2 แสดงสถาปัตยกรรมต่าง ๆ ที่จัดว่าเป็นระบบฐานข้อมูลแบบรวมศูนย์



หมายเหตุ : OSnet = Network communication portion of operating system

OSdm = Data management portion of operating system

รูปที่ 3.2 (ต่อ) แสดงสถาปัตยกรรมต่าง ๆ ที่จัดว่าเป็นระบบฐานข้อมูลแบบรวมศูนย์



(ค) แสดงระบบผู้ใช้-ผู้ให้บริการ ที่มีผู้ให้บริการและฐานข้อมูลเดียวกัน

หมายเหตุ : OSnet = Network communication portion of operating system

OSdm = Data management portion of operating system

รูปที่ 3.2 (ต่อ) แสดงสถาปัตยกรรมต่าง ๆ ที่จัดว่าเป็นระบบฐานข้อมูลแบบรวมศูนย์

2. ระบบฐานข้อมูลแบบกระจาย (Distributed database system)

โดยทั่วไปแล้ว ฐานข้อมูลแบบกระจาย จะหมายถึง [12,13,20] ข้อมูลของระบบใดระบบหนึ่ง ซึ่งกระจายอยู่ตามสถานี (sites หรือ nodes) ต่างๆ ภายในระบบเครือข่ายคอมพิวเตอร์ โดยผู้ใช้จะอยู่ที่สถานีใดๆก็ได้ในเครือข่าย และสามารถเรียกใช้ข้อมูลซึ่งเก็บอยู่ตามสถานีต่างๆได้ เสมือนว่าข้อมูลดังกล่าวเก็บอยู่ที่เดียวกัน นอกจากนี้แต่ละสถานียังมีความสามารถในการดำเนินงาน และควบคุมระบบงานของตนได้อย่างอิสระอีกด้วย

2.1 ส่วนประกอบของระบบฐานข้อมูลแบบกระจาย

ระบบฐานข้อมูลแบบกระจายสามารถแบ่งออกเป็นสองส่วน คือ ส่วนของฮาร์ดแวร์ และส่วนของซอฟแวร์ จากรูป 3.3 สามารถอธิบายได้ดังนี้ [16]

2.1.1 ในระดับฮาร์ดแวร์ ปัจจัยหลักที่เป็นตัวแยกความแตกต่างของระบบฐานข้อมูลแบบกระจาย ออกจากฐานข้อมูลแบบรวมศูนย์ มีดังต่อไปนี้

1. ระบบฐานข้อมูลแบบกระจาย ระบบนี้จะประกอบไปด้วยเครื่องคอมพิวเตอร์มากกว่าหนึ่งเครื่อง หรือมากกว่าหนึ่งสถานี (sites or notes)

2. สถานีต่าง ๆ เหล่านี้จะถูกเชื่อมต่อเข้าด้วยกัน โดยใช้เครือข่ายท้องถิ่น (local area network - LAN) หรือเครือข่ายระยะไกล (wide area network - WAN) เพื่อให้สามารถส่งข้อมูลและคำสั่งระหว่างสถานีได้

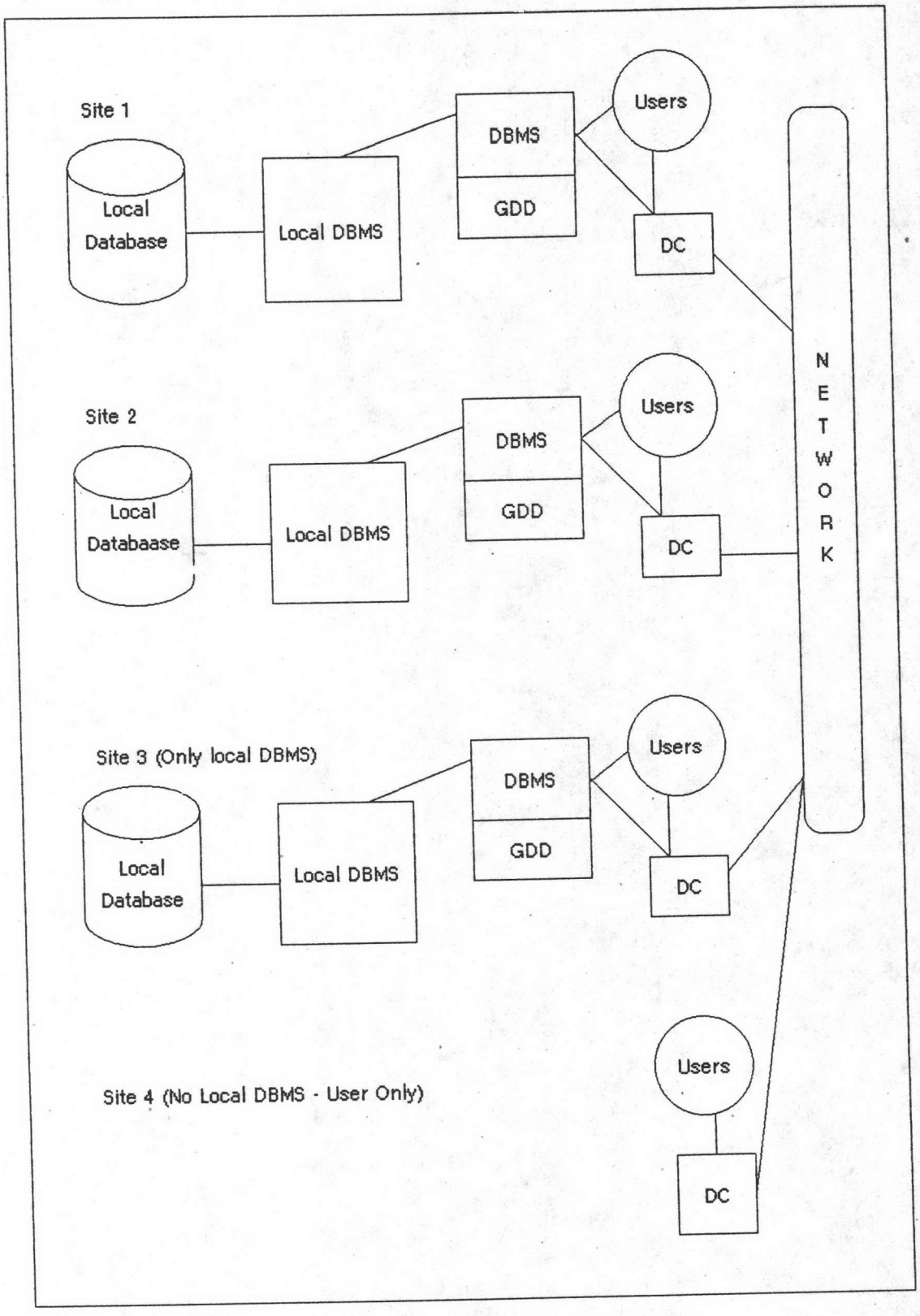
2.1.2 ส่วนประกอบที่เป็นซอฟแวร์ ซึ่งจำเป็นต่อการสร้างฐานข้อมูลแบบกระจาย มีดังต่อไปนี้ คือ

1. ส่วนของการสื่อสารข้อมูล (Data Communication Component - DC) หมายถึง ซอฟแวร์สำหรับจัดการเกี่ยวกับระบบสื่อสารพื้นฐานต่าง ๆ ที่ใช้ในการส่งข้อมูล และคำสั่งระหว่างสถานี นอกจากนี้ยังทำหน้าที่ในการเก็บรายละเอียดต่าง ๆ ทั้งหมดเกี่ยวกับสถานี และลิงค์ (link) ที่มีภายในระบบเครือข่ายอีกด้วย

2. ส่วนของตัวจัดการฐานข้อมูลท้องถิ่น (Local Database Management Component - DBMS) หมายถึง ระบบจัดการฐานข้อมูลมาตรฐานนั่นเอง ซึ่งเป็นตัวควบคุมฐานข้อมูลท้องถิ่นของแต่ละสถานี

3. ส่วนของพจนานุกรมข้อมูลส่วนรวม (Global Data Dictionary - GDD) เป็นที่เก็บสารสนเทศของฐานข้อมูล ที่กระจายอยู่ตามสถานีต่าง ๆ ว่าข้อมูลเหล่านั้นถูกเก็บอยู่ที่สถานีใดบ้าง พร้อมทั้งรายละเอียดอื่น ๆ ของข้อมูลที่มีอยู่ในระบบ

4. ส่วนของตัวจัดการฐานข้อมูลแบบกระจาย (Distributed Database Management Component - DDBMS) เป็นตัวจัดการเกี่ยวกับระบบทั้งหมด เช่น เตรียมการติดต่อกับผู้ใช้ (user interface) โดยทำให้ผู้ใช้มองเห็นราวกับว่าข้อมูลทั้งหมด ถูก



รูปที่ 3.3 แสดงส่วนประกอบของระบบฐานข้อมูลแบบกระจาย

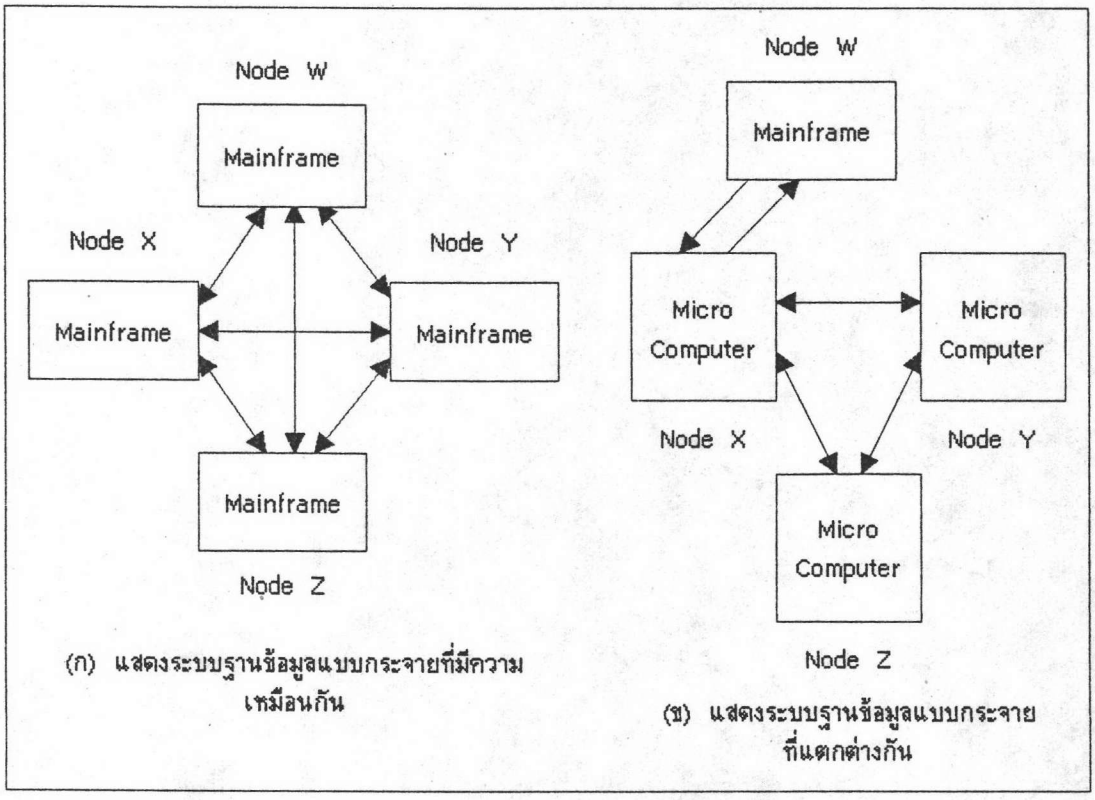
เก็บอยู่ที่สถานีของตนเพียงแห่งเดียว ประมวลผลการสอบถามที่เกิดขึ้น นอกจากนี้ยังทำหน้าที่ในการควบคุมความถูกต้องตรงกันของข้อมูลชุดต่างๆ การกู้ข้อมูล (recovery) และแปลงคำสั่งและข้อมูลระหว่างสถานี ในกรณีที่ระบบฐานข้อมูลแบบกระจายเป็นชนิดที่ไม่มีความเหมือนกันอีกด้วย

2.2 ประเภทของระบบฐานข้อมูลแบบกระจาย [12,14,16,20]

ระบบฐานข้อมูลแบบกระจายอาจแบ่งอย่างกว้างๆ ได้สองประเภท คือ

2.2.1 ระบบที่มีความเหมือนกัน (homogeneous) หมายถึงระบบที่ทุกสถานีมีระบบจัดการฐานข้อมูลท้องถิ่นและมีฮาร์ดแวร์เหมือนกัน ตัวอย่างดังแสดงในรูป3.4 (ก) ซึ่งเป็นระบบฐานข้อมูลแบบกระจายที่แต่ละสถานี ใช้ฮาร์ดแวร์เหมือนกัน คือเป็นคอมพิวเตอร์ระดับใหญ่ (Mainframe) ทั้ง 4 สถานี และใช้ตัวจัดการฐานข้อมูลท้องถิ่น R* ของบริษัท IBM เหมือนกัน

2.2.2 ระบบที่แตกต่างกัน (heterogeneous) หากมีสถานีใดใช้ระบบจัดการฐานข้อมูลท้องถิ่น และหรือใช้ฮาร์ดแวร์ต่างไปจากสถานีอื่นแล้ว จะถือว่าระบบนี้เป็นระบบฐานข้อมูลแบบกระจายที่แตกต่างกัน ในกรณีเช่นนี้ระบบจัดการฐานข้อมูลแบบกระจาย จะต้องทำหน้าที่ในการแปลงรูปแบบข้อมูล และหรือคำสั่งจากสถานีหนึ่งไปยังอีกสถานีหนึ่ง จากรูป 3.4(ข)



รูปที่ 3.4 แสดงชนิดของระบบฐานข้อมูลแบบกระจาย

สมมุติว่าระบบฐานข้อมูลแบบกระจายประกอบด้วย สถานีซึ่งเป็นระบบคอมพิวเตอร์เครื่องใหญ่ ต่อกับระบบคอมพิวเตอร์เครื่องเล็ก ผ่านระบบเครือข่าย กรณีเช่นนี้ผู้ออกแบบฐานข้อมูล ควรเลือกใช้ระบบจัดการฐานข้อมูลที่สามารถใช้ได้กับทุกฮาร์ดแวร์ เช่น อาจเลือกใช้ระบบจัดการฐานข้อมูลของออราคัล เป็นต้น

2.3 สถาปัตยกรรมของระบบฐานข้อมูลแบบกระจาย (Distributed database architecture) [12,20]

สถาปัตยกรรมของระบบฐานข้อมูลแบบกระจาย แบ่งการจัดองค์กรข้อมูล (data organization) ออกเป็นประเภทต่างๆ ดังแสดงอยู่ในรูปที่ 3.5 คือ

2.3.1 ฐานข้อมูลแบบภายนอก (external schema) คือส่วนของข้อมูลที่ผู้ใช้ หรือแอพพลิเคชั่นมองเห็น

2.3.2 ฐานข้อมูลเชิงมโนภาพในระดับรวม (global conceptual schema) คือส่วนที่เก็บรายละเอียดของข้อมูลทั้งหมดในฐานข้อมูลแบบกระจาย ราวกับว่าข้อมูลนั้นถูกเก็บรวมศูนย์อยู่ที่เดียวกัน

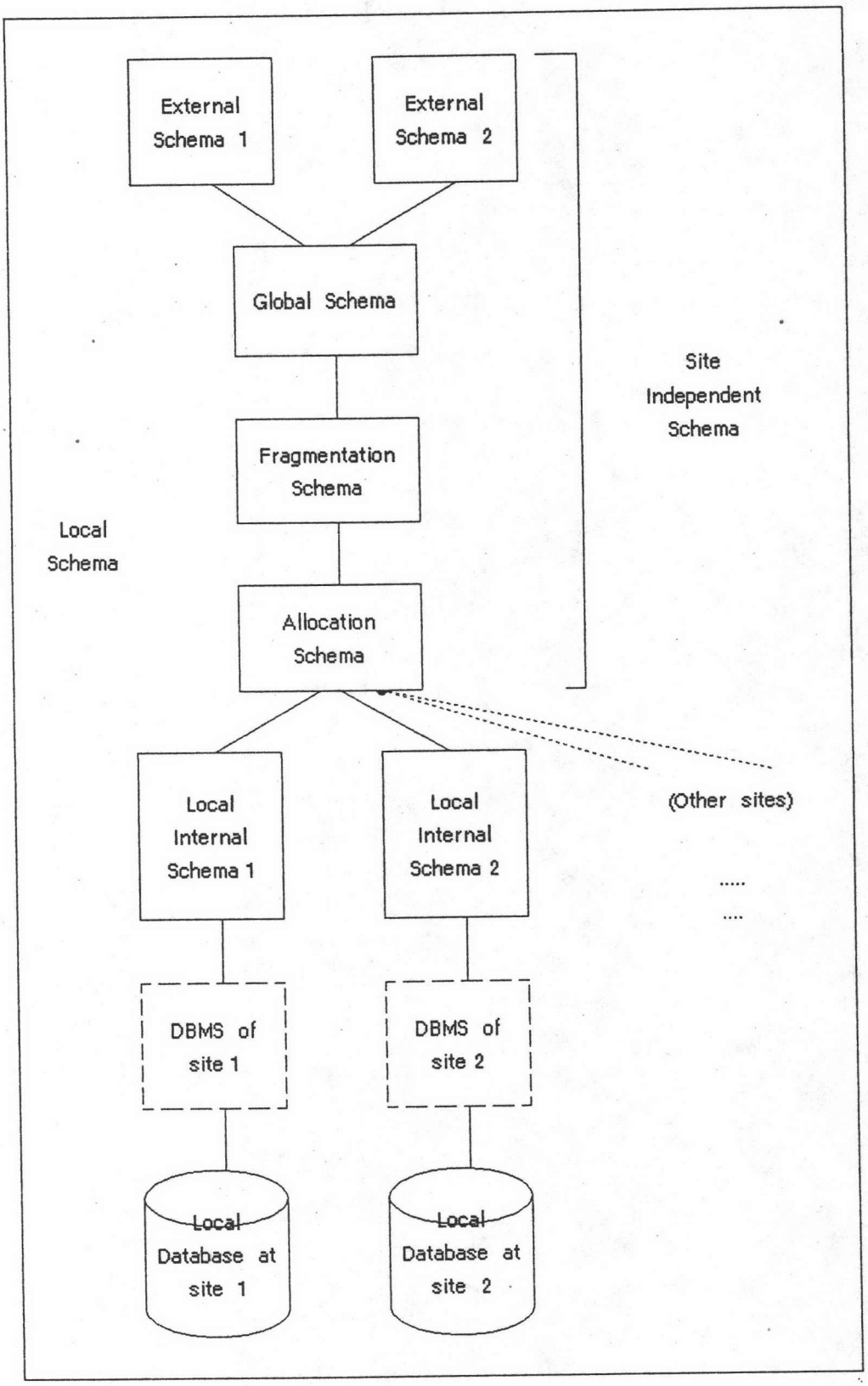
2.3.3 ฐานข้อมูลเชิงมโนภาพในระดับท้องถิ่น (local conceptual schema) จะเป็นตัวอธิบายลักษณะข้อมูลทางตรรกภายในระดับท้องถิ่นว่า ข้อมูลถูกแบ่งและถูกกำหนดสถานที่เก็บอย่างไร ฐานข้อมูลระดับนี้ จะประกอบด้วย 2 ส่วน คือ

1. ฐานข้อมูลแบบแตกกระจาย (fragmentation schema) เนื่องจากฐานข้อมูลเชิงมโนภาพในระดับรวมสามารถถูกแบ่งออกเป็นส่วนๆ โดยแต่ละส่วนจะไม่คาบเกี่ยวกัน ส่วนต่างๆเหล่านี้จะถูกกระจายไปตามสถานีต่างๆ ในเครือข่าย ดังนั้นในระดับนี้จึงเป็นการแมป (map) จากฐานข้อมูลเชิงมโนภาพในระดับรวม ให้เป็นส่วนเล็ก ๆ หลาย ๆ ส่วนนั่นเอง

2. ฐานข้อมูลแบบตามสถานที่จัดเก็บ (allocation schema) ระดับนี้จะเป็นการเก็บรายละเอียดว่า ส่วนของฐานข้อมูลต่าง ๆ นั้นถูกเก็บอยู่ที่สถานีใดบ้าง และยังสามารถบอกได้ว่าข้อมูลซึ่งถูกกระจายอยู่นั้น เป็นแบบซ้ำซ้อน (redundant) หรือไม่

2.3.4 ฐานข้อมูลแบบภายในของระดับท้องถิ่น (local internal schema) ระดับนี้เป็นการแปลงส่วนต่างๆของฐานข้อมูล (fragmented database) ซึ่งถูกกำหนดสถานที่ในการจัดเก็บแล้ว ให้เป็นสิ่งซึ่งระบบจัดการข้อมูลท้องถิ่นรู้จัก

ฐานข้อมูลเชิงมโนภาพในระดับรวม และฐานข้อมูลเชิงมโนภาพในระดับท้องถิ่น จะไม่ขึ้นอยู่กับโมเดลข้อมูลที่แต่ละสถานีเลือกใช้ ต่างจากฐานข้อมูลแบบภายในของระดับท้องถิ่น ซึ่งจะขึ้นอยู่กับระบบจัดการฐานข้อมูลท้องถิ่นที่ใช้



รูปที่ 3.5 แสดงสถาปัตยกรรมของระบบฐานข้อมูลแบบกระจาย

2.4 การออกแบบฐานข้อมูลแบบกระจาย (Distributed database design)

การออกแบบฐานข้อมูลแบบกระจายเป็นงานที่ยากมาก เพราะมีปัญหาทั้งทางเทคนิคและองค์กร ปัญหาด้านเทคนิคที่เกิดขึ้น เช่น การเชื่อมต่อระหว่างสถานีในเครือข่าย และปัญหาเกี่ยวกับการกระจายข้อมูล และงานประยุกต์ที่เหมาะสม เพื่อให้เกิดประสิทธิภาพสูงสุด ส่วนปัญหาด้านองค์กรนั้น พบว่าการกระจายระบบ โดยเฉพาะด้านงานประยุกต์มักส่งผลกระทบต่อองค์กรเสมอ [12,20]

2.4.1 กลยุทธ์หลักที่ใช้ในการออกแบบฐานข้อมูลแบบกระจาย มีสองแนวทาง ดังนี้

1. การออกแบบจากบนลงล่าง (Top down approach) วิธีนี้เริ่มจากการออกแบบฐานข้อมูลเชิงมโนภาพในระดับรวม แล้วจึงออกแบบลักษณะการกระจายของข้อมูลและรูปแบบการเก็บข้อมูล เพื่อเป็นการกำหนดฐานข้อมูลเชิงมโนภาพระดับท้องถิ่น ต่อมาจึงทำการออกแบบฐานข้อมูลเชิงกายภาพ (physical database design) ซึ่งเป็นการแมปจากฐานข้อมูลเชิงมโนภาพระดับท้องถิ่นที่ได้ ไปยังอุปกรณ์ที่ใช้เก็บข้อมูลจริง วิธีนี้จะเหมาะกับการสร้างระบบฐานข้อมูลแบบกระจายขึ้นใหม่ โดยที่ยังไม่เคยมีระบบฐานข้อมูลใดๆ อยู่ก่อน

2. การออกแบบจากล่างขึ้นบน (Bottom-up approach) วิธีนี้เหมาะสำหรับการพัฒนาฐานข้อมูลแบบกระจาย ซึ่งต้องรวบรวมระบบฐานข้อมูลหลาย ๆ ระบบที่มีอยู่เข้าเป็นระบบเดียวกัน นั่นคือเป็นการรวบรวมฐานข้อมูลเชิงมโนภาพระดับท้องถิ่นของแต่ละระบบ ให้เป็นฐานข้อมูลเชิงมโนภาพในระดับรวมเดียวกัน ซึ่งต้องแก้ปัญหาเรื่องการขัดแย้งกันระหว่างรูปแบบฐานข้อมูลเดิมของแต่ละสถานี ขั้นตอนการออกแบบวิธีดังกล่าวสามารถสรุปได้ดังนี้

2.1 เลือกโมเดลข้อมูลที่เป็นกลาง มาอธิบายแบบแผนฐานข้อมูลในระดับรวม โดยเป็นโมเดลที่สามารถอธิบายข้อมูลเดิมของทุกสถานีได้

2.2 แปลงแบบแผนฐานข้อมูลระดับท้องถิ่นของแต่ละสถานีให้อยู่ในโมเดลข้อมูลที่ได้เลือกไว้

2.3 รวบรวมแบบแผนฐานข้อมูลท้องถิ่น ขึ้นเป็นแบบแผนฐานข้อมูลในระดับรวม

2.4.2 ขั้นตอนการออกแบบฐานข้อมูลแบบกระจาย

การออกแบบฐานข้อมูลแบบกระจายประกอบด้วยขั้นตอนต่าง ๆ ดังนี้

1. การออกแบบฐานข้อมูลเชิงมโนภาพในระดับรวม เป็นการ

อธิบายข้อมูลของระบบทั้งหมด

2. การออกแบบฐานข้อมูลเชิงกายภาพระดับท้องถิ่น โดยกำหนดลักษณะการเก็บข้อมูลและวิธีการเข้าถึงข้อมูล

3 การออกแบบลักษณะการแบ่งข้อมูล โดยกำหนดว่าจะแบ่งรีเลชันในระดับรวมออกตามแนวตั้ง แนวนอน หรือแบบผสมดี

4. การออกแบบสถานที่เก็บข้อมูล เป็นการกำหนดว่าจะเก็บส่วนของฐานข้อมูลที่ถูกแบ่งแล้วนั้นอย่างไร

ขั้นตอน 1. และ 2. จะมีลักษณะเช่นเดียวกับการออกแบบฐานข้อมูลแบบรวมศูนย์ โดยเท่ากับเป็นการออกแบบโมเดลข้อมูลเชิงมโนภาพ และโมเดลข้อมูลเชิงกายภาพตามลำดับ ส่วนขั้นตอน 3. และ 4. จะเป็นขั้นตอนการออกแบบฐานข้อมูลแบบกระจายจริง ๆ นอกจากนี้ทั้งสองขั้นตอนดังกล่าวยังมีความสัมพันธ์กันด้วย นั่นคือการออกแบบลักษณะการแบ่งข้อมูล จะเป็นการกำหนดเงื่อนไขทางตรรกว่า จะแบ่งรีเลชันในระดับรวมที่มีอย่างไร ในขณะที่การออกแบบสถานที่เก็บข้อมูล จะเป็นการกำหนดสถานที่ทางกายภาพ ที่จะใช้เก็บส่วนต่าง ๆ ของข้อมูลที่ได้กำหนดเงื่อนไขในการแบ่งแล้ว

อย่างไรก็ตามการออกแบบฐานข้อมูลแบบกระจาย ยังต้องอาศัยความรู้ในเรื่องของความต้องการใช้งานประยุกต์ที่สำคัญ ๆ ซึ่งถูกเรียกใช้งานบ่อย และมีส่วนสำคัญต่อประสิทธิภาพของระบบงาน ความรู้เหล่านี้ประกอบด้วย

1. สถานีที่เป็นจุดกำเนิดของงานประยุกต์

2. ความบ่อยซึ่งงานประยุกต์ถูกเรียก หมายถึง จำนวนครั้งที่งานประยุกต์ถูกเรียกใช้ต่อหน่วยเวลา

3. จำนวนครั้ง ชนิด และสถิติของการเรียกใช้ข้อมูลที่กระจายอยู่ตามสถานีต่าง ๆ

2.4.2 การออกแบบโมเดลข้อมูล (data model)

การออกแบบโมเดลข้อมูลในระบบฐานข้อมูลแบบกระจาย มีลักษณะเช่นเดียวกับการออกแบบโมเดลข้อมูลระบบฐานข้อมูลแบบรวมศูนย์

2.5 การแตกกระจายฐานข้อมูล

หน่วยที่เหมาะสมสำหรับการกระจายข้อมูล ไม่ใช่รีเลชันทั้งหมด แต่เป็นส่วนใดส่วนหนึ่งของรีเลชันเท่านั้น ทั้งนี้เพราะเวลาที่งานประยุกต์มองข้อมูล จะมองเฉพาะส่วนใดส่วนหนึ่งของข้อมูลทั้งหมดเท่านั้น ดังนั้นหากเราจะกระจายข้อมูลรีเลชันทั้งหมดแล้ว จะเป็นการ

สิ้นเปลืองพื้นที่โดยใช่เหตุ

รีเลชันระดับรวมสามารถแตกกระจายออกเป็นรีเลชันย่อย ๆ ได้หลายส่วน ด้วยวิธีต่าง ๆ กัน รีเลชันที่ได้จากการแตกกระจายนี้ เรียกว่า แฟรกเมนต์ (Fragment)

2.5.1 กฎการแตกกระจายฐานข้อมูล [20]

การกำหนดวิธีการแตกกระจายฐานข้อมูล สามารถทำได้ภายใต้กฎ 3 ข้อ ดังนี้

1. เงื่อนไขของความครบถ้วน (Completeness condition) ข้อมูลทั้งหมดที่เป็นส่วนของรีเลชันระดับรวม จะต้องถูกแมปไปยังแฟรกเมนต์ต่าง ๆ ได้ครบ นั่นคือ ถ้ารีเลชัน R ถูกแตกกระจายออกเป็นแฟรกเมนต์ R_1, R_2,...,R_n แล้ว ข้อมูลที่ถูกพบในรีเลชัน R จะต้องถูกพบในแฟรกเมนต์ R_i อย่างน้อยหนึ่งแฟรกเมนต์

2. เงื่อนไขของการสร้างใหม่ (Reconstruction condition) รีเลชันระดับรวม สามารถสร้างได้ใหม่เสมอจากแฟรกเมนต์ต่าง ๆ เงื่อนไขนี้มีความสำคัญที่เห็นได้ชัด เพราะในความเป็นจริงแล้ว ฐานข้อมูลที่ถูกเก็บจริง ๆ จะเป็นส่วนของแฟรกเมนต์เท่านั้น ดังนั้นถ้าต้องการใช้รีเลชันระดับรวม ก็จำเป็นต้องสร้างใหม่จากแฟรกเมนต์ต่าง ๆ เหล่านี้ ดังนั้นถ้ารีเลชัน R ถูกแบ่งออกเป็นแฟรกเมนต์ R_1, R_2,...,R_n แล้ว เราสามารถสร้างรีเลชันระดับรวมได้ โดย

$$R = \nabla R_i, \quad \nabla R_i \in F_R$$

โดยที่ ∇ หมายถึงตัวกระทำ (operator) ซึ่งจะแตกต่างกันไปตามรูปแบบการแตกกระจายที่ใช้

F_R หมายถึงแฟรกเมนต์ ที่ถูกแตกกระจายจากรีเลชัน R

3. เงื่อนไขของการไม่คาบเกี่ยวกัน (Disjointness condition) นั่นคือส่วนต่าง ๆ ของแฟรกเมนต์ จะต้องไม่คาบเกี่ยวกัน เพื่อที่ว่าการทำข้อมูลซ้ำ (Replication of data) จะถูกควบคุมโดยขั้นตอนของการกำหนดสถานที่เท่านั้น (Allocation) โดยถ้ารีเลชัน R ถูกแตกกระจายตามแนวนอนออกเป็นแฟรกเมนต์ R_1,R_2,...,R_n ดังนั้น หาก d_i คือข้อมูลที่อยู่ใน R_j แล้ว d_i จะต้องไม่อยู่ในส่วนของแฟรกเมนต์ R_k โดยที่ $k \neq i$ เงื่อนไขนี้ ทำให้มั่นใจได้ว่าการแตกกระจายข้อมูลตามแนวนอนนี้ จะเป็นไปตามเงื่อนไขของการไม่คาบเกี่ยวกัน ส่วนถ้ารีเลชัน R ถูกแตกกระจายตามแนวตั้งแล้ว แอตตริบิวที่เป็นคีย์หลักจะถูกซ้ำอยู่ในทุก ๆ ส่วนของแฟรกเมนต์ ดังนั้นในกรณีของการแตกกระจายตามแนวตั้ง เงื่อนไขของการไม่คาบเกี่ยวกัน จะใช้กำหนดลักษณะการกระจายกับแอตตริบิวที่ไม่ใช่คีย์หลักเท่านั้น

2.5.2 รูปแบบของการแตกกระจาย [12,20]

ในที่นี้สมมุติว่าโมเดลข้อมูลที่ใช้เป็นรีเลชันนัลโมเดล ดังนั้น จึงสามารถแตกกระจายฐานข้อมูลออกเป็นส่วนๆ ตามลักษณะการใช้งานของท้องถิ่นได้ดังนี้ คือ

2.5.2.1 การแตกกระจายตามแนวนอน (Horizontal Fragmentation) เป็นการแบ่งทูเปิล (tuples) ของรีเลชันระดับรวมออกเป็นกลุ่มย่อย ๆ โดยแต่ละกลุ่มจะต้องมีคุณสมบัติร่วมกัน เราสามารถกำหนดการแตกกระจายตามวิธีนี้ได้ โดยการชีเลคชัน (selection operation) จากรีเลชันระดับรวม ดังตัวอย่างต่อไปนี้ จากตารางที่ 3.1 (ก) สมมุติรีเลชันระดับรวม STUDENT คือ

STUDENT(STUID,STUNAME,MAJOR,CREDIT)

ดังนั้นถ้าให้แฟรกเมนต์ $STUDENT_1$ เก็บเฉพาะข้อมูลของนักเรียน ที่มีวิชาเอกคือ คณิตศาสตร์ และให้แฟรกเมนต์ $STUDENT_2$ เก็บเฉพาะข้อมูลของนักเรียน ที่มีวิชาเอกคือ ศิลปะ แล้ว ข้อตกลงของการแตกกระจายตามแนวนอน สามารถกำหนดได้ ดังนี้

$$STUDENT_1 = SL_{MAJOR="Math"}STUDENT$$

$$STUDENT_2 = SL_{MAJOR="Art"}STUDENT$$

โดยที่ SL_FR หมายถึง การชีเลคชันจากรีเลชัน R โดยมีเงื่อนไขตาม F

การตรวจสอบความถูกต้อง สามารถทำได้ดังนี้

1. เงื่อนไขของความครบถ้วน ความครบถ้วนจะเป็นไปได้ ก็เมื่อข้อตกลงที่ใช้ในการกำหนดเงื่อนไขของการแตกกระจาย เป็นข้อตกลงที่สมบูรณ์ สามารถครอบคลุมลักษณะของรีเลชันระดับรวมได้ทั้งหมด จากตัวอย่างข้างต้น ข้อตกลงจะมีความครบถ้วนก็ต่อเมื่อ "Math" และ "Art" เป็นค่าที่เป็นไปได้ในแอตตริบิว MAJOR เท่านั้น

2. เงื่อนไขของการสร้างใหม่ เราสามารถสร้างรีเลชันระดับรวม STUDENT ได้ใหม่จากแฟรกเมนต์ต่าง ๆ ของมัน โดยใช้คำสั่ง "ยูเนี่ยน" (union operation) ดังนี้

$$STUDENT = STUDENT_1 \text{ UN } STUDENT_2$$

โดยที่ R UN S หมายถึงการนำรีเลชัน R และรีเลชัน S มายูเนี่ยนกัน

3. เงื่อนไขของการไม่คาบเกี่ยวกัน เงื่อนไขนี้เป็นจริงเสมอสำหรับการแตกกระจายตามแนวนอน เพราะข้อตกลงที่ใช้กำหนดเงื่อนไขของการแตกกระจายเป็นข้อตกลงที่แยกจากกันเด็ดขาด (mutually exclusive) จากตัวอย่างข้างต้น จะเห็นว่าส่วนของ $STUDENT_1$ และ $STUDENT_2$ ถูกกำหนดโดยค่าของแอตตริบิว MAJOR คนละค่ากัน

2.5.2.2 การแตกกระจายตามแนวตั้ง (Vertical Fragmentation) ถ้ารีเลชัน R ถูกแตกกระจายตามแนวตั้งออกเป็นแฟรกเมนต์ R_1, R_2,..., R_r แล้ว แต่ละแฟรกเมนต์จะประกอบไปด้วยแอตตริบิวต่าง ๆ ที่เป็นส่วนหนึ่งของรีเลชัน R รวมทั้งคีย์หลักของรีเลชัน R ด้วย วัตถุประสงค์ของการแตกกระจายตามแนวตั้ง คือ การแบ่งรีเลชันออกเป็นกลุ่มของรีเลชันย่อย ๆ เพื่อให้งานประยุกต์หลาย ๆ งาน สามารถใช้รีเลชันย่อย ๆ เพียงรีเลชันเดียว โดยวิธีการแตกกระจายที่เหมาะสม (optimal) จะเป็นวิธีที่ทำให้เวลาในการปฏิบัติงานของงานประยุกต์บนแฟรกเมนต์เหล่านี้ต่ำสุด (minimizes excution time)

การแตกกระจายตามแนวตั้งนี้ สามารถทำได้ 2 วิธีคือ

1. การรวมกลุ่ม (Grouping) เริ่มจากการกำหนดให้แต่ละแอตตริบิวเป็นหนึ่งแฟรกเมนต์ แล้วจึงนำแฟรกเมนต์เหล่านั้นมารวมกัน จนกว่าจะครบตามเกณฑ์ (criterion) ที่ได้กำหนดไว้

2. การแบ่งกลุ่ม (Splitting) วิธีนี้เป็นการแบ่งกลุ่มของแอตตริบิวออกเป็นแฟรกเมนต์ โดยมีพื้นฐานอยู่บนพฤติกรรมการเรียกใช้ข้อมูลของงานประยุกต์ที่มีต่อแอตตริบิวเหล่านั้น วิธีนี้มักใช้กับกลยุทธ์ในการออกแบบจากบนลงล่าง และเป็นวิธีซึ่งสามารถสร้างรีเลชันที่ไม่คาบเกี่ยวกัน และใกล้เคียงกับรีเลชันเดิม ได้มากกว่าวิธีการรวมกลุ่ม

เราสามารถกำหนดการแตกกระจายตามวิธีนี้ได้ โดยแตกกระจายรีเลชันระดับรวมตามแอตตริบิว ด้วยวิธีโปรเจคชัน (Projection operation) ตัวอย่างของการแตกกระจายตามแนวตั้ง แสดงได้ดังตารางที่ 3.1(ข) ดังนี้ จากรีเลชันระดับรวมข้างต้น ถ้าให้แฟรกเมนต์ $STUDENT_1$ เก็บข้อมูลของนักเรียนเฉพาะรหัสนักเรียน ชื่อนักเรียน และคะแนนที่ได้ และให้แฟรกเมนต์ $STUDENT_2$ เก็บข้อมูลของนักเรียนเฉพาะรหัสนักเรียนและวิชาเอกเท่านั้น ข้อตกลงของการแตกกระจายตามแนวตั้ง จึงสามารถกำหนดได้ดังนี้

$$STUDENT_1 = PJ_{STUID,STUNAME,CREDITS}STUDENT$$

$$STUDENT_2 = PJ_{STUID,MAJOR}STUDENT$$

โดยที่ $PJ_{Attr}R$ หมายถึง การโพรเจคชันจากรีเลชัน R โดยเลือกเฉพาะแอตตริบิวบางตัวตามที่ระบุใน Attr

การตรวจสอบความถูกต้อง สามารถทำได้ดังนี้

1. เงื่อนไขของความครบถ้วน เงื่อนไขนี้ถูกรับรองโดยกระบวนการแบ่งอยู่ในตัว เพราะแอตตริบิวแต่ละตัวของรีเลชันระดับรวม จะถูกแมปไปยังแฟรกเมนต์หนึ่งแฟรกเมนต์เสมอ

STUID	STUNAME	MAJOR	CREDIT
S1001	Smith, Tom	Art	90
S1015	Jones, Mary	Math	42
S1012	Chin, David	Math	36
S1020	Riversa, Jane	Art	15
S1013	McCarthy, Owen	Math	9

(ก) การแตกกระจายตามแนวนอน : $STUDENT_1$ = SLmajor = 'Math' STUDENT

STUID	STUNAME	MAJOR	CREDIT
S1001	Smith, Tom	Art	90
S1015	Jones, Mary	Math	42
S1012	Chin, David	Math	36
S1020	Riversa, Jane	Art	15
S1013	McCarthy, Owen	Math	9

(ข) การแตกกระจายตามแนวตั้ง : $STUDENT_2$ = PJ STUID='Major' STUDENT

STUID	STUNAME	MAJOR	CREDIT
S1001	Smith, Tom	Art	90
S1015	Jones, Mary	Math	42
S1012	Chin, David	Math	36
S1020	Riversa, Jane	Art	15
S1013	McCarthy, Owen	Math	9

(ค) การแตกกระจายแบบผสม : $STUDENT_{21}$ = SL MAJOR='Math' PJ STUID,MAJOR STUDENT

ตารางที่ 3.1 แสดงรูปแบบการแตกกระจาย

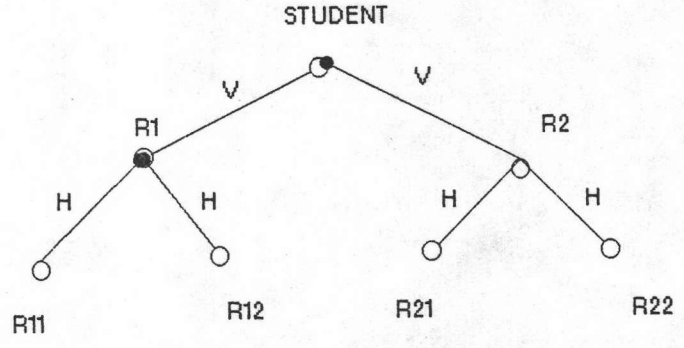
2. เงื่อนไขของการสร้างใหม่ การสร้างรีเลชันระดับรวมสามารถทำได้ โดยใช้การจอย (join operation) จากตัวอย่างเราสามารถสร้างรีเลชันระดับรวม STUDENT ขึ้นใหม่ โดย

$$STUDENT = STUDENT_1 SJ_{STUID=STUID} STUDENT_2$$

โดยที่ $RSJ_F S$ หมายถึง การนำรีเลชัน R และรีเลชัน S มาเซมิจอย (semi-join relation) ตามเงื่อนไข F

การแตกกระจายตามแนวตั้งนี้ จะเป็นไปตามเงื่อนไขของการสร้างใหม่เสมอ ตราบเท่าที่การแตกกระจายดังกล่าว มีคุณสมบัติของความครบถ้วน

2.5.2.3 การแตกกระจายฐานข้อมูลแบบผสม (Hybrid, mixed or nested fragmentation) วิธีนี้เป็นการผสมระหว่างการแตกกระจายตามแนวนอนและแนวตั้ง ดังนั้นการแตกกระจายฐานข้อมูลตามวิธีนี้จึงต้องใช้ทั้งชีเลคชัน และโปรเจคชันร่วมกัน เพื่อช่วยลดความซับซ้อนที่เกิดขึ้น เราควรสร้างแผนภาพต้นไม้ (fragmentation tree) จากตัวอย่างข้างต้น เราอาจโปรเจคชันเฉพาะแอตตริบิวเลขประจำตัวนักเรียน (STUID) และวิชาหลัก (MAJOR) แล้วจึงชีเลคชันเฉพาะนักเรียนที่เรียนวิชาคณิตศาสตร์ (Math) เป็นวิชาหลัก ดังแสดงอยู่ในตารางที่ 3.1(ค) และเราสามารถสร้างต้นไม้ของการแตกกระจายได้ดังแสดงอยู่ในรูป 3.6



รูปที่ 3.6 แสดงแผนภาพต้นไม้ของการแตกกระจายแบบผสม

2.6 การกำหนดสถานที่เก็บข้อมูล

หลังจากทำการแตกกระจายฐานข้อมูลออกเป็นแฟรกเมนต์แล้ว ขั้นตอนต่อไปก็คือ การกระจายแฟรกเมนต์ออกไปตามสถานีต่าง ๆ

2.6.1 ปัจจัยที่ใช้ในการกำหนดสถานที่เก็บข้อมูล

การกระจายแฟรกเมนต์ไปยังสถานีต่าง ๆ ถูกกำหนดโดยปัจจัยต่าง ๆ ดังนี้คือ

1. การอ้างอิงข้อมูลแบบท้องถิ่น (Locality of data reference) ข้อมูลควรจะถูกเก็บอยู่ ณ สถานีที่มีการเรียกใช้บ่อยที่สุด โดยผู้ออกแบบควรศึกษาว่าแต่ละสถานีมีการใช้งานประยุกต์อะไรบ้าง เพื่อจะได้รู้ว่าควรจะเก็บข้อมูลอย่างไร

2. ความน่าเชื่อถือของข้อมูล (Reliability of data) ผู้ออกแบบควรคำนึงถึงการกู้ข้อมูล เมื่อเกิดความเสียหายทางกายภาพขึ้น โดยการเก็บข้อมูลมากกว่าหนึ่งสำเนา (multiple copies of data) ไว้ที่สถานีต่าง ๆ มากกว่าหนึ่งสถานี

3. ความมีอยู่ของข้อมูล (Data availability) เช่นเดียวกับเพื่อความน่าเชื่อถือของข้อมูล การเก็บข้อมูลมากกว่าหนึ่งสำเนา ทำให้ผู้ใช้สามารถมีข้อมูลใช้หรือทำงานได้ตลอด แม้ว่าอีกสถานี ซึ่งปกติจะไปเรียกใช้ข้อมูล จะเสียก็ตาม โดยสามารถเรียกใช้ข้อมูลจากสถานีอื่น ที่เก็บข้อมูลอีกสำเนาหนึ่งแทน

4. ความสามารถในการเก็บข้อมูล (Storage capacities) แต่ละสถานีอาจมีความสามารถในการเก็บข้อมูลต่างกัน จึงต้องตัดสินใจว่าจะเก็บข้อมูลไว้ที่สถานีไหนดี

5. การกระจายภาระในการทำงาน (Distribution of processing load) เหตุผลหนึ่งที่ทำให้เราเลือกใช้ระบบกระจาย คือ การกระจายภาระในการทำงาน เพื่อเพิ่มประสิทธิภาพในการประมวลผลข้อมูล อย่างไรก็ตามปัจจัยนี้จะต้องสมดุลย์กับปัจจัยด้านการอ้างอิงข้อมูลแบบท้องถิ่น

6. ต้นทุน สมมุติ $F = \{F_1, F_2, \ldots, F_n\}$ คือ แฟรกเมนต์ต่าง ๆ ของฐานข้อมูลระดับรวม และ $S = \{S_1, S_2, \ldots, S_n\}$ คือ สถานีต่าง ๆ ในเครือข่าย ดังนั้นการจะเลือกกลยุทธ์การกระจายข้อมูลแบบใดถึงจะเหมาะสมที่สุด ควรจะต้องพิจารณาถึงต้นทุนที่เกิดขึ้น เช่น ต้นทุนในการเก็บแฟรกเมนต์ F_i ไว้ที่สถานี S_j ต้นทุนในการสอบถามส่วนต่าง ๆ ของฐานข้อมูล F_i ซึ่งเก็บอยู่ที่สถานี S_j ต้นทุนของการแก้ไขส่วนต่าง ๆ ของฐานข้อมูล F_i ในทุกๆ สถานี รวมทั้งต้นทุนในการสื่อสารข้อมูลระหว่างสถานีในเครือข่ายด้วย

2.6.2 ทางเลือกในการกระจายข้อมูล

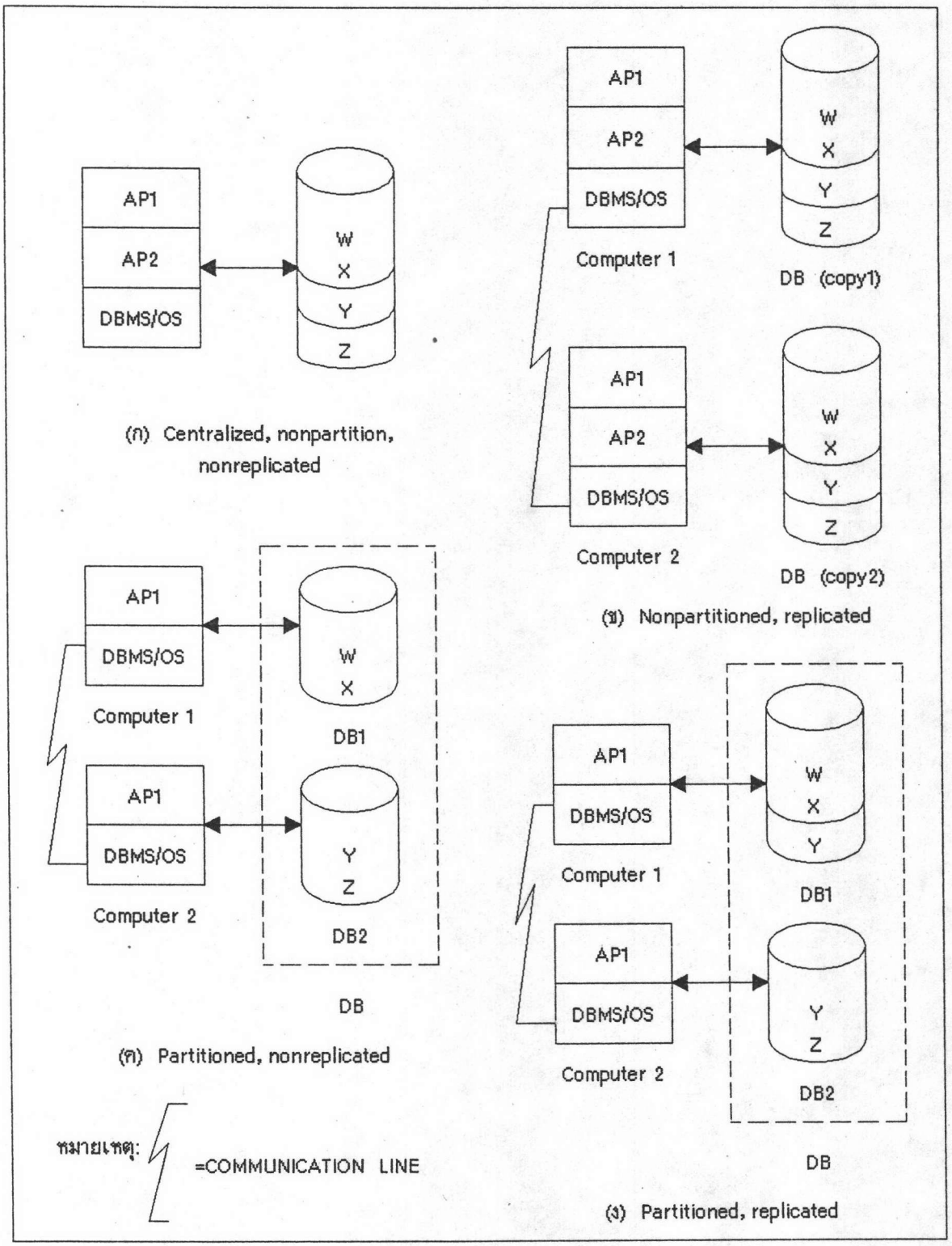
จากรูป 3.7 สมมุติฐานข้อมูลในที่นี้ ประกอบไปด้วยส่วนของฐานข้อมูล 4 ส่วน (แฟรกเมนต์) คือ ส่วน W X Y และ Z ดังนั้นการกระจายข้อมูลไปตามสถานีต่างๆ มีทางเลือกอยู่ 4 วิธี ดังนี้ [13,18]

1. แบบรวมศูนย์ (Centralized) วิธีนี้เป็นการเก็บฐานข้อมูลทั้งหมด และระบบจัดการฐานข้อมูลไว้ที่สถานีเดียวกัน แต่ผู้ใช้จะถูกกระจายออกไปตามสถานีต่าง ๆ วิธีนี้เป็นเพียงการกระจายการประมวลผลเท่านั้น ไม่ได้กระจายข้อมูลจริง ๆ แต่อย่างใด วิธีนี้ต้นทุนในการเรียกใช้ข้อมูลจะสูง เพราะแต่ละสถานีจะต้องอาศัยเครือข่ายในการเรียกใช้ข้อมูลทั้งสิ้น ยกเว้นสถานี ซึ่งเป็นที่เก็บข้อมูลเพียงสถานีเดียวเท่านั้น แต่วิธีนี้มีต้นทุนในการเก็บข้อมูลต่ำ เพราะข้อมูลถูกเก็บอยู่ที่สถานีเดียว การอ้างอิงข้อมูลแบบท้องถิ่น ความน่าเชื่อถือ และความมีอยู่ของข้อมูลต่ำ เพราะหากสถานีที่เก็บข้อมูลเกิดขัดข้องขึ้น ก็ไม่มีสถานีใดสามารถใช้งานได้เลย

2. แบบเก็บซ้ำ (Replicated) ทางเลือกนี้เป็นการเก็บสำเนาข้อมูลทั้งหมดไว้ที่สถานีต่าง ๆ ประโยชน์ที่ได้รับจากวิธีนี้คือ การอ้างอิงข้อมูลแบบท้องถิ่น ความน่าเชื่อถือ ความมีอยู่ของข้อมูล และการกระจายภาระในการทำงานมีสูงสุด แต่ต้นทุนที่ใช้ในการเก็บข้อมูลจะสูงที่สุดเช่นกัน ส่วนต้นทุนที่ใช้ในการติดต่อสื่อสารเพื่อให้ได้ข้อมูลจะต่ำ ขณะที่ต้นทุนในการแก้ไขข้อมูลจะสูง เพราะข้อมูลถูกเก็บอยู่หลายที่ จึงจำเป็นจะต้องแก้ไขให้ตรงกัน วิธีนี้เหมาะกับระบบ ซึ่งข้อมูลไม่ค่อยมีการเปลี่ยนแปลงบ่อยนัก

3. แบบเก็บบางส่วน (Partitioned) วิธีนี้เป็นการกระจายส่วนของฐานข้อมูลเพียงสำเนาเดียว ไปยังสถานีต่างๆ โดยแต่ละสำเนาต้องไม่มีส่วนของข้อมูลที่คาบเกี่ยวกัน หากข้อมูลถูกเก็บตามสถานีที่เรียกใช้ข้อมูลบ่อย วิธีนี้ก็จะมีการอ้างอิงข้อมูลแบบท้องถิ่นสูง แต่ความน่าเชื่อถือ และความมีอยู่ของข้อมูลต่ำ เพราะข้อมูลถูกเก็บอยู่ที่สถานีเดียวเท่านั้น อย่างไรก็ตามวิธีนี้ยังดีกว่าวิธีแบบรวมศูนย์ เพราะในกรณีนี้หากสถานีใดขัดข้อง สถานีอื่นๆ ยังคงทำงานได้อย่างปกติ หากไม่มีการเรียกใช้ข้อมูลที่เก็บอยู่ที่สถานีที่ขัดข้อง ต้นทุนในการเก็บข้อมูลจะต่ำ และหากได้รับการออกแบบอย่างดีแล้ว ต้นทุนในการสื่อสารก็จะต่ำด้วยเช่นกัน การกระจายภาระการทำงานจะขึ้นอยู่กับลักษณะการกระจายข้อมูล

4. แบบผสม วิธีนี้แต่ละสถานีจะเก็บส่วนของฐานข้อมูล ซึ่งแตกต่างกันออกไป ทั้งนี้ขึ้นอยู่กับรูปแบบการใช้ข้อมูล เช่น ข้อมูลส่วนใดที่ใช้เฉพาะสถานีนั้นๆ ก็จะถูกเก็บแบบบางส่วน แต่หากมีข้อมูลที่ถูกเรียกใช้ร่วมกันโดยสถานีต่าง ๆ แล้ว ข้อมูลส่วนนั้นจะถูกเก็บแบบซ้ำ เนื่องจากทางเลือกนี้ถูกออกแบบมาเพื่อการกระจายข้อมูลให้เหมาะสมที่สุด ดังนั้นจึงเป็นการรวมส่วนดีของวิธีอื่น ๆ เข้ามาไว้ด้วยกัน



รูปที่ 3.7 แสดงทางเลือกในการกระจายฐานข้อมูล

เราสามารถสรุปข้อดี และข้อเสียของทางเลือกที่ใช้ในการกระจายข้อมูลได้ดังตาราง 3.2

	ALTERNATIVE			
CRITERION	CENTRALIZED	REPLICATE	PARTITION	HYBRID
LOCALITY OF REFERENCE	lowest	highest	should be high *	should be high *
RELIABILITY	lowest	highest	hight for system low for item	*
AVAILABILITY	lowest	highest	hight for system low for item	*
STORAGE COSTS	lowest	highest	lowest	should be average *
LOAD DISTRIBUTION	poor	best	good	should be good *
COMMUNICATION COST	highest	low exoept for updates	should be low *	should be low *

* depends on exact placement decisions made

ตารางที่ 3.2 แสดงการเปรียบเทียบข้อดี-ข้อเสียของทางเลือกในการกระจายฐานข้อมูล

โมเดลข้อมูลเชิงตรรก (Logical Data Model) [12,13]

โมเดลข้อมูลเชิงตรรกเป็นเทคนิคที่ดีที่สุด ที่ใช้ในการสร้าง และบำรุงรักษาการควบคุมทรัพยากรข้อมูลขององค์กร ปรัชญาของโมเดลข้อมูลเชิงตรรก คือ การวิเคราะห์และบันทึกความจริงขององค์กรธุรกิจที่มีอยู่จริง และเป็นอิสระจากวิธีการเข้าถึงข้อมูล ผู้ใช้ และไม่ว่าจะนำคอมพิวเตอร์มาใช้หรือไม่ก็ตาม เช่น การสั่งซื้อสินค้า การรับชำระเงิน ความจริงเหล่านี้สามารถแทนลงในโมเดลข้อมูลเชิงตรรก เพื่อให้เราสามารถเข้าใจลักษณะของธุรกิจได้มากขึ้น แม้ว่าจะไม่ได้ติดตั้งความจริงที่เกิดขึ้นได้ครบก็ตาม

1. ลักษณะของโมเดลข้อมูลเชิงตรรก

1.1 สามารถแสดงได้ด้วยแผนภาพ (Graphical diagrams) เทคนิคที่ใช้ในการทำโมเดลข้อมูลส่วนใหญ่จะมีภาษา และรูปภาพเฉพาะตัว ซึ่งรูปภาพนี้จะช่วยแสดงถึงภาพรวมและรายละเอียดของข้อมูลได้อย่างดี ภาพที่ใช้ส่วนมากมักใช้สี่เหลี่ยมหรือวงกลม เพื่อแทนเอนติตี (entity) และใช้เส้นโค้งหรือเส้นตรง แทนรีเลชันชิป (relationship) ระหว่างเอนติตี

1.2 การแสดงความหมายของข้อมูลที่ชัดเจน (Explicit representation of semantic) ระดับความชอบของแต่ละคนในการใช้จำนวนรูปภาพ เพื่อแสดงความหมายของข้อมูลต่างกัน แต่จริง ๆ แล้วเทคนิคของการทำโมเดลข้อมูล เพียงเพื่อต้องการแผนภาพที่สามารถอธิบายความหมายของข้อมูลที่ชัดเจนเท่านั้น ดังนั้นแผนภาพที่ใช้จึงควรง่าย ไม่ซับซ้อน โดยเฉพาะอย่างยิ่งสัญลักษณ์หนึ่ง ๆ ไม่ควรมีหลายความหมาย

1.3 การแสดงรายละเอียดในระดับที่เหมาะสม (Appropriate level of Detail) โมเดลข้อมูลควรสนับสนุนระดับของรายละเอียดที่ต่างกัน นั่นคือ โมเดลข้อมูลระดับบน (High level logical data model) ควรมีรายละเอียดของเอนติตี รีเลชันชิป และข้อบังคับต่าง ๆ น้อยกว่าโมเดลข้อมูลระดับล่าง หรือเรียกอีกอย่างหนึ่งว่าโมเดลข้อมูลระดับกายภาพ (Low level data model or Physical data model)

1.4 ความเป็นอิสระจากระบบจัดการฐานข้อมูล (DBMS independence) โมเดลที่ได้จากการออกแบบไว้ ควรเป็นอิสระจากระบบจัดการฐานข้อมูลที่มีอยู่ และควรใช้ได้กับระบบจัดการฐานข้อมูลหลายแบบ เช่น ระบบจัดการฐานข้อมูลแบบความสัมพันธ์ แบบลำดับชั้น และแบบเครือข่าย

1.5 ง่ายต่อการศึกษา และใช้งาน (Easy to learn and use) โมเดลที่ใช้จะต้องง่ายเพียงพอที่ผู้ใช้จะสามารถทำความเข้าใจ และนำไปใช้งานได้

2. ขั้นตอนการออกแบบโมเดลข้อมูลเชิงตรรก

การสร้างโมเดลข้อมูลเชิงตรรก จำเป็นต้องพิจารณาโครงสร้างของมุมมองของผู้ใช้ ซึ่งเป็นความต้องการต่าง ๆ ของผู้ใช้หรือระบบ โครงสร้างจากมุมมองของผู้ใช้นี้ ถูกใช้เป็นข้อมูลของการออกแบบฐานข้อมูล รวมทั้งการพัฒนาต้นแบบของฐานข้อมูลด้วย

ต้นแบบ หมายถึง การทดลองสร้างบางส่วนของการติดตั้งทั้งหมด (ในที่นี้หมายถึงการติดตั้งฐานข้อมูล) เพื่อทดสอบว่าระบบมีความต้องการเพียงพอหรือไม่

ขั้นตอนการออกแบบโมเดลข้อมูลเชิงตรรก เป็นการกำหนดเอนติตี รีเลชันชิป คีย์หลัก (Primary Key) คีย์รอง (Alternate Key) คีย์ภายนอก (Foreign Key) คีย์กฎธุรกิจ (Key business Rule) และการกำหนดระดับการมองเห็นข้อมูลของผู้ใช้ (User view) โดยเราสามารถสรุปขั้นตอนการออกแบบโมเดลข้อมูลเชิงตรรก ได้ดังปรากฏในรูป 3.8 และสามารถอธิบายได้ดังนี้

2.1 การกำหนดเอนติตีหลัก

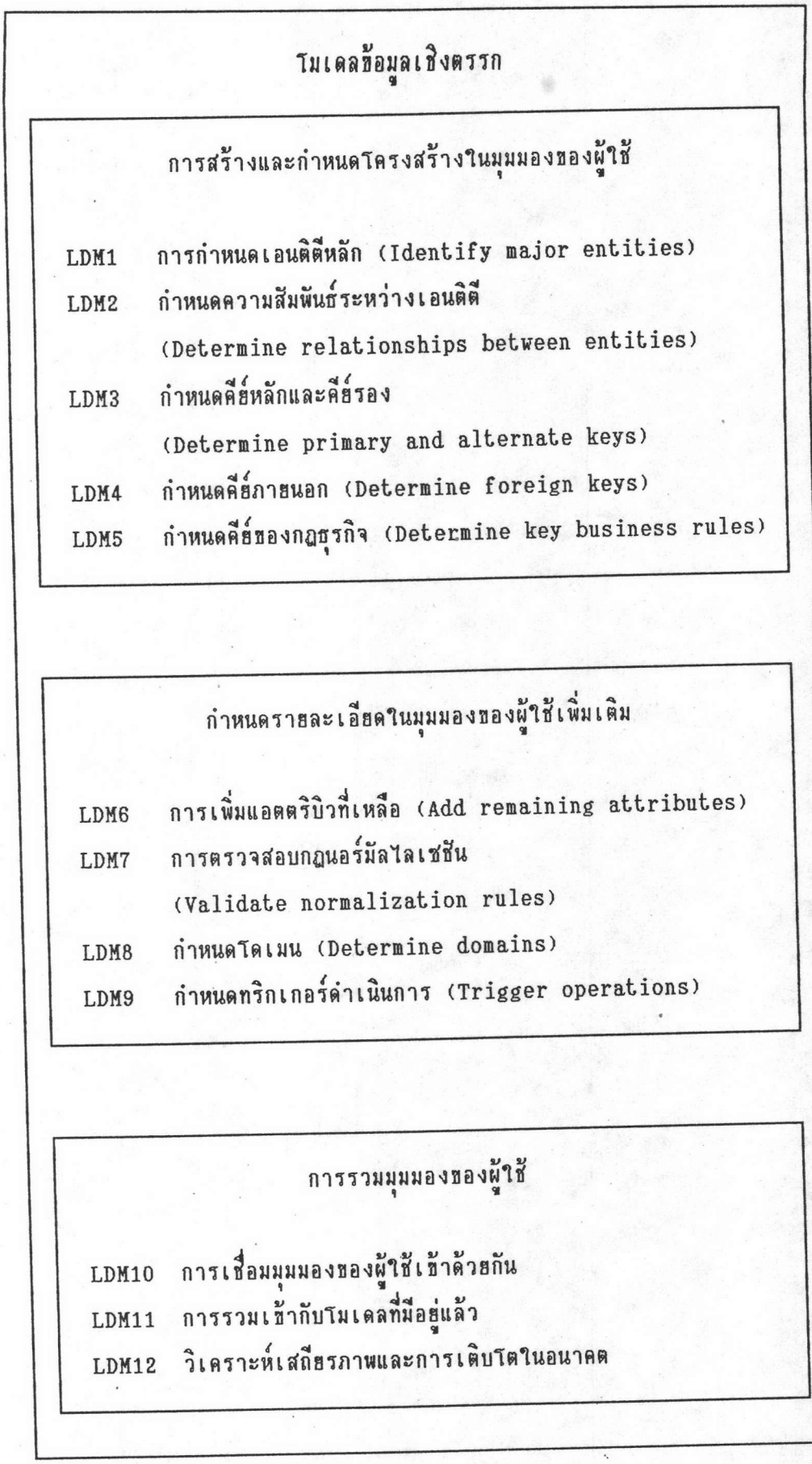
เอนติตี หมายถึง คน สิ่งของ สถานที่ หรือความคิดที่เราสนใจ เอนติตีหนึ่ง ๆ สามารถแบ่งออกเป็นซับไทป์ หลาย ๆ ซับไทป์ (Subtypes) โดยเอนติตีซับไทป์จะมีความหมาย และคุณสมบัติ เฉพาะตัวมากขึ้นกว่าเอนติตีที่เป็นซุปเปอร์ไทป์ (Supertype) เช่น เราสามารถแบ่งเอนติตีของการสั่งซื้อ ออกเป็นเอนติตีของการยกเลิกการสั่งซื้อ และการสั่งซื้อที่สมบูรณ์ โดยทั่วไปเอนติตี X เป็น ซับไทป์ของเอนติตี Y และเอนติตี Y เป็นซุปเปอร์ไทป์ของเอนติตี X ก็ต่อเมื่อ

2.1.1 เอนติตี X และเอนติตี Y ต้องแทนสิ่งเดียวกัน

2.1.2 เอนติตี X จะมีคุณสมบัติเช่นเดียวกับเอนติตี Y และยังมีคุณสมบัติพิเศษ ซึ่งต่างไปจากเอนติตี Y

2.1.3 ทุก ๆ ค่าของ X จะปรากฏใน Y ได้เพียงหนึ่ง ในทางกลับกันค่าที่ปรากฏใน Y ไม่จำเป็นต้องมีอยู่ใน X

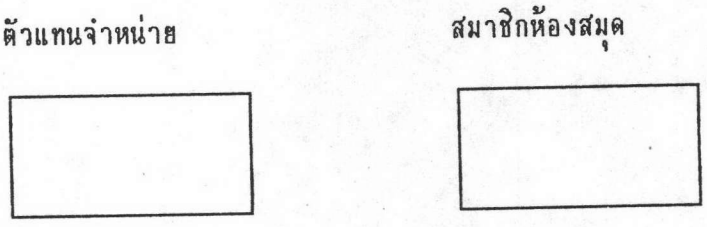
นอกจากนี้เรายังสามารถแบ่งกลุ่มของซับไทป์ ออกเป็นเซตของซับไทป์ซึ่งแยกจากกันอย่างเด็ดขาด (mutually exclusive subtypes) เราเรียกซับไทป์ชนิดนี้ว่าแคทิกอรี (categories) นั่นคือเหตุการณ์ที่เกิดขึ้นในซุปเปอร์ไทป์ จะปรากฏในซับไทป์แคทิกอรีได้เพียงหนึ่งซับไทป์เท่านั้น



รูปที่ 3.8 ขั้นตอนการออกแบบโมเดลข้อมูลเชิงตรรก

หลังจากกำหนดเอนติตีหลักได้แล้ว ต่อไปจะเป็นการกำหนดชื่อ แผนภาพและสารสนเทศที่สัมพันธ์กัน แล้วบันทึกไว้ในพจนานุกรมข้อมูล เพื่อเป็นเอกสารอ้างอิงในขั้นตอนต่อไป

แผนภาพโมเดลข้อมูลเชิงตรรก เป็นภาพที่ใช้แทนโมเดลข้อมูลเชิงตรรก ซึ่งประกอบด้วยสารสนเทศของความสัมพันธ์ต่าง ๆ เอนติตีเป็นสิ่งแรกที่ถูกวาดลงในแผนภาพ โดยแสดงด้วยรูปสี่เหลี่ยม และเขียนชื่อของเอนติตีไว้ที่มุมบนด้านซ้ายของรูปสี่เหลี่ยม ดังแสดงในรูป 3.9



รูปที่ 3.9 ตัวอย่างเอนติตีหลักของระบบห้องสมุด

2.2 การกำหนดรีเลชันชิประหว่างเอนติตี

รีเลชันชิป หมายถึง ความจริงที่เกี่ยวข้อง หรือความสัมพันธ์ระหว่างเอนติตี 2 เอนติตี เราสามารถจำแนกรีเลชันชิป ออกเป็น 3 ชนิด คือ

2.2.1 รีเลชันชิปที่มีอยู่จริง (Existance Relationships) เช่น ลูกจ้างมีลูก

2.2.2 รีเลชันชิปตามหน้าที่ (Functional Relationships) เช่น ครูสอนนักเรียน

2.2.3 รีเลชันชิปตามเหตุการณ์ (Event Relationships) เช่น ลูกค้าสั่งสินค้า

ในขณะทำการกำหนดรีเลชันชิป จะต้องกำหนดชื่อ แผนภาพและรายละเอียดที่สัมพันธ์กัน แล้วบันทึกลงในพจนานุกรมข้อมูลเช่นเดียวกับการกำหนดเอนติตีหลัก รีเลชันชิปจะมีทิศทางอยู่ระหว่างเอนติตี 2 เอนติตีที่เกี่ยวข้องกัน โดยจะมีทิศทางจากเอนติตีหนึ่ง ซึ่งเป็นเอนติตีแม่ (Parent Entity) ไปยังอีกเอนติตีหนึ่ง ซึ่งเรียกว่าเอนติตีลูก (Child Entity)

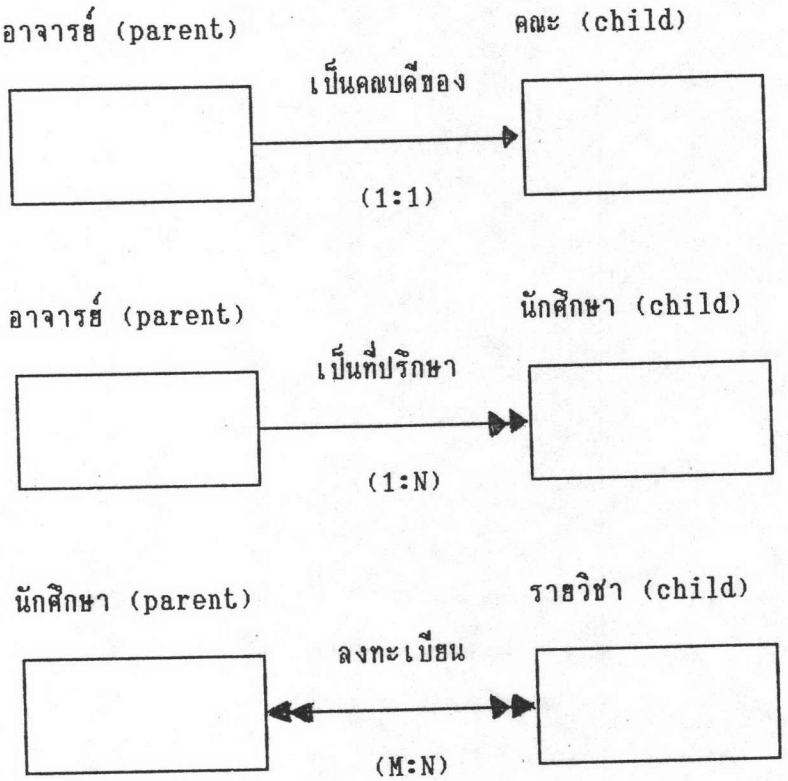
รีเลชันชิปสามารถแบ่งออกเป็นชนิดต่าง ๆ ตามสัดส่วนของความสัมพันธ์ (Cardinality Ratio) ได้ 3 แบบ ซึ่งแสดงอยู่ในรูป 3.10 และสามารถอธิบายได้ดังนี้คือ

1. แบบหนึ่งต่อหนึ่ง (one-to-one, 1:1 relationship) แต่ละค่าของเอนติตีแม่ จะมีความสัมพันธ์กับเอนติตีลูกได้อย่างมากที่สุดเพียงหนึ่งค่า สัญลักษณ์ที่ใช้ คือ

(P) ------> (C)

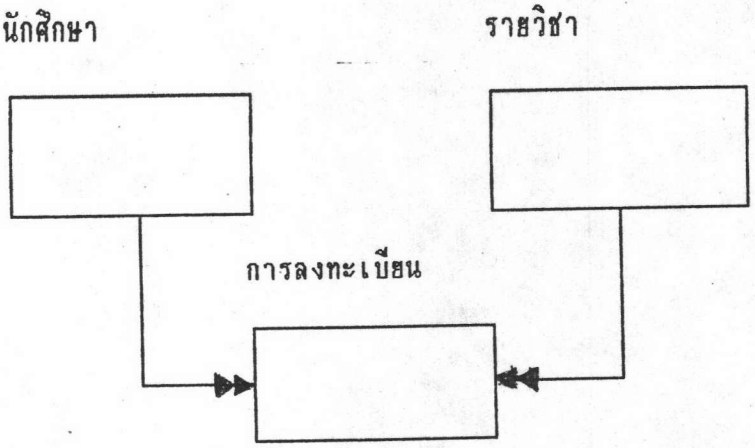
2. แบบหนึ่งต่อหลาย (one-to-many, 1:N relationship) แต่ละค่าในเอนติตีแม่จะมีความสัมพันธ์กับเอนติตีลูกได้หลายค่า แต่ในทางกลับกัน แต่ละค่าในเอนติตีลูกจะมีความสัมพันธ์กับเอนติตีแม่ได้เพียงค่าเดียว สัญลักษณ์ที่ใช้ คือ (P) ------>> (C)

3. แบบหลายต่อหลาย (many-to-many, M:N relationship) แต่ละค่าในเอนติตีแม่จะมีความสัมพันธ์กับเอนติตีลูกได้หลายค่า และเช่นเดียวกัน แต่ละค่าในเอนติตีลูกจะมีความสัมพันธ์กับเอนติตีแม่ได้หลายค่า สัญลักษณ์ที่ใช้ คือ (P) <<------>> (C)



รูปที่ 3.10 แสดงชนิดของรีเลชันชิป ซึ่งขึ้นอยู่กับสัดส่วนความสัมพันธ์

ความสัมพันธ์แบบหลายต่อหลายเป็นความสัมพันธ์ที่ต้องพิจารณาเป็นพิเศษ เพราะวิธีการแทนความสัมพันธ์นี้ ในการทำโมเดลข้อมูลเชิงตรรก และการออกแบบฐานข้อมูลค่อนข้างยุ่งยาก จึงจำเป็นต้องแปลงความสัมพันธ์นี้ ออกเป็นความสัมพันธ์แบบหนึ่งต่อหลายสองความสัมพันธ์ เช่น จากความสัมพันธ์แบบหลายต่อหลาย ที่ว่านักศึกษาหนึ่งคน สามารถลงทะเบียนเรียนได้มากกว่าหนึ่งวิชา และวิชาหนึ่ง ๆ ก็สามารถมีนักศึกษาลงทะเบียนเรียนได้มากกว่าหนึ่งคน ดังนั้นเราสามารถแปลงความสัมพันธ์นี้ ออกเป็น 3 เอนติตี และมีความสัมพันธ์กันดังรูปที่ 3.11



รูปที่ 3.11 แสดงการแปลงความสัมพันธ์แบบหลายต่อหลาย

2.3 การกำหนดคีย์หลักและคีย์รอง

ในขั้นนี้เป็นการกำหนดรายละเอียดต่าง ๆ ของเอนติตี โดยทำการกำหนดแอตตริบิว ซึ่งเป็นหน่วยที่เล็กที่สุดของข้อมูล ที่มีความเกี่ยวข้องกับเอนติตี และเป็นความจริงที่ไม่สามารถแบ่งเป็นส่วนย่อย ๆ ไปกว่านี้ได้ ตัวอย่างแอตตริบิวที่ใช้อธิบายเอนติตีสมาชิกได้แก่ เลขประจำตัวสมาชิก ชื่อสมาชิก ที่อยู่สมาชิก เป็นต้น

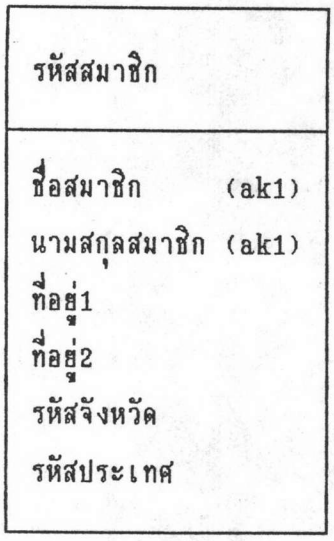
แอตตริบิวตัวแรกที่เราจะกำหนดให้กับเอนติตีในโมเดลข้อมูลเชิงตรรก คือ คีย์หลักและคีย์รอง โดยแอตตริบิวหนึ่งแอตตริบิว หรือเซตของแอตตริบิวใด ๆ ที่สามารถใช้ระบุค่าใดค่าหนึ่งของเอนติตีได้เป็นหนึ่งค่า (Unique) จะเรียกแอตตริบิวนั้น หรือเซตของแอตตริบิวนั้นว่าเป็น คีย์แข่งขัน (Candidate key) ซึ่งถ้าแคนดิเดทคีย์นั้นประกอบด้วยแอตตริบิวมากกว่าหนึ่งตัว หรือเป็นเซตของแอตตริบิวแล้ว เราจะเรียกแคนดิเดทคีย์นั้นว่า คีย์ประกอบ (Composite key)

การกำหนดคีย์หลักของทุกเอนติตี จะเลือกจากแคนดิเดทคีย์ตัวใดตัวหนึ่งที่มี ซึ่งแคนดิเดทคีย์ที่ไม่ได้ถูกเลือกเป็นคีย์หลักจะเป็นคีย์รอง ในกรณีที่คีย์หลักและคีย์รองเป็นคีย์ประกอบแอตตริบิวหนึ่งอาจเป็นส่วนของคีย์หลักและคีย์รองได้มากกว่าหนึ่งคีย์

สิ่งที่สำคัญอีกอันคือ เอนติตีที่เป็นซับไทป์จะต้องมีคีย์หลักอันเดียวกับเอนติตีที่เป็นซุปเปอร์ไทป์ของมัน หลังจากกำหนดคีย์หลักและคีย์รองแล้ว ให้บันทึกชื่อและความหมายลงในพจนานุกรมข้อมูลด้วย

สำหรับแผนภาพโมเดลข้อมูลเชิงตรรก คีย์หลักจะถูกเขียนอยู่ภายในกรอบสี่เหลี่ยม ซึ่งใช้แทนเอนติตี โดยเขียนให้อยู่เหนือเส้นนอน และเขียนชื่อแอตตริบิวที่เป็นคีย์รองไว้ใต้เส้นนอน ตามด้วยสัญลักษณ์ akn โดยที่ n หมายถึงคีย์รองตัวที่ n ของเอนติตีนั้น ๆ ดังแสดงในรูปที่ 3.12

สมาชิก

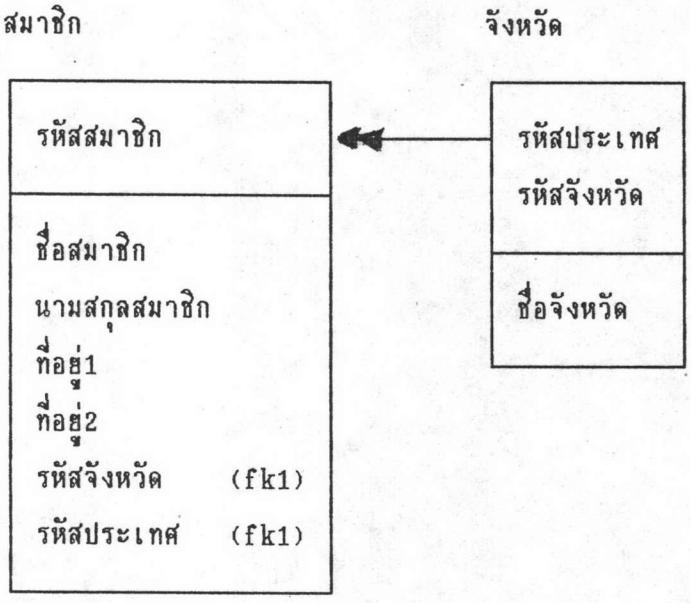


รูปที่ 3.12 แสดงเอนติตีสมาชิก ซึ่งแสดงให้เห็นถึงคีย์หลัก และคีย์รอง

2.4 การกำหนดคีย์ภายนอก

คีย์ภายนอก เป็นแอตตริบิว หรือเซตของแอตตริบิว ซึ่งมีความสัมพันธ์ที่ถูกกำหนดโดยเอนติตีแม่ คีย์ภายนอกจะถูกกำหนดไว้ที่เอนติตีลูก โดยที่แอตตริบิว หรือเซตของแอตตริบิวดังกล่าวเป็นคีย์หลักของเอนติตีแม่

จากรูป 3.13 จะเห็นว่า เอนติตีจังหวัดเป็นเอนติตีแม่ ส่วนเอนติตีลูกในที่นี้คือ เอนติตีสมาชิก โดยมีรหัสจังหวัดเป็นคีย์ภายนอกในเอนติตีลูก และเป็นคีย์หลักในเอนติตีแม่



รูปที่ 3.13 แสดงความสัมพันธ์ระหว่าง 2 เอนติตี แสดงให้เห็นถึงคียภายนอก

ความสำคัญของคีย์ภายนอก คือ

1. แสดงให้เห็นถึงกฎธุรกิจ (Business rules) ระหว่างเอนติตีต่างๆ

2. แม้จะดูเหมือนว่าการกำหนดคีย์ภายนอกทำให้เกิดความซ้ำซ้อน แต่จริง ๆ แล้วมันจะเป็นตัวช่วยในการออกแบบ เพื่อให้เราสามารถตรวจสอบได้ว่า เอนติตีอันไหนเป็นเอนติตีลูก อันไหนเป็นเอนติตีแม่

3. ช่วยทำให้ขั้นตอนการออกแบบฐานข้อมูลง่ายขึ้น

ข้อควรระวังคือ ไม่ควรกำหนดให้คีย์รองเป็นคีย์ภายนอกด้วย เพราะจะทำให้เกิดความซับซ้อนโดยไม่จำเป็น นอกจากนี้คีย์รองยังสามารถมีค่าเป็นนัลได้ ดังนั้นหากใช้คีย์รองเป็นคีย์ภายนอก จะทำให้เกิดรีเลชันชิปที่ไม่สมบูรณ์ขึ้น

เมื่อกำหนดคีย์ภายนอกเสร็จแล้ว ก็ให้บันทึกชื่อ ความหมายและความสัมพันธ์ที่เกี่ยวข้องลงในพจนานุกรมข้อมูล

<u>ข้อสังเกต</u>

1. ในโมเดลข้อมูลเราจะกำหนดคีย์หลักไว้ในส่วนบนของเส้นแบ่งภายในเอนติตีและกำหนดคีย์รองและคีย์ภายนอกไว้ใต้เส้นแบ่ง ยกเว้นคีย์ภายนอกที่เป็นส่วนหนึ่งของคีย์หลัก

2. ความสัมพันธ์แบบ (1:1) จะบรรจุคีย์ภายนอกไว้ในเอนติตีลูกเท่านั้น โดยไม่จำเป็นต้องบรรจุคีย์ภายนอกไว้ในเอนติตีแม่อีก

3. เอนติตีที่เป็นซับไทป์ หรือแคทิกอรี่ จะมีคีย์ภายนอกเหมือนคีย์หลักของซุปเปอร์ไทป์ทั้งหมด

2.5 การกำหนดคีย์ของกฎธุรกิจ

กฎธุรกิจเป็นส่วนหนึ่งของการทำโมเดลข้อมูลเชิงตรรก เพื่อความสมบูรณ์และความถูกต้องตรงกันของข้อมูล

กฎธุรกิจแบ่งได้เป็น 3 ประเภทคือ

1. คีย์ของกฎธุรกิจ (Key business rules) เป็นการกำหนดกฎเพื่อความสมบูรณ์ของรีเลชันชิป กฎเหล่านี้เป็นตัวควบคุมผลของการปฏิบัติการต่าง ๆ ต่อรีเลชันชิป อันได้แก่ การเพิ่ม,ลบและแก้ไขข้อมูล เช่น ผลกระทบอะไรบ้าง ที่จะเกิดขึ้นจากการเปลี่ยนค่าของคีย์หลัก หรือคีย์ภายนอก

2. โดเมน (Domains) เป็นการกำหนดความเป็นบูรณภาพของแอตตริบิว และกำหนดข้อบังคับของค่าที่เป็นไปได้สำหรับแอตตริบิว

3. ทริกเกอร์ดำเนินการ (Triggering operations) เป็นการกำหนดผลกระทบจากการเพิ่ม, ลบ, แก้ไข หรือดึงข้อมูล ที่เกิดกับเอนติตีอื่น หรือแอตตริบิวอื่น ๆ ภายในเอนติตีเดียวกัน

ในขั้นนี้ เราจะพิจารณาเฉพาะการกำหนดคีย์ของกฎธุรกิจก่อน ส่วนโดเมนและทริกเกอร์ดำเนินการจะพิจารณาภายหลังในขั้นตอนที่ 8 และ 9 การกำหนดคีย์ของกฎธุรกิจมีขั้นตอนดังต่อไปนี้

2.5.1 กำหนดกฎการแทรก (Insert Rule) ให้กับแต่ละรีเลชันชิป เป็นการกำหนดเงื่อนไขที่เป็นไปได้ในการแทรกเอนติตีลูก หรือเมื่อมีการปรับปรุงคีย์ภายนอกของเอนติตีลูก

กฎการแทรกแบ่งออกเป็น 6 ประเภท คือ

2.5.1.1 ขึ้นต่อกัน (Dependent) เป็นการอนุญาตให้มีการแทรกเอนติตีลูกได้ เมื่อมีค่าที่ตรงกันกับเอนติตีแม่ที่มีอยู่เท่านั้น

2.5.1.2 อัตโนมัติ (Automatic) อนุญาตให้ทำการแทรกเอนติตีลูกได้เสมอ ไม่ว่าจะมีค่าในเอนติตีแม่หรือไม่ก็ตาม และหากไม่มีค่าอยู่ในเอนติตีแม่ก็ให้สร้างค่าในเอนติตีแม่ด้วย

2.5.1.3 กำหนดให้มีค่าเป็นนัล (Nullify) อนุญาตให้แทรกเอนติตีลูกได้เสมอ แต่เมื่อหาค่าในเอนติตีแม่ที่ตรงกันไม่ได้ ให้กำหนดค่าคีย์ภายนอกในเอนติตีลูกเป็นนัล

2.5.1.4 กำหนดให้มีค่าโดยปริยาย (Default) อนุญาตให้ทำการแทรกเอนติตีลูกได้เสมอ แต่หากหาค่าที่ตรงกันกับเอนติตีแม่ไม่ได้ ให้กำหนดค่าคีย์ภายนอกของเอนติตีลูกเท่ากับค่าโดยปริยาย

2.5.1.5 แบบมีธรรมเนียม (Customized) จะอนุญาตให้ทำการแทรกเอนติตีลูกได้ ก็ต่อเมื่อได้ทำการตรวจสอบค่าว่าถูกต้อง ตรงกับเงื่อนไขที่ได้กำหนดไว้

2.5.1.6 ไม่มีผลกระทบ (No effect) อนุญาตให้ทำการแทรกเอนติตีลูกได้โดยไม่มีเงื่อนไขใด ๆ

2.5.2 กำหนดกฎการลบ (Delete Rule) ให้กับแต่ละรีเลชันชิป เป็นการกำหนดเงื่อนไขสำหรับการลบ หรือแก้ไขคีย์หลักในเอนติตีแม่

กฎการลบแบ่งออกเป็น 6 ประเภท คือ

2.5.2.1 มีข้อจำกัด (Restric) จะอนุญาตให้ลบ หรือแก้ไขคีย์หลักในเอนติตีแม่ได้ ถ้าในเอนติตีลูกไม่มีคีย์ภายนอกอ้างถึงเอนติตีแม่

2.5.2.2 แบบต่อเนื่อง (Cascade) อนุญาตให้ทำการลบหรือแก้ไขคีย์หลักในเอนติตีแม่ได้เสมอ และให้ลบเอนติตีลูกด้วย หากมีการอ้างถึงเอนติตีแม่ตัวนั้น ๆ

2.5.2.3 กำหนดให้มีค่าเป็นนัล (Nullify) อนุญาตให้ทำการลบหรือแก้ไขคีย์หลักในเอนติตีแม่ได้เสมอ และถ้าในเอนติตีลูกมีการอ้างถึงเอนติตีแม่ด้วย ให้เปลี่ยนค่าคีย์ภายนอกในเอนติตีลูกเป็นนัล

2.5.2.4 กำหนดให้มีค่าโดยปริยาย (Default) อนุญาตให้ทำการลบหรือแก้ไขคีย์หลักในเอนติตีแม่ได้เสมอ และถ้าในเอนติตีลูกมีการอ้างถึงเอนติตีแม่ด้วย ให้เปลี่ยนค่าคีย์ภายนอกในเอนติตีลูกเป็นค่าโดยปริยาย

2.5.2.5 แบบมีธรรมเนียม (Customized) จะลบหรือแก้ไขข้อมูลในเอนติตีแม่ได้ ก็ต่อเมื่อทำการตรวจสอบเงื่อนไขแล้วว่าถูกต้อง

2.5.2.6 แบบไม่มีผลใดๆ (No effect) สามารถลบหรือแก้ไขข้อมูลในเอนติตีแม่โดยไม่มีเงื่อนไขใด ๆ

ในทางปฏิบัติแล้ว เราควรจะหลีกเลี่ยงการกำหนดคีย์ของกฎธุรกิจ แบบกำหนดให้มีค่าเป็นนัล โดยเฉพาะอย่างยิ่งถ้าคีย์ภายนอกในเอนติตีลูกนั้น เป็นส่วนหนึ่งของคีย์หลักจะใช้แบบกำหนดเป็นนัลไม่ได้โดยเด็ดขาด ซึ่งหากมีความจำเป็นจริง ๆ ให้ใช้การกำหนดค่าโดยปริยายแทน ทั้งนี้เพราะระบบจัดการฐานข้อมูลแต่ละตัวจะให้ความหมายของค่านัลต่างกัน

นอกจากนี้กรณีรีเลชันชิปแบบซับไทป์-ซุปเปอร์ไทป์ ควรกำหนดกฎการแทรกเป็นแบบอัตโนมัติ และกฎการลบเป็นแบบต่อเนื่อง ทั้งนี้เพราะเอนติตีซับไทป์-ซุปเปอร์ไทป์ แสดงถึงสิ่งเดียวกัน และจากคำนิยามแล้ว ซับไทป์จะมีอยู่ไม่ได้ ถ้าซุปเปอร์ไทป์ไม่มีอยู่ ดังนั้นถ้าจะลบข้อมูลในเอนติตีแม่ ก็ต้องลบทั้งเอนติตีแม่และเอนติตีลูกด้วย และเวลาจะเพิ่มข้อมูลในเอนติตีแม่ก็ควรจะเพิ่มข้อมูลในเอนติตีลูก (หรือเอนติตีซับไทป์) ด้วยเช่นกัน ซึ่งในกรณีการแทรกข้อมูลจะต้องแน่ใจว่ายังไม่มีข้อมูลระเบียนใดในเอนติตีซับไทป์ มีคีย์ภายนอกที่เหมือนกัน อ้างไปยังเอนติตีซุปเปอร์ไทป์ เนื่องจากว่ารีเลชันชิปแบบนี้เป็นแบบ(1:1) จึงมีการเชื่อมด้วยคีย์เดียวกันได้เพียงหนึ่งระเบียนเท่านั้น

2.6 การกำหนดแอตตริบิวที่เหลือเพิ่มเติม

ขั้นตอนนี้เป็นการกำหนดแอตตริบิวอื่น ๆ ที่ไม่ใช่คีย์เพิ่มเติม บางครั้งเราไม่สามารถแยกได้ว่าสิ่งนั้นเป็นแอตตริบิวหรือเป็นเอนติตี ทั้งนี้ขึ้นอยู่กับการมองของผู้ออกแบบ เช่น

เราจะจัดว่า "สี" เป็นแอตตริบิว หรือเป็นเอนติตี

ในฐานะที่เป็นเอนติตี "สี" จะต้องประกอบด้วยส่วนขยายอื่น ๆ เช่น สูตรทางเคมีที่เป็นส่วนประกอบของสี ชื่อทางเคมี หรือชื่อทางการค้า

ในฐานะที่เป็นแอตตริบิว เราจะถือว่า "สี" เป็นแอตตริบิวได้ เมื่อไม่มีรายละเอียดใดปลีกย่อยไปกว่านี้ กรณีนี้ "สี" อาจเป็นแอตตริบิวหนึ่งของเอนติตีชิ้นส่วนก็ได้

แนวทางการกำหนดแอตตริบิวที่เหลือ ซึ่งไม่ใช่คีย์ มีดังต่อไปนี้

2.6.1 แอตตริบิวแต่ละตัวที่จะเพิ่มเข้ามานั้น ต้องขึ้นกับคีย์หลักของเอนติตีนั้นทั้งหมด

2.6.2 ให้กำหนดแอตตริบิวตัวนั้น ในเอนติตีระดับสูงสุดเท่าที่เป็นไปได้ นั่นคือต้องกำหนดแอตตริบิวส่วนเพิ่มนี้ ให้กับเอนติตีแม่ ไม่ใช่เอนติตีลูก

2.6.3 ถ้าแอตตริบิวดังกล่าว ซึ่งขึ้นกับคีย์หลักทั้งหมด แต่มีค่ามากกว่าหนึ่งค่า (multivalued) ให้แตกออกเป็นอีกหนึ่งเอนติตีที่มีความสัมพันธ์กับเอนติตีเดิมแบบ (1:N)

2.6.4 กำหนดชื่อ ความหมาย และรายละเอียดอื่น ๆ ที่เกี่ยวข้องลงในพจนานุกรมข้อมูล

2.6.5 ถ้ามีแอตตริบิวบางตัวที่ดูเหมือนจะขยายรีเลชันชิป แทนที่จะขยายเอนติตีแล้ว ให้สร้างรีเลชันชิปขึ้นใหม่ เป็นเอนติตีลูกของ 2 เอนติตีเดิม

2.6.6 หลีกเลี่ยงการแทนค่าของแอตตริบิวโดยการเข้ารหัส นอกเสียจากว่ารหัสนั้นเป็นที่รู้กันดี เหตุที่ต้องหลีกเลี่ยงการเข้ารหัสของแอตตริบิว คือ

1. รหัสไม่สามารถสื่อความหมายกับผู้ใช้ได้ ซึ่งจะขัดกับหลักการของการออกแบบโมเดลข้อมูลเชิงตรรกที่ว่า ทำให้มุมมองของผู้ใช้กระจ่างขึ้น

2. ทำให้ต้องมีเอนติตีใหม่เพิ่มขึ้นมาในโมเดล เพื่ออธิบายรหัสเหล่านั้น และต้องเป็นเอนติตีแม่ของทุกเอนติตีที่มีรหัสนั้นปรากฏอยู่

3. รหัสทำให้การสอบถามของผู้ใช้ยุ่งยาก เพราะผู้ใช้จะต้องรู้ถึงรหัสที่ตั้งไว้

2.6.7 หลีกเลี่ยงการใช้แอตตริบิวที่เป็นแฟลก (flag)

2.6.8 ถ้าต้องแทนแอตตริบิวด้วยการเข้ารหัสแล้ว ต้องกำหนดให้รหัสนั้นมีค่าแยกจากกันอย่างเด็ดขาด โดยไม่รวมความหมายมากกว่าหนึ่งอย่างไว้ในรหัสเดียวกัน พิจารณาจากตาราง 3.3 จะเห็นว่ามีรหัสซึ่งคาบเกี่ยวกันอยู่ ซึ่งจะทำให้การสอบถามข้อมูลซับซ้อน เช่นถ้าต้องการทราบถึงลูกจ้าง ที่ได้ค่าจ้างเป็นรายเดือน ก็ต้องค้นรหัสถึง 2 ตัวด้วยกัน คือ ME และ MM กรณีเช่นนี้ ควรแยกออกเป็น 2 เอนติตี ดังแสดงในตาราง 3.4

รหัส	ความหมาย
WE	ลูกจ้างที่เป็นวิศวกร ซึ่งรับเงินเดือนแบบรายสัปดาห์
WM	ลูกจ้างที่เป็นผู้จัดการ ซึ่งรับเงินเดือนแบบรายสัปดาห์
ME	ลูกจ้างที่เป็นวิศวกร ซึ่งรับเงินเดือนแบบรายเดือน
MM	ลูกจ้างที่เป็นผู้จัดการ ซึ่งรับเงินเดือนแบบรายเดือน

ตาราง 3.3 แสดงแอตตริบิวในรูปของการเข้ารหัส และความหมายของรหัส โดยที่รหัสมีค่าคาบเกี่ยวกัน

รหัส	ความหมาย
W	รับเงินเดือนแบบรายสัปดาห์
M	รับเงินเดือนแบบรายเดือน

รหัส	ความหมาย
E	ลูกจ้างที่เป็นวิศวกร
M	ลูกจ้างที่เป็นผู้จัดการ

ตาราง 3.4 แสดงแอตตริบิวของการเข้ารหัส และความหมายของรหัส โดยที่รหัสมีความเป็นอิสระจากกัน

2.6.9 ในกรณีที่แอตตริบิวใด ๆ สามารถหาได้จากสูตร หรือคำนวณได้จากแอตตริบิวอื่น จะเรียกว่าเป็นดีไรฟ์วแอตตริบิว (Derived Attribute) และให้ระบุตัว 'd' ไว้ที่ท้ายชื่อของแอตตริบิวนั้นในแผนภาพโมเดลข้อมูล และบันทึกสูตร หรือวิธีคำนวณไว้ในพจนานุกรมข้อมูลด้วย

2.6.10 กำหนดสัญลักษณ์พิเศษให้กับแอตตริบิว ที่เป็นตัวระบุชับไทป์ (Subtype Identifier) โดยใส่ตัวอักษร 's' ไว้ที่ท้ายชื่อแอตตริบิวนั้นในแผนภาพโมเดลข้อมูลเชิงตรรก

2.6.11 รวมแอตตริบิวที่มีความหมายร่วมกันระหว่างเอนติตีซุปเปอร์ไทป์กับเอนติตีซับไทป์ โดยให้รวมแอตตริบิวนั้นไว้เป็นส่วนหนึ่งของเอนติตีซุปเปอร์ไทป์

2.6.12 รวมเอนติตีที่มีคีย์หลักเหมือนกัน และแทนสิ่งเดียวกันเข้าไว้ด้วยกัน

2.6.13 รวมเอนติตีซับไทป์หลาย ๆ ซับไทป์ที่มีแอตตริบิว และรีเลชันชิป เหมือนกัน เข้าไว้เป็นหนึ่งซับไทป์

2.6.14 ถ้าเอนติตีซับไทป์ เป็นส่วนขยายซุปเปอร์ไทป์ ให้รวมเอนติตีซับไทป์เข้าไว้เป็นส่วนหนึ่งของเอนติตีซุปเปอร์ไทป์

2.6.15 เอนติตีที่ไม่มีแอตตริบิวอื่นที่ไม่ใช่คีย์ และถ้าเอนติตีนั้นมีเอนติตีลูกด้วย ให้ยุบรวมกันกับเอนติตีลูกที่มี

2.7 การตรวจสอบด้วยกฎของนอร์มัลไลซ์เซชัน

นอร์มัลไลซ์เซชัน เป็นการวิเคราะห์ และแยกโครงสร้างของข้อมูลให้เป็นเซตของความสัมพันธ์ที่มีคุณสมบัติตามที่ต้องการ การทำนอมัลไลซ์เซชันไม่มีผลทำให้สารสนเทศหายไป หรือเกิดสารสนเทศใหม่ ที่ไม่จริงขึ้นมาแต่อย่างใด โมเดลที่ผ่านกระบวนการนอมัลไลซ์เซชันแล้ว จะเป็นโมเดลข้อมูลเชิงตรรกที่ดี ทำให้การออกแบบฐานข้อมูลเป็นไปได้สะดวก ถูกต้อง คงที่ ไม่ซับซ้อน และมีเสถียรภาพ

ประโยชน์ของการทำนอมัลไลซ์เซชัน คือ

1. ลดเนื้อที่ที่ใช้ในการเก็บข้อมูล
2. ลดความผิดพลาด ความไม่ตรงกันของข้อมูลในฐานข้อมูล
3. ลดความเป็นไปได้ของการแก้ไข หรือลบข้อมูลที่ก่อให้เกิดความผิดพลาด
4. เพิ่มเสถียรภาพของโครงสร้างฐานข้อมูล

การทำนอมัลไลซ์เซชัน ประกอบด้วยขั้นตอนต่าง ๆ ดังนี้

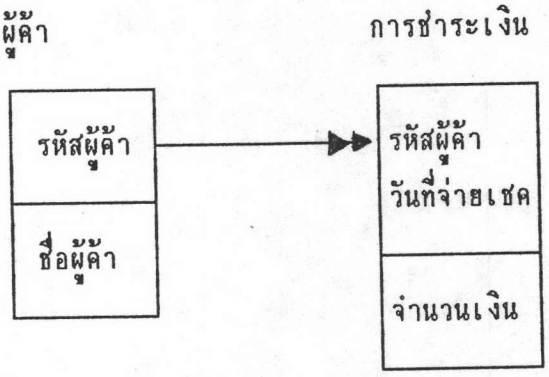
2.7.1 นอร์มัลฟอร์มระดับที่ 1 (First normal form : 1NF) เป็นการกำจัดแอตตริบิวซึ่งมีค่าซ้ำ ๆ กัน (Repeating group) หรือมีหลายค่า (Multivalued) ออก ทำให้มีเพียงแอตตริบิวที่เป็นค่าเดียวโดด ๆ ไม่เป็นกลุ่ม หรืออาจไม่มีค่าเลยก็ได้ (null) ยกเว้นแอตตริบิวที่เป็นคีย์หลักเท่านั้น โดยสร้างเป็นเอนติตีขึ้นใหม่ ซึ่งเป็นเอนติตีลูกของเอนติตีเดิม

จากตารางที่ 3.5 จะเห็นว่าเอนติตีการชำระเงิน เป็นเอนติตีที่ไม่อยู่ใน 1NF เพราะประกอบด้วยแอตตริบิววันที่จ่ายเชคและจำนวนเงิน ซ้ำ ๆ กัน เพื่อให้เอนติตีผู้ค้าอยู่ใน 1NF เราจะต้องกำจัดแอตตริบิวที่มีค่าซ้ำกันออก โดยสร้างเป็นเอนติตีใหม่ชื่อว่า เอนติตีการชำระเงิน ซึ่งจะมีแผนภาพดังรูปที่ 3.14 และมีตัวอย่างข้อมูลดังตารางที่ 3.6

ผู้ค้า

รหัสผู้ค้า	ชื่อผู้ค้า	วันที่จ่ายเช็ค	จำนวนเงิน	วันที่จ่ายเช็ค	จำนวนเงิน	วันที่จ่ายเช็ค	จำนวนเงิน
10001	ศูนย์หนังสือจุฬาฯ	15/01/93	3500	15/02/93	5500	15/03/93	2300
10002	ดวงกมล	20/01/93	2250	20/02/93	1750	20/03/93	3950
10005	สาธิตา	20/01/93	1750	20/02/93	2750	20/03/93	1550

ตารางที่ 3.5 แสดงข้อมูลของเอนติตีผู้ค้า ซึ่งไม่นอมัลไลซ์



รูปที่ 3.14 แสดงรีเลชันชิปของเอนติตีที่อยู่ใน 1NF

ผู้ค้า

รหัสผู้ค้า	ชื่อผู้ค้า
10001	ศูนย์หนังสือจุฬาฯ
10002	ดวงกมล
10005	สาธิตา

การชำระเงิน

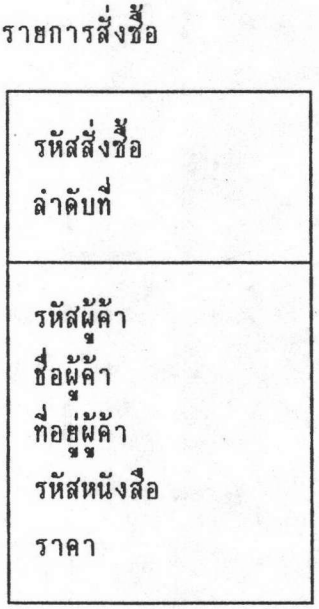
รหัสผู้ค้า	วันที่จ่ายเช็ค	จำนวนเงิน
10001	15/01/93	3500
10001	15/02/93	5500
10001	15/03/93	2300
10002	20/01/93	2250
10002	20/02/93	1750
10002	20/03/93	3950
10005	20/01/93	1750
10005	20/02/93	2750
10005	20/03/93	1550

ตารางที่ 3.6 แสดงตัวอย่างข้อมูลของเอนติตีในรูปที่ 3.14

วัตถุประสงค์ของการทำนอมัลไลซ์ขั้นที่ 1 คือ

1. ทำให้โมเดลข้อมูลเชิงตรรกง่ายขึ้น
2. เพื่อให้สามารถทำนอมัลไลซ์เชชันขั้นต่อไปได้
3. ทำให้สามารถแปลงโมเดลข้อมูลเชิงตรรก ไปเป็นโครงสร้างของฐานข้อมูลแบบใดก็ได้ โดยไม่สูญเสียหน้าที่ต่าง ๆ ของโครงสร้างฐานข้อมูล

2.7.2 นอมัลไลซ์ขั้นที่ 2 (Second Normal Form หรือ 2NF) ทุกๆ แอตตริบิวที่อยู่ภายในเอนติตีเดียวกัน จะต้องขึ้นอยู่กับคีย์หลักเท่านั้น และเอนติตีนี้จะต้องอยู่ใน 1NF ด้วย วิธีทำให้อยู่ใน 2NF คือให้แยกเอนติตีที่มีแอตตริบิวที่ขึ้นกับส่วนหนึ่งของคีย์ออกเป็นเอนติตีใหม่ ซึ่งจะสามารถตัดปัญหาความซ้ำซ้อนในบางจุดลงได้ แต่ก็ยังคงมีความซ้ำซ้อนอยู่ พิจารณาตัวอย่างจากแผนภาพรูปที่ 3.15 และตารางที่ 3.7 ซึ่งแสดงเอนติตีของรายการสั่งซื้อ



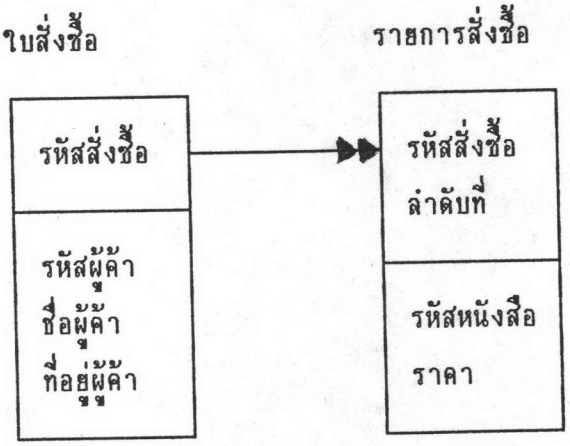
รูปที่ 3.15 แสดงแผนภาพโมเดลข้อมูลของรายการสั่งซื้อ

รายการสั่งซื้อ

รหัสสั่งซื้อ	ลำดับที่	รหัสผู้ค้า	ชื่อผู้ค้า	ที่อยู่ผู้ค้า	รหัสหนังสือ	ราคา
00001	1	10001	ศูนย์หนังสือจุฬาฯ	กรุงเทพ	B100-001	750
00001	2	10001	ศูนย์หนังสือจุฬาฯ	กรุงเทพ	B100-101	1250
00018	1	10002	ดวงกมล	กรุงเทพ	A201-900	550
00018	2	10002	ดวงกมล	กรุงเทพ	B044-330	580
00025	1	10005	สาธิตา	กรุงเทพ	A200-930	1550

ตารางที่ 3.7 แสดงตัวอย่างข้อมูลจากแผนภาพ 3.15

จากตัวอย่างจะเห็นว่าชื่อร้านค้า และที่อยู่ร้านค้า ขึ้นอยู่กับรหัสการสั่งซื้อเท่านั้น ไม่ได้ขึ้นกับคีย์หลักทั้งหมด ดังนั้นจึงต้องกำจัดแอตตริบิวดังกล่าวออกไปเป็นเอนติตีใหม่ คือ เอนติตีใบสั่งซื้อ ดังแผนภาพรูปที่ 3.16 และตารางที่ 3.8



รูปที่ 3.16 แสดงแผนภาพโมเดลข้อมูลของรายการสั่งซื้อ ซึ่งอยู่ใน 2NF

ใบสั่งซื้อ

รหัสสั่งซื้อ	รหัสผู้ค้า	ชื่อผู้ค้า	ที่อยู่ผู้ค้า
00001	10001	ศูนย์หนังสือจุฬาฯ	กรุงเทพ
00018	10002	ดวงกมล	กรุงเทพ
00025	10005	สาธิตา	กรุงเทพ

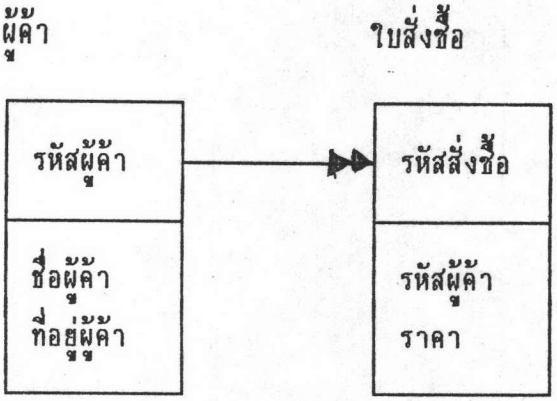
รายการสั่งซื้อ

รหัสสั่งซื้อ	ลำดับที่	รหัสหนังสือ	ราคา
00001	1	B100-001	750
00001	2	B100-101	1250
00018	1	A201-900	550
00018	2	B044-330	580
00025	1	A200-930	1550

ตารางที่ 3.8 แสดงตัวอย่างข้อมูลจากแผนภาพ 3.16

2.7.3 นอร์มัลไลซ์ขั้นที่ 3 (Third normal form หรือ 3NF) เป็นการกำจัดแอตตริบิวที่ไม่เพียงขึ้นกับคีย์หลักเท่านั้น แต่ยังขึ้นกับแอตตริบิวอื่นที่ไม่ใช่คีย์ ออกมาเป็นเอนติตีใหม่ โดยที่เอนติตีเดิมนั้นยังคงอยู่ใน 2NF

พิจารณาจากรูป 3.16 จะเห็นว่าแอตตริบิวชื่อ และที่อยู่ผู้ค้า นอกจากจะขึ้นอยู่กับรหัสสั่งซื้อแล้ว ยังขึ้นอยู่กับรหัสผู้ค้าอีก ดังนั้นการทำ 3NF จึงนำเอาที่อยู่ผู้ค้าออกมาอยู่ในเอนติตีผู้ค้า ดังรูปที่ 3.17



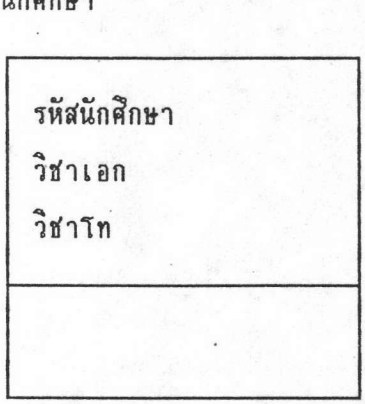
รูปที่ 3.17 แสดงแผนภาพโมเดลข้อมูล ซึ่งอยู่ใน 3NF

2.7.4 บอยซ์/คอดด์นอร์มัลไลซ์ (Boyce/Codd normal form หรือ BCNF) เอนติตีใด ๆ จะอยู่ใน BCNF ก็ต่อเมื่อทุกแอตตริบิวของเอนติตีนั้น ถูกกำหนดโดยแคนดิเดทคีย์ (คีย์หลัก และคีย์รอง) ทุกตัว ไม่ใช่เพียงส่วนใดส่วนหนึ่งของแคนดิเดทคีย์เท่านั้น

2.7.5 นอมัลไลซ์ขั้นที่ 4 (Fourth Normal form หรือ 4NF) เป็นการกำจัดแอตตริบิว ที่เป็นส่วนหนึ่งของคีย์หลัก ซึ่งมีค่าได้หลายค่า และเป็นแอตตริบิวที่ไม่ขึ้นต่อกัน ออกเป็นเอนติตีใหม่ 2 เอนติตี โดยคงเอนติตีเดิมไว้ หากเอนติตีนั้นมีแอตตริบิวอื่น ที่ไม่ใช่คีย์อยู่ จากรูป 3.18 และตารางที่ 3.9 เอนติตีนักศึกษา ซึ่งประกอบไปด้วยแอตตริบิวรหัสนักศึกษา วิชาเอก และวิชาโท จะเห็นว่านักศึกษาคนหนึ่ง สามารถเลือกเรียนวิชาเอกได้หลายวิชา และสามารถเลือกเรียนวิชาโทได้หลายวิชาเช่นกัน โดยที่วิชาเอก และวิชาโทเป็นอิสระซึ่งกันและกัน

ปัญหาในที่นี้ก็คือ จะต้องมีวิชาโทซ้ำกันมากกว่าหนึ่งครั้งสำหรับวิชาเอกแต่ละวิชาของนักศึกษาหนึ่งคน

วิธีแก้ปัญหาคือ สร้างให้เป็นเอนติตีใหม่ 2 เอนติตี ดังรูปที่ 3.19 และต้องคงเอนติตีเดิมไว้ หากเอนติตีประกอบด้วยแอตตริบิวอื่น ๆ ที่ไม่ใช่คีย์



รูปที่ 3.18 แสดงเอนติตีนักศึกษา

นักศึกษา

รหัสนักศึกษา	วิชาเอก	วิชาโท
35401	การเงิน	อังกฤษ
35401	การธนาคาร	จิตวิทยา
35401	การบริหาร	สถิติ

ตารางที่ 3.9 แสดงตัวอย่างข้อมูลของเอนติตีนักศึกษา ตามรูป 3.18

วิชาเอกที่นักศึกษาเลือก

รหัสนักศึกษา วิชาเอก

วิชาโทที่นักศึกษาเลือก

รหัสนักศึกษา วิชาโท

รูปที่ 3.19 แสดงเอนติตีนักศึกษา ซึ่งได้จากการทำ 4NF

2.7.6 นอมัลไลซ์ขั้นที่ 5 (Fifth Normal Form หรือ 5NF) เป็นการนำเอนติตีที่ได้จากการทำ 4NF มากำจัดแอตตริบิว ที่เป็นส่วนประกอบของคีย์หลัก ซึ่งขึ้นต่อกันเป็นวง (cyclic dependencies) โดยสร้างเป็นเอนติตีใหม่ ซึ่งจะได้เอนติตีตั้งแต่ 3 เอนติตีขึ้นไป

ข้อควรระลึก คือ เมื่อได้ผ่านกระบวนการนอมัลไลซ์เซชันแล้ว ห้ามแตกเอนติตีดังกล่าวออกเป็นเอนติตีย่อย ๆ ลงไปอีก

2.8 การกำหนดโดเมน

โดเมน หมายถึง เซตของค่าที่เป็นไปได้ของคุณสมบัติที่กำหนดให้กับแอตตริบิวใด ๆ คุณสมบัติดังกล่าวประกอบไปด้วย

1. ชนิดของข้อมูล (data type) เช่น จำนวนเต็ม (integer) วันที่

(date) ตัวอักษร (character)

2. ความยาว (length) เช่น 7 หลัก, 60 ตัวอักษร

3. รูปแบบข้อมูล (format) เช่น dd/mm/yy(วันที่), ccc-cccc (เบอร์โทรศัพท์)

4. ค่าที่อนุญาต (allowable value) เช่น เป็นได้เฉพาะวันเสาร์

5. ช่วงของข้อมูลหรือข้อกำหนดอื่นๆ (constraints, range) เช่น 1-2 เดือน

6. ความหมาย (meaning) อธิบายความหมายของแอตตริบิวนั้นว่าคืออะไร

7. ความเป็นหนึ่งเดียว (uniqueness) ต้องมีค่าเป็นหนึ่งเดียว

8. การเป็นนัล (null support) อนุญาตให้เป็นนัลได้หรือไม่

9. ค่าโดยปริยาย (default value) กำหนดให้มีค่าเป็น 0

คุณสมบัติบางประการ จะเป็นอิสระจากเอนติตี ที่แอตตริบิวนั้นปรากฏอยู่ เช่น ชนิดของข้อมูล และความยาว ของแอตตริบิวหนึ่ง ควรจะเหมือนกันตลอดทั้งโมเดล แต่มีคุณสมบัติบางประการ ที่จะขึ้นอยู่กับเอนติตีที่แอตตริบิวปรากฏอยู่ เช่น คุณสมบัติของความเป็นหนึ่ง โดยหากแอตตริบิวนั้น เป็นส่วนหนึ่งของคีย์หลัก แอตตริบิวดังกล่าวจะต้องมีความเป็นหนึ่ง แต่หากแอตตริบิวนั้นไปปรากฏอยู่ที่อื่นในโมเดล ก็ไม่จำเป็นต้องมีความเป็นหนึ่ง

ขั้นตอนการกำหนดโดเมน คือ

2.8.1 กำหนดโดเมนของคีย์หลัก โดยพิจารณาตามกฎต่าง ๆ ดังนี้

1. คีย์หลัก ต้องมีความเป็นหนึ่ง

2. กรณีคีย์หลักเป็นคีย์ประกอบ แอตตริบิวแต่ละตัวที่ประกอบเป็นคีย์หลัก ไม่จำเป็นต้องมีค่าเป็นหนึ่ง

3. คีย์หลัก หรือแอตตริบิวที่เป็นส่วนประกอบของคีย์หลัก จะมีค่าเป็นนัลไม่ได้

4. คีย์หลัก และแอตตริบิวที่เป็นส่วนประกอบของคีย์หลัก จะสามารถรับค่าโดยปริยายได้ แต่ต้องคงไว้ซึ่งความเป็นหนึ่ง

2.8.2 กำหนดโดเมนของคีย์รอง จะมีกฎเหมือนคีย์หลัก ต่างกันที่คีย์รองสามารถเป็นนัลได้ ดังนี้

1. คีย์รอง ต้องมีความเป็นหนึ่ง

2. แอตตริบิวที่เป็นส่วนประกอบเป็นคีย์รอง ไม่จำเป็นต้องมีค่าเป็นหนึ่ง

3. คีย์รอง หรือแอตตริบิวที่เป็นส่วนประกอบของคีย์รองอาจมีค่าเป็นนัลได้

4. คีย์รอง และแอตตริบิวที่เป็นส่วนประกอบของคีย์หลัก จะสามารถรับค่าโดยปริยายได้ แต่ต้องคงไว้ซึ่งความเป็นหนึ่ง

2.8.3 กำหนดโดเมนของคีย์ภายนอก โดยพิจารณาตามกฎต่าง ๆ ดังนี้

1. ชนิดของข้อมูล ความยาว และรูปแบบของคีย์ภายนอกต้องเหมือนกับชนิดของข้อมูล ความยาว และรูปแบบ เมื่อเป็นคีย์หลักในเอนติตีแม่

2. คุณสมบัติความเป็นหนึ่งของคีย์ภายนอก จะขึ้นอยู่กับชนิดของความสัมพันธ์ โดยถ้าความสัมพันธ์เป็นแบบหนึ่งต่อหนึ่ง คีย์ภายนอกก็จะมีคุณสมบัติความเป็นหนึ่ง แต่ถ้าความสัมพันธ์ เป็นแบบหนึ่งต่อหลายแล้ว คีย์ภายนอกก็จะไม่มีความเป็นหนึ่ง

2.8.4 กำหนดโดเมนของดิไรฟ์แอตตริบิว โดยพิจารณาตามกฎต่าง ๆ ดังนี้

1. ให้กำหนดอัลกอริทึม (Algorithm) ของดิไรฟ์แอตตริบิวไว้ที่คุณสมบัติของโดเมนตรงค่าที่เป็นไปได้

2. ชนิดของข้อมูลของดิไรฟ์แอตตริบิว จะต้องเป็นชนิดเดียวกับแอตตริบิวที่เป็นแหล่งที่มาของแอตตริบิวดิไรฟ์แอตตริบิวนั้น มิฉะนั้นอาจถูกกำหนดโดยอัลกอริทึม

3. ความหมายของดิไรฟ์แอตตริบิว ถูกกำหนดจากอัลกอริทึม และจากความหมายของแอตตริบิวที่เป็นแหล่งที่มาของดิไรฟ์แอตตริบิว

2.8.5 กำหนดโดเมนของคีย์หลักของเอนติตีที่เป็นซับไทป์ โดยโดเมนนี้จะต้องเป็นซับเซตของโดเมนของคีย์หลักของเอนติตี ที่เป็นซุบเปอร์ไทป์ ดังนี้

1. ชนิดของข้อมูล ความยาว และรูปแบบจะต้องเหมือนกับคีย์หลักของเอนติตีที่เป็นซุปเปอร์ไทป์

2. ค่าที่เป็นไปได้จะขึ้นอยู่กับตัวระบุซับไทป์

3. ความหมายจะต้องเหมือนกับคีย์หลักของซุปเปอร์ไทป์ แต่ขึ้นอยู่กับตัวระบุซับไทป์

4. ต้องมีคุณสมบัติความเป็นหนึ่ง

5. ส่วนประกอบของคีย์หลักไม่ต้องมีคุณสมบัติความเป็นหนึ่ง

6. จะมีค่าเป็นนัลไม่ได้

7. สามารถกำหนดให้มีค่าเป็นค่าโดยปริยายได้

2.8.6 กำหนดโดเมนให้กับแอตตริบิวต่าง ๆ ที่ไม่ใช่คีย์ ซึ่งปรากฏอยู่ในโมเดลข้อมูล

2.8.7 บันทึกโดเมนที่กำหนดให้กับแอตตริบิวต่าง ๆ ลงในพจนานุกรมข้อมูล

2.9 กำหนดทริกเกอร์ดำเนินการ

ทริกเกอร์ดำเนินการ เป็นกฎที่ใช้ควบคุม และตรวจสอบความถูกต้อง ของการใช้คำสั่งเพื่อการแทรก ลบ ปรับปรุง และดึงข้อมูล รวมทั้งผลลัพธ์ของการทำคำสั่งเหล่านี้ ซึ่งมีผลกระทบกับเอนติตีอื่น ๆ หรือแอตตริบิวอื่น ๆ ในเอนติตีเดียวกัน ทริกเกอร์ดำเนินการจะทำให้ค่าของแอตตริบิวมีความเป็นบูรณภาพ และมีความคงที่ เมื่อกำหนดทริกเกอร์ดำเนินการต่าง ๆ แล้วให้เก็บลงในพจนานุกรมข้อมูล โดยมีรูปแบบดังนี้

1. เหตุการณ์ที่ก่อให้เกิดการทำงานของทริกเกอร์ เช่น การเพิ่ม แก้ไข ลบ หรือดึงข้อมูล

2. ตัวถูกกระทำจากเหตุการณ์ เช่น ชื่อของเอนติตี และหรือแอตตริบิวที่จะถูกกระทำ

3. เงื่อนไขที่ทำให้ทริกเกอร์ดำเนินการทำงาน

4. การกระทำอื่น ๆ ที่เกิดขึ้นจากการทำงานของทริกเกอร์ดำเนินการ เช่น การปฏิเสธข้อมูล (reject) เป็นต้น

นอกจากนี้ยังต้องกำหนดทริกเกอร์ดำเนินการสำหรับทุกแอตตริบิว ที่เป็นแหล่งที่มาของดีไรฟ์แอตตริบิว ทริกเกอร์ดำเนินการสำหรับซับไทป์-ซุปเปอร์ไทป์ เช่น ถ้าซับไทป์ถูกลบ ต้องลบซุปเปอร์ไทป์ด้วย และต้องกำหนดเงื่อนไขภายใต้เรื่องของเวลา เช่น ถ้าระเบียนข้อมูลนี้มีอายุเกิน 1 ปีให้ทำการลบทิ้ง

2.10 การเชื่อมมุมมองผู้ใช้ทุกคนเข้าด้วยกัน

ขั้นตอนต่าง ๆ ที่ผ่านมา เป็นการพัฒนารายละเอียดตามความต้องการสารสนเทศของผู้ใช้ หรือหน้าที่ส่วนหนึ่งส่วนใดของระบบเท่านั้น ซึ่งการรวบรวมความต้องการต่างๆ จากผู้ใช้หลายคน หรือหลายมุมมอง อาจก่อให้เกิดความเหลื่อมล้ำ หรือซ้ำซ้อนขึ้นได้ ขั้นตอนต่อไปจะเป็นการปรับแต่งมุมมองต่าง ๆ เข้าด้วยกัน เพื่อขจัดส่วนที่ซ้ำซ้อน และแก้ปัญหาความไม่ตรงกันของข้อมูล ซึ่งเป็นขั้นตอนที่ต้องใช้ทั้งประสบการณ์และทักษะในการวิเคราะห์มาก และนับว่าเป็นส่วนที่สำคัญที่สุด

จุดประสงค์ของการเชื่อมมุมมองของผู้ใช้ทุกคนเข้าด้วยกัน คือ

1. เพื่อเป็นตัวแทนของมุมมองต่าง ๆ ได้อย่างแม่นยำ

2. เพื่อกำจัดความซ้ำซ้อน

3. เพื่อแก้ปัญหาการไม่คงที่ ที่เกิดขึ้นระหว่างมุมมองต่าง ๆ

4. เพื่อเพิ่มรีเลชันชิปใหม่ ๆ ที่เกิดขึ้นมาระหว่างมุมมองต่าง ๆ

การในการเชื่อมมุมมองของผู้ใช้ มีวิธีการดังนี้

2.10.1 รวบรวมเอนติตี และกฎธุรกิจที่เกี่ยวข้อง ซึ่งแนวทางในการพิจารณามีดังนี้

1. นำเอนติตีที่มีคีย์หลักเหมือนกัน และมีโดเมนของคีย์หลักที่เทียบเท่ากันมาไว้รวมกัน เป็นเอนติตีเดียวที่มีแอตตริบิวรวมของสองเอนติตีเดิม โดยตัดแอตตริบิวที่ซ้ำกันออก แต่การรวมมุมมองนี้ มีข้อแม้ว่า จะต้องไม่รวมเอนติตีซับไทป์ และเอนติตีซุปเปอร์ไทป์ที่เป็นมุมมองของผู้ใช้เดียวกัน แม้ว่าเอนติตีนั้นจะมีคีย์หลักเหมือนกันก็ตาม

2. ให้สร้างรีเลชันชิปแบบซับไทป์-ซุปเปอร์ไทป์ขึ้น ระหว่างเอนติตีสองอันที่มีคีย์หลักเหมือนกัน และค่าที่เป็นไปได้ของคีย์หลักของเอนติตีที่จะเป็นซับไทป์ เป็นซับเซตของอีกเอนติตีหนึ่งที่เป็นซุปเปอร์ไทป์ โดยตัดแอตตริบิวที่มีแล้วในเอนติตีซุปเปอร์ไทป์ออกจากเอนติตีที่เป็นซับไทป์

3. ถ้าเอนติตีสองตัวมีคีย์หลักเหมือนกัน และค่าที่เป็นไปได้ของคีย์หลักมีค่าเท่าเทียมกัน แม้ว่าจะมีเงื่อนไขของค่าที่เป็นไปได้ต่างกันก็ตาม ให้กำหนดเอนติตีซุปเปอร์ไทป์ร่วมขึ้นมาอันหนึ่ง ซึ่งสัมพันธ์กันระหว่างสองเอนติตีเดิม

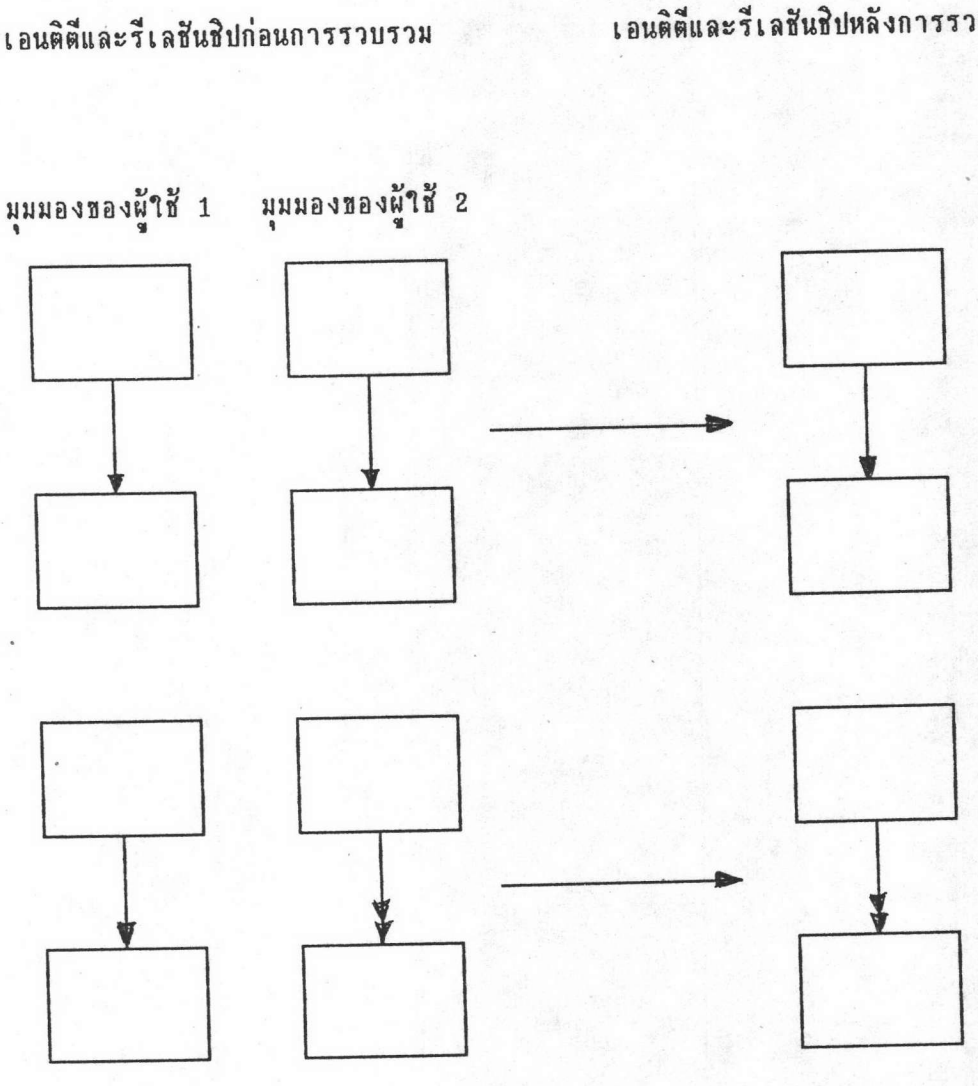
4. รวมเอนติตีสองตัว ที่คีย์หลักของเอนติตีตัวใดตัวหนึ่งเป็นแคนดิเดทคีย์ของเอนติตีอีกตัวหนึ่ง ขึ้นเป็นเอนติตีใหม่ มีแอตตริบิวอื่น ๆ ซึ่งรวมระหว่างแอตตริบิวของสองเอนติตีเดิม โดยตัดแอตตริบิวที่ซ้ำซ้อนออกเสีย และให้เลือกว่าจะนำคีย์หลักของเอนติตีตัวใดมาเป็นคีย์หลักของเอนติตีใหม่ ส่วนคีย์หลักของเอนติตีที่ไม่ได้ถูกเลือก จะกลายมาเป็นคีย์รองของเอนติตีใหม่

5. การรวมเอนติตีใด ๆ ก็ตามต้องไม่มีผลไปเปลี่ยนแปลงเอนติตีอื่นที่ไม่ได้เกี่ยวข้อง

6. ให้คงกฎธุรกิจที่เกี่ยวกับแคนดิเดทคีย์ไว้ ยกเว้นในกรณีที่คีย์หลักของเอนติตีเดิม กลายมาเป็นคีย์รองของเอนติตีใหม่ ดังนั้นแอตตริบิวนี้สามารถจะมีค่าเป็นนัลได้

2.10.2 รวบรวมรีเลชันชิป และกฎธุรกิจที่เกี่ยวข้อง แนวทางในการพิจารณา คือ

1. หลังจากที่ได้รวบรวมเอนติตีเข้าไว้ด้วยกันแล้ว รีเลชันชิปของเอนติตีเหล่านั้นก็จะถูกรวบรวมด้วย ถ้ารีเลชันชิปของเอนติตีเหล่านั้น มีความหมายเหมือนกัน แต่สัดส่วนของความสัมพันธ์ จะขึ้นอยู่กับความสัมพันธ์เดิม เช่นจากรูปที่ 3.20 หากความสัมพันธ์เดิมเป็นแบบ (1:1) หรือ (1:N) ทั้งคู่ ความสัมพันธ์ใหม่ก็จะเป็น (1:1) หรือ (1:N) ด้วย แต่หาก ความสัมพันธ์เดิมเป็น (1:1) กับ (1:N) แล้ว ความสัมพันธ์ใหม่ จะเป็น (1:N) ซึ่งถ้าความสัมพันธ์ใหม่เป็น (M:N) จะต้องทำการแตกให้เป็นรีเลชันชิปแบบ (1:N) สองอัน



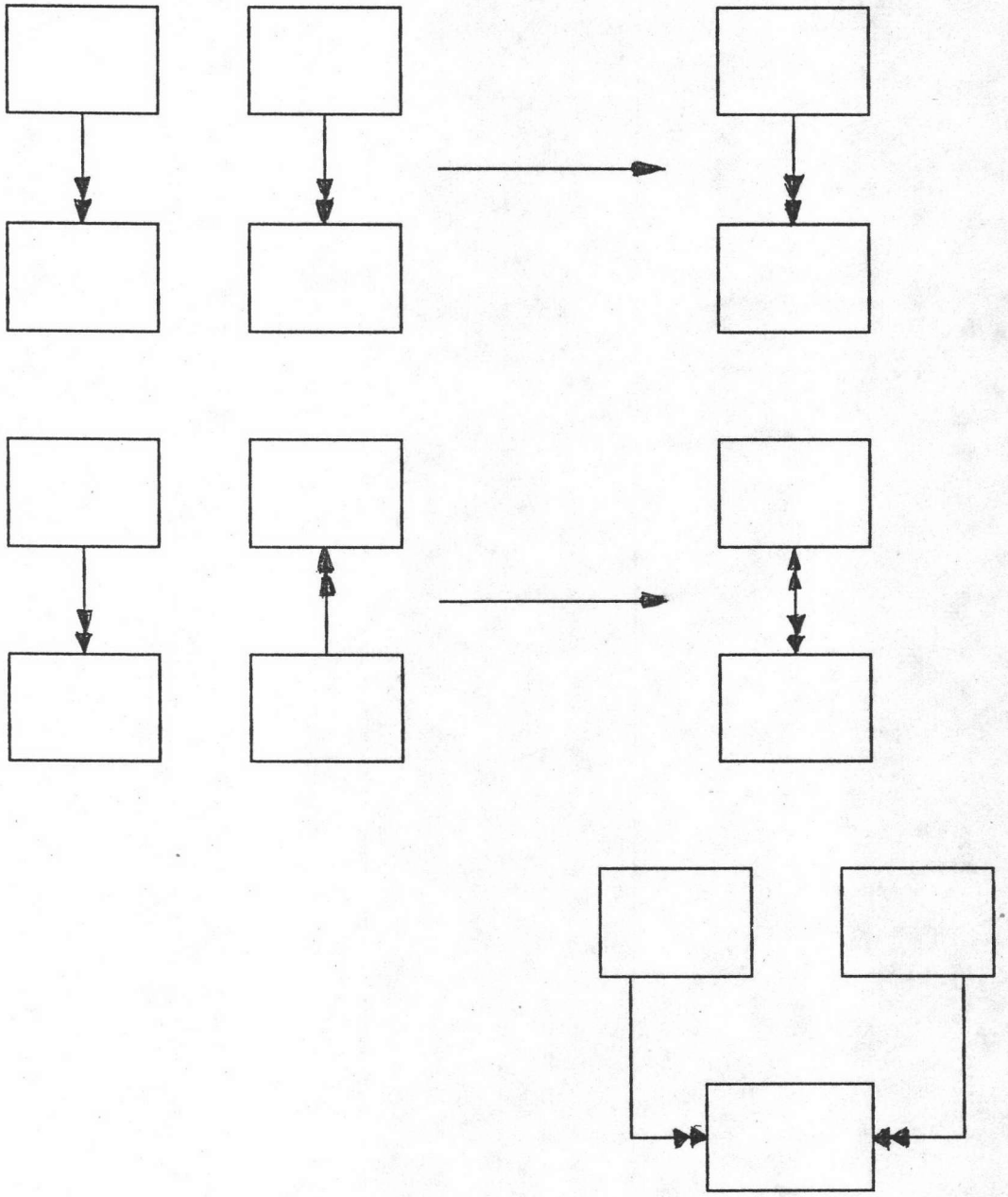
รูปที่ 3.20 แสดงเอนติตี และรีเลชันชิปก่อน และหลังการรวบรวม

เอนติตีและรีเลชันชิปก่อนการรวบรวม

เอนติตีและรีเลชันชิปหลังการรวบรวม

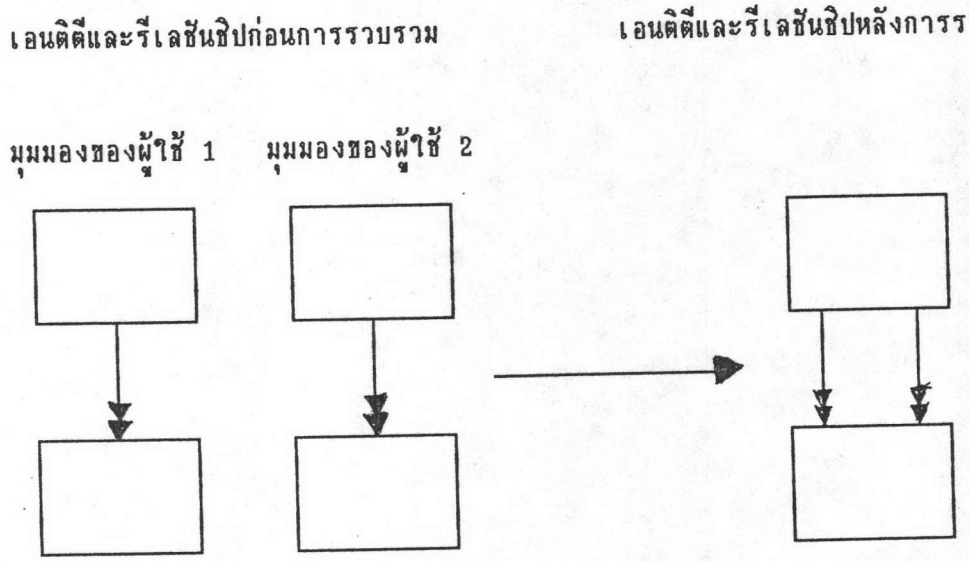
มุมมองของผู้ใช้ 1

มุมมองของผู้ใช้ 2



รูปที่ 3.20(ต่อ) แสดงเอนติตี และรีเลชันชิปก่อน และหลังการรวบรวม

แต่ถ้าความสัมพันธ์ของมุมมองของผู้ใช้นั้น ไม่เหมือนกันแล้ว ก็ไม่สามารถรวมความสัมพันธ์เข้าด้วยกันได้ ดังตัวอย่างในรูป 3.21



รูปที่ 3.21 แสดงเอนติตีและรีเลชันชิปก่อน และหลังการรวบรวม โดยที่มุมมองของผู้ใช้ไม่เหมือนกัน

2. การรวมรีเลชันชิปใด ๆ ต้องไม่ไปกระทบกับรีเลชันชิปอื่นที่ไม่ต้องการการเปลี่ยนแปลง และต้องกำจัดรีเลชันที่ซ้ำซ้อน

3. พิจารณาว่ามีรีเลชันชิปใด ขาดหายไปบ้าง ถ้ามีก็ให้ทำการเพิ่มขึ้นใหม่เพื่อความเหมาะสม

4. จากการรวมเอนติตีที่มีคีย์หลักเป็นแคนดิเดทคีย์ของเอนติตีอีกตัว ให้ตรวจสอบคีย์ภายนอกของเอนติตีที่รวมแล้วว่า อ้างถึงคีย์รองของเอนติตีแม่หรือไม่ ถ้าใช่ให้แก้แอตตริบิวนั้นใหม่ โดยให้ใช้คีย์หลักของเอนติตีแม่แทน

5. เมื่อรวมมุมมองต่าง ๆ แล้ว ให้กำหนดคีย์ของกฎธุรกิจสำหรับรีเลชันชิปใหม่ด้วย โดยต้องคงกฎธุรกิจของรีเลชันเดิมไว้ และกำหนดใหม่สำหรับรีเลชันชิปใหม่

2.10.3 รวบรวมแอตตริบิว และกฎธุรกิจที่เกี่ยวข้อง แนวทางในการพิจารณา คือ

1. ทำการรวมแอตตริบิวที่มีความหมายเหมือนกัน ภายในเอนติตีเดียวกัน และรวมค่าที่เป็นไปได้และทริกเกอร์ดำเนินการเข้าด้วยกันด้วย

และให้พิจารณาค่าที่เป็นไปได้ของแอตตริบิวอื่นด้วยว่าเปลี่ยนไปหรือไม่

2. หลังจากรวมเอนติตีแล้วให้พิจารณาตัดแอตตริบิวที่เป็นดีไรฟ์หรือแฟลกซ์ที่ไม่จำเป็นทิ้งเสีย

3. หลังจากได้รวม ตัด หรือเพิ่มรีเลชันชิปแล้ว ให้ทำการนอร์มัลไลซ์อีกครั้งเพื่อตัดสิ่งที่ซ้ำซ้อนออก

2.11 การรวมเข้ากับโมเดลข้อมูลที่มีอยู่แล้ว

ขั้นตอนนี้เป็นการรวบรวมโมเดลข้อมูลเชิงตรรกที่ได้ใหม่ กับของเดิมที่มีอยู่แล้ว โดยให้พัฒนาโมเดลใหม่ควบคู่ไปกับการพิจารณากฎเกณฑ์ข้อบังคับของเดิม ซึ่งอาจมีใช้เอนติตีหรือรีเลชันชิปร่วมกับของเดิม และมีการกำหนดเอนติตีขึ้นมาใหม่ด้วย

2.12 การวิเคราะห์เสถียรภาพและการเติบโตในอนาคต

ขั้นตอนต่าง ๆ ที่กล่าวมาแล้วนั้น เป็นการพิจารณาข้อมูล และความต้องการที่มีในปัจจุบันเท่านั้น แต่ในขั้นนี้จะเป็นการพิจารณาถึงสิ่งที่อาจเกิดขึ้นหรือเป็นได้ได้ในอนาคต เช่น

1. อาจมีเอนติตีหรือรีเลชันชิปใหม่เกิดขึ้น ทำให้ต้องเพิ่มคีย์ภายนอกในเอนติตีของเดิม

2. สัดส่วนความสัมพันธ์อาจเปลี่ยนแปลงไป เช่นจากเดิมความสัมพันธ์เป็นแบบ (1:N) อาจกลายเป็น (M:N)

3. คีย์หลักอาจเปลี่ยนไปเนื่องจากของเดิมไม่เป็นหนึ่งเดียวแล้ว กรณีนี้จะส่งผลกระทบต่อคีย์ภายนอกของเอนติตีที่สัมพันธ์กัน

4. อาจมีแอตตริบิวใหม่เพิ่มขึ้นหรือแอตตริบิวเดิมที่มีอยู่ อาจถูกกำจัดออกไป

5. ขนาดของเอนติตี (ปริมาณข้อมูลในเอนติตี) เพิ่มขึ้น หรือลดลง

สิ่งเหล่านี้เมื่อเราพิจารณาแล้วอาจทำการดัดแปลงโมเดลไว้เพื่อรองรับหรือจดบันทึกเก็บไว้ก่อนเฉย ๆ ก็ได้

เสถียรภาพ และการเติบโต ของโมเดลข้อมูลเชิงตรรก ขึ้นอยู่กับความสามารถในการคาดคะเนถึงการเปลี่ยนแปลงความต้องการในอนาคต

การออกแบบฐานข้อมูลแบบรีเลชันนัล [17],[19]

การออกแบบฐานข้อมูลแบบรีเลชันนัล เป็นกระบวนการในการแปลโมเดลข้อมูลเชิงตรรก ให้อยู่ในรูปฐานข้อมูลรีเลชันนัล โดยใช้โมเดลข้อมูลที่เรียกว่า โมเดลข้อมูลแบบรีเลชันนัล โมเดลข้อมูลชนิดนี้ เป็นแนวคิดที่แสดงถึงข้อมูลที่ผู้ใช้มองเห็น การดำเนินการกับข้อมูล และกฎเกณฑ์ต่าง ๆ ดังนี้

1. โครงสร้างข้อมูล (Data structure) เป็นการจัดองค์กรข้อมูล เพื่อให้ผู้ใช้สามารถมองเห็นข้อมูลต่าง ๆ ซึ่งถูกเก็บในรูปของตาราง ที่เรียกว่ารีเลชัน โดยแต่ละรีเลชันจะประกอบด้วยสดมภ์จำนวนหนึ่ง สดมภ์เหล่านี้จะถูกเรียกว่าแอตตริบิว และแถวของตารางถูกเรียกว่าทูเปิล และรีเลชันสกีมา (Relation Schema) ของรีเลชันใดๆจะหมายถึง แอตตริบิวต่าง ๆ ที่ประกอบขึ้นเป็นรีเลชันนั้น ๆ เช่น จากตารางที่ 3.1 รีเลชันสกีมาของรีเลชัน STUDENT สามารถเขียนอยู่ในรูปประโยคสัญลักษณ์ได้ดังนี้

STUDENT(STUID,STUNAME,MAJOR,CREDIT)

2. การจัดการกับข้อมูล (Data Manipulation) เป็นวิธีในการดำเนินการกับข้อมูลแบบรีเลชัน โดยทั่วไปจะแบ่งคำสั่งที่ใช้ออกเป็น สองประเภทหลักๆ คือ

2.1 คำสั่งที่ใช้ในการกำหนดรีเลชัน (relational assignment)

2.2 คำสั่งที่ใช้จัดการกับรีเลชัน ซึ่งประกอบด้วยคำสั่งต่างๆ ดังนี้ การซีเลค โพรเจค โพรดัก (product) จอย ยูเนี่ยน อินเตอร์เซคชัน (intersection) ดิฟเฟอร์เรนซ์ (difference) และดิวิชัน (division)

3. ความเป็นบูรณภาพของข้อมูล (Data Integrity) เป็นกฎและข้อบังคับที่ใช้กับข้อมูลในรีเลชัน อันประกอบด้วย

3.1 กฎความเป็นบูรณภาพของเอนติตี (Entity Integrity rule) หมายความว่า จะไม่มีแอตตริบิวที่เป็นส่วนประกอบของคีย์หลักของรีเลชันใด ๆ มีค่าเป็นนัล นั่นคือคีย์หลักของตารางจะต้องมีค่าเสมอ (not null)

3.2 กฎความเป็นบูรณภาพของการอ้างอิง (Referential Integrity rule) ถ้า FK เป็นคีย์ภายนอกของรีเลชัน R_2 ซึ่งตรงกับคีย์หลัก PK ของรีเลชัน R_1 แล้ว ทุก ๆ ค่าของ FK ของ R_2 ต้อง

1. เท่ากับค่าของ PK ในบางทูเปิลของรีเลชัน R_1 หมายความว่าค่าของมันคีย์ภายนอกจะต้องสัมพันธ์ กับอีกตารางหนึ่ง ซึ่งมีคีย์ภายนอกนี้เป็นคีย์หลัก เพื่อว่าความสัมพันธ์ระหว่างตาราง และเส้นทางที่ใช้ในการอ้างอิงจะยังคงถูกรักษาไว้

2. คีย์ภายนอก หรือแอตตริบิวที่เป็นส่วนประกอบของคีย์ภายนอก

อาจมีค่าเป็นนัลได้

อย่างไรก็ตาม กฎข้อนี้ยังไม่สามารถควบคุมปัญหาด้านความเป็นบูรณภาพของการอ้างอิงได้อย่างสมบูรณ์ กล่าวอีกนัยหนึ่งก็คือ กฎข้อนี้ก็มีความจำเป็นแต่ไม่เพียงพอ ทั้งนี้เพราะในบางกรณีอาจมีการยกเว้นกฎนี้ได้

3.3 กฎความเป็นบูรณภาพของทั้งหมด (miscelaneous integrity rule) หมายถึงความครบถ้วนของทุก ๆ สดมภ์ในตาราง รวมทั้งคีย์หลัก คีย์ภายนอก และสดมภ์ที่ไม่ใช่คีย์

ข้อดีของโมเดลข้อมูลรีเลชันนัล

1. โมเดลข้อมูลที่ใช้อยู่ในรูปที่ง่ายต่อการเข้าใจ

2. ผู้ใช้สามารถดำเนินการใด ๆ กับข้อมูลได้ โดยไม่ต้องอาศัยความรู้ หรือไม่ต้องเกี่ยวข้องกับวิธีในการเก็บข้อมูลภายในเครื่อง และวิธีการเข้าถึงข้อมูลแต่อย่างไร

3. การดำเนินการกับข้อมูล จะได้ผลลัพธ์เป็นตารางข้อมูลย่อยอีกชุดหนึ่ง (set oriented) ลักษณะการดำเนินการเช่นนี้ ทำให้โมเดลข้อมูลรีเลชันนัลเหมาะสำหรับฐานข้อมูลแบบกระจายมากกว่าโมเดลประเภทอื่น เพราะต้นทุนในการเรียกใช้ข้อมูล จะถูกกว่าโมเดลที่ดำเนินการกับข้อมูลทีละระเบียน (record-at-a-time)

ขั้นตอนการออกแบบฐานข้อมูลแบบรีเลชันนัล ประกอบด้วย 6 ขั้นตอนดังนี้

RDD1 การกำหนดตาราง (Identify Tables) เป็นการแปลงเอนติตีในโมเดลข้อมูลเชิงตรรก ให้เป็นตาราง โดยหนึ่งเอนติตี จะแปลงได้เป็นหนึ่งตารางเท่านั้น

RDD2 การกำหนดสดมภ์ (Indentify Columes) เมื่อได้ตารางจากขั้น RDD1 แล้ว ในขั้นนี้จะเป็นการกำหนดสดมภ์ต่าง ๆ ที่ประกอบกันเป็นตารางในฐานข้อมูลแบบรีเลชันนัล โดยสดมภ์หนึ่งสดมภ์ จะได้มาจากแอตตริบิวหนึ่งแอตตริบิวของเอนติตีในโมเดลข้อมูลเชิงตรรกนั่นเอง

RDD3 การปรับโครงสร้างข้อมูล ให้สัมพันธ์กับสภาพแวดล้อมของระบบจัดการฐานข้อมูลที่มีนั่นเอง (Adapt Data Structure to Product Environment) เพื่อประสิทธิภาพของระบบ ในขั้นนี้จะต้องรู้ถึงโครงสร้างภายในของระบบจัดการฐานข้อมูลที่ใช้ เช่น วิธีการเก็บข้อมูล วิธีการเข้าถึงข้อมูล รวมถึงการจองการใช้ข้อมูลของระบบจัดการฐานข้อมูลว่าเป็นอย่างไร

RDD4 การกำหนดกฎธุรกิจที่เกี่ยวข้องกับเอนติตี (Design for Business Rule About Entities) ตามวิธีของการติดตั้งระบบฐานข้อมูลที่เลือกใช้ ซึ่งประกอบด้วย

1. การกำหนดคุณสมบัติทางตรรกของคีย์หลัก เช่น การกำหนดความเป็นหนึ่งจำนวนแอตตริบิวที่ประกอบกันเป็นคีย์หลัก หรือไม่อนุญาตให้คีย์หลักมีค่าเป็นนัล

2. การกำหนดคุณสมบัติทางตรรกของคีย์รอง เช่น การกำหนดความเป็นหนึ่งจำนวนแอตตริบิวที่ประกอบกันเป็นคีย์รอง หรืออนุญาตให้คีย์รองมีค่าเป็นนัลได้

RDD5 การกำหนดกฎธุรกิจที่เกี่ยวข้องกับรีเลชันชิป (Design for Business Rule About Relationship) เป็นการดูแลรักษาการอ้างอิงที่เหมาะสมจากคีย์ภายนอกของเอนติตีหนึ่ง ที่มีต่อคีย์หลักของเอนติตีอื่น ซึ่งก็คือการรักษาความเป็นบูรณภาพของการอ้างอิงนั่นเอง เช่น เงื่อนไขการเพิ่ม ลบ และแก้ไขคีย์ ตามวิธีของการติดตั้งระบบฐานข้อมูลที่เลือกใช้

RDD6 การกำหนดกฎธุรกิจที่เกี่ยวข้องกับแอตตริบิว (Design for Additional Business Rule About Attributes) ซึ่งประกอบด้วย การกำหนดค่าเป็นไปได้ และการกำหนดทริกเกอร์ดำเนินการ ตามวิธีของการติดตั้งระบบฐานข้อมูลที่เลือกใช้

ระบบเครือข่าย [8,11]

ระบบเครือข่าย จะต้องประกอบด้วย โครงสร้างการเชื่อมต่อในส่วนของฮาร์ดแวร์ ซึ่งจะเชื่อมโยงเข้าเป็นเครือข่าย โดยใช้สื่อโทรคมนาคม และหลักการของซอฟแวร์ ซึ่งเป็นตัวควบคุมความถูกต้องของการสื่อสาร

การนำคอมพิวเตอร์มาเชื่อมโยงกัน เพื่อวัตถุประสงค์ ดังนี้

1. เพื่อสามารถแบ่งปัน และใช้ทรัพยากรร่วมกัน เช่น การใช้ฐานข้อมูล หรือการใช้เครื่องพิมพ์ เป็นต้น

2. เพื่อการติดต่อสื่อสาร ค้นหาและแลกเปลี่ยนข้อมูลระหว่างกันในลักษณะการโต้ตอบ (on-line)

3. ช่วยการบริหาร และควบคุมส่วนต่าง ๆ ของระบบที่อยู่กระจัดกระจาย

4. ช่วยให้สามารถเชื่อมต่ออุปกรณ์ทั้งฮาร์ดแวร์ และซอฟแวร์ ต่างชนิดกันเข้าด้วยกันได้ โดยอาศัยมาตรฐานของเครือข่ายคอมพิวเตอร์ เช่น X.25, IEEE 802.3, IEEE 802.4 IEEE 802.5 เป็นต้น

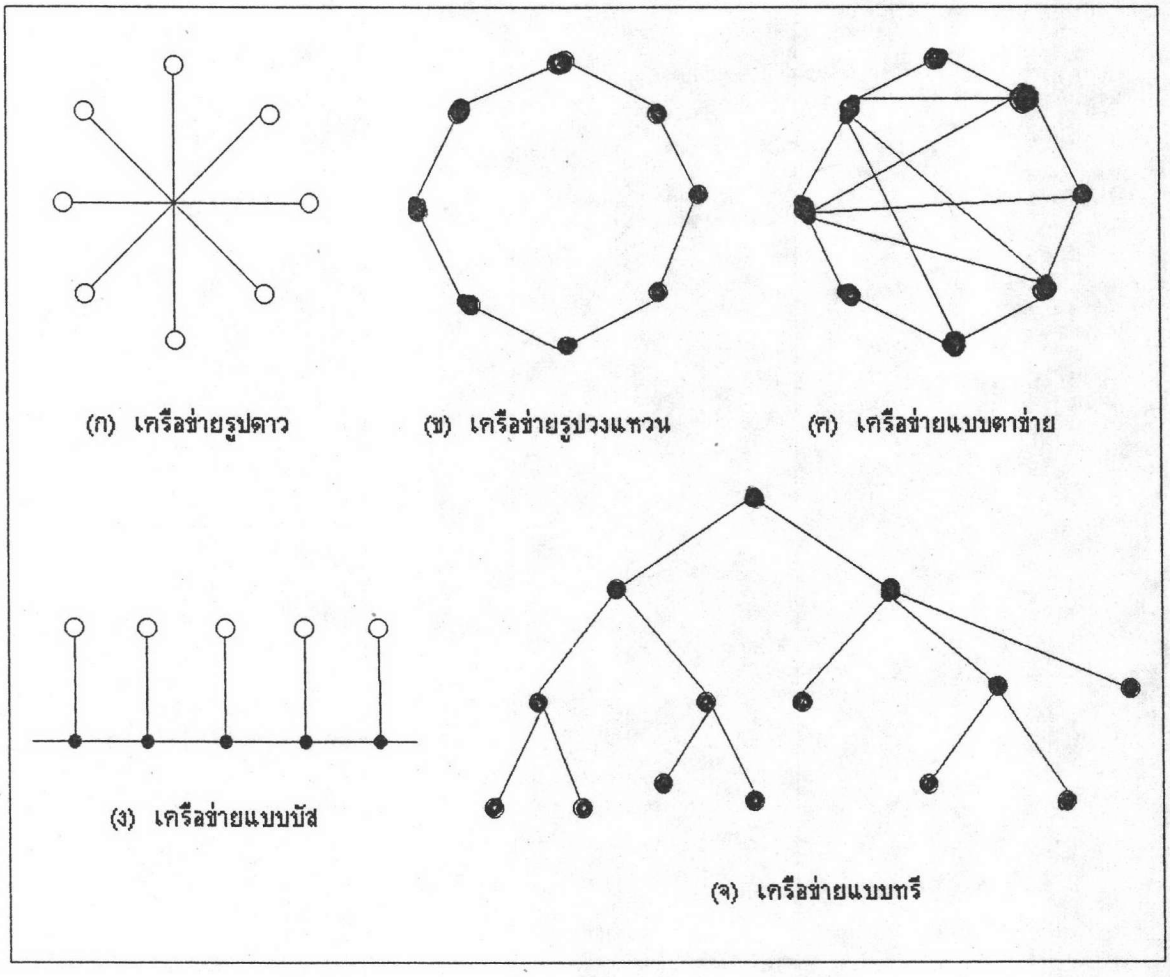
เครือข่ายถูกแบ่งออกเป็นหลายประเภท ตามลักษณะต่างๆ ดังนี้

1. แบ่งตามโครงสร้างการเชื่อมต่อระหว่างสถานี (topology) แบ่งรูปแบบของเครือข่ายสื่อสาร ออกเป็น

1.1 ระบบเครือข่ายรูปดาว (star) มีรูปแบบการติดต่อโดยนำสถานีต่าง ๆ หลายสถานีมาต่อรวมกับหน่วยสวิตชิงกลาง (switching center) ดังแสดงในรูป 3.22 (ก) และรูป 3.23 โดยการติดต่อสื่อสารระหว่างสถานีจะกระทำผ่านทางวงจรสวิตชิง ซึ่งทำหน้าที่ติดต่อ

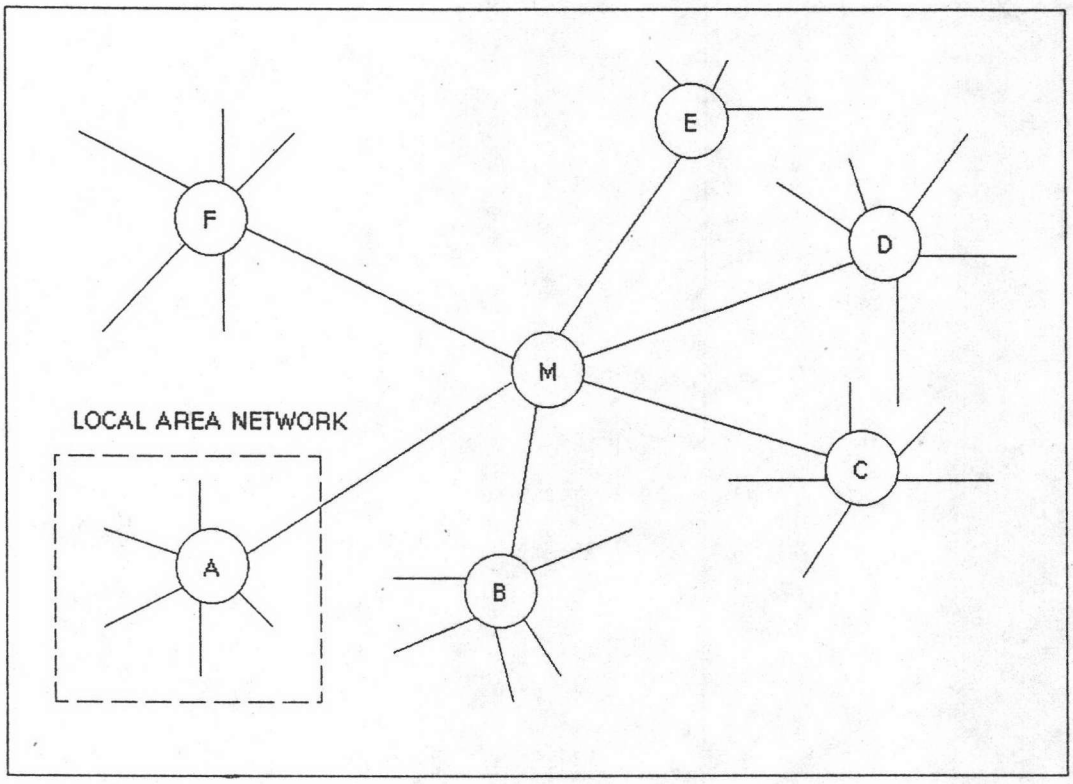
วงจรให้แต่ละสถานีเชื่อมโยงกัน การทำงานของระบบ จะเริ่มจากสถานีที่ต้องการส่งข้อมูล จะส่งข้อความขอให้สวิตชิงกลางรับทราบว่าต้องการติดต่อกับสถานีปลายทางใด อุปกรณ์สวิตชิงจะทำการเชื่อมโยงให้สองสถานีติดต่อกันได้ ดังนั้นการติดต่อของระบบเครือข่ายแบบดาว จึงเป็นการสร้างลิงค์ (link) ระหว่างจุดต้นทางกับปลายทาง

นอกจากนี้ แต่ละสถานียังสามารถติดต่อกับอุปกรณ์รับ-ส่งข้อมูล (terminal) อีกหลายเครื่องก็ได้ กรณีนี้จะถือเป็น Multipoint line เช่น การประมวลผลข้อมูลแบบกระจาย โดยกระจายเครื่องคอมพิวเตอร์ออกไปตามจังหวัดต่าง ๆ ของประเทศ ซึ่งบางจังหวัดอาจต่ออุปกรณ์ออกไป เป็นเครือข่ายสื่อสารท้องถิ่น (local area network) ซึ่งส่วนใหญ่ก็จะเป็นเครือข่ายรูปดาวเช่นกัน โดยคอมพิวเตอร์แต่ละเครื่องนี้ จะเชื่อมโยงกับคอมพิวเตอร์ตัวหนึ่ง ซึ่งเลือกไว้ทำหน้าที่เป็นศูนย์กลาง



รูปที่ 3.22 แสดงโครงสร้างการเชื่อมต่อระหว่างสถานีเครือข่ายชนิดต่าง ๆ

ระบบเครือข่ายรูปดาวนี้ อาศัยการควบคุมการเชื่อมต่อระหว่างสถานี ในลักษณะต่อถึงกันโดยตรง (แบบจุดต่อจุด) และการติดต่อเมื่อสั่งการเชื่อมต่อกันได้แล้ว จะต้องคงไว้เช่นนั้นจนกว่าจะเลิกใช้ ดังนั้นสถานีอื่นจะติดต่อด้วยไม่ได้ ต้องรอจนกระทั่งสายว่าง และหากสถานีที่ทำหน้าที่เป็นศูนย์กลางเกิดเสียขึ้นมา การติดต่อระหว่างสถานีจะทำไม่ได้เลย

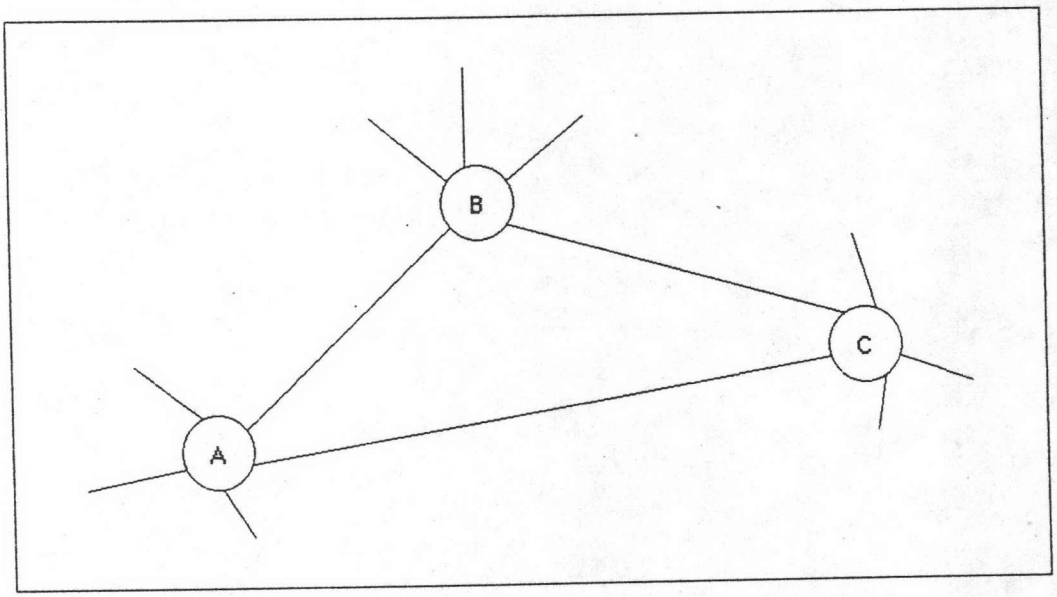


รูปที่ 3.23 แสดงการเชื่อมโยงคอมพิวเตอร์หลาย ๆ ระบบเข้าด้วยกัน โดยใช้เครือข่ายรูปดาว โดยที่เครื่องคอมพิวเตอร์แต่ละเครื่อง ยังทำหน้าที่เป็นศูนย์เครือข่ายท้องถิ่นอีกด้วย

1.2 ระบบเครือข่ายแบบวงแหวน (ring) เครือข่ายแบบวงแหวนนี้ ประกอบด้วยตัว รีพีทเตอร์ (repeater) โดยรีพีทเตอร์ตัวหนึ่งจะต่อกับสถานีหนึ่ง และรีพีทเตอร์จะมีลิงค์ต่อถึงกันเป็นลูป (loop) ดังแสดงในรูป 3.22(ข) ตัวรีพีทเตอร์เป็นอุปกรณ์ที่สามารถรับข้อมูลจากลิงค์ด้านหนึ่ง และส่งออกด้วยความเร็ว เท่ากับที่รับเข้ามา โดยที่ไม่มีการเก็บบัฟเฟอร์ไว้ ลักษณะของลิงค์ที่ต่อเป็นลูปที่มีทิศทางเดียว ซึ่งอาจจะวนทางด้านใดด้านหนึ่งเพียงด้านเดียวเท่านั้น

จากข้อเสียของเครือข่ายรูปดาวข้างต้น เราอาจปรับปรุงเครือข่ายเสียใหม่ โดยใช้รูปแบบวงแหวนร่วมกับรูปดาว ดังรูป 3.24 จะเห็นว่าที่เครื่องคอมพิวเตอร์แต่ละตัวจะควบคุมเครือข่ายแบบดาวของตนเอง และแต่ละเครื่องจะเชื่อมต่อถึงกันแบบวงแหวน ทำให้เกิด

ช่องทางการสื่อสารเพิ่มขึ้น โดยการติดต่อระหว่างเครื่องคอมพิวเตอร์ 2 เครื่องใด ๆ สามารถทำได้ 2 ช่องทาง ดังนั้นหากช่องทางหนึ่งใช้ไม่ได้ ก็สามารถใช้อีกช่องทางหนึ่งได้ และหากมีเครื่องใดในวงแหวนเสียไป การติดต่อจะทำไม่ได้ เฉพาะอุปกรณ์รับ-ส่งข้อมูล ที่ต่อเชื่อมกับเครื่องคอมพิวเตอร์เครื่องที่เสียเท่านั้น ส่วนอื่น ๆ ยังคงสื่อสารกันได้อยู่ เพราะยังมีเส้นทางสำรองเส้นอื่น

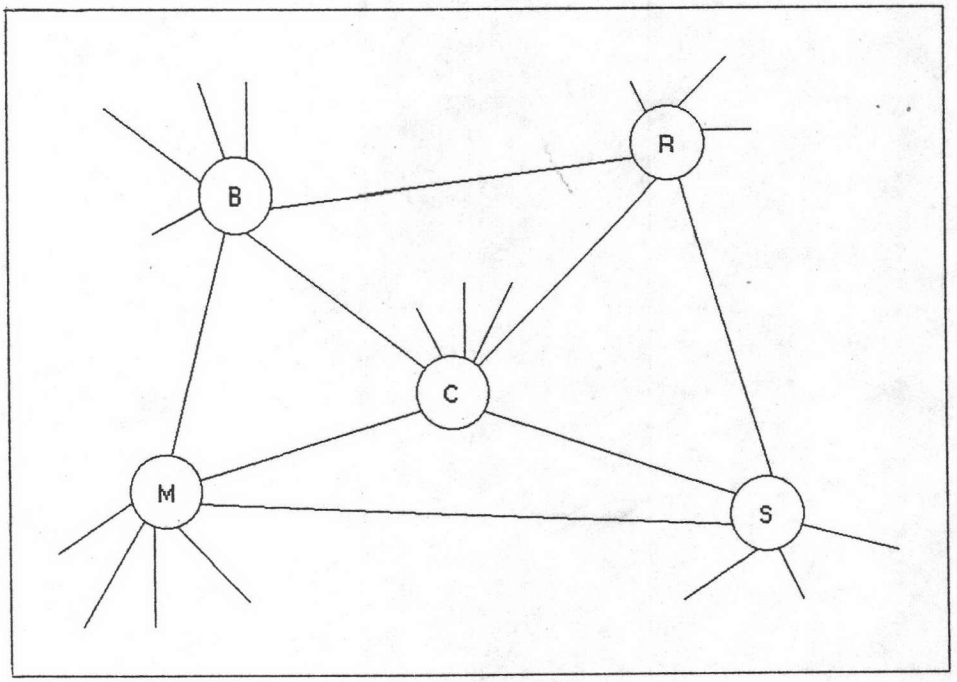


รูปที่ 3.24 แสดงการเชื่อมโยงคอมพิวเตอร์หลาย ๆ ระบบเข้าด้วยกัน โดยใช้เครือข่ายรูปวงแหวน โดยที่เครื่องคอมพิวเตอร์แต่ละเครื่อง ยังทำหน้าที่เป็นศูนย์เครือข่ายท้องถิ่นอีกด้วย

1.3 ระบบเครือข่ายแบบตาข่าย (meshed) เป็นการต่อเครื่องคอมพิวเตอร์จากเครื่องหนึ่งไปยังอีกเครื่องหนึ่ง โดยมีเส้นทางการส่งข้อมูลมากกว่า 1 เส้นทาง ใช้ในกรณีที่ต้องการส่งผ่านข้อมูลในปริมาณที่สูง และเป็นการส่งข้อมูลระยะไกล เราอาจใช้ระบบเครือข่ายช่วย เพราะเป็นการเชื่อมต่อคอมพิวเตอร์แบบถึงกันโดยตรง โดยไม่ต้องผ่านเครื่องอื่น หรือหากจำเป็นต้องผ่านก็ลดจำนวนเครื่องที่ต้องผ่านลง ดังรูปที่ 3.22 (ค) และ รูปที่ 3.25 ถ้าเครื่องคอมพิวเตอร์ B ต้องการติดต่อกับเครื่อง S จะสามารถเลือกช่องทางการสื่อสารได้มากขึ้น

การตัดสินใจเลือกรูปแบบเครือข่าย จะต้องพิจารณาถึงค่าใช้จ่ายในเรื่องสายส่งสัญญาณ ลักษณะทางภูมิศาสตร์ที่เครือข่ายไปได้ถึง และปริมาณข้อมูลที่ต้องส่งผ่าน

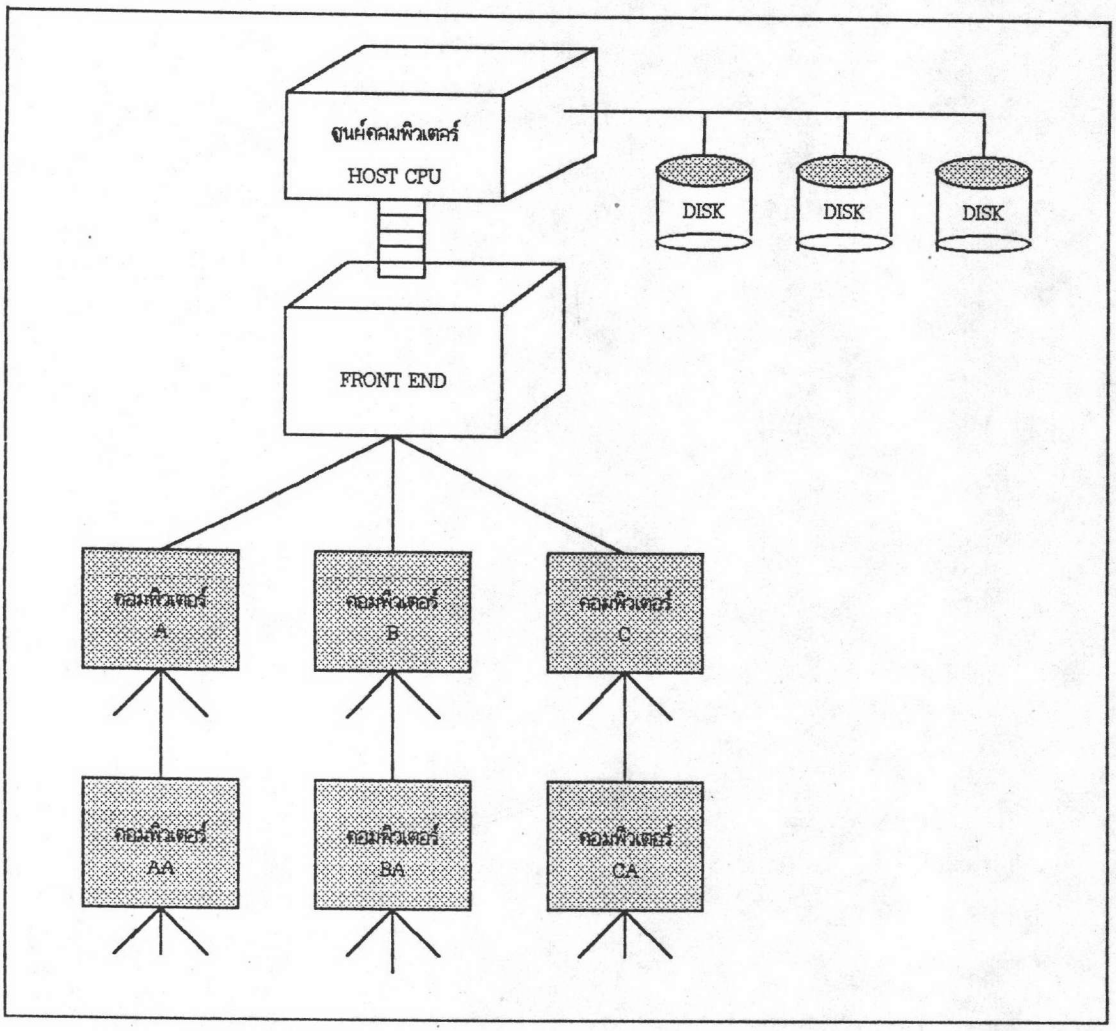
1.4 ระบบเครือข่ายแบบบัส (bus) เครือข่ายแบบนี้มีโครงสร้างที่ไม่ยุ่งยาก ทุกสถานีจะต่อเข้าหาบัส โดยผ่านอุปกรณ์อินเตอร์เฟส (Interface) ที่เป็นฮาร์ดแวร์ การส่งข้อมูลลงบัสนี้ สามารถทำได้ทีละสถานี และข้อมูลจะไปถึงได้ทุกสถานี เพราะอยู่บนบัสเดียวกัน ดังรูป 3.22 (ง)



รูปที่ 3.25 แสดงการเชื่อมโยงคอมพิวเตอร์หลาย ๆ ระบบเข้าด้วยกัน โดยใช้เครือข่ายแบบตาข่าย โดยที่เครื่อง คอมพิวเตอร์แต่ละเครื่องยังทำหน้าที่เป็นศูนย์เครือข่ายท้องถิ่นอีกด้วย

1.5 ระบบเครือข่ายแบบลำดับชั้น (hierarchical) เครือข่ายชนิดนี้จะประกอบไปด้วยเครื่องคอมพิวเตอร์หลายระดับ ต่อเชื่อมซึ่งกันและกันเป็นชั้น ๆ ดูคล้ายรากของต้นไม้ เราจึงเรียกอีกอย่างว่า เครือข่ายแบบทรี ดังแสดงในรูป 3.22 (จ) และ รูปที่ 3.26

จะเห็นว่าระบบเครือข่ายแบบทรี ก็เป็นแบบบัสนั่นเอง เพียงแต่สายตัวกลางจะถูกแยกออกไปเป็นกิ่งก้าน ไม่เป็นวงรอบ การรับส่งข้อมูลจะผ่านตัวกลางเข้าไปยังตัวอื่น ๆ ได้หมด เพราะทุกสถานีอยู่บนบัสเดียวกัน



รูปที่ 3.26 แสดงเครือข่ายแบบลำดับขั้น

2. แบ่งตามรูปแบบการส่งข้อมูล วิธีนี้จะแบ่งระบบเครือข่ายออกเป็นสองชนิด

2.1 เครือข่ายประเภทส่งข้อมูลแบบจุดต่อจุด (point-to-point channel) โดยสถานีทุกคู่จะเชื่อมโยงถึงกันด้วยสายสื่อสาร (link) และจะไม่ยอมให้ผู้อื่นใช้สายร่วมด้วย เช่น เครือข่ายรูปดาว เครือข่ายแบบวงแหวน และ เครือข่ายแบบลำดับขั้น เป็นต้น

2.2 เครือข่ายประเภทส่งข้อมูลทีเดียวให้หลายจุด (multi-point network) เครือข่ายประเภทนี้จะใช้ช่องทางการสื่อสาร (channel) ร่วมกัน เช่น เครือข่ายแบบบัส (bus) และเครือข่ายที่ส่งข้อมูลโดยใช้ดาวเทียม เป็นต้น

3. แบ่งตามพื้นที่ทางภูมิศาสตร์ ระบบเครือข่ายจะถูกแบ่งออกเป็นสองประเภท คือ

3.1 เครือข่ายระยะใกล้ หรือเครือข่ายท้องถิ่น [6]

คุณสมบัติของเครือข่ายท้องถิ่น คือ

1. สามารถต่อเชื่อมได้ในสถานที่เดียวกัน หรือสถานที่ที่อยู่ใกล้กัน

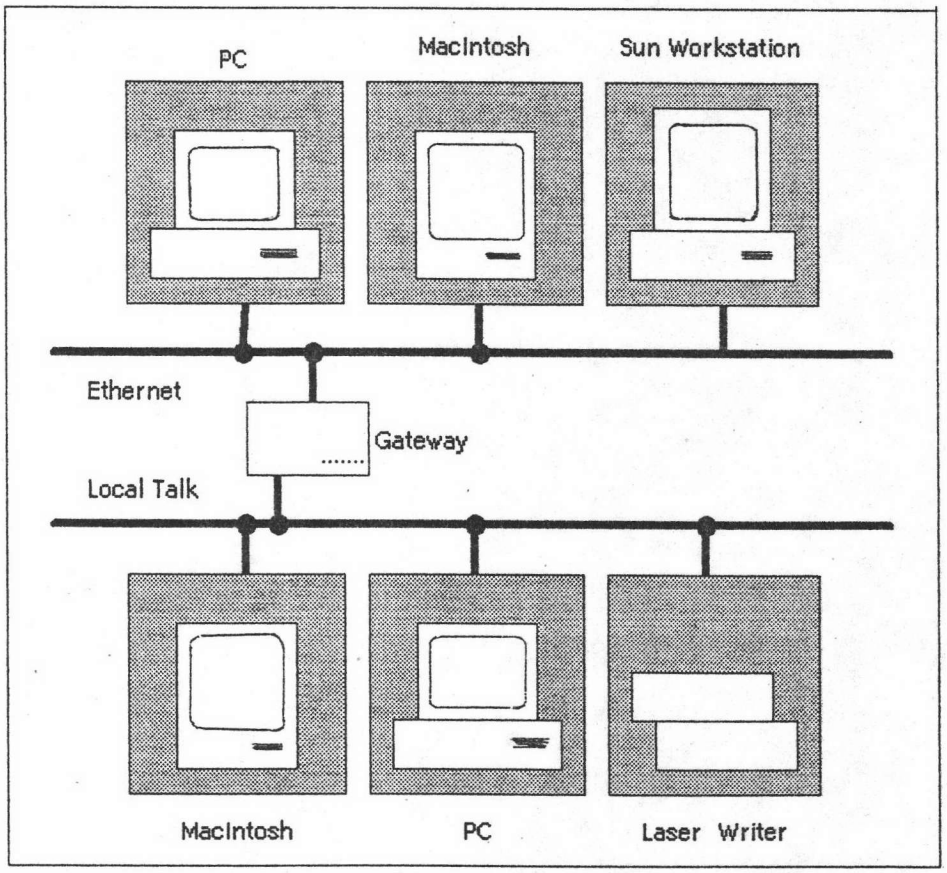
2. ระยะทางในการต่อเชื่อมมีขีดจำกัด และอยู่ในรัศมีไม่เกินหนึ่งกิโลเมตร

3. สามารถต่อเชื่อมกับคอมพิวเตอร์ต่างชนิดกันได้ โดยใช้อุปกรณ์ตัวเชื่อมที่เหมาะสม เช่น เกทเวย์ (Gateway) บริดจ์ (Bridge) หรือ เราท์เตอร์ (Router)

4. มีความเร็วในการส่งข้อมูลสูง โดยสามารถส่งข้อมูลด้วยความเร็วตั้งแต่ 1 ล้าน - 10 ล้านตัวอักษร/วินาที

5. สามารถติดตั้งได้เอง ไม่ต้องขออนุญาตจากรัฐ

พิจารณาจากรูป 3.27 กลุ่มคอมพิวเตอร์ด้านบน เป็นการเชื่อมต่อเครื่องคอมพิเตอร์ IBM PC เครื่อง McIntosh และเครื่อง Sun Workstation เข้าด้วยกัน โดยมีการต่อแบบบัส ด้วยอีเทอร์เนตการ์ด (Ethernet card) ส่วนกลุ่มคอมพิวเตอร์ด้านล่าง เป็นการเชื่อมต่อเครื่องคอมพิเตอร์ McIntosh เครื่อง IBM PC และเครื่อง Laser Writer เข้าด้วยกัน โดยมีการต่อแบบบัส ด้วยสายสื่อสารของเครื่อง McIntosh ที่เรียกว่าโลคอลทอล์ค



รูปที่ 3.27 แสดงการเชื่อมต่อเครือข่ายคอมพิวเตอร์แบบต่างรูปแบบกัน

3.2 เครือข่ายระยะไกล (Wild Area Network - WAN)

เครือข่ายระยะไกลมีคุณสมบัติ 5 ประการดังนี้

1. สามารถต่อเชื่อมอุปกรณ์ระบบคอมพิวเตอร์จากสถานที่หลาย ๆ แห่งเข้าด้วยกัน

2. ระยะทางในการต่อเชื่อมไม่มีขีดจำกัด สามารถเชื่อมได้ตั้งแต่ 10 ถึง 10,000 กิโลเมตร ทั้งภายในและต่างประเทศ

3. มีการกำหนดมาตรฐาน หรือจัดหาอุปกรณ์ที่เป็นมาตรฐาน ทำให้สามารถต่อเชื่อมเครื่องคอมพิวเตอร์ชนิดเดียวกัน หรือต่างชนิดกันได้

4. ความเร็วในการส่งข้อมูลต่ำกว่าเครือข่ายท้องถิ่น โดยอยู่ระหว่าง 1,000-100,000 ตัวอักษร/วินาที (สายขององค์การโทรศัพท์ สามารถใช้สื่อสารได้ตั้งแต่ 2,400 ถึง 4,800 ตัวอักษร/วินาที)

5. ดำเนินการโดยรัฐ หรือเอกชนเป็นผู้ให้บริการ

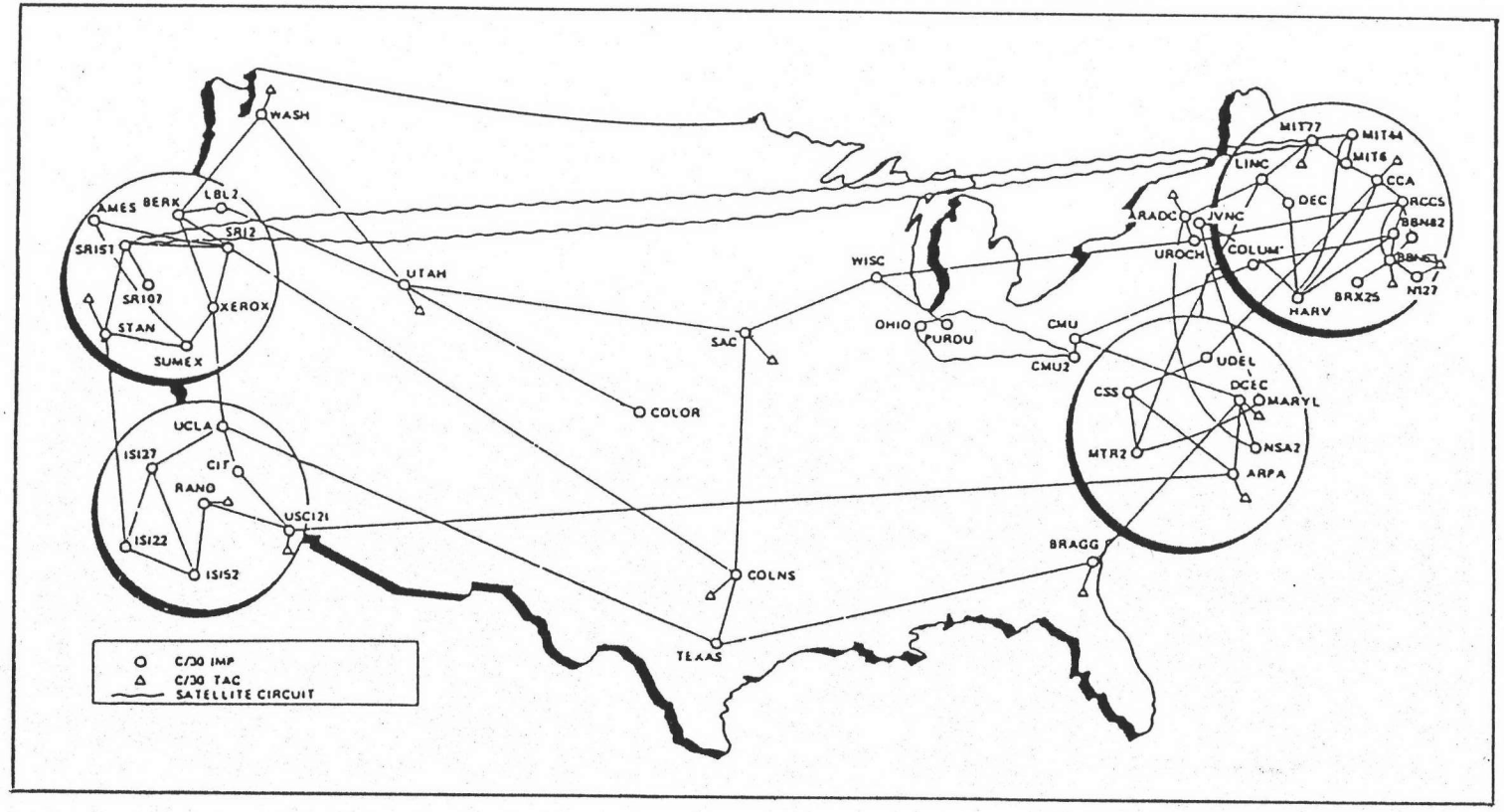
ตัวอย่างเครือข่ายระยะไกลภายในประเทศ เช่น

1. เครือข่าย Datanet โดยบริษัทชินวัตร ฯ เป็นผู้ได้รับสัมปทานในการวางเครือข่าย สำหรับการสื่อสารข้อมูลภายในกรุงเทพมหานคร ความเร็งในการส่งข้อมูลประมาณ 9,600 ตัวอักษร/วินาที

2. เครือข่าย THAIPAK โดยการสื่อสารแห่งประเทศไทยเป็นผู้รับผิดชอบเครือข่ายนี้ใช้มาตรฐานของ CCITT (องค์กรมาตรฐานโลกด้านโทรคมนาคมและการสื่อสารข้อมูล) โดยใช้ X.25 Packet Switching ต่อเชื่อมข้อมูลประมาณ 40 จังหวัด ความเร็วในการส่งข้อมูลใกล้เคียงกับ Datanet

3. เครือข่ายการสื่อสารผ่านดาวเทียม โดยใช้ดาวเทียมเป็นสื่อโทรคมนาคมในการสื่อสารข้อมูล ดำเนินการโดยบริษัทสามารถเทเลคอม ฯ เป็นผู้ได้รับสัมปทาน

ส่วนเครือข่ายระยะไกลในต่างประเทศ ได้แก่ เครือข่าย Arpanet ของสหรัฐอเมริกา เป็นการเชื่อมโยงศูนย์คอมพิวเตอร์ของเมือง และรัฐต่าง ๆ เข้าด้วยกัน โดยมีลักษณะการต่อแบบตาข่าย ใช้สื่อโทรคมนาคมทั้ง LAN LINE (สายที่ลากผ่านพื้นดิน) และสื่อที่เป็นดาวเทียม ดังรูปที่ 3.28



รูปที่ 3.28 แสดงเครือข่ายระยะไกล Arpanet ของประเทศสหรัฐอเมริกา