บทที่ 2

แนวคิดและทฤษฎี

การเปลี่ยนระบบแฟ้มข้อมูลเป็นโมเดลฐานข้อมูลแบบสัมพันธ์ ต้องศึกษาถึงโครงสร้างและการจัดการองค์กรของแฟ้มข้อมูลที่มีใช้งานอยู่ในระบบงานปัจจุบัน และทฤษฎีที่เกี่ยวข้องกับโมเดลทางด้านฐานข้อมูล โดยเฉพาะโมเดลของฐานข้อมูลแบบสัมพันธ์ เพื่อกำหนดแนวทางของการเปลี่ยนแปลงระบบแฟ้มข้อมูลให้กลายเป็นโมเดลฐานข้อมูลแบบสัมพันธ์ และหาวิธีการที่นำคอมพิวเตอร์มาสร้างเครื่องมือเพื่อช่วยในการทำงาน ต่อไปนี้ขอกล่าวถึงทฤษฎีและแนวคิดที่นำมาพิจารณาใช้งาน แยกเป็นเรื่อง ๆ ดังนี้

การจัดการองค์กรของแฟ้มข้อมูล

สำหรับการจัดองค์กรแฟ้มข้อมูล หมายถึง วิธีการเก็บบันทึกข้อมูลลงไปในแฟ้มข้อมูลบนสื่อบันทึกข้อมูล และการเข้าถึงข้อมูลที่อยู่ในแฟ้มข้อมูล จาก EVEREST GORDON C.(1986) ได้มีการแบ่งวิธีการจัดเก็บและวิธีเข้าถึงที่ใช้งานองค์กรแฟ้มข้อมูลอยู่ 4 แบบ ได้แก่

1 การจัดการแฟ้มข้อมูลแบบแฟ้มต่อเนื่อง (serial file)

เป็นวิธีการเก็บข้อมูลลงแฟ้มข้อมูลตามลำดับการเกิดของข้อมูล ข้อมูลชุดใดเกิดก่อนให้เก็บบันทึกลงไปในแฟ้มข้อมูลก่อน ข้อมูลที่เกิดทีหลังให้เก็บบันทึกลงในแฟ้มข้อมูลต่อจากข้อมูลชุดที่เกิดก่อน เป็นรูปแบบที่ได้รับความนิยมน้อยมาก

2 การจัดการแฟ้มข้อมูลแบบแฟ้มลำดับ (sequential file)

เป็นวิธีการเก็บข้อมูลลงแฟ้มข้อมูลที่มีการเรียงลำดับตามเขตข้อมูลใดเขตข้อมูลหนึ่งหรือ กลุ่มของเขตข้อมูล วิธีสร้างแฟ้มข้อมูลประเภทนี้ทำได้โดยนำแฟ้มต่อเนื่อง มาเรียงลำดับตามเขตข้อมูลหรือกลุ่มของเขตข้อมูลแล้วจัดเก็บผลลัพธ์ที่ได้ลงแฟ้มข้อมูลใหม่ เรียกแฟ้มข้อมูลใหม่ว่า แฟ้มลำดับ

3 การจัดการแฟ้มข้อมูลแบบแฟ้มตรง (direct file)

เป็นการเก็บบันทึกข้อมูลที่ทำให้สามารถเข้าถึงข้อมูลได้โดยตรง การเก็บบันทึกข้อมูลของการจัดองค์กรแบบนี้สามารถทำได้โดย ทำการสร้างความสัมพันธ์ระหว่างตำแหน่งที่ใช้เก็บข้อมูลแต่ละระเบียนข้อมูล กับ เขตข้อมูลหรือกลุ่มของเขตข้อมูลที่ใช้ในการอ้างถึงข้อมูลชุดนั้น เรียกฟังก์ชันที่ใช้ในการสร้างความสัมพันธ์นี้ว่า การหาที่อยู่แบบแฮช(hashing function)

4 การจัดการแฟ้มข้อมูลแบบแฟ้มลำดับดรรชนี (indexed sequential file)

เป็นการเก็บบันทึกข้อมูลลงสู่แฟ้มข้อมูลที่มีการเรียงลำดับตาม เขตข้อมูลใดเขตข้อมูลหนึ่ง คล้ายกับการจัดองค์กรแฟ้มลำดับ แต่แฟ้มลำดับดรรชนีจะมีการแยกเขตข้อมูลหรือกลุ่มของเขตข้อมูลที่ใช้ในการเรียงลำดับและใช้สำหรับอ้างถึงข้อมูลแต่ละชุดไปสร้างเป็นแฟ้มข้อมูลที่มีโครงสร้างพิเศษอีกหนึ่งแฟ้ม เรียกแฟ้มข้อมูลที่สร้างขึ้นใหม่ว่า แฟ้มดรรชนี (index file) และแฟ้มที่สร้างขึ้นนี้จะมีประโยชน์ช่วยในการค้นหาข้อมูลที่อยู่ในแฟ้มข้อมูลได้เร็วขึ้นอย่างมาก

โมเดลข้อมูล

โมเดลข้อมูล เป็น ภาพที่ใช้อธิบายเอ็นติตี้ และความสัมพันธ์ระหว่างเอ็นติตี้กับเอ็นติตี้ว่าเป็นอย่างไร โดยมีนิยามที่ควรเข้าใจดังนี้

เอ็นติตี้ (ENTITY) หมายถึง สิ่งที่กำลังสนใจและมีคุณสมบัติแตกต่างจากวัตถุชิ้นอื่น ๆ สามารถจับต้องได้หรือไม่ก็ได้ โดยเอ็นติตี้สามารถ เป็น คน, สัตว์, สิ่งของ ความคิด โดยความหมายของเอ็นติตี้ก็อาจหมายถึงตารางข้อมูลในฐานข้อมูล หรือแฟ้มข้อมูลในระบบแฟ้มข้อมูล

แอตทริบิว (ATTRIBUTE) ใช้เพื่ออธิบายคุณลักษณะของเอ็นติตี้ ซึ่งอาจหมายถึงสิ่งที่เอ็นติตี้เป็น หรือสิ่งที่เอ็นติตี้มีก็ได้ ส่วนมากใช้เป็นคอลัมภ์ในตารางข้อมูล หรือเขตข้อมูลในแฟ้มข้อมูล

คีย์ (KEY) เป็นเซทของแอตทริบิว(หมายถึง แอตทริบิวหนึ่งแอตทริบิว หรือกลุ่มของแอตทริบิวก็ได้) ใช้สำหรับแยกแต่ละบรรทัดในตารางข้อมูลออกจากกัน หรือแยกแต่ละระเบียนข้อมูลออกจากแฟ้มข้อมูลโดยเด็ดขาด โดยสามารถแบ่งคีย์ออกเป็นประเภทต่าง ๆ คือ

1. คีย์ง่าย (SIMPLE KEY) หมายถึง คีย์ที่ประกอบด้วยแอตทริบิวหนึ่งแอตทริบิว ในรูปที่ 2.1 แสดงให้เห็นตารางข้อมูลนักศึกษา(STUDENT) ที่มีแอตทริบิว รหัสนักศึกษา (Student-id)เป็นคีย์ง่ายที่เป็นคีย์หลักของตารางข้อมูลเพียงแอตทริบิวเดียว

Student-id	last-name	first-name	birth-year	address	major-dept.	minor-dept
12345	Victory	Elizabeth	1986	100 Sun st.	Computer Science	Economics
12348	Howards	Jane	1965	200 Dorms	Arts	Economics
43532	Wood	Michael	1964	110 Dorms	Arts	Economics

รูปที่ 2.1 แสดงตารางข้อมูลนักศึกษาที่มีรหัสนักศึกษาเป็นคีย์ของตาราง

2. คีย์ประกอบ (COMPOSITE KEY) หมายถึง คีย์ที่ประกอบขึ้นจากแอตทริบิวมากกว่าหนึ่งแอตทริบิว (เซทของแอตทริบิว) ตัวอย่างเช่น ตารางข้อมูลการลงทะเบียน (COURSE ENROLLMENT) ในรูปที่ 2.2 มีแอตทริบิวรหัสนักศึกษา(student-id) กับแอตทริบิวชื่อวิชา (course-name)จำนวน 2 แอตทริบิวรวมกันเป็นคีย์หลักของตารางข้อมูล

Instruction-id	course-name	year	season	student-id	final-grade
11332	Gastronomy	1987	Fall	12345	100
11332	Gastronomy	1987	Fall	12348	70
11332	Databases	1987	Fall	12348	80

รูปที่ 2.2 แสดงตารางการลงทะเบียนที่มีคีย์หลักเป็นคีย์ประกอบ

3. คีย์คู่แข่ง (CANDIDATE KEY) หมายถึง การที่หนึ่งเอ็นติตี้มีคีย์ที่ใช้ในการแยกแยะแถวมากกว่าหนึ่งคีย์ เหตุการณ์เช่นนี้เรียกว่าเอ็นติตี้นั้นเกิดมีคีย์คู่แข่งขึ้น โดยในทาง

ปฏิบัติหากเกิดมีคีย์คู่แข่งขึ้นในเอ็นติตี้ใดแล้ว ให้ทำการเลือกเพียงหนึ่งคีย์จากคีย์คู่แข่งในเอ็นติตี้มาเป็น คีย์หลัก (PRIMARY KEY) ของตารางข้อมูล ตัวอย่างตารางข้อมูลที่มีคีย์คู่แข่ง แสดงให้เห็นในรูปที่ 2.3 ว่า ตารางข้อมูลพนักงาน (EMPLOYEE) ที่สามารถใช้ แอตทริบิวรหัสพนักงาน (ENO) หรือแอตทริบิวชื่อพนักงาน (ENAME) เป็นคีย์หลักของตารางข้อมูลก็ได้

ENO	ENAME	MGR	DEPT	SALARY
2351	J. NORDBY	5114	LANGUAGES	39000
5114	S.AGARWAL	4016	DBSYSTEMS	35000
3040	G.CANDOR	4016	DBSYSTEM	40000
2011	D.SCHRADER	4016	DBSYSTEMS	65000
4016	K.SMITH	5015	STORAGESYS	75000

รูปที่ 2.3 แสดงตารางลูกค้าที่มีคีย์คู่แข่ง

4. คีย์นอก (FOREIGN KEY) หมายถึง แอตทริบิว หรือกลุ่มของแอตทริบิวในเอ็นติตี้หนึ่งที่ทำหน้าที่เป็นคีย์หลักของอีกเอ็นติตี้หนึ่ง การมีคีย์นอกนี้เป็นการสร้างความสัมพันธ์ระหว่างเอ็นติตี้ที่มีคีย์นอก กับเอ็นติตี้ที่มีคีย์นอกเป็นคีย์หลัก สำหรับตัวอย่างของคีย์นอกให้พิจารณาตารางข้อมูลในรูปที่ 2.2 เป็นตารางข้อมูลการลงทะเบียน ที่มีแอตทริบิวรหัสนักศึกษา เป็นคีย์นอกเพราะแอตทริบิวรหัสนักศึกษานี้เป็นคีย์หลักของตารางข้อมูลนักศึกษาที่ปรากฏอยู่ในภาพที่ 2.1 และการที่มีแอตทริบิวรหัสนักศึกษาในตารางข้อมูลการลงทะเบียนทำให้เกิดความสัมพันธ์ระหว่างตารางข้อมูลการลงทะเบียนกับตารางข้อมูลนักศึกษา

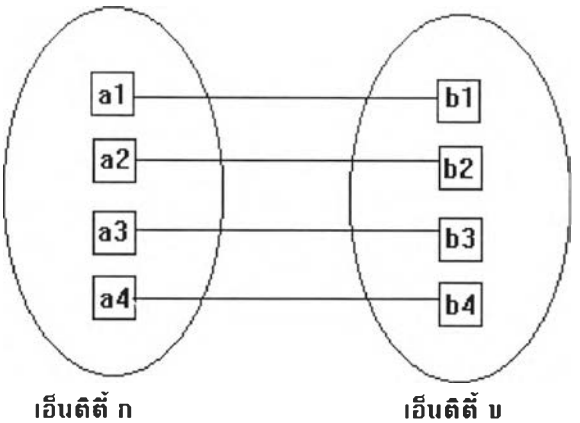
<u>แมปปิง คอนสเตรทส์ (Mapping constraint)</u>

KORTH H.F. และ SILBERSCHATZ (1991) ได้ให้นิยามของ Mapping constraint ว่าเป็นผังแสดงความสัมพันธ์ระหว่างเอ็นติตี้กับเอ็นติตี้ว่าเป็นแบบใด

เพราะว่าความสัมพันธ์ระหว่างเอ็นติตี้กับเอ็นติตี้มีได้หลายแบบ และเอ็นติตี้หนึ่งเอ็นติตี้สามารถมีความสัมพันธ์กับเอ็นติตี้อื่น ๆ ได้พร้อมกันคราวละหลายเอ็นติตี้ เพื่อให้ง่ายกับการกล่าวถึง จึงแสดงความสัมพันธ์ที่เป็นไปได้ของเอ็นติตี้โดยใช้ความสัมพันธ์แบบทวิ(BINARY RELATIONSHIP) ซึ่งหมายถึงความสัมพันธ์ที่เกิดระหว่างเอ็นติตี้ 2 เอ็นติตี้ โดยให้พิจารณาจากเซทของเอ็นติตี้ที่ประกอบด้วยเอ็นติตี้ 2 เอ็นติตี้ ชื่อ เอ็นติตี้ ก และเอ็นติตี้ ข และความสัมพันธ์ที่เกิดขึ้นระหว่างสมาชิกในเอ็นติตี้ทั้งสองความสัมพันธ์นี้สามารถนำมาแสดงรูปแบบได้ในหลายรูปแบบ ดังนี้

1. ความสัมพันธ์แบบหนึ่งต่อหนึ่ง (ONE TO ONE)

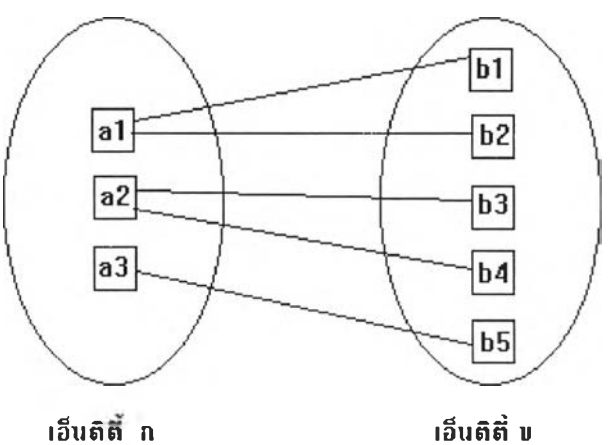
เอ็นติตี้ ก มีความสัมพันธ์แบบหนึ่งต่อหนึ่งไปยังเอ็นติตี้ ข ก็ต่อเมื่อทุก ๆ ค่าของข้อมูลที่เป็นสมาชิกของเอ็นติตี้ ก สามารถจับคู่กับค่าของข้อมูลที่เป็นสมาชิกของเอ็นติตี้ ข ได้เพียงหนึ่งค่าเท่านั้น และทุก ๆ ค่าของข้อมูลที่เป็นสมาชิกของเอ็นติตี้ ข ก็สามารถจับคู่กับค่าของข้อมูลที่เป็นสมาชิกของเอ็นติตี้ ก ได้หนึ่งค่าข้อมูลเช่นกัน ดังรูปที่ 2.4



รูปที่ 2.4 แสดงความสัมพันธ์แบบหนึ่งต่อหนึ่งจากเอ็นติตี้ ก ไปยังเอ็นติตี้ ข

2. ความสัมพันธ์แบบหนึ่งต่อกลุ่ม (ONE TO MANY)

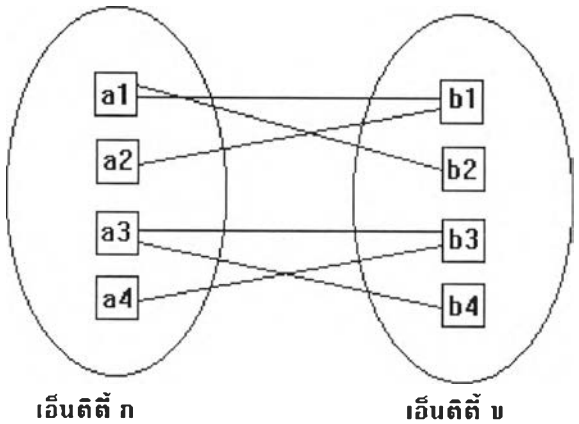
เอ็นติตี้ ก มีความสัมพันธ์แบบหนึ่งต่อกลุ่มไปยังเอ็นติตี้ ข ก็ต่อเมื่อแต่ละค่าของอมูลที่เป็นสมาชิกของเอ็นติตี้ ก สามารถจับคู่กับค่าของข้อมูลที่เป็นสมาชิกของเอ็นติตี้ ข มากกว่าหนึ่งค่าในขณะที่แต่ละค่าของข้อมูลที่เป็นสมาชิกของเอ็นติตี้ ข ที่สามารถจับคู่กับค่าของข้อมูลที่เป็นสมาชิกของเอ็นติตี้ ก ได้เพียงหนึ่งค่าข้อมูลเท่านั้น ดังรูปที่ 2.5



รูปที่ 2.5 แสดงแสดงความสัมพันธ์แบบหนึ่งต่อกลุ่มจากเอ็นติตี้ ก ไปยังเอ็นติตี้ ข

3. ความสัมพันธ์แบบกลุ่มต่อกลุ่ม (MANY TO MANY)

เอ็นติตี้ ก มีความสัมพันธ์แบบกลุ่มต่อกลุ่มไปยังเอ็นติตี้ ข เมื่อค่าของข้อมูลที่เป็นสมาชิกของเอ็นติตี้ ก แต่ละค่าสามารถจับคู่กับค่าของข้อมูลที่เป็นสมาชิกของเอ็นติตี้ ข ได้มากกว่าหนึ่งค่า และค่าของข้อมูลที่เป็นสมาชิกของเอ็นติตี้ ข แต่ละค่าก็สามารถจับคู่กับค่าของข้อมูลที่เป็นสมาชิดของเอ็นติตี้ ก ได้มากกว่าหนึ่งค่าเช่นกัน



รูปที่ 2.6 แสดงความสัมพันธ์แบบกลุ่มไปยังกลุ่มจากเอ็นติตี้ ก ไปยังเอ็นติตี้ ข

โมเดลฐานข้อมูลแบบสัมพันธ์ (Relational Model)

FLEMING CANDACE C. และ HALLE BARBARA V. (1986) ได้กล่าวถึงโมเดลฐานข้อมูลแบบสัมพันธ์ ไว้ว่าโมเดลแบบสัมพันธ์นี้มีองค์ประกอบที่สำคัญ 3 ส่วน ได้แก่

1. ส่วนของโครงสร้างข้อมูล (Data structure)

หมายถึง รูปแบบการจัดองค์กรของข้อมูลที่ผู้ใช้งานข้อมูลสามารถรับรู้หรือมองเห็นได้ และข้อมูลที่มีความสัมพันธ์เหล่านี้ถูกนำมาจัดเก็บลงในตารางข้อมูลแบบสัมพันธ์ (relational table) โดยให้เปรียบเทียบโครงสร้างข้อมูลนี้ว่าเป็นตารางแบบสองทาง โดยที่ตารางแบบสองทางที่กล่าวมานี้เป็นตารางที่มีคุณสมบัติพิเศษไม่เหมือนกับตารางสองทางที่ใช้ในการเรียนการสอนวิชาสถิติ

ตารางข้อมูลแบบสัมพันธ์ เป็นตารางที่ประกอบด้วยกลุ่มของ คอลัมภ์ที่ต้องมีชื่อกำกับ กับแถวของข้อมูลที่ไม่ต้องมีชื่อกำกับจำนวนหนึ่ง ค่าที่อยู่ในแต่ละคอลัมภ์ที่มีชื่อกำกับนี้ต้องมีโดเมนที่สัมพันธ์กับคอลัมภ์เองคอลัมภ์ละ 1 หนึ่งโดเมน นอกจากนั้น ตารางข้อมูลแบบสัมพันธ์ต้องมีคุณสมบัติต่อไปนี้

1.1. ข้อมูลทั้งหมดที่อยู่ในคอลัมภ์ต้องมีค่าที่เป็นไปได้เพียงค่าเดียว

หมายความว่า ข้อมูลที่อยู่ในทุก ๆ คอลัมภ์ในแต่ละแถวจะมีค่าได้เพียงค่าเดียวเท่านั้น(เรียกว่ามี repeating group ไม่ได้) ตัวอย่างของตารางที่ไม่เป็นไปตามกฎดังกล่าวนี้ได้แก่ตารางของผู้ขายของหวานที่ปรากฏในรูปที่ 2.7

VENDOR*NAME	FLAVOR*NAME
Mr. Chip	vanilla, chocolate
Mrs.Chip	avocado, date-nut-cream
Diet-Cream	cottage cheese, skim milk, Celery delight

รูปที่ 2.7 แสดงตัวอย่างของตารางข้อมูลที่ไม่เป็นไปตามคุณสมบัติข้อแรกของ relation table

จากการพิจารณาตารางผู้ขายของหวานในรูปที่ 2.7 พบว่าในตารางมีคอลัมภ์ชื่อ FLAVOR*NAME ที่ไม่ได้มีค่าของข้อมูลเพียงค่าเดียว โดยคอลัมภ์นี้ประกอบด้วยค่าของข้อมูลหลาย ๆ ค่า เช่น ในแถวที่คอลัมภ์ชื่อ VENDOR*NAME เป็น Mr.Chip นั้นมีค่าของ แอตทริบิว FLAVOR*NAME อยู่ 2 ค่า ได้แก่ vanilla กับ chocolate

1.2 ข้อมูลทั้งหมดที่อยู่ในคอลัมภ์ต้องเป็นข้อมูลประเภทเดียวกัน

หมายถึงประเภทข้อมูลที่ใช้ในแต่ละคอลัมภ์ในตารางข้อมูลต้องเป็นประเภทเดียวกันหมดทุกแถว จะใช้ข้อมูลต่างประเภทมาผสมปะปนกันไม่ได้ เช่น ในคอลัมภ์หนึ่งมีประเภทข้อมูลเป็น ตัวอักษร ในแถวหนึ่งแล้วจะมีประเภทข้อมูลเป็น ตัวเลขในอีกแถวหนึ่งไม่ได้

1.3 ทุกแถวในตารางต้องมีข้อมูลที่ไม่ซ้ำกัน

ความหมายของคุณสมบัติของนี้ คือ ในแต่ละแถวของข้อมูลต้องมีอย่างน้อย 1 คอลัมภ์ที่มีค่าที่แตกต่างกันทั้งตารางข้อมูลเพื่อใช้ในการอ้างถึงหรือแยกแยะแถวของข้อมูลออกจากกัน

1.4 ลำดับของคอลัมภ์(จาก ซ้ายไปขวา) ต้องไม่มีความสำคัญ

ความหมายของคุณสมบัติข้อนี้ก็คือ ตำแหน่งการเรียงลำดับของคอลัมภ์ที่ผู้ใช้สามารถใช้งานในตารางข้อมูลสามารถมีได้หลายรูปแบบ ขึ้นอยู่กับลำดับของคอลัมภ์ที่ปรากฏในคำสั่งที่ใช้ในการเรียกข้อมูลจากตารางข้อมูล

1.5 ลำดับของแถว(จาก บนลงล่าง) ต้องไม่มีความสำคัญ

ความหมายของคุณสมบัติของนี้ก็คือ ผู้ใช้หลายคนสามารถใช้ประโยชน์จากการเรียกข้อมูลในลำดับของแถวที่แตกต่างกันได้ ขึ้นอยู่กับ มุมมอง (VIEW) ของผู้ใช้แต่ละคน

1.6 แต่ละคอลัมภ์มีชื่อคอลัมภ์ที่ไม่ซ้ำกันไม่ได้

คุณสมบัติข้อนี้บังคับให้ทุกคอลัมภ์ที่อยู่ในตารางข้อมูลเดียวกันจะมีชื่อซ้ำกันไม่ได้ แต่ไม่ได้หมายความทั้งฐานข้อมูลแต่มีชื่อซ้ำกันไม่ได้

2. ส่วนของการดำเนินการกับข้อมูล (Data manipulation)

หมายถึง ประเภทของการปฏิบัติการที่ผู้ใช้สามารถปฏิบัติการกับโครงสร้างข้อมูลแบบสัมพันธ์ได้ และเพื่อให้ง่ายต่อการแสดงให้เห็นการปฏิบัติการแบบต่างๆ จึงกำหนดตารางข้อมูลชื่อ VENDOR กับ SUPPLY ในรูปที่ 2.8 ขึ้นเพื่อใช้ประกอบการยกตัวอย่างในหัวข้อการปฏิบัติการแต่ละหัวข้อ

VENDOR

NAME	LOCATION
Mr. Chip	New York
Bean, Inc.	New York
Mrs. Mousse	New Jersey
Mr. Mousse	New Jersey
Sancho	California
Diet-Cream	Colorda
Fressze-It	Alaska

SUPPLY

VENDOR*NAME	FLAVOR*NAME
Mr.Chip	vanilla
Mr.Chip	chocolate
Mr.Chip	avocado
Mr.Chip	date-nut-cream
Diet-Cream	cottage cheese
Diet-Cream	skim milk
Diet-Cream	celery delight

รูปที่ 2.8 แสดงตารางเพื่อประกอบการอธิบายการทำงาน

สำหรับโมเดลฐานข้อมูลแบบสัมพันธ์แล้วมีตัวกระทำการที่สำคัญ 8 ตัวกระทำการ ได้แก่

2.1 เซลเลกต์(Select)

เป็นการเลือกใช้ข้อมูลหนึ่งคอลัมภ์ หรือ หลายคอลัมภ์ ที่เป็นเซทย่อยของแถวข้อมูลทั้งหมดจากตารางข้อมูล ตัวอย่างการใช้งานคำสั่งเลือก แถวที่มีสถานที่ตั้งอยู่ในรัฐ New Jersey จากตาราง VENDOR ผลการทำงานนี้สามารถแสดงได้ดังรูปที่ 2.9

NAME	LOCATION
Mrs. Mousse	New Jersey
Mr. Mousse	New Jersey

รูปที่ 2.9 แสดงการใช้คำสั่งเซลเลกต์เพื่อเลือกข้อมูลของ VENDOR ที่อยู่ในรัฐ New Jersey

2.2 โปรเจกต์(Project)

เป็นการเรียกใช้ข้อมูลที่เป็นเซทย่อย ของคอลัมภ์จากตารางข้อมูล หลังจากดำเนินการจนได้ผลลัพท์จากการเลือกแล้วต้องขจัดแถวที่ซ้ำกันออกจากเซทย่อยที่ได้ด้วย ตัวอย่างการใช้คำสั่งโปรเจกต์ เพื่อเลือกคอลัมภ์ LOCATION ออกมาจากตารางข้อมูล VENDOR แสดงผลลัพท์ดังรูปที่ 2.10

LOCATION
New York
New Jersey
California
Colorda
Alaska

รูปที่ 2.10 แสดงผลลัพท์การใช้คำสั่งโปรเจกต์เพื่อเลือกคอลัมภ์ LOCATION

2.3 โปรดักต์(Product)

เป็นการสร้างเซทของแถวข้อมูลขึ้นมาจากการนำทุกแถวข้อมูลจากตารางข้อมูล 2 ตารางมาเชื่อมต่อกัน การโปรดักต์อาจเป็นการโปรดักต์จากตารางข้อมูล 2 ตารางที่ต่างกันหรือเป็นการโปรดักต์ ภายในตารางข้อมูลเดียวกันก็ได้ รูปที่ 2.11 เป็นการโปรดักต์ ตาราง VENDOR กับ SUPPLY

SUPPLY. VENDOR*NAME	SUPPLY. FLAVOR*NAME	VENDOR. NAME	VENDOR. LOCATION
Mr. Chip	vanilla	Mr. Chip	New York
Mr. Chip	vanilla	Bean, Inc.	New York
Mr. Chip	vanilla	Mrs. Mousse	New Jersey
Mr. Chip	vanilla	Mr. Mousse	New Jersey
Mr. Chip	vanilla	Sancho	California
Mr. Chip	vanilla	Diet-cream	Colorado
Mr. Chip	vanilla	Freeze-It	Alaska
Mr. Chip	chocolate	Mr. Chip	New York
Mr. Chip	chocolate	Bean, Inc.	New York
Mr. Chip	chocolate	Mrs. Mousse	New Jersey
Mr. Chip	chocolate	Mr. Mousse	New Jersey
Mr. Chip	chocolate	Sancho	California
Mr. Chip	chocolate	Diet-cream	Colorado
Mr. Chip	chocolate	Freeze-It	Alaska
Mrs. Chip	avocado	Mr. Chip	New York
Mrs. Chip	avocado	Bean, Inc.	New York
Mrs. Chip	avocado	Mrs. Mousse	New Jersey
Mrs. Chip	avocado	Mr. Mousse	New Jersey
Mrs. Chip	avocado	Sancho	California
Mrs. Chip	avocado	Diet-cream	Colorado
Mrs. Chip	avocado	Freeze-It	Alaska
Mrs. Chip	date-nut-cream	Mr. Chip	New York
Mrs. Chip	date-nut-cream	Bean, Inc.	New York
Mrs. Chip	date-nut-cream	Mrs. Mousse	New Jersey
Mrs. Chip	date-nut-cream	Mr. Mousse	New Jersey
Mrs. Chip	date-nut-cream	Sancho	California
Mrs. Chip	date-nut-cream	Diet-cream	Colorado
Mrs. Chip	date-nut-cream	Freeze-It	Alaska
Diet-Cream	cottage cheese	Mr. Chip	New York
Diet-Cream	cottage cheese	Bean, Inc.	New York
Diet-Cream	cottage cheese	Mrs. Mousse	New Jersey
Diet-Cream	cottage cheese	Mr. Mousse	New Jersey
Diet-Cream	cottage cheese	Sancho	California
Diet-Cream	cottage cheese	Diet-cream	Colorado
Diet-Cream	cottage cheese	Freeze-It	Alaska
Diet-Cream	skim milk	Mr. Chip	New York

Diet-Cream	skim milk	Bean, Inc.	New York
Diet-Cream	skim milk	Mrs. Mousse	New Jersey
Diet-Cream	skim milk	Mr. Mousse	New Jersey
Diet-Cream	skim milk	Sancho	California
Diet-Cream	skim milk	Diet-cream	Colorado
Diet-Cream	skim milk	Freeze-It	Alaska

รูปที่ 2.11 แสดงให้เห็นผลการโปรดักต์ตารางข้อมูลของ VENDOR กับ SUPPLY

2.4 จอย(Join)

เป็นการรวมข้อมูลในแนวนอนระหว่างตารางข้อมูล หรือเป็นการรวมข้อมูลภายในตารางข้อมูลเดียวก็ได้ การรวมข้อมูลนี้ให้ดำเนินการเฉพาะแถวที่มีคุณสมบัติตรงตามเงื่อนไขความสัมพันธ์ระหว่างตารางข้อมูลเท่านั้น รูปที่ 2.12 เป็นการจอยระหว่างตาราง SUPPLY กับ ตาราง VENDOR โดยใช้เงื่อนไขในการจอย คือ ค่าใน SUPPLY.VENDOR*NAME มีค่าเท่ากับค่าใน VENDOR.NAME

SUPPLY. VENDOR*NAME	SUPPLY. FLAVOR*NAME	VENDOR. NAME	VENDOR. LOCATION
Mr.Chip	vanilla	Mr.Chip	New York
Mr.Chip	chocolate	Mr.Chip	New York
Diet-Cream	cottage cheese	Diet-Cream	Colorado
Diet-Cream	skim milk	Diet-Cream	Colorado
Diet-Cream	celery delight	Diet-Cream	Colorado

ภาพที่ 2.12 แสดงผลลัพท์การจอยข้อมูลระหว่าง ตาราง VENDOR กับตาราง SUPPLY โดยเงื่อนไขคือชื่อ(VENDOR*NAME กับ NAME)ในทั้ง 2 ตารางเป็นชื่อเดียวกัน

ถ้าการจอยข้อมูลระหว่างตารางข้อมูลเกิดขึ้น โดยใช้เครื่องหมาย เท่ากับ เรียกการจอยแบบนั้นว่าเป็น อีควิจอย (equi-join) และถ้าการจอยมีการขจัดแถวของข้อมูลที่มีเงื่อนไขไม่ตรงตามความต้องการออก (ภาพที่ 2.12) เรียกการจอยแบบนี้ว่า เนเจอร์รัลจอย(Nutural Join)

นอกจากเนเจอร์รัลจอย แล้ว ยังมีการจอยอีกแบบหนึ่งที่เรียกว่า เอาเตอร์จอย (outer join) เป็นการจอยที่รวมข้อมูลทั้งแถวที่สามารถจับคู่กันระหว่าง 2 ตารางได้ และแถวที่จับคู่ระหว่าง 2 ตารางไม่ได้ โดยสำหรับแถวที่ไม่สามารถจับคู่ได้ ค่าที่อยู่ในคอลัมภ์ที่ไม่มีคู่จะถูกกำหนดให้เป็นค่านัล (Null) ตัวอย่าง รูปที่ 2.13 เป็นการทำการเอาเตอร์จอยระหว่าง ตาราง SUPPLY กับ ตาราง VENDOR

SUPPLY VENDOR*NAME	SUPPLY FLAVOR*NAME	VENDOR NAME	VENDOR LOCATION
Mr. Chip	vanilla	Mr. Chip	New York
Mr. Chip	chocolate	Mr. Chip	New York
Mrs. Chip	avocado	null	null
Mrs. Chip	date-nut-cream	null	null
Diet-Cream	cottage cheese	Diet-Cream	Colorado
Diet-Cream	skim milk	Diet-Cream	Colorado
Diet-Cream	celery delight	Diet-Cream	Colorado
null	null	Bean, Inc.	New York
null	null	Mrs. Cheese	New Jersey
null	null	Mr. Cheese	New Jersey
null	null	Sancho	California
null	null	Freeze-It	Alaska

รูปที่ 2.13 แสดงให้เห็นการจอยข้อมูลแบบเอาเตอร์จอย

2.5 ยูเนียน (Union)

เป็นการรวมข้อมูลในแนวตั้งจากตารางข้อมูลตารางหนึ่งกับอีกตารางข้อมูล หรืออาจเป็นการรวมข้อมูลภายในตารางข้อมูลเดิมก็ได้ โดยหลังจากทำการรวมแล้ว ให้ทำการขจัดแถวที่ซ้ำกันออกจากผลลัพท์ของการยูเนียน รูปที่ 2.14 แสดงให้เห็นผลการยูเนียนของตารางข้อมูล

EAST COAST VENDOR

NAME	LOCATION
Mr. Chip	New York
Bean. Inc.	New York
Mrs. Mousse	New Jersey
Sunnyside	Florida

WEST COAST VENDOR

NAME	LOCATION
Sancho	California
Freeze-It	Alaska
Rocky	Colorado

ผลจากยูเนียน 2 ตารางข้อมูล

NAME	LOCATION
Mr. Chip	New York
Bean. Inc.	New York
Mrs. Mousse	New Jersey
Sunnyside	Florida
Sancho	California
Freeze-It	Alaska
Rocky	Colorado

รูปที่ 2.14 แสดงการยูเนียนตารางข้อมูล

2.6 อินเตอร์เซกชั่น(Intersection)

เป็นการเลือกแถวที่เป็นแถวที่มีข้อมูลร่วมกันระหว่างสองตารางข้อมูล มาเป็นผลลัพท์ รูปที่ 2.15 จะเป็นภาพที่แสดงให้เห็นการอินเตอร์เซกชั่นระหว่างตาราง MATH CLUB กับตาราง SKI CLUB

MATH CLUB

FIRST NAME	LAST NAME
Dizzy	Frump
Mary	Sue
Joe	Einstein
Squeaky	Jones
Grumpy	Bones

SKI CLUB

FIRST NAME	LAST NAME
Mary	Sue
Alen	Einstein
Dizzy	Frump
Squeaky	Thomas
Grumpy	Bones

ผลจากการอินเตอร์เซกชั่น 2 ตาราง

FIRST NAME	LAST NAME
Dizzy	Frump
Mary	Sue
Grumpy	Bones

รูปที่ 2.15 แสดงให้เห็นการอินเตอร์เซกชั่น

2.7 ดิฟเฟอร์เรนซ์(Difference)

เป็นการเลือกแถวที่ปรากฏในตารางข้อมูลหนึ่งที่เป็นตารางข้อมูลตัวตั้ง แต่ไม่ปรากฏในอีกตารางข้อมูลที่เป็นตารางตัวลบ มาเป็นผลลัพท์ โดยถ้านำตาราง MATH CLUB กับ ตาราง SKI CLUB ที่อยู่ในรูปที่ 2.15 มาทำการดิฟเฟอร์เรนซ์ ผลลัพท์ที่ได้จะเป็นดังรูปที่ 2.16

ผลของการดิฟเฟอร์เรนซ์

FIRST NAME	LAST NAME
Joe	Einstein
Squeaky	Jones

รูปที่ 2.16 แสดงผลของการดิฟเฟอร์เรนซ์ตารางข้อมูล

2.8 การดิวิชั่น(Division)

ผลลัพท์ที่ได้จากการทำงานนี้คือ ทุกคอลัมภ์ในตารางข้อมูลตัวตั้งที่ไม่ปรากฏในอีกตารางข้อมูลหนึ่งที่เป็นตารางตัวหาร และทุกค่าของข้อมูลในตารางที่เป็นตัวหารปรากฏในทุกแถวที่อยู่ในตารางตัวตั้ง และเพื่อการง่ายในการ อธิบายขอให้ดูรูป 2.17 เพื่อประกอบการอธิบาย

COMPLETED COURSE (ตัวตั้ง)

STUDENT NUMBER	STUDENT NAME	COURSE NUMBER	COURSE NAME
10	Casey	Rel101	Data Structure
10	Casey	Psy101	Schizophrenia
10	Casey	Rel201	Data Integrity
10	Casey	Rel301	Data Manipulation
30	Christopher	Rel101	Data Structure
20	Stephanie	Ped101	Terrible Twos
30	Christopher	Psy101	Schizophrenia
30	Christopher	Rel301	Data Manipulation
30	Christopher	Rel201	Data Integrity

RELATIONAL MAJOR (ตัวหาร)

COURSE NUMBER	COURSE NAME
Rel101	Data Structure
Rel201	Data Integrity
Rel301	Data Manipulation

ผลลัพท์จากการหาร

STUDENT NUMBER	STUDENT NAME
10	Casey
30	Christopher

รูปที่ 2.17 แสดงการดิวิชั่น

รูป 2.17 แสดงผลการดิวิชั่นของ ตาราง COMPLETE COURSE กับ ตาราง RELATIONAL MAJOR ได้ผลลัพท์ได้เป็นตารางข้อมูลใหม่ที่มี 2 คอลัมภ์ คือ Student number กับ Student name และข้อมูลที่อยู่ในตารางข้อมูลนี้เป็นนักศึกษาที่ลงทะเบียนเรียนในวิชา RELATIONAL MAJOR ครบทุกวิชา

3. ส่วนของการควบคุมความสมบูรณ์ของข้อมูล (Data integrity)

เป็นกลุ่มของกฎทางธุรกิจ (business rule) ที่ใช้ควบคุมพฤติกรรมของค่าของข้อมูลว่าเป็นอย่างไรเมื่อดำเนินการตามตัวกระทำการทางด้าน relational โดยทั่วไปแล้วสำหรับโมเดลข้อมูลแบบสัมพันธ์ มีกฎมาตราฐานสำหรับรักษาความสมบูรณ์ของข้อมูลที่อยู่ในตารางข้อมูลอยู่ 2 กฎ ได้แก่

3.1 กฎความสมบูรณ์ของเอ็นดิตี้ (entity integrity rule)

กฎข้อนี้คือ คีย์หลักต้องมีค่าทุกแถวในเอ็นดิตี้ เพราะ คีย์หลักถูกใช้กลไกสำคัญในการระบุแต่ละแถวของตารางออกจากทุกแถวในตาราง ดังนั้นจะไม่มีค่าไม่ได้

3.2 กฎความสมบูรณ์ของการอ้างถึง (referential integrity rule)

กฎข้อนี้ คือ การรักษาความสมบูรณ์ของคีย์นอกที่อยู่ในตารางข้อมูล กฎนี้ใช้กับตารางข้อมูลที่มีคีย์นอก โดยคีย์นอกที่อยู่ในตารางสามารถมีค่าได้ 2 ค่า คือ ค่าเป็นนัล(ไม่มีค่า) หรือ ต้องมีค่าตรงกับค่าที่มีอยู่ในตารางข้อมูลที่มี คีย์นอกเป็นคีย์หลัก

นอกจากกฎการรักษาความสมบูรณ์ของข้อมูล 2 ข้อข้างต้นนั้นแล้ว ในตารางข้อมูลยังต้องมีกฎความสมบูรณ์ของโดเมน ที่นำมาใช้เพื่อควบคุมความสมบูรณ์ของทุกคอลัมภ์ในตารางข้อมูล ไม่ว่าคอลัมภ์นั้นจะเป็น คีย์หลัก, คีย์นอก, คอลัมภ์ที่ไม่ใช้คีย์ และคอลัมภ์อย่างอื่น ๆ ตัวอย่างสำหรับกฎของโดเมน ที่ใช้รักษาความสมบูรณ์ข้อมูล เช่น การตรวจสอบประเภทข้อมูล, การกำหนดความยาวของข้อมูลในคอลัมภ์, การตรวจสอบช่วงของข้อมูล เป็นต้น

ดาต้าเบสสกีมา (Database schema)

เป็นวิธีการเขียนโครงสร้างเพื่อแทนตารางข้อมูล วิธีการเขียนมีรูปแบบดังนี้คือ

ชื่อตาราง = (แอตทริบิวที่ 1[.แอตทริบิวที่ 2,แอตทริบิวที่ 3,.....])

ชื่อตาราง ให้เขียนชื่อตารางทางซ้ายมือของเครื่องหมายเท่ากับ(=)

แอตทริบิว ให้เขียนแอตทริบิวทุกตัวภายใต้เครื่องหมายวงเล็บที่อยู่ทางขวาของเครื่องหมายเท่ากับ และคั่นแต่ละแอตทริบิวด้วยเครื่องหมายจุลภาค (,)

คีย์ ให้ขีดเส้นใต้ทุกแอตทริบิวที่ประกอบกันเป็นคีย์ของเอ็นติตี้

ตัวอย่างการเขียน

CUSTOMER = (CUST-N. CUST-M, CUST-CR-A, CUST-ADDR-T, CUST-SHIP-T)

จากตัวอย่าง เป็นการเขียนโครงสร้างของตารางข้อมูลลูกค้าที่มีคีย์หลักเป็นแอตทริบิว 1 แอตทริบิว ได้แก่ แอตทริบิวชื่อ CUST-N

การทำนอร์มอลไลเซชั่น (NORMALIZATION)

การทำนอร์มอลไลซ์ มีจุดประสงค์ที่ต้องการแปลงข้อมูลที่รวบรวมมาจากการวิเคราะห์ข้อมูลให้เป็นเอ็นติตี้ของข้อมูลที่มีคุณลักษณะต่อไปนี้ (สำนักการศึกษาระบบสารสนเทศ สถาบันบัณฑิตพัฒนบริหารศาสตร์, 2534)

1 ไม่ให้มีความซ้ำซ้อน (NON-REDUNDANT)

2 มีความคงที่ข้อมูล (CONSISTENT)

3 มีการขึ้นระหว่างกัน (INTER-DEPENDENT)

4 มีความสัมพันธ์ (RELATIONS)

การทำนอร์มอลไลซ์เป็นการแปลงข้อมูลให้อยู่ในรูปนอร์มอลฟอร์ม (normal form) ในระดับต่าง ๆ โดยเอ็นติตี้ที่ได้หลังการทำนอร์มอลไลซ์ให้ในแต่ละระดับต่าง ๆ แล้วจะมีคุณลักษณะใกล้กับคุณลักษณะที่ต้องการ สำหรับระดับของการนอร์มอลไลซ์ที่ยอมรับได้ทั่วไปในปัจจุบันนี้ คือตั้งแต่ระดับสามเอ็นเอฟเป็นต้นไป โดยในการทำนอร์มอลไลซ์จะเป็นการแปลงข้อมูลให้มีคุณลักษณะดังนี้

1. เฟิร์สนอร์มอลฟอร์ม หรือ หนึ่งเอ็นเอฟ(FIRST NORMAL FORM OR 1NF)

เป็นการทำให้ค่าที่อยู่ในทุกแอตทริบิวที่อยู่ในเอ็นติตี้เป็นแอตทริบิวที่มีค่าของแอตทริบิวเพียงค่าเดียวไม่สามารถแบ่งแยกออกได้(atom) จากนั้นทำให้การกำหนดคีย์หลักให้กับแต่ละเอ็นติตี้

2. เซคคั่นนอร์มอลฟอร์ม หรือ สองเอ็นเอฟ(SECOND NORMAL FORM OR 2NF)

การทำให้เอ็นติตี้อยู่ในระดับสองเอ็นเอฟให้พิจารณาจากเอ็นติตี้ที่อยู่ในระดับหนึ่งเอ็นเอฟแล้วโดยเลือกเฉพาะเอ็นติตี้ที่มีคีย์หลักเป็นคีย์ประกอบเท่านั้น การทำสองเอ็นเอฟเป็นการขจัดแอตทริบิวใด ๆ ที่ค่าของข้อมูลที่อยู่ในแอตทริบิวนั้นไม่ขึ้นกับคีย์หลักที่เป็นคีย์ประกอบทั้งชุดแต่ขึ้นกับบางแอตทริบิวที่เป็นส่วนประกอบของคีย์หลัก โดยให้ทำการแยกแอตทริบิวนั้นกับส่วนประกอบของคีย์ที่แอตทริบิวนั้นขึ้นอยู่มาสร้างเป็นเอ็นติตี้ใหม่ พร้อมกำหนดให้แอตทริบิวที่เป็นส่วนประกอบของคีย์หลักที่แอตทริบิวนั้นขึ้นอยู่เป็นคีย์หลักของเอ็นติตี้ใหม่ จากนั้นทำการลบแอตทริบิวที่แยกออกมานั้นจากเอ็นติตี้เดิม

3. เธิร์ดนอร์มอลฟอร์ม หรือ สามเอ็นเอฟ (THIRD NORMAL FORM OR 3NF)

การทำให้เอ็นติตี้อยู่ในระดับสามเอ็นเอฟให้พิจารณาจากเอ็นติตี้ที่อยู่ในระดับสองเอ็นเอฟแล้ว การทำสามเอ็นเอฟเป็นการขจัดแอตทริบิวที่ค่าของแอตทริบิวนั้นไม่ขึ้นกับคีย์หลักของเอ็นติตี้แต่ขึ้นกับแอตทริบิวอื่น ๆ หรือ เซทของแอตทริบิวที่อยู่ในเอ็นติตี้นั้น การแก้ไขทำโดยแยกแอตทริบิวที่ขึ้นอยู่กับแอตทริบิวอื่น กับแอตทริบิว หรือ เซทของแอตทริบิวที่แอตทริบิวนั้นขึ้นอยู่ มาสร้างเป็นเอ็นติตี้ใหม่ กำหนดให้แอตทริบิว หรือ กลุ่มของแอตทริบิว ที่แอตทริบิวที่กำลังพิจารณาขึ้นอยู่เป็นคีย์หลักของเอ็นติตี้ใหม่ ในกรณีที่แอตทริบิวที่กำลังพิจารณาไม่ใช่ คีย์หลัก หรือ องค์ประกอบของคีย์หลักให้ทำการลบแอตทริบิวที่กำลังพิจารณาออกจากเอ็นติตี้เดิม

หลังจากผ่านการทำนอร์มอลไลซ์แต่ละระดับแล้ว ให้นำเอ็นติตี้ที่ได้จากการนอร์มอลไลซ์แต่ละระดับมารวมกันเพื่อสร้างเอ็นติตี้ใหม่ ก่อนทำนอร์มอลไลซ์ในระดับสูงขึ้นต่อไป

ภาษาเอสคิวแอล (Structure query language : SQL)

เป็นภาษาที่ถูกพัฒนาขึ้นมาเป็นมาตราฐานในการจัดการกับฐานข้อมูล ประเภทของคำสั่งที่ใช้แบ่งเป็น 3 กลุ่มได้แก่

1. กลุ่มกำหนดโครงสร้างข้อมูล (Data Defination Language :DDL)

คำสั่งในกลุ่มนี้ใช้เพื่อทำการอธิบายโครงสร้างของฐานข้อมูล, กฎเพื่อความสมบูรณ์ของข้อมูลและสิทธิต่าง ๆ ของผู้ใช้ข้อมูล ตัวอย่างได้แก่ คำสั่ง CREATE TABLE และ CREATE VIEW

2. กลุ่มที่ใช้ในการจัดการกับข้อมูล (Data Manipulation Language: DML)

คำสั่งกลุ่มนี้ใช้เพื่อการปรับปรุง หรือเรียกใช้ข้อมูล ตัวอย่างได้แก่ คำสั่ง UPDATE, INSERT และ คำสั่ง SELECT

3. กลุ่มโมดูล(Module Language)

คำสั่งกลุ่มนี้เป็นคำสั่งที่ใช้ในการจัดการกับข้อมูล เหมือนกับคำสั่งใน DML ต่างกันที่ คำสั่งในกลุ่มถูกนำไปเขียนในภาษาที่เขียนโปรแกรมได้ เช่น ภาษาซี หรือ ภาษาโคบอล

โดยคำสั่งในภาษาเอสคิวแอลนี้สามารถครอบคลุมตัวกระทำการทั้ง 8 ตัวกระทำการที่สำคัญที่ใช้การปฏิบัติการฐานข้อมูล และสำหรับการนำมาใช้ในการสร้างเครื่องมือนี้จะใช้คำสั่ง CREATE TABLE ที่เป็น DDL โดยคำสั่ง CREATE TABLE มีโครงสร้างของคำสั่งที่ปรากฏในหนังสือของ Rick F. van der Lans (1989) ที่เป็นเอสคิวเอลมาตราฐาน ANSI 86 ดังนี้

CREATE TABLE <table name> (<table element> [{ <table element>} ....])

<table name> ::= [<authorization identifier>.] <table identifier>

<table element> ::= <column definition> | <unique contraint definition>

<column definition> :: = <column name> <data type> [NOT NULL [UNIQUE]]

<unique contraint definition> ::= UNIQUE(<column list>)

<column list> :: = <column name> [{ ,<column name> }....]

<u>แนวคิดในการสร้าง LDM (logical data model)</u>

FLEMING CANDACE C. และ HALLE BARBARA (1986) ได้เสนอแนะแนวทางที่ใช้ในการสร้าง LDM โดยมีขั้นตอนที่เสนอแนะนั้นมีทั้งหมด 12 ขั้นตอน สรุปได้ดังต่อไปนี้

ขั้น LDM 1 : ระบุเอ็นติตี้หลัก

1.1 เป็นการกำหนดเอ็นติตี้, ตั้งชื่อ, สร้างไดอะแกรม และจัดทำเอกสาร

ขั้น LDM 2 : กำหนดความสัมพันธ์ระหว่างเอ็นติตี้

2.1 เป็นการกำหนด, ตั้งชื่อ, สร้างไดอะแกรม และทำบันทึกเอกสารเกี่ยวกับ *ความสัมพันธ์ของเอ็นติตี้*

2.2 แบ่งแยกความสัมพันธ์ออกเป็นแบบ *หนึ่ง-ต่อ-หนึ่ง* หรือ *หนึ่ง-ต่อ-กลุ่ม* และทำการขจัด ความสัมพันธ์ *กลุ่ม-ต่อ-กลุ่ม*

2.3 ทำการจัดกลุ่ม *ความสัมพันธ์ที่ซับซ้อน* ให้เหมือนกับเป็นเอ็นติตี้

2.4 กำจัดความสัมพันธ์ที่ซ้ำซ้อน

2.5 สร้างความสัมพันธ์แบบ หนึ่ง-ต่อ-หนึ่ง ระหว่าง suppertypes กับsubtypes และ ระหว่าง supertypes กับ catagory

2.6 แทนที่ ความสัมพันธ์แบบ หนึ่ง-ต่อ-กลุ่ม ในลักษณะ bill-of-material และขจัดความสัมพันธ์แบบ กลุ่ม-ต่อ-กลุ่ม ในลักษณะ bill-of-material

ขั้น LDM 3 : กำหนดคีย์หลักและคีย์เลือก

3.1 เลือก คีย์หลัก สำหรับแต่ละเอ็นติตี้

3.2 ระบุคีย์เลือก (alternate key) สำหรับแต่ละเอ็นติตี้

3.3 เลือก คีย์หลัก ของเอ็นติตี้ที่เป็น subtypes และเอ็นติตี้ทีเป็น supertypes เป็นคีย์เดียวกัน

3.4 ตั้งชื่อ, สร้างไดอะแกรม และทำบันทึกเอกสารเกี่ยวกับ *คีย์ต่าง ๆ*

3.5 สร้างมาตราฐานการตั้งชื่อ

3.6 ใช้มาตราฐานการย่อคำช่วยในการตั้งชื่อ

ขั้น LDM 4 : กำหนดคีย์นอก

4.1 สำหรับทุกความสัมพันธ์ ให้ทำการกำหนด *คีย์นอก*

4.2 ตั้งชื่อ, สร้างไดอะแกรม และทำบันทึกเอกสารเกี่ยวกับ *คีย์นอก*

ขั้น LDM 5 : กำหนดกฎเกณฑ์ธุรกิจสำหรับคีย์

5.1 ระบุกฎเกณฑ์ที่ใช้ในการเพิ่มข้อมูลสำหรับแต่ละความสัมพันธ์

5.2 ระบุกฎเกณฑ์ที่ใช้ในการลบข้อมูลสำหรับแต่ละความสัมพันธ์

5.3 หลีกเลี่ยงการใช้กฎของ nullify สำหรับการเพิ่มหรือลบข้อมูล ให้ใช้ กฎของค่า default แทน

5.4 ไม่ควรกำหนดกฎของ nullify ในการเพิ่มหรือลบข้อมูล เมื่อมีคีย์นอกเป็นส่วนหนึ่งของ คีย์หลัก ของ เอ็นติตี้ที่เป็นลูก

5.5 ควรที่จะกำหนดกฎการเพิ่มข้อมูล สำหรับความสัมพันธ์แบบ subpertype-subtypes ให้เป็นแบบ automatic หรือ dependent และ กำหนด กฎการลบข้อมูล ให้เป็นแบบ cascade

ขั้น LDM 6 : เพิ่มแอตทริบิวส่วนที่เหลือของเอ็นติตี้

6.1 ให้ทำการเพิ่มทุกแอตทริบิวที่ขึ้นอยู่กับ คีย์หลักทั้งเซท เข้าไปในเอ็นติตี้

6.2 ให้นำแอตทริบิวที่ไม่ใช้คีย์ไปเพิ่ม ณ ตำแหน่งสูงสุดเท่าที่เป็นไปได้ใน logical data model

6.3 ถ้าแอตทริบิวที่อยู่ในเอ็นติตี้เหล่านั้นขึ้นอยู่กับ คีย์หลัก แต่เป็น multivalued ให้ทำการจับกลุ่มของแอตทริบิว เป็น เอ็นติตี้ลูก(Child Entity)เอ็นติตี้ใหม่ ของเอ็นติตี้นั้น

6.4 ตั้งชื่อ, วาดลงไดอะแกรม, และจดบันทึกข้อมูลเกี่ยวกับแอตทริบิวลงสู่ Data dictionary

6.5 ถ้าแอตทริบิวที่อยู่ในเอ็นติตี้เหล่านั้นใช้อธิบายความสัมพันธ์ ให้ทำการจับกลุ่มของแอตทริบิว เป็น เอ็นติตี้ลูก(Child Entity)เอ็นติตี้ใหม่ ของเอ็นติตี้นั้น

6.6 หลีกเลี่ยงการแทนที่ข้อมูลที่อยู่ในแอตทริบิวในรูปแบบการเข้ารหัส

6.7 อย่าได้นำ flags ที่เกิดจากการประมวลผลข้อมูลมาจัดเก็บเป็นแอตทริบิวของเอ็นติตี้

6.8 ถ้าไม่สามารถหลีกเลี่ยงข้อมูลที่อยู่ในรูปแบบการเข้ารหัส ให้แน่ใจว่ารหัสนั้นมีความอิสระพอ

6.9 ให้เก็บ derived data เป็นแอตทริบิวหนึ่งของเอ็นติตี้ ถ้าหากว่าแอตทริบิวนี้สามารถทำประโยชน์ทางด้านธุรกิจได้

6.10 ใช้การออกแบบพิเศษสำหรับ *ชื่อของ subtypes*

6.11 นำแอตทริบิวที่เป็นแอตทริบิวทั่วไปที่ปรากฏใน supertypes ไปเป็นแอตทริบิวของ subtypes

6.12 ให้ทำการรวมเอ็นติตี้ 2 เอ็นติตี้ที่มีคีย์หลักเหมือนกันเป็นเอ็นติตี้เดียวกัน

6.13 ให้ทำการรวมเอ็นติตี้ที่เป็น subtypes ทั้งหมดที่มีแอตทริบิว เหมือนกันและมีความสัมพันธ์เหมือนกันเป็นเอ็นติตี้เดียว

6.14 ให้ทำการรวม supertypes กับ subtypes ที่ใช้แบ่งประเภทเป็นเอ็นติตี้เดียว

6.15 ให้ทำการรวมเอ็นติตี้ที่ไม่มี แอตทริบิวอื่นนอกจากคีย์กับเอ็นติตี้ที่เป็นเอ็นติตี้ลูก (Child Entity) เป็นเอ็นติตี้เดียวกัน

ขั้น LDM 7 : ตรวจสอบความถูกต้องของเอ็นติตี้ตามกฎการนอร์มอลไลเซชั่น

7.1 ลดรูปเอ็นติตี้ทุกเอ็นติตี้ให้อยู่ใน หนึ่งเอ็นเอฟ

7.2 ลดรูปเอ็นติตี้ทุกเอ็นติตี้ที่อยู่ในหนึ่งเอ็นเอฟ ให้เป็น สองเอ็นเอฟ

7.3 ลดรูปเอ็นติตี้ทุกเอ็นติตี้ที่อยู่ในสองเอ็นเอฟ ให้เป็น สามเอ็นเอฟ

7.4 ลดรูปเอ็นติตี้ทุกเอ็นติตี้ที่อยู่ในสามเอ็นเอฟ ให้เป็น บอยด์-คอดเอ็นเอฟ

7.5 ลดรูปเอ็นติตี้ทุกเอ็นติตี้ที่อยู่ในบอยด์-คอดเอ็นเอฟ ให้เป็น สี่เอ็นเอฟ

7.6 ลดรูปเอ็นติตี้ทุกเอ็นติตี้ที่อยู่ในสี่เอ็นเอฟ ให้เป็น ห้าเอ็นเอฟ

7.7 โดยทั่วไปแล้ว ห้ามไม่ให้ทำการแบ่งเอ็นติตี้ที่อยู่ในรูปแบบของนอร์มอลไลซ์แล้ว ให้เป็นเอ็นติตี้เล็ก ๆ อีก

7.8 ทำการประเมินค่าของเอ็นติตี้ที่ได้ทำการนอร์มอลไลซ์ว่า เป็นเอ็นติตี้ที่อ่อนด้อยในการเพิ่มหรือลบข้อมูล และต้องพิจารณาเกี่ยวข้องกับเวลาด้วยหรือไม่

ขั้น LDM 8 : กำหนด โดเมน

8.1 หาโดเมนให้กับแต่ละ แอตทริบิว

8.2 จดบันทึก โดเมน รวมทั้ง ประเภทข้อมูล, ความยาว, รูปแบบ และหน้ากาก, ค่าที่ใช้หรือยอมให้แอตทริบิวเป็นได้, ความหมาย, การมีค่าที่ซ้ำกันได้หริอไม่, สนับสนุนการไม่มีค่าข้อมูลหรือไม่, และค่าตั้งต้น

8.3 กำหนดโดเมน สำหรับ คีย์หลัก

8.4 กำหนดโดเมน สำหรับ คีย์เลือก

8.5 กำหนดโดเมน สำหรับ คีย์นอก

8.6 กำหนดโดเมน สำหรับ แอตทริบิวที่ได้จากการสังเคราะห์

8.7 กำหนดโดเมน สำหรับ คีย์หลักของ subtypes ให้เป็นเซทย่อยของ โดเมนของคีย์หลักของ supertypes

ขั้น LDM 9 : กำหนดกฎเกณฑ์ทางธุรกิจสำหรับ แอตทริบิวที่ไม่ใช้คีย์

9.1 กำหนดทุกกฎทางธุรกิจและ ทริกเกอร์ (ทริกเกอร์ หมายถึง เหตุการณ์, วัตถุของเหตุการณ์, และเงื่อนไข) ที่ใช้เพื่อสำหรับรักษาความสมบูรณ์และความถูกต้องของข้อมูล

9.2 ทำการจดบันทึกกฎทางธุรกิจ รวมทั้งทริกเกอร์ และการกระทำการ

9.3 กำหนด ทริกเกอร์โอเปอร์เรชั่น สำหรับทุก แอตทริบิว ที่เป็นแหล่งข้อมูลสำหรับแอตทริบิวที่เป็นแอตทริบิวสังเคราะห์(derived attributes)

9.4 กำหนด ทริกเกอร์ให้เอ็นติตี้ที่เป็น subtypes เช่น เมื่อข้อมูลที่อยู่ในเอ็นติตี้ที่เป็น subtypes ถูกลบแล้วให้ไปลบข้อมูลที่อยู่ใน เอ็นติตี้ที่เป็น supertypes ด้วย

9.5 กำหนด ทริกเกอร์สำหรับ การควบคุมความสมบูรณ์ของข้อมูลที่เกี่ยวกับเวลา

ขั้น LDM 10 : รวมมุมมองของผู้ใช้

10.1 รวมทุกเอ็นติตี้ที่มี คีย์หลัก และ โดเมนหลักเหมือนกันเป็นเอ็นติตี้เดียว

10.2 สร้างความสัมพันธ์ในลักษณะ supertypes-subtypes ในกรณีที่มี คีย์หลักเหมือนกัน แต่ โดเมนของเอ็นติตี้หนึ่งเป็น เซทย่อยของโดเมนของอีก เอ็นติตี้

10.3 สร้าง common supertypes เพื่อสร้างความสัมพันธ์ระหว่างเอ็นติตี้ 2 เอ็นติตี้ที่มี คีย์หลักเหมือนกัน และมีโดเมนของคีย์หลักที่เท่าเทียมกัน

10.4 รวมเอ็นติตี้ที่มี คีย์หลักทำหน้าที่เป็นคีย์คู่แข่งของอีกเอ็นติตี้หนึ่ง

10.5 รวมทุกเอ็นติตี้ที่มี คีย์หลักแตกต่างกัน ของแต่ละมุมมองเข้าเป็น LDM ใหม่

10.6 หลังจากการรวมเอ็นติตี้แล้ว ต้องแน่ใจว่ากฎทางธุรกิจสำหรับคีย์คู่แข่ง ต้องไม่ถูกทำลายลง

10.7 รวมความสัมพันธ์ระหว่างเอ็นติตี้เฉพาะที่มีความหมายเหมือนกันและตัวเอ็นติตี้เองเป็นผลมาจากการรวม

10.8 รวมความสัมพันธ์ของทุกมุมมองที่มีความหมายแตกต่างกันและให้ขจัดความสัมพันธ์ที่ซ้ำซ้อนออกจาก LDM ใหม่

10.9 ค้นหาว่ามีการตกหล่นของความสัมพันธ์จากแต่ละมุมมองหรือไม่

10.10 ปรับแต่งให้ คีย์นอก ทุกตัวที่เกิดจากการรวมมุมมองว่ามีผลกระทบกับ คีย์หลัก หรือไม่

10.11 เริ่มติดตั้งและรวมกฎทางธุรกิจ เหมือนกับที่เริ่มใช้ในมุมมองหลัก พร้อมกับเพิ่มกฎทางธุรกิจให้กับความสัมพันธ์ใหม่ เพื่อแก้ปัญหาความไม่คงที่ของข้อมูล

10.12 รวมแอตทริบิวที่มีความหมายเหมือนกันใน เอ็นติตี้เดียวกัน

10.13 กำจัด Flag ที่สังเคราะห์มาจากแอตทริบิว

10.14 ทำนอร์มอลไลซ์ เพื่อกำจัดความซ่ำซ้อนใหม่ที่เกิดขึ้น

10.15 ตรวจสอบแอตทริบิว เพื่อระบุถึงการเปลี่ยนแปลงคุณลักษณะของโดเมนท์

ขั้น LDM 11 : รวบรวมและสร้างโมเดลที่สมบูรณ์จากโมเดลที่ มีอยู่

11.1 รวบรวมและสร้างฐานข้อมูล โดยเปรียบเทียบกับ แผนผังที่กำหนดขึ้นภายใต้ความเข้าใจในเรื่อง logical data model

11.2 สร้าง business conceptual schema โดยใช้ logical data model ที่รวบรวมขึ้นใหม่นี้

11.3 ระบุ ความสัมพันธ์ระหว่าง logical data model กับ business conceptual schema

ขั้น LDM 12 : วิเคราะห์ความคงที่และการเจริญเติบโต

12.1 รวบรวมและจดบันทึก การที่เปลี่ยนแปลงที่ใกล้เข้ามา และความสำคัญ และหรือ ความแน่นอนของการเปลี่ยนแปลงนั้น