บทที่ 2

คลังข้อมูลและการค้นหาข้อมูล

สำหรับบทนี้ กล่าวถึงคุณสมบัติของคลังข้อมูล ความแตกต่างระหว่างฐานข้อมูลดำเนินการและคลังข้อมูล สถาปัตยกรรมของคลังข้อมูล แบบจำลองเชิงแนวคิดของคลังข้อมูล และการเข้าถึงข้อมูล

เนื่องจากองค์กรมีข้อมูลเป็นจำนวนมาก แต่ไม่สามารถเข้าถึงได้ ดังนั้นองค์กรจึงต้องพยายามหาเทคนิควิธีที่ทำให้ผู้ใช้สามารถได้รับข้อมูลโดยตรง ซึ่งต้องเป็นวิธีการที่ใช้ง่ายและมีความสะดวกรวดเร็ว วิธีการหนึ่งที่นิยมกันก็คือ การทำคลังข้อมูล

เทคโนโลยีคลังข้อมูล เป็นกลุ่มของเทคโนโลยีที่จัดเตรียมสถาปัตยกรรมและเครื่องมือในการรวบรวมและทำความเข้าใจข้อมูลอย่างเป็นระบบ เพื่อใช้เป็นกลยุทธ์ในการสนับสนุนการตัดสินใจของบุคคลที่ทำงานเกี่ยวกับความรู้ เช่น ผู้บริหาร ผู้จัดการ นักวิเคราะห์ ให้สามารถทำการตัดสินใจได้ดีและรวดเร็วขึ้น [12]

คลังข้อมูล เป็นฐานข้อมูลที่ถูกปรับเปลี่ยนให้แตกต่างจากฐานข้อมูลดำเนินการทั่วไป โดยระบบคลังข้อมูลจะอนุญาตให้มีการรวมกันของข้อมูลจากระบบที่แตกต่างกันได้ และสนับสนุนการประมวลผลข้อมูลในอดีต เป้าหมายของคลังข้อมูล คือ เป็นที่เก็บข้อมูลที่มีขนาดใหญ่เพื่อช่วยในการสนับสนุนการตัดสินใจ โดยคลังข้อมูลจะต้องแสดงให้เห็นชัดเจนว่าอะไรสำคัญและข้อมูลในคลังข้อมูลจะต้องมีความสอดคล้องตรงกัน ผู้ใช้จะต้องได้รับข้อมูลเพื่อสนับสนุนการตัดสินใจมากกว่าข้อเท็จจริง

W.H. Inmon ซึ่งเป็นผู้ริเริ่มการสร้างระบบคลังข้อมูล ให้นิยามไว้ว่า “คลังข้อมูล คือ กลุ่มของข้อมูลที่ช่วยในการสนับสนุนการตัดสินใจ ซึ่งมีสมบัติดังต่อไปนี้ [2]

- **การกำหนดทิศทางหรือมุ่งเน้นไปที่หัวข้อ (Subject-oriented)**

คลังข้อมูลถูกจัดการโดยมุ่งเน้นไปที่หัวเรื่องเป็นหลัก เช่น ลูกค้า, ผู้ขาย, สินค้า และการขาย ซึ่งมีการรวบรวมข้อมูลการดำเนินงานในแต่ละวันและการประมวลผลทรานแซคชันขององค์กรเข้าไว้ด้วยกัน โดยเลือกเอาเฉพาะข้อมูลที่เป็นประโยชน์ในการสนับสนุนการตัดสินใจมาเก็บไว้ในคลังข้อมูลเท่านั้น แล้วจัดการให้มีรูปแบบที่ง่ายและรัดกุมเหมาะสำหรับการวิเคราะห์เพื่อสนับสนุนการตัดสินใจภายใต้หัวข้อเรื่องที่ผู้ใช้สนใจ

- **การรวมเข้าด้วยกันเป็นหนึ่งเดียว (Integrated)**

คลังข้อมูลถูกสร้างขึ้นโดยการรวบรวมข้อมูลจากหลากหลายแหล่งที่มา เช่น ฐานข้อมูลเชิงสัมพันธ์ ไฟล์ข้อมูลทั่วไป และรายการทรานแซคชันออนไลน์ (On-Line

Transaction Record) มารวมเข้าไว้ด้วยกัน ซึ่งข้อมูลจากแต่ละแหล่งที่มามีความแตกต่างกัน ไม่สอดคล้องกัน เช่น การตั้งชื่อ การกำหนดค่า โครงสร้างการเข้ารหัส หน่วยการวัด เป็นต้น ดังนั้นก่อนที่จะมีการนำข้อมูลจากแหล่งต่าง ๆ มาเก็บไว้ในคลังข้อมูล จะต้องมีเทคนิคการทำความสะอาดข้อมูล และแปลงข้อมูล เพื่อให้ข้อมูลมีความสอดคล้องกัน และมีรูปแบบที่เป็นหนึ่งเดียวกัน

- **มีเวลาเข้ามาเกี่ยวข้อง (Time-variant)**

ในโครงสร้างของคลังข้อมูลจะต้องประกอบด้วยเวลาเป็นส่วนสำคัญเสมอ เนื่องจากคลังข้อมูลจะต้องจัดเตรียมข้อมูลที่เป็นอดีตเอาไว้ ซึ่งเป็นสิ่งที่จำเป็นสำหรับการสนับสนุนการตัดสินใจเพื่อดูแนวโน้มของข้อมูล โดยข้อมูลที่ถูกเก็บไว้มักเป็นข้อมูลย้อนหลังไปในอดีตเป็นเวลา 5 ปี หรือ 10 ปี ตรงกันข้ามกับฐานข้อมูลดำเนินการทั่วไปที่มีการรวบรวมข้อมูลที่เป็นปัจจุบัน หรือมีการเก็บข้อมูลไว้เป็นช่วงเวลาที่ไม่นาน ดังนั้นจึงอาจไม่มีเวลาเป็นองค์ประกอบด้วยก็ได้

- **ข้อมูลมีความเสถียรหรือไม่เปลี่ยนแปลงได้ง่าย (Nonvolatile)**

การดำเนินการบนคลังข้อมูลมี 2 แบบเท่านั้น คือ การโหลดข้อมูลเริ่มต้น และการเข้าถึงข้อมูล โดยไม่มีการแก้ไขและการเขียนข้อมูลซึ่งเป็นการดำเนินการที่เกิดขึ้นบนฐานข้อมูลดำเนินการทั่วไป ทำให้ข้อมูลบนคลังข้อมูลมีความเสถียร ดังนั้นจึงไม่จำเป็นต้องมีกลไกในการกู้คืน การควบคุมความสอดคล้องตรงกัน และการเกิดขึ้นพร้อมกัน เช่น การอ่านพร้อมการเขียน หรือการอ่านพร้อมการเปลี่ยนแปลง

ในปัจจุบันเราพบว่า เทคโนโลยีคลังข้อมูลได้ถูกนำไปประยุกต์ใช้ในด้านต่าง ๆ ดังนี้[12]

- ด้านการผลิต ได้แก่ การวิเคราะห์รายการสินค้าที่ส่งไป การสนับสนุนลูกค้า
- ด้านการค้า ได้แก่ การวิเคราะห์ประวัติลูกค้า การวิเคราะห์รายการสินค้า
- ด้านการเงิน ได้แก่ การวิเคราะห์การอ้างสิทธิ์ (Claim) การวิเคราะห์ความเสี่ยง การตรวจจับการโกง
- ด้านการขนส่ง ได้แก่ การวิเคราะห์และจัดการยานพาหนะ
- ด้านการสื่อสาร ได้แก่ การวิเคราะห์การติดต่อ การตรวจจับการโกง
- ด้านการใช้ประโยชน์ให้คุ้มค่า ได้แก่ การวิเคราะห์การใช้พลังงาน
- ด้านการดูแลสุขภาพ ได้แก่ การวิเคราะห์ผลสุขภาพ

2.1 ฐานข้อมูลดำเนินการและคลังข้อมูล (Operational Database vs. Data Warehouse)

ในหัวข้อนี้ กล่าวถึงความแตกต่างระหว่างฐานข้อมูลดำเนินการและคลังข้อมูล ทั้งในด้านการใช้งาน โครงสร้างการออกแบบ และลักษณะทั่วไปที่สำคัญ ซึ่งทำให้เราทราบว่า

"เหตุใดคลังข้อมูลจึงต้องแยกจากฐานข้อมูลดำเนินการ" และ "เหตุใดจึงไม่ทำการวิเคราะห์ข้อมูลโดยตรงบนฐานข้อมูลดำเนินการ แทนที่จะต้องเสียค่าใช้จ่ายและทรัพยากรเพิ่มในการสร้างคลังข้อมูล"

2.1.1 ฐานข้อมูลดำเนินการ (Operational Database)

การทำงานหลักของระบบฐานข้อมูลดำเนินการ คือ การประมวลผลทรานแซคชันและการสอบถามแบบออนไลน์ ซึ่งเราเรียกระบบนี้ว่า OLTP (On-Line Transaction Processing) เป็นระบบที่ครอบคลุมการดำเนินการประจำวันขององค์กร เช่น การจัดซื้อ คลังสินค้า การผลิต การธนาคาร ค่าจ้าง และการลงทะเบียน เป็นต้น ผู้ใช้งานคือ เสมียน ลูกค้า หรือผู้เชี่ยวชาญทางด้านข้อมูล

2.1.2 คลังข้อมูล (Data Warehouse)

การทำงานหลักของคลังข้อมูล คือ การวิเคราะห์ข้อมูลและสนับสนุนการตัดสินใจ โดยมีการรวบรวมข้อมูลจากหลายแหล่งที่มาที่อาจมีรูปแบบต่างกัน นำมาปรับให้มีความสอดคล้องตรงกันเก็บไว้ในที่เดียวกัน เพื่อรองรับความต้องการที่แตกต่างกันของผู้ใช้ ซึ่งเราเรียกระบบนี้ว่า OLAP (On-Line Analytical Processing) ผู้ใช้งานคือ ผู้บริหาร นักวิเคราะห์

การเปรียบเทียบความแตกต่างระหว่างระบบฐานข้อมูลดำเนินการและคลังข้อมูล แบ่งตามลักษณะต่าง ๆ สามารถพิจารณาได้ดังตาราง 2-1 [2,12,13]

ตาราง 2-1 แสดงการเปรียบเทียบฐานข้อมูลดำเนินการและคลังข้อมูล

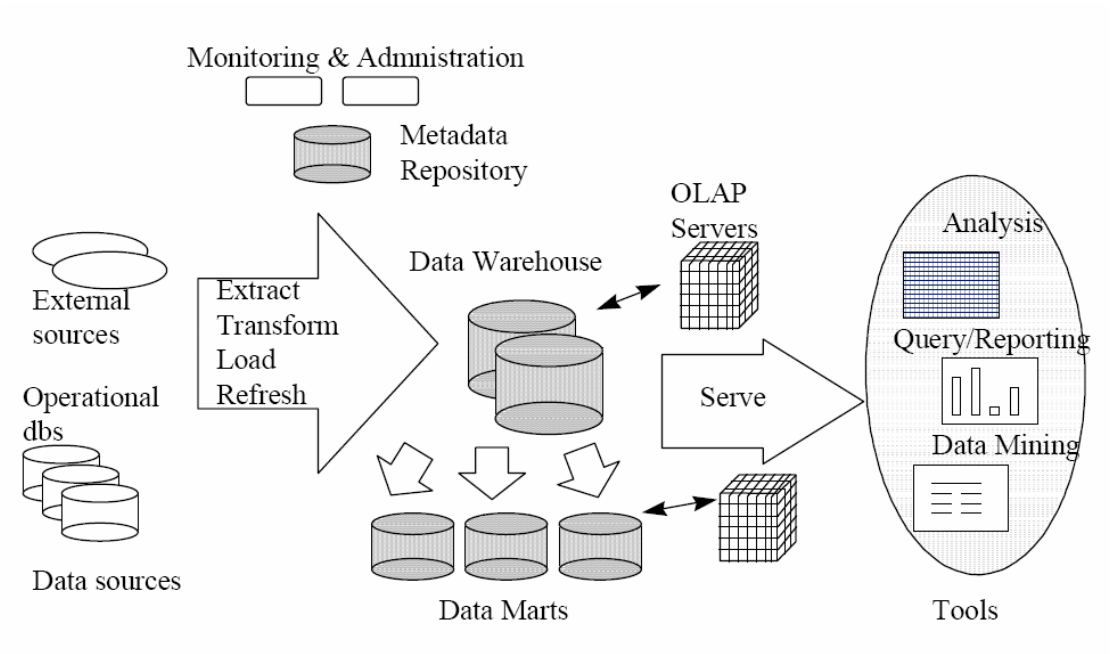
ลักษณะ	ฐานข้อมูลดำเนินการ	คลังข้อมูล
การประมวลผล	- แบบ OLTP	- แบบ OLAP
การกำหนดทิศทาง	- มุ่งเน้นไปที่ลูกค้าเป็นสำคัญ	- มุ่งเน้นด้านการตลาด และหัวเรื่องในการวิเคราะห์เป็นสำคัญ
ผู้ใช้ระบบ	- เสมียน นักธุรกิจ ผู้ออกแบบระบบ	- ผู้จัดการ ผู้บริหาร นักวิเคราะห์
การทำงาน	- การดำเนินงานประจำวัน เป็นการทำงานแบบซ้ำ ๆ ใช้เวลาไม่นาน	- การดำเนินงานวิเคราะห์ข้อมูลเพื่อสนับสนุนการตัดสินใจ จึงต้องการข้อมูลในช่วงระยะเวลานาน
ข้อมูล	- ข้อมูลปัจจุบันที่ทันสมัย - ข้อมูลมีการเปลี่ยนแปลงได้ - มุ่งประเด็นนำข้อมูลดิบ(Data)เข้า	- ข้อมูลในอดีตจนถึงปัจจุบันที่มีความถูกต้องแม่นยำในช่วงเวลานาน ๆ - ข้อมูลไม่มีการเปลี่ยนแปลง - มุ่งประเด็นนำข้อมูล(Information)ออก
ขนาดของฐานข้อมูล	- 100 MB ถึง GB	- 100 GB ถึง TB
การออกแบบฐานข้อมูล	- ใช้แบบจำลองอี-อาร์ (E-R Model)	- ใช้แบบจำลองหลายมิติ (Multidimensional Model)
การมอง (View)	- แสดงรายละเอียดหน่วยเล็กที่สุด - โครงสร้างข้อมูลไม่เปลี่ยนแปลง แต่ข้อมูลที่อยู่ข้างในมีการเปลี่ยนแปลงได้	- แสดงข้อมูลที่มีการสรุปไว้ในลักษณะหลายมิติ - โครงสร้างมีการเปลี่ยนแปลงได้ แต่ข้อมูลที่อยู่ข้างในไม่มีการเปลี่ยนแปลง
การเข้าถึง	- อ่าน/เขียน - ต้องมีกลยุทธ์ในการควบคุมความสอดคล้องตรงกัน และกู้คืน - จำนวนเรคอร์ดที่ถูกเข้าถึงเป็นสิบ - จำนวนผู้เข้าถึงข้อมูลเป็นพัน	- ส่วนมากเป็นการอ่าน - ไม่ต้องมีกลยุทธ์ในการควบคุมความสอดคล้องตรงกันและกู้คืน - จำนวนเรคอร์ดที่ถูกเข้าถึงเป็นล้าน - จำนวนผู้เข้าถึงข้อมูลเป็นร้อย

ตาราง 2-1 แสดงการเปรียบเทียบฐานข้อมูลดำเนินการและคลังข้อมูล (ต่อ)

ลักษณะ	ฐานข้อมูลดำเนินการ	คลังข้อมูล
การสอบถาม	- เป็นการสอบถามแบบสั้น ๆ ง่าย ๆ - มีการจัดเตรียมไว้ล่วงหน้า	- เป็นการสอบถามที่ซับซ้อน - เป็นแบบทันทีทันใด ไม่ทราบล่วงหน้าว่าคำถามจะเป็นเช่นไร
การสรุปความ	- ง่าย ๆ ไม่ซับซ้อน ต้องการรายละเอียดมากและมีความถูกต้องแม่นยำในขณะที่มีการเข้าถึง	- มีการสรุปข้อมูลในอดีตไว้ให้ใช้และสามารถแสดงข้อมูลในอดีตได้อย่างรวดเร็ว
มาตรวัดประสิทธิภาพ	- ปริมาณทรานแซคชัน	- ปริมาณการสอบถาม เน้นเวลาในการตอบสนอง

เราจะเห็นว่า ฐานข้อมูลดำเนินการและคลังข้อมูลมีความแตกต่างกันหลายประการ เหตุผลหลักที่เราจะต้องแยกทั้งสองระบบออกจากกัน คือ เพื่อเพิ่มความสามารถในการทำงานของทั้งสองระบบ เนื่องจากความแตกต่างทางด้านโครงสร้างและสิ่งที่บรรจุอยู่ข้างใน โดยฐานข้อมูลดำเนินการถูกออกแบบและปรับให้สอดคล้องกับภาระงาน มีการจัดเตรียมความสามารถในการทำดัชนีและแฮชบนคีย์หลัก และมีการค้นหาข้อมูลเป็นเรคอร์ด ซึ่งการสอบถามได้ถูกจัดเตรียมไว้ล่วงหน้าก่อนแล้ว ส่วนในคลังข้อมูลนั้น การสอบถามจะมีความซับซ้อนและเป็นแบบทันทีทันใด และต้องมีการเข้าถึงข้อมูลเป็นล้านเรคอร์ด นอกจากนั้นยังต้องมีการสแกนและการใช้เงื่อนไขที่ซับซ้อนเป็นจำนวนมาก ซึ่งในบางครั้งต้องมีวิธีการพัฒนาระบบ การเข้าถึง และการจัดการแบบพิเศษบนข้อมูลที่มีลักษณะเป็นหลายมิติ เพื่อให้การประมวลผลของระบบคลังข้อมูลมีประสิทธิภาพมากขึ้น ดังนั้นหากเราดำเนินการวิเคราะห์ข้อมูลแบบออนไลน์โดยตรงบนฐานข้อมูล (OLAP Operation) พร้อมกับการดำเนินการประมวลผลทรานแซคชัน (OLTP Operation) บนฐานข้อมูลดำเนินการแล้ว จะก่อให้เกิดความเสียหายอย่างรุนแรง และเป็นการลดประสิทธิภาพของทั้งระบบ OLTP และ OLAP ด้วย

2.2 สถาปัตยกรรมของคลังข้อมูล



ภาพประกอบ 2-1 แสดงสถาปัตยกรรมของคลังข้อมูล(Architecture of Data Warehouse)

สถาปัตยกรรมของคลังข้อมูลแสดงดังภาพประกอบ 2-1 แบ่งเป็น 4 ส่วนหลัก คือ [12]

- **แหล่งข้อมูล (Data Sources)**

แหล่งข้อมูลที่นำมาใช้ในการสร้างคลังข้อมูล มาจากแหล่งข้อมูลภายนอก (External Source) และฐานข้อมูลดำเนินการ (Operational Database) โดยได้มาจากหลายแหล่งที่มา เช่น ไฟล์ข้อมูลประวัติการสั่งสินค้า ฐานข้อมูลของสาขาต่าง ๆ เป็นต้น

- **หน่วยเก็บข้อมูล (Data Storage)**

หน่วยเก็บข้อมูลประกอบด้วย คลังข้อมูล (Data Warehouse) คลังข้อมูลย่อย (Data Mart) และหน่วยจัดการความรู้ (Metadata Repository) เมื่อมีการดึงข้อมูล (Extracting) จากแหล่งข้อมูลมาทำความสะอาด (Data Cleaning) เพื่อเปลี่ยนรูปข้อมูล (Transforming) ให้สอดคล้องตรงกันแล้ว หลังจากนั้นจะเป็นการโหลดข้อมูล (Load) เข้าสู่คลังข้อมูล และเมื่อข้อมูลจากแหล่งที่มามีการเปลี่ยนแปลงก็จะต้องทำข้อมูลในคลังข้อมูลให้ใหม่ (Refresh) ด้วย

o *การดึง การทำความสะอาด และการเปลี่ยนรูปข้อมูล (Extracting, Cleaning, Transforming)* เนื่องจากมีการดึงข้อมูลจากหลากหลายแหล่งที่มาเพื่อ

เก็บไว้ในคลังข้อมูล จึงทำให้ข้อมูลมีปริมาณมาก ซึ่งข้อมูลจากแหล่งที่แตกต่างกัน ย่อมมีโครงสร้างและลักษณะที่แตกต่างกัน และมีโอกาสพบข้อมูลที่ผิดได้สูง ด้วยเหตุที่ว่าคลังข้อมูลจะต้องถูกนำไปใช้ในการสนับสนุนการตัดสินใจ ข้อมูลในคลังข้อมูลจึงจำเป็นต้องมีความถูกต้อง ดังนั้นจึงต้องมีเครื่องมือในการช่วยตรวจจับความผิดปกตินั้นและทำการแก้ไขให้ถูกต้อง ได้แก่ การทำความสะอาดและเปลี่ยนรูปข้อมูลให้อยู่ในรูปแบบที่ตรงกัน ซึ่งจะต้องเสียค่าใช้จ่ายมาก เช่น การทำให้ข้อมูลอยู่ในช่วงค่าที่กำหนด การทำให้ชนิดของข้อมูลสอดคล้องกัน การจัดการข้อมูลส่วนที่สูญหายไป และการจัดการเมื่อมีการฝ่าฝืนข้อจำกัด เป็นต้น

- *การโหลดข้อมูล (Load)* หลังจากที่ผ่านการดึง การทำความสะอาด และเปลี่ยนรูปข้อมูลเรียบร้อยแล้ว ต่อไปเป็นขั้นตอนของการนำข้อมูลเข้าสู่คลังข้อมูล ซึ่งมีขั้นตอนที่ต้องทำก่อน เช่น ตรวจสอบความถูกต้องของข้อมูล เรียงลำดับ การคำนวณต่าง ๆ เพื่อนำมาสร้างเป็นตารางเก็บไว้ในคลังข้อมูล การสร้างดัชนีและเส้นทางการเข้าถึงข้อมูล เป็นต้น การโหลดข้อมูลสำหรับคลังข้อมูลต้องจัดการข้อมูลที่มีขนาดใหญ่กว่าฐานข้อมูลดำเนินการมาก ดังนั้นจึงต้องเลือกทำในเวลาที่ไม่สำคัญ เช่น ช่วงกลางคืน ซึ่งเป็นช่วงเวลาที่คลังข้อมูลออฟไลน์ ถ้ามีการโหลดแบบลำดับ (Sequential) จะต้องใช้เวลานานมากอาจเป็นสัปดาห์หรือเป็นเดือน จึงมีการทำงานแบบขนานเพื่อให้โหลดได้เร็วขึ้น ในขณะที่เรากำลังโหลดอยู่นั้น ฐานข้อมูลที่ใช้อยู่ในปัจจุบันจะต้องสามารถสนับสนุนการสอบถามได้ตามปกติ และถ้าในการโหลดนั้นมีการกำหนด เช็คพอยน์ไว้ ก็จะเป็นการรับประกันว่าเมื่อมีระบบมีความล้มเหลวเกิดขึ้น และทำการเริ่มต้นกระบวนการใหม่ก็สามารถเริ่มทำที่เช็คพอยน์ได้เลย โดยไม่ต้องทำใหม่ตั้งแต่จุดเริ่มต้น
- *การทำให้ข้อมูลใหม่ (Refresh)* จะต้องพิจารณา 2 ประเด็น คือ “จะทำการ refresh เมื่อไร” และ “จะทำการ refresh อย่างไร” คลังข้อมูลจะถูกทำให้ข้อมูลใหม่เป็นช่วง ๆ ไม่จำเป็นต้องทำบ่อย ๆ เพราะการสอบถามแบบ OLAP มักไม่ต้องการข้อมูลปัจจุบัน การทำให้ข้อมูลใหม่นั้นจะขึ้นอยู่กับการควบคุมของผู้ดูแลคลังข้อมูลซึ่งแตกต่างกันไปตามความต้องการ

ในบางครั้งการใช้งานจากคลังข้อมูลที่มีขนาดใหญ่โดยตรง อาจทำได้ไม่สะดวกนักและอาจเกินความจำเป็นสำหรับบางหน่วยงานที่ต้องการใช้เพียงข้อมูลของตนเองเท่านั้น จึงต้องมีการแยกจัดเก็บข้อมูลออกเป็นคลังข้อมูลย่อย (Data Mart) ให้ตรงกับการทำงานของแต่ละส่วน เช่น คลังข้อมูลแผนกการขาย คลังข้อมูลแผนกการผลิต เป็นต้น หน่วยเก็บข้อมูลที่สำคัญอีกหน่วยหนึ่งก็คือ หน่วยจัดการความรู้ (Metadata Repository) ทำหน้าที่จัดเก็บข้อมูลเกี่ยวกับเครื่องมือที่ใช้ในการควบคุมระบบคลังข้อมูล โดยมีผู้สร้างระบบเป็นผู้ดูแล

- OLAP Servers

OLAP Server ทำหน้าที่ในการจัดเตรียมการเข้าถึงข้อมูลเพื่อให้การประมวลมีความรวดเร็ว ได้แก่ การทำดัชนี การทำ Materialize View การเปลี่ยนรูปการสอบถามที่ซับซ้อน และการประมวลผลแบบขนานเพื่อลดเวลาในการตอบคำถาม นอกจากนั้นยังจัดเตรียม

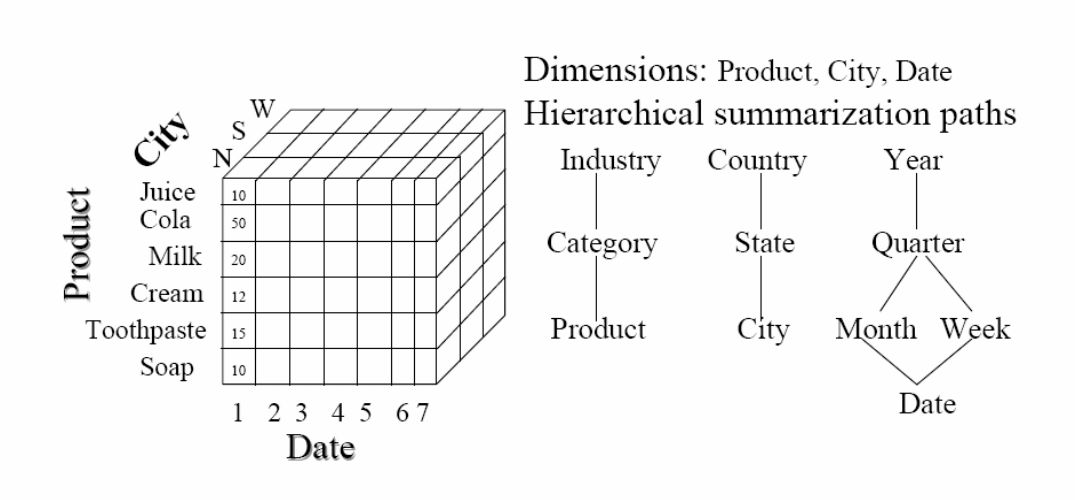
ข้อมูลที่สำคัญสำหรับการวิเคราะห์เอาไว้ด้วย เช่น ค่าเฉลี่ย ผลรวม โดยในคลังข้อมูลอาจใช้มากกว่า 1 Server ก็ได้

- เครื่องมือสำหรับผู้ใช้ (Front-End-Tools)

เราสามารถมองมิติ (Dimension) ของข้อมูลได้แตกต่างกันขึ้นอยู่กับเครื่องมือที่ใช้ ได้แก่ เครื่องมือในการวิเคราะห์ข้อมูล(Analysis) เครื่องมือสำหรับสอบถาม (Query) เครื่องมือสำหรับออกรายงาน (Report) และเครื่องมือในการทำเหมืองข้อมูล (Data Mining)

2.3 แบบจำลองเชิงแนวคิดของคลังข้อมูล(Conceptual Model of Data Warehouse)

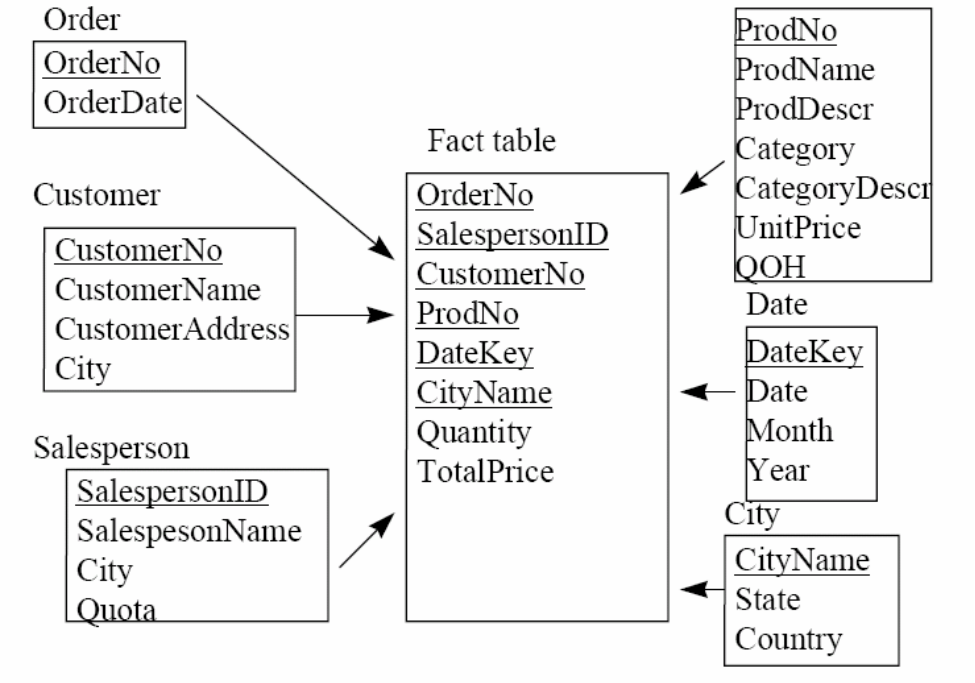
แนวคิดเกี่ยวกับแบบจำลองข้อมูลจะมีผลต่อการออกแบบฐานข้อมูล การสอบถาม และเครื่องมือในสำหรับผู้ใช้ ในการออกแบบฐานข้อมูลเชิงสัมพันธ์นิยมใช้แบบจำลองอี-อาร์ (Entity-Relationship Model) ซึ่งจะประกอบด้วยเซตของเอนติตี้และความสัมพันธ์ระหว่างเอนติตี้ แบบจำลองนี้เหมาะสำหรับการประมวลผลทรานแซคชันแบบออนไลน์ (On-Line Transaction Processing) แต่ไม่เหมาะสำหรับระบบสนับสนุนการตัดสินใจที่ต้องการประสิทธิภาพในการสอบถามที่ซับซ้อนและโหลดข้อมูลสูง แบบจำลองของคลังข้อมูลจะต้องมีความรัดกุม มุ่งเน้นไปที่หัวเรื่องที่สนใจเพื่อการวิเคราะห์ข้อมูลแบบออนไลน์สะดวกรวดเร็วขึ้น แบบจำลองของคลังข้อมูลที่นิยมใช้กัน คือ แบบจำลองข้อมูลหลายมิติ (Multidimensional Data Model) ซึ่งประกอบด้วยส่วนที่เป็นตัววัด (Numeric Measure) เพื่อใช้ในการวิเคราะห์ ตัวอย่างเช่น เงินลงทุน ยอดขาย และผลกำไร เป็นต้น แต่ละการวัดจะขึ้นกับกลุ่มของมิติ (Dimension) ที่จัดเตรียมไว้สำหรับการวัด ตัวอย่างเช่น มิติที่สัมพันธ์กับยอดขาย ได้แก่ เมือง (City) สินค้า (Product) วันที่ (Date) ซึ่งแต่ละมิติจะแสดงแอทริบิวต์ต่าง ๆ ที่อยู่ในมิตินั้น อาทิเช่น มิติสินค้า (Product Dimension)ประกอบด้วย 4 แอทริบิวต์ คือ กลุ่มสินค้า ประเภทสินค้า ปีที่นำเข้า และกำไรเฉลี่ย เช่น โซดา เป็นสินค้ากลุ่มเครื่องดื่ม ประเภทอาหาร นำเข้าปี 1996 และมีกำไรเฉลี่ย 40 % เป็นต้น ในบางครั้งแอทริบิวต์ของ dimension จะสัมพันธ์กันแบบลำดับขั้น(Hierarchy) ดังภาพประกอบ 2-2 จะเห็นว่า ประเทศ (Country) รัฐ (State) และเมือง (City) มีความสัมพันธ์กันแบบลำดับขั้น นอกจากนั้นในแบบจำลองเชิงแนวคิดสำหรับคลังข้อมูลยังอนุญาตให้มีการรวมกันของการวัดไว้แบบเป็นลำดับขั้นด้วย เช่น ผลรวมของยอดขายในแต่ละเมืองหรือในแต่ละปี เป็นต้น โดยส่วนใหญ่แล้วบนคลังข้อมูลจะต้องมีมิติของเวลา (Time) อยู่ด้วย เพราะเวลาเป็นมิติที่สำคัญต่อการสนับสนุนการตัดสินใจ เพราะเราจะต้องนำไปใช้ในการวิเคราะห์ทิศทางของข้อมูลตามกาลเวลา ในที่นี้จะกล่าวถึงแบบจำลองข้อมูลหลายมิติที่นิยม 2 แบบด้วยกัน



ภาพประกอบ 2-2 แสดงข้อมูลหลายมิติ (Multidimensional Data)[12]

2.3.1 โครงสร้างแบบดาว (Star Schema)

โครงสร้างแบบดาว ประกอบด้วยตารางข้อเท็จจริง(Fact Table) อยู่ตรงกลาง ล้อมรอบด้วยตารางมิติ(Dimension Table) ดังภาพประกอบ 2-3

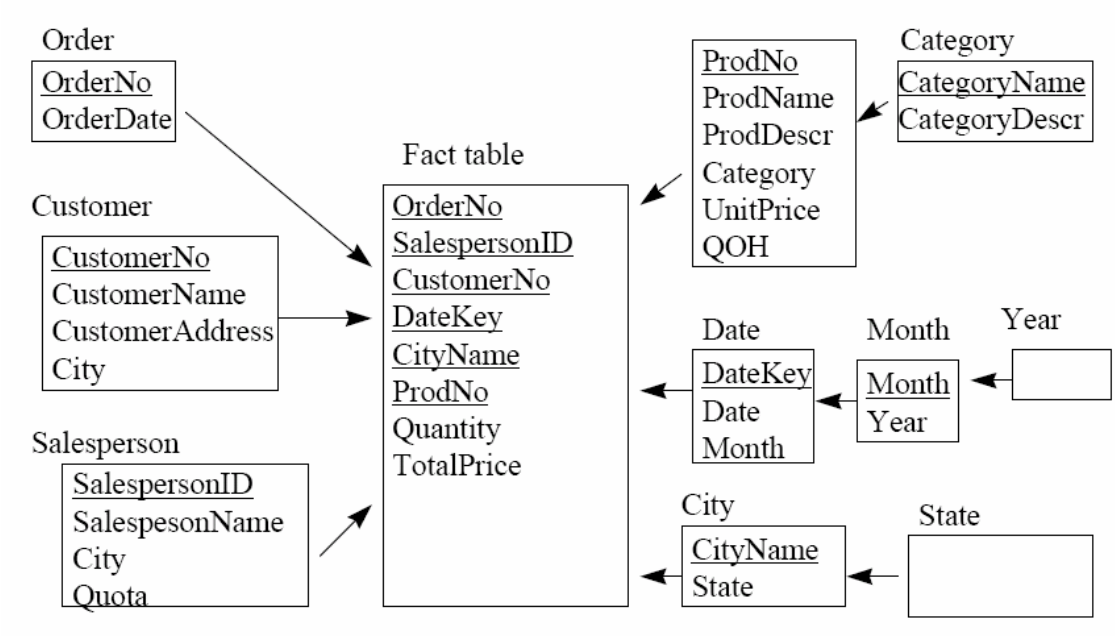


ภาพประกอบ 2-3 แสดงโครงสร้างแบบดาว (Star Schema)[12]

ตารางข้อเท็จจริง ประกอบด้วย คีย์ (Fact Table Key) ที่ใช้เชื่อมโยงไปยังตารางมิติต่าง ๆ จำนวนคอลัมน์ของคีย์จะเพิ่มขึ้นตามจำนวนของตารางมิติ ตัวอย่างเช่น จากภาพประกอบ 2-3 มีตารางมิติ 6 ตาราง จำนวนคอลัมน์ของคีย์จึงเท่ากับ 6 อีกส่วนหนึ่งที่สำคัญที่อยู่ในตารางข้อเท็จจริง คือ ตัววัด (Measure) จะเก็บข้อมูลที่เป็นตัวเลขของสิ่งที่เราสนใจวัด ซึ่งในที่นี้คือ Quantity และ TotalPrice

2.3.2 โครงสร้างผลึกหิมะ (Snowflake Schema)

โครงสร้างแบบผลึกหิมะ มีตารางข้อเท็จจริงอยู่ตรงกลางเช่นเดียวกันกับโครงสร้างแบบดาว แต่ตารางมิติที่ล้อมรอบจะมีการเชื่อมโยงไปยังตารางย่อย ๆ ต่อไปอีกหลายระดับ



ภาพประกอบ 2-4 แสดงโครงสร้างแบบผลึกหิมะ (Snowflake Schema)[12]

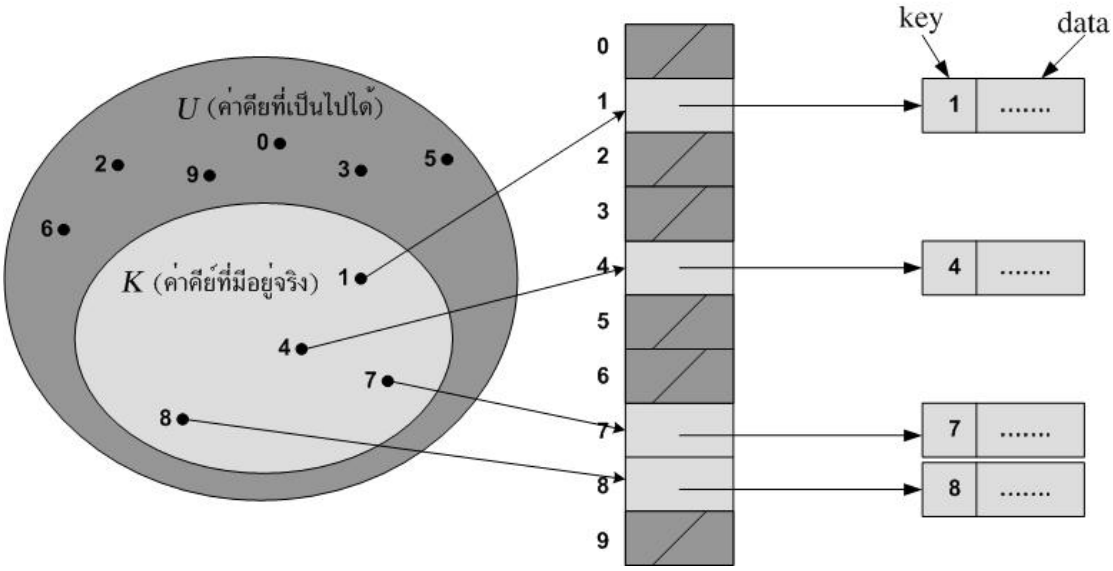
ข้อดีของโครงสร้างแบบผลึกหิมะ คือ การดูแลบำรุงรักษาง่ายและประหยัดพื้นที่ในการจัดเก็บข้อมูล ทำให้เราเห็นความสัมพันธ์ของแอทริบิวต์ของแต่ละมิติได้ชัดเจนขึ้น [12] แต่มีข้อเสีย คือ การประมวลผลข้อมูลทำได้ช้ากว่าโครงสร้างแบบดาว เนื่องจากมิติ (Dimension) ประกอบด้วยข้อมูลที่มาจากหลายตารางทำให้ต้องเสียเวลาในการเชื่อม (Join) ข้อมูล ทำให้โครงสร้างแบบผลึกหิมะได้รับความนิยมน้อยกว่าโครงสร้างแบบดาว

2.4 การเข้าถึงข้อมูล

การเข้าถึงข้อมูลถือเป็นสิ่งสำคัญในการประมวลผลข้อมูลด้วยคอมพิวเตอร์ ซึ่งความรวดเร็วขึ้นอยู่กับอัลกอริทึมที่ใช้ในการเข้าถึงข้อมูล ในบทนี้จะกล่าวถึง 2 วิธี คือ แฮชชิ่ง (Hashing) และการทำดัชนี (Indexing)

2.4.1 แฮชชิ่ง (Hashing)

การเข้าถึงข้อมูลโดยตรงสามารถทำได้ง่าย เมื่อค่าคีย์ที่เป็นไปได้ของข้อมูลมีจำนวนไม่มาก จนกระทั่งไม่มีคีย์ของข้อมูลใดซ้ำกันเลย ตัวอย่างเช่น ค่าคีย์ที่เป็นไปได้คือ $U = \{0,1,...,m-1\}$ โดยที่ m เป็นจำนวนเต็มที่มีค่าน้อย ๆ เราสามารถใช้อาเรย์ $T = [0,1,...,m-1]$ ในการเก็บข้อมูลได้ หรือกล่าวว่าตาราง T มี m สล๊อต ซึ่งสล๊อตที่ k ชี้ไปยังข้อมูลที่มีคีย์เท่ากับ k ถ้าไม่มีข้อมูลที่มีค่าคีย์เท่ากับ k แล้วสล๊อตนั้นก็จะว่าง หรือ $T[k] = \text{NULL}$ ดังนั้นการค้นหาข้อมูลจึงมีความรวดเร็ว ใช้เวลา $O(1)$ เท่านั้น ตัวอย่างเช่น ค่าคีย์ที่เป็นไปได้ คือ $U = \{0,1,...,9\}$ และค่าคีย์ที่มีอยู่จริงที่เก็บไว้บนสล๊อตที่มีพอยน์เตอร์ชี้ไปยังข้อมูลที่มีค่าคีย์นั้น คือ $K = \{1,4,7,8\}$ แสดงภาพประกอบ 2-5

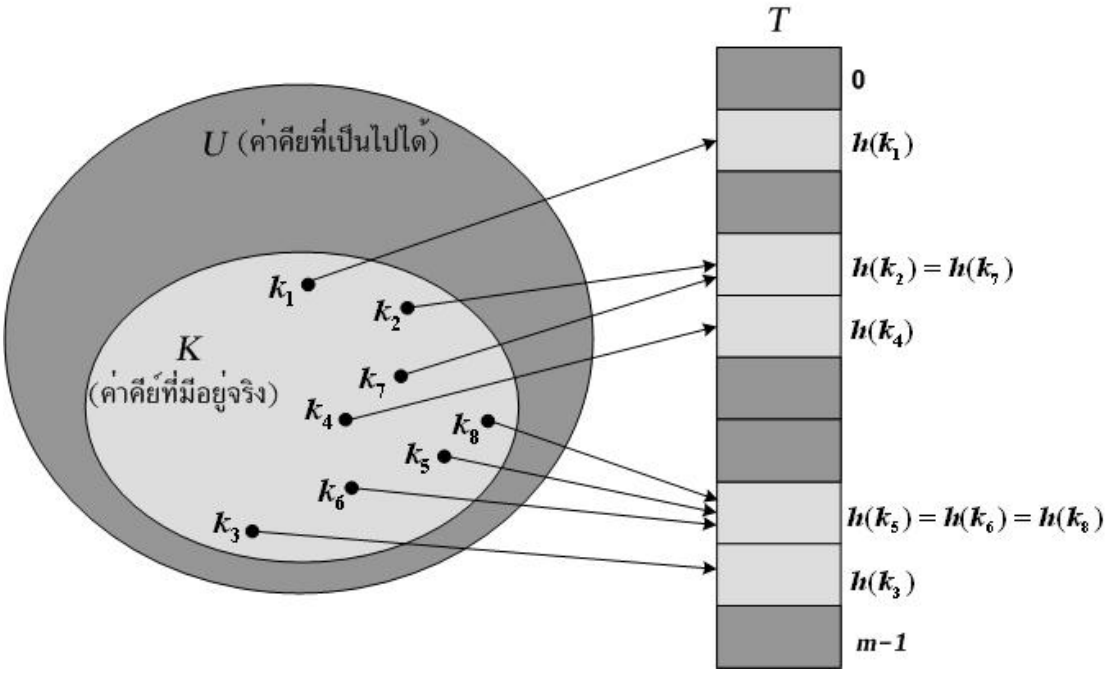


ภาพประกอบ 2-5 แสดงการเข้าถึงข้อมูลโดยตรง

การเข้าถึงข้อมูลโดยตรงทำได้ยากมากขึ้น เมื่อขนาดของค่าคีย์ที่เป็นไปได้ $(|U|)$ ใหญ่ขึ้น ก็จะทำให้ตารางที่เก็บค่าคีย์ต้องมีขนาดใหญ่ขึ้นด้วย จึงจัดการได้ยากและเป็นไปไม่ได้ที่จะมีหน่วยความจำเพียงพอ อย่างไรก็ตามถ้าขนาดของค่าคีย์ที่มีอยู่จริง $(|K|)$ มีขนาดเล็กกว่าขนาดของค่าคีย์ที่เป็นไปได้อยู่มาก ซึ่งหากจองหน่วยความจำไว้เท่ากับ $|U|$ เพื่อสร้างตารางการเข้าถึงข้อมูลโดยตรงก็จะเป็นการสิ้นเปลืองอย่างมาก

แฮชชิ่ง เป็นวิธีการหนึ่งที่สามารถช่วยแก้ปัญหาดังกล่าวนี้ได้ ในขณะที่ยังคงสามารถค้นหาข้อมูลได้อย่างรวดเร็วเช่นเดิม (ใช้เวลา $O(1)$) แฮชชิ่งประกอบด้วย *ตารางแฮช* เป็นอาเรย์ที่ใช้เก็บข้อมูล และ *ฟังก์ชันแฮช* เป็นฟังก์ชันที่ใช้ในการเทียบค่าคีย์กับช่วงค่าที่มีขนาดใหญ่ เพื่อคำนวณหาตำแหน่งที่อยู่ในอาเรย์

แนวคิดของแฮชชิ่ง คือ เราจะไม่เก็บค่าคีย์ที่เท่ากับ k ไว้ในสล๊อตที่ k แต่จะต้องนำค่าคีย์ไปคำนวณก่อน โดยใช้ฟังก์ชันแฮช ซึ่งเขียนแทนด้วย h ดังนั้น ค่าคีย์ที่เท่ากับ k จะถูกเทียบค่าไปยังสล็อตที่ $h(k)$ บนตารางแฮช ดังภาพประกอบ 2-6



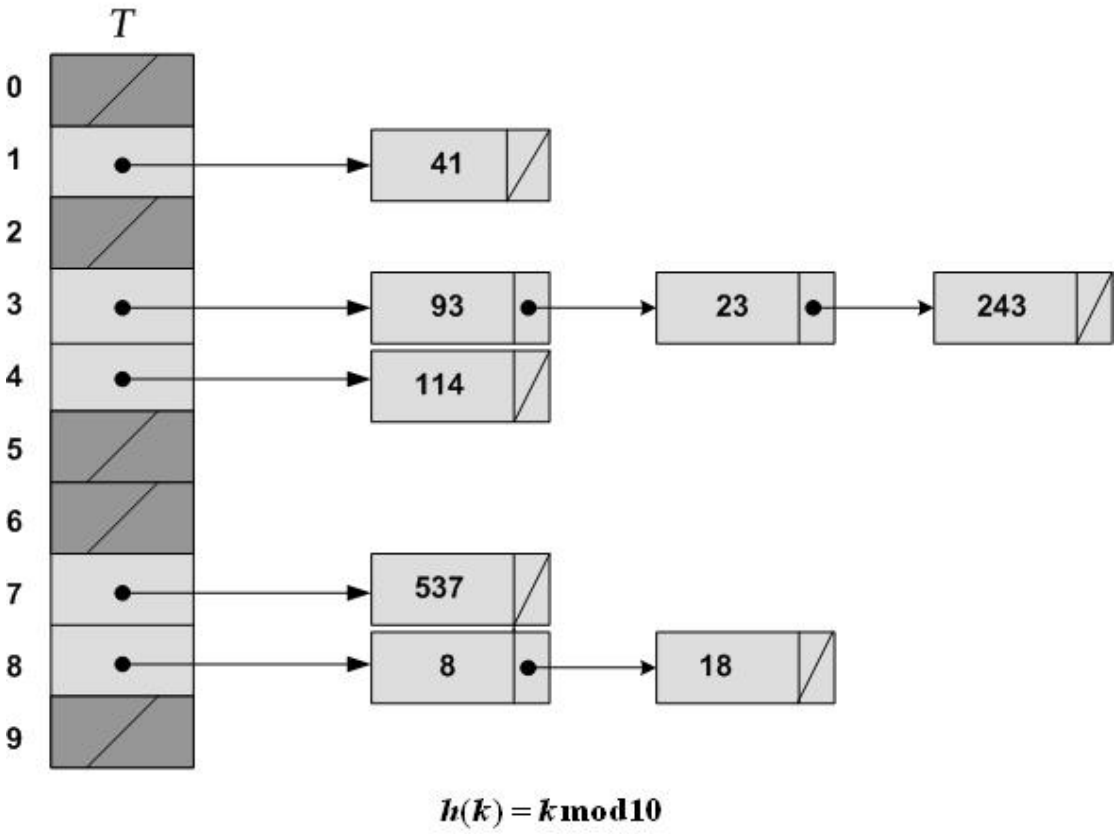
ภาพประกอบ 2-6 แสดงการใช้ฟังก์ชันแฮชในการเทียบค่าคีย์ไปยังสล๊อตบนตารางแฮช

อย่างไรก็ตาม แฮชชิ่งก็ยังมีปัญหาเกิดขึ้น คือ การชนกันของคีย์ (Collision) ดังภาพประกอบ 2-6 เกิดการชนกันของ $h(k_2) = h(k_7)$ และ $h(k_5) = h(k_6) = h(k_8)$ ซึ่งฟังก์ชันแฮชที่ดีจะต้องหลีกเลี่ยงการเกิดการชนกันของคีย์ ดังนั้นจะต้องมีการออกแบบฟังก์ชันแฮชให้มีการกระจายตัวของค่าคีย์ให้มากที่สุด แต่ก็ยากที่สร้างฟังก์ชันแฮชที่จะไม่ให้มีการเกิดการชนกันของคีย์เลย จึงต้องมีวิธีในการแก้ปัญหาเมื่อมีการชนกันของคีย์เกิดขึ้น ซึ่งจะกล่าวถึง 2 วิธีด้วยกัน คือ เชนนิ่ง (Chaining) และลิเนียร์โพรบบิง (Linear Probing)[4,5]

2.4.1.1 วิธีแก้ปัญหาการชนกัน (Collision Resolution)

- **วิธีการแก้ปัญหาการชนกันด้วยเชนนิ่ง (Collision Resolution by Chaining)**

ทุกค่าข้อมูลที่มีค่าคีย์เดียวกันจะถูกเก็บอยู่สล๊อตเดียวกัน โดยเชื่อมต่อกันเป็นแบบลิงค์ลิสต์ ตัวอย่างเช่น ตารางแฮช T มี 10 สล๊อต คือสล๊อตที่ 0 ถึง สล๊อตที่ 9 ฟังก์ชันแฮช คือ $h(k) = k \bmod 10$ ค่าคีย์ของข้อมูลที่ต้องการจัดเก็บคือ 114, 537, 93, 8, 23, 18, 243 และ 41 เมื่อดำเนินการแฮชชิ่งจะเกิดการชนกันขึ้น พิจารณาการแก้ปัญหาการชนกันด้วยเชนนิ่งดังภาพประกอบ 2-7



ภาพประกอบ 2-7 แสดงวิธีการแก้ปัญหาการชนกันด้วยเชนนิ่ง

จากภาพประกอบ 2-7 จะเห็นว่า ค่าคีย์ 93, 23 และ 243 เมื่อผ่านฟังก์ชันแฮชแล้วมีค่าคีย์เดียวกันคือ 3 ค่าคีย์ 23 จึงเชื่อมต่อจาก 93 และ 243 เชื่อมต่อจาก 23 ในทำนองเดียวกันค่าคีย์ 8 และ 18 เมื่อผ่านฟังก์ชันแฮชแล้วมีค่าคีย์เดียวกันคือ 8 นั่นคือมีค่าการชนกันของคีย์เกิดขึ้น ค่าคีย์ 18 จึงเชื่อมต่อจาก 8 ซึ่งวิธีเชนนิ่งสามารถช่วยแก้ปัญหาการชนกันได้ไม่จำกัดจำนวน สำหรับการค้นหาข้อมูลสามารถทำได้โดยนำค่าคีย์ของข้อมูลที่ต้องการค้นหามาผ่านฟังก์ชันแฮชเป็นค่าคีย์ใหม่ แล้วค้นหาข้อมูลที่สล๊อตเดียวกับค่าคีย์ที่ได้ ถ้ายังไม่พบก็ให้เลื่อนพอยน์เตอร์ไปอีกหนึ่งตำแหน่ง แล้วตรวจสอบว่าใช้ข้อมูลที่ต้องการหรือไม่ ถ้าไม่ใช่ก็ให้เลื่อน

พอยน์เตอร์ไปอีกเช่นเดิมจนกว่าจะพบข้อมูลที่ต้องการ หรือสิ้นสุดลิสต์นั้นแล้วก็หมายความว่าไม่มีข้อมูลที่ต้องการอยู่

- **วิธีการแก้ปัญหาการชนกันด้วยลิเนียร์โพรบบิง (Collision Resolution by Linear Probbing)**

วิธีนี้แก้ปัญหาการชนกันโดยการนำค่าคีย์ที่ทำให้เกิดการชนกันมาผ่านฟังก์ชันแฮชอีกครั้งหนึ่ง ดังนี้

$$h(k,i) = (h'(k) + i) \bmod m$$

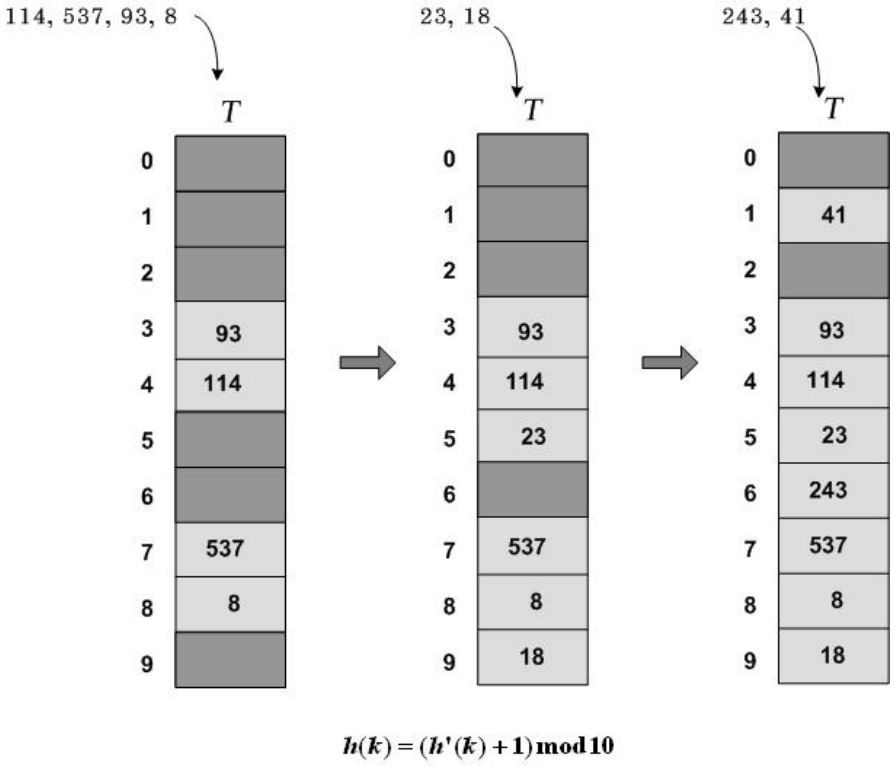
เมื่อ m คือ จำนวนสล๊อตของตารางแฮช

k คือ ค่าคีย์ของข้อมูลที่ต้องการค้นหา และ

$h'(k)$ คือ ค่าคีย์ที่ได้จากฟังก์ชันแฮชเดิมซึ่งทำให้เกิดการชน

$i = 0,1,...,m-1$

ในกรณีที่เรากำหนดให้ $i = 1$ จะหมายความว่า หากมีการชนกันของคีย์เกิดขึ้นให้หาสล๊อตที่ว่างถัดไปเพื่อเก็บค่าคีย์ที่ทำให้เกิดการชนนั้น ถ้ายังเกิดการชนอีกก็ให้หาที่ว่างถัดไปอีก ทำเช่นนี้ไปเรื่อยๆ พิจารณาการแก้ปัญหาการชนกันด้วยลิเนียร์โพรบบิ่งดังภาพประกอบ 2-8



ภาพประกอบ 2-8 แสดงวิธีการแก้ปัญหาการชนกันด้วยลิเนียร์โพรบบิ่ง

จากภาพประกอบ 2-8 จะเห็นว่าเมื่อเก็บข้อมูลที่มีค่าคีย์ 23 และ 18 จะเกิดการชนกันขึ้น จึงต้องหาที่ว่างสล๊อตถัดไปได้สล๊อตที่ 5 สำหรับเก็บค่าคีย์ 23 และสล๊อตที่ 9 สำหรับเก็บค่าคีย์ 18 ทำนองเดียวกันค่าคีย์ 243 ก็ทำให้เกิดการชนกันด้วยเช่นกัน จึงต้องนำไปเก็บไว้สล๊อตที่ 6 ซึ่งยังว่างอยู่ สำหรับการค้นหาข้อมูลทำได้โดยนำค่าคีย์ของข้อมูลมาผ่านฟังก์ชันแฮชแล้วไปค้นหาสล๊อตที่ตรงกับคีย์ที่ได้ ถ้ายังไม่ใช่ข้อมูลที่ต้องการก็เลื่อนไปยังสล๊อตถัดไปจนกว่าจะพบข้อมูล

2.4.1.2 ข้อดีของแฮชชิ่ง

แฮชชิ่งเหมาะสำหรับการค้นหาข้อมูลที่เป็นเซตซึ่งมีค่าคีย์ไม่ซ้ำกัน และการค้นหาข้อมูลที่มีคำตอบเพียงเรคอร์ดเดียว เช่น การค้นหาตำแหน่งของหน่วยความจำบนดิสก์ ซึ่งสามารถค้นหาข้อมูลดังกล่าวได้อย่างรวดเร็ว (ใช้เวลา $O(1)$)

2.4.1.3 ข้อจำกัดของแฮชชิ่ง

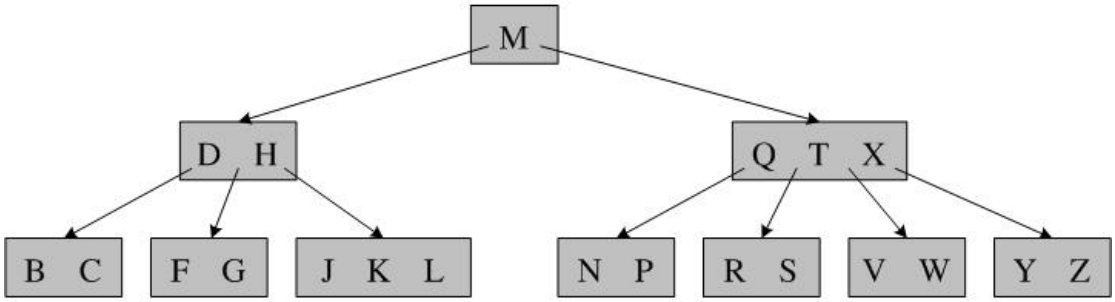
แฮชชิ่งไม่เหมาะสำหรับการค้นหาข้อมูลที่มีค่าคีย์ซ้ำกัน หรือมีคำตอบได้หลายเรคอร์ด นอกจากนั้นการหาฟังก์ชันแฮชที่ดีที่สุดที่ทำให้ไม่เกิดการชนกันของคีย์นั้นสามารถทำได้ยาก

2.4.2 ดัชนี (Indexing)

ดัชนีที่ใช้ในการค้นหาข้อมูลมีประโยชน์เหมือนกับดัชนีที่อยู่ในหนังสือ ที่ช่วยทำให้เราสามารถค้นหาข้อมูลได้เร็วขึ้น ถ้าไม่มีดัชนีแล้วจะต้องใช้วิธีการตรวจสอบข้อมูลที่มีอยู่ทั้งหมด ซึ่งจะต้องใช้เวลามาก เนื่องจากการใช้ดัชนีจะมีการเดินทางที่สั้นกว่าในการที่จะไปถึงแถวของข้อมูล ซึ่งเป็นการลด I/O ที่จะเกิดขึ้นได้[24] ในระบบคลังข้อมูลมีปริมาณข้อมูลมากและมีสภาพแวดล้อมที่มีการอ่านสูง จึงมีการนำดัชนีมาใช้ ซึ่งก็มีอยู่หลายแบบด้วยกัน เช่น ดัชนีแบบ B-Tree และดัชนีบิตแมป เป็นต้น ในบทที่นี้จะกล่าวถึงดัชนีแบบ B-Tree ส่วนดัชนีบิตแมปจะกล่าวในบทต่อไป

2.4.2.1 ดัชนีแบบ B-Tree

สำหรับโครงสร้างทั่วไปของดัชนีแบบ B-Tree คือ โหนดภายในที่มี t คีย์ สามารถมีโหนดลูกได้ทั้งหมด $t+1$ โหนด ทุกคีย์ที่อยู่บนโหนดจะเรียงจากน้อยไปหามากและมีพอยน์เตอร์ระหว่างคีย์เป็นตัวจัดการแบ่งช่วงค่าของคีย์ได้ $t+1$ ช่วงค่า โหนดลูกที่เชื่อมมาจากพอยน์เตอร์ทางซ้ายของคีย์จะต้องมีค่าน้อยกว่าหรือเท่ากับคีย์นั้น ส่วนโหนดลูกที่เชื่อมมาจากพอยน์เตอร์ทางขวาของคีย์จะต้องมีค่ามากกว่าคีย์นั้น[4,5] พิจารณาตัวอย่างดัชนีแบบ B-Tree ดังภาพประกอบ 2-9



ภาพประกอบ 2-9 แสดงตัวอย่างดัชนีแบบ B-Tree

ความสูงของดัชนีแบบ B-Tree ขึ้นอยู่กับจำนวนคีย์ที่อนุญาตให้มีได้ในแต่ละโหนด ถ้าจำนวนคีย์ทั้งหมดคือ n และแต่ละโหนดมีคีย์ได้ t คีย์ ความสูงของดัชนีแบบ B-Tree ก็จะเท่ากับ $\log_t n$ เมื่อมีการการค้นหาข้อมูลมาถึงโหนดที่มี t คีย์ ก็จะมีทางเลือกในการค้นหา $t+1$ เส้นทาง ดังนั้นการคนหาข้อมูลบนดัชนีแบบ B-Tree จึงใช้เวลาเป็น $O(\log_t n)$

2.4.2.1.1 ข้อดีของดัชนีแบบ B-Tree

ดัชนีแบบ B-Tree เป็นโครงสร้างข้อมูลที่ถูกออกแบบมาสำหรับทำงานเข้าถึงโดยตรงบนหน่วยความจำสำรอง มีการจัดการเกี่ยวกับ I/O ได้ดี ซึ่งระบบฐานข้อมูลส่วนใหญ่ก็มีการนำดัชนีแบบ B-Tree รวมถึงโครงสร้างอื่นที่อยู่บนพื้นฐานของดัชนีแบบ B-Tree เช่น B+ Tree และ B* Tree ไปใช้กับแอทริบิวต์ของข้อมูลที่มีลักษณะเป็นช่วงค่าหรือมีคาร์ดินอลิตี้สูง ๆ

2.4.2.1.2 ข้อจำกัดของดัชนีแบบ B-Tree

ดัชนีแบบ B-Tree ไม่เหมาะสำหรับการสอบถามที่ซับซ้อนและแอทริบิวต์มีคาร์ดินอลิตี้ต่ำ เนื่องจากจะต้องมีการสร้างคีย์ผสมขึ้นมาหากเป็นการสอบถามที่ต้องการข้อมูลมากกว่า 1 แอทริบิวต์ และเกิดความยุ่งยากและเสียค่าใช้จ่ายมากขึ้น

การสอบถามที่ซับซ้อนบนแอทริบิวต์ที่มีคาร์ดินอลิตี้ต่ำเกิดขึ้นบนคลังข้อมูลอยู่บ่อยครั้ง ซึ่งดัชนีที่เหมาะสำหรับการสอบถามในลักษณะนี้ คือ ดัชนีบิตแมป ซึ่งจะอธิบายในบทถัดไป