บทที่ 2
หลักการและทฤษฎี

2.1 หลักการและทฤษฎีในส่วนของฐานข้อมูล

2.1.1 ระบบฐานข้อมูล

ปัจจุบันการนำคอมพิวเตอร์มาใช้ในการจัดการเกี่ยวกับฐานข้อมูล (Database) ได้รับความนิยมมากโดยเฉพาะอย่างยิ่งในองค์กรที่มีขนาดใหญ่ ๆ ทั้งนี้เนื่องจากการจัดการสามารถทำได้รวดเร็ว และถูกต้องแม่นยำทำให้ประสิทธิภาพโดยรวมในการดำเนินการขององค์กรสูงขึ้นด้วยระบบฐานข้อมูล (Database System) คือการจัดเก็บข้อมูลอย่างเป็นระบบ ซึ่งผู้ใช้สามารถเรียกใช้ข้อมูลดังกล่าวได้ในลักษณะต่าง ๆ เช่น การเพิ่มข้อมูล (Add Data) การแทรกข้อมูล (Insert Data) ไปตามกำหนด

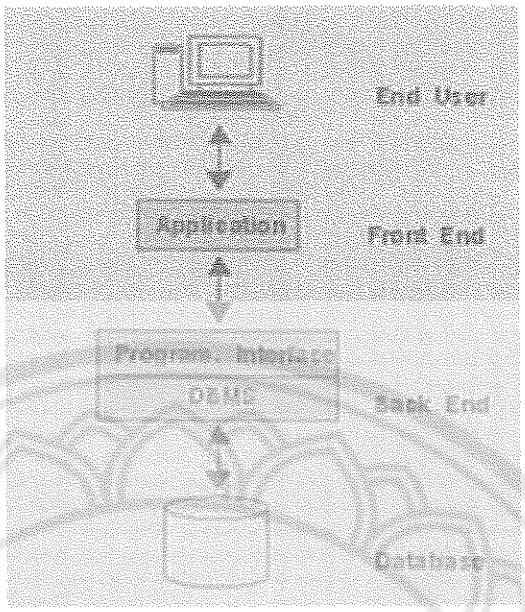
ฐานข้อมูล (Database) คือ กลุ่มของข้อมูลที่มีความสัมพันธ์เกี่ยวข้องเป็นเรื่องเดียวกัน เช่น กลุ่มข้อมูลเกี่ยวกับพนักงานบริษัท ประกอบด้วย รหัสพนักงาน ชื่อ นามสกุล เบอร์โทรศัพท์ และกลุ่มข้อมูลดังกล่าวถูกจัดเก็บอยู่รวมกันหลาย ๆ กลุ่ม ซึ่งอาจจะเก็บอยู่ในรูปแฟ้มเอกสารหรืออยู่ในคอมพิวเตอร์

ระบบฐานข้อมูล (Database System) คือ การจัดรวบรวมแฟ้มข้อมูลต่าง ๆ ไว้เป็นส่วนกลาง (Centralized Database System) โดยแฟ้มข้อมูลเหล่านี้ถูกจัดให้มีความสัมพันธ์กันและสามารถที่จะเรียกข้อมูลนั้น ๆ มาใช้ร่วมกันได้การแก้ไขและลบข้อมูล การเพิ่มข้อมูลตลอดจนการเคลื่อนย้าย ช่วยทำให้การประมวลผลมีประสิทธิภาพมากขึ้น ลดความซ้ำซ้อนของข้อมูล และยังทำให้ประหยัดเนื้อที่หน่วยความจำ

หรือ ระบบฐานข้อมูล อาจหมายถึง การจัดทำฐานข้อมูลเพื่อสนับสนุนการดำเนินงานหรือกิจกรรมอย่างใดอย่างหนึ่ง

2.1.2 โครงสร้างของระบบ (Structure of Database)

ระบบฐานข้อมูลในมุมมองของผู้ใช้สามารถแบ่งออกตามลักษณะโครงสร้าง ซึ่งประกอบไปด้วยโครงสร้างหลัก 2 ส่วน ได้แก่ส่วน Front End และ Back End ดังรูปที่ 2.1



รูปที่ 2.1 โครงสร้างของระบบ

(ที่มา: ณัฐพล อุ่นยัง, 2544. หน้า 18)

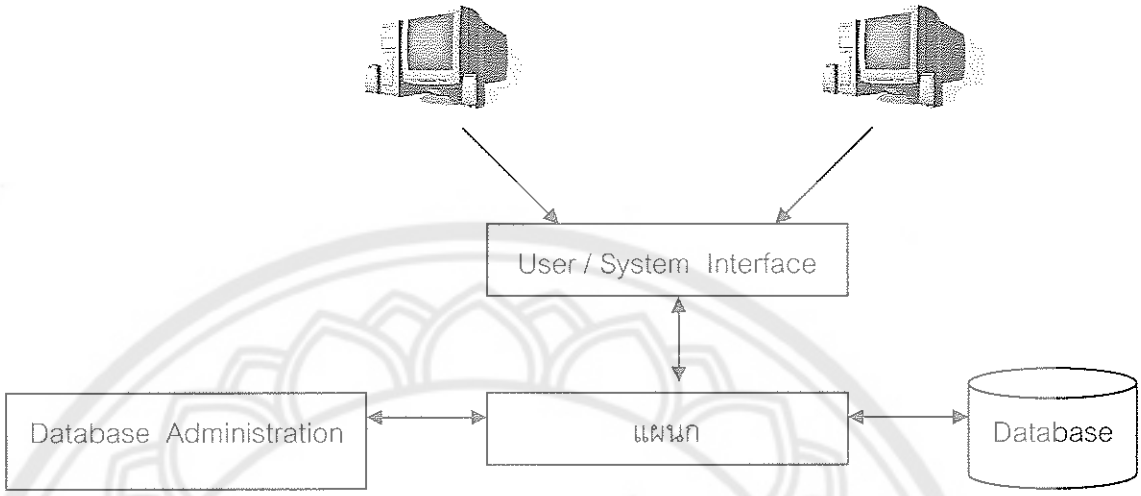
2.1.2.1 Front End

เป็นโปรแกรมประยุกต์ (Application) ที่อาจจะสร้างจากภาษาต่าง เช่น ภาษาระดับสูง CASE หรือ ภาษาอื่น ส่วนนี้โดยปกติจะรองรับการทำงานของผู้ใช้ (End User) เพื่อทำหน้าที่ติดต่อกับระบบ

2.1.2.2 Back End

เป็นส่วนที่ทำหน้าที่ในการจัดการกับระบบฐานข้อมูลทั้งหมด ในแง่ของการจัดเก็บและเรียกใช้ข้อมูลจากแหล่งข้อมูลจริง ได้แก่ การปฏิบัติการต่าง ๆ กับข้อมูล , การจัดทำ Backup , การควบคุมความถูกต้องในการใช้ข้อมูลพร้อมกัน รวมไปถึงการควบคุมความปลอดภัยของระบบ เป็นต้น

2.1.3 องค์ประกอบของระบบฐานข้อมูล



รูปที่ 2.2 การติดต่อกับระบบฐานข้อมูลของบุคลากร
(ที่มา: ณัฐพล อุ่นยัง, 2544. หน้า 19)

ระบบฐานข้อมูลโดยส่วนใหญ่แล้ว เป็นระบบที่มีการนำเอาคอมพิวเตอร์ มาช่วยในกระบวนการจัดเก็บข้อมูล ค้นหาข้อมูล ประมวลผลข้อมูล เพื่อให้ได้สารสนเทศที่ต้องการแล้วนำไปใช้ในการปฏิบัติงานและบริหารงานของผู้บริหาร โดยอาศัยโปรแกรมเข้ามาช่วยจัดการข้อมูล ดังรูปที่ 2.2 จากกระบวนการดังกล่าวนี้ระบบฐานข้อมูลจึงมีองค์ประกอบ 5 ประเภท คือ

2.1.3.1 ฮาร์ดแวร์ (Hardware) ในส่วนของฮาร์ดแวร์ ที่เกี่ยวข้องกับระบบจะพิจารณาถึงส่วนประกอบที่สำคัญ 2 ประการ ส่วนแรกคือ สื่อในการเก็บข้อมูล (secondary storage) ได้แก่ การเก็บข้อมูลด้วย magnetic disk รวมไปถึงการติดต่อระหว่างอุปกรณ์ที่เกี่ยวข้อง เช่น I/O device ต่าง ๆ ส่วนที่สอง จะเกี่ยวข้องกับความเร็วในการทำงานของ Processor และ Memory ซึ่งจะขึ้นอยู่กับขนาดของข้อมูลในระบบและจำนวนของข้อมูลในระบบและจำนวนของผู้ใช้เป็นตัวกำหนด

2.1.3.2 โปรแกรม (Program หรือ Software) ซึ่งมีหน้าที่ควบคุมดูแลการสร้างฐานข้อมูล การเรียกใช้ข้อมูล และการจัดทำรายงาน เรียกว่า โปรแกรมระบบจัดการฐานข้อมูล (Database Management System : DBMS)

2.1.3.3 ข้อมูล (Data) เนื่องจากฐานข้อมูลเป็นการจัดเก็บรวบรวมข้อมูลให้มีลักษณะเป็นศูนย์กลางข้อมูลอย่างเป็นระบบ ในกรณีที่มีผู้ใช้ร่วมกันหลายคน (Multi - User) ข้อมูลจะต้อง

สามารถเรียกใช้ร่วมกันได้ ซึ่งในทางปฏิบัติผู้ใช้จะมองภาพของข้อมูลที่แตกต่างกันไป ตามระดับของการออกแบบระบบ

2.1.3.4 บุคลากร (People - ware) คือ ผู้ใช้งาน (User) พนักงานปฏิบัติการ (Operator) นักวิเคราะห์และออกแบบระบบ (System Analyst) ผู้เขียนโปรแกรมประยุกต์ใช้งาน (Programmer) และผู้บริหารฐานข้อมูล (Database Administrator : DBA)

2.1.4 แนวคิดการออกแบบฐานข้อมูล (Database Approach)

ระบบฐานข้อมูลจะมีแนวคิดในการจัดการกับตัวข้อมูลโดยตรงนั่นคือ ความพร้อมของข้อมูลที่จะถูกเรียกใช้ได้ทันทีที่ต้องการ นอกจากนี้แล้วข้อมูลในระบบจะถูกใช้ร่วมกัน (Shared Data) โดยผู้ใช้แต่ละคนจะมองเห็นระบบฐานข้อมูลที่แตกต่างกันตามลักษณะการทำงานที่ได้ถูกกำหนดไว้โดยผู้ออกแบบระบบ

2.1.5 ผลกระทบของการประมวลผลด้วยระบบฐานข้อมูล

2.1.5.1 ข้อดีของการประมวลผลด้วยระบบฐานข้อมูล

1. ลดความซ้ำซ้อนของข้อมูล (Minimal Data Redundancy)

การจัดเก็บข้อมูลในลักษณะเป็นแฟ้มข้อมูล อาจทำให้ข้อมูลประเภทเดียวกันถูกเก็บไว้หลาย ๆ แห่งทำให้เกิดความซ้ำซ้อนของข้อมูลขึ้นได้ ดังนั้นการนำข้อมูลรวมมาเก็บไว้ในระบบฐานข้อมูลจะช่วยลดปัญหาความซ้ำซ้อนของข้อมูลได้

2. หลีกเลี่ยงความขัดแย้งของข้อมูลได้ (Consistency of Data)

การจัดเก็บข้อมูลในลักษณะเป็นแฟ้มข้อมูล โดยที่ข้อมูลเป็นเรื่องเดียวกัน อาจมีอยู่หลายแฟ้มซึ่งก่อให้เกิดความขัดแย้งของข้อมูลขึ้นได้ ทั้งนี้อาจเนื่องมาจากการแก้ไขข้อมูลที่แฟ้มแห่งหนึ่งแต่มิได้แก้ไขข้อมูลเรื่องเดียวกันที่อยู่ในไฟล์อื่น ๆ ทำให้ข้อมูลนั้น ๆ แตกต่างกันได้

3. จำกัดความผิดพลาดของข้อมูลให้น้อยที่สุด (Data integrity)

บางครั้งความผิดพลาดของข้อมูล อาจเกิดขึ้นจากการป้อนข้อมูลที่ไม่ถูกต้องเข้าสู่ระบบ ดังนั้น ในระบบจัดการฐานข้อมูลจึงจำเป็นที่จะต้องกำหนดกฎเกณฑ์ในการรับข้อมูลจากการป้อนของผู้ใช้เพื่อรักษาความถูกต้องของข้อมูลให้มากที่สุดเท่าที่จะทำได้

4. สามารถใช้ข้อมูลร่วมกันได้ (Sharing of data)

เนื่องจากระบบฐานข้อมูลเป็นการจัดเก็บข้อมูลไว้ในที่เดียวกัน เมื่อผู้ใช้ต้องการเรียกใช้ข้อมูลจากแฟ้มที่แตกต่างกัน ก็จะสามารถทำได้โดยง่าย

5. สามารถกำหนดความเป็นมาตรฐานเดียวกันได้ (Enforcement of Standard)

การเก็บข้อมูลไว้ด้วยกันจะสามารถกำหนด และควบคุมความมีมาตรฐานของข้อมูลให้เป็นไปในทิศทางเดียวกันได้ ดังนั้นจึงทำให้ระบบเกิดความเชื่อมั่นมากยิ่งขึ้น

6. สามารถกำหนดระบบความปลอดภัยของข้อมูลได้ (Security and Privacy Control)

เนื่องจากระบบจะทำการกำหนดระดับของผู้ใช้แต่ละคนตามลำดับความสำคัญของผู้ใช้ ดังนั้น จึงสามารถที่จะควบคุม และดูแลความปลอดภัยของข้อมูลภายในระบบได้ดียิ่งขึ้น

7. ข้อมูลมีความเป็นอิสระ (Data Independence)

ระบบฐานข้อมูลจะทำหน้าที่เป็นตัวเชื่อมโยงกับโปรแกรมประยุกต์ ที่ทำงานกับข้อมูลโดยตรง การแก้ไขข้อมูล เช่น ต้องการเปลี่ยนรหัสไปรษณีย์จากเลข 4 หลัก เป็นเลข 5 หลัก ก็จะกระทำการแก้ไขข้อมูลที่เป็นข้อมูลที่เป็นรหัสไปรษณีย์เท่านั้น ส่วนโปรแกรมอื่นจะเป็นอิสระต่อการเปลี่ยนแปลงนี้

2.1.5.2 ข้อเสียของการประมวลผลด้วยระบบฐานข้อมูล

1. ขั้นตอนการออกแบบดำเนินการและการบำรุงรักษามีต้นทุนที่สูง เนื่องจากระบบต้องใช้ผู้เชี่ยวชาญเฉพาะในการออกแบบระบบ ไม่ว่าจะเป็นทางด้าน Hardware และ Software รวมไปถึงราคาอุปกรณ์ที่ใช้มีราคาค่อนข้างสูง

2. ระบบมีความซับซ้อนจำเป็นต้องมีผู้ดูแลระบบที่ถูกฝึกมาอย่างดี เพื่อรองรับสถานการณ์ที่ผิดพลาดอันอาจจะเกิดขึ้นได้

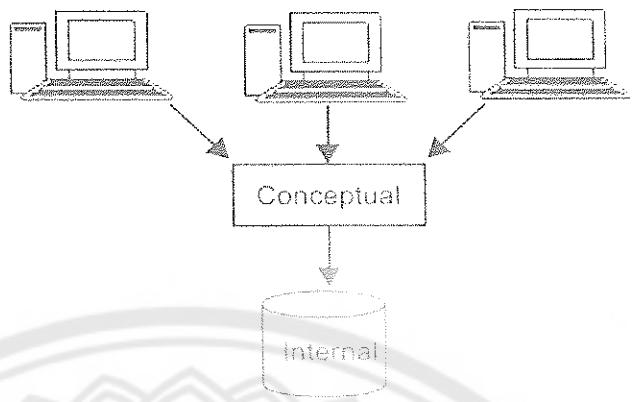
3. การเสี่ยงต่อการหยุดชะงักของระบบ เนื่องจากข้อมูลอาจถูกจัดเก็บแบบรวมศูนย์ (Centralized Database System) ความล้มเหลวของการทำงานบางส่วน อาจทำให้ระบบฐานข้อมูลโดยรวมหยุดชะงักการทำงานได้

2.1.6 สถาปัตยกรรมของระบบฐานข้อมูล

ระบบฐานข้อมูลถูกออกแบบมา เพื่อรองรับโครงสร้างข้อมูลที่มีผู้ใช้หลายคน ดังนั้นจึงต้องมีการแบ่งระดับของข้อมูลออกเป็นหลายระดับ ทั้งนี้เพื่อให้เหมาะสมกับความต้องการของผู้ใช้แต่ละคน เช่น ผู้บริหาร ผู้ทำหน้าที่ดูแลระบบ ผู้ใช้ทั่วไป เป็นต้น

2.1.6.1 การแบ่งระดับสถาปัตยกรรมของฐานข้อมูล

การแบ่งระดับดังกล่าวนี้บางครั้งอาจจะเรียกรวมได้ว่า สถาปัตยกรรมของระบบฐานข้อมูล ซึ่งจะอาศัยลักษณะในการมองภาพรวม (View) ของระบบเพื่อจำแนกความแตกต่างออกได้เป็น 3 ระดับ ดังรูปที่ 2.3



รูปที่ 2.3 การแบ่งระดับสถาปัตยกรรมของฐานข้อมูล
(ที่มา: ณัฐพล อุ่นยัง, 2544. หน้า 25)

2.1.6.2 Internal Level

เป็นระดับที่ใช้ในการเก็บข้อมูลจริง ได้แก่ ส่วนที่ทำหน้าที่ในการจัดเก็บข้อมูลของระบบ ซึ่งจะครอบคลุมไปถึง การกำหนดชนิดของข้อมูลที่เหมาะสมตามโครงสร้างที่กำหนด นอกจากนี้ยังรวมไปถึงการจัดการเกี่ยวกับวิธีการในการเข้าถึงข้อมูลแบบต่าง ๆ อีกด้วย

2.1.6.3 Conceptual Level

เป็นการมองภาพรวมที่เกี่ยวข้องกับข้อมูลทั้งหมดที่ปรากฏอยู่ฐานข้อมูลของระบบ ในเชิงการออกแบบระบบฐานข้อมูล

2.1.6.4 External Level

เป็นระดับของข้อมูลที่สนองตอบต่อการใช้แต่ละคน ซึ่งจะมีการมองภาพของข้อมูลที่แตกต่างกัน ดังนั้นมุมมองและวิธีการเข้าหาข้อมูลของผู้ใช้แต่ละคนก็แตกต่างกันไปด้วย โดยทั่วไปจะเป็นเพียงการใช้ข้อมูลกับฐานข้อมูลเป็นบางส่วนแล้วแต่ผู้ออกแบบระบบจะเป็นผู้กำหนด

2.1.7 **ขั้นตอนในการออกแบบฐานข้อมูล**

2.1.7.1 การกำหนดชนิดของเอ็นติตี้

ก่อนอื่นเราต้องวิเคราะห์ข้อมูลแล้วตัดสินใจว่าจะแบ่งข้อมูลออกมาเป็นกี่เอนติตี้ ค่าของเอนติตี้ แต่ละตัวจะถูกเก็บอยู่ในรูปของไฟล์ ในขณะเดียวกันค่าของ Attribute ก็จะได้แก่ ค่า field นั่นเอง ส่วนความสัมพันธ์ระหว่างเอนติตี้ ได้แก่ ความสัมพันธ์ระหว่างไฟล์ที่ถูกนำเสนอในรูปของการกำหนดค่าของ field ในไฟล์หนึ่ง เพื่อแสดงความสัมพันธ์ไปยังอีกไฟล์หนึ่งนั่นเอง

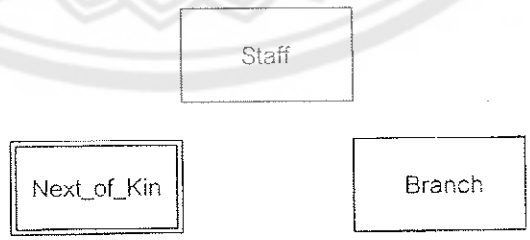
เอนทิตี้ (Entity) หมายถึง ชื่อของสิ่งหนึ่งสิ่งใด เปรียบเสมือนคำนาม ได้แก่ บุคคล สถานที่ สิ่งของ วัตถุ หรือเหตุการณ์ที่ทำให้เกิดกลุ่มของข้อมูลที่ต้องการจัดเก็บ รวมทั้งสามารถบ่งชี้ถึงความเป็นเอกลักษณ์เฉพาะตัวได้ (Uniquely identifiable) ดังรูปที่ 2.4 ตัวอย่างของแต่ละเอ็นติตี้ต่าง ๆ ประกอบด้วย

- บุคคล (Person) เช่น customer , department , division , employee , student , supplier
- สถานที่ (Place) เช่น building , room , branch , office , campus
- วัตถุ (Objects) เช่น book , machine , product , part , raw material
- เหตุการณ์ (Events) เช่น invoice , order , registration , reservation
- แนวความคิด (Concepts) เช่น account , bond , course , stock

รูปที่ 2.4 ตัวอย่างเอ็นติตี้พนักงาน (Staff) และเอ็นติตี้สาขา (Branch)
(ที่มา: โอภาส เอี่ยมสิริวงศ์, 2546. หน้า 91)

เอ็นติตี้ยังสามารถแบ่งออกเป็น 2 ประเภทด้วยกัน ดังรูปที่ 2.5 คือ

1. Strong Entity เป็นเอ็นติตี้ที่เกิดขึ้นได้ด้วยตัวเอง โดยไม่ขึ้นกับเอ็นติตี้ใด ๆ ใช้สัญลักษณ์รูปสี่เหลี่ยมผืนผ้า สามารถเรียก Strong Entity ได้อีกชื่อหนึ่งว่า Regular Entity
2. Weak Entity เป็นเอ็นติตี้อ่อนแอ กล่าวคือ ชีวิตของเอ็นติตี้ชนิดนี้จะขึ้นอยู่กับเอ็นติตี้ชนิดอื่น ๆ ใช้สัญลักษณ์รูปสี่เหลี่ยมผืนผ้าเช่นกันแต่เป็นเส้นคู่



รูปที่ 2.5 Strong Entity และ Weak Entity
(ที่มา: โอภาส เอี่ยมสิริวงศ์, 2546. หน้า 91)

2.1.7.2 **การกำหนดชนิดของความสัมพันธ์**

เราต้องกำหนดว่าจะเชื่อมต่อข้อมูลจากเอ็นติตี้หนึ่งไปอีกเอ็นติตี้หนึ่งยังไง จะเชื่อมต่อแบบ one to one , one to many , many to many เพื่อจัดกลุ่มข้อมูลสำหรับการจัดเก็บ และพิจารณาความสัมพันธ์ ในด้านการประมวลผล เพื่อแสดงผลที่ต้องการได้อย่างมีประสิทธิภาพ

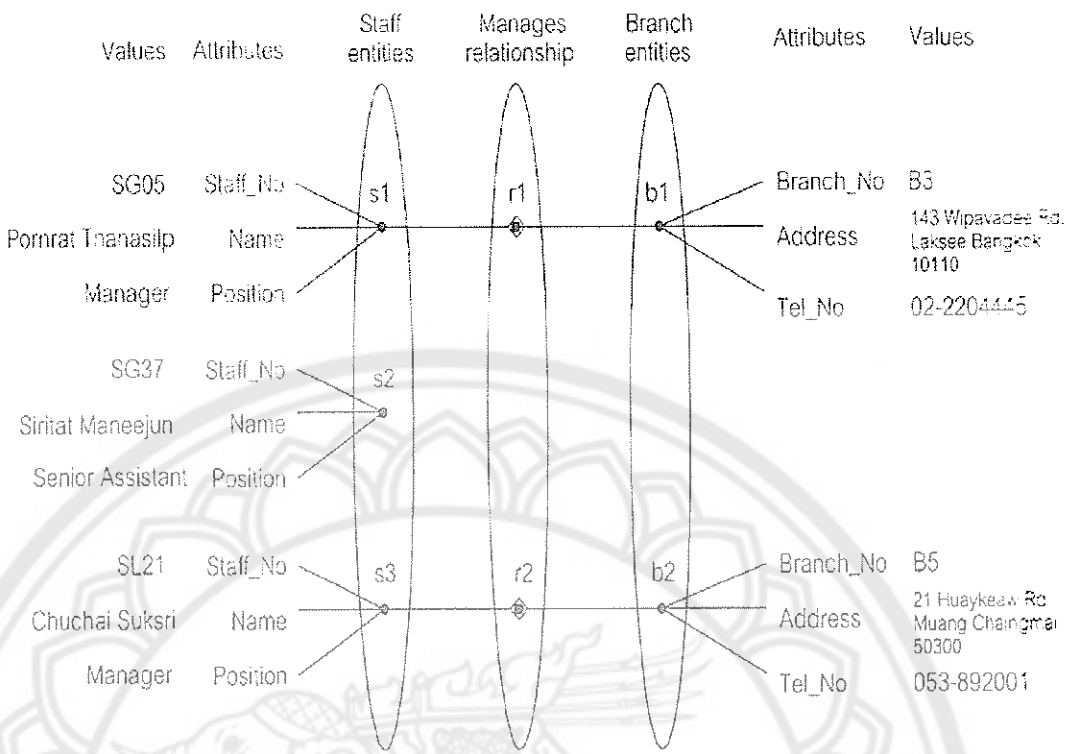
ความสัมพันธ์ (Relationship) หมายถึง คำกริยาที่แสดงความสัมพันธ์ระหว่าง Entity เช่น ความสัมพันธ์ระหว่างหลักสูตรวิชาและนักศึกษา ก็เป็นในลักษณะหลักสูตรวิชาที่นักศึกษานั้น ๆ เรียนอยู่ ส่วนความสัมพันธ์ระหว่างนักศึกษากับหลักสูตรวิชา ก็เป็นในลักษณะที่ว่า นักศึกษาเรียนในหลักสูตรวิชานั้น ๆ

ความสัมพันธ์ระหว่าง Entity สามารถเขียนแทนได้ด้วย สัญลักษณ์ หัวลูกศร แบ่งชนิดของความสัมพันธ์ออกได้เป็น 3 ลักษณะ ดังต่อไปนี้

1. **ความสัมพันธ์แบบ** 1 **ต่อ** 1 (One – To – One Relationship) หมายถึง ในช่วงระยะเวลาที่กำหนด ค่าของ Entity A มีความสัมพันธ์กับค่าของ Entity B เพียงค่าเดียวเท่านั้น นั่นคือ หากทราบว่าค่าของ Entity A ก็สามารถหาค่าของ Entity B ได้ด้วย ดังรูปที่ 2.6

รูปที่ 2.6 ความสัมพันธ์แบบหนึ่งต่อหนึ่ง
(ที่มา: โอภาส เอี่ยมสิริวงศ์, 2546. หน้า 97)

จากรูปที่ 2.7 เป็นความสัมพันธ์แบบหนึ่งต่อหนึ่ง กล่าวคือพนักงาน (Staff) หนึ่งคนจะดูแลหนึ่งสาขา นั่นหมายถึง พนักงานที่เป็นหัวหน้าจะดูแลสาขาหนึ่งสาขา ในขณะที่สาขาจะมีหัวหน้าพนักงานดูแลได้เพียงหนึ่งคนและเพื่อให้การนำเสนอรายละเอียดได้ชัดเจนยิ่งขึ้นจึงสามารถนำเสนอในลักษณะของแผนภาพแบบ semantic net model



รูปที่ 2.7 Semantic Net Model ของความสัมพันธ์ Staff<Manages>Branch (1:1)

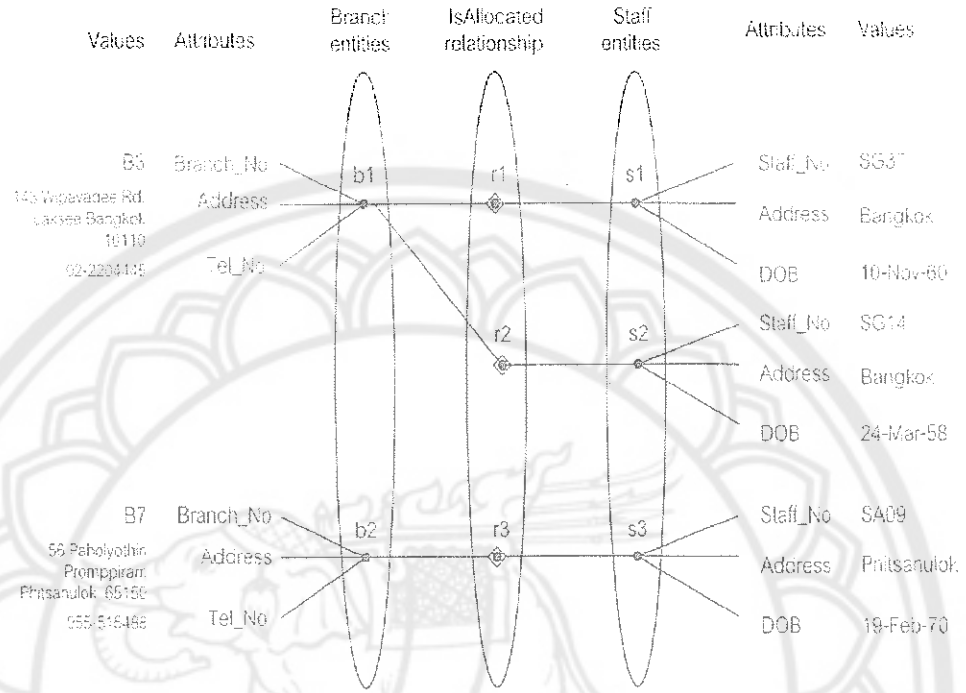
(ที่มา: โอภาส เอี่ยมสิริวงศ์, 2546. หน้า 97)

2. **ความสัมพันธ์แบบหนึ่งต่อกลุ่ม** (One to Many Relationships) หมายถึง ในช่วงระยะเวลาที่กำหนดค่าของ Entity A จะมีความสัมพันธ์กับค่าของ Entity B ได้มากกว่าหนึ่งเท่า ความสัมพันธ์ในลักษณะนี้จะเกิดขึ้นเป็นส่วนใหญ่ในระบบ ดังรูปที่ 2.8

รูปที่ 2.8 ความสัมพันธ์แบบหนึ่งต่อกลุ่ม

(ที่มา: โอภาส เอี่ยมสิริวงศ์, 2546. หน้า 97)

จากรูปที่ 2.9 เป็นความสัมพันธ์แบบหนึ่งต่อกลุ่ม กล่าวคือ สาขาหนึ่งจะมีพนักงานอยู่หลายคนโดยที่พนักงานหลาย ๆ คนจะสังกัดอยู่หนึ่งสาขา

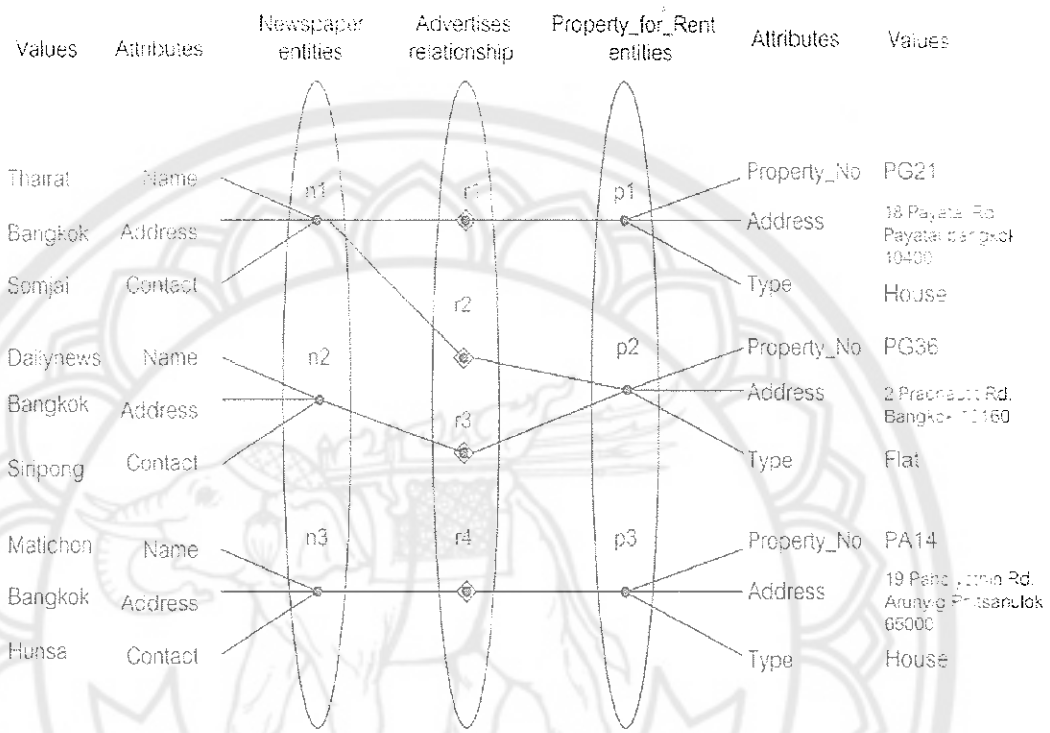


รูปที่ 2.9 Semantic Net Model ของความสัมพันธ์ Branch<IsAllocated>Staff (1:M)
(ที่มา: โอภาส เอี่ยมสิริวงศ์, 2546. หน้า 98)

3. **ความสัมพันธ์แบบกลุ่มต่อกลุ่ม** (Many to Many relationships) หมายถึง ในช่วงระยะเวลาที่กำหนด ทั้งค่าของ Entity A มีความสัมพันธ์กับค่าของ Entity B ได้มากกว่า 1 ค่า ดังรูปที่ 2.10

รูปที่ 2.10 ความสัมพันธ์แบบกลุ่มต่อกลุ่ม
(ที่มา: โอภาส เอี่ยมสิริวงศ์, 2546. หน้า 98)

จากรูปที่ 2.11 เป็นความสัมพันธ์แบบกลุ่มต่อกลุ่ม กล่าวคือ บ้านเช่าหลาย ๆ หลัง สามารถประกาศโฆษณาลงในหนังสือพิมพ์หลาย ๆ ฉบับได้ ในขณะที่หนังสือพิมพ์หลายฉบับก็สามารถลงโฆษณาบ้านเช่าได้หลายหลังเช่นกัน



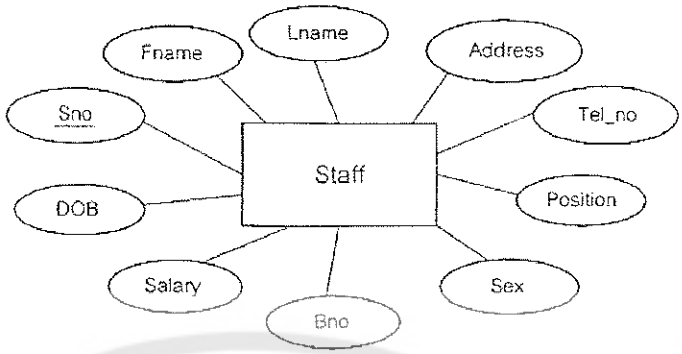
รูปที่ 2.11 Semantic Net Model ของความสัมพันธ์ (M:N)

(ที่มา: โอภาส เอี่ยมสิริวงศ์, 2546. หน้า 99)

2.1.7.3 **กำหนดแอตตริบิวท์ให้กับเอ็นติตี้**

ก็คือการกำหนดรายละเอียดในเอ็นติตี้ ว่าในเอ็นติตี้หนึ่ง ๆ ควรจะมีข้อมูลอะไรบ้าง

แอททริบิวท์ (Attribute) หมายถึง รายละเอียดของข้อมูลใน Entity หนึ่ง **ซึ่งเป็น**ข้อมูลที่แสดงลักษณะและคุณสมบัติของ Entity เช่น เอ็นติตี้ staff ประกอบด้วย แอตตริบิวท์ หมายเลขพนักงาน (Sno) ชื่อ (Fname) สกุล (Lname) ที่อยู่ (Address) โทรศัพท์ (Tel_No) ตำแหน่ง (Position) เพศ (Sex) วันเกิด (DOB) เงินเดือน (Salary) และรหัสสาขา (Bno) สัญลักษณ์แอตตริบิวท์ใน ER – Diagram จะใช้สัญลักษณ์รูปวงรี และแอตตริบิวท์ใดเป็นคีย์ก็จะมีการขีดเส้นใต้ชื่อแอตตริบิวท์นั้น ดังรูปที่ 2.12



รูปที่ 2.12 แอตตริบิวท์

(ที่มา: โอภาส เอี่ยมสิริวงศ์, 2546. หน้า 92)

แอตตริบิวท์ ยังแบ่งออกเป็นหลายประเภทด้วยกัน ดังรูปที่ 2.13 คือ

1. Attribute Domain คือ การกำหนดขอบเขตค่าข้อมูลและชนิดข้อมูลของแต่ละแอตตริบิวท์ นั่นหมายถึง โดเมนจะเป็นตัวกำหนดความเป็นไปได้ของข้อมูล

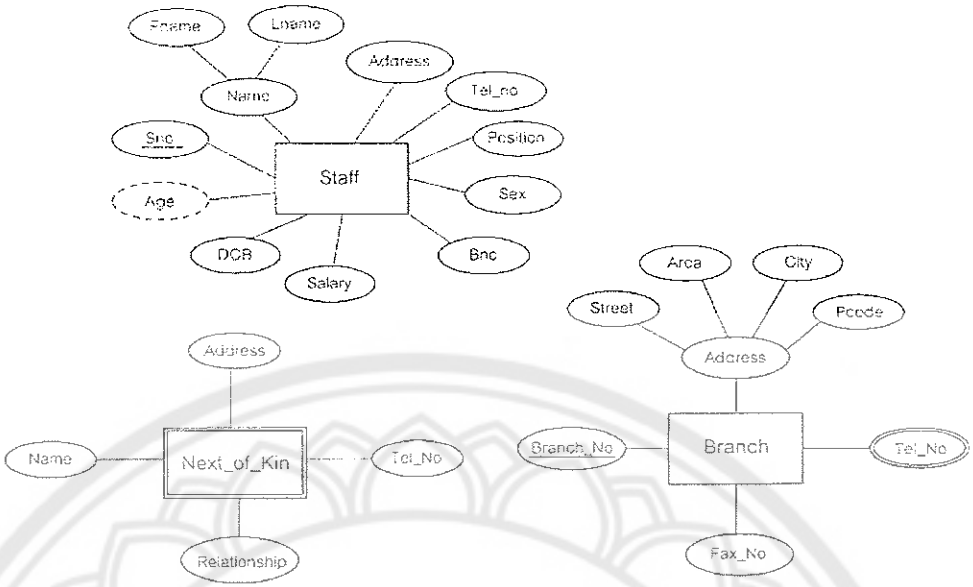
2. Simple Attribute คือ แอตตริบิวท์ที่มีองค์ประกอบเดียวที่เป็นอิสระ เช่น แอตตริบิวท์ Sex และ Salary ในบางครั้งอาจเรียก Simple Attribute อีกชื่อหนึ่งว่า Atomic Attributes

3. Composite Attribute คือ แอตตริบิวท์ที่มีองค์ประกอบอยู่หลาย ๆ ตัว โดยแต่ละตัวจะมีความเป็นอิสระต่อกัน เช่น แอตตริบิวท์ Address ประกอบด้วย Street , Area , City , และ Postcode เป็นต้น

4. Single – value Attribute คือ แอตตริบิวท์ที่บรรจุค่าเพียงค่าเดียว เช่น เอ็นติตี้ branch จะมีแอตตริบิวท์ Bno เป็น Single – value Attribute เช่น รหัสสาขา B3 นั่นหมายถึง Bno (Branch_no) จะนำไปอ้างอิงได้เพียงหนึ่งค่าเท่านั้น

5. Multi - value Attribute คือ แอตตริบิวท์ที่ประกอบด้วยค่าหลาย ๆ ค่าผสมกัน เช่น ในเอ็นติตี้ Branch จะมีแอตตริบิวท์ Tel_No ซึ่งเป็นเบอร์โทรศัพท์ของสาขา โดยหมายเลขโทรศัพท์นั้นจะประกอบด้วยรหัสพื้นที่ และตามด้วยหมายเลขโทรศัพท์ เป็นต้น

6. Derived Attribute คือ แอตตริบิวท์ที่ได้จากการประยุกต์ด้วยแอตตริบิวท์อื่น ๆ เช่น แอตตริบิวท์ Age ซึ่งสามารถหาอายุพนักงานได้ด้วยการนำแอตตริบิวท์ DOB (date of birth) มาประยุกต์ด้วยการนำปีปัจจุบันลบด้วยปีของ DOB



รูปที่ 2.13 เอ็นติตี้ Staff , Branch และ Next_of_Kin กับแอตตริบิวท์ของแต่ละเอ็นติตี้
(ที่มา: โอภาส เอี่ยมสิริวงศ์, 2546. หน้า 93)

2.1.7.4 การกำหนดคีย์หลักและคีย์คู่แข่ง

คีย์หลัก (Primary Key) คือ แอททริบิวท์ที่มีค่าของข้อมูลเป็นเอกลักษณ์หรือเฉพาะเจาะจง หรือเป็นค่าที่ไม่ซ้ำกันในแต่ละทูเพิล ในหนึ่งตารางจะมีข้อมูลอยู่หลายอย่าง ในการกำหนดคีย์หลักก็ควรจะเลือกข้อมูลตัวที่ไม่ซ้ำกลับข้อมูลตัวอื่น ๆ ยกตัวอย่างเช่น ลำดับเลขของเอกสาร เนื่องจากลำดับเลขของเอกสารจะไม่ซ้ำกันอยู่แล้ว เพราะฉะนั้นลำดับเลขเอกสารจึงเหมาะที่จะเป็นคีย์หลัก อาจจะใช้หลักการของ Normalization มาช่วยในการกำหนดคีย์หลักคีย์รองเพื่อที่จะลดการซ้ำซ้อนของข้อมูล

2.1.7.5 การลดความซ้ำซ้อนด้วยการ Normalization

ระดับนอร์มัลไลเซชัน เป็นกระบวนการเพื่อพัฒนาการ เชื่อมต่อของข้อมูลเพื่อแก้ปัญหาของรีเลชั่น ที่ว่าการออกแบบฐานข้อมูลทั้งทางตรรกะและทางกายภาพที่ได้ออกมาใช้ได้หรือยัง การนอร์มัลไลเซชันแบ่งออกได้เป็นหลายระดับ ได้แก่

1. First Normal Form (1 NF)

นอร์มัลไลเซชันระดับที่ 1 (First normal form : 1NF) เป็นการขจัดแอตตริบิวท์ หรือกลุ่มแอตตริบิวท์ที่ซ้ำกันไปอยู่ในเอนทิตี้ลูก เพื่อแต่ละรายการในเอนทิตี้ ไม่มีค่าของแอตตริบิวท์หรือค่าของกลุ่มแอตตริบิวท์ที่ซ้ำกัน สำหรับ 1 NF จะมีข้อเสียในการแก้ไข การลบ และการเพิ่มข้อมูล ดังนี้

- การแก้ไขข้อมูล (Update) เนื่องจากมีข้อมูลอยู่หลาย tables จะต้องแก้ไขทุก tables นั่นคือ ต้องมีการแก้ไขข้อมูลมากกว่าหนึ่งแห่ง

- การลบข้อมูล (Delete) ถ้าต้องการลบข้อมูลบางส่วนออกไป จะทำให้ลบข้อมูลอื่นออกไปด้วยโดยไม่ตั้งใจ

- การเพิ่มข้อมูล (Insert) อาจจะทำให้ไม่สามารถเพิ่มข้อมูลบางอย่างไม่ได้ หรือเพิ่มแล้วขัดแย้งกับข้อมูลเดิม

2. Second Normal Form (2NF)

ต้องเป็น First Normal Form (1NF) และต้องมี key (บางตำรา อาจจะเรียกว่า index) ที่ทุก Non - key จะต้องขึ้นอยู่ (depends on) กับ key นี้ และมีเพียง key เดียวในหนึ่งตาราง ซึ่งเรียกว่า Primary Key การที่ทุกตาราง (Table) ต้องมี Key ก็เพราะเราต้องการให้แน่ใจว่าทุกข้อมูลใน record ต่าง ๆ สามารถค้นหาได้โดยใช้ key นอร์มัลไลเซชันระดับที่ 2 (Second normal form : 2NF) เป็นการขจัดแอตตริบิวท์ที่ไม่ขึ้นกับทั้งส่วนของคีย์หลักออกไป เพื่อให้แอตตริบิวท์อื่นทั้งหมดขึ้นตรงกับส่วนที่เป็นคีย์หลักทั้งหมดเท่านั้น

3. Third Normal Form (3NF)

ต้องเป็น Second Normal Form (2NF) และไม่มี Transitive dependence หรือ เป็นการขจัดแอตตริบิวท์ที่ไม่เป็นคีย์ที่ขึ้น (Transitive dependent) ตรงกับแอตตริบิวท์อื่นที่ไม่ใช่คีย์หลักออกไปเพื่อให้แอตตริบิวท์ที่ไม่ใช่คีย์หลักต้องขึ้นตรงกับทั้งส่วนที่เป็นคีย์หลัก และไม่ขึ้นกับแอตตริบิวท์อื่นที่ไม่ใช่คีย์หลัก

4. BCNF (Boyce / Codd Normal Form)

ต้องเป็น 3NF และไม่มี attribute อื่นในรีเลชันที่สามารถระบุค่าของ attribute ที่เป็นคีย์หลัก หรือส่วนหนึ่งส่วนใดของคีย์หลักในกรณีที่คีย์หลักเป็นคีย์ผสม

โดยทั่วไปรูปแบบ BCNF จะอยู่ในรูปแบบ 3NF แต่ไม่จำเป็นเสมอไปที่รูปแบบ 3NF จะอยู่ในรูปแบบ BCNF ทั้งนี้เนื่องจากรูปแบบนี้เป็นการขยายขอบเขตของรูปแบบ 3NF ให้เหมาะสมยิ่งขึ้น โดยรูปแบบที่ต้องทำให้เป็น BCNF มักจะมีคุณสมบัติ ดังนี้

- เป็นรีเลชันที่มีคีย์คู่แข่งหลายคีย์ (Multiple Candidate Key) โดยที่

- คีย์คู่แข่งเป็นคีย์ผสม (Composite Key) และ

- คีย์คู่แข่งนั้นมีบางส่วนซ้ำซ้อนกัน (Overlapped) มี attribute บางตัวร่วมกันอยู่

5. 4NF (Forth Normal Form)

ต้องอยู่ในรูปแบบ BCNF และเป็นรีเลชันที่ไม่มีความสัมพันธ์ในการระบุค่าของ attribute แบบหลายค่าโดยที่ attribute ที่ถูกระบุค่าเหล่านี้ไม่มีความสัมพันธ์กัน (Independently Multi valued Dependency)

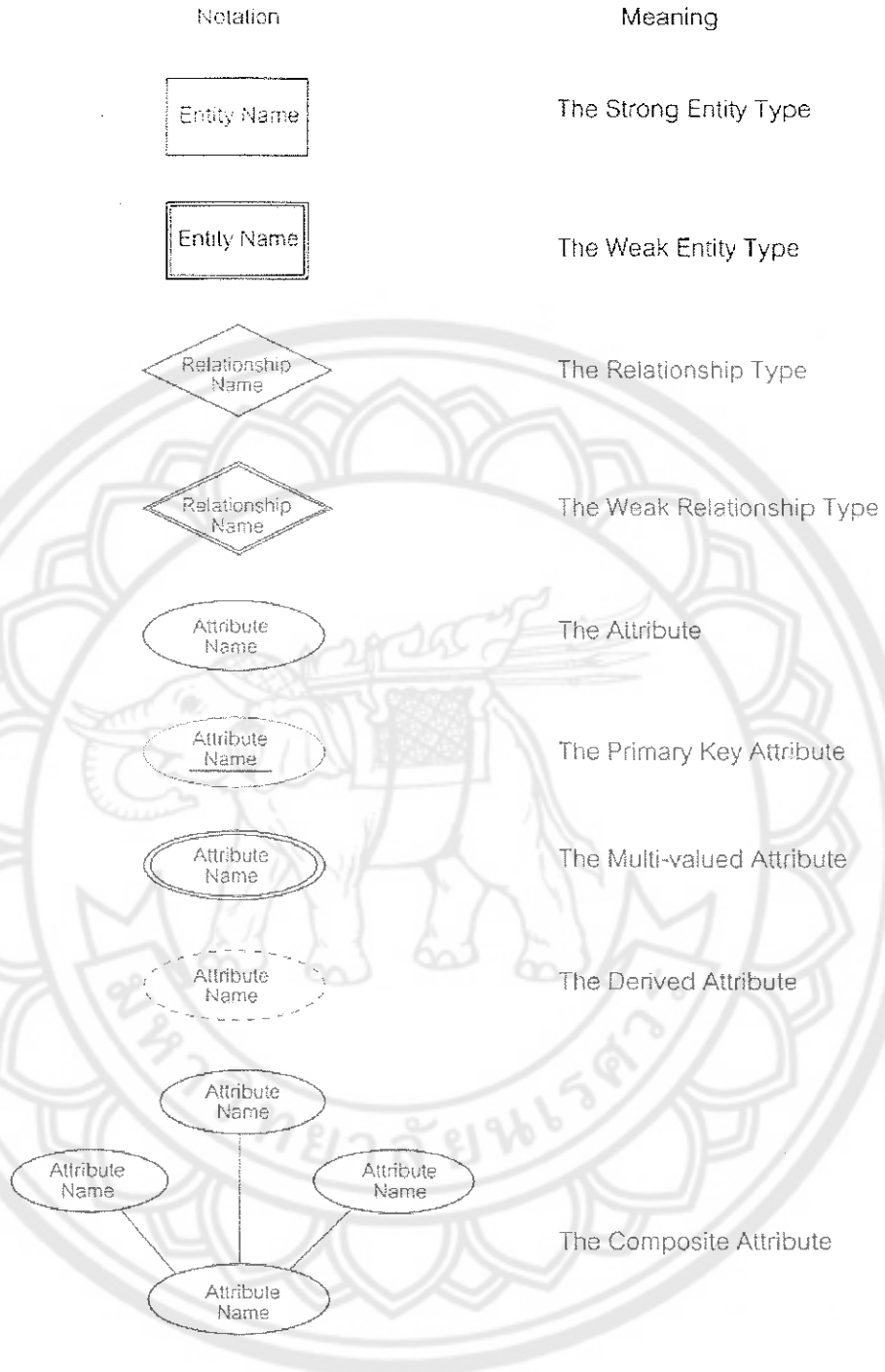
6. 5NF (Fifth Normal Form)

5NF หรือเรียกว่า Project - Join Normal Form (PJ / NF) ต้องอยู่ในรูปแบบ 4NF และไม่มี Symmetric Constraint กล่าวคือ หากมีการแตกรีเลชันออกเป็นรีเลชันย่อย (Projection) และเมื่อทำการเชื่อมโยงรีเลชันย่อยทั้งหมด (Joint) จะไม่ก่อให้เกิดข้อมูลใหม่ที่ไม่เหมือนรีเลชั่นเดิม (Spurious Tuples)

ในการแตกรีเลชันออกมาจากรูปแบบ 4NF นั้น ถ้าทำการเชื่อมโยงรีเลชันย่อยนั้นใหม่หากไม่มีข้อมูลที่แตกต่างไปจากรีเลชันเดิม ก็จะสามารถแตกรีเลชันนั้นได้ แต่ถ้าหากแตกเป็นรีเลชันย่อยแล้วเกิดข้อมูลไม่เหมือนกับรีเลชันเดิม ก็ไม่ควรแตกรีเลชัน และให้ถือว่ารีเลชันเดิมอยู่ใน 5NF แล้ว

2.1.7.6 เขียน Entity – Relationship Diagram

เนื่องจาก ER - Diagram เป็นการนำเสนอเพียงระดับแนวความคิด ดังนั้นรายละเอียดต่าง ๆ จะไม่ได้กล่าวถึง วัตถุประสงค์เพื่อให้ผู้ใช้งานสามารถมองภาพของข้อมูลในระบบได้ชัดเจนยิ่งขึ้น ซึ่งจัดเป็นเพียงหลักการและไม่ขึ้นกับ DBMS ดังรูปที่ 2.14



รูปที่ 2.14 สัญลักษณ์และความหมายใน ER - Diagram

(ที่มา: โอภาส เอี่ยมสิริวงศ์, 2546. หน้า 95)

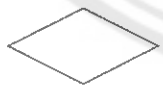
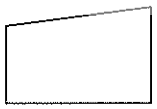
2.1.7.7 ทบทวนและตรวจสอบร่วมกับยูสเซอร์

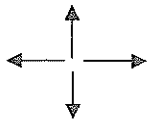
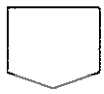
หลังจากที่เราออกแบบฐานข้อมูลเสร็จแล้ว เราควรจะไปคุยกับผู้ใช้งานว่าที่เราออกแบบฐานข้อมูลมานั้นตรงกับความต้องการของผู้ใช้งานหรือเปล่า ถ้าตรงตามที่ต้องการแล้วจึงจะเริ่มสร้างฐานข้อมูลจริง ๆ

2.2 หลักการและทฤษฎีในส่วนของโปรแกรม

2.2.1 แผนภาพการทำงานของโปรแกรม

Flow Chart หรือแผนภาพจะเป็นเครื่องมือที่โปรแกรมเมอร์ใช้ในการเปลี่ยน Algorithm ความคิดหรือความต้องการของผู้ใช้ ให้อยู่ในรูปของแผนภาพการทำงานของโปรแกรม โดยทั่วไป Flow Chart จะมีลักษณะที่ไม่ขึ้นกับภาษาคอมพิวเตอร์ใด ๆ ทำให้เราสามารถใช้ Flow Chart เป็นเสมือนเครื่องมือสื่อสารระหว่างโปรแกรมเมอร์ หรือระหว่างโปรแกรมเมอร์กับผู้ใช้ ว่าแผนงานหรือการประมวลผลของโปรแกรมจะมีลักษณะขั้นตอนตามนี้ นอกจากนั้น Flow Chart ยังเป็นเสมือนแผนภาพโดยรวมของโปรแกรม ที่เราสามารถนำไปแปลงให้เป็นภาษาคอมพิวเตอร์ก็ได้ การเขียน Flow Chart จะประกอบด้วยสัญลักษณ์หลัก ๆ ดังรูปที่ 2.15

สัญลักษณ์	ความหมาย
	เริ่มต้นการทำงานและจบการทำงาน
	การรับข้อมูลหรือการแสดงผลของข้อมูล
	การประมวลผลข้อมูล
	การตัดสินใจเลือกอย่างใดอย่างหนึ่ง
	การป้อนข้อมูลหรือคำสั่งผ่านทางคีย์บอร์ด

สัญลักษณ์	ความหมาย
	เส้นแสดงการเชื่อมต่อทางเดินของการประมวลผล
	จุดต่ออยู่ต่างหน้ากัน
	พิมพ์ข้อมูลออกทางเครื่องพิมพ์
	แสดงผลข้อมูลออกทางจอภาพ

รูปที่ 2.15 สัญลักษณ์และความหมายของแผนภาพการทำงานของโปรแกรม
(ที่มา: กฤษณา พลูสวัสดิ์ และคณะ, 2546. หน้า 20)

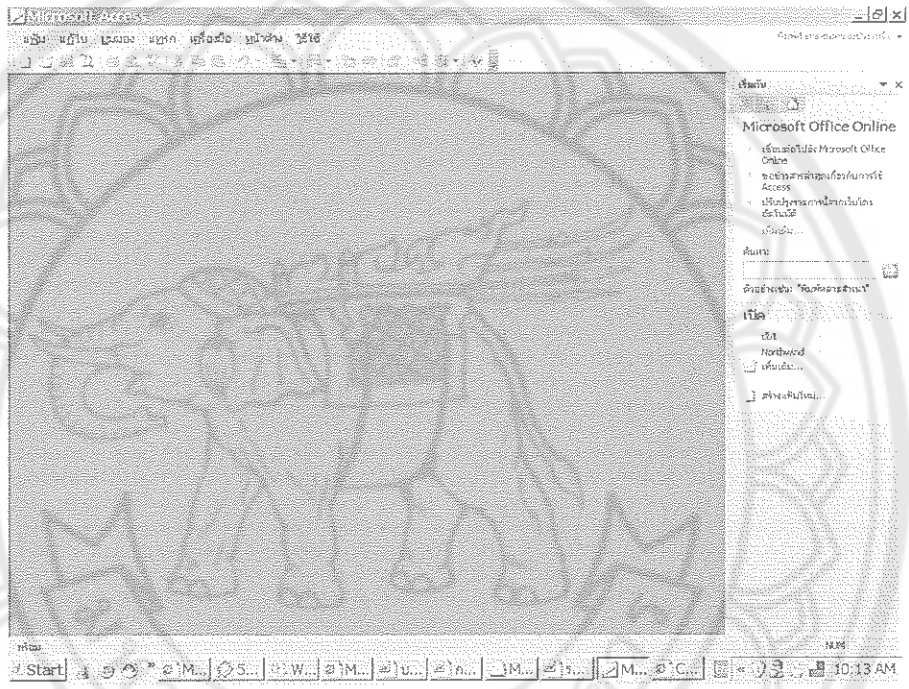
กฎการเขียนผังโปรแกรม

- ต้องใช้สัญลักษณ์ตรงกับความหมาย
- เขียนคำอธิบายในสัญลักษณ์แสดงถึงหน้าที่
- ใช้เส้นลูกศรแสดงการไหลของข้อมูล
- ผังงานโปรแกรมมองจากบนลงล่าง

2.2.2 โปรแกรม Microsoft Access

Microsoft Access เป็นโปรแกรมฐานข้อมูลที่นิยมใช้กันอย่างแพร่หลาย เนื่องจาก Access เป็นโปรแกรมฐานข้อมูลที่มีความสามารถในหลาย ๆ ด้าน ใช้งานง่าย ซึ่งผู้ใช้สามารถเริ่มทำได้ตั้งแต่ออกแบบฐานข้อมูล จัดเก็บข้อมูล เขียนโปรแกรมควบคุม ตลอดจนการทำรายงานแสดงผลข้อมูล

Microsoft Access เป็นโปรแกรมฐานที่ใช้ได้ง่าย โดยไม่จำเป็นต้องมีความเข้าใจในการเขียนโปรแกรม ก็สามารถใช้งานได้โดยไม่จำเป็นต้องศึกษารายละเอียดในการเขียนโปรแกรมให้ยุ่งยาก และสำหรับนักพัฒนาโปรแกรมมืออาชีพนั้น Microsoft Access ยังตอบสนองความต้องการในระดับสูงขึ้นไปอีก เช่น การเชื่อมต่อระบบฐานข้อมูลอื่น เช่น SQL SERVER , ORACLE แม้แต่การนำข้อมูลออกสู่ระบบเครือข่ายอินเตอร์เน็ตก็สามารถทำได้โดยง่าย ดังรูปที่ 2.16



รูปที่ 2.16 โปรแกรม Microsoft Access

2.2.2.1 ความสามารถของโปรแกรม Microsoft Access

1. สามารถสร้างระบบฐานข้อมูลใช้งานต่าง ๆ ได้โดยง่าย เพราะ Microsoft Access มีเครื่องมือต่าง ๆ ให้ใช้ในการสร้างโปรแกรมได้โดยง่ายและรวดเร็ว

2. โปรแกรมที่สร้างขึ้นสามารถตอบสนองผู้ใช้งานได้ตามต้องการ

3. สามารถสร้างระบบฐานข้อมูล เพื่อนำไปใช้ร่วมกับฐานข้อมูลอื่น ได้โดยง่าย เช่น SQL SERVER , ORACLE ได้

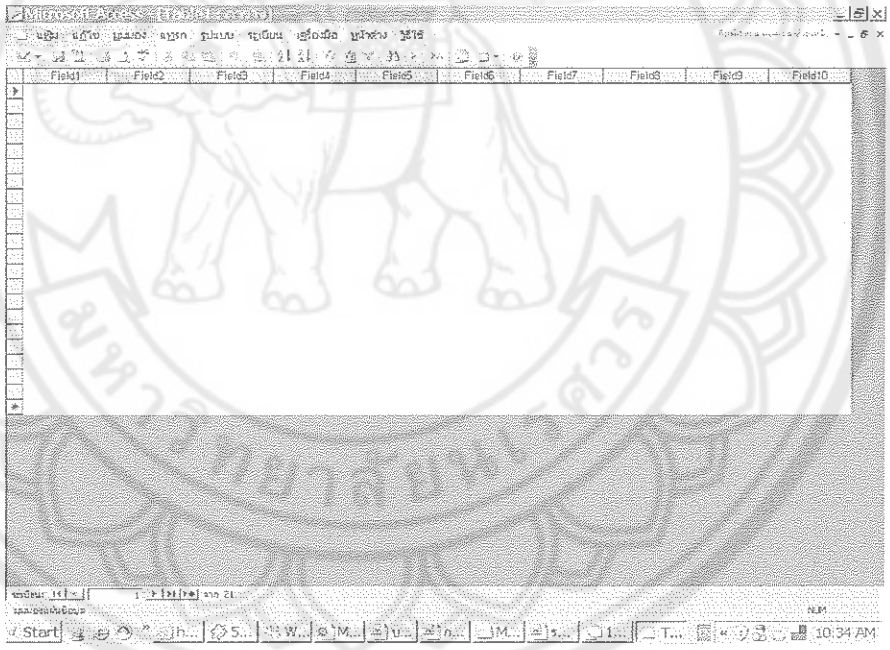
4. สามารถนำเสนอข้อมูลออกสู่ระบบเครือข่ายอินเตอร์เน็ต ก็สามารถทำได้โดยง่าย

2.2.2.2 ส่วนประกอบต่าง ๆ ของข้อมูลใน Microsoft Access

1. Table ทำหน้าที่ในการเก็บข้อมูลและเป็นแหล่งข้อมูล (Data source) ของอ๊อบเจคอื่น ได้แก่ คิวรี่ ฟอร์มและรายงาน เพื่อทำให้เข้าใจได้ง่ายถึงวิธีการจำแนกข้อมูลที่นำไปเก็บไว้ Table จึงแบ่งลักษณะการประยุกต์ Table ตาม ดังรูปที่ 2.17 วัตถุประสงค์ในการเก็บข้อมูลออกเป็น 2 ลักษณะ คือ

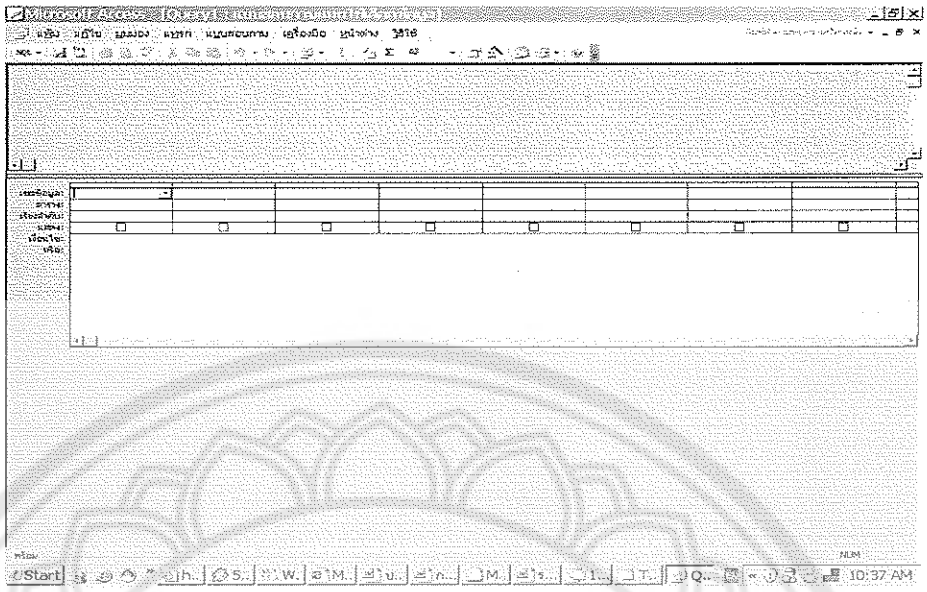
- Table เก็บข้อมูล หรือ Transaction file ข้อมูลในที่นี้ หมายถึง ข้อมูลต่าง ๆ ที่มีการบันทึกเป็นประจำ และเป็นข้อมูลที่แสดงการเคลื่อนไหวของระบบงาน เช่น รายการขายสินค้า รายการรับเข้าสินค้า การมาทำงานของพนักงาน เป็นข้อมูลที่จะนำมาวิเคราะห์และประมวลผล

- Table เก็บค่าคงที่ หรือ Master file ค่าคงที่ในที่นี้ หมายถึง ข้อมูลที่ใช้การประกอบในการวิเคราะห์ มีการเปลี่ยนแปลงน้อย และทำหน้าที่เป็นข้อมูลหลักสำหรับการอ้างอิง เช่น รายชื่อลูกค้า รายชื่อพนักงาน รหัสเครื่องจักร



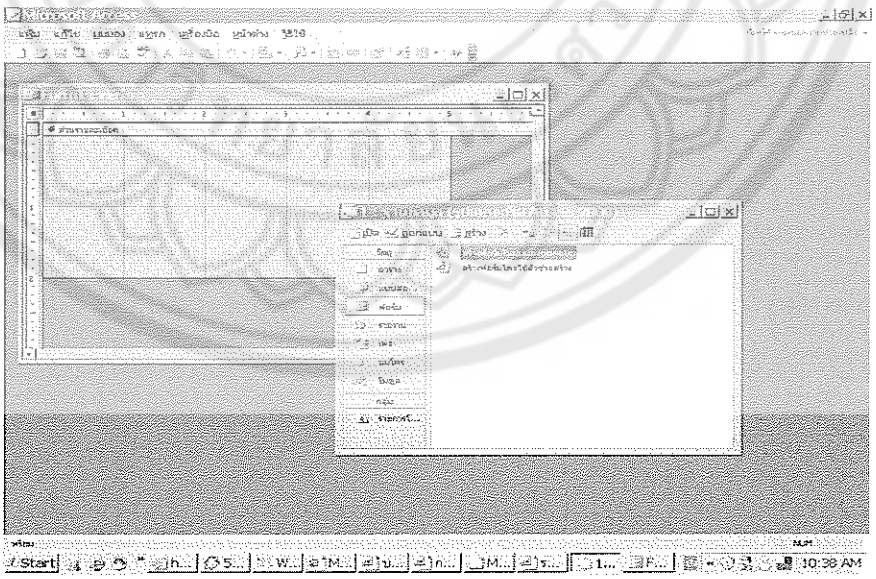
รูปที่ 2.17 วิธีสร้างตาราง

2. คิวรี่ (Query) เป็นอ๊อบเจคที่สำคัญมาก นอกจากจะเป็นแหล่งข้อมูลให้กับฟอร์มและรายงาน คิวรี่ มีชุดคำสั่งในการประมวลผล เช่น การเรียงลำดับ การหาผลรวม การคำนวณด้วยฟังก์ชัน การกำหนดเงื่อนไขคัดเลือกข้อมูล รวมถึงการแสดงผล โดยเรียกข้อมูลจากหลาย ๆ Table ที่สัมพันธ์กัน ออกมาเป็นกลุ่มข้อมูลเดียวกัน (Record set) ดังรูปที่ 2.18



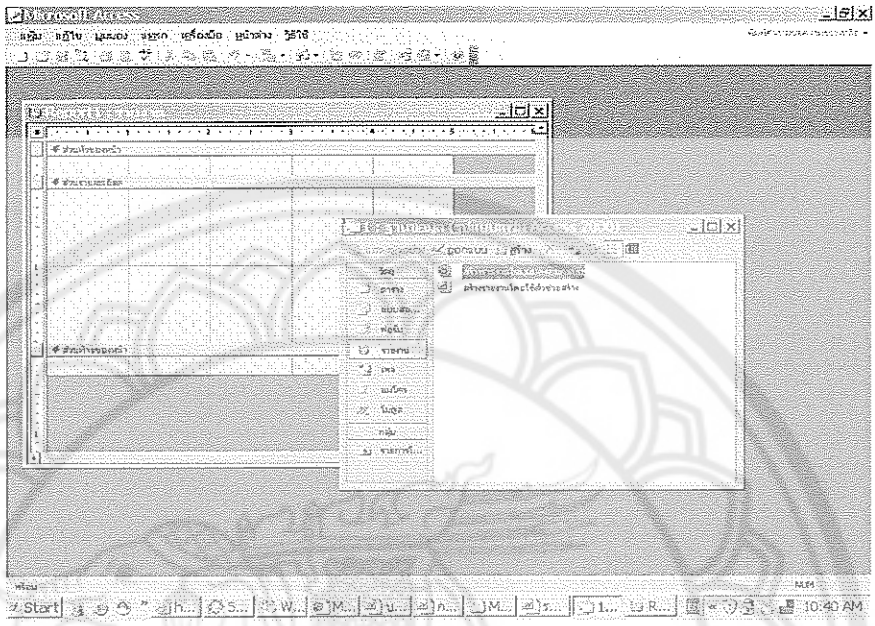
รูปที่ 2.18 วิธีสร้างคิวรี่

3. **ฟอร์ม** (Form) เป็นอ๊อบเจคที่ทำหน้าที่เป็นส่วนติดต่อกับผู้ใช้ผ่านจอภาพ ทำหน้าที่ได้ทั้งการป้อนข้อมูลและแสดงผล โดยเฉพาะการป้อนข้อมูล จะทำหน้าที่ได้ดีกว่า Table และคิวรี่ เพราะมีเครื่องมือต่าง ๆ อำนวยความสะดวกในการป้อนข้อมูลและการควบคุมความถูกต้องของค่า ดังรูปที่ 2.19



รูปที่ 2.19 วิธีสร้างฟอร์ม

4. รายงาน (Report) เป็นการแสดงผลลัพธ์ที่ได้ทำการประมวลแล้วออกมาทางเครื่องพิมพ์ ดังรูปที่ 2.20



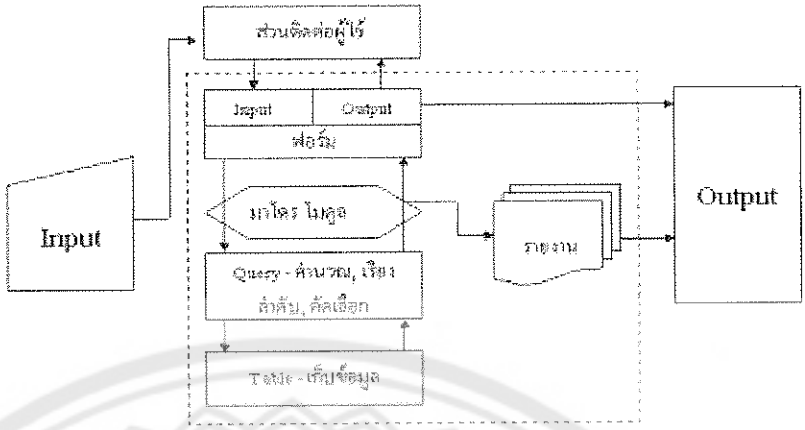
รูปที่ 2.20 วิธีสร้างรายงาน

5. มาโคร (Macro) เป็นชุดคำสั่งแบบสำเร็จรูป เพื่อจัดการและบริหารอ๊อบเจคของ Access เป็นส่วนที่ทำให้มีความสะดวกกับผู้พัฒนาโปรแกรม ในการสร้างชุดคำสั่งอย่างมาก

6. โมดูล (Module) เป็นส่วนที่ให้ผู้พัฒนาโปรแกรม เขียนชุดคำสั่งได้เอง ด้วยภาษา Visual Basic เพื่อใช้เป็นคำสั่งควบคุม การคำนวณและฟังก์ชันในการคำนวณ

7. เพจ (Access data page) เป็นอ๊อบเจคที่ทำหน้าที่เป็น ส่วนติดต่อกับผู้ใช้ในแบบ Home page เพื่อใช้งานกับเว็บ ซึ่งมีลักษณะคล้ายกับฟอร์ม

ในการพัฒนาโปรแกรมจะต้องทำเครื่องมือต่าง ๆ ของ Access มาใช้ตั้งแต่การรับข้อมูลจนถึงการแสดงผล จากเครื่องมือที่มีทำให้ผังการทำงานสามารถกำหนดเป็นรูปธรรมมากขึ้น ดังรูปที่ 2.21



รูปที่ 2.21 ผังแสดงความสัมพันธ์ของเครื่องมือ

(ที่มา: http://www.webthaidd.com/access/main.php)

ผังข้างบนได้แสดงความสัมพันธ์ของเครื่องมือต่าง ๆ ใน Access ที่นำมาประกอบขึ้นเป็นโปรแกรมฐานข้อมูล คือ มีส่วนติดต่อกับผู้ใช้ เครื่องมือในการประมวลและฐานข้อมูล