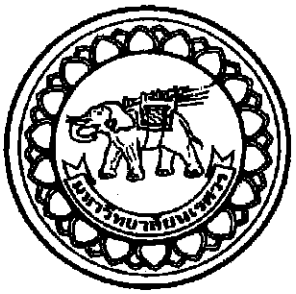


การจัดการระบบข้อมูลคลังสินค้าสำหรับโรงงานเกษตรบ้านกร่าง

Management Inventory system for Kaset - Bankrang Factory

นายวุฒิพงษ์ ปิ่นนาค รหัสประจำตัว 43360569

นายอัตถสิทธิ์ เพียรวงศ์ รหัสประจำตัว 43360684

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2546

ใบรับรองโครงงานวิศวกรรรม

หัวข้อโครงงาน การจัดการระบบข้อมูลคลังสินค้าสำหรับโรงงานเกษตรบ้านกร่าง

ผู้ดำเนินโครงงาน นายวุฒิพงษ์ ปิ่นนาค รหัสประจำตัว 43360569

นายอัตถสิทธิ์ เพียรวงศ์ รหัสประจำตัว 43360684

อาจารย์ที่ปรึกษา อาจารย์สุชาติ แย้มเม่น

สาขาวิชา วิศวกรรมคอมพิวเตอร์

ภาควิชา วิศวกรรมไฟฟ้าและคอมพิวเตอร์

ปีการศึกษา 2546

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรวิศวกรรมศาสตร์บัณฑิต สาขาวิศวกรรมคอมพิวเตอร์

คณะกรรมการสอบโครงงานวิศวกรรม

.....ประธานกรรมการ

(อาจารย์สุชาติ แย้มเม่น)

.....กรรมการ

(อาจารย์พงศ์พันธ์ กิจสนาโยธิน)

.....กรรมการ

(อาจารย์พนมขวัญ ริยะมงคล)

หัวข้อโครงงาน	การจัดการระบบข้อมูลคลังสินค้าสำหรับโรงงานเกษตรบ้านกร่าง
ผู้ดำเนินโครงงาน	นายวุฒิพงษ์ ปิ่นนาค รหัสประจำตัว 43360569
	นายอัตถสิทธิ์ เพียรวงศ์ รหัสประจำตัว 43360684
อาจารย์ที่ปรึกษา	อาจารย์สุชาติ แย้มเม่น
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2546

บทคัดย่อ

โครงงานนี้เป็นการพัฒนาระบบข้อมูลคลังสินค้าสำหรับโรงงานเกษตรบ้านกร่างให้มีประสิทธิภาพมากยิ่งขึ้น โดยมีวัตถุประสงค์เพื่อพัฒนาระบบข้อมูลคลังสินค้าด้วยการจัดสร้างโปรแกรมที่สามารถติดต่อผ่านฐานข้อมูลเพื่อดูแลข้อมูลคลังสินค้าได้

จากผลการทดสอบประสิทธิภาพของโปรแกรมการจัดการระบบข้อมูลคลังสินค้าสำหรับโรงงานเกษตรบ้านกร่าง พบว่า การทำงานของโปรแกรมสามารถทำงานได้เป็นอย่างดีแต่อย่างไรก็ตาม ทางผู้จัดทำเห็นว่าโปรแกรมยังมีข้อจำกัด โดยที่ในอนาคตสามารถที่จะมีการนำเอามาพัฒนาต่อไปเพื่อให้สามารถช่วยพยากรณ์และวางแผนการผลิตได้

Project Title	Management Inventory system for Kaset - Bankrang Factory		
Name	Mr.Vuttipong	Pinnak	ID. 43360569
	Mr.Athasit	Pianwong	ID. 43360684
Project Advisor	Mr.Suchart	Yammen	
Major	Computer Engineering		
Department	Electrical and Computer Engineering		
Academic Year	2003		

ABSTRACT

The purpose of this project is the development of management inventory system for Kaset – Bankrang factory to increase the efficiency of administration by using a program via interactive database.

In term of the efficiency of the program of the management inventory system for Kaset – Bankrang factory, we found that the program has a good satisfaction in general. However, we think that this program has a limitation; for example, this program has not yet had forecasting and production planning etc. For development of the program to have more completely, we should also create a program which runs a forecasting and production planning.

กิตติกรรมประกาศ

โครงงานฉบับนี้สำเร็จลงได้ด้วยความอนุเคราะห์อันดียิ่งจาก ท่านอาจารย์สุชาติ แย้มเม่น ที่ได้กรุณาเป็นอาจารย์ที่ปรึกษาให้แนวคิดช่วยเหลือให้ความเอาใจใส่ตลอดจนสละเวลาอันแสนมีค่าเพื่อตรวจแก้ไขข้อบกพร่องต่างๆจนสำเร็จลุล่วงไปด้วยดีอีกทั้งยังคอยอบรมสั่งสอนผู้จัดทำให้เป็นคนดีมีระเบียบวินัยผู้จัดทำขอขอบพระคุณเป็นอย่างสูงไว้ ณ โอกาสนี้

ทั้งนี้ขอกราบขอบพระคุณ อาจารย์ทุกท่านที่ได้อบรมสั่งสอนให้ความรู้และคำแนะนำที่ดีเสมอมาและขอขอบพระคุณ คุณลุงชลอ เจ้าของโรงงานเกษตรบ้านกร่างและพี่ๆในโรงงานที่ได้ให้โอกาสและความช่วยเหลือต่างๆในเรื่องของข้อมูลเป็นอย่างดี รวมทั้งเพื่อนๆพี่ๆวิศวกรรมคอมพิวเตอร์ทุกคนที่ให้ความช่วยเหลือไม่ว่าเรื่องของโปรแกรมหรือการจัดทำเอกสาร

ท้ายนี้ขอกราบขอบพระคุณ บิดา-มารดา ผู้ที่ทุ่มเททั้งชีวิตจิตใจเพื่อให้การศึกษาที่ดีที่สุดและสิ่งต่างๆที่ดีที่สุดแก่ผู้จัดทำ เป็นผู้ที่เฝ้าคอยดูความสำเร็จแต่ละก้าวย่างของผู้จัดทำและยังคอยเป็นห่วงเป็นใยและให้กำลังใจเสมอมา จนสามารถทำงานครั้งนี้จนสำเร็จลุล่วงด้วยดี

นายวุฒิพงษ์ ปิ่นนาค
นายอัตถสิทธิ์ เพียรวงศ์

สารบัญ

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญ(ต่อ)

สารบัญตาราง

สารบัญรูป

รูปที่ หน้า

สารบัญรูป(ต่อ)

สารบัญรูป(ต่อ)

บทที่ 1
บทนำ

ในบทนี้จะกล่าวถึงความเป็นมาของโครงงานการจัดการระบบสินค้าคงคลังสำหรับโรงงานเกษตรบ้านกร่าง, วัตถุประสงค์, ขอบข่าย, ขั้นตอนการดำเนินงาน, ผลที่คาดว่าจะได้รับ

1.1 ที่มาและความสำคัญของโครงงาน

หลังจากที่อาจารย์ที่ปรึกษาพร้อมด้วยกลุ่มของข้าพเจ้าได้เดินทางเพื่อสอบถามรายละเอียดในเบื้องต้นของโครงงานที่โรงงานเกษตรบ้านกร่างและเมื่อได้ทำการพูดคุยและสำรวจดูโดยรอบๆของโรงงานแล้วนั้นในเบื้องต้นทำให้ได้ทราบว่าทางโรงงานนั้นมีการก่อตั้งและดำเนินการมาแล้วเป็นเวลานานร่วมสิบปีแล้วโดยที่มี คุณลุงชลอ เจ้าของโรงงานเป็นผู้ดูแล สภาพโดยทั่วไปที่เห็นและจากการได้สัมผัสพูดคุย บรรยากาศภายในโรงงานนั้นจะค่อนข้างเป็นกันเองเป็นการทำงานแบบเป็นครอบครัวซึ่งทางเจ้าของโรงงานได้เปรยว่าบางครั้งแล้วก็ทำให้การควบคุมดูแลนั้นเป็นไปได้ลำบากด้วยความที่เป็นเหมือนลูกเหมือนหลาน และบางครั้งก็จะมีพวกลักเล็กขโมยน้อยทำให้วัตถุดิบบางส่วนขาดหายไปโดยที่ไม่อาจตรวจสอบได้

อย่างที่เราทราบกันโดยทั่วไปว่าประเทศไทยถือเป็นประเทศเกษตรกรรมโดยแท้เพราะไม่ว่ากี่ยุคกี่สมัยจะเปลี่ยนแปลงไปการเกษตรก็ยังคงมีบทบาท และความสำคัญที่ไม่เคยหายไปจากวิถีชีวิตของคนไทย เพราะว่าผลผลิตที่ได้จากการทำการเกษตรนั้นถือเป็นส่วนสำคัญเพราะคนไทยต่างมีความต้องการผลผลิตทางการเกษตรเหล่านั้นโดยเฉพาะอย่างยิ่งข้าว เหตุเพราะว่าคนไทยส่วนมากแทบจะทั้งหมดรับประทานข้าวเป็นอาหารหลัก และยังมีข้าวหอมมะลิของไทยก็เป็นที่ยอมรับของต่างชาติ จึงทำให้หน่วยงานและองค์กรต่างๆที่เข้ามามีบทบาทหน้าที่เข้ามาเกี่ยวข้องในด้านนี้นั้นจึงได้สนใจ และให้ความสำคัญในเรื่องของการควบคุมดูแลให้การผลิตในขั้นตอนต่างๆเป็นไปได้ด้วยดีเทคโนโลยีต่างๆจึงได้มีการคิดค้นพัฒนาขึ้นเรื่อย ๆ เพื่อรองรับความต้องการทั้งภายในและภายนอกประเทศซึ่งเราจึงได้มีโอกาสได้เห็นโครงการต่างๆที่เข้ามารองรับ และช่วยเหลือดูแลทั้งในภาคของเกษตรกรรมที่เป็นพื้นฐาน และในส่วนของภาคอุตสาหกรรมที่เป็นส่วนรองรับไม่ว่าจะเป็นในส่วนที่ทำการแปรรูปผลิตผลทางการเกษตรหรือกลุ่มอุตสาหกรรมที่เป็นส่วนการผลิตอุปกรณ์ทางการเกษตรเพื่อนำมาใช้ในวงจรการผลิตทั้งในส่วนของขนาดใหญ่ และขนาดย่อม

ปัจจุบันเทคโนโลยีได้เปลี่ยนแปลงไปมากความสะดวกสบายล้วนแล้วแต่เป็นที่ต้องการของทุกคนความเปลี่ยนไปนี้ส่งผลให้การทำการเกษตรได้เปลี่ยนแปลงตามไปจากเดิมที่การเกษตรต้องพึ่งพาแรงงานของเพื่อนบ้านที่ต้องมีการช่วยเหลือผลัดเปลี่ยนกันไปที่เราเคยได้รู้จักวัฒนธรรม

การเอาแรง หรือลงแขก ซึ่งปัจจุบันนั้นค่อยๆหายไปจากสังคมไทยและแปรเปลี่ยนไปเป็นการว่าจ้างจบจนทุกวันนี้ได้มีเครื่องมือที่เข้ามาช่วยอำนวยความสะดวกทำให้การทำการเกษตรมีความสะดวกและประหยัดในส่วนของแรงงานมากขึ้นโรงงานเกษตรบ้านกร่างถือได้ว่าเป็นหนึ่งในอีกหลายโรงงานที่มีบทบาทในการผลิต และประกอบอุปกรณ์เครื่องไม้เครื่องมือเพื่อการเกษตร โดยที่เน้นไปที่อุปกรณ์ใช้ในการทำนาปลูกข้าว ซึ่งจากเดิมที่การแข่งขันยังไม่มีมากมายอย่างเช่นปัจจุบัน อีกทั้งขนาดของโรงงานมีขนาดเล็กจำนวนลูกจ้างไม่มากการดูแลจัดการเป็นไปได้โดยสะดวกแต่ปัจจุบันด้วยขนาดโรงงานที่ได้มีการขยายใหญ่ขึ้นจำนวนลูกจ้างมากขึ้นและการแข่งขันที่มีมากขึ้นทำให้การจัดการดูแลเป็นไปอย่างลำบากซึ่งจากเดิมนั้นการดูแลและตรวจสอบวัตถุดิบและสินค้าที่มีอยู่ภายในคลังไม่ใช่เรื่องที่ต้องสนใจแต่ตอนนี้เป็นเรื่องที่จำเป็นแต่การดูแล และจัดการไม่ได้รับการดูแลมาก่อนทำให้เรื่องการดูแลคงคลังเป็นเรื่องยากทำให้การตรวจสอบและตัดสินใจต่าง ๆ ไม่สามารถทำได้เต็มที่

หลังจากนั้นกลุ่มของข้าพเจ้าก็ได้ปรึกษากับท่านอาจารย์ที่ปรึกษาเพื่อที่จะวางแผนเพื่อที่จะกำหนดแนวทางในการศึกษา และขอบเขตของงานที่จะต้องทำโดยมุ่งเน้นไปที่การตรวจสอบและดูแลคงคลังซึ่งถือเป็นส่วนสำคัญของงานในครั้งนี้โดยเห็นว่าการนำเอาความรู้เกี่ยวกับซึ่งเกี่ยวกับระบบฐานข้อมูลและการออกแบบโปรแกรมเพื่อรองรับการทำงานนำเข้ามาช่วยในการจัดการถือเป็นแนวทางที่สะดวก และเป็นผลดีต่อโรงงานเพราะจะช่วยทำให้การดำเนินงานสามารถที่จะดูแลและควบคุมได้สะดวกดียิ่งขึ้นเพราะเจ้าของโรงงานสามารถที่จะตรวจสอบข้อมูลต่าง ๆ ได้เพียงแค่นั่งอยู่ที่หน้าจอหลังเลิกงาน โดยที่จะมีเจ้าหน้าที่ที่ทำหน้าที่คอยตรวจสอบคงคลังทุกวัน

1.2 วัตถุประสงค์ของโครงงาน

1. เพื่อเพิ่มศักยภาพของระบบการผลิตเดิมที่มีอยู่ ให้มีประสิทธิภาพมากขึ้น โดยการนำเอาความรู้ทางด้านการวิเคราะห์ระบบและการจัดการข้อมูลของคงคลังมาประยุกต์ใช้
2. เพื่อฝึกแก้ปัญหาจากประสบการณ์จริง
3. เพื่อศึกษาการจัดทำและติดต่อกับฐานข้อมูลที่ได้ทำการรวบรวม
4. เพื่อศึกษาการสร้างโปรแกรมที่เอื้อประโยชน์ต่อระบบใหม่ที่ได้ผ่านการวิเคราะห์แล้ว

1.3 ขอบข่ายของโครงงาน

โปรแกรมสามารถตรวจสอบและดูแลเกี่ยวกับปริมาณวัตถุดิบที่เข้ามาตลอดจนผลิตภัณฑ์สำเร็จที่ผ่านกระบวนการผลิตแล้ว

1.3.1 จัดทำระบบฐานข้อมูลภายในโรงงานคือ สินค้าและ วัตถุดิบ

1.3.2 จัดทำซอฟท์แวร์โดยใช้ Visual Basic 6.0ซึ่งคุณสมบัติของโปรแกรมที่พัฒนามีดังนี้

- ทำงานบน Windows 98
- ทำงานร่วมกับฐานข้อมูล ODBC / My SQL

1.4 กิจกรรมการดำเนินการ

กิจกรรม	ปี 2545		ปี 2546									
	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.	มิ.ย.	ก.ค.	ส.ค.	ก.ย.	ต.ค.
1. เริ่มศึกษาและวางแผนการทำงาน	←	→										
2. เก็บข้อมูลจากโรงงาน	←					→						
3. วิเคราะห์และแก้ไขระบบที่ได้			←				→					
4. ศึกษาโปรแกรมที่จะนำมาสร้างแอพพลิเค	←					→						
5. ศึกษาโปรแกรมที่ใช้สร้างฐานข้อมูล	←					→						
6. สร้างโปรแกรมจากระบบที่ได้			←					→				
7. ตรวจสอบโปรแกรมปรับปรุงและแก้ไขปัญหาที่เกิดขึ้น					←						→	
8. จัดทำเอกสาร			←									→
9.ส่งโครงงานฉบับสมบูรณ์											←	→

1.5 ผลที่คาดว่าจะได้รับ

- โปรแกรมสามารถตรวจสอบเกี่ยวกับของคงคลังได้สะดวก
- โปรแกรมสามารถแสดงข้อมูลเพื่อตรวจสอบการผลิต
- ได้รับความรู้เกี่ยวกับการเขียนโปรแกรมติดต่อฐานข้อมูล และ ระบบฐานข้อมูล

- ได้รับความรู้ทางด้านการบริหารและจัดการของคงคลัง

1.6 งบประมาณที่ใช้

- ค่าอุปกรณ์ และเอกสาร 1,000 บาท
- ค่าใช้จ่ายวัสดุคอมพิวเตอร์ 800 บาท

* หมายเหตุ ทุกรายการสามารถถัวเฉลี่ย

บทที่ 2
หลักการและทฤษฎีที่เกี่ยวข้อง

ในบทนี้จะกล่าวถึงหลักการ และทฤษฎีที่เกี่ยวข้องเพื่อใช้ในการศึกษา และพัฒนาโครงงาน ได้แก่ทฤษฎีการบริหารคงคลัง, ระบบฐานข้อมูล, หลักการเขียนโปรแกรมด้วย Visual basic 6.0 เพื่อใช้ในการติดต่อกับฐานข้อมูล

2.1 หลักการของการบริหารของคงคลัง (Inventory Management)

การบริหารของคงคลังนับได้ว่าเป็นสิ่งที่มีความสำคัญมากในเกือบจะทุก ๆ ประเภทของธุรกิจ ไม่ว่าจะเป็นธุรกิจที่เกี่ยวข้องกับสินค้า หรือการให้บริการ ในด้านของการผลิตนั้น จะต้องควบคุมต้นทุนค่าใช้จ่ายในการดำเนินงาน ซึ่งต้นทุนประเภทที่มีความสำคัญเป็นอย่างมาก ก็คือค่าใช้จ่ายที่ลงทุนไปกับวัตถุดิบ วัสดุสิ้นเปลือง งานระหว่างการผลิต และผลิตภัณฑ์สำเร็จรูปที่ยังไม่ได้ทำการจัดส่งถ้าหากว่าการลงทุนในค่าใช้จ่ายเหล่านี้มีมากจนเกินไปก็จะส่งผลให้ค่าใช้จ่ายของเงินทุนสูง ค่าใช้จ่ายของการดำเนินงานสูง และทำให้ประสิทธิภาพของการผลิตลดลง เมื่อมีการใช้พื้นที่มากเกินไปในการดูแลรักษาของคงคลัง[4]

การควบคุมของคงคลังจึงถือว่าเป็นสิ่งที่ไม่อาจจะมองข้ามไปได้ยิ่งควรที่จะให้ความสำคัญเป็นพิเศษ ทั้งนี้ก็เพราะของคงคลังเป็นทรัพย์สินที่มีมูลค่าสูงที่สุดในกลุ่มของทรัพย์สินหมุนเวียนของการผลิต ปัญหาที่เกิดขึ้นในการควบคุมของคงคลังอาจเป็นสาเหตุหนึ่งที่นำไปสู่ความล้มเหลวทางธุรกิจได้ ในธุรกิจอุตสาหกรรมนั้น ถ้าวัตถุดิบ และชิ้นส่วนประกอบต่างๆ มีอยู่ไม่เพียงพอกับความต้องการของการผลิตแล้ว ก็อาจจะทำให้เกิดปัญหาถึงขั้นการผลิตหยุดชะงักได้ และอาจส่งปัญหาถึงขั้นการส่งสินค้าไม่ทันตามกำหนดเวลาซึ่งเป็นปัญหาที่จะส่งผลกระทบอย่างมากต่อธุรกิจเพราะจะทำให้ลูกค้าขาดความเชื่อถือ ในทางกลับกันถ้าหากเราต้องการที่จะพยายามมีของคงคลังเอาไว้มากๆ เพื่อป้องกันการเกิดปัญหาการขาดแคลนวัตถุดิบ ชิ้นส่วน หรือผลิตภัณฑ์สำเร็จรูป ก็จำเป็นที่จะต้องใช้เงินเป็นมูลค่ามหาศาล ในการควบคุมของคงคลังที่ดีนั้นจึงเป็นเรื่องที่เกี่ยวข้องกับความพยายามในการทำให้วัตถุประสงค์ 2 ประการ ในการดำเนินการให้มีของคงคลังเกิดความสมดุลในระดับที่จะเหมาะสมที่สุด วัตถุประสงค์ประการแรกนั้นก็คือ เพื่อให้การลงทุนทั้งสิ้นในของคงคลังต่ำที่สุดไม่เกิดการจมทุนกับการจัดเก็บไว้เกินกว่าความจำเป็น ส่วนวัตถุประสงค์ประการที่สอง คือการที่พยายามทำให้ระดับการบริการลูกค้าและการให้บริการแผนกผลิตสูงที่สุด ดังนั้นในการควบคุมของคงคลังที่ดีย่อมทำให้เกิดผลดีทั้งในแง่ของการเพิ่มประสิทธิภาพและลดค่าใช้จ่ายในการดำเนินงาน

2.1.1 ประเภทและความสำคัญของของคงคลัง

เมื่อเรามองของคงคลังในมุมของการผลิต สามารถที่จะจัดแบ่งประเภทของคงคลังออกได้เป็น 4 ประเภท ดังนี้

1. วัตถุดิบและชิ้นส่วนที่สั่งซื้อ (Raw Materials and Purchased Components)

- ของคงคลังเหล่านี้เป็นวัสดุขั้นต้นที่ใช้ในการทำชิ้นส่วนและผลิตภัณฑ์สำเร็จรูป สำหรับชิ้นส่วนที่สั่งซื้อก็เปรียบเสมือนวัตถุดิบ แตกต่างกันก็เพียงแต่ว่า บริษัทภายนอกเป็นผู้ดำเนินการผลิตชิ้นส่วนนั้นทั้งหมดหรือเพียงบางส่วน

2. ของคงคลังระหว่างกระบวนการผลิต (In-process Inventory)

- หลังจากที่กระบวนการผลิตเริ่มต้นโดยการนำวัตถุดิบ และชิ้นส่วนประกอบที่สั่งซื้อจากภายนอกเข้าสู่กระบวนการผลิต จะมีอยู่ช่วงเวลาหนึ่ง ก่อนที่กระบวนการผลิตจะเสร็จสิ้น ช่วงเวลาระหว่างนั้น ของคงคลังเหล่านั้นอยู่ในระหว่างกระบวนการผลิต เพื่อรอคอยการผลิตขั้นต่อไปให้เป็นผลิตภัณฑ์สำเร็จรูป

3. ผลิตภัณฑ์สำเร็จรูป (Finished Product)

- ผลิตภัณฑ์สำเร็จรูปอาจจะเก็บในโรงงานหรือในคลังสินค้าก่อนที่จะส่งให้กับลูกค้า ของคงคลังประเภทนี้ประกอบด้วยชิ้นส่วนเพื่อบริการและผลิตภัณฑ์ขั้นสุดท้าย

4. ของคงคลังที่เป็นเครื่องมือและชิ้นส่วนเพื่อการซ่อมบำรุงและซ่อมแซม (Maintenance , Repair and Tooling Inventories)

- ของคงคลังเหล่านี้ได้แก่ เครื่องมือกัด และอุปกรณ์จับยึดชิ้นงานที่ใช้กับเครื่องจักรในโรงงาน และชิ้นส่วนเพื่อการซ่อมแซมที่จำเป็นต่อการปรับเครื่องจักรเมื่อเครื่องจักรเกิดเสียหายขึ้นมา รวมทั้งชิ้นส่วนที่เป็นอะไหล่เครื่องไฟฟ้าก็รวมอยู่ในของคงคลังประเภทนี้ด้วยของคงคลังเหล่านี้มีส่วนสำคัญในการดำเนินงานมาก โดยเฉพาะในโรงงานอุตสาหกรรมซึ่งพอสรุปให้เห็นคร่าว ๆ ได้ดังนี้

ของคงคลังที่เป็นผลิตภัณฑ์สำเร็จรูป

- ช่วยป้องกันความผิดพลาดอันเกิดมาจากการที่ความต้องการมีมากกว่าที่พยากรณ์ไว้
- ช่วยให้การผลิตเป็นไปได้อย่างสม่ำเสมอ ไม่ต้องเปลี่ยนแปลงไปตามฤดูกาลเหมือนต้องการผลิตภัณฑ์

ของคงคลังระหว่างกระบวนการผลิต

- ช่วยให้การผลิตในแต่ละหน่วยผลิตสามารถดำเนินไปได้อย่างต่อเนื่องโดยไม่จำเป็นต้องพึ่งพิงกันมากนัก
- ช่วยให้การผลิตดำเนินไปได้อย่างสม่ำเสมอ ถึงแม้ว่าการทำงานในแต่ละหน่วยผลิตจะมีความเร็วไม่เท่ากัน
- เพื่อป้องกันการขาดแคลนวัตถุดิบหรือชิ้นส่วน อันเนื่องจากการล่าช้าด้วยเหตุผลหลาย

ประการ

2.1.2 เหตุผลและความจำเป็นที่ต้องมีของคงคลัง

- เพื่อลดค่าใช้จ่ายในการขนส่งและการผลิต
- เพื่อให้การผลิตสามารถดำเนินไปได้อย่างต่อเนื่อง
- ปรับให้เกิดความสมดุลระหว่างความต้องการที่เกิดขึ้นและการจัดหาของคงคลังเข้ามาเก็บไว้ในคลัง
- เพื่อตอบสนองความต้องการของตลาดที่มีความไม่แน่นอน

2.2 หลักการและทฤษฎีของระบบฐานข้อมูล (Database System)

คำว่าฐานข้อมูลนั้นออกจะเป็นคำที่ค่อนข้างคุ้นหูสำหรับหลายๆคนสำหรับเรื่องที่มาของฐานข้อมูลนั้น ก็มาจาก ข้อมูล (data) ซึ่ง หมายถึง ตัวอักษร ข้อความ ตัวเลข หรือสัญลักษณ์ต่างๆ ที่ใช้แทนข้อเท็จจริงสิ่งใดสิ่งหนึ่ง ส่วนคำว่า ฐานข้อมูล (database) นั้น หมายถึงกลุ่มข้อมูลต่างๆ ที่มีความสัมพันธ์กันและนำมาเก็บรวบรวมไว้ในที่เดียวกัน โดยมีหัวเรื่องหรือจุดประสงค์อย่างใดอย่างหนึ่ง เพื่อความสะดวกในการเรียกใช้งานภายหลัง ซึ่งฐานข้อมูลเหล่านี้อาจจะเก็บอยู่ในรูปแฟ้มเอกสารหรือเก็บไว้ในคอมพิวเตอร์[9]

ฐานข้อมูลเป็นการรวบรวมข้อมูลต่างๆ อย่างมีโครงสร้างและมีความสัมพันธ์กัน ไว้ด้วยกัน โดยสามารถที่จะจัดการกับข้อมูลนั้นได้อย่างสะดวก และรวดเร็ว ส่วนประกอบที่เป็นพื้นฐานของฐานข้อมูลคือ ตาราง (Table) ซึ่งตารางจะมีการจัดเรียงแบบเป็นแถว (Row) และคอลัมน์ (Column)

ในฐานข้อมูลจะมีการจัดเก็บข้อมูลจาก Field ถ้าหลายๆ Fields รวมกันเรียกว่า Record หลายๆ Records รวมกันเรียกว่า Table ซึ่งหลายๆ Tables รวมกันเรียกว่า Database หรือ ฐานข้อมูล การรวบรวมข้อมูลต่างๆอย่างมีโครงสร้าง และมีความสัมพันธ์กัน มาไว้ด้วยกันอย่างมีระบบ โดยสามารถที่จะจัดการกับระบบฐานข้อมูลนั้นได้อย่างสะดวกและรวดเร็วขึ้น (บางครั้งฐานข้อมูลอาจมีเพียงตารางเดียว)

ข้อมูลต่างๆ ที่ถูกจัดเก็บเป็นฐานข้อมูลนอกจากจะต้องเป็นข้อมูลที่มีความสัมพันธ์กันแล้ว ยังจะต้องเป็นข้อมูลที่ใช้สนับสนุนการดำเนินงานอย่างน้อยอย่างใดอย่างหนึ่งขององค์กร ดังนั้นจึงอาจกล่าวได้ว่าแต่ละฐานข้อมูลจะเทียบเท่ากับระบบแฟ้มข้อมูล 1 ระบบและจะเรียกฐานข้อมูลที่จัดทำขึ้นเพื่อสนับสนุนการดำเนินงานอย่างใดอย่างหนึ่งว่า "ระบบฐานข้อมูล" (Database System)

ในปัจจุบันการจัดโครงสร้างข้อมูลให้เป็นแบบฐานข้อมูลกำลังเป็นที่นิยม เกือบทุกหน่วยงานที่มีการใช้ระบบสารสนเทศจะจัดทำข้อมูลให้เป็นแบบฐานข้อมูล เนื่องจากปริมาณข้อมูลมีมากถ้าจัดข้อมูลเป็นแบบแฟ้มข้อมูลจะทำให้มีแฟ้มข้อมูลเป็นจำนวนมาก ซึ่งจะทำให้เกิดข้อมูลที่

ซ้ำซ้อนกันได้ ข้อมูลที่ซ้ำซ้อนนี้จะก่อให้เกิดปัญหามากมาย องค์กรส่วนใหญ่หันมาให้ความสนใจกับระบบฐานข้อมูลกันมาก เนื่องจากระบบฐานข้อมูลมีประโยชน์ดังต่อไปนี้

1. ลดความซ้ำซ้อนของข้อมูล

เนื่องจากการใช้งานของระบบฐานข้อมูลนั้นต้องมีการออกแบบฐานข้อมูลเพื่อให้มีความซ้ำซ้อนของข้อมูลน้อยที่สุด จุดประสงค์หลักของการออกแบบระบบฐานข้อมูลเพื่อการลดความซ้ำซ้อน สาเหตุที่ต้องลดความซ้ำซ้อน เนื่องจากความยากในการปรับปรุงข้อมูล กล่าวคือถ้าเก็บข้อมูลซ้ำซ้อนกันหลายแห่ง เมื่อมีการปรับปรุงข้อมูลแล้วปรับปรุงข้อมูลไม่ครบทำให้ข้อมูลเกิดความขัดแย้งกันของข้อมูลตามมา และยังเปลืองเนื้อที่การจัดเก็บข้อมูลด้วย เนื่องจากข้อมูลชุดเดียวกันจัดเก็บซ้ำกันหลายแห่งนั่นเอง ถึงแม้ว่าความซ้ำซ้อนช่วยให้ออกรายงานและตอบคำถามได้เร็วขึ้น แต่ข้อมูลจะเกิดความขัดแย้งกัน ในกรณีที่ต้องมีการปรับปรุงข้อมูลหลายแห่งการออกรายงานจะทำได้เร็วเท่าใดนั้นจึงไม่มีความหมายแต่อย่างใด และเหตุผลที่สำคัญอีกประการหนึ่งคือปัญหาเรื่องความขัดแย้งกันของข้อมูลที่แก้ไขไม่ได้ด้วยฮาร์ดแวร์ ขณะที่การทำงานช้านั้นใช้ความสามารถของฮาร์ดแวร์ช่วยได้

2. รักษาความถูกต้องของข้อมูล

เนื่องจากระบบจัดการฐานข้อมูลสามารถตรวจสอบกฎบังคับความถูกต้องของข้อมูลให้ได้โดยนำกฎเหล่านั้นมาไว้ที่ฐานข้อมูล ซึ่งถือเป็นหน้าที่ของระบบจัดการฐานข้อมูลที่จะจัดการเรื่องความถูกต้องของข้อมูลให้แทน แต่ถ้าเป็นระบบแฟ้มข้อมูลผู้พัฒนาโปรแกรมต้องเขียนโปรแกรมเพื่อควบคุมกฎระเบียบต่างๆ (data integrity)เองทั้งหมด ถ้าเขียนโปรแกรมครอบคลุมกฎระเบียบใดไม่ครบหรือขาดหายไปบางกฎอาจทำให้ข้อมูลผิดพลาดได้ และยังช่วยลดค่าใช้จ่ายในการบำรุงรักษาและพัฒนาโปรแกรมด้วย เนื่องจากระบบจัดการฐานข้อมูลจัดการให้นั่นเอง เนื่องจากระบบจัดการฐานข้อมูลสามารถรองรับการใช้งานของผู้ใช้หลายคนพร้อมกันได้ ดังนั้นความคงสภาพและความถูกต้องของข้อมูลจึงมีความสำคัญมากและต้องควบคุมให้ดีเนื่องจากผู้ใช้อาจเปลี่ยนแปลงแก้ไขข้อมูลได้ ซึ่งจะทำให้เกิดความผิดพลาดกระทบต่อการใช้ข้อมูลของผู้ใช้อื่นทั้งหมดได้ ดังนั้นประโยชน์ของระบบฐานข้อมูลในเรื่องนี้จึงมีความสำคัญมาก

3. มีความเป็นอิสระของข้อมูล

เนื่องจากมีแนวคิดที่ว่าทำอย่างไรให้โปรแกรมเป็นอิสระจากการเปลี่ยนแปลงโครงสร้างข้อมูล ในปัจจุบันนี้ถ้าไม่ใช้ระบบฐานข้อมูลการแก้ไขโครงสร้างข้อมูลจะกระทบถึงโปรแกรมด้วย เนื่องจากในการเรียกใช้ข้อมูลที่เก็บอยู่ในระบบแฟ้มข้อมูลนั้น ต้องใช้โปรแกรมที่เขียนขึ้นเพื่อเรียกใช้ข้อมูลในแฟ้มข้อมูลนั้นโดยเฉพาะ เช่น เมื่อต้องการรายชื่อพนักงานที่มีเงินเดือนมากกว่า 100,000 บาทต่อเดือน โปรแกรมเมอร์ต้องเขียนโปรแกรมเพื่ออ่านข้อมูลจากแฟ้มข้อมูลพนักงานและพิมพ์รายงานที่แสดงเฉพาะข้อมูลที่ตรงตามเงื่อนไขที่กำหนด กรณีที่มีการเปลี่ยนแปลงโครงสร้างของแฟ้มข้อมูลข้อมูลเช่น ให้มีดัชนี (index) ตามชื่อพนักงานแทนรหัสพนักงาน ส่งผลให้

รายงานที่แสดงรายชื่อพนักงานที่ได้รับเงินเดือนมากกว่า 100,000 บาทต่อเดือน ซึ่งแต่เดิมกำหนดให้เรียงตามรหัสพนักงานนั้นไม่สามารถพิมพ์ได้ ทำให้ต้องมีการแก้ไขโปรแกรมตามโครงสร้างดัชนี (index) ที่เปลี่ยนแปลงไป ลักษณะแบบนี้เรียกว่าข้อมูลและโปรแกรมไม่เป็นอิสระต่อกัน

สำหรับระบบฐานข้อมูลนั้นข้อมูลภายในฐานข้อมูลจะเป็นอิสระจากโปรแกรมที่เรียกใช้ (data independence) สามารถแก้ไขโครงสร้างทางกายภาพของข้อมูลได้ โดยไม่กระทบต่อโปรแกรมที่เรียกใช้ข้อมูลจากฐานข้อมูลเนื่องจากระบบฐานข้อมูลมีระบบจัดการฐานข้อมูลทำหน้าที่แปลงรูป (mapping) ให้เป็นไปตามรูปแบบที่ผู้ใช้ต้องการ เนื่องจากในระบบแฟ้มข้อมูลนั้นไม่มีความเป็นอิสระของข้อมูล ดังนั้นระบบฐานข้อมูลได้ถูกพัฒนาขึ้นมาเพื่อแก้ปัญหาด้านความเป็นอิสระของข้อมูล นั่นคือระบบฐานข้อมูลมีการทำงานไม่ขึ้นกับรูปแบบของฮาร์ดแวร์ที่นำมาใช้กับระบบฐานข้อมูลและไม่ขึ้นกับโครงสร้างทางกายภาพของข้อมูล และมีการใช้ภาษาสอบถามในการติดต่อกับข้อมูลภายในฐานข้อมูลแทนคำสั่งของภาษาคอมพิวเตอร์ในยุคที่ 3 ทำให้ผู้ใช้เรียกใช้ข้อมูลจากฐานข้อมูลโดยไม่จำเป็นต้องทราบรูปแบบการจัดเก็บข้อมูล ประเภทหรือขนาดของข้อมูลนั้นๆ

4. มีความปลอดภัยของข้อมูลสูง

ถ้าหากทุกคนสามารถเรียกดูและเปลี่ยนแปลงข้อมูลในฐานข้อมูลทั้งหมดได้ อาจก่อให้เกิดความเสียหายต่อข้อมูลได้และข้อมูลบางส่วนอาจเป็นข้อมูลที่ไม่อาจเปิดเผยได้หรือเป็นข้อมูลเฉพาะของผู้บริหาร หากไม่มีการจัดการด้านความปลอดภัยของข้อมูล ฐานข้อมูลก็จะไม่สามารถใช้เก็บข้อมูลบางส่วนได้ระบบฐานข้อมูลส่วนใหญ่จะมีการรักษาความปลอดภัยของข้อมูล ดังนี้

- มีรหัสผู้ใช้ (user) และรหัสผ่าน (password) ในการเข้าใช้งานฐานข้อมูลสำหรับผู้ใช้แต่ละคนระบบฐานข้อมูลมีระบบการสอบถามชื่อพร้อมรหัสผ่านของผู้เข้ามาใช้ระบบงานเพื่อให้ทำงานในส่วนที่เกี่ยวข้องเท่านั้น โดยป้องกันไม่ให้ผู้ที่ไม่ได้รับอนุญาตเข้ามาเห็นหรือแก้ไขข้อมูลในส่วนที่ต้องการปกป้องไว้

- ในระบบฐานข้อมูลสามารถสร้างและจัดการตารางข้อมูลทั้งหมดในฐานข้อมูล ทั้งการเพิ่มผู้ใช้ ระงับการใช้งานของผู้ใช้ อนุญาตให้ผู้ใช้สามารถเรียกดู เพิ่มเติม ลบและแก้ไขข้อมูล หรือบางส่วนของข้อมูลได้ในตารางที่ได้รับอนุญาตระบบฐานข้อมูลสามารถกำหนดสิทธิการมองเห็นและการใช้งานของผู้ใช้ต่างๆ ตามระดับสิทธิและอำนาจการใช้งานข้อมูลนั้นๆ

- ในระบบฐานข้อมูลสามารถใช้วิว (view) เพื่อประโยชน์ในการรักษาความปลอดภัยของข้อมูลได้เป็นอย่างดี โดยการสร้างวิวที่เสมือนเป็นตารางของผู้ใช้จริงๆ และข้อมูลที่ปรากฏในวิวจะเป็นข้อมูลที่เกี่ยวข้องกับงานของผู้ใช้เท่านั้น ซึ่งจะไม่กระทบกับข้อมูลจริงในฐานข้อมูล

- ระบบฐานข้อมูลจะไม่ยอมให้โปรแกรมใดๆ เข้าถึงข้อมูลในระดับกายภาพ (physical) โดยไม่ผ่าน ระบบการจัดการฐานข้อมูล และถ้าระบบเกิดความเสียหายขึ้นระบบจัดการฐานข้อมูลรับรองได้ว่าข้อมูลที่ยืนยันการทำงานสำเร็จ (commit) แล้วจะไม่สูญหาย และถ้ากลุ่มงานที่ยังไม่สำเร็จ (rollback) นั้นระบบจัดการฐานข้อมูลรับรองได้ว่าข้อมูลเดิมก่อนการทำงานของกลุ่มงานยังไม่สูญหาย

- มีการเข้ารหัสและถอดรหัส (encryption/decryption) เพื่อปกปิดข้อมูลแก่ผู้ที่ไม่เกี่ยวข้อง เช่น มีการเข้ารหัสข้อมูลรหัสผ่าน

5. ใช้ข้อมูลร่วมกันโดยมีการควบคุมจากศูนย์กลาง

มีการควบคุมการใช้ข้อมูลในฐานข้อมูลจากศูนย์กลาง ระบบฐานข้อมูลสามารถรองรับการทำงานของผู้ใช้หลายคนได้กล่าวคือระบบฐานข้อมูลจะต้องควบคุมลำดับการทำงานให้เป็นไปอย่างถูกต้อง เช่นขณะที่ผู้ใช้คนหนึ่งกำลังแก้ไขข้อมูลส่วนหนึ่งยังไม่เสร็จ ก็จะไม่อนุญาตให้ผู้ใช้คนอื่นเข้ามาเปลี่ยนแปลงแก้ไขข้อมูลนั้นได้ เนื่องจากข้อมูลที่เข้ามายังระบบฐานข้อมูลจะถูกนำเข้าโดยระบบงานระดับปฏิบัติการตามหน่วยงานย่อยขององค์กร ซึ่งในแต่ละหน่วยงานจะมีสิทธิในการจัดการข้อมูลไม่เท่ากัน ระบบฐานข้อมูลจะทำการจัดการว่าหน่วยงานใดใช้ระบบจัดการฐานข้อมูลในระดับใดบ้าง ใครเป็นผู้นำข้อมูลเข้า ใครมีสิทธิแก้ไขข้อมูล และใครมีสิทธิเพียงเรียกใช้ข้อมูลเพื่อที่จะให้สิทธิที่ถูกต้องบนตารางที่สมควรให้ใช้

2.3 ความรู้พื้นฐานเกี่ยวกับ Visual basic

เนื่องจากภาษา Visual Basic เป็นเครื่องมือสำหรับเขียนโปรแกรมที่ได้รับความนิยมทั่วโลก รวมทั้งในเมืองไทย และเป็นภาษาที่เข้าใจได้ไม่ยากนักซึ่งความเป็นมาของภาษา Basic ถูกสร้างขึ้นในปี 1963 โดย Hohn Keneny และ Thomas Kutz ที่มหาวิทยาลัย Dartmouth สร้างขึ้นเพื่อใช้ในการสอนการเขียนโปรแกรมโดยเน้นภาษาง่ายต่อการเข้าใจและการใช้งานโดยที่เราสามารถทำการสร้างแอพพลิเคชั่นที่สามารถใช้งานกับระบบ windows ได้ง่ายขึ้นไม่เหมือนในอดีตที่ต้องเขียนโปรแกรมที่ซับซ้อนโดยที่บริษัทไมโครซอฟท์ได้เล็งเห็นว่าถ้าสร้างแอพพลิเคชั่นยากก็เหมือนว่าฆ่า windows ทางอ้อมด้วย นั่นเองทำให้มีการสร้างการเขียนโปรแกรมแบบ Visual Programming กำเนิดขึ้นมาซึ่งรูปแบบนี้ก็คือ การเขียนโปรแกรมพร้อมกับการเห็นผลลัพธ์ที่เกิดขึ้น Visaul Basic เป็นหนึ่งใน Visual Programming ที่ไมโครซอฟท์สร้างขึ้นมา และด้วยความเรียบง่ายของภาษาและการเขียนโปรแกรมที่รวดเร็ว ทำให้ได้รับความนิยมสูงในปัจจุบัน[1]

2.3.1 จุดเด่นของ Visual Basic

สร้างแอพพลิเคชั่นที่รวดเร็ว Visual Basic ได้รับการวางตัวให้เป็นเครื่องมือที่ช่วยให้สร้างแอพพลิเคชั่นที่ง่ายและรวดเร็วเพื่อลดเวลาการสร้างแอพพลิเคชั่นที่สั้นที่สุด ซึ่งรูปแบบนี้เรียกว่า

Raid Application Development หรือ RAD ทั้งนี้เพราะขจัดงานที่ต้องทำซ้ำ ๆ ซาก ๆ ของโปรแกรมออกไป ขจัดสิ่งที่ไม่จำเป็นเกี่ยวกับการควบคุมฮาร์ดแวร์ออกไป การจัดการภายในของ Windows ออกไปเหลือเฉพาะโฟกัสของปัญหาของงานจริงๆ แล้วเขียนโปรแกรมจัดการกับปัญหานั้นๆ ส่วนที่เหลือให้ Visual Basic จัดการ การเขียนโปรแกรมง่ายต่อการเรียนรู้

ใน Visual Basic ถ้าลองดูแล้วภาษาที่ใช้นั่นง่ายเพราะอ่านแล้วใกล้เคียงกับภาษปกติจึงทำให้เราสามารถเรียนรู้ได้ง่ายกว่าภาษาอื่นๆรวมเครื่องมืออำนวยความสะดวกในการเขียนโปรแกรม

นอกจากง่ายต่อการเรียนรู้แล้ว Visual Basic ยังมีเครื่องมือที่ช่วยในการเขียนโปรแกรมที่ไม่ยุ่งยาก เพราะเครื่องมือนั่นทำให้เราไม่ต้องจำหลักไวยากรณ์ที่ยุ่งยาก ตรวจสอบอัตโนมัติว่าโปรแกรมถูกต้องตามหลักไวยากรณ์หรือไม่ มีการแยกส่วนของโปรแกรมอย่างเป็นระเบียบ

2.3.2 การเขียนโปรแกรมเพื่อสร้างแอพพลิเคชั่นจาก Visual Basic

จะมีวิธีต่างจากการเขียนโปรแกรมแบบอื่นที่เราเคยเรียนรู้มาเพราะโปรแกรมแบบ VisualBasic จะเขียนแบบ Event-Driven ซึ่งจริงๆ ก็คือการเขียนโปรแกรมแบบนี้ " ถ้าเหตุการณ์เกิดขึ้น เราจะดำเนินการอย่างไร " กล่าวคือถ้าเหตุการณ์นี้เกิดขึ้นเราจะจัดการกับเหตุการณ์นี้อย่างไร เป็นโปรแกรมที่รองรับกับเหตุการณ์เช่น เราใช้เมาส์กดปุ่ม Exit ให้เราสอบถามว่าแน่ใจหรือไม่ ถ้าแน่ใจก็จบการทำงาน เป็นต้น Visual Basic สนับสนุนการเขียนโปรแกรมแบบ Event-Driven โดยมีเครื่องมือต่างๆ ช่วยจัดการกับเหตุการณ์ต่างๆ ที่น่าจะเกิดขึ้น

ฐานข้อมูลคืออะไร (Database) คือการจัดเก็บข้อมูลที่สัมพันธ์กันอย่างเป็นระเบียบ ทำให้ง่ายต่อการใช้งาน และค้นหาข้อมูลซึ่งฐานข้อมูลที่รู้จักกันเป็นอย่างดีคือ ฐานข้อมูลเชิงสัมพันธ์(Relation Database) เป็นรูปแบบการจัดเก็บข้อมูลที่สัมพันธ์กัน โดยมองในลักษณะตารางต่างๆที่มีความสัมพันธ์กันการใช้งานของฐานข้อมูลต้องมีองค์ประกอบหลายอย่างซึ่งจะประกอบด้วย

2.3.3 แอพพลิเคชั่นฐานข้อมูลที่เหมาะสม (Database Application)

แอพพลิเคชั่นฐานข้อมูลเป็นแอพพลิเคชั่นที่สร้างไว้ให้ผู้ใช้งานสามารถติดต่อกับฐานข้อมูลได้อย่าสะดวกซึ่งมีรูปแบบการติดต่อกับฐานข้อมูลแบบเมนูหรือกราฟิกโดยไม่จำเป็นต้องมีความรู้เกี่ยวกับฐานข้อมูลเลยก็สามารถเรียกใช้งานได้

2.3.4 ระบบจัดการกับฐานข้อมูล (Database Management System หรือ DBMS)

เป็นซอฟต์แวร์ที่จัดการข้อมูลในฐานข้อมูลทั้งการจัดเก็บ,การแสดงผล,การค้นหา,การสำรองข้อมูล ฯลฯ โดยจะเป็นเครื่องมือในการทำงานของผู้บริหารฐานข้อมูล และเป็นตัวกลางที่เชื่อมผ่านระหว่างแอพพลิเคชั่นฐานข้อมูลที่สร้างขึ้นกับตัวข้อมูล ตัวอย่างเช่น Microsoft Access, Foxpro, SQL sever เป็นต้น

2.3.5 Database Management System (DBMS)

เป็นโปรแกรมที่ทำหน้าที่เป็นตัวกลางในการติดต่อระหว่างผู้ใช้กับฐานข้อมูลเพื่อจะจัดการและควบคุมความถูกต้องความซ้ำซ้อน และความสัมพันธ์ระหว่างข้อมูลต่างๆ ภายในฐานข้อมูล ซึ่ง

ต่างจากระบบแฟ้มข้อมูลที่หน้าที่เหล่านี้จะเป็นหน้าที่ของโปรแกรมเมอร์ ในการติดต่อกับข้อมูลในฐานข้อมูลไม่ว่าจะด้วยการใช้คำสั่งในกลุ่มคำสั่ง DML หรือ DDL หรือจะด้วยโปรแกรมต่าง ๆ ทุกคำสั่งที่ใช้กระทำกับข้อมูลจะถูกโปรแกรม DBMS แล้วจึงนำมาแปล (Compile) เป็นการกระทำ (Operation) ต่างๆ ภายใต้คำสั่งนั้นๆ เพื่อนำไปกระทำกับตัวข้อมูลในฐานข้อมูลต่อไปสำหรับส่วนการทำงานต่างๆ ภายในโปรแกรม DBMS ที่ทำหน้าที่ในการแปรคำสั่งไปเป็นการกระทำต่างๆที่จะกระทำกับตัวข้อมูลนั้น ประกอบด้วยส่วนการกระทำต่างๆ ดังนี้

- ทำหน้าที่แปลงคำสั่งที่ใช้ในการจัดการกับข้อมูลภายในฐานข้อมูลให้อยู่ในรูปแบบที่ฐานข้อมูลเข้าใจ

- ทำหน้าที่ในการนำคำสั่งต่าง ๆ ซึ่งได้รับการแปลแล้ว ไปส่งให้ฐานข้อมูลทำงาน เช่นการเรียกใช้ข้อมูล (Retrieve) การจัดเก็บข้อมูล (Update) การลบข้อมูล (Delete) การเพิ่มข้อมูล (Add) เป็นต้น

- ทำหน้าที่ป้องกันความเสียหายที่จะเกิดขึ้นกับข้อมูลภายในฐานข้อมูล โดยจะคอยตรวจสอบว่าคำสั่งใดที่สามารถทำงานได้และคำสั่งใดที่ไม่สามารถทำงานได้

- ทำหน้าที่รักษาความสัมพันธ์ของข้อมูลภายในฐานข้อมูลให้มีความถูกต้องอยู่เสมอ

- ทำหน้าที่เก็บรายละเอียดต่างๆ ที่เกี่ยวข้องกับข้อมูลภายในฐานข้อมูลใน Data Dictionary ซึ่งรายละเอียดเหล่านี้มักจะถูกเรียกว่า "ข้อมูลของฐานข้อมูล" (Metadata)

- ทำหน้าที่ควบคุมให้ฐานข้อมูลทำงานได้อย่างถูกต้องและมีประสิทธิภาพ

โปรแกรม DBMS นี้ได้ถูกพัฒนาขึ้นมาเพื่อแก้ปัญหาทางด้าน Data Independence ที่ไม่มีในระบบแฟ้มข้อมูลดังนั้นจึงมีความอิสระจากทั้งตัว Hardware และตัวข้อมูลภายในฐานข้อมูล กล่าวคือ โปรแกรม DBMS จะมีการทำงานที่ไม่ขึ้นกับรูปแบบ (Platform) ของตัว Hardware ที่นำมาใช้กับระบบฐานข้อมูลรวมทั้งมีรูปแบบในการอ้างถึงข้อมูลที่ไม่ขึ้นอยู่กับโครงสร้างทางกายภาพของข้อมูล ด้วยการใช้ Query Language ในการติดต่อกับข้อมูลในฐานข้อมูลแทนคำสั่งของภาษาคอมพิวเตอร์ในยุคที่ 3 ส่งผลให้ผู้ใช้สามารถเรียกใช้ข้อมูลจากฐานข้อมูลโดยไม่จำเป็นต้องทราบประเภทของฐานข้อมูลนั้น หรือสามารถกำหนดลำดับที่ของ Filed ในการแสดงผลได้โดยไม่ต้องคำนึงถึงลำดับที่จริงของ Filed นั้น

2.3.6 Data Dictionary และ File Manager

ทุกฐานข้อมูลจะต้องมีส่วนที่ใช้เก็บข้อมูลในลักษณะ Metadata (ข้อมูลที่ใช้อธิบายเรื่องราวของข้อมูลที่เราจัดเก็บมา) เช่น โครงสร้างของข้อมูล โครงสร้างของ Table โครงสร้างของ Index กฎที่ใช้ควบคุมความถูกต้องของข้อมูล (Integrity Rule) กฎที่ใช้ในการรักษาความปลอดภัยให้กับข้อมูล (Security Rule) เป็นต้น ข้อมูลเหล่านี้จัดเป็นข้อมูลที่มีความจำเป็นต่อโปรแกรม DBMS ในการตัดสินใจที่จะดำเนินการใด ๆ กับฐานข้อมูลเช่น ข้อมูลที่เกี่ยวข้องกับกฎที่ใช้ในการรักษาความปลอดภัยให้กับข้อมูล จะถูกนำมาใช้ในการพิจารณาแก่ผู้ใช้ในการใช้งานฐานข้อมูล เป็น

ต้น สำหรับส่วนที่ใช้ในการจัดเก็บข้อมูลในลักษณะของ Metadata นี้ได้แก่ Data Dictionary หรือ Catalogสำหรับ File Manager เป็นส่วนที่ทำหน้าที่บริหารและจัดการกับข้อมูลที่เก็บอยู่ในฐานข้อมูลในระดับกายภาพ

2.4 ความรู้ทั่วไปเกี่ยวกับ Relation Database

Relation มักจะถูกเรียกว่า "Table" เนื่องจาก Relation เป็นหน่วยที่ใช้จัดเก็บข้อมูลที่อยู่ในรูปแบบของตารางขนาด 2 มิติ ที่ประกอบด้วยแถว (Row) และสดมภ์ (Column) แถวของ Relation ได้แก่ ข้อมูล 1 รายการ ซึ่งเทียบเท่ากับ Record ในระบบแฟ้มข้อมูล ส่วนแต่ละสดมภ์ของ Relation ได้แก่ คุณลักษณะต่าง ๆ ของข้อมูลในแต่ละแถวซึ่งเทียบเท่ากับ Field ของ Record ในระบบแฟ้มข้อมูล สำหรับชื่อของแต่ละแถวของ Relation จะถูกเรียกว่า "Tuple" ส่วนชื่อของแต่ละสดมภ์ของ Relation จะถูกเรียกว่า "Attribute" [2]

Relation Database คือกลุ่มของข้อมูลที่สัมพันธ์กัน ซึ่งจัดเก็บอยู่ในรูปของตาราง แต่ละตารางจะประกอบไปด้วยกลุ่มของ Record โดยแต่ละ Record จะประกอบไปด้วยกลุ่มของ Field ดังนั้นจึงอาจกล่าวได้ว่าแต่ละตารางก็เปรียบเสมือนกับ Spread Sheet ที่มี Record แสดงอยู่ในรูปของแถว (row) และ Field แสดงอยู่ในรูปของสดมภ์ (column)

ในการเข้าถึงข้อมูลในแต่ละ Table อาจจะกำหนดขอบเขตในการเข้าถึงข้อมูลได้ เช่น ต้องการให้ Table "ข้อมูลพนักงาน"แสดงข้อมูลของพนักงานโดยเฉพาะผู้ที่มีเงินเดือนมากกว่า 10,000บาท ในขณะเดียวกันถ้าต้องการให้สะดวกต่อการดู ก็อาจสั่งให้ Table นั้นเรียงลำดับ (Sort) ข้อมูลเช่น เรียงลำดับข้อมูลตามเงินเดือน และถ้าต้องการให้เข้าถึงข้อมูลได้ด้วยความรวดเร็ว ก็อาจจะใช้ดัชนี(index)มาช่วยในการเข้าถึงข้อมูลได้ นอกจากนี้ความสามารถที่น่าสนใจอย่างหนึ่งของ Relation Database ได้แก่ การนำข้อมูลจากหลายๆ Table ที่มีความสัมพันธ์กันมาใช้งานร่วมกัน (Join) ได้ในการทำ Index แต่ละ Table จะต้องมี Field ที่ใช้เป็น Key ซึ่งแบ่งออกเป็น Primary Key และ Foreign Key โดย Field ที่ใช้ในการอ้างถึงข้อมูลใน Table ที่สัมพันธ์กันจะเรียกว่า"Foreign Key"

2.4.1 ข้อแตกต่างระหว่าง Relation กับแฟ้มข้อมูลในระบบแฟ้มข้อมูล มีดังนี้

- Relation จะเป็นส่วนที่จัดเก็บข้อมูลในระดับแนวความคิด ดังนั้น ผู้ใช้ในทุกระดับ ไม่ว่าจะเป็น นักวิเคราะห์ระบบโปรแกรมเมอร์ หรือผู้ใช้ทั่วไป จะมองข้อมูลที่จัดเก็บอยู่ใน Relation นี้ในรูปของตารางเช่นเดียวกัน ทำให้เวลานำข้อมูลจาก Relation ไปใช้งาน หรือจัดเก็บข้อมูลลงใน Relation ผู้ที่นำข้อมูลไปใช้หรือจัดเก็บ ไม่จำเป็นที่จะต้องทราบถึงโครงสร้างทางกายภาพของ Relation นั้น ซึ่งต่างจากแฟ้มข้อมูลในระบบแฟ้มข้อมูล ที่ผู้ใช้ในแต่ละระดับจะมีมุมมองต่อข้อมูลเดียวกันต่างกัน ดังที่กล่าวมาแล้วในบทที่ผ่านมา ดังนั้น Relation ใน Relational Model จึงมีความเป็นอิสระจากทั้งส่วนของ Software และ Hardware

- การจัดเก็บ Relation ในหน่วยความสำรองจะอยู่ในที่เดียวกัน ซึ่งต่างจากแฟ้มข้อมูลในระบบแฟ้มข้อมูลที่แต่ละแฟ้มข้อมูลจะจัดเก็บอยู่อย่างกระจัดกระจาย สำหรับ Relation ที่ข้อมูลมีความสัมพันธ์กัน และนำมาจัดเก็บรวมกันนี้ จะเป็นฐานข้อมูล 1 ฐานข้อมูล

2.4.2 คุณสมบัติของ Relation

คุณสมบัติ Tuple และ Attribute ของแต่ละ Relation จะประกอบด้วย

- เนื่องจาก Relation ใน Relational Model อยู่ในรูปแบบของเซตทางคณิตศาสตร์ ที่ภายในเซตจะต้องประกอบด้วยสมาชิกที่มีค่าไม่ซ้ำกัน ดังนั้น ภายใน Relation ใดๆ จึงต้องมี Attribute ใด Attribute หนึ่ง ที่ทำให้แต่ละ Tuple ใน Relation มีข้อมูลที่ไม่ซ้ำกัน

- ด้วยเหตุผลเช่นเดียวกับข้อที่ 1 ลำดับที่ของสมาชิกภายในเซตใด ๆ จะไม่มีผลต่อเซตนั้น ดังนั้น ภายใน Relation จึงไม่มีการกำหนดลำดับที่ให้กับแต่ละ Tuple ใน Relation กล่าวคือ จะไม่มีการกล่าวถึงคำว่า Tuple แรก หรือ Tuple สุดท้าย หรือ Tuple ลำดับที่ 5 หรือ Tuple ถัดไป หรือ Tuple ที่ผ่านมาใน Relation

- ภายใน Relation จะไม่มีกำหนดลำดับที่ให้กับแต่ละ Attribute เนื่องจากในการอ้างถึง Attribute ใน Relation จะใช้ชื่อของ Attribute นั้นในการอ้างถึง ดังนั้น จึงไม่มีการกล่าวถึงคำว่า Attribute แรก หรือ Attribute สุดท้ายหรือ Attribute ลำดับที่ 5 หรือ Attribute ที่ผ่านมา หรือ Attribute ถัดไปใน Relation เช่นเดียวกับ Tuple

- ค่าในทุก Attribute ของ Relation จะต้องมีคุณสมบัติ Atomicity ซึ่งเป็นคุณสมบัติที่กำหนดให้ค่าข้อมูลในแต่ละ Attribute ของ Relation จะต้องมีความหมายใดความหมายหนึ่งเพียงความหมายเดียว ไม่ใช่กลุ่มของสิ่งใด สิ่งหนึ่ง หรือกล่าวอีกนัยหนึ่ง ข้อมูลในแต่ละ Attribute ของ Relation จะต้องไม่ใช้ข้อมูลในลักษณะ Repeating Group เช่น กรณีที่พนักงานสามารถสังกัดฝ่ายได้มากกว่า 1 ฝ่าย ข้อมูลใน Attribute "DeptID" ของแต่ละ Tuple ของ Relation "EMPLOYEE" ซึ่งใช้เก็บรหัสของฝ่ายที่พนักงานแต่ละคนสังกัด จะไม่สามารถจัดเก็บทุกรหัสของฝ่ายที่พนักงานคนนั้นสังกัดภายใน Tuple

- ชื่อของแต่ละ Attribute ใน Relation เดียวกัน จะต้องมีชื่อที่ไม่ซ้ำกัน

- ค่าที่ปรากฏในแต่ละ Attribute ใน Relation เดียวกัน จะต้องใช้แทนข้อมูลที่มีความหมายเดียวกัน

2.4.3 ประเภทของ Relation

Relation สามารถแบ่งออกเป็นประเภทต่างๆ ได้ดังนี้

- Named Relation เป็น Relation ที่สร้างขึ้นด้วยคำสั่ง SQLซึ่งอาจเป็น Relation จริงในฐานข้อมูล หรือเป็นเพียง Relation ที่สร้างขึ้นด้วยคำสั่งของ Query Language

- Base Relation เป็น Named Relation ในส่วนที่เป็น Relation จริงในฐานข้อมูล ซึ่งใช้เก็บข้อมูลในหน่วยความจำสำรองดังนั้นจึงเป็น Relation จริงที่เกิดขึ้นจากการออกแบบฐานข้อมูล

- Derived Relation เป็น Named Relation ในส่วนของ Relation ซึ่งได้มาจากการใช้เงื่อนไขประกอบกับคำสั่งของ Query Language กับ Base Relation

- Expressible Relation เป็น Relation ที่ได้มาจากการกระทำกับ Named Relation ด้วยเงื่อนไขทางด้านความสัมพันธ์ของข้อมูลระหว่าง Named Relation ที่ต้องการ ดังนั้นเมื่อนำเอาทุก Expressible Relation มาประกอบกัน ผลลัพธ์ที่ได้ จึงได้แก่ ทุกๆ Base Relation และ Derived Relation ที่ Expressible Relation นั้นใช้สร้างขึ้น

- View เป็น Derived Relation ประเภทหนึ่ง แต่จะเป็น Relation เสมือน (Virtual Relation) ที่ถูกสร้างขึ้นไว้ในฐานข้อมูล

- Snapshot เป็น Derived Relation ประเภทหนึ่งเช่นเดียวกัน View แต่จะต่างกันที่ Snapshot เป็น Relation ในฐานข้อมูล ที่สามารถอ่านข้อมูลได้เพียงอย่างเดียว และสามารถกำหนดเวลาในการปรับปรุงค่าของข้อมูลใน Snapshot ได้เช่น ทุกวัน ทุกสัปดาห์ เป็นต้น

- Query Result เป็น Relation ชั่วคราว ซึ่งเกิดจากการใช้ประโยคคำสั่งของ Query Language กับ Relation ในฐานข้อมูล ข้อมูลใน Relation ประเภทนี้ จะเกิดขึ้นก็ต่อเมื่อมีการเรียกใช้เท่านั้น และหายไปเมื่อเลิกใช้งาน

- Immediate Result เป็น Relation ชั่วคราวที่เกิดขึ้นในขณะทำการประมวลผลประโยคคำสั่งของ Query Language ที่มีความซับซ้อน เช่น คำสั่งต่อไปนี้ในการประมวลผล จะเริ่มจากคำสั่งในวงเล็บก่อน ซึ่งได้แก่ คำสั่ง "S JOIN SP" จากคำสั่งนี้ จะก่อให้เกิด Relation ชั่วคราวที่ใช้เก็บข้อมูลที่ได้จากการรวมกันของ "S" และ "SP" ก่อนที่จะนำไปใช้ในขั้นตอนต่อไป

- Stored Relation เป็น Expressible Relation ที่สามารถจัดเก็บค่าของข้อมูลได้ดังนั้น ในบางครั้งจึงอาจกล่าวว่า Relation ประเภทนี้เป็น Base Relation

2.4.4 ประเภทของความสัมพันธ์ระหว่าง Table แบ่งออกเป็น 3 ประเภท คือ

ประเภทที่ 1 ความสัมพันธ์แบบ One-to-Many หมายถึง ข้อมูลหนึ่ง Record ใน Table หนึ่งจะความสัมพันธ์กับข้อมูลในอีก Table หนึ่งมากว่าหนึ่ง Record เช่น ใบเสร็จหนึ่งใบสามารถที่จะมีรายการสินค้าได้มากกว่าหนึ่งรายการ

ประเภทที่ 2 ความสัมพันธ์แบบ One-to-One หมายถึง ข้อมูลแต่ละ Record ใน Table หนึ่งจะมีความสัมพันธ์กับข้อมูลในอีก Table หนึ่งเพียง Record เดียว เช่น จังหวัดหนึ่งจังหวัดสามารถมีผู้ว่าราชการจังหวัดได้เพียงหนึ่งคนเท่านั้น

ประเภทที่ 3 ความสัมพันธ์แบบ Many-to-Many หมายถึง หลาย Record ใน Table หนึ่งจะมีความสัมพันธ์กับอีกหลายRecord ในอีก Table หนึ่งเช่น เจ้าของบัญชีเงินฝากหนึ่งคน สามารถเป็นเจ้าของบัญชีเงินฝากได้มากกว่าหนึ่งบัญชี และแต่ละบัญชีเงินฝากสามารถมีเจ้าของบัญชีได้มากกว่าหนึ่งคน

2.4.5 Domain

เป็นการนิยามขอบเขตของค่าที่เป็นไปได้กับข้อมูลในแต่ละ Attribute เพื่อป้องกันไม่ให้เกิดการป้อนข้อมูลที่เกินขอบเขตที่กำหนด เช่น

1. ค่าที่นิยมให้กับข้อมูลจะต้องมีค่าเป็น Scalar กล่าวคือ จะต้องเป็นค่าที่มีความหมายในหน่วยที่เล็กที่สุด ที่ไม่ปรากฏโครงสร้างที่สามารถแยกย่อยออกเป็นโครงสร้างย่อย ๆได้อีกสำหรับข้อมูลที่มีค่าเป็น Scalar นี้จะเรียกข้อมูลนั้นว่ามีคุณลักษณะของ Atomicity

2. ข้อมูลที่สามารถนำมากำหนด Domain ได้จะต้องเป็นข้อมูลที่เป็นอิสระจากข้อมูลอื่น

3. ข้อมูลที่สามารถนำมากำหนด Domain ได้จะต้องเป็นข้อมูลประเภทเดียวกัน

4. Domain ที่กำหนดให้กับ Attribute ที่จะต้องถูก Attribute อื่นอ้างถึง สามารถถ่ายทอด Domain ของตนให้กับ Attribute ในอีก Relation หนึ่งที่อ้างถึงไปด้วย

5. ค่าของ Domain ที่กำหนดให้กับข้อมูลไม่จำเป็นที่จะต้องปรากฏอยู่ในข้อมูลนั้นๆ

ประโยชน์ของการกำหนด Domain ให้กับข้อมูลนอกเหนือจากจะเป็นการกำหนดค่าที่เป็นไปได้ ที่ผู้ใช้สามารถกำหนดให้กับข้อมูลในส่วนนั้นๆ เพื่อป้องกันไม่ให้ผู้ใช้ป้อนข้อมูลเกินขอบเขตที่กำหนดไว้แล้ว ยังสามารถสร้างความเชื่อมั่นในการนำข้อมูลที่สัมพันธ์กันมาเปรียบเทียบกันได้อีกด้วย

ข้อมูลคือเนื้อหาที่เราใช้งานนั่นเองซึ่งจะจัดเก็บในหน่วยความจำของคาต้าเบสเซิร์ฟเวอร์ ซึ่งจะถูกเรียกมาใช้งานจากระบบจัดการฐานข้อมูล

ผู้บริหารข้อมูลเป็นคนที่ทำหน้าที่ดูแลข้อมูลในฐานข้อมูลผ่านระบบจัดการฐานข้อมูลซึ่งจะควบคุมการทำงานให้เป็นไปอย่างเรียบร้อย นอกจากนี้ยังทำหน้าที่กำหนดผู้ที่มีสิทธิ์ที่ใช้งานระบบฐานข้อมูล,กำหนดถึงความปลอดภัยของการใช้งานพร้อมดูแลคาต้าเบสเซิร์ฟเวอร์ให้เป็นปกติ Visaul Basic จะมองเห็นฐานข้อมูลหนึ่งๆเป็นตาราง (Table) หลายๆ ตารางในแต่ละตารางจะประกอบด้วยคอลัมน์ต่างๆ แยกข้อมูลแต่ละพวกออกจากกัน โดยตารางจะแบ่งข้อมูลแต่ละชุดเป็นแถว ๆ (Row)

2.5 การติดต่อฐานข้อมูลด้วย Visual Basic

ใน Visual Basic เราแบ่งการเชื่อมต่อออกได้เป็นประเภทใหญ่ๆ ได้แก่

การเข้าถึงข้อมูลด้วยData Control Visual Basic เป็นภาษาทางคอมพิวเตอร์ที่พัฒนาในลักษณะ Front-End ซึ่งผู้พัฒนาโปรแกรมสามารถกำหนดรูปแบบของจอภาพได้ง่ายต่อการใช้งาน เพื่อลดความซับซ้อนของ Database Management System (DBMS) Visual Basic จะอาศัย Control ชื่อ "Data" (ซึ่งมักจะเรียกว่า Data Control) ในการทำงานร่วมกับ Databaseโดยที่ 1 Data Control จะอ้างถึง 1 Table หรือมากกว่าในฐานข้อมูลข้อมูลจะถูกอ่านจากTableมาเก็บไว้ที่ Data Control จะเรียกว่า Record set [8]

2.5.1 ในการทำงานร่วมกับ Data Control จะต้องอาศัย Property ต่างๆดังนี้

1.Connect ใช้สำหรับกำหนดประเภทของฐานข้อมูลพวก dBase , Excel , Lotus ,Text File ทั่วๆ ไป

2.Database Name ใช้สำหรับกำหนด Path และชื่อของ Database

3.RecordsetType ใช้กำหนดประเภทของ Record set ซึ่งประกอบด้วย

3.1 Table เป็น Record set ที่กระทำกับ Table เดียว

3.2 Dyna setเป็น Record set ที่กระทำกับTable ตั้งแต่ 1 Tableขึ้นไป Record set ประเภทนี้สามารถแก้ไขข้อมูลได้ โดยข้อมูลที่แก้ไขจะถูกส่งผ่านไปยังTableโดยอัตโนมัติ

3.3 Snapshot เป็น Recordset ที่กระทำกับ Table ตั้งแต่ 1 Table ขึ้นไป แต่ Recordset ประเภทนี้ไม่สามารถแก้ไขข้อมูลได้ส่วนใหญ่ใช้ในการแสดงผลอย่างเดียว

4.RecordSource ใช้สำหรับกำหนดชื่อของ Table เมื่อคลิกใน Propertyนี้จะปรากฏเป็น ListBox ที่ประกอบด้วยชื่อของ Table ต่างๆ ในDatabase ที่กำหนดไว้ใน Property "DatabaseName"

5. BOFAction ใช้กำหนดการทำงานให้กับ Database เมื่อเกิดสถานะ "Begin of File" ขึ้น ซึ่งสามารถกำหนดได้ 2 ลักษณะดังนี้

5.1 "0-MoveFile" (ค่า Default) เพื่อเลื่อน Pointer ไปยัง Recrod แรก

5.2 "1-BOF"เพื่อกำหนดค่าของ Propertyเป็น True ซึ่งจะส่งผลให้ DataControlใช้ Method "Movelastไม่ได้

6.EOFAction ใช้ทำงานให้กับData Control เมื่อเกิดสถานะ "End of File"ซึ่งสามารถกำหนดได้ 3 ลักษณะ

6.1 "0-MoveLast"เพื่อเลื่อน Pointer ไปยัง Record สุดท้าย

6.2 "1-EOF"เพื่อกำหนดค่าของProperty "EOF"เป็นTrueซึ่งส่งผลให้Data Controlใช้คำสั่ง "MoveNext"ไม่ได้

6.3 "2-ADDNew" เพื่อเพิ่มRecordให้กับRecordset โดยอัตโนมัติ

7. ReadOnly เป็น Propertyที่มีลักษณะข้อมูลแบบตรรกะใช้สำหรับกำหนดให้DataControlอ่านค่าได้อย่างเดียวเมื่อกำหนดค่าเป็น True และสามารถแก้ไขได้เมื่อกำหนดค่าเป็น False

Bound Control ได้แก่ Control ต่างๆ ที่สามารถใช้งานร่วมกับ Recordset เพื่อใช้การแสดงผลและรับข้อมูลทางจอภาพ Bound Control มาตราฐานใน Visual Basic ได้แก่ TextBox,ListBox,CheckBox,Image,Label และPicture DBCombo และDBList DBComboทำงานคล้ายกับ ComboBox ส่วน DBList ที่มีลักษณะการทำงานคล้ายกับ ListBox ทั้ง 2 อย่างต่างกันตรงที่จะไม่สามารถเข้าถึงข้อมูลได้โดยตรง แต่เนื่องจากทั้ง 2 Control นี้ไม่ได้เป็น Bound Control มาตราฐานจึงต้องเพิ่มเข้าไปใน ToolBox ก่อน DBGrid เป็นการนำเสนอข้อมูลในลักษณะของ

spreadsheet และเป็นอีก Control หนึ่งซึ่งไม่ได้เป็น Bound ConTrol มาตราฐานจึงต้องเพิ่มไว้ใน ToolBoxก่อน

Data Access Object หรือ DAO เป็นเทคโนโลยีที่แอพพลิเคชั่นฐานข้อมูลทำการติดต่อกับฐานข้อมูลโดยใช้กลุ่มของออบเจ็กต์ที่เตรียมไว้ให้ในการติดต่อ ออบเจ็กต์ต่างๆ ใน DAO ไม่ใช่ออบเจ็กต์เดี่ยวๆ แต่เป็นกลุ่มของออบเจ็กต์ หรืออาจจะมีออบเจ็กต์เก็บไว้ภายในซึ่งเรียกว่าคอลเล็กชั่นซึ่ง DAO มีข้อดีดังนี้

- ง่ายต่อการตรวจสอบและกลั่นกรองข้อมูลก่อนที่จะบันทึกลงในฐานข้อมูล
- ลดความยุ่งยากจากการใช้ฐานข้อมูล เมื่อใช้งานร่วมกับผู้ใช้หลายๆ คน
- สนับสนุนการประมวลผลแบบ Transaction อันเป็นการเพิ่มความน่าเชื่อถือ และความมีเสถียรภาพของแอพพลิเคชั่น
- จัดการเกี่ยวกับการล็อคและข้อผิดพลาดง่ายกว่า Datacontrol

การติดต่อผ่านทาง ODBC โดยตรง (ODBC Direct) เป็นการติดต่อกับฐานข้อมูลแบบ 32 bit ที่สนับสนุนมาตรฐาน ODBC (Open DataBase Connectivity) ที่ JET Engine ไม่สามารถจัดการได้ เช่น ฐานข้อมูลของ Oracle , ฐานข้อมูลที่เกิดจาก Microsoft SQL Server เป็นต้น โดยจะใช้ Remote Data Object-RDO เป็นตัวเข้าถึงข้อมูลแทน ซึ่งจะเป็นการติดต่อระหว่างแอพพลิเคชั่นที่สร้างด้วย VB กับฐานข้อมูลโดยตรง โดยอาศัยมาตราฐาน ODBC ในการเชื่อมโยง ซึ่งก็คือ

- คอนโทรล Remote Data (Remote Data Control - RDC)
- อ๊อบเจ็กต์ Remote Data (Remote Data Object - RDO)

- OLE ย่อมาจาก "Object Linking and Embedding" จัดเป็นวิธีการสนับสนุนแนวความคิดทางด้าน Reusability เนื่องจากเราสามารถเรียกใช้ Object ที่สร้างขึ้นด้วยโปรแกรมอื่น ๆ ที่สนับสนุนเทคนิค OLE มาใช้งานภายใน Object ที่สร้างขึ้นมาเพื่อให้มีประสิทธิภาพมากขึ้น

OLE เป็นวิธีที่ใช้เชื่อมโยงข้อมูลของโปรแกรมที่ต่างกันที่อยู่ Windows เข้าด้วยกันซึ่งฐานข้อมูลแบบนี้อาจมีหลายโปรแกรมรวมกัน OLE จะติดต่อกับฐานข้อมูลด้วยออบเจ็กต์ ActiveX Data โดยใช้ Data Access Object เป็นตัวกลางการเชื่อมต่อข้อมูลในแนวความคิดของ OLE มีอยู่ด้วยกัน 2 ลักษณะด้วยกันคือ

- การเชื่อมข้อมูลแบบObject Embedding เป็นการสร้าง OLE Document ขึ้น โดยนำเอาเอกสารหรือข้อมูลที่สร้างขึ้นจากโปรแกรมอื่นมาเก็บไว้ในตัว OLE Document นั้นโดยตรง
- การเชื่อมโยงแบบ Object Linking เป็นการสร้าง OLE Document ขึ้นเช่นเดียวกัน แต่แตกต่างโดยการเชื่อมโยงข้อมูลแบบ Object Embedding กล่าวคือ เอกสารหรือข้อมูลที่สร้างขึ้นจากโปรแกรมอื่นปรากฏอยู่ใน OLE Document ที่ใช้ในการเชื่อมโยงข้อมูลแบบ Object Link นี้ไม่ปรากฏอยู่ในตัว OLE Document นั้นโดยตรงแต่จะเก็บเพียง Path ที่อ้างถึงเอกสารหรือข้อมูลนั้นแทน

ในการเชื่อมโยงแบบ OLE จะต้องประกอบด้วยโปรแกรมทำงาน 2 ส่วนด้วยกันดังนี้

- Container Application ได้แก่โปรแกรมที่จะเรียกใช้เอกสารหรือข้อมูลที่โปรแกรมอื่นทำงาน โดย OLE Document ที่สร้างขึ้น อาจเรียกใช้เอกสารหรือข้อมูลของโปรแกรมอื่นในรูปของ Object Embedding หรือ Object Linking

- Server Application ได้แก่ โปรแกรมที่ผู้สร้างเอกสารหรือข้อมูลที่จะถูกเรียกใช้โดย Container Application โปรแกรมที่เป็น Server Application นี้จะต้องสนับสนุนการทำงานในลักษณะลากแล้วปล่อยการสำเนาข้อมูลไปยัง Clipboard เพื่อที่จะนำเอาเอกสารหรือข้อมูลที่สร้างขึ้นด้วย Server Application ไปใช้งานในแบบ Object Embedding และObject Linking

Application server / Database server

การทำงานในลักษณะ Application server นี้จะซับซ้อนกว่าระบบ File server อีกระดับหนึ่ง โดยเป็นการแบ่งเอาภาระบางอย่างโปรแกรมประยุกต์หรือ Application จากเครื่องที่เป็นเวิร์กสเตชั่น หรือ Client เข้ามาทำที่ตัวเซิร์ฟเวอร์ ซึ่งงานที่จะแบ่งมานี้ก็จะต้องเป็นงานที่ซ้ำๆ กันในระหว่างหลายเครื่องที่เป็นเวิร์กสเตชั่นด้วยกันจึงจะเหมาะสม ตัวอย่างที่เราพบกันบ่อยๆ ก็คือ database server หรือ SQL server ซึ่งย้ายหน้าที่การค้นหาข้อมูลจากฐานข้อมูลหรือ database มาไว้ที่เซิร์ฟเวอร์เองซึ่งขอยกตัวอย่างในรายละเอียดดังนี้

ตามปกติถ้าเป็นการทำงานแบบ File server นั้นเครื่องทีเป็นเวิร์กสเตชั่นต้องการ Record ใด Record หนึ่งที่มีเงื่อนไขตรงตามที่กำหนด ก็ต้องร้องขอให้ File server ส่งข้อมูลทุก Record คือทั้งไฟล์หรือทั้ง database มาให้โปรแกรมเวิร์กสเตชั่นเปรียบเทียบทีละ Record ไล่ไปเรื่อยๆ จนกว่าจะพบ ซึ่งข้อเสียของวิธีนี้คือ จะต้องมีการส่งข้อมูลจำนวนมากผ่านสาย LAN คือข้อมูลทั้งไฟล์นอกจากนี่หากมีหลายๆ เวิร์กสเตชั่นทำงานในแบบเดียวกัน File server ก็จะต้องอ่านข้อมูลทั้งไฟล์นั้นซ้ำแล้วซ้ำอีก เพื่อส่งให้กับแต่ละเครื่องโดยไม่สามารถทำการ optimize หรือหาวิธีการทำงานที่ง่ายกว่านั้นได้ แต่ถ้าเปลี่ยนเป็น database server แล้ววิธีการทำงานจะเปลี่ยนไป ทางฝั่งเวิร์กสเตชั่นเพียงส่งชื่อไฟล์และเงื่อนไขที่ต้องการค้นหามาให้ จากนั้น database server จะอ่านข้อมูลทั้งไฟล์ขึ้นมาทำการค้นหาและเปรียบเทียบจนกว่าจะได้ Record ที่ต้องการให้เองแล้วจึงค่อยส่งเฉพาะนั้นกลับไป ซึ่งจะไม่เกิดการรับส่งข้อมูลผ่าน LAN มากเกินความจำเป็น และหากมีหลายๆ เวิร์กสเตชั่นทำงานในลักษณะเดียวกันหรือคล้ายกัน database server ก็จะสามารถพิจารณาหาทางยุบรวมหรือปรับวิธีการทำงานในตัวให้มีประสิทธิภาพดีขึ้นได้เองด้วย

การส่งเงื่อนไขให้ข้อมูลมาที่ database นี้ก็ต้องมีวิธีมาตรฐาน ซึ่งในปัจจุบันนิยมใช้เป็นภาษา SQL (Structure Query Language) ซึ่งเป็นภาษาที่ใช้ค้นข้อมูลจากบานข้อมูลแบบสัมพันธ์หรือ Relational database

จึงอาจเรียก database server ได้อีกอย่างหนึ่งว่า SQL server ซึ่งก็เป็นตัวอย่างหนึ่งของ Application server เท่านั้น ยังมีอีกความเป็นไปได้ที่จะแบ่งภาระของ Application จากเวิร์กสเตชั่นมาที่ server อีกหลายแบบ ทั้งนี้แล้วแต่ลักษณะของงานเป็นสำคัญ

2.5.2 Method ที่ใช้ในการจัดการข้อมูล

ในการทำงานกับ Databadse โดยทั่วไปจะเกี่ยวข้องกับการเพิ่ม แก้ไข ลบและค้นหาข้อมูล ซึ่งแต่ละการทำงานจะอาศัย Method ที่แตกต่างกันไป ดังนั้นในส่วนนี้ เราจะมาเรียนรู้การใช้ Method ต่างๆในการจัดการกับข้อมูลใน Database

2.5.2.1 การเพิ่มข้อมูลเข้าไปยัง Table ต่าง ๆ ใน Database จะอาศัย Method "AddNew" เพื่อสร้าง Record ว่างเพิ่มเข้าไปใน Recordset

Recordset.AddNew

โดยที่ Recordset หมายถึง ชื่อ Object ที่เป็นเจ้าของ Recordset ในกรณีที่ใช้ Data Control ให้กำหนดในรูป Datacontrol.Recorset โดย datacontrol ได้แก่ ชื่อของ Datacontrol นั้น

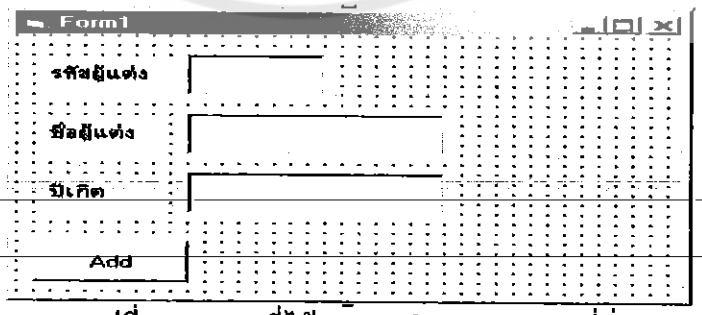
ตัวอย่าง การใช้งาน Method "AddNew"

1. วาด Control "CommandButton" ต่างๆ ลงบน From
2. กำหนด Property ของ CommandButton ดังตาราง

ตารางที่ 2.1 กำหนดค่า

Object	Property	ค่าที่กำหนด
CommandButton	Name	AddCmnd
	Caption	&Add

เมื่อเสร็จแล้วจะได้จอภาพดังรูป



รูปที่ 2.1 From ที่ได้จากการทำตาม Control ที่สั่ง

3. พิมพ์คำสั่งใน Event"click"ของปุ่ม Add ดังนี้

```
Private Sub AddCmnd_Click()
        Author .Recordset.AddNew
```

End Sub

4. Run จากนั้นให้คลิกที่ปุ่ม Add จะสังเกตเห็นว่า ข้อมูลใน TextBox ต่างๆจะหายไป

2.5.2.2 การแก้ไขข้อมูล Method ที่ใช้ในการบันทึกข้อมูลที่มีการแก้ไข ได้แก่ Method "Update" โดยจะบันทึกข้อมูลที่ปรากฏอยู่ใน Object ที่ Bound Control กลับลงไปยัง Recordset รูปแบบของคำสั่งเป็นดังนี้

Recordset.Update

โดยที่ Recordset หมายถึง ชื่อ Object ที่เป็นเจ้าของ Recordset ในกรณีที่ใช้ DataControl ให้กำหนดในรูปDatacontrol.Recordset โดย Datacontrol ได้แก่ ชื่อของ Data Control นั้น

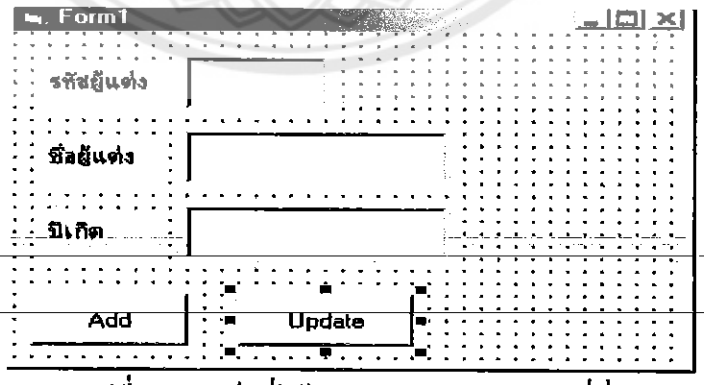
ตัวอย่าง การใช้ Method "Update"

1. จากตัวอย่างที่แล้ววาด Control"CommandButton"เพิ่มลงไปบน Form อีก 1 Object
2. กำหนดคุณสมบัติ ของ Object ดังตาราง

ตารางที่ 2.2 การกำหนดค่า

Object	Property	ค่าที่กำหนด
CommandButton	Name	UpdateCmnd
	Caption	&Update

เมื่อเสร็จ แล้วจะได้จอภาพดังรูป



รูปที่ 2.2 ฟอร์มที่ได้จากการทำตาม Control ที่สั่ง

3. พิมพ์คำสั่งลงใน Event "Click"ของปุ่ม Update ดังนี้

Private Sub UpdateCmnd_Click()

```
Author.Recordset.Update
End Sub
```

4. Run จากนั้นคลิกที่ปุ่ม Add ข้อมูลในแต่ละ Text Box จะถูก Clear ไป ให้ทดลองป้อนข้อมูลในแต่ละ TextBox

2.5.2.3 การลบข้อมูล Method ที่ใช้สำหรับลบข้อมูล Record ปัจจุบันออกจาก Recordset ได้แก่ Method "Delete"

```
recordset.Delete
```

โดยที่ recordset หมายถึง ชื่อ Object ที่เป็นเจ้าของ Recordset ในกรณีที่ใช้ Datacontrolให้กำหนดในรูปdatacontrol.Recordset โดย datacontrol ได้แก่ ชื่อของ Datacontrolนั้น
ในการลบ Record ทุกครั้งหลังจากลบควรที่จะมีการขยับ Pointer เนื่องจาก Record ปัจจุบันได้ถูกลบไปเพื่อป้องกันการสับสนว่า Pointer ชี้ไปที่Record ใดสำหรับ Method ที่ใช้ในการเลื่อน Pointer ได้แก่ Method "MoveFirst","MoveNext","MoveLast",และ"MovePrevious"

```
Recordset.{MoveFirst|MoveLast|MoveNext|MovePrevious}
```

โดยที่ Recordset หมายถึงชื่อ Object ที่เป็นเจ้าของ recordset ในกรณีที่ใช้ DataControlให้กำหนดในรูปdatacontrol.Recorset โดย datacontrol ได้แก่ ชื่อของ datacontrol นั้นทั้ง 4 Method นี้ จะมีรูปแบบของคำสั่งเช่นเดียวกันสำหรับหน้าที่ของแต่ละ Method มีดังนี้

1. MoveFirst ใช้สั่งให้ DataControl เลื่อน Pointer ไปยัง record แรกใน recordset
2. MoveLast ใช้สั่งให้ DataControl เลื่อน Pointer ไปยัง record สุดท้ายใน recordset
3. MoveNext ใช้สั่งให้ DataControl เลื่อน Pointer ไปยัง record ถัดไปใน recordset
4. MovePrevious ใช้สั่งให้ DataControl เลื่อน Pointer ไปยัง record ก่อนหน้าใน recordset

ในกรณีที่ Pointer ของ Record ปัจจุบันชี้อยู่ที่ record แรกของ Recordset เมื่อใช้ Method "MovePrevious" ค่าของ Property "BOF" ของ Datacontrol จะถูกกำหนดให้มีค่าเป็น True และก็เช่นเดียวกัน กรณีที่ Pointer ของ Record ปัจจุบันชี้อยู่ที่record สุดท้ายของrecordset เมื่อใช้ Method"MoveNext"ค่าของ Property "EOF"จะถูกกำหนดให้มีค่าเป็น True เช่นกัน และถ้าไม่ใช่ทั้ง 2 กรณีแล้วค่าของ Property "BOF"และ"EOF"จะถูกกำหนดให้มีค่าเป็น False เสมอ

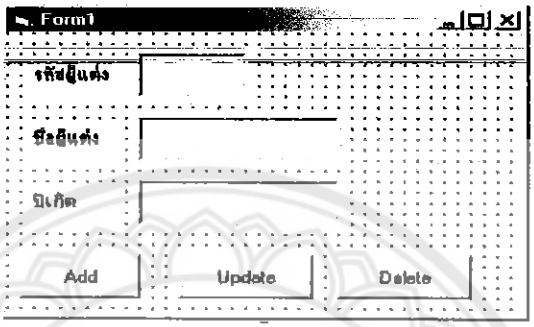
ตัวอย่าง การใช้ Method "Delete"

1. จากตัวอย่างที่แล้ววาด CommandButtom เพิ่มเติมลงบน Form จากนั้นกำหนด Property ให้กับ Object ดังกล่าวดังตาราง

ตารางที่ 2.3 กำหนดค่า

Object	Property	ค่าที่กำหนด
Command Button	Name	DeleteCmnd
	Caption	&Delete

เมื่อเสร็จแล้วจะได้จอภาพดังรูป



รูปที่ 2.3 From ที่ได้จากการทำตาม Control ที่สั่ง

2. พิมพ์คำสั่งให้กับปุ่ม Delete ดังนี้

```
Private Sub DeleteCmnd_Click()
        Author.Recordset.Delete
        Author.Recordset.MoveLast
End Sub
```

3. Run แล้วคลิกที่ปุ่มหมายเลข 4 ใน DataControl เพื่อขยับไปยัง Record สุดท้ายใน Recordset

4.คลิกที่ปุ่ม Delete ข้อมูลบนจอภาพจะเปลี่ยนเป็นข้อมูล Record ก่อนหน้า เนื่องจาก Record สุดท้ายได้ถูกลบออกไปแล้ว

2.5.2.4 การค้นหาข้อมูล Method ที่ใช้ในการค้นหาข้อมูลใน Recordset จะประกอบไปด้วย

1. FindFirst ใช้สำหรับค้นหา Record โดยทิศทางในการหาจะเริ่มจาก Record แรกไปยัง Record สุดท้ายใน Recordset จนกระทั่งพบ record ซึ่งตรงกับเงื่อนไขการค้นหา

2. FindLast ใช้สำหรับค้นหา Record โดยทิศทางในการหาจะเริ่มจาก Record สุดท้ายไปยัง Record แรกใน Recordset จนกระทั่งพบ record ซึ่งตรงกับเงื่อนไขการค้นหา

3. FindNextใช้สำหรับค้นหา Record โดยทิศทางในการหาจะเริ่มจาก Record ปัจจุบันไปยัง Record สุดท้ายใน Recordset จนกระทั่งพบ record ซึ่งตรงกับเงื่อนไขการค้นหา

4. FindPrevious ใช้สำหรับค้นหา Record โดยทิศทางในการหาจะเริ่มจาก Record ปัจจุบันไปยัง Record แรกใน Recordset จนกระทั่งพบ record ซึ่งตรงกับเงื่อนไขการค้นหา

Recordset.{FindFirst|FindLast|FindNext|FindPrevious}criteria

โดยที่ recordset หมายถึง ชื่อ Object ที่เป็นเจ้าของ Recordset ในกรณีที่ใช้ Datacontrol ให้กำหนดในรูปdatacontrol.Recordset โดย Datacontrol ได้แก่ ชื่อของ Datacontrol นั้น Criteria หมายถึง ประโยคเงื่อนไขที่ใช้ในการค้นหา

2.5.3 DBCombo และ DBList

DBCombo เป็น control ที่มีลักษณะการทำงานคล้ายกับcomboBox ส่วน DBList เป็น control ที่มีลักษณะการทำงานคล้ายกับ ListBox แต่ต่างกันที่2control นี้ จะสามารถเข้าถึงข้อมูลใน Recordset ได้โดยตรง แต่เนื่องจากทั้ง2control นี้ไม่ได้เป็น BoundControlมาตรฐาน ดังนั้นจึงต้องเพิ่มเข้ามาไว้ใน Toolbox ก่อน

2.5.4 DBGrid

DBgrid เป็น control ที่ใช้การนำเสนอข้อมูลในลักษณะของ Spreadsheet และเป็นอีก Controlหนึ่งที่ไม่ใช่Bound Control มาตราฐาน ดังนั้น จึงต้องเพิ่มเข้ามาไว้ใน Toolbox โดยเลือก "Microsoft Data Bound Grid Control (SP3)"ในจอภาพComponents สำหรับ Icon ที่ใช้แทน DBGrid ใน Toolbox

2.6 การเขียนโปรแกรมกับ DataEnvironment

วิธีการเลื่อนตำแหน่ง Record

ใน DataEnvironment ในการควบคุมออบเจ็กต์ Recordset ที่เกิดจาก DataEnvironment คุณสามารถทำได้ 2 วิธี คือ

1. สร้างชุดคำสั่ง SQL ให้กับออบเจ็กต์ Command
2. แก้ไขคุณสมบัติ หรือเมธอดของออบเจ็กต์ Command

ทั้ง 2 วิธี ให้ผลเช่นเดียวกัน กล่าวคือ เป็นการสั่งออบเจ็กต์ Command ให้เข้าถึงข้อมูลในฐานข้อมูลนั่นเอง อาจกล่าวได้อีกนัยหนึ่งว่า เมื่อต้องการจัดการข้อมูลในออบเจ็กต์ Recordset ที่เกิดจาก DataEnvironment จึงต้องอาศัยออบเจ็กต์ Command ที่อยู่ภายใต้การเชื่อมต่อเดียวกันนั่นเอง

ตัวอย่างการเลื่อน Record ด้วย DataEnvironment

โค้ดตัวอย่างต่อไปนี้ เป็นการเลื่อนตำแหน่ง Record ผ่านทาง DataEnvironment

โค้ดตัวอย่างการสร้างปุ่มเลื่อน Record ด้วย DataEnvironment

```
Private Sub cmdMoveFirst_Click ()
    With DataEnvironment1.rsCommand1          'เรียกใช้ออบเจ็กต์ Recordset ของ Command1
    .MoveFirst
```

```
    End With
End Sub

Private Sub cmdMoveLast_Click ()
    With DataEnvironment1.rsCommand1
        .MoveLast
    End With
End Sub

Private Sub cmdMovePrev_Click ()
    With DataEnvironment1.rsCommand1
        .MovePrevious
    If .BOF Then
        .MoveFirst
    End If
    End With
End Sub

Private Sub cmdMoveNext_Click ()
With DataEnvironment1.rsCommand1
    .MoveNext
    If .EOF Then
        .MoveLast
    End If
End With
```

อธิบายการทำงานของโปรเจ็กต์

ในโพรซีเจอร์ cmdMoveFirst_Click() เราจะใช้เมธอด MoveFirst ของออบเจ็กต์ Recordset สำหรับเลื่อนไปยัง Recordแรก โดยปกติแล้ว VBIDE จะกำหนดชื่อให้กับออบเจ็กต์ Recordset ที่เกิดจาก DataEnvironment โดยใช้ rs นำหน้าชื่อออบเจ็กต์ Recordset เสมอ ดังนั้น เมื่อคุณต้องการใช้งานออบเจ็กต์ Recordset จึงต้องอ้างอิงถึงชื่อ rsRecordset1 นั่นเอง

```
    With DataEnvironment1.rsCommand1
```

```
        .MoveFirst
    End With
```

ส่วนในโพรซีเจอร์ cmdMoveLast_Click() จะใช้เมธอด MoveLast ของออบเจ็กต์ Recordset สำหรับเลื่อนไปยัง Recordสุดท้ายเช่นกัน

```
    With DataEnvironment1.rsCommand1
        .MoveLast
    End With
```

1. สำหรับในโพรซีเจอร์ cmdMovePrev_Click() และ cmdMoveNext_Click() ก็จะมีการตรวจสอบตำแหน่งของเคอร์เซอร์ด้วย จะเหมือนกับกรณีการคอนโทรล ADO Data

```
Private Sub cmdMovePrev_Click ()
With DataEnvironment1.rsCommand1
        .MovePrevious
        If .BOF Then
            .MoveFirst
        End If
End With
End Sub

Private Sub cmdMoveNext_Click ()
With DataEnvironment1.rsCommand1
        .MoveNext
        If .EOF Then
            .MoveLast
End If
End With
```

2.6.1 การเพิ่ม, ลบ, แก้ไข, ยกเลิกการเพิ่ม และค้นหาข้อมูลโดยอาศัย DataEnvironment

การจัดการข้อมูลในฐานข้อมูลผ่านทาง DataEnvironment โดยการใช้งานกลุ่มคอนโทรลด้านฐานข้อมูลร่วมกับ DataEnvironment นั้นมีขั้นตอนเดิมมาจากแบบที่ใช้ร่วมกับคอนโทรล ADO Data อยู่อย่างเดียวคือ จะต้องระบุชื่ออออบเจ็กต์ Command ให้กับคุณสมบัติ DataMember ด้วย

เนื่องจากว่าใน DataEnvironment สามารถมีได้หลายการเชื่อมต่อ (ออบเจ็กต์ Connection) และในแต่ละการเชื่อมต่อสามารถมีออบเจ็กต์ Command ได้หลายตัว ดังนั้น จึงต้องระบุชื่อออบเจ็กต์ Command เพื่อกำหนดคอนโทรลด้านฐานข้อมูลตัวนั้น ๆ สามารถเข้าถึงฐานข้อมูลในฐานข้อมูล โดยอาศัยออบเจ็กต์ Command ได้อย่างถูกต้องนั่นเอง

ตัวอย่างการจัดการข้อมูลผ่าน DataEnvironment

สำหรับโปรเจ็กต์แรกนั้นเป็นการจัดการข้อมูลในฐานข้อมูลผ่านทาง DataEnvirnoment ซึ่งมีขั้นตอนการสร้างและทดสอบดังต่อไปนี้

1. เชื่อมต่อเข้ากับฐานข้อมูลที่ต้องการ ในกรณีนี้คือ ฐานข้อมูล student.mdb
2. เพิ่มออบเจ็กต์ Command1 ให้กับ Connection1 โดยคลิกขวาที่ Connection1 แล้วเลือกคำสั่ง Add Command จากนั้นคลิกขวาที่ Command1 แล้วเลือกคำสั่ง Properties... เพื่อติดต่อกับตารางที่ต้องการใช้งาน กรณีนี้คือ ตาราง student
3. สิ่งสำคัญคุณต้องกำหนดให้ออบเจ็กต์ Recordset ที่เกิดจากออบเจ็กต์ Command อยู่ในสถานะที่สามารถแก้ไขข้อมูลได้ คุณกำหนดได้ที่แท็บ Advanced ในไดอะล็อกบ็อกซ์ Command1 Properties ที่ช่อง Lock Type ให้กำหนดเป็น 3 – Optimistic
4. จากนั้นลาก Command1 ด้วยเมาส์ปุ่มซ้าย ไปวางไว้ที่ฟอร์ม
5. เพิ่มเติมคอนโทรลอื่น ๆ
6. แก้ไขคุณสมบัติของคอนโทรลที่เพิ่มเข้าไปแต่ละตัวดังนี้

ตารางที่ 2.4 กำหนดค่า

คอนโทรล	คุณสมบัติ	ค่าที่กำหนด
Command1	Name	cmdAdd
	Caption	&Add
Command2	Name	cmdUpdate
	Caption	&Update
Command3	Name	cmdDelete
	Caption	&Delete
Command4	Name	cmdEnd

ตารางที่ 2.4 กำหนดค่า

คอนโทรล	คุณสมบัติ	ค่าที่กำหนด
	Caption	&End
Command5	Name	cmdOK
	Caption	&OK
Command6	Name	cmdCancel
	Caption	&Cancel
Command7	Name	cmdSearch
	Caption	&Search
Label1	Caption	
Frame1	Caption	Recordที่:
Text1	Text	

7. จากนั้นให้คุณเขียนโค้ด เพื่อจัดการข้อมูลในฐานข้อมูลโดยอาศัย DataEnvironment ดังนี้

```
Private Sub Form_Load()
        cmdOK.Visible = False                    ' ซ่อนปุ่ม OK
        cmdCancel.Visble = False                 ' ซ่อนปุ่ม Cancel
With DataEnvironment1.rsCommand1     'แสดง Recordปัจจุบัน
        Label1.Caption = "Recordที่:" & .AbsolutePostion & "/" & .RecordCount
        End With
End Sub

Private Sub cmdAdd_Click()
Dim tmp As Long
With DataEnvironment1.rsCommand1
        .MoveLast
        tmp = .Fields("StudentID").Value + 1     'เพิ่มรหัสนักศึกษาอีก  1  แล้วเก็บไว้ที่ตัวแปร
tmp.
        .AddNew                                  'เพิ่ม Recordว่าง
        txtStudentID.Text = tmp
        txtStudentID.Enabled = False
        Label1.Caption = "Recordที่:" & .AbsolutePosition & "/" & .RecordCount
```

```
        cmdAdd.Enabled = False          'กำหนดให้ปุ่ม Add อยู่ในสถานะใช้ไม่ได้ชั่วคราว
        cmdUpdate.Enabled = False       'กำหนดให้ปุ่ม Update อยู่ในสถานะใช้ไม่ได้ชั่วคราว
        cmdDelete.Enabled = False       'กำหนดให้ปุ่ม Delete อยู่ในสถานะใช้ไม่ได้ชั่วคราว
        cmdEnd.Enabled = False          'กำหนดให้ปุ่ม End อยู่ในสถานะใช้ไม่ได้ชั่วคราว
        'กำหนดให้ปุ่มเลื่อนตำแหน่ง Recordใช้ไม่ได้ชั่วคราว
        cmdMoveFirst.Enabled = False
        cmdMovePrev.Enabled = False
        cmdMoveNext.Enabled = False
        cmdMoveLast.Enabled = False

        cmdOK.Visible = True            'แสดงปุ่ม OK
        cmdCancel.Visible = True        'แสดงปุ่ม Cancel
        cmdSearch.Enabled = False       'กำหนดให้ปุ่ม Search ใช้ไม่ได้ชั่วคราว
        Text1.Enabled = False           'กำหนดให้คอนโทรล Text1 ใช้ไม่ได้ชั่วคราว
End Sub

Private Sub cmdUpdate_Click()
With DataEnvironment1.rsCommand1
        .Field("StudentID").Value = txtStudentID.Tex
        .Field("FirstName").Value = txtFirstName.Text
        .Field("LastName").Value = txtLastName.Text
        .Field("MajorID").Value = txtMajorID.Text
        .Update                         'เพิ่มข้อมูลไปที่ Recordว่าง
        .txtStudentID.Enabled = True
End With
        cmdAdd.Enabled = True           'กำหนดให้ปุ่ม Add อยู่ในสถานะใช้ได้ตามปกติ
        cmdUpdate.Enabled = True        'กำหนดให้ปุ่ม Update อยู่ในสถานะใช้ได้ตามปกติ
        cmdDelete.Enabled = True        'กำหนดให้ปุ่ม Delete อยู่ในสถานะใช้ได้ตามปกติ
        cmdEnd.Enabled = True           'กำหนดให้ปุ่ม End อยู่ในสถานะใช้ได้ตามปกติ
'กำหนดให้ปุ่มเลื่อนตำแหน่ง Recordใช้ได้ตามปกติ
        cmdMoveFirst.Enabled = True
        cmdMovePrev.Enabled = True
```

```
        cmdMoveNext.Enabled = True
        cmdMoveLast.Enabled = True

        cmdOK.Visible = False                   'ซ่อนปุ่ม OK
        cmdCancel.Visible = False               'ซ่อนปุ่ม Cancel

        Text1.Enabled = True                    'กำหนดให้คอนโทรล Text1 อยู่ในสถานะปกติ
        cmdSearch.Enabled = True                'กำหนดให้ปุ่ม Search ใช้ได้ตามปกติ
End Sub

Private Sub cmdDelete_Click()                   'ก่อนลบ Record ต้องการคำยืนยัน
If MsgBox("ต้องการลบ Record นี้ ใช่หรือไม่?", vbYesNo, "ยืนยันการลบ Record") = vbYes Then
        With DataEnvironment1.rsCommand1
        .Delete                                 'ลบ Record
        .MoveNext                               'เลื่อนไป Record ถัดไป
        If .EOF Then                            'ตรวจสอบว่าใช่จุดสิ้นสุดหรือไม่
                .MoveLast                       'ถ้าใช่ ให้เลื่อนเคอร์เซอร์ไปที่ Record สุดท้าย
        End If
        End With
End If
End Sub

Private Sub cmdEnd_Click()
'ต้องการคำยืนยันก่อนจบการทำงาน
If MsgBox("ถ้าต้องการออกจากโปรแกรม ใช่หรือไม่?", vbYesNo, "คำยืนยัน") = vbYes Then
        End                                     'จบการทำงาน
End If
End Sub
Private Sub cmdOK_Click()
'ตรวจสอบก่อนว่า ใส่ข้อมูลครบทุกช่องหรือไม่
If (txtStudentID.Text = "") or (txtFirstName.Text = "") or (txtLastName.Text = "") _
Or (txtMajorID.Text = "") Then              'ถ้าไม่ครบ
```

```
        MsgBox "คุณยังใส่ข้อมูลไม่ครบ", vbOKOnly, "ข้อผิดพลาด"
Else
        cmdUpdate.Enabled = True        'กำหนดให้ปุ่ม Update อยู่ในสถานะปกติ
End If
End Sub
Private Sub cmdCancel_Click()
With DataEnvironment1.rsCommand1
        .CancelUpdate                   'ยกเลิกการเพิ่ม Record ว่าง
End With
        cmdAdd.Enabled = True           'กำหนดให้ปุ่ม Add อยู่ในสถานะใช้ได้ตามปกติ
        cmdUpdate.Enabled = True        'กำหนดให้ปุ่ม Update อยู่ในสถานะใช้ได้ตามปกติ
        cmdDelete.Enabled = True        'กำหนดให้ปุ่ม Delete อยู่ในสถานะใช้ได้ตามปกติ
        cmdEnd.Enabled = True           'กำหนดให้ปุ่ม End อยู่ในสถานะใช้ได้ตามปกติ

        cmdOK.Visible = False           'ซ่อนปุ่ม OK
        cmdCancel.Visible = False       'ซ่อนปุ่ม Cancel

        Text1.Enabled = True            'กำหนดให้คอนโทรล Text1 อยู่ในสถานะปกติ
        cmdSearch.Enabled = True        'กำหนดให้ปุ่ม Search ใช้ได้ตามปกติ

'กำหนดให้ปุ่มเลื่อนตำแหน่ง Record ใช้ได้ตามปกติ
        cmdMoveFirst.Enabled = True
        cmdMovePrev.Enabled = True
        cmdMoveNext.Enabled = True
        cmdMoveLast.Enabled = True
End Sub

Private Sub cmdSearch_Click()
Dim userCriteria As String              'ตัวแปรสำหรับเก็บข้อความที่ผู้ใช้ป้อนเข้ามา
'สร้างเงื่อนไข โดยการใช้ฟิลด์ StudentID เปรียบเทียบกับข้อความที่ป้อนเข้ามา
userCriteria = "StudentID like' " &Text1.Text&" ' "
If Text1.Text = "" Then                 'ถ้าผู้ใช้ไม่ได้ป้อนข้อความ
```

```
        MsgBox"ถ้ายังไม่ได้ใส่รหัสนักศึกษา!    กรุณาใส่รหัสนักศึกษาด้วย",vbYes,"ค้นหาข้อมูล
ผิดพลาด"
ElseIf IsNumeric(Text1.Text) Then
        With DataEnvironment1.rsCommand1
                .MoveFirst                          'เลื่อนไปที่ Record แรก
                .Find userCriteria, , adSearchForward 'ค้นหาข้อมูลตามเงื่อนไข
                Label1.Caption = "Recordที่: "& .AbsolutePosition & "/" & .RecordCount
        End With
End If
        If DataEnviroment1.rsCommand1
                MsgBox "ไม่พบข้อมูล", vbYes, "ผลการค้นหา"
        Label1.Caption = "Recordที่: "
        End If
End Sub

Private Sub cmdMoveFirst_Click()
With DataEnvironment1.rsCommand1
        .MoveFirst                          'เลื่อนไปที่ Record แรก
Label1.Caption = "Recordที่: " & .AbsolutePosition & "/" & .RecordCount
End With
End Sub
Private Sub cmdMovePrev_Click()
With DataEnvironment1.rsCommand1
        .MovePrevious                       'เลื่อนไป Record ก่อนหน้า
        If .BOF Then                        'ตรวจสอบว่าใช่จุดเริ่มต้นหรือไม่
                .MoveFirst                  'ถ้าใช่ ให้เลื่อนเคอร์เซอร์ไปที่ Record แรก
        End If
Label1.Caption = "Recordที่: " & .AbsolutePosition & "/" & .RecordCount
End With
End Sub

Private Sub cmdMoveNext_Click ()
```

```
With DataEnvironment1.rsCommand1
        .MoveNext                          'เลื่อนไปยัง Record ถัดไป
        If .EOF Then                       'ตรวจสอบว่า ใช่จุดสิ้นสุดหรือไม่
                .MoveLast                  'ถ้าใช่ ให้เลื่อนเคอร์เซอร์ไปที่ Record สุดท้าย
        End If
Label1.Caption = "Record ที่: " & .AbsolutePosition & "/" & .RecordCount
End With
End Sub
Private Sub cmdMoveLast_Click()
With DataEnvironment1.rsCommand1
        .MoveLast                          'เลื่อนไปที่ Record สุดท้าย
Label1.Caption = "Recordที่: " & .AbsolutePosition & "/" & .RecordCount
End With
End Sub

Private Sub Text1_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then   ' ถ้าผู้ใช้กดปุ่ม Enter แล้วให้โพรซีเจอร์ cmdSearch_Click() ทำงาน
                Call cmdSearch_Click
        End If
End Sub
```

2.7 ออบเจ็กต์ Command

เมื่อต้องการเข้าถึงฐานข้อมูลในฐานข้อมูล ต้องสร้างการเชื่อมต่อด้วยออบเจ็กต์ Connection หลังจากเชื่อมต่อกับฐานข้อมูลแล้ว เราจะใช้ออบเจ็กต์ Command ในการเข้าถึงข้อมูลในฐานข้อมูล อาจกล่าวได้ว่า ออบเจ็กต์ Command เป็นตัวกลางที่ทำหน้าที่รับคำสั่ง เช่น ชุดคำสั่ง SQL จากออบเจ็กต์ Connection หรือจากการใช้เมธอด execute ของตัวมันเองเข้าไปยังฐานข้อมูล เพื่อที่จะได้ข้อมูลกลับมาในรูปของออบเจ็กต์ Recordset ภายในการเชื่อมต่อของออบเจ็กต์ Connection เดียวกัน

แต่เนื่องจากว่าออบเจ็กต์ต่าง ๆ ที่อยู่ในโมเดล ADO มีลักษณะที่สำคัญอย่างหนึ่งก็คือ ออบเจ็กต์แต่ละตัวมีอิสระต่อกัน ดังนั้นคุณจึงสามารถใช้ออบเจ็กต์ Command เชื่อต่อกับฐานข้อมูลโดยที่ไม่ต้องใช้ออบเจ็กต์ Connection ได้เช่นกัน

วิธีการเชื่อมต่อกับฐานข้อมูลด้วยออบเจ็กต์ Command

สามารถใช้ออบเจ็กต์ Command เชื่อมต่อกับฐานข้อมูล และส่งคำสั่งไปยังฐานข้อมูลได้เช่นกัน โดยที่การเชื่อมต่อเข้ากับฐานข้อมูล อาจใช้การเชื่อมต่อเดิมที่มาจากออบเจ็กต์ Connection หรือสร้างการเชื่อมต่อใหม่ด้วยออบเจ็กต์ Command เองก็ได้ โดยการกำหนดข้อความเชื่อมต่อให้กับคุณสมบัติ Active Connection ก็ได้

โค้ดตัวอย่างการเชื่อมต่อเข้ากับฐานข้อมูลด้วยออบเจ็กต์ Command

```
Private Sub From_Load()
Dim Cmd As ADODB.Command
Set Cmd = New ADODB.Command
With Cmd
        .CommandText = "Select * From student"
        .CommandType = adCmdText
                .ActiveConnection = "Provider = Microsoft.Jet.OLEDB.4.0;" & _
                        "Data Source = D:\student.mdb;Persist Security Info = False"
End With
End Sub
```

จากนั้นให้เขียนโค้ด เพื่อค้นหาข้อมูลด้วยออบเจ็กต์ Command ดังนี้

โค้ดโปรเจ็กต์ การค้นหาข้อมูลด้วยออบเจ็กต์ Command

```
ประกาศตัวแปรร่วม
Private Conn As ADODB.Connection
Private rs As ADODB.Recordset
Private Cmd As ADODB.Command

Private Sub Form_Load()
'กำหนดให้ตัวแปร Conn สร้างการเชื่อมต่อใหม่
Set Conn = New ADODB.Connection

'เชื่อมต่อกับฐานข้อมูลตัวออบเจ็กต์
With Conn
        .ConnectionString = "Provider = Microsoft.Jet.OLEDB.4.0;" & _
                "Data Source = D:\student.mdb;Persist Security info = False"
        .CommandTimeout = 25                    'กำหนดระยะเวลาในการเชื่อมต่อ
```

```
        .CursorLocation = adUseClient          'กำหนดให้ใช้เคอร์เซอร์ของฝั่งไคลเอนต์
        .Open                                  'เปิดฐานข้อมูล
    End With
    End Sub

    Private Sub cmdSearch_Click()
    'สร้างตัวแปรออบเจ็กต์ Command และ Recordset ขึ้นมาใหม่
    Set Cmd = New ADODB.Command
    Set rs = New ADODB.Recordset

    If Text3.ext <> "" Then                    'ถ้าผู้ใช้ป้อนข้อความ
        With Cmd                               'เลือก Record ที่ตรงกับรหัสนักศึกษา
            .CommandText="Select * Form Student Where StudentID=" & Val(Text 3.Text)
            .CommandType = adCmdText
            .ActiveConnection = Conn
        End With
    'กำหนดให้ตัวแปร rs เก็บ Record ที่ได้
    Set rs = Cmd.Execute
        If Not rs.EOF Then                     'ถ้ามีรหัสนักศึกษาตรงกับที่ป้อนเข้ามา
            'แสดงข้อมูลแต่ละฟิลด์
            Text1.Text = rs("FirstName").Value
            Text2.Text = rs("LastName").Value
        Else                                   'ถ้าไม่ตรงกัน
            MsgBox "ไม่มีรหัสนักศึกษาตามที่คุณต้องการ", vbOKOnly, "ผลการค้นหา"
            Text3.SetFocus
        End If
    Else                                       'กรณีไม่ได้ป้อนข้อความ
        MsgBox     "คุณยังไม่ได้ใส่รหัสนักศึกษา", vbOKOnly, "ข้อผิดพลาด"
    End If
    End Sub
    Private Sub Text3_KeyPass (KeyAscii As Integer)
    If KeyAscii = 13 Then
```

```
        Call cmdSearch_Click
End If
End Sub

Private Sub cmdEnd_Click ()
If Msgbox ("คุณต้องการออกจากโปรแกรมนี้ ใช่หรือไม่?", vbYesNo, "คำยืนยัน")= vbYes Then
        End
End If
End Sub
```

ทดสอบการทำงานของโปรเจ็กต์ ได้ผลดังนี้

อธิบายการทำงาน

เริ่มต้นประกาศตัวแปรในระดับฟอร์ม เนื่องจากต้องการใช้งานตัวแปรออบเจ็กต์เหล่านี้ในหลายโพรซิเจอร์ โดยที่ตัวแปรแต่ละตัว จะแทนออบเจ็กต์ Connection, Recordset และออบเจ็กต์ Command

```
Private Conn As ADODB.Connection
Private rs As ADODB.Recordset
Private Cmd As ADODB.Command
```

ในโพรซีเจอร์ Form_Load () ตัวอย่างนี้ ผู้เขียนใช้ออบเจ็กต์ Connection เชื่อมต่อกับฐานข้อมูล พร้อมทั้งกำหนดระยะเวลาในการเชื่อมต่อ และใช้งานเคอร์เซอร์ทางฝั่งไคลเอนต์ด้วย

```
Set Conn = New ADODB.Connection
With Conn
        .ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;" & _
                "Data Source = D:\student.mdb;Persist Security Info = False"
        .CommandTimeout = 25
        .CursorLocation = adUseClient
        .Open
End With
```

1. ส่วนในโพรซีเจอร์ cmdSearch_Click () เริ่มต้น ต้องสร้างออบเจ็กต์ Command และออบเจ็กต์ Recordset ขึ้นมาอย่างละตัว เพราะว่า
   - ออบเจ็กต์ Command ใช้สั่งคำสั่งไปยังฐานข้อมูล
   - ออบเจ็กต์ Recordset ใช้เก็บ Record ที่ได้มาจากฐานข้อมูล

```
Set Cmd = New ADODB.Command
```

```
Set rs = New ADODB.Recordset
```

ต่อมาตรวจสอบว่า ผู้ใช้ป้อนข้อความเข้ามาหรือไม่ ถ้าป้อนข้อความเข้ามา ก็จะสร้างชุดคำสั่ง SQL ให้กับคุณสมบัติ CommandText ของออบเจ็กต์ Command โดยมีเงื่อนไขว่า ต้องการ Record ที่มีค่าฟิลด์ StudentID ตรงกับที่ผู้ใช้ป้อนเข้ามาที่คอนโทรล Text3

2. จากนั้น คุณต้องกำหนดชนิดของออบเจ็กต์ Command ด้วย กรณีนี้ เป็นออบเจ็กต์ Command ที่เก็บชุดคำสั่ง SQL จึงกำหนดเป็นค่าคงที่ adCmdText ให้คุณสมบัติ CommandType และกำหนดให้ใช้การเชื่อมต่อเดิมของออบเจ็กต์ Connection ด้วยการกำหนดคุณสมบัติ ActiveConnection = Conn

```
If Text3.Text <> "" Then
    With Cmd
        .CommandText="Select*From student Where StudentID="& Val(Text3.Text)
        .CommandType = adCmdText
        .ActiveConnection = Conn
    End With
```

ต่อมากำหนดให้ rs ซึ่งเป็นตัวแปรออบเจ็กต์ Recordset เก็บ Record ที่ได้จากการรันชุดคำสั่ง SQL ของออบเจ็กต์ Command ด้วยเมธอด execute

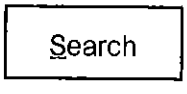
```
Set rs = Cmd.Execute
```

จากนั้นตรวจสอบว่า ถ้ามี Record อยู่ในออบเจ็กต์ Recordset ให้แสดงออกมาด้วยคุณสมบัติ Value ของออบเจ็กต์ Recordset ที่คอนโทรล TextBox ทั้ง 2 ตัว

```
If Not rs.EOF Then
        Text1.Text = rs("FirstName").Value
        Text2.Text = rs("LastName").Value
```

ในกรณีที่ไม่พบ Record ที่ตรงกับเงื่อนไข ให้แสดงข้อความแจ้งกับผู้ใช้และกำหนดให้คอนโทรล TextBox3 อยู่ในสถานะถูกโฟกัส

```
Else
        MsgBox "ไม่มีรหัสนักศึกษาตามที่คุณต้องการ", vbOKOnly, "ผลการค้นหา"
        Text3.SetFocus
End If
```

3. ในกรณีที่ผู้ใช้ไม่ได้ป้อนข้อความ แล้วคลิกที่ปุ่ม 

ก็จะมีข้อความเตือนเช่นกัน

```
Else
        MsgBox "คุณยังไม่ได้ใส่รหัสนักศึกษา", vbOKOnly, "ข้อผิดพลาด"
```

```
End If
```

วิธีสร้างออบเจ็กต์ Recordset ด้วยออบเจ็กต์ Command

เป็นวิธีการส่งชุดคำสั่ง SQL โดยการใช้เมธอด execute ของออบเจ็กต์ Command ไปยังตารางที่ต้องการเพื่อที่จะให้ได้ออบเจ็กต์ Recordset กลับมาก่อน แล้วแสดงข้อมูลที่อยู่ในออบเจ็กต์ Recordset ดังกล่าวออกมาในตัวอย่างซึ่งมีขั้นตอนการสร้างดังนี้

1. ออกแบบฟอร์ม
2. ให้แก้สมบัติของคอนโทรลต่าง ๆ ดังต่อไปนี้

ตารางที่ 2.5 แก้สมบัติของคอนโทรลต่าง ๆ ดังต่อไปนี้

คอนโทรล	คุณสมบัติ	ค่าที่กำหนด
Command	Name	cmdRun
	Caption	&Run

3. จากนั้นเขียนโค้ดเพื่อสร้างออบเจ็กต์ Recordset ดังนี้

โค้ดโปรเจ็กต์ที่ 7 – 3 การสร้างออบเจ็กต์ Recordset ด้วยออบเจ็กต์ Command

```
Private Sub cmdRun_Click ()
Dim Conn As New ADODB.Connection
Dim Cmd As New ADODB.Command
Dim rs As New ADODB.Recordset

With Conn
        .ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;" & _
                * Data Source = D:'student.mdb;Persist Security Info = False"
        .Open
End With
With Cmd
        .ActiveConnection = Conn
        .CommandText = "Select * From Student"
        .CommandType = adCmdText
End With
        Set rs = Cmd.Execute
        List1.Clear
```

```
While Not rs.EOF
List1.AddItem rs.Field("FirstName").Value
Rs.MoveNext
Wend
Set rs = Nothing
Set Cmd = Nothing
Set Conn = Nothing
End Sub
```

4. ทดสอบการทำงานของโปรเจ็กต์ ได้ผลดังนี้

อธิบายการทำงานของโปรเจ็กต์ จากรูปที่ เมื่อคุณคลิกที่ปุ่ม Run แล้ว ส่งผลให้โพรซีเจอร์ cmdRun_Click () ทำงาน เริ่มต้นสร้างตัวแปรสำหรับแทนออบเจ็กต์ทั้ง 3 ตัว

```
Dim Conn As New ADODB.Connection
Dim Cmd As New ADODB.Command
Dim rs As New ADODB.Recordset
```

1. เชื่อมต่อกับฐานข้อมูลด้วยออบเจ็กต์ Connection

```
With Conn
        .ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; & _
                * Data Source = D:\strdent.mdb;Persist Security Info = False"
        .Open
End With
```

2. จากนั้นสร้างชุดคำสั่ง SQL ให้กับออบเจ็กต์ Command

```
With Cmd
        .ActiveConnection = Conn
        .CommandText = "Select * From Student"
        .CommandType = adCmdText
End With
```

3. ที่สำคัญอยู่ตรงที่แสดงข้อมูลจากฟิลด์ชื่อ FirstName ที่อยู่ในออบเจ็ก์ Recordset เข้าไปในคอนโทรล ListBox

```
Set rs = Cmd.Execute
        List1.Clear
        While Not rs.EOF
                List1.AddItem rs.Fields("FirstName").Value
                rs.MoveNext
        Wend
```

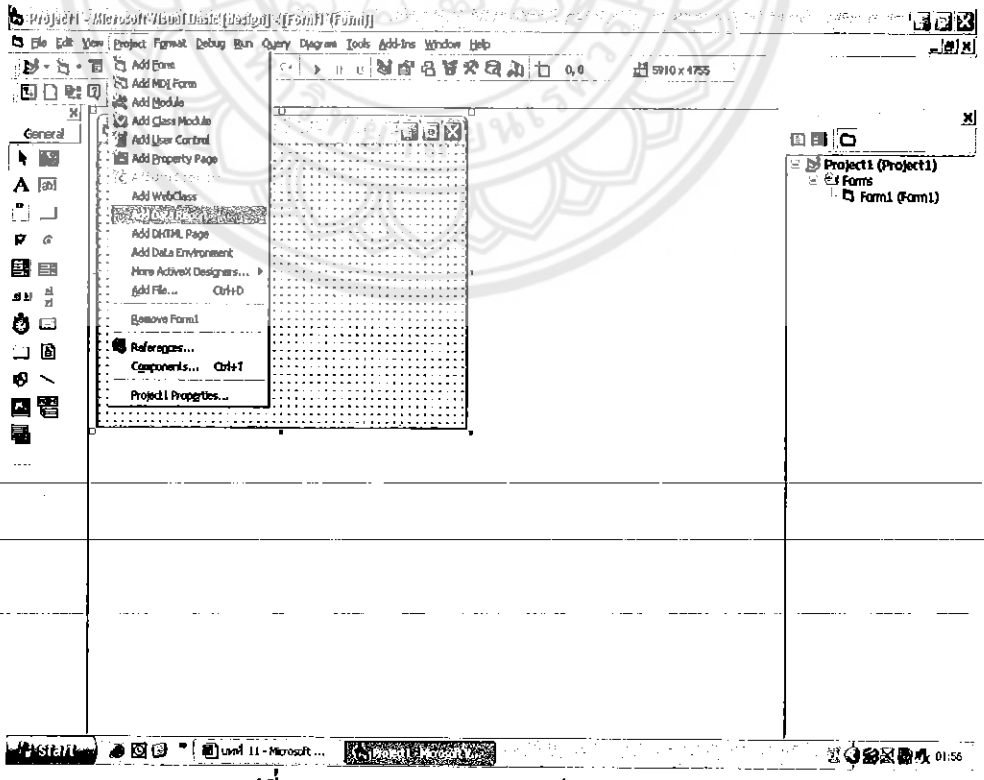
2.8 การสร้างรายงานด้วย DataReport

การทำรายงานถือได้ว่าเป็นรูปแบบของการแสดงข้อมูลอีกชนิดหนึ่ง ที่เป็นการสรุปข้อมูลที่ได้จากฐานข้อมูลใน Visaul Basic 6.0 มีเรื่องที่เรียกว่า DataReport ใช้สำหรับทำงานโดยเฉพาะ ซึ่งจะใช้ควบคู่กับ DataEnvironment

DataReport สนับสนุนการใช้งานแบบ Drag & Drop เช่นเดียว DataEnvironment ทำให้คุณสามารถใช้งาน DataReport ในการสร้างรายงานได้อย่างรวดเร็ว

2.8.1 วิธีเพิ่ม DataReport เข้ามาใน VBIDE ชนิด Standard EXE

กรณีที่ใช้VBIDE ชนิด DataReport จะถูกสร้างขึ้นมาอัตโนมัติแต่กรณีที่คุณใช้VBIDE ชนิด Standard EXE จะเพิ่มเข้ามาเอง โดยเลือกคำสั่ง Project>Add Data Report ดังรูปที่

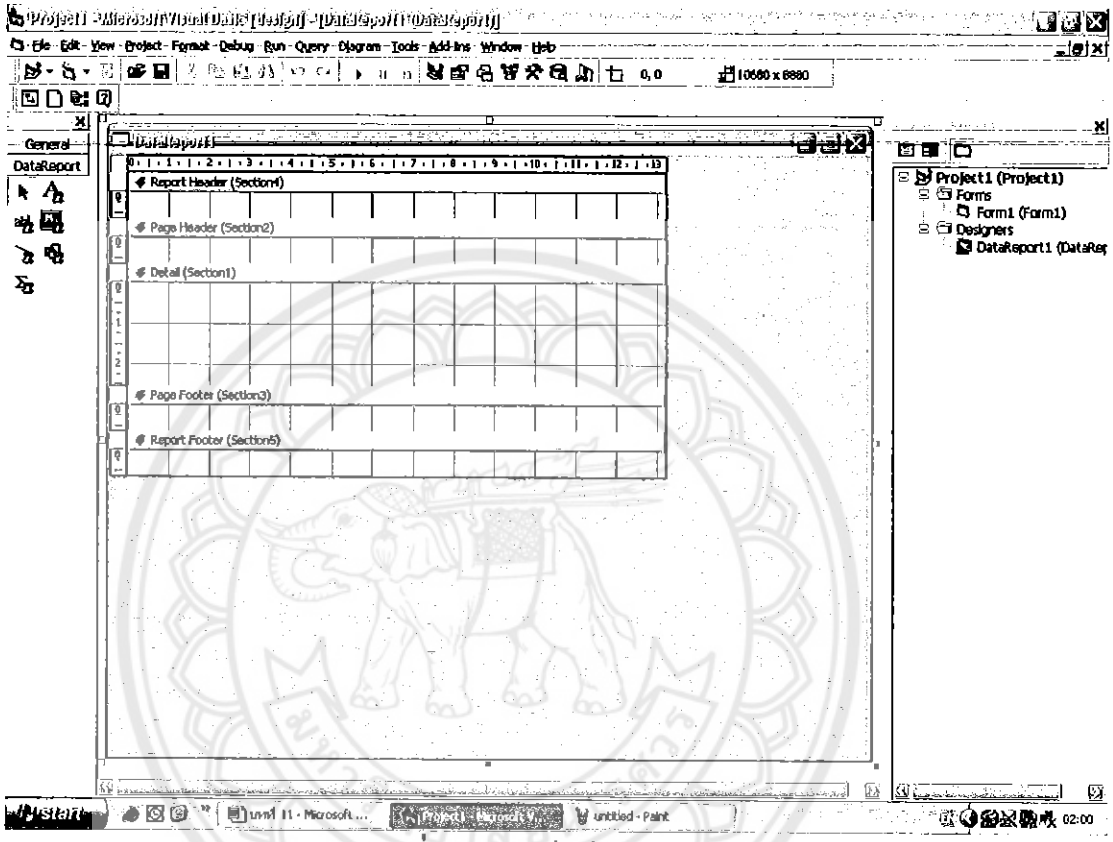


รูปที่ 2.4 แทรก DataReport เข้ามาในโปรแกรม

2.8.2 ส่วนประกอบของ DataReport

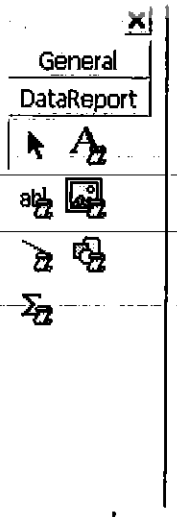
DataReport Designer ประกอบไปด้วย 2 ส่วนหลัก คือ

-แบบฟอร์มรายงาน – เรียกอีกอย่างว่า ออบเจ็กต์ DataReport(DataReport Object) มีหน้าที่คล้ายกับแบบฟอร์มของ VBIDE กล่าวคือ ใช้สำหรับทำแบบฟอร์มรายงาน



รูปที่ 2.5 ออบเจ็กต์ DataReport

กลุ่มคอนโทรลที่ใช้ในการทำรายงาน เป็นกลุ่มคอนโทรลที่อยู่ในแถบเครื่องมือ DataReport



รูปที่ 2.6 กลุ่มคอนโทรลที่ใช้สำหรับทำรายงาน

สำหรับออบเจ็กต์ DataReport นั้นแบ่งได้เป็น 5 ส่วนได้แก่ คือ

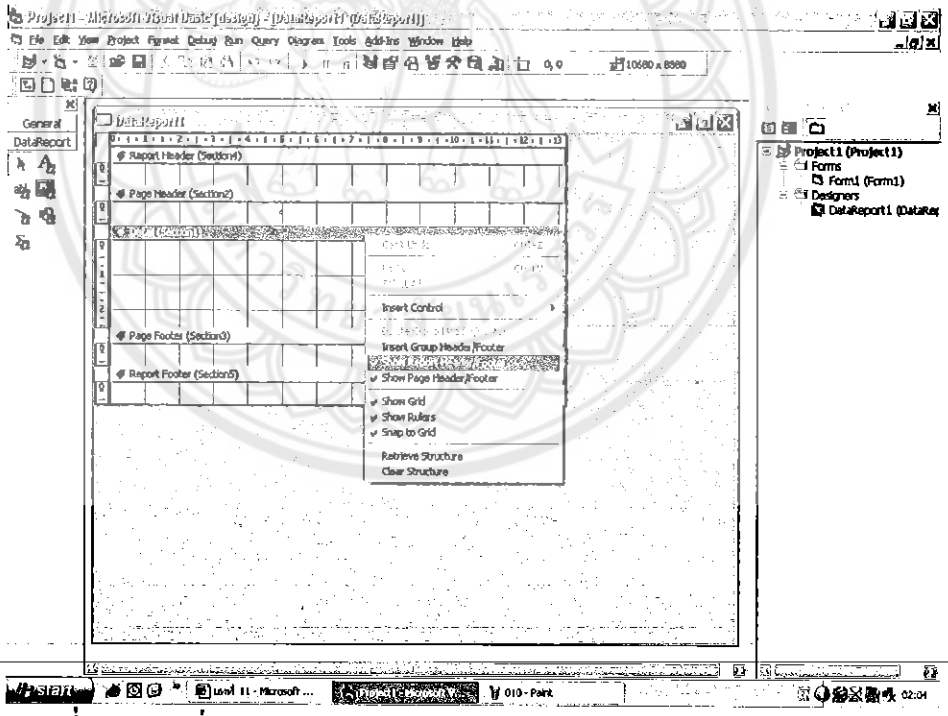
Report Header - เป็นข้อความที่ปรากฏอยู่บนสุดของแบบฟอร์มรายงาน จะแสดงที่หน้าแรกของรายงานเพียงหน้าเดียวเท่านั้น

Page Header - เป็นข้อความที่คุณต้องการให้ปรากฏในรายงานแต่ละหน้า เช่น ชื่อหนังสือ , ชื่อบท หรือ หมายเลขหน้าก็ได้

Detail - เป็นส่วนที่ใช้ในการแสดงข้อมูลจากฐานข้อมูล หรือข้อความถือเป็นส่วนที่สำคัญที่สุด DataReport

Page Footer - มีลักษณะเช่นด้วยกับ Page Header แต่อยู่ส่วนล่างของแบบฟอร์มรายงานในหน้าสุดท้ายเพียงหน้าเดียวเท่านั้น มักใช้ในการแสดงข้อมูลที่เป็นผลรวมของข้อมูลจากส่วน Detail เช่น ยอดขายทั้ง , ยอดคงเหลือสินขายทั้งหมดเป็นต้น

ในกรณี VBDIE ชนิด DataProject ออบเจ็กต์ DataReport จะแสดงออกมาเพียง 3 Section คุณสามารถเพิ่มเข้ามาได้โดยการคลิกขวาที่บริเวณ DataReport เลือกคำสั่ง Show Report Header / Footer ดังรูป



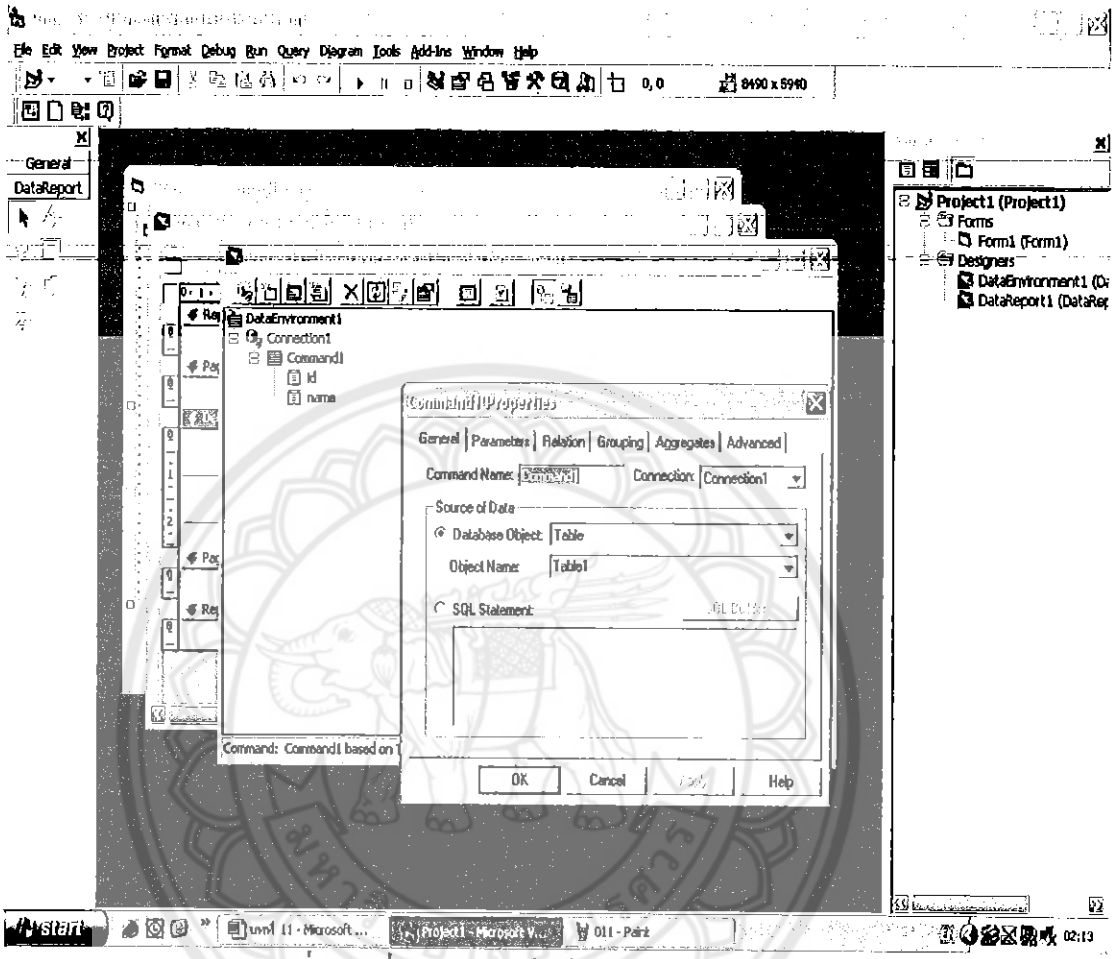
รูปที่ 2.7 การเพิ่ม Section Report Header / Footer ให้กับออบเจ็กต์ DataReport

2.8.3 สร้างแบบฟอร์มรายงานด้วยวิธี Drag & Drop

คุณสามารถสร้างแบบฟอร์มรายงาน ด้วยวิธี Drag & Drop ซึ่งมีขั้นตอนเช่นเดียวกับการแสดงข้อมูลใน DataEnvironment ได้อย่างรวดเร็วและง่ายดาย ดังนี้

1. เชื่อมต่อ Connection 1 เท่ากับฐานข้อมูลที่ต้องการใช้งาน กรณีนี้คือ db1.mdb

2. กำหนดให้ ออบเจ็กต์ Command เชื่อมโยงกับตารางที่คุณต้องการสร้างแบบฟอร์มรายงาน กรณีนี้คือ ตาราง table1



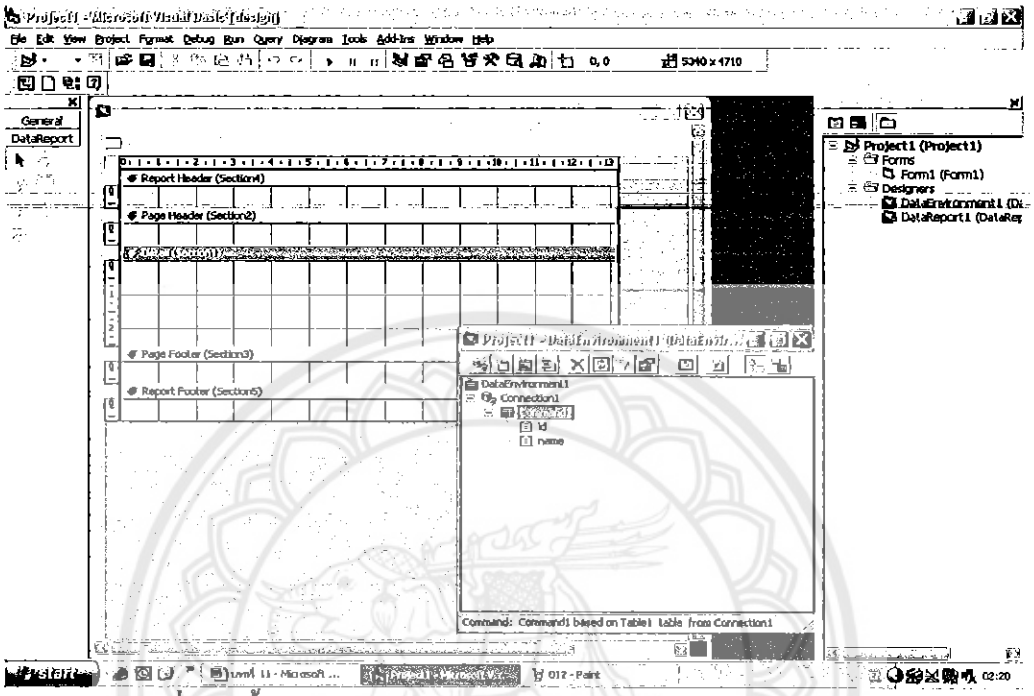
รูปที่ 2.8 เชื่อมต่อออบเจ็กต์ Command เข้ากับตาราง

ตารางที่ 2.6 จากนั้นกำหนดคุณสมบัติดังต่อไปนี้

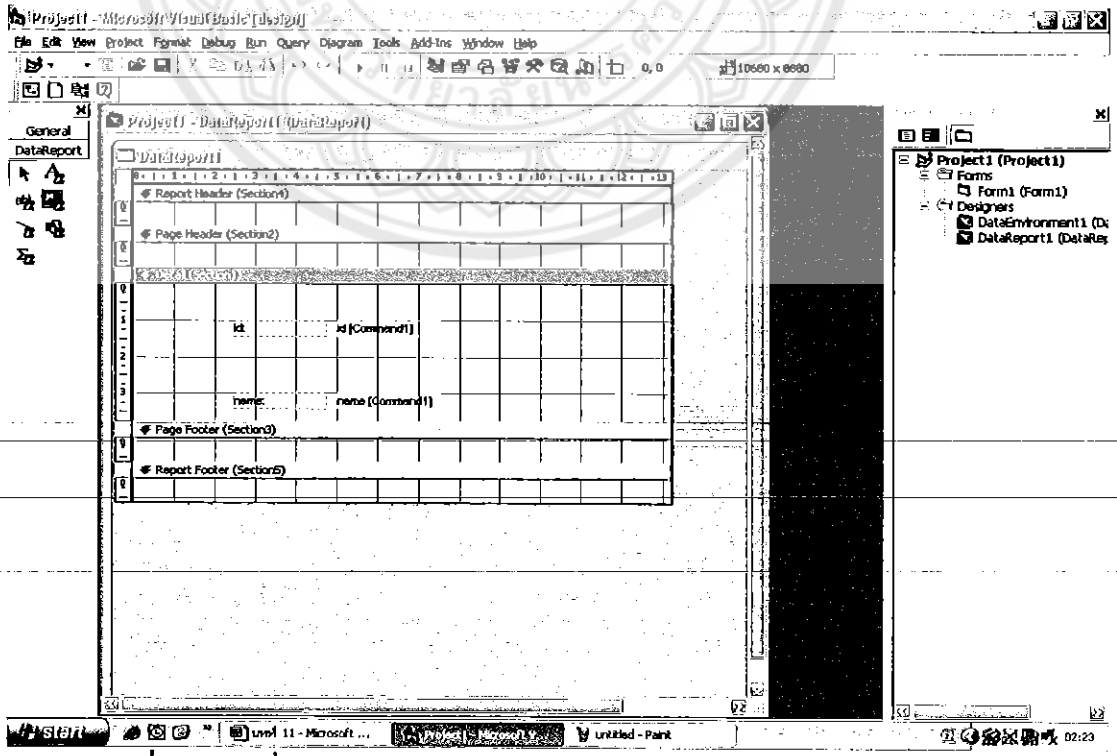
คอนโทรล	คุณสมบัติ	ค่าที่กำหนด
DataReport 1	DataSource	DataEnvironment 1
	DataMember	Command 1

การระบุชื่อออบเจ็กต์ Command .ให้กับคุณสมบัติ DataNember เพื่อใช้ในการเข้าถึงข้อมูลในฐานข้อมูล จะเห็นได้ว่า มีขั้นตอนเช่นเดียวกับ DataEnvironment

3. ลากออบเจ็กต์ Command แสดงข้อมูลทุกฟิลด์ หรือลากชื่อฟิลด์เพื่อแสดงข้อมูลเพียงบางฟิลด์ ไปยังส่วน Detail (Section 1) ของ DataReport ในกรณีนี้ สมมติว่าต้องการแสดงข้อมูลทุกฟิลด์ จึงลากออบเจ็กต์ Command 1



รูปที่ 2.9 ขั้นตอนการลากออบเจ็กต์ Command ไปยังส่วน Detail (Section 1)



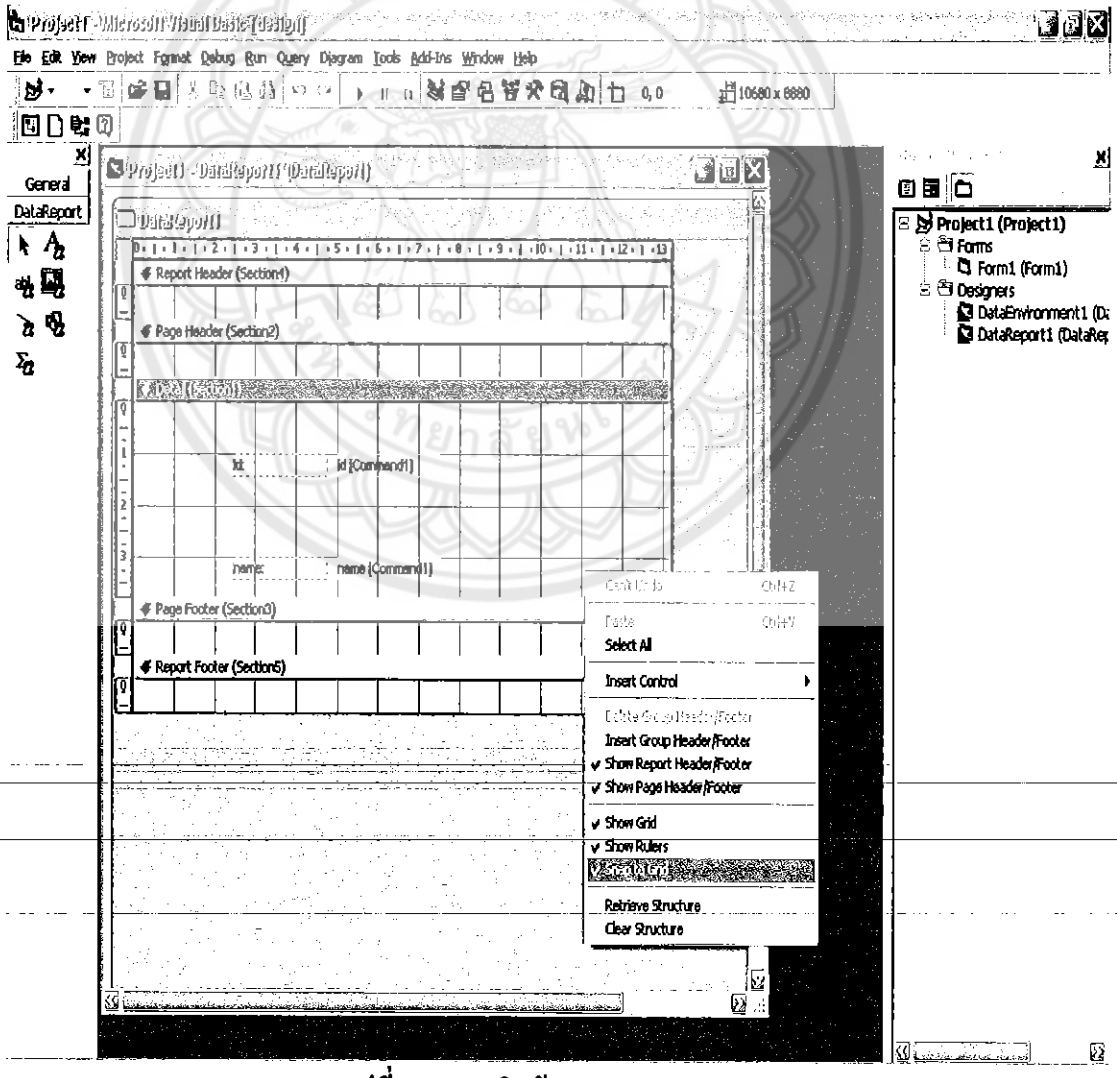
รูปที่ 2.10 ผลที่ได้หลังจากลากออบเจ็กต์ Command ไปยังส่วน Detail (Section 1)

สิ่งที่ปรากฏขึ้นมา ประกอบด้วย 2 ส่วน ก็คือ

- ชื่อฟิลด์ในตาราง จะใช้คอนโทร ล RptLabel ในการแสดงข้อความ
- ข้อมูลในฟิลด์ จะใช้คอนโทรล RptTextBox

คุณสามารถแก้ไขชื่อฟิลด์ , จัดวางตำแหน่ง รวมถึงลบชื่อฟิลด์ หรือลบข้อมูลจากฟิลด์จากฟิลด์ใดฟิลด์หนึ่งได้อย่างอิสระเพื่อปรับปรุงให้การแสดงข้อมูลในแบบฟอร์มรายงานเหมาะสมกับจุดประสงค์ในการใช้งานของคุณ ซึ่งมีลักษณะเช่นเดียวกับการแก้ไขคุณสมบัติของกลุ่ม Bound Controls

วิธีการที่แก้ไขข้อความที่ปรากฏอยู่ที่คอนโทรล RptLabel ก็คือ ให้แก้ไขที่คุณสมบัติ Caption ในการเปลี่ยนขนาดของฟิลด์ จะพบว่า คุณจะต้องกำหนดให้ได้ขนาดเท่ากับเส้น Grid ของ DataReport เสมอ ซึ่งจะทำให้ไม่ได้ขนาดตามที่คุณต้องการ ให้คุณคลิกขวาที่คุณต้องการ DataReport แล้วคลิกเลือกคำสั่ง Snap to Grid เพื่อยกเลิกข้อจำกัดนี้

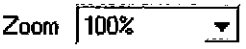


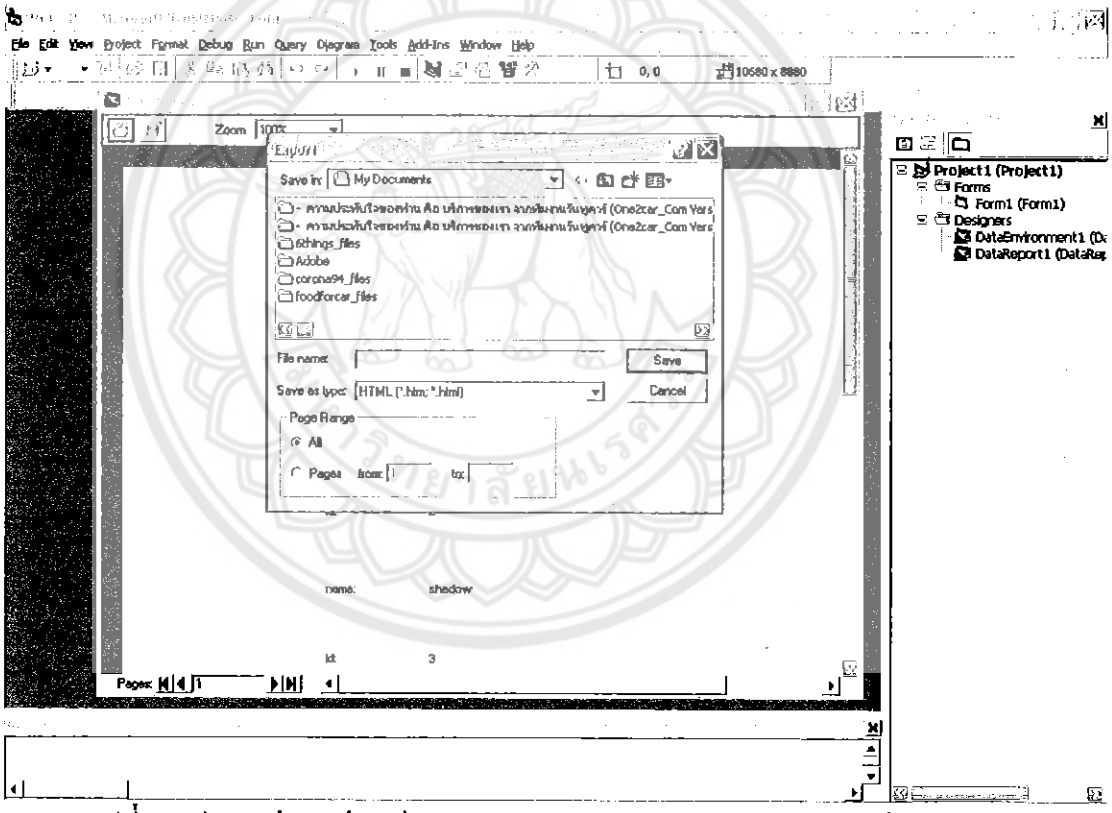
รูปที่ 2.11 ยกเลิกข้อจำกัด Snap to Grid

2.8.4 รายละเอียดการทำงานของ DataReport

ข้อมูลที่ปรากฏขึ้นมาในรูปที่เกิดจาก DataReport ซึ่งจะอ่านข้อมูลที่ถูกเก็บอยู่ในออบเจ็กต์ Recordset ที่เกิดจาก Command 1 ภายใต้การเชื่อมต่อ Connection 1

จากรูปจะมีส่วนที่น่าสนใจอยู่ 4 ส่วนดังนี้

- ไอคอน  ใช้สำหรับพิมพ์เอกสารออกทางเครื่องพิมพ์
- ไอคอน  ใช้สำหรับแปลงเอกสารเป็นไฟล์ข้อความ (*.txt) หรือไฟล์ html (*.htm, *.htm)
- รายการ  ใช้สำหรับย่อ – ขนาด เอกสาร
- ปุ่ม  ใช้สำหรับเลื่อนหน้าเอกสาร



รูปที่ 2.12 ไดอะล็อกบล็อกซ์Export ของไอคอน Export แปลงเอกสารเป็นไฟล์ html

ในไดอะล็อกบล็อกซ์ Export ให้คุณเลือกเอกสารที่ต้องการแปลงด้วยที่ Page Rang โดยที่

- ตัวเลือก All หมายถึง แปลงเอกสารทุกหน้าที่อยู่ใน DataReport
- ตัวเลือก Pages form....to.... หมายถึงกำหนดหน้าเอกสารเริ่มต้นและสิ้นสุดในการแปลงเอกสาร

การใช้งานกลุ่มคอนโทรลด้านการทำรายงาน

ตารางที่ 2.7 กลุ่มคอนค้านการทำรายงาน ประกอบไปด้วยคอนโทรล 6 ตัว ดังนี้

ไอคอน	หน้าที่
	แสดงข้อความปนรายงาน เช่นชื่อฟิลด์ , ข้อความประกอบ หรือข้อความเพิ่มเติม
	แสดงข้อมูลที่มาจากฟิลด์ใดฟิลด์หนึ่งในตาราง
	แสดงรูปภาพประกอบรายงาน
	แสดงเส้นตรงในรายงาน
	แสดงรูปทรงเรขาคณิตต่าง ๆ มีอยู่ 6 ชนิด โดยการกำหนดที่คุณสมบัติ Shape
	ใช้สำหรับคำนวณในรูปแบบต่าง ๆ มีทั้งสิ้น 8 ฟังก์ชัน ในการกำหนดที่คุณสมบัติ FunctionType รูป

ตารางที่ 2.8 หน้าที่ของฟังก์ชันทั้ง 8 ฟังก์ชันมีดังนี้

ค่าคงที่	ฟังก์ชัน	หน้าที่
0 – rptFuncSum	Sum	หาผลรวม
1 – rptFuncAve	Average	หาค่าเฉลี่ย
2 - rptFuncMin	Minimum	หาค่าต่ำสุด
3 - rptFuncMax	Maximum	หาค่าสูงสุด
4 - rptFuncRCut	Row Count	หาจำนวน Record ใน Section
5 - rptFuncVCut	Value Count	หาจำนวน Record ที่ไม่มีค่าฟิลด์ใดฟิลด์หนึ่งเป็น null
6 - rptFuncSDEV	Standard Deviation	คำนวณค่าเบี่ยงเบนมาตรฐาน
7 - rptFuncSERR	Standard Error	คำนวณค่าผิดพลาดมาตรฐาน

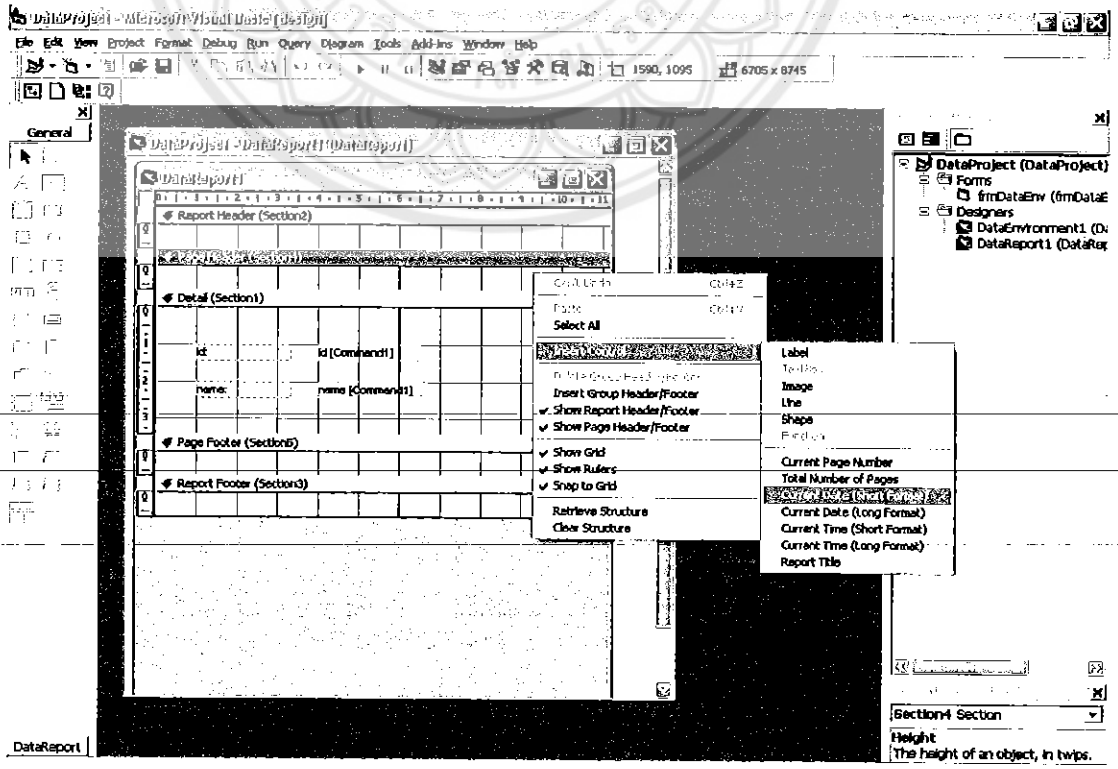
2.8.5 วิธีการเพิ่มเติมข้อมูลอื่น ๆ เข้าไปในแบบฟอร์มรายงาน

ใน DataReport ยังมีฟังก์ชันสำเร็จรูปอีกกลุ่มหนึ่ง ที่มีหน้าที่เพิ่มเติมข้อมูลในฟอร์มรายงาน ให้สมบรูณ์มากยิ่งขึ้น โดยที่คุณไม่ต้องเขียนโค้ดเลย เป็นข้อมูลมาตรฐานที่คุณพบเห็นได้ในแบบฟอร์มรายงานโดยทั่ว ๆ ไป ดังนี้

ตารางที่ 2.9 ความหมายของแต่ละฟังก์ชัน

คำสั่ง	ความหมาย
Current Page Number	แสดงหมายเลขหน้าปัจจุบัน
Total Number of Pages	แสดงจำนวนหน้าทั้งหมด
Current Data (Short Format)	แสดงวันที่ปัจจุบันแบบสั้น
Current Data (Long Format)	แสดงวันที่ปัจจุบันแบบยาว (เต็มรูปแบบ)
Current Data (Short Format)	เสดงเวลาปัจจุบันแบบสั้น
Current Data (Short Format)	แสดงเวลาปัจจุบันแบบยาว (แสดงหน่วยวินาที)
Report Titel	แสดงส่วน Title ของแบบฟอร์มรายงาน

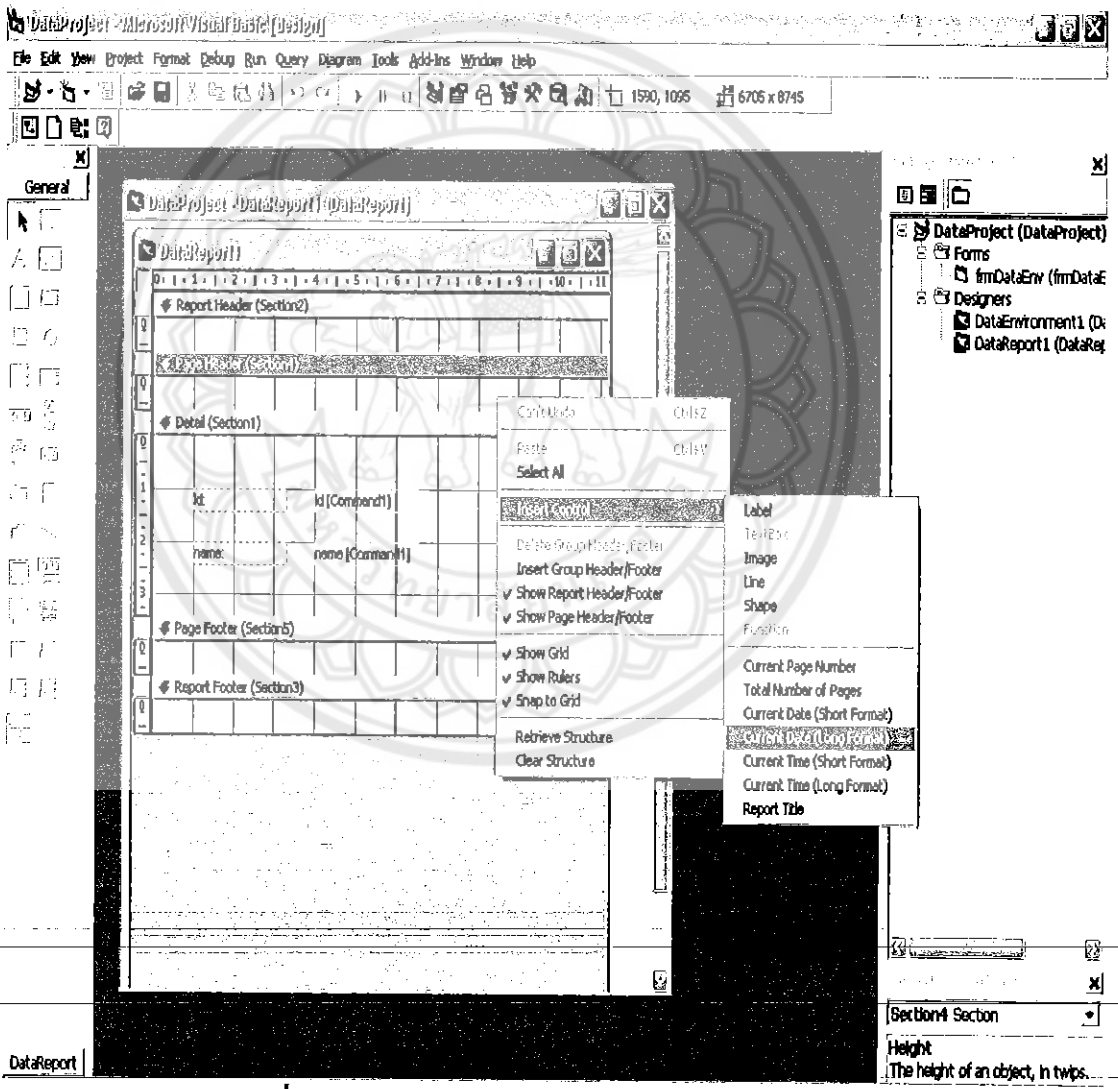
วิธีการใช้งานกลุ่มฟังก์ชันดังกล่าวก็คือ ให้คุณคลิกขวาที่บริเวณ Section ใดก็ได้ของ DataReport แล้วเลือกคำสั่ง Insert Control ดังรูป



สำหรับการใช้ข้อมูลจากกลุ่มฟังก์ชันนี้ จะขึ้นอยู่กับว่า รายงานของคุณเป็นรูปแบบใด ข้อมูลของคุณมีอะไรบ้าง ให้คุณเลือกใส่ในแต่ละ Section ตามความเหมาะสมของข้อมูลของคุณ ผู้เขียนจะนำมาใส่ข้อมูลเพิ่มเติมได้

แนวความคิดก็คือ ในแบบฟอร์มรายงานเฉพาะหน้าแรกจะแสดงวันที่และเวลา โดยที่ในทุก ๆ หน้าจะแสดงหน้าปัจจุบัน / จำนวนหน้าทั้งหมด มีขั้นตอนดังนี้

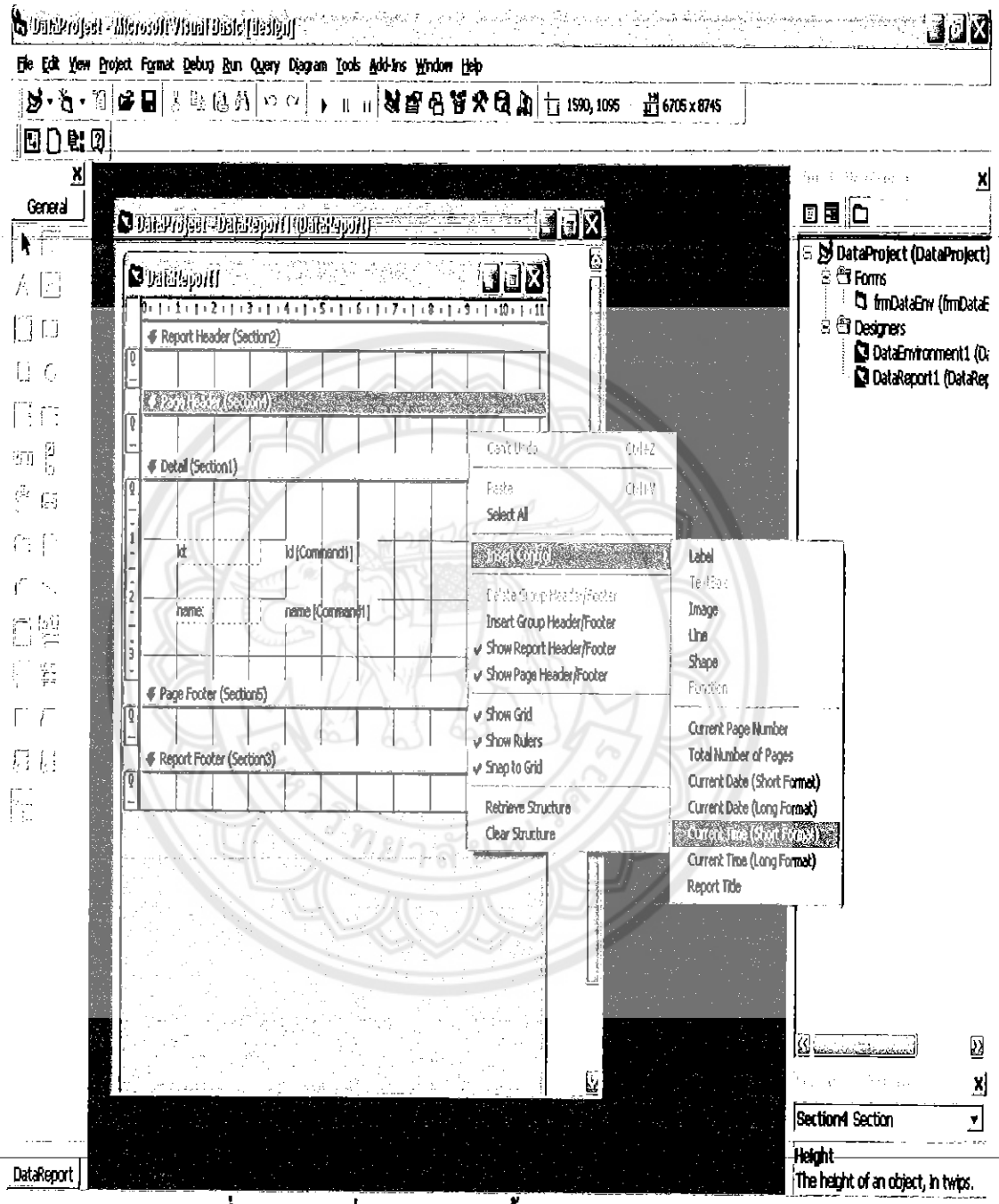
1. จากตัวอย่างที่ Report Header ให้เพิ่มคอนโทรล RptLabel 1 ตัว และแก้ไขคุณสมบัติ Caption = วันที่ จากนั้นคลิกขวา เลือกคำสั่ง Insert Control > Current Data (Long Format) ดังรูปที่ 2.10



รูปที่ 2.14 แสดงเมนู pop – up ของส่วน Report Header

จากรูปที่ 2.10 เหตุผลที่ใส่ไว้ในส่วนนี้ เนื่องจากว่า วันที่จะปรากฏขึ้นมาในรายงานหน้าแรกเท่านั้น ส่วนเวลาให้ทำเช่นกัน โดยการเพิ่มคอนโทรล RptLabel อีก 1 ตัว ให้แก้ไขคุณสมบัติ

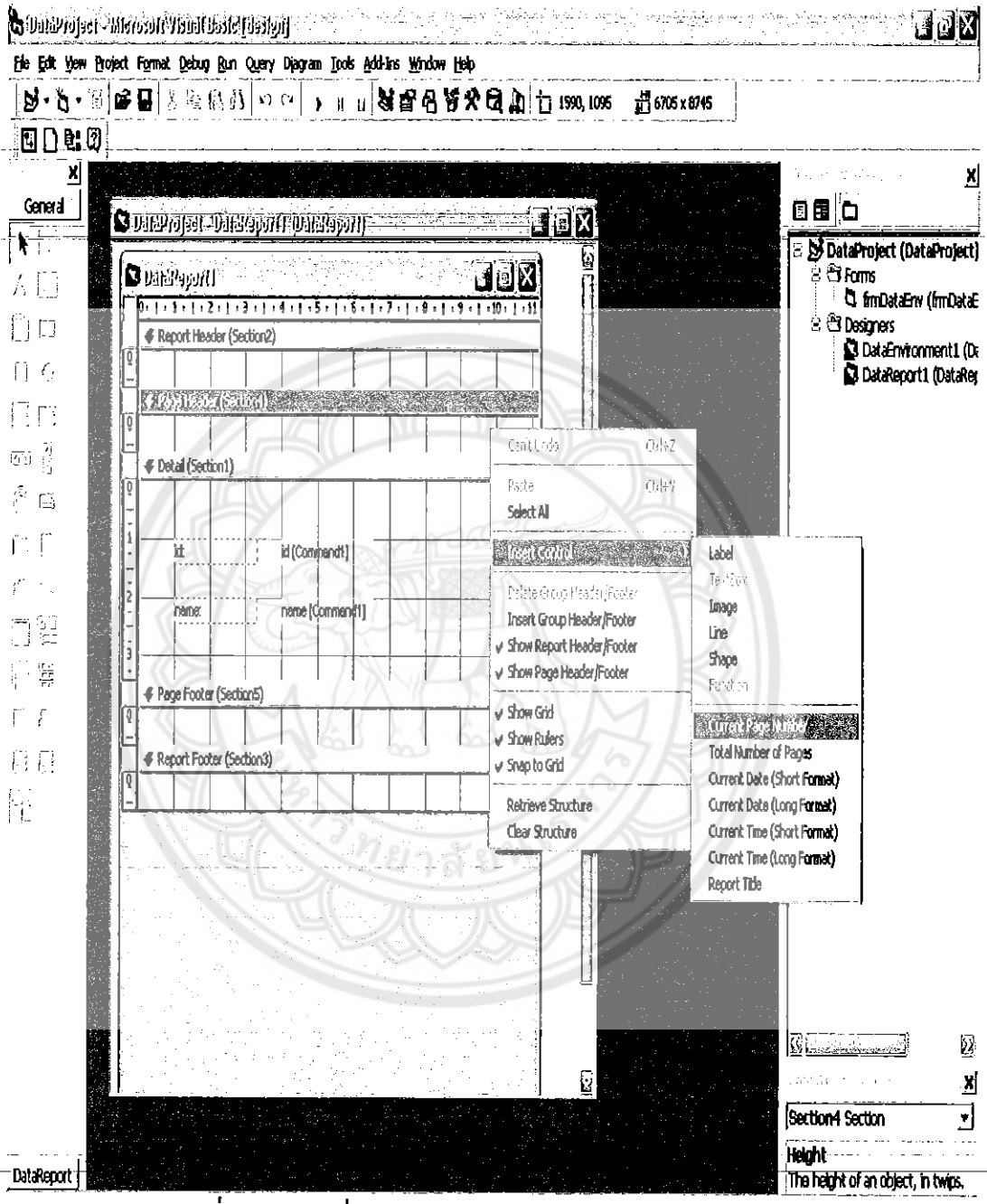
Caption = เวลา จากนั้นคลิกขวา เลือกคำสั่ง Insert Control > Current Time (Short Format) ดังรูป



รูปที่ 2.15 การเพิ่มเวลารูปแบบสั้นให้กับส่วน Report Header

2. ที่ส่วน Page Header (Section2) จะใช้สำหรับแสดงหมายเลขหน้าปัจจุบันและจำนวนหน้าทั้งหมด เช่น หน้า 1 / 5 , หน้า 2 / 5 , หน้า 3 / 5 .... เป็นต้น เหตุผลที่ใช้ในส่วนนี้ เพื่อต้องการให้ปรากฏอยู่ในรายงานทุกหน้า

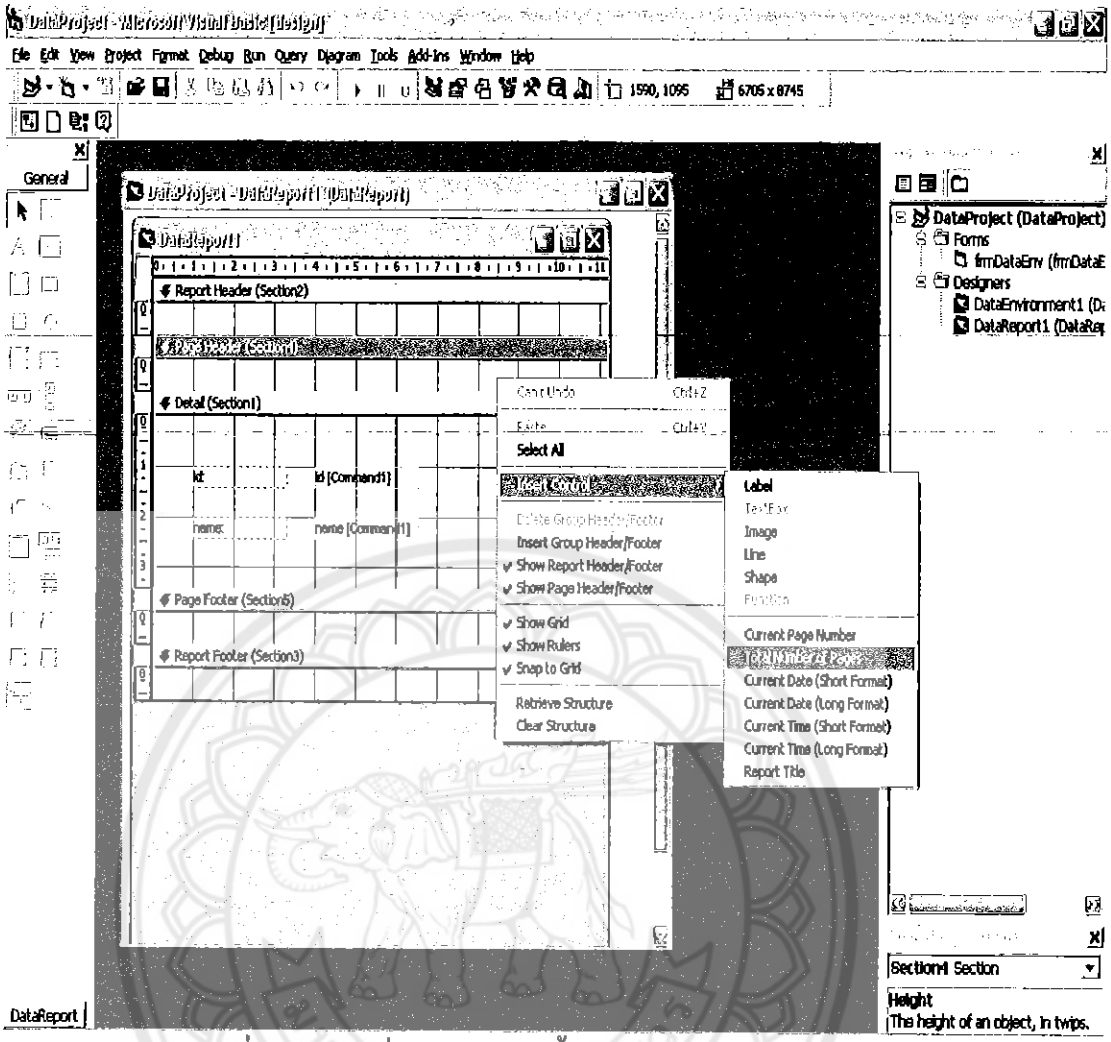
3. ให้คุณเพิ่มคอนโทรล RptLabel 1 ตัวแล้วแก้ไขคุณสมบัติ Caption = หน้า จากนั้น คลิกขวา เลือกคำสั่ง Insert Control > Current Page Number ดังรูป



รูปที่ 2.16 การเพิ่มหมายเลขหน้าปัจจุบันให้กับส่วน Page Header

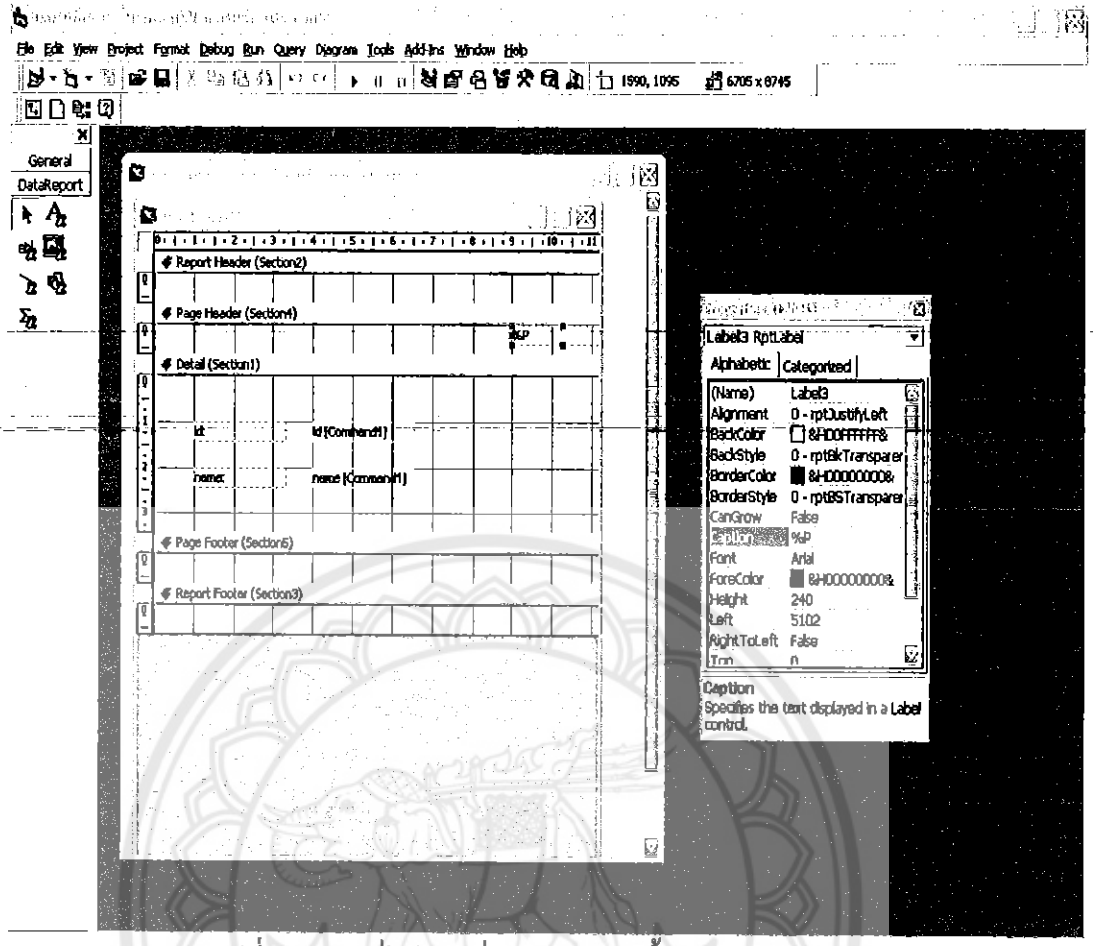
4. ต่อมาให้คุณเพิ่มคอนโทรล RptLabel อีก 1 ตัว แก้ไขคุณสมบัติ Caption / จากนั้น คลิกขวา เลือกคำสั่ง Insert Control > Total Number of Pages เพื่อแสดงจำนวนหน้าทั้งหมด ดังรูปที่ 2.15

ส่วนรูปที่ 2.16และ 2.17 เป็นการรันโปรเจ็กต์ เป็นรายงานที่มีทั้งสิ้น 2 หน้า



รูปที่ 2.17 การเพิ่มจำนวนหน้าทั้งหมดให้กับส่วน Page Header

จากตัวอย่างโปรเจ็กต์ที่ 11 – 3 คุณจะพบว่า กลุ่มฟังก์ชันพิเศษเหล่านี้ แท้ที่จริงแล้วก็คือ การใช้คอนโทรล RptLabel ร่วมกับการใส่ตัวอักษรพิเศษให้กับคุณสมบัติ Caption นั่นเอง เช่น จำนวนหน้าทั้งหมด (Total Number of Pages) จะใช้ตัวอักษร % P ดังรูปที่ 2.16



รูปที่ 2.18 ผลที่ได้การเพิ่มจำนวนหน้าทั้งหมดให้กับส่วน Page Header

ตารางที่ 2.10 ตารางแสดงความหมายของตัวอักษรพิเศษที่กำหนดให้กับคุณสมบัติ Caption ของคอนโทรล RptLabel

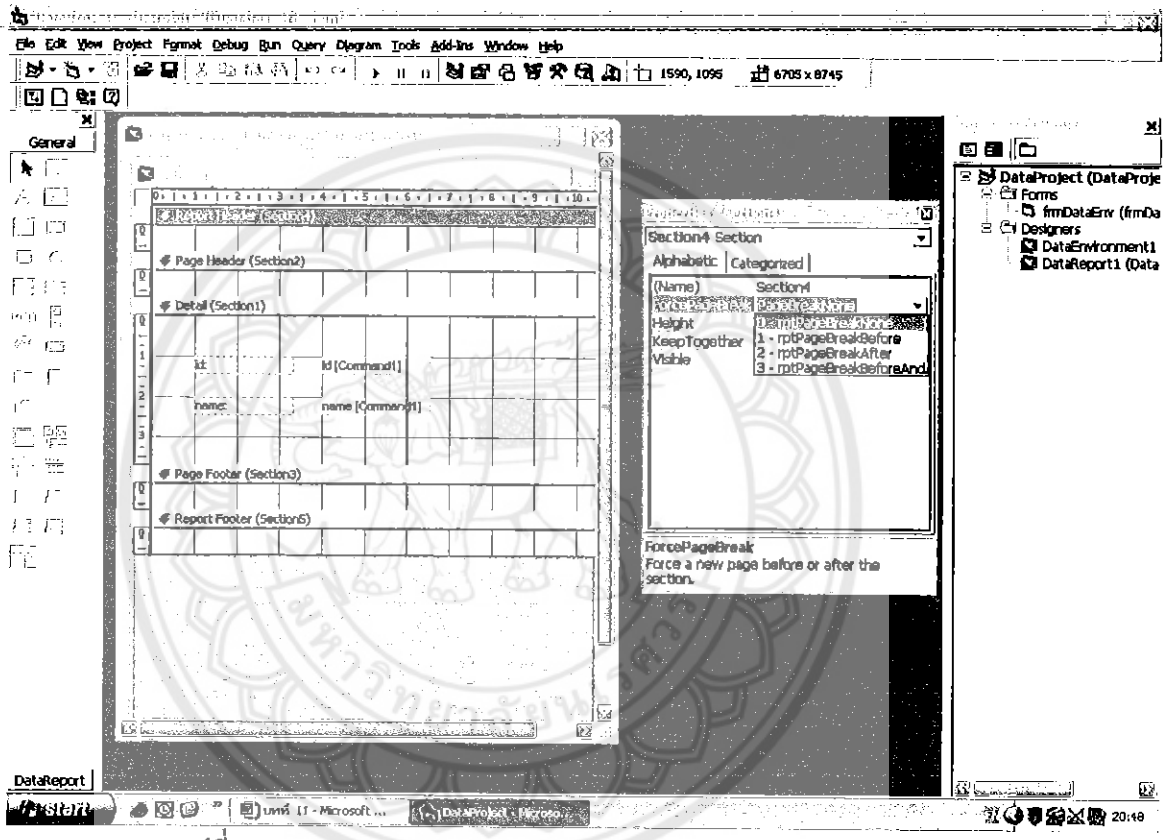
คำสั่ง	ตัวอักษรพิเศษ
Current Page Number	%p
Total Number of Pages	%p
Current Data (Short Format)	%d
Current Data (Long Format)	%D
Current Data (Short Format)	%t
Current Data (Short Format)	%T
Report Titel	%i

2.8.6 วิธีการกำหนด Page Break

การกำหนด Page Break มีจุดประสงค์เพื่อกำหนดให้ข้อความขึ้นหน้าใหม่ ดูรสามารถเพิ่ม Page Break ได้ทั้งสิ้น 2 จุด คือ

- ส่วน Report Header
- ส่วน Report Footer

ขั้นตอนในการกำหนด Page Break ก็คือ ที่ส่วน Report Header หรือ Report Footer ให้คุณแก้ไขคุณสมบัติ ForcePageBreak ดังรูปที่ 2.17



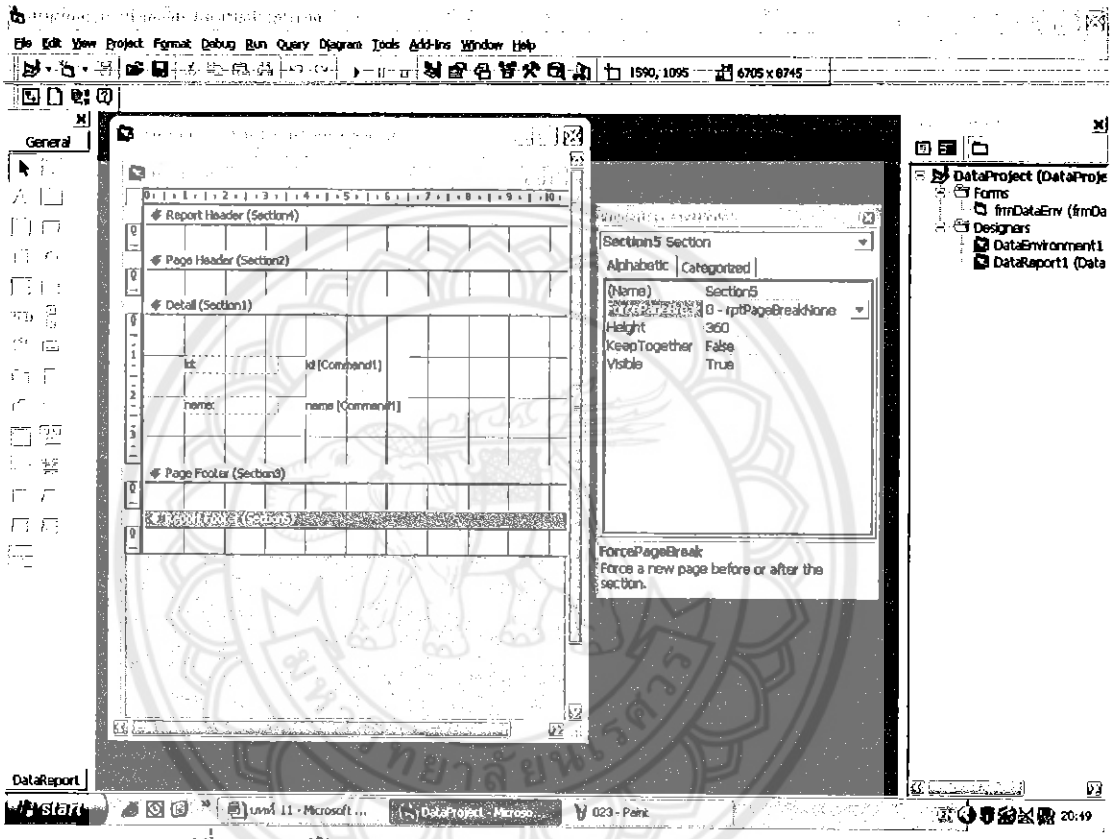
รูปที่ 2.19 แสดงคุณสมบัติForcePageBreakของส่วน Report Header

ตารางที่ 2.11 ความหมายของค่าคงที่แต่ละชนิด แสดงดังตารางต่อไปนี้

ค่าคงที่	ความหมาย
0 - rptPageBreakNone	(Default) ไม่มีการใช้ PageBreak จะขึ้นหน้าใหม่พื้นที่แบบฟอร์มรายงานไม่สามารถแสดงข้อความได้อีก
1 - rptPageBreakBefore	กำหนดให้ส่วนนี้ ขึ้นหน้าใหม่ทันนี้
2 - rptPageBreakAfter	กำหนดให้ส่วนที่ต่อจากนี้ ขึ้นหน้าใหม่
3 - rptPageBreakBeforeAndAfter	กำหนดให้ส่วนนี้ขึ้นหน้าใหม่ และส่วนที่ต่อ

จากนี้ขึ้นหน้าใหม่เช่นกัน

ให้คุณนำโปรเจ็กต์ มาเพิ่ม Page Break เป็นตัวอย่างโปรเจ็กต์ส่วน Report Footer (Section5) ซึ่งแสดงจำนวน Record ทั้งหมดที่มีอยู่ ต้องการให้แสดงหน้าใหม่ ให้แก้ไขคุณสมบัติ ForcePageBreak 1 - rptPageBreakBefore ดังรูปที่ 2.18



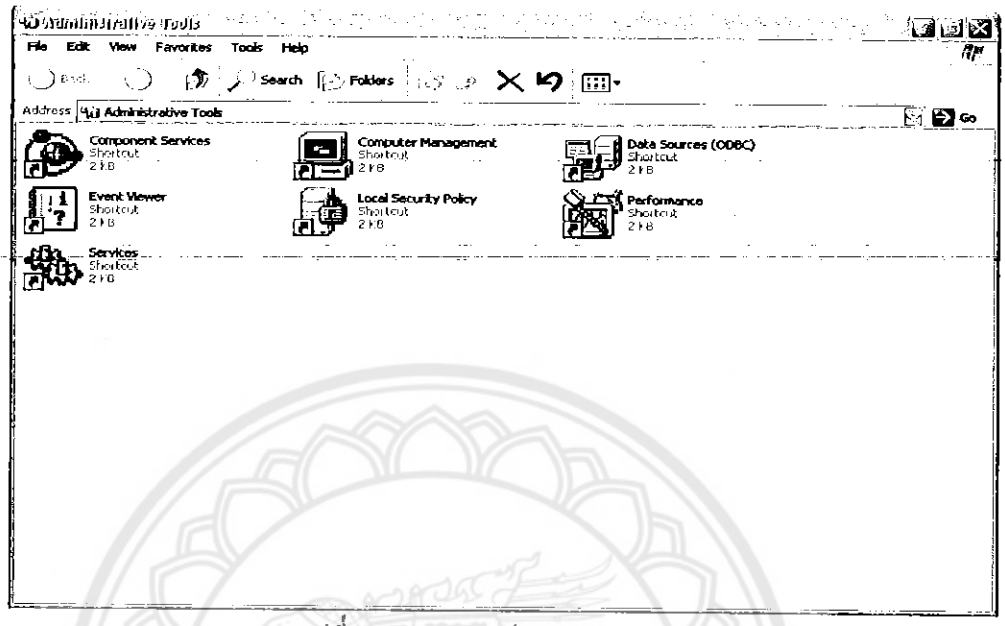
รูปที่ 2.20 แก้ไขคุณสมบัติ ForcePageBreak ของส่วน Report Footer

จากผลการรันจะพบว่า มีจำนวนหน้าเพิ่มขึ้นมาอีก 1 หน้า เนื่องจากข้อความแสดงจำนวน Record ถูกกำหนดให้อยู่ในหน้า

2.9 ODBC Data Source Name

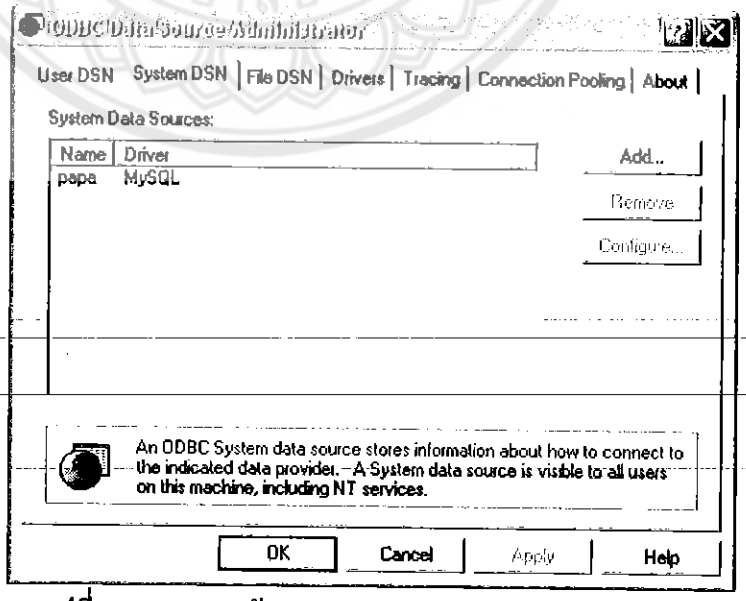
วิธีนี้เรียกทับศัพท์ว่า การ Setup DSN เป็นการกำหนดชื่อ DNS เพื่อเชื่อมโยงไปยังฐานข้อมูลที่ต้องการใช้งาน ผ่านทาง ODBC เมื่อคุณต้องการใช้งานฐานข้อมูลในภายหลัง ก็จะใช้ชื่อ DSN ที่คุณตั้งขึ้นมา เป็นชื่ออ้างอิงไปยังฐานข้อมูลที่คุณกำหนดไว้นั่นเอง มีขั้นตอนดังนี้

1. ที่ desktop ของ Windows เลือก Start > Settings > Control Panel แล้วดับเบิ้ลคลิกที่ไอคอน Administrative Tools



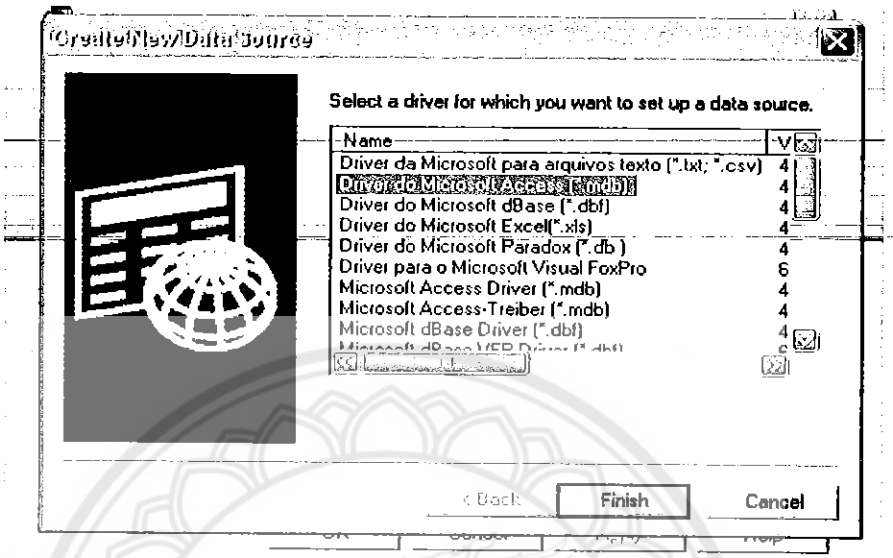
รูปที่ 2.21 แสดงหน้าจอ Administrative Tools

2. ดับเบิ้ลคลิกที่ไอคอน Data Sources (ODBC) ในไดอะล็อกบ็อกซ์ Administrative Tools คลิกที่แท็บ System DSN และคลิกที่ Add... เพื่อเพิ่มชื่อ DSN เข้าไปในระบบ



รูปที่ 2.22 แสดงหน้าจอ ODBC Data Source Administrator

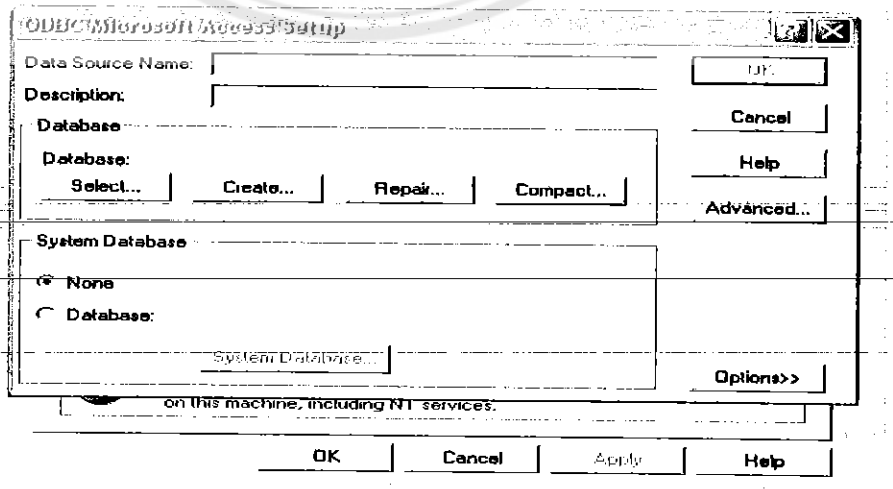
3. จากนั้นคลิกเลือกชนิดฐานข้อมูลที่ต้องการติดต่อ ในกรณีเป็น Access ต้องเลือก Microsoft Access Drivers (*.mdb) แล้วคลิกที่ Finish



รูปที่ 2.23 แสดงหน้าจอการเลือก driver

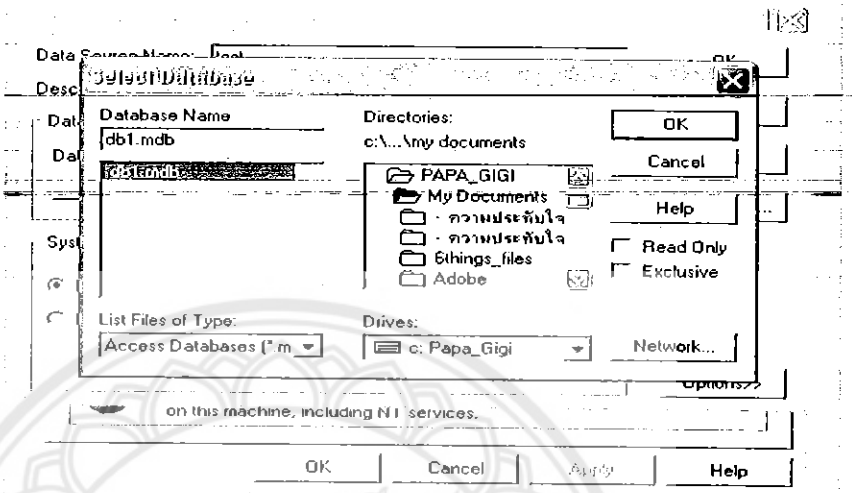
4. ในไดอะล็อกบ็อกซ์ ODBC Microsoft Access Setup มีรายละเอียดดังนี้

- ช่อง Data Source Name ให้คุณตั้งชื่อ DSN ตามที่คุณต้องการ มีข้อเสนอแนะว่า ให้คุณตั้งชื่อ DSN เป็นชื่อเดียวกับฐานข้อมูลที่ต้องการใช้งาน กรณีต้องการใช้งาน student.mdb จึงตั้งชื่อ DSN ว่า student
- ช่อง Description ใช้สำหรับระบุรายละเอียด คุณจะใส่หรือไม่ก็ได้



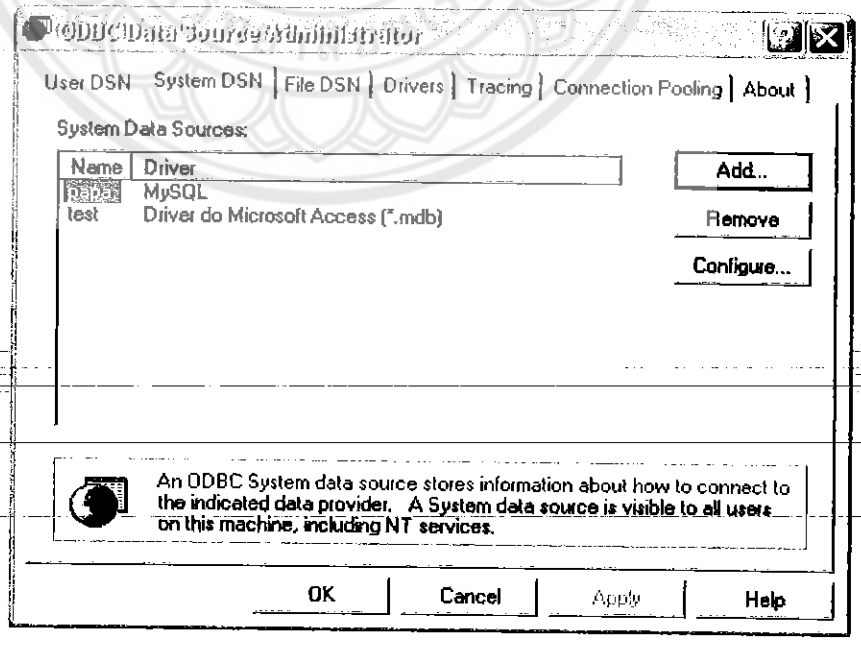
รูปที่ 2.24 แสดงหน้าจอรับข้อมูลการกำหนดชื่อ Data Source Name

จากนั้นคลิกที่ Select... เพื่อเลือกฐานข้อมูลที่ต้องการติดต่อ ให้คุณเลือกพาธของฐานข้อมูลที่ต้องการเขียน ต่อจากนั้นคลิกที่ OK ในไดอะล็อกบ็อกซ์ Select Database และคลิกที่ OK ในไดอะล็อกบ็อกซ์ ODBC Microsoft Access Setup



รูปที่ 2.25 แสดงหน้าจอเลือกฐานข้อมูล

5. จะเห็นได้ว่าในไดอะล็อกบ็อกซ์ ODBC Data Source Administrator มีชื่อ DSN ที่คุณตั้งไว้เพิ่มเข้ามา แล้วคลิกที่ OK



รูปที่ 2.26 แสดงผลที่ได้จากการกำหนดตามขั้นตอน

2.10 วิธีใช้ Connection String

เป็นวิธีที่เขียนข้อความเชื่อมต่อกับฐานข้อมูลที่ต้องการใช้งาน โดยที่จะกำหนดตัว OLEDB Provider และชื่อฐานข้อมูลที่ต้องการติดต่อ ข้อความข้างล่างนี้ได้ copy มาจากช่อง Use Connection String

Provicer=Microsoft.Jet.OLEDB.4.0; Data Source=C:\Documents and Setting\My Document\db1.mdb; Persist Security Info=False

จะต้องเขียนข้อความนี้เอง เมื่อเรียกใช้งานกลุ่มออบเจ็กต์ ADO โดยตรง สามารถแบ่งออกได้ 3 ส่วน คือ ในส่วนแรกเป็น OLEDB Provider ที่ใช้งาน ข้อความต่อมาที่ผู้เขียนเน้นเป็นพาธของฐานข้อมูลที่ต้องการติดต่อ ส่วนสุดท้ายเป็นการกำหนดการรักษาความปลอดภัย แต่ละส่วนจะถูกคั่นด้วยเครื่องหมาย ;

2.11 โปรแกรมฐานข้อมูลที่นำมาใช้ในโครงงานนี้

MySQL จัดเป็นระบบฐานข้อมูลเชิงสัมพันธ์ (RDBMS:Relational Database Management Systemd) ตัวหนึ่ง ซึ่งเป็นที่นิยมกันมากในปัจจุบันโดยเฉพาะอย่างยิ่งในโลกอินเตอร์เน็ต สาเหตุก็เพราะว่า MySQL เป็นฟรีแวร์ทางด้านฐานข้อมูลที่มีประสิทธิภาพสูง เป็นทางเลือกใหม่จากผลิตภัณฑ์ระบบจัดการฐานข้อมูลในตลาดปัจจุบัน ที่มักจะเป็นการผูกขาดของผลิตภัณฑ์เพียงไม่กี่ตัว นักพัฒนาระบบฐานข้อมูลที่เคยใช้ MySQL ต่างยอมรับในความรวดเร็ว การรองรับจำนวนผู้ใช้และขนาดของข้อมูลจำนวนมหาศาล ทั้งยังสนับสนุน การใช้งานบนระบบปฏิบัติการมากมาย ไม่ว่าจะเป็น Unix, OS/2, Mac OS หรือ Windows ก็ตาม นอกจากนี้ MySQL ยังสามารถใช้ร่วมงานกับ Web Development Platform ทั้งหลาย ไม่ว่าจะเป็น C, C++, Java, Perl, PHP, Python, Tcl หรือ ASP ก็ตามที ดังนั้นจึงไม่น่าแปลกใจทำไม MySQL จึงได้รับความนิยมมากในปัจจุบัน และยังมีแนวโน้มสูงขึ้นยิ่งขึ้นต่อไปในอนาคต

MySQL ได้รับการยอมรับและทดสอบความรวดเร็วในการใช้งาน โดยมีการทดสอบและเปรียบเทียบ กับผลิตภัณฑ์ทางด้านฐานข้อมูลอื่นอยู่เสมอ มีการพัฒนาอย่างต่อเนื่อง โดยเริ่มตั้งแต่เวอร์ชันแรก ๆ ที่ยังไม่ค่อยมีความสามารถมากนัก จนถึงทุกวันนี้ MySQL ได้รับการพัฒนาให้มีความสามารถยิ่งขึ้น รองรับข้อมูลจำนวนมหาศาล สามารถใช้งานหลายผู้ใช้ได้พร้อม ๆ กัน (Multi-user) มีการออกแบบให้สามารถแตกงานออกเพื่อช่วยการทำงานให้เร็วยิ่งขึ้น (Multi-threaded) และการเชื่อมต่อที่ดีขึ้น การกำหนดสิทธิและการรักษาความปลอดภัยของข้อมูลมีความน่าเชื่อถือยิ่งขึ้น เครื่องมือหรือโปรแกรมสนับสนุนทั้งของตัวเองและผู้พัฒนาอื่นๆ นอกจากนี้สิ่งที่สำคัญคือ MySQL

ได้รับการพัฒนาไปในแนวทางตามข้อกำหนดมาตรฐาน SQL ดังนั้นเราสามารถใช้คำสั่ง SQL ในการทำงานกับ MySQL ได้

จากการใช้งาน MySQL พบว่าการทำงานกับฐานข้อมูล MySQL นั้นจะต้องกระทำผ่านบรรทัดคำสั่ง ซึ่งเป็นเรื่องที่ไม่สะดวก ด้วยเหตุนี้ Tobias Retschiller จึงได้เขียนสคริปต์ PHP ขึ้นมาชุดหนึ่ง เพื่อใช้จัดการควบคุม และเปลี่ยนแปลงรายละเอียดต่างๆในฐานข้อมูล MySQL สคริปต์ชุดนี้ถูกเรียกว่า phpMyAdmin ซึ่งมีความสามารถหลักๆดังนี้

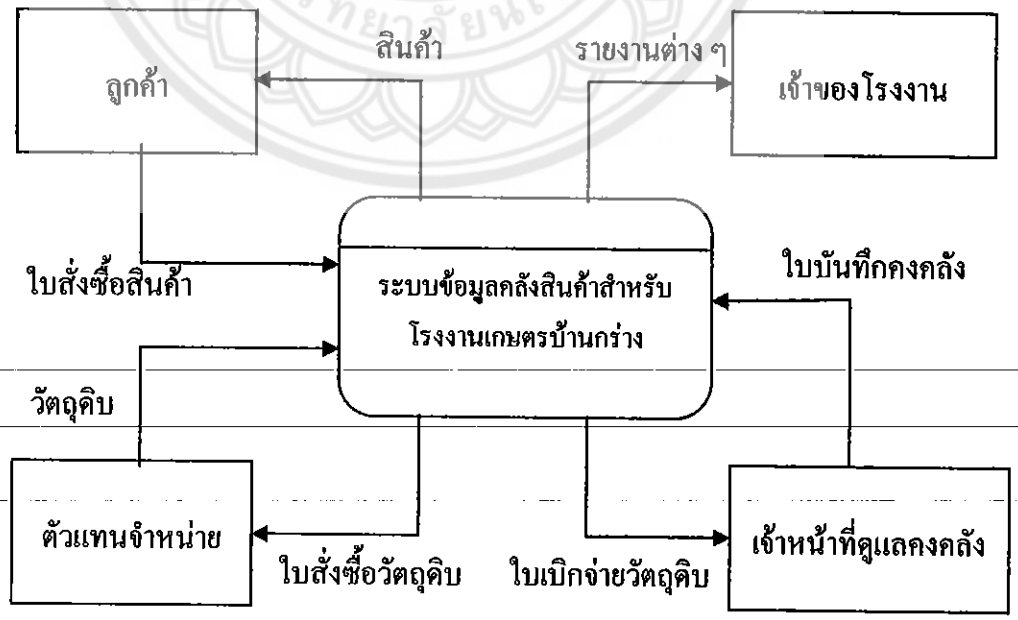
- สร้างและลบฐานข้อมูล
- สร้าง, ก๊อบปี้ และลบเทเบิล
- เพิ่มเติม, ลบ และแก้ไขฟิลด์ต่าง ๆ ของเทเบิล
- ประมวลผลคำสั่ง SQL
- Dump โครงสร้างและข้อมูลในเทเบิลออกมาเป็นไฟล์ข้อความ (Text File)
  โหลดข้อมูลจากไฟล์ข้อความเข้าไปยังเทเบิล

บทที่ 3
การศึกษาและพัฒนาโปรแกรมการจัดการระบบข้อมูลคลังสินค้า สำหรับโรงงานเกษตรบ้านกร่าง

จากการที่ได้เข้าไปรับทราบ และได้เก็บข้อมูลเบื้องต้นแล้ว พอที่จะเห็นปัญหาที่เกิดขึ้นภายในโรงงานที่ได้ติดต่อเอาไว้ ซึ่งปัญหาหลักก็คือขาดการดูแลที่เป็นระบบ การที่จะตรวจสอบข้อมูลวัสดุอุปกรณ์ต่าง ๆ นั้นเป็นไปได้ยาก เพียงแต่มีการประมาณการเอาคร่าว ๆ ว่าจากวัสดุเท่านี้ควรที่จะได้ผลิตภัณฑ์สำเร็จออกมาสักเท่าใด ไม่มีการเตรียมการในเรื่องของการวางแผนการผลิต เพราะบางครั้งผลิตทิ้งเอาไว้จนที่ภายในโรงงานไม่สามารถจัดเก็บได้ แต่บางครั้ง กลับไม่สามารถผลิตส่งให้กับลูกค้าได้ทัน ทั้งที่เปิดทำการมาไม่น้อยกว่า 10 ปีแล้ว ดังนั้นปัญหาแรกที่ควรจัดการแก้ไขจึงควรที่จะเป็น ปัญหาการจัดการระบบของคงคลังสินค้า

ในบทนี้จึงจะขอกล่าวถึงแนวทางในการศึกษาและพัฒนาโครงงาน โดยได้นำเอาขั้นตอนการวิเคราะห์และออกแบบระบบ มาใช้ในการจัดการระบบและฐานข้อมูลที่จะใช้งานในโครงงาน โดยมีรายละเอียดดังต่อไปนี้

3.1 การออกแบบในระบบหลักการ (Context Diagram)



รูปที่ 3.1 Context Diagram

3.2 แผนภาพกระแสข้อมูล (Data Flow Diagram)

เป็นแผนภาพกระแสข้อมูลที่ได้มีการวิเคราะห์ในแบบเชิงโครงสร้าง โดยแผนภาพกระแสข้อมูลนี้ใช้เป็นเครื่องมือในการพัฒนาระบบงาน เพื่อแสดงความสัมพันธ์ระหว่างโปรเซสกับข้อมูลที่เกี่ยวข้องโดยข้อมูลในแผนภาพจะทำให้ทราบถึง ที่มาของข้อมูลว่าข้อมูลดังกล่าวมาจากไหนและข้อมูลไปที่ไหน ข้อมูลเก็บที่ใดและเกิดเหตุการณ์ใดกับข้อมูลในระหว่างทาง

3.2.1 แผนภาพกระแสข้อมูลระดับที่ 1

แผนภาพกระแสข้อมูลระดับที่ 1 (Data flow Diagram level -1) จะนำเอา context diagram มาแตกรายละเอียด (exploded) โดยจะแสดงถึงโปรเซสหลัก ๆ และผู้ที่เกี่ยวข้องกับระบบ รวมทั้งข้อมูลที่เป็น primary data

จากระบบข้อมูลคลังสินค้าโรงงานเกษตรบ้านกร่างสามารถทำการวิเคราะห์เพื่อที่จะแสดงรายละเอียดของ boundaries, data และ process ดังรายละเอียดต่อไปนี้

List of Boundaries

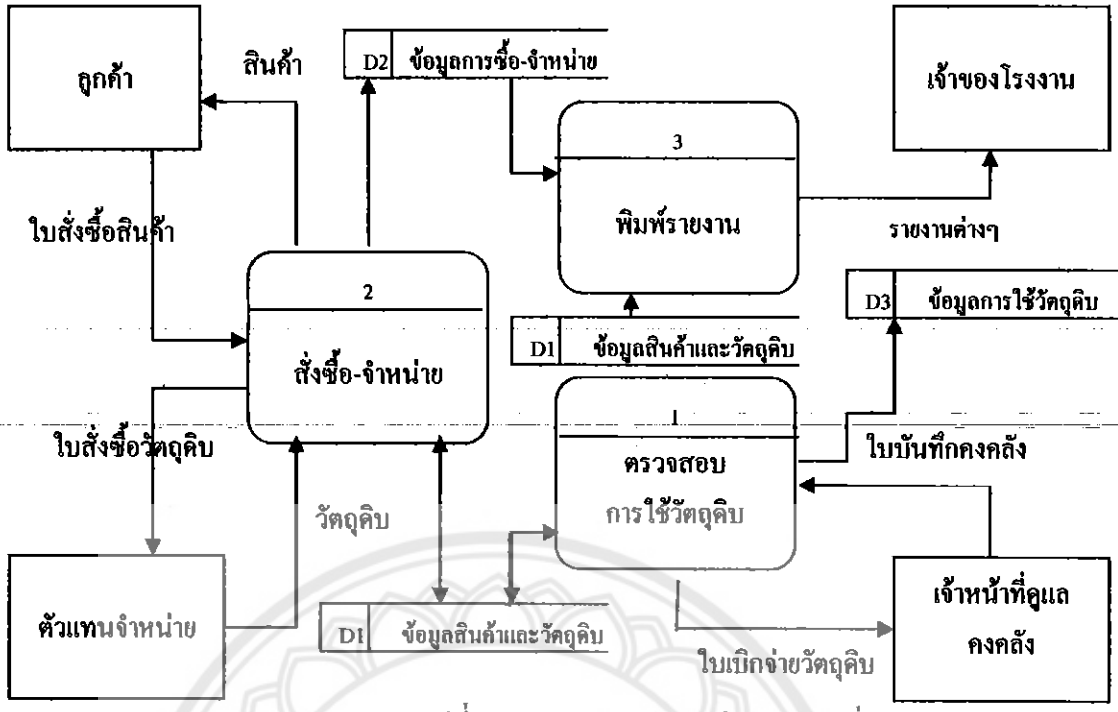
1. เจ้าของโรงงาน
2. ตัวแทนจำหน่าย (วัตถุดิบ)
3. ลูกค้า
4. เจ้าหน้าที่ดูแลคงคลัง

List of processes

1. ตรวจสอบการใช้วัตถุดิบ
2. สั่งซื้อ-จำหน่าย
3. พิมพ์รายงาน
4. จำหน่ายสินค้า
5. สั่งซื้อวัตถุดิบ
6. แสดงข้อมูลการใช้วัตถุดิบ
7. บันทึกการใช้วัตถุดิบ

List of data

1. ข้อมูลสินค้าและวัตถุดิบ
2. ข้อมูลการซื้อ- จำหน่าย
3. ข้อมูลการใช้วัตถุดิบ



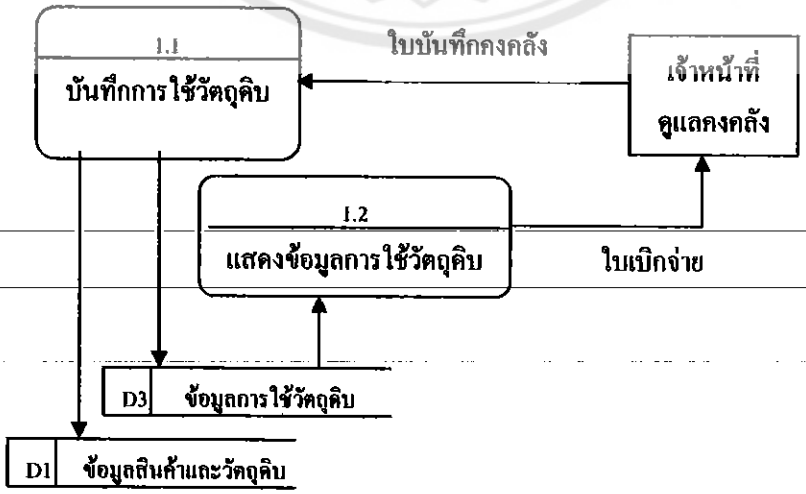
รูปที่ 3.2 แผนภาพกระแสข้อมูลระดับที่ 1

3.2.2 แผนภาพกระแสข้อมูลระดับที่ 2

แผนภาพกระแสข้อมูลระดับที่ 2 (Data flow Diagram level -2) แสดงถึงโปรเซสย่อยในแผนภาพกระแสข้อมูลระดับที่ 1 โดยแผนภาพกระแสข้อมูลระดับที่ 2 ของโปรเซสที่ 1ประกอบด้วยโปรเซสย่อยๆ 2 โปรเซสคือ

โปรเซสที่ 1.1 บันทึกการใช้วัตถุดิบ

โปรเซสที่ 1.2 แสดงข้อมูลการใช้วัตถุดิบ

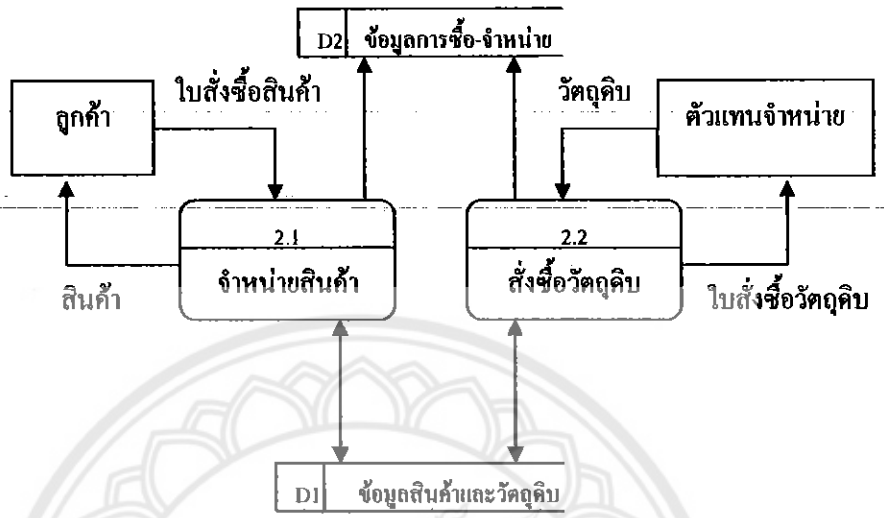


รูปที่ 3.3 แผนภาพกระแสข้อมูลระดับที่ 2 ของโปรเซสที่ 1

แผนภาพกระแสข้อมูลระดับที่ 2 ของโปรเซสที่ 2 ประกอบด้วย 2 โปรเซสย่อยๆคือ

โปรเซสที่ 2.1 จำหน่ายสินค้า

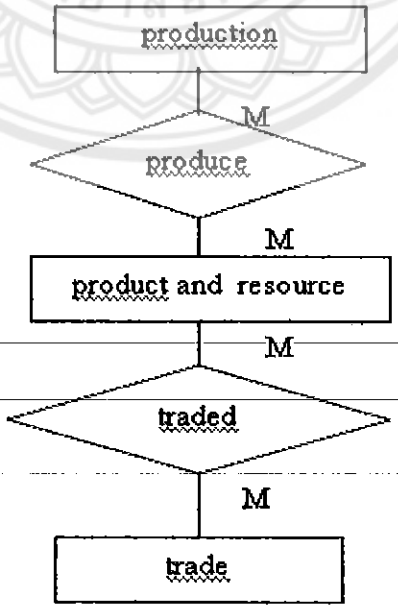
โปรเซสที่ 2.2 สั่งซื้อวัตถุดิบ



รูปที่ 3.4 แผนภาพกระแสข้อมูลระดับที่ 2 ของโปรเซสที่ 2

3.3 การออกแบบฐานข้อมูล

หลังจากที่ได้ทำการศึกษาในส่วนของฐานข้อมูลแล้วเมื่อทำการออกแบบได้ผังแสดงความสัมพันธ์ของระบบฐานข้อมูลดังรูปที่ 3.5



รูปที่ 3.5 ผังแสดงความสัมพันธ์ของระบบฐานข้อมูล

โดยที่ตาราง production จะจัดเก็บในส่วนของข้อมูลการใช้วัตถุดิบและสินค้าที่ได้เมื่อได้ผ่านกระบวนการผลิตเรียบร้อยแล้ว

ตารางที่ 3.1 ตาราง พจนานุกรมข้อมูลของตาราง Production

Attribute	Description	type	Primary Key
p_date	วันที่ทำการบันทึกข้อมูลการใช้วัตถุดิบและผลผลิตที่ได้	varchar	Y
p_pr_id	รหัสสินค้าและวัตถุดิบ	varchar	
p_pr_name	ชื่อสินค้าและวัตถุดิบ	varchar	
p_pr_quan	จำนวนที่ได้	varchar	

ในส่วนของ Product and Resource นั้น จะแยกส่วนข้อมูลที่ไม่ค่อยมีการเปลี่ยนแปลงเป็นส่วนที่ แสดงถึงรายละเอียด แยกไว้ในตาราง pr ส่วนตาราง stockจะเก็บในส่วนที่แสดงจำนวนที่มีในคงคลัง

ตารางที่ 3.2 ตารางพจนานุกรมข้อมูลของตาราง pr

Attribute	Description	type	Primary Key
pr_id	รหัสสินค้าและวัตถุดิบ	varchar	Y
pr_name	ชื่อสินค้าและวัตถุดิบ	varchar	
pr_unit	หน่วยที่ใช้เรียก	varchar	
pr_type	ประเภทของสินค้าและวัตถุดิบ	varchar	
pr_zone	บริเวณที่ใช้จัดเก็บ	varchar	

ตารางที่ 3.3 ตารางพจนานุกรมข้อมูลของตาราง stock

Attribute	Description	type	Primary Key
s_pr_id	รหัสสินค้าและวัตถุดิบ	varchar	Y
s_quan	จำนวน	varchar	
s_min	ขั้นต่ำ	varchar	
s_type	ประเภทของสินค้าและวัตถุดิบ	varchar	

ส่วนของการซื้อขาย (Trade) ก็จะใช้ทฤษฎี Master-Detail แบ่งเป็นตาราง trade_master กับตาราง trade_detail โดยที่ตาราง trade_master จะเก็บข้อมูลหลักประเภทชื่อของลูกค้าหรือตัวแทนจำหน่ายโดยมี t_type เป็นตัวช่วยแยก และเก็บข้อมูลราคารวม กับจำนวนรวมรายการที่ซื้อขายครั้งนั้น แต่ว่าส่วน trade_detail จะเก็บข้อมูลปลีกย่อย เช่น ชื่อ , จำนวน เป็นต้น

ตารางที่ 3.4 ตาราง พจนานุกรมข้อมูลของตาราง trade_master

Attribute	Description	type	Primary Key
t_type	ประเภทการซื้อขาย	varchar	Y
t_idm	เลขที่การสั่งซื้อหรือจำหน่าย	int	Y
t_date	วันที่ซื้อขาย	varchar	
t_total	จำนวนรวมรายการที่ซื้อ-ขาย	int	
Attribute	Description	type	Primary Key
t_totalprice	ราคารวมที่ซื้อขาย	int	
t_contact	ชื่อลูกค้าหรือตัวแทนจำหน่าย	varchar	
t_tel	เบอร์ติดต่อ	varchar	
t_recieve	ผู้รับเงินหรือรับของ	varchar	

ตารางที่ 3.5 ตารางพจนานุกรมข้อมูลของตาราง trade_detail

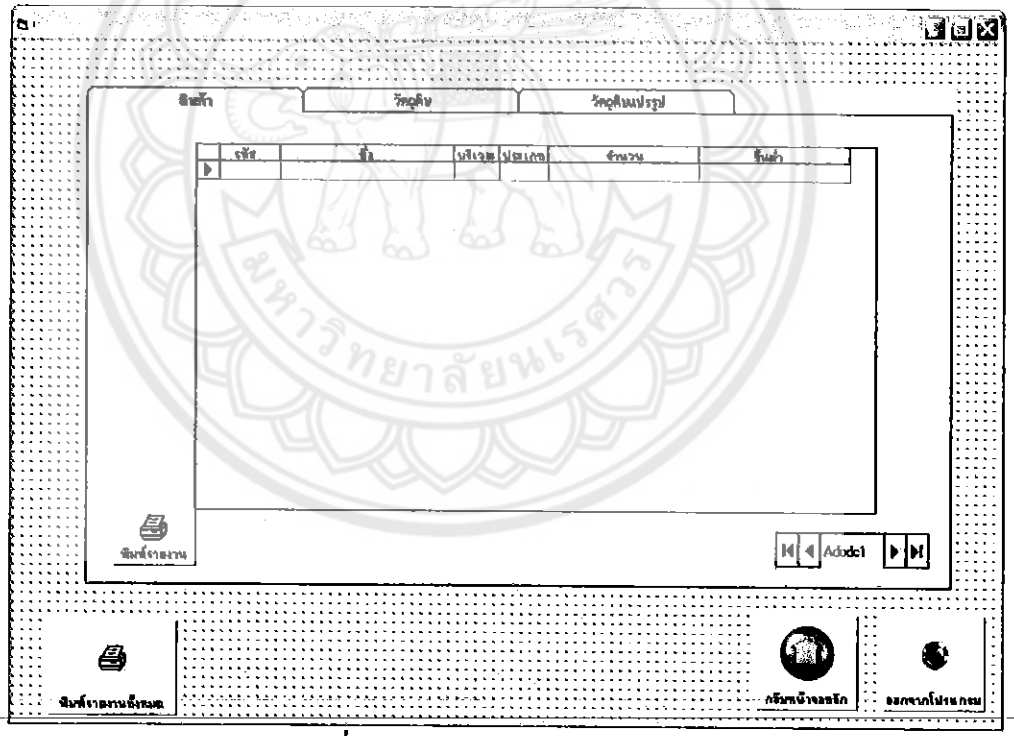
Attribute	Description	type	Primary Key
t_type	ประเภทการซื้อขาย	varchar	Y
t_idd	เลขที่การสั่งซื้อหรือจำหน่าย	int	Y
t_no	ลำดับที่รายการที่ซื้อขาย	int	Y
t_pr_name	ชื่อสินค้าและวัตถุดิบ	varchar	
t_pr_id	รหัสสินค้าและวัตถุดิบ	varchar	
t_cost	ราคา	int	
t_unit	หน่วยที่ใช้เรียก	varchar	
t_quan	จำนวน	int	

3.4 การออกแบบรูปแบบของหน้าตาโปรแกรม

โปรแกรมที่จัดทำขึ้นแบ่งออกเป็น 3 ส่วนใหญ่ๆ ได้แก่

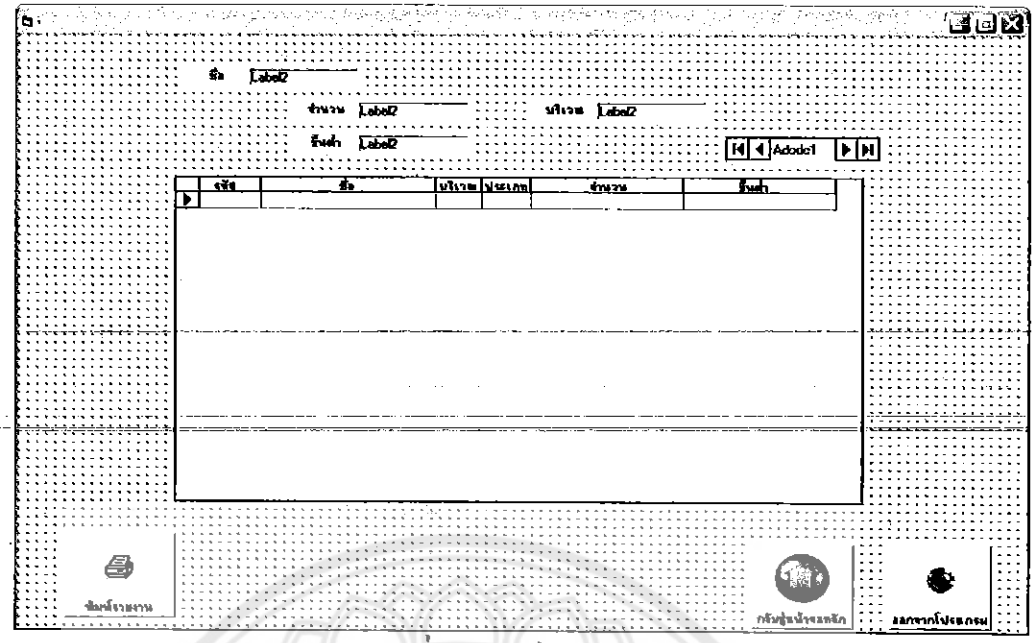
3.4.1 ข้อมูลสินค้าและวัตถุดิบ โดยที่ในส่วนของข้อมูลสินค้าและวัตถุดิบนั้นจะแสดงข้อมูลต่าง ๆ ที่เกี่ยวข้องไม่ว่าจะเป็นบริเวณที่จัดเก็บ หน่วยเรียกที่ใช้ของแต่ละชนิด จำนวนคงคลังที่มีเหลือ และจำนวนขั้นต่ำที่จัดเก็บ นอกจากนั้นแล้วหลังจากที่ได้ศึกษาข้อมูลจากโรงงานแล้วบวกกับได้ปรึกษากับทางโรงงานแล้วได้ทำการแยกประเภทของวัตถุดิบออกเป็นสองส่วน คือ วัตถุดิบและวัตถุดิบแปรรูป โดยที่วัตถุดิบ จะหมายถึงวัตถุดิบที่สั่งซื้อเข้ามาและจะนำเข้าสู่กระบวนการเพื่อแปรรูปก่อนนำไปประกอบกับส่วนอื่น และวัตถุดิบแปรรูปหมายถึงวัตถุดิบสั่งซื้อประเภทที่ได้มีการผ่านกระบวนการแปรรูปเป็นวัตถุดิบที่พร้อมจะนำเข้าสู่กระบวนการประกอบ โดยบางครั้งอาจเป็นจำพวกที่โรงงานสามารถผลิตได้เองแต่ในบางครั้งนั้นโรงงานเองไม่สามารถที่จะผลิตได้ทัน ดังนั้นจึงต้องสั่งซื้อรวมทั้งวัตถุดิบแปรรูปบางตัวก็เป็นประเภทที่โรงงานไม่สามารถที่จะแปรรูปเองได้จึงได้ทำการจัดแบ่งเอาไว้ดังที่กล่าวไว้ข้างต้น เพื่อให้เกิดการแตกต่างที่เห็นได้ชัดเจนทำให้การตรวจสอบเป็นไปได้โดยง่ายขึ้นและทางฝ่ายของโรงงานเองก็เป็นที่พอใจ

หน้าจอโปรแกรมหลังทำการออกแบบฟอร์มแสดงข้อมูลสินค้าและวัตถุดิบดังรูปที่ 3.6



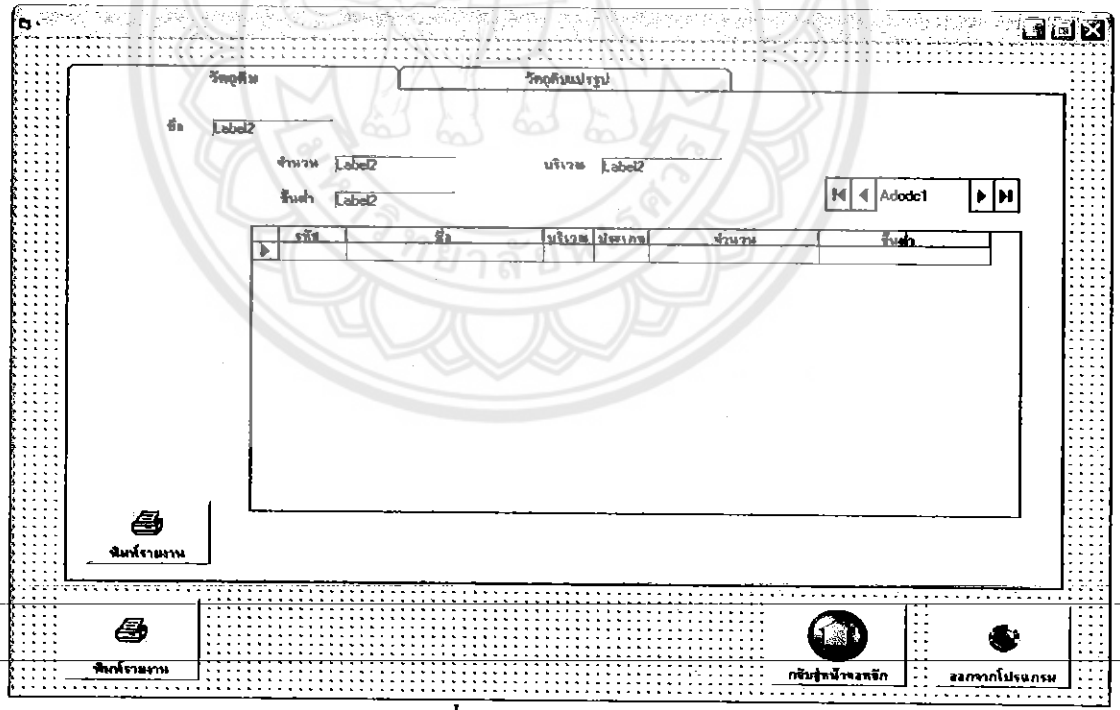
รูปที่ 3.6 ฟอร์มแสดงข้อมูลสินค้าและวัตถุดิบ

หน้าจอโปรแกรมในส่วนการออกแบบฟอร์มแสดงข้อมูลสินค้าหลังจากได้รับการจัดรูปแบบดังรูปที่ 3.7



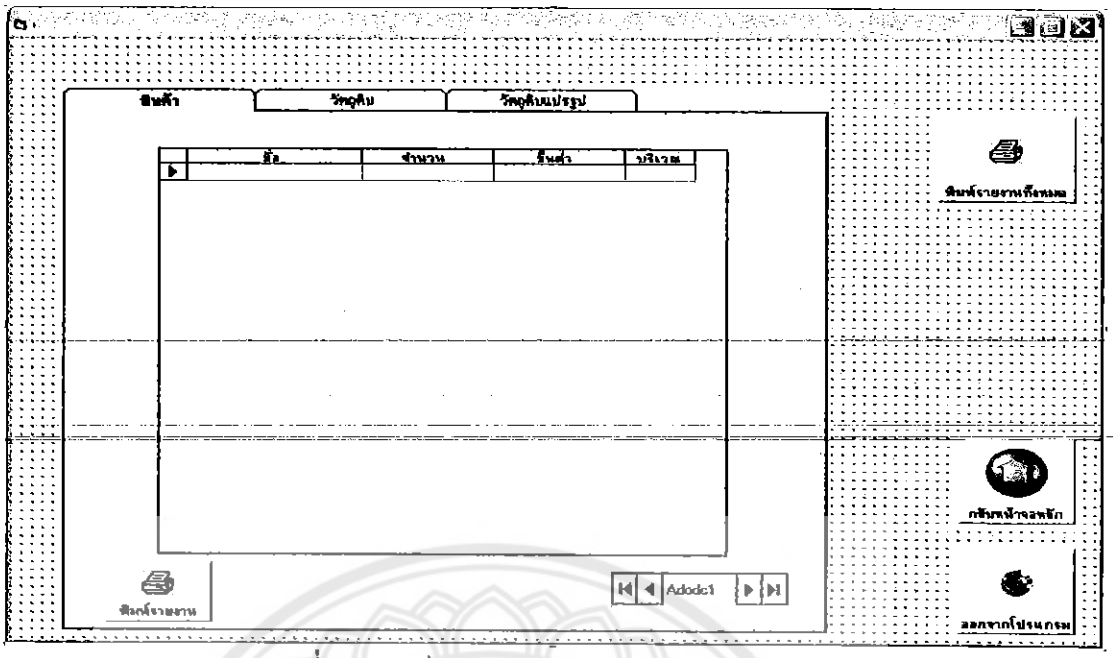
รูปที่ 3.7 ฟอร์มแสดงข้อมูลสินค้า

แสดงผลการออกแบบโปรแกรมส่วนแสดงข้อมูลวัตถุดิบดังรูปที่ 3.8



รูปที่ 3.8 ฟอร์มแสดงข้อมูลวัตถุดิบ

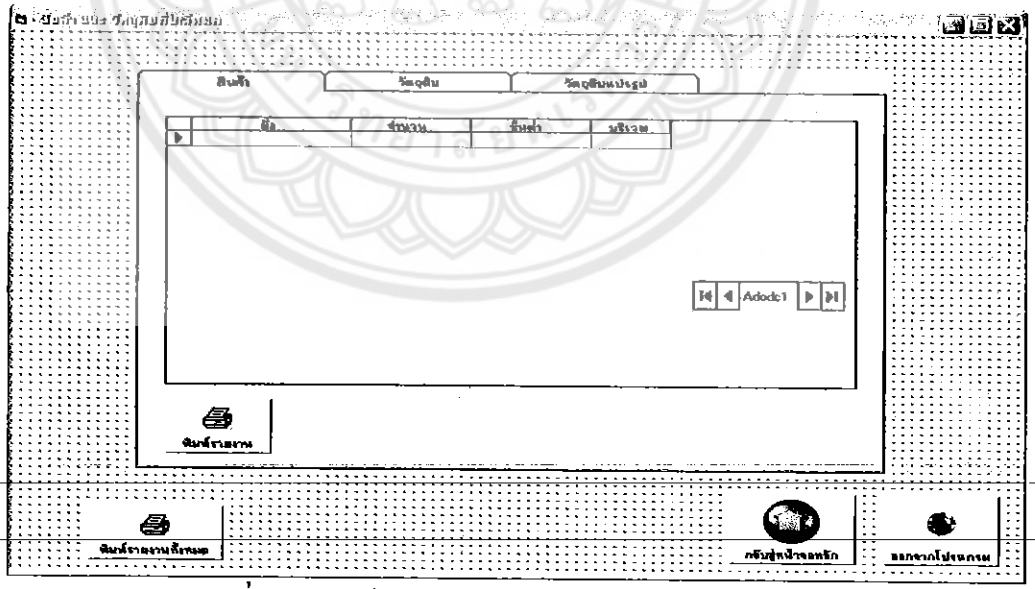
ในรูปที่ 3.9 จะเป็นการออกแบบฟอร์มเพื่อแสดงเฉพาะจำนวนสินค้าและวัตถุดิบ



รูปที่ 3.9 ฟอร์มแสดงเฉพาะจำนวนสินค้าและวัตถุดิบ

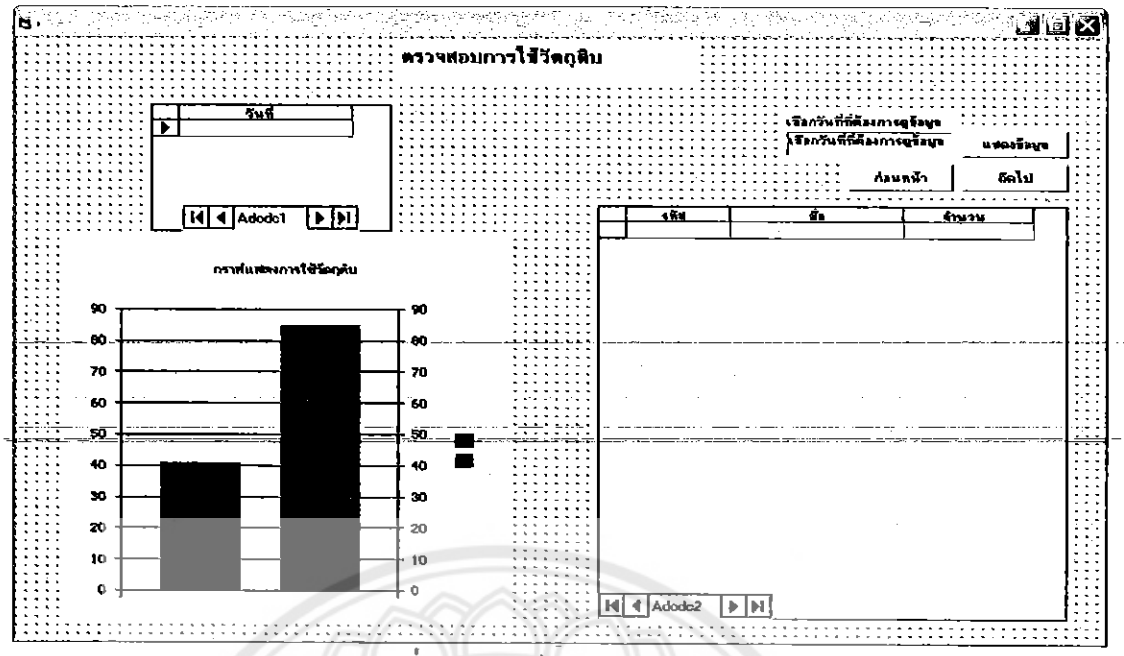
3.4.2 **ตรวจสอบการใช้วัตถุดิบ** ในส่วนนี้นั้นจะแยกออกเป็นการบันทึกการใช้วัตถุดิบและส่วนแสดงการใช้วัตถุดิบ

ผลการออกแบบฟอร์มเพื่อเตือนสินค้าและวัตถุดิบที่ใกล้หมดดังรูปที่ 3.10



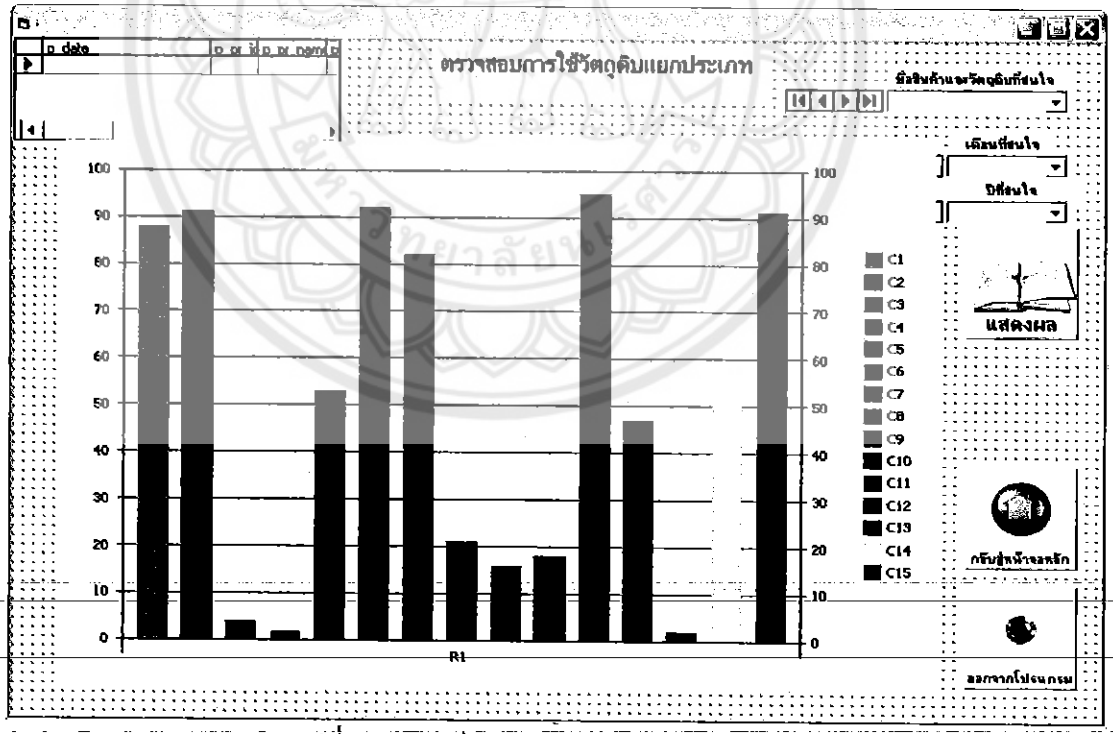
รูปที่ 3.10 ฟอร์มแสดงข้อมูลสินค้าและวัตถุดิบใกล้หมด

ในรูปที่ 3.11 เป็นผลที่ได้จากการออกแบบฟอร์มแสดงข้อมูลการใช้วัตถุดิบ



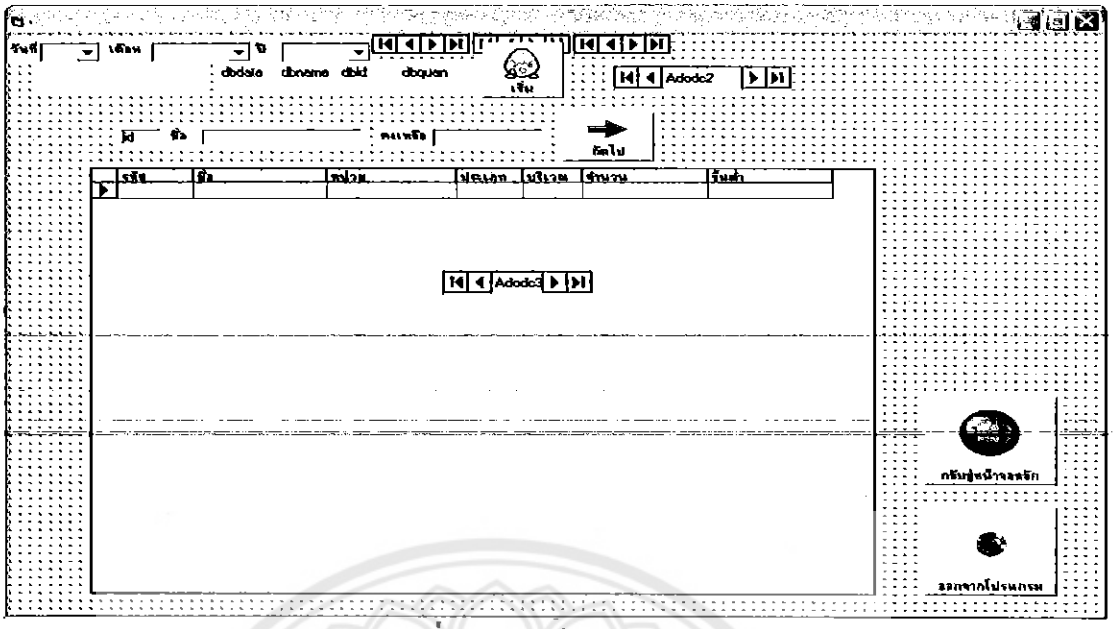
รูปที่ 3.11 ฟอร์มแสดงข้อมูลการใช้วัตถุดิบ

ทำการออกแบบฟอร์มดังรูปที่ 3.12 เพื่อแสดงข้อมูลแยกตามชื่อสินค้าและวัตถุดิบ



รูปที่ 3.12 ฟอร์มแสดงข้อมูลการใช้วัตถุดิบแยกประเภท

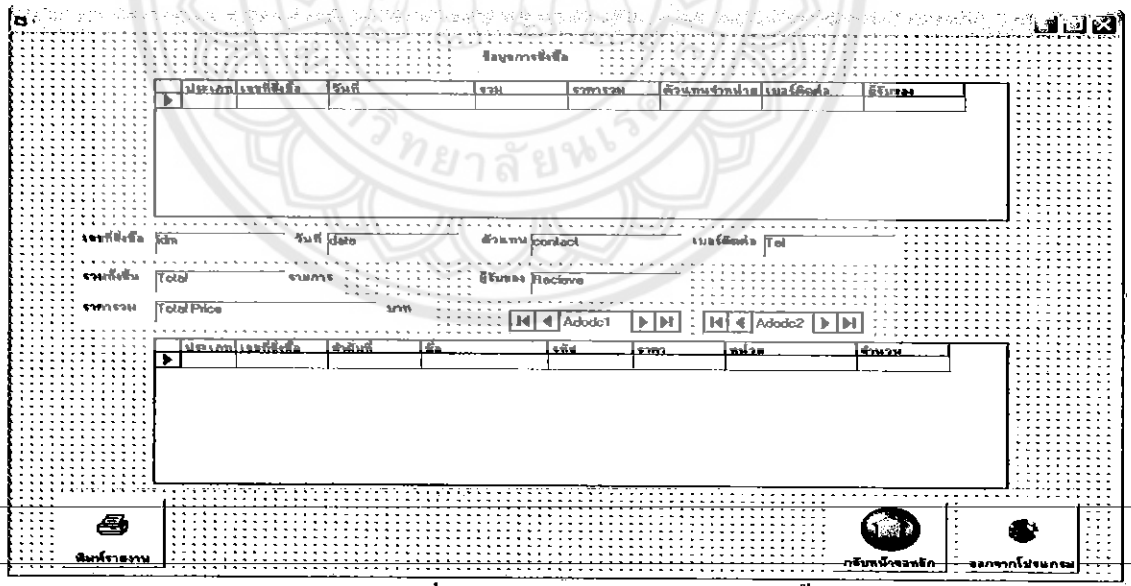
แบบฟอร์มในรูปที่ 3.13 เป็นผลจากการออกแบบเพื่อใช้บันทึกการใช้วัตถุดิบ



รูปที่ 3.13 ฟอร์มบันทึกการใช้วัตถุดิบ

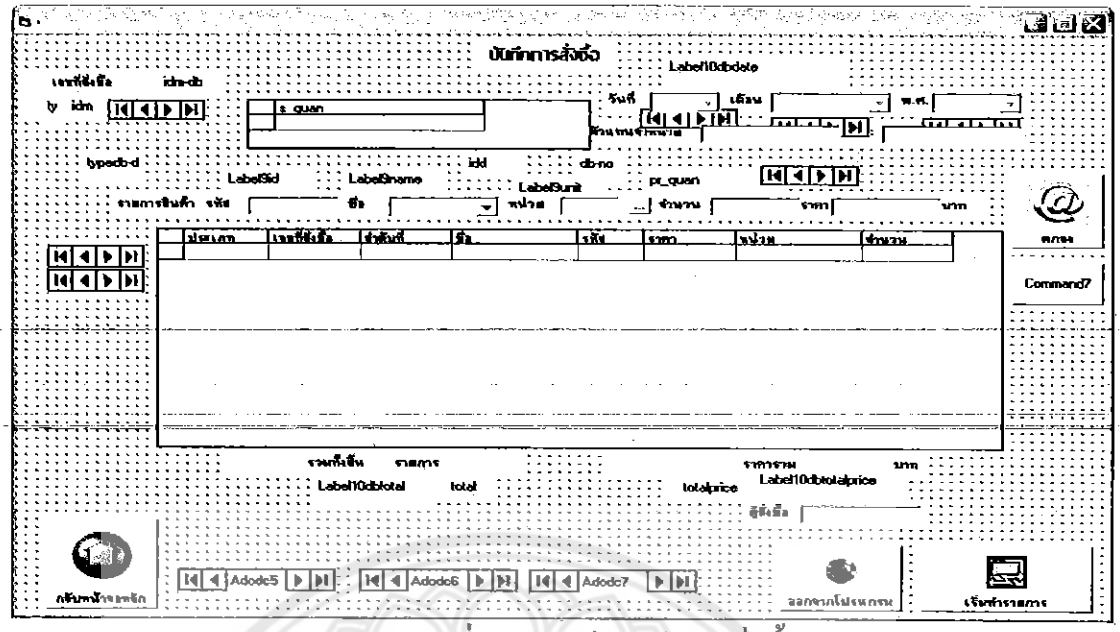
3.4.3 การสั่งซื้อและจำหน่าย แยกออกเป็นส่วนที่ทำการบันทึก**การสั่งซื้อและจำหน่าย** อีกส่วนคือส่วนที่ตรวจสอบข้อมูลการซื้อขาย

โดยในรูปที่ 3.14 เป็นการออกแบบฟอร์มแสดงข้อมูลการสั่งซื้อ



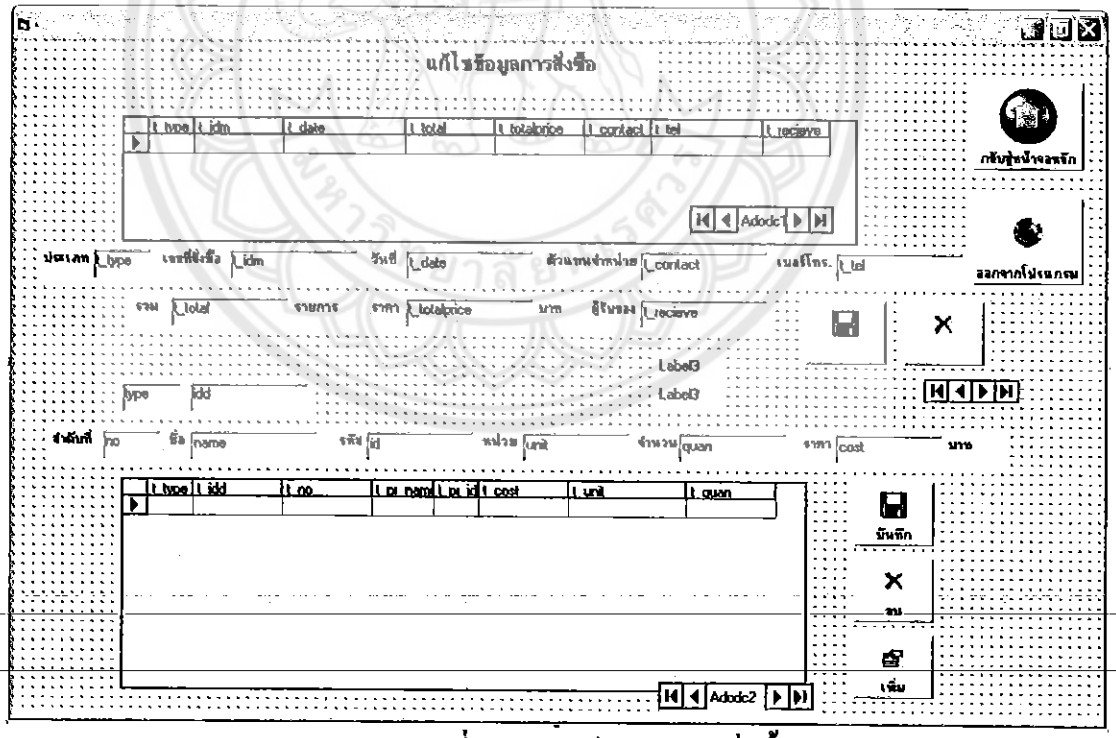
รูปที่ 3.14 ฟอร์มแสดงข้อมูลการสั่งซื้อ

ส่วนในรูปที่ 3.15 เป็นการออกแบบฟอร์มบันทึกข้อมูลการสั่งซื้อ



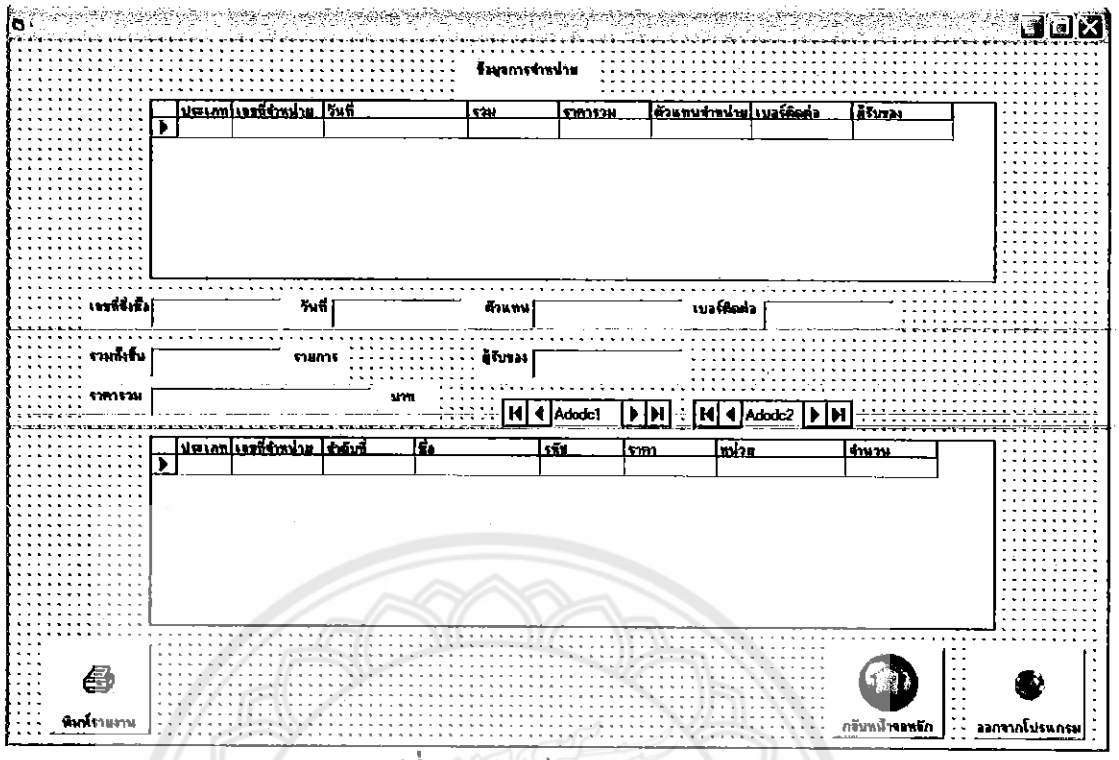
รูปที่ 3.15 ฟอร์มบันทึกการสั่งซื้อ

ส่วนของฟอร์มแก้ไขข้อมูลการสั่งซื้อเมื่อได้รับการออกแบบแล้วแสดงดังรูป 3.16



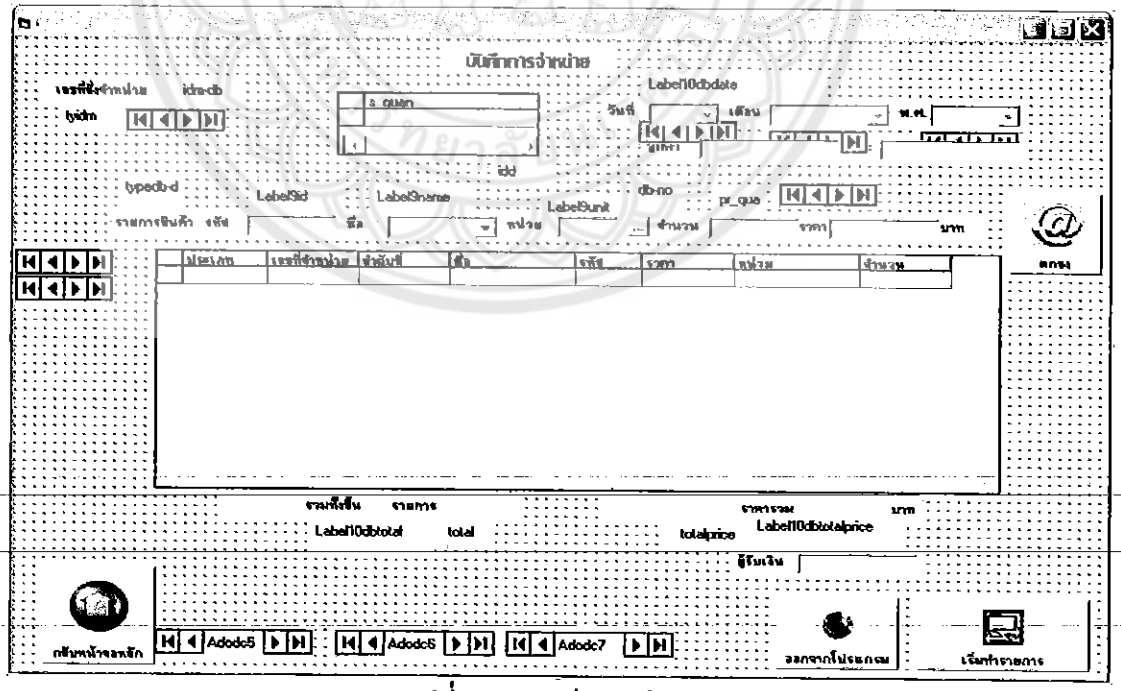
รูปที่ 3.16 ฟอร์มแก้ไขการสั่งซื้อ

ในรูปที่ 3.17 เป็นผลที่ได้จากการออกแบบฟอร์มเพื่อแสดงข้อมูลการจำหน่าย



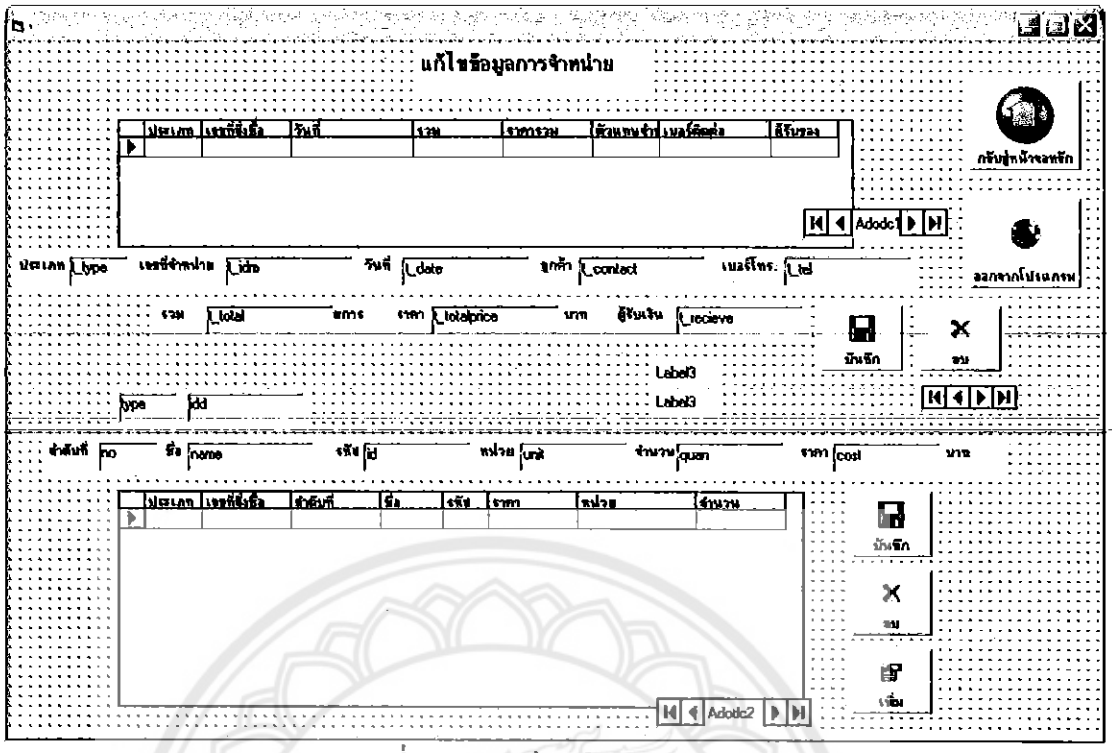
รูปที่ 3.17 ฟอร์มแสดงข้อมูลการจำหน่าย

แบบฟอร์มบันทึกข้อมูลการจำหน่ายเมื่อได้รับการออกแบบแล้วแสดงดังในรูปที่ 3.18



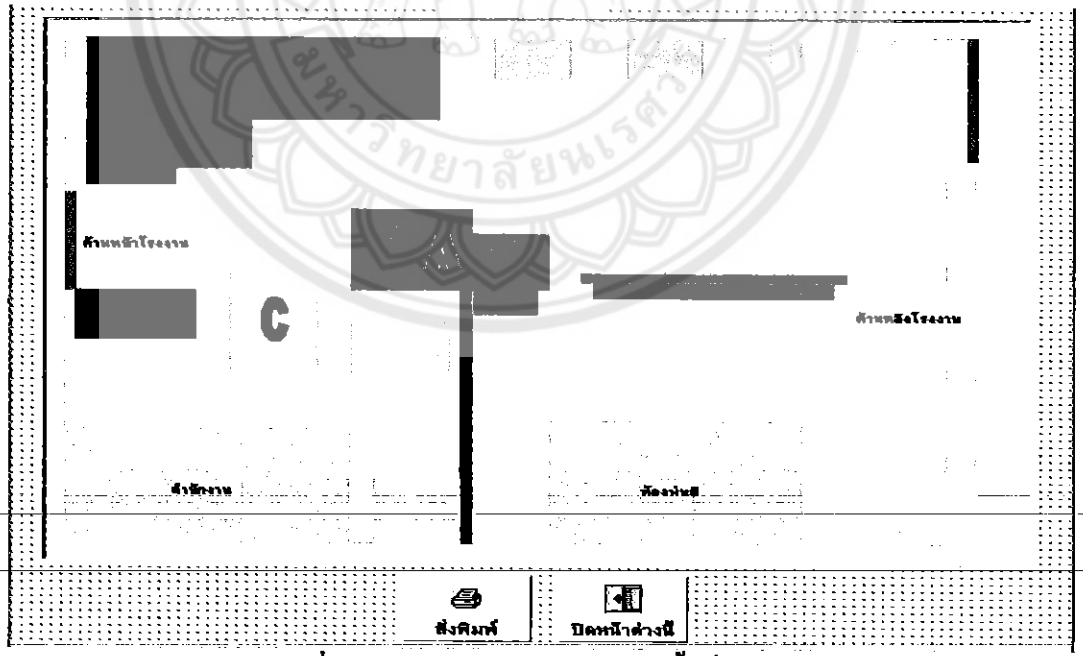
รูปที่ 3.18 ฟอร์มบันทึกการจำหน่าย

ในรูปที่ 3.19 เป็นรูปที่แสดงผลการออกแบบฟอร์มแก้ไขข้อมูลการจำหน่าย



รูปที่ 3.19 ฟอร์มแก้ไขการจำหน่าย

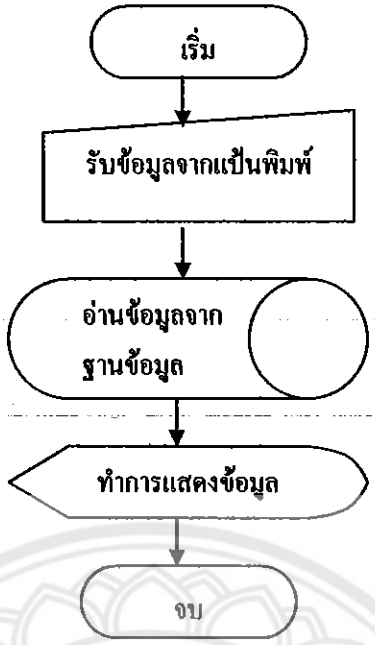
นอกจากนั้นแล้วยังมีส่วนที่แสดงแผนผังในโรงงานเพื่อให้ทราบว่าจัดเก็บที่ส่วนใด



รูปที่ 3.20 ฟอร์มแสดงการจัดแบ่งพื้นที่ในโรงงาน

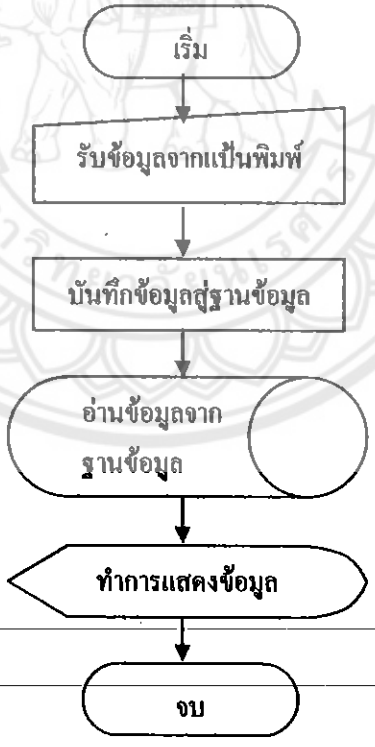
3.5 การออกแบบผังงานการทำงานของโปรแกรมในส่วนต่างๆ

จากรูปที่ 3.21 เป็นผังงานขั้นตอนการทำงานของโปรแกรมในส่วนแสดงข้อมูลทั่วไป



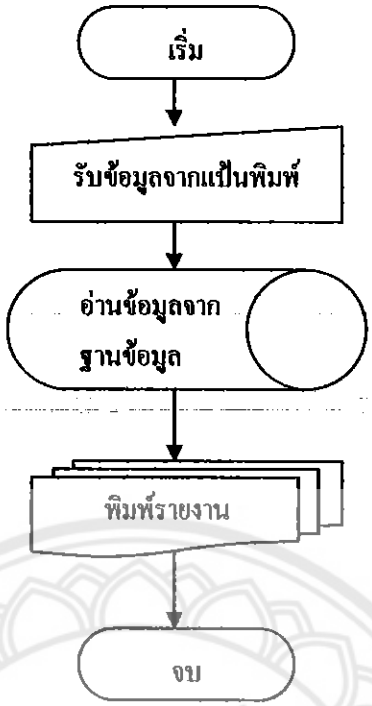
รูปที่ 3.21 ผังงานของโปรแกรมในส่วนแสดงข้อมูลทั่วไป

ในรูปที่ 3.22 แสดงขั้นตอนการทำงานของโปรแกรมในกรณีที่แสดงข้อมูลแบบมีเงื่อนไข



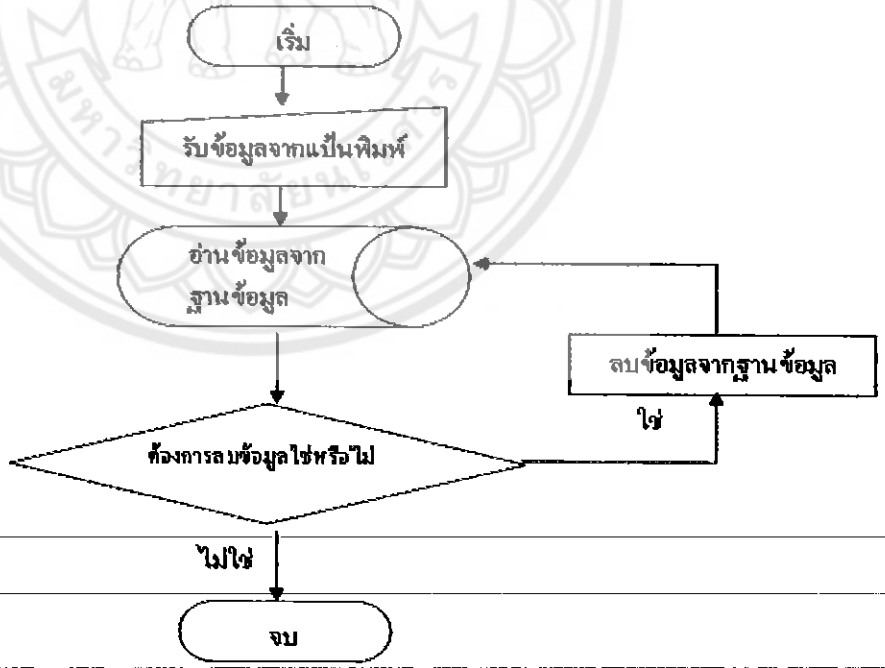
รูปที่ 3.22 ผังงานของโปรแกรมในส่วนแสดงข้อมูลแบบมีเงื่อนไข

ในกรณีของการสั่งพิมพ์รายงานนั้นจะมีขั้นตอนในการทำงานของโปรแกรมดังรูปที่ 3.23



รูปที่ 3.23 ผังงานของโปรแกรมในส่วนแสดงข้อมูลทั่วไปเพื่อพิมพ์รายงาน

ในส่วนของการทำการลบนั้นจะมีขั้นตอนในการดำเนินการดังในรูปที่ 3.24



รูปที่ 3.24 ผังงานของโปรแกรมในส่วนลบข้อมูลจากฐานข้อมูล

บทที่ 4

การทดสอบและวิเคราะห์การทำงาน

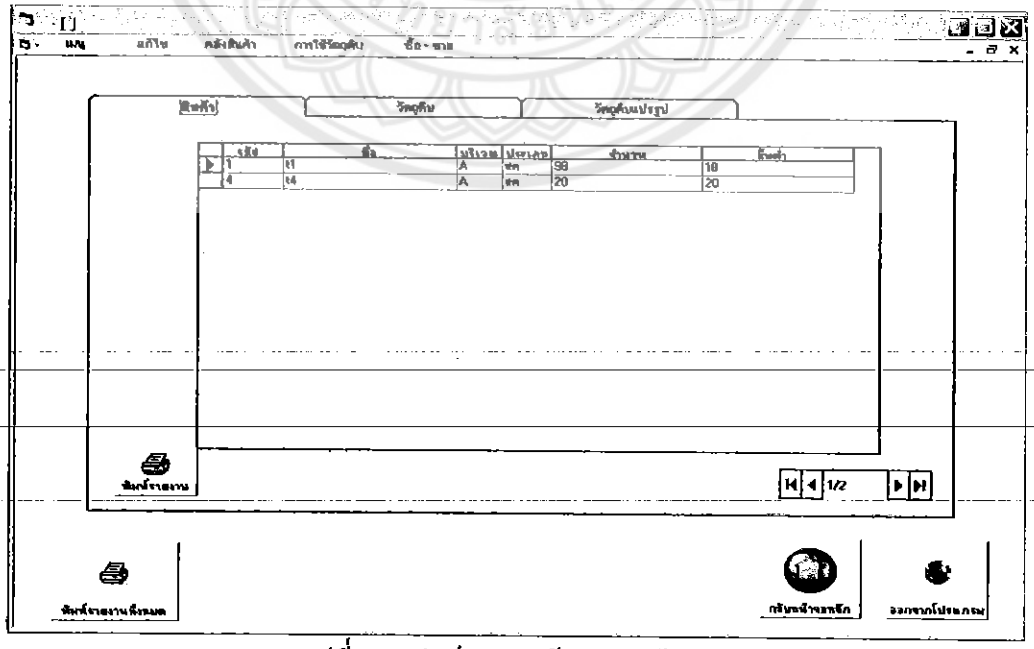
ในบทนี้จะกล่าวถึงการทดสอบและวิเคราะห์การใช้งานจริงของโปรแกรม ซึ่งโปรแกรมที่ได้จัดทำขึ้นนั้นได้ใช้โปรแกรม Visual Basic 6.0 ทำการติดต่อกับฐานข้อมูล เพื่อจะได้ทราบว่าสามารถทำงานได้ถูกต้องตามที่ออกแบบไว้หรือไม่ โดยการทดสอบโปรแกรมนั้นแบ่งออกเป็น 3 ส่วนได้แก่

1. ข้อมูลสินค้าและวัตถุดิบ
2. ตรวจสอบการใช้วัตถุดิบ
3. การสั่งซื้อและการจำหน่าย

4.1 ข้อมูลสินค้าและวัตถุดิบ

4.1.1 ข้อมูลสินค้าและวัตถุดิบ

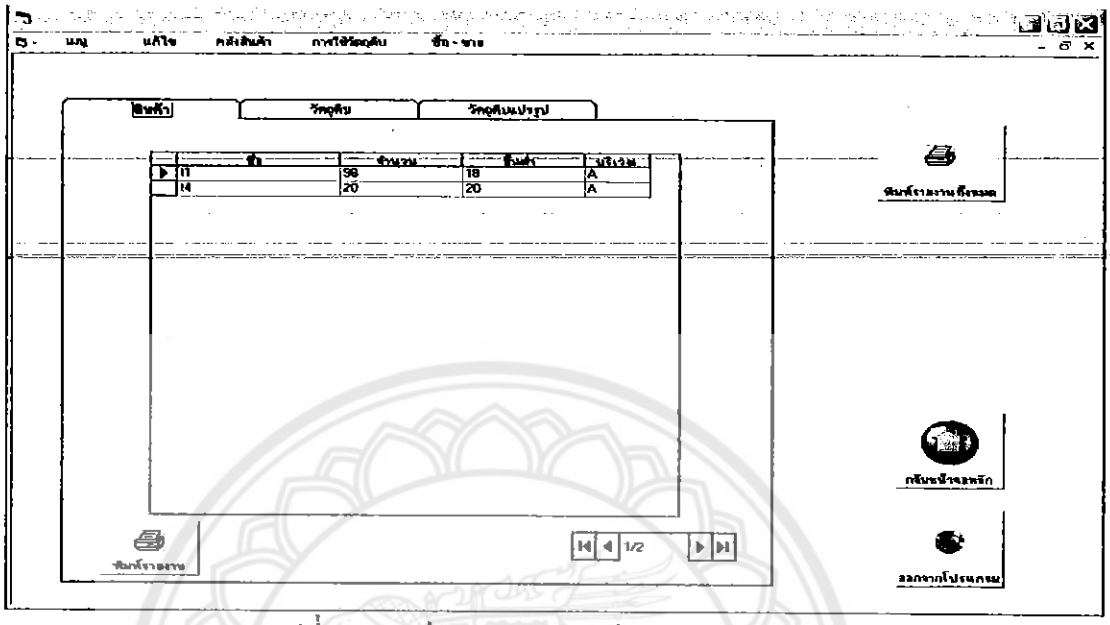
ฟอร์มแสดงข้อมูลสินค้าและวัตถุดิบจะเป็นส่วนที่แสดงให้ทราบถึงข้อมูลต่างๆ ของสินค้าและวัตถุดิบที่มีอยู่ทั้งหมดโดยจะแสดงรหัสและชื่อของสินค้าและวัตถุดิบ รวมถึงปริมาณของสินค้าและวัตถุดิบที่มีอยู่ทั้งหมดรวมทั้งจำนวนของสินค้าและวัตถุดิบขั้นต่ำที่จัดเก็บพร้อมกับบอกให้ทราบถึงบริเวณที่จัดเก็บ โดยในรูปที่ 4.1 จะเป็นผลที่ได้จากการรันฟอร์มแสดงข้อมูลสินค้าและวัตถุดิบ



รูปที่ 4.1 ฟอร์มแสดงข้อมูลสินค้าและวัตถุดิบ

4.1.2 ตรวจสอบสินค้าและวัตถุดิบคงคลัง

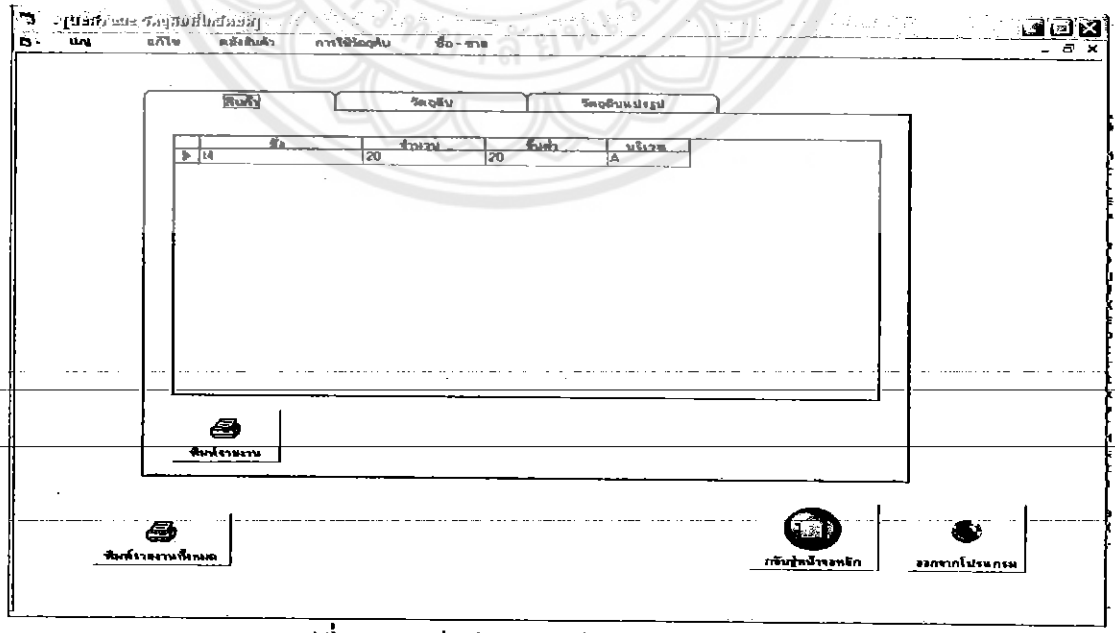
ฟอร์มตรวจสอบสินค้าและวัตถุดิบคงคลังจะแสดงเฉพาะจำนวนคงคลังเท่านั้นซึ่งในรูปที่ 4.2 คือผลที่ได้



รูปที่ 4.2 ฟอร์มตรวจสอบสินค้าและวัตถุดิบคงคลัง

4.1.3 ข้อมูลสินค้าและวัตถุดิบใกล้หมด

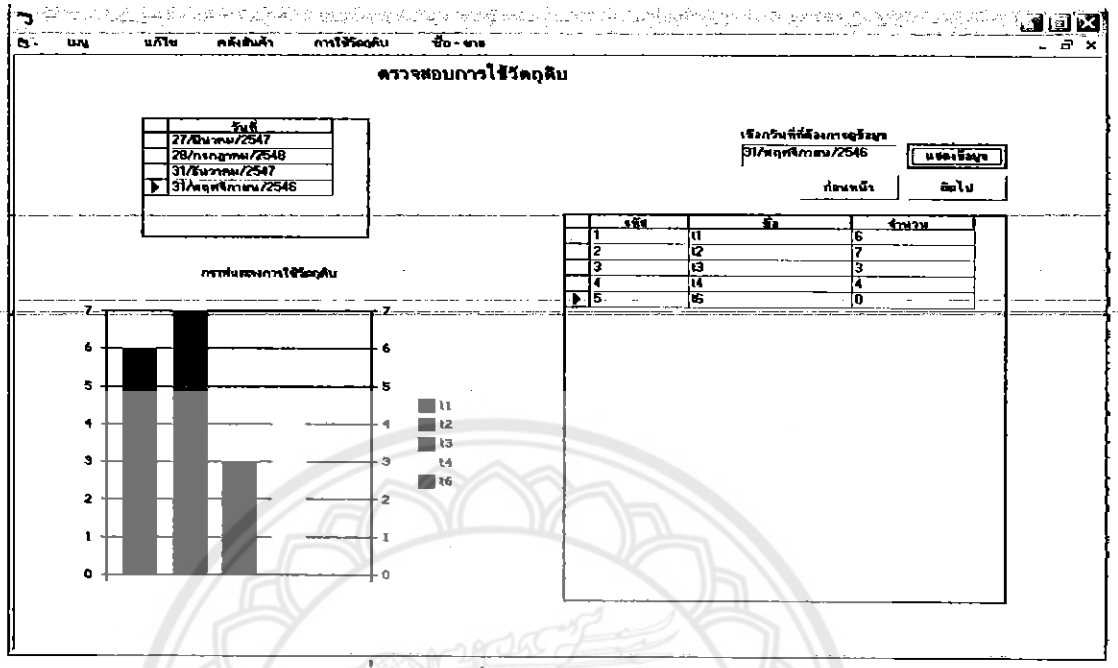
เมื่อสั่งรัน จะได้ผลดังในรูปที่ 4.3



รูปที่ 4.3 ฟอร์มข้อมูลสินค้าและวัตถุดิบใกล้หมด

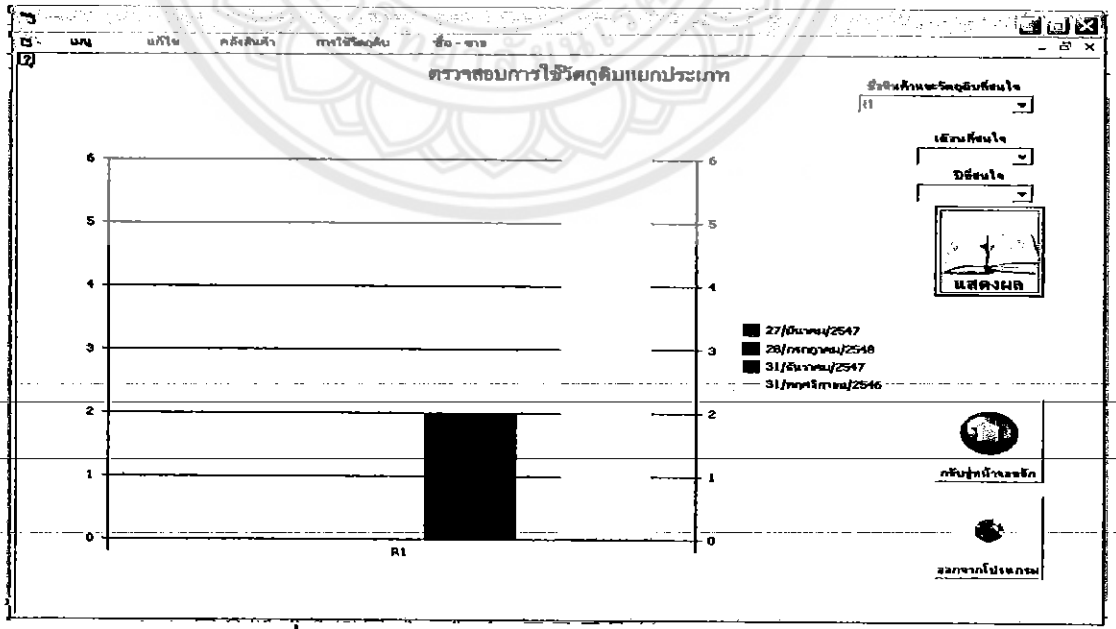
4.2 ตรวจสอบการใช้วัตถุดิบ

4.2.1 แสดงข้อมูลการใช้วัตถุดิบ โดยจะแสดงผลรันในรูปที่ 4.4



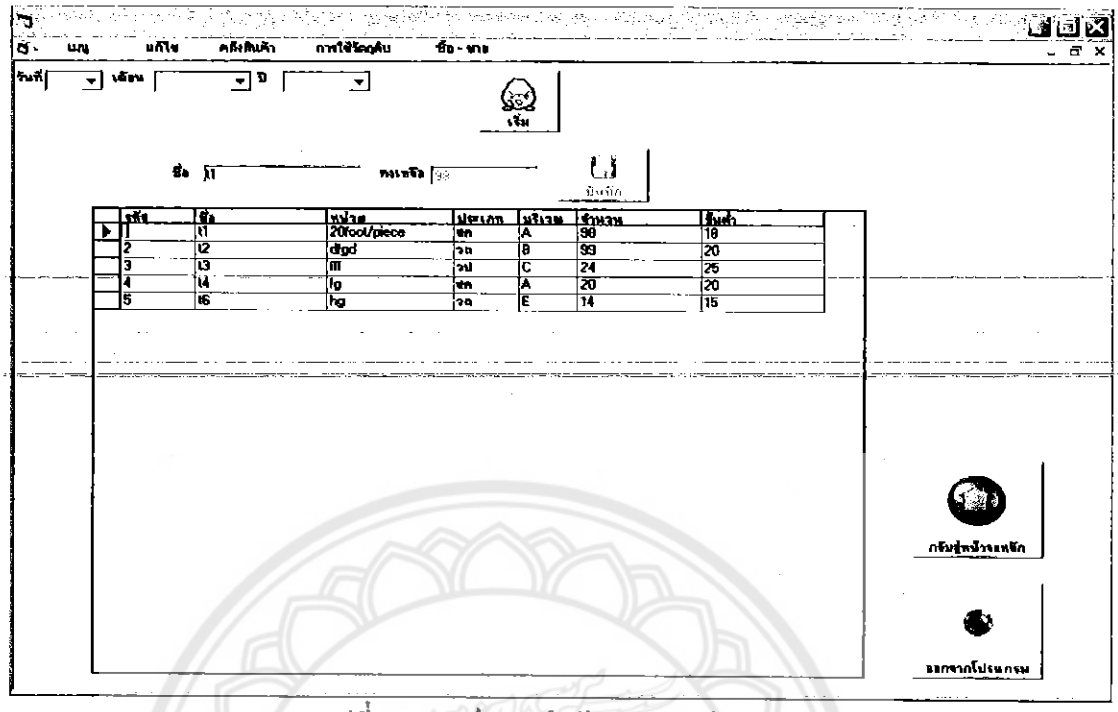
รูปที่ 4.4 ฟอร์มแสดงข้อมูลการใช้วัตถุดิบ

4.2.2 แสดงข้อมูลการใช้วัตถุดิบแยกตามชนิด เป็นการแสดงตามรายชื่อที่เลือก โดยผลที่ได้แสดงในรูปที่ 4.5



รูปที่ 4.5 ฟอร์มแสดงข้อมูลการใช้วัตถุดิบแยกตามชนิด

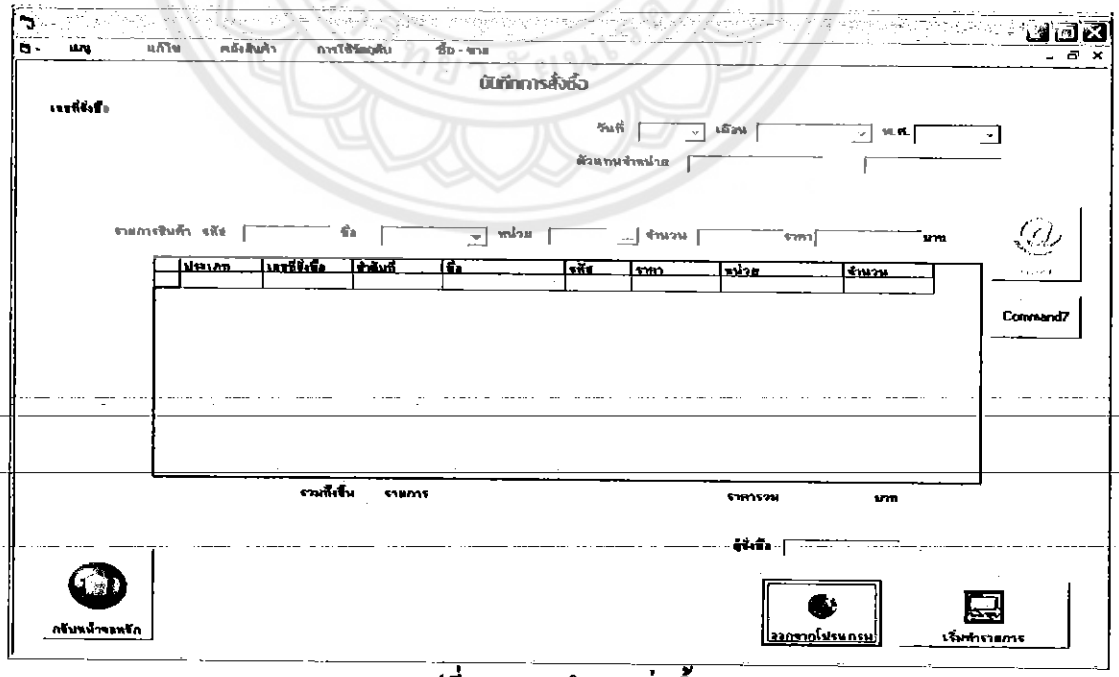
4.2.3 บันทึกข้อมูลการใช้ ในรูปที่ 4.6 เป็นการแสดงผลรันของฟอร์มบันทึกข้อมูลการใช้วัตถุดิบ



รูปที่ 4.6 ฟอร์มบันทึกข้อมูลการใช้วัตถุดิบ

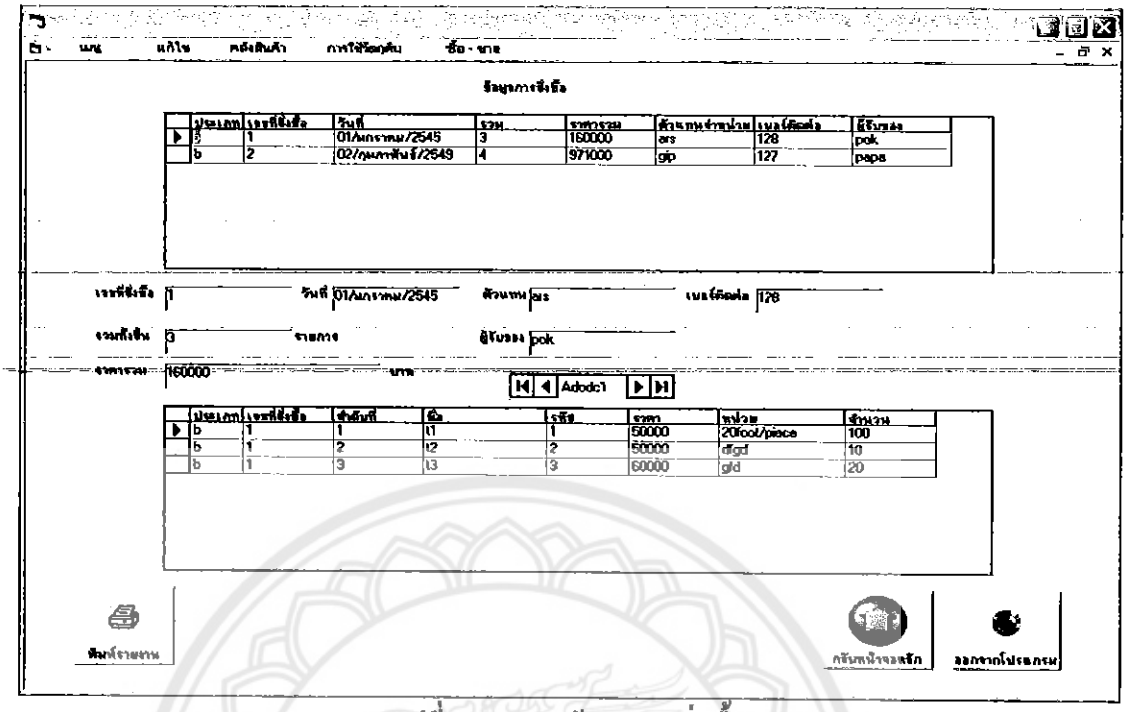
4.3 การสั่งซื้อและการจำหน่าย

4.3.1 **บันทึกการสั่งซื้อ** แสดงผลการรันฟอร์มบันทึกการสั่งซื้อดังรูปที่ 4.7



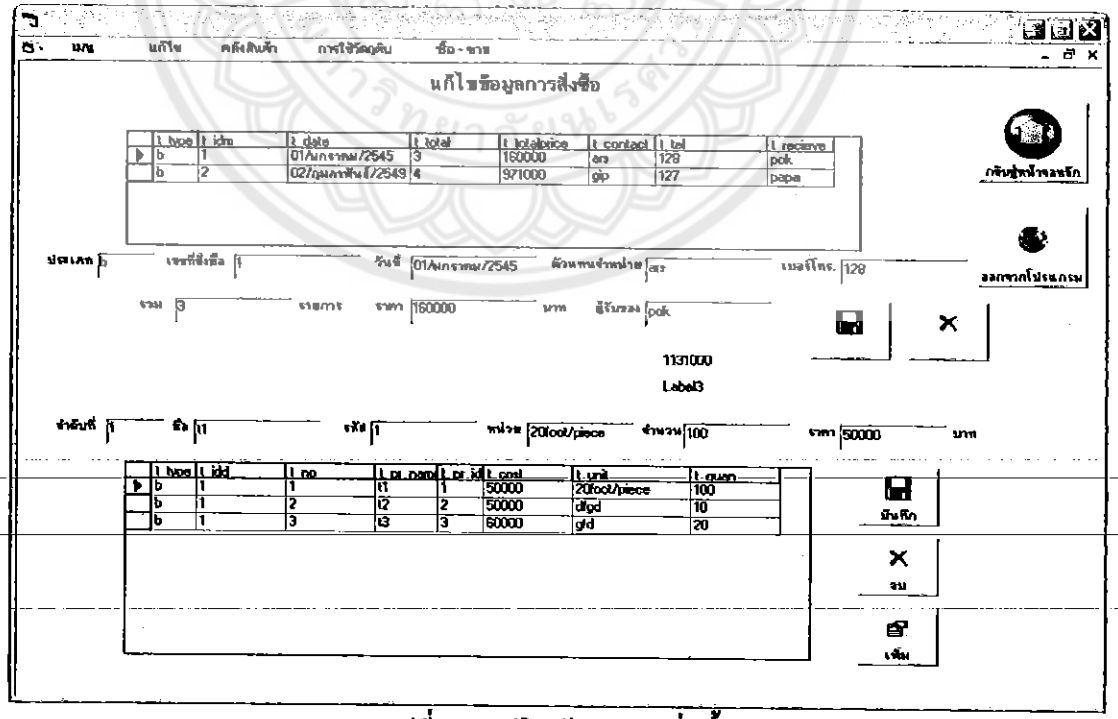
รูปที่ 4.7 บันทึกการสั่งซื้อ

4.3.2 แสดงข้อมูลการสั่งซื้อ เมื่อทำการรันฟอร์มข้อมูลการซื้อขายจะได้ผลดังรูปที่ 4.8



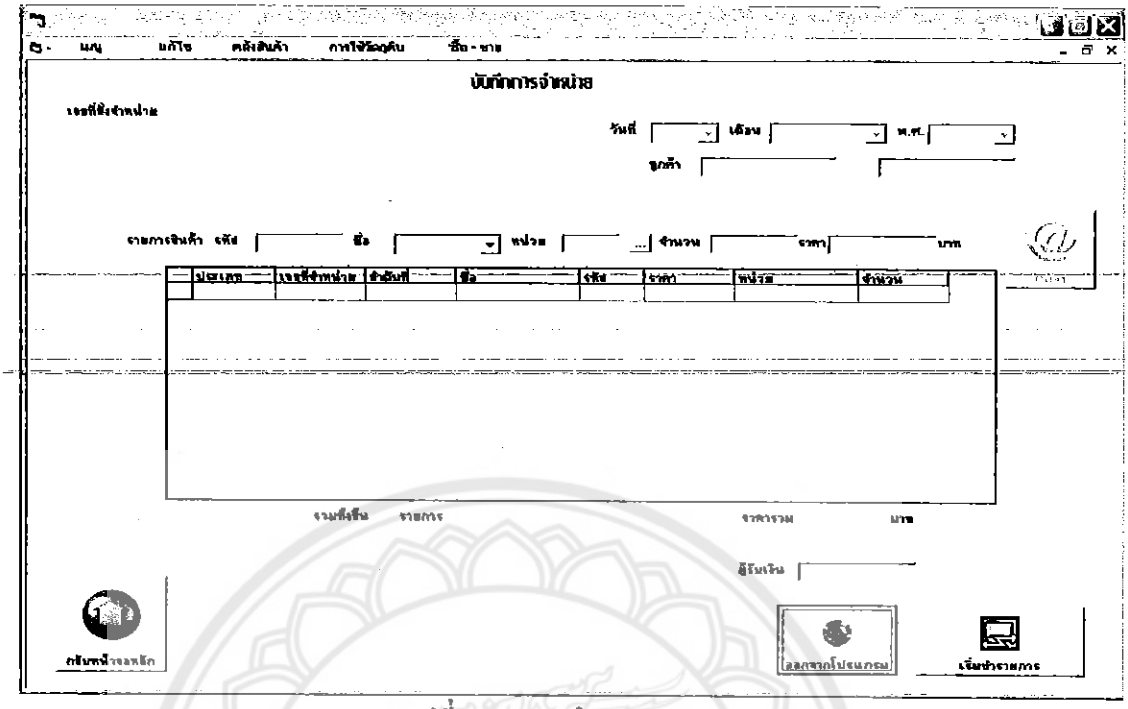
รูปที่ 4.8 แสดงข้อมูลการสั่งซื้อ

4.3.3 แก้ไขข้อมูลการสั่งซื้อ รูปที่ 4.9 เป็นผลที่ได้จากการรันฟอร์มแก้ไขข้อมูลการสั่งซื้อ



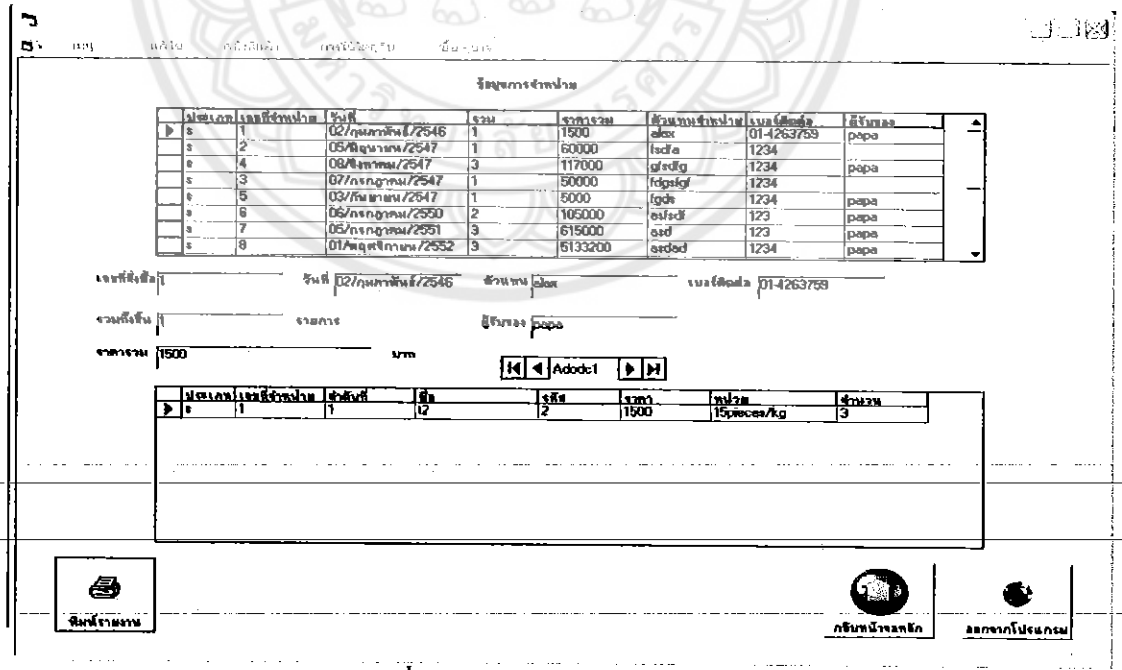
รูปที่ 4.9 แก้ไขข้อมูลการสั่งซื้อ

4.3.4 บันทึกการจำหน่าย เมื่อทำการรันฟอร์มบันทึกการจำหน่ายได้ผลดังรูปที่4.10



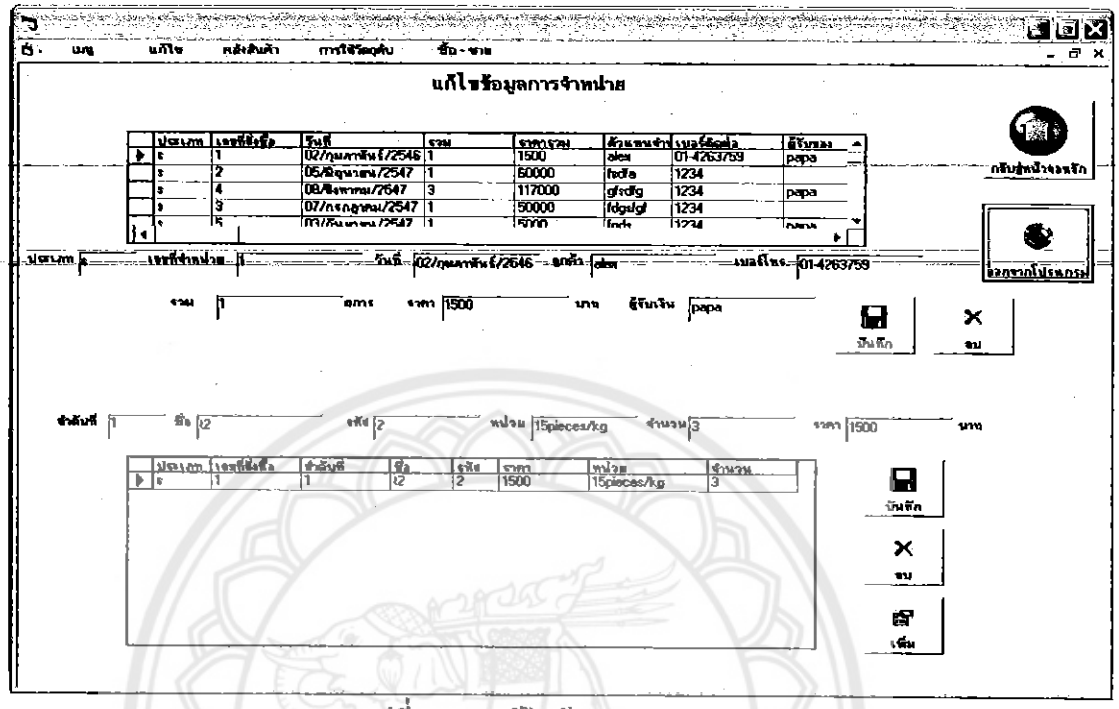
รูปที่ 4.10 บันทึกการจำหน่าย

4.3.5 แสดงข้อมูลการจำหน่าย รูปที่ 4.11 เป็นผลการรันฟอร์มแสดงข้อมูลการจำหน่าย



รูปที่ 4.11 แสดงข้อมูลการจำหน่าย

4.3.6 แก้ไขข้อมูลการจำหน่าย เมื่อทำการรันฟอร์มแก้ไขข้อมูลการจำหน่ายและทดลองใส่ค่าต่างๆ จะได้ผลดังรูป



รูปที่ 4.12 แก้ไขข้อมูลการจำหน่าย

จากผลการทดสอบโปรแกรมข้างต้นผลที่ได้เป็นที่น่าพอใจเพราะโปรแกรมสามารถทำงานได้ตามความต้องการที่ตั้งใจเอาไว้

บทที่ 5
สรุปผลดำเนินงานและข้อเสนอแนะ

ในบทนี้จะขอกล่าวถึงการสรุปผลการดำเนินงานที่ได้จากการศึกษาและดำเนินการเพื่อทำการจัดการระบบข้อมูลคลังสินค้าสำหรับโรงงานเกษตรบ้านกร่างจนได้โครงงานที่ผู้พัฒนาคิดว่ามีประสิทธิภาพในการจัดเก็บข้อมูลที่เป็นระบบและมีประสิทธิภาพมากยิ่งขึ้น

5.1 สรุปผลการดำเนินโครงงาน

จากการดำเนินงานการจัดการระบบข้อมูลคลังสินค้าสำหรับโรงงานเกษตรบ้านกร่าง มีวัตถุประสงค์เพื่อจัดทำ ระบบฐานข้อมูลระบบสินค้าคงคลังเพื่อให้มีประสิทธิภาพมากยิ่งขึ้น ได้แก่ การตรวจสอบเกี่ยวกับข้อมูลสินค้าและวัตถุดิบคงคลัง เพื่อเพิ่มประสิทธิภาพในการจัดเก็บข้อมูลของโรงงาน และพัฒนาให้โรงงานมีการจัดการภายในโรงงานที่มีประสิทธิภาพสูงสุดซึ่งได้ผลการดำเนินงานดังนี้

- สามารถตรวจสอบข้อมูลเกี่ยวกับวัตถุดิบและสินค้าคงคลังได้สะดวกขึ้น เนื่องจากการใช้เวลาในการจัดเก็บข้อมูลที่น้อยลง การตรวจสอบและค้นหาข้อมูลที่เร็วขึ้น อีกทั้งข้อมูลที่ได้นั้นยังมีความถูกต้องและเที่ยงตรงมากขึ้นกว่าเดิม
- สามารถนำข้อมูลที่ได้ทำการจัดเก็บไว้นั้นนำออกมาทำการประมวลผลเพื่อช่วยตัดสินใจทางด้านการผลิตและการสั่งซื้อวัตถุดิบได้

5.2 ข้อเสนอแนะ

ผู้ศึกษาโครงงานมีข้อเสนอแนะในการทำโครงงานดังต่อไปนี้

- จากการศึกษาโครงงานสามารถพัฒนาโปรแกรม การจัดการระบบข้อมูลคลังสินค้าสำหรับโรงงานเกษตรบ้านกร่างเพื่อให้ใช้ได้กับโรงงานอุตสาหกรรมขนาดย่อมแห่งอื่นๆที่มีระบบการจัดการที่มีความคล้ายคลึงกันโดยมีความแตกต่างไม่มากนักอาจสามารถปรับเปลี่ยนบางส่วน เนื่องจากโปรแกรมที่ได้จัดทำขึ้นนี้ได้จากการศึกษากระบวนการผลิต และการศึกษาปัญหา และข้อมูลต่างๆ ล้วนมาจากภายในโรงงานเกษตรบ้านกร่างโดยนำข้อมูลที่ได้มาประกอบการเขียนโปรแกรมเพื่อแก้ปัญหาที่เกิดขึ้นจริงและตรงตามความต้องการของผู้ประกอบการ
- เนื่องจากโปรแกรมที่จัดทำขึ้นมีความเหมาะสมและใช้ได้กับการทำงานของโรงงานขนาดย่อมในปัจจุบันเท่านั้น ซึ่งในอนาคตหากโรงงานมีการเปลี่ยนแปลงกระบวนการผลิตให้ซับซ้อนมากยิ่งขึ้นรวมทั้งการจัดการที่แตกต่างออกไปจากปัจจุบัน เพราะโปรแกรมที่ได้จัดทำขึ้นใน

ครั้งนี้นั้นไม่สามารถรองรับการทำงานของระบบได้ดีเท่าที่ควรดังนั้นจึงควรที่จะมีการพัฒนาโปรแกรมการจัดการระบบข้อมูลคลังสินค้าสำหรับโรงงานเกษตรบ้านกร่างอย่างต่อเนื่องเพื่อให้ทันสมัยและใช้งานได้อย่างมีประสิทธิภาพอยู่ตลอดเวลา

- เนื่องจากการที่โรงงานได้มีการนำระบบคอมพิวเตอร์เข้ามาใช้งานอย่างจริงจังเป็นครั้งแรกซึ่งถือเป็นของใหม่สำหรับบุคลากรที่ทำงานอยู่เดิมดังนั้นจึงควรที่จะมีการอบรมให้ความรู้เกี่ยวกับคอมพิวเตอร์และการใช้งานโปรแกรมให้กับบุคลากรเพื่อที่จะได้บุคลากรที่มีคุณภาพมากยิ่งขึ้นและจะส่งผลให้การทำงานและใช้งานโปรแกรมที่จัดทำมีประสิทธิภาพมากยิ่งขึ้นตามไปด้วย
- โปรแกรมการจัดการระบบข้อมูลคลังสินค้าสำหรับโรงงานเกษตรบ้านกร่างนั้นยังมีข้อจำกัดอยู่ที่จัดการดูแลวัตถุดิบและสินค้าคงคลังเพียงเท่านั้นยังไม่ได้ครอบคลุมในส่วนของการพยากรณ์และวางแผนการผลิต ซึ่งหากเป็นไปได้สามารถที่จะพัฒนาต่อไปให้มีประสิทธิภาพมากขึ้นได้ในอนาคตโดยเพิ่มเติมส่วนช่วยพยากรณ์และวางแผนการผลิตเข้ามาและอาจมีการเพิ่มในส่วนของการดูแลในเรื่องของข้อมูลลูกจ้างการดูแลเรื่องเงินเดือน เป็นต้น

เอกสารอ้างอิง

[1] กิตติ ภักดีวัฒนกุล , จำลอง ครูอุตสาหะ. Visual Basic 6 ฉบับโปรแกรมเมอร์. พิมพ์ครั้งที่ 2. กรุงเทพฯ. ไทยเจริญการพิมพ์. 2542.

[2] กิตติ ภักดีวัฒนกุล , จำลอง ครูอุตสาหะ. Visual Basic 6 ฉบับฐานข้อมูล. พิมพ์ครั้งที่ 4. กรุงเทพฯ. ไทยเจริญการพิมพ์. 2544.

[3] นิภาภรณ์ คำเจริญ. เรียนรู้การใช้งาน Microsoft Access 2000. กรุงเทพฯ. เอส.พี.ซี. บุคส์. 2543.

[4] รศ. พิภพ ลลิตาภรณ์. การบริหารของคงคลังระบบ MRP และ ROP. พิมพ์ครั้งที่ 3. กรุงเทพฯ. สำนักพิมพ์ ส.ส.ท. 2543.

[5] รศ. ระพีพรรณ พิริยะกุล. การวิเคราะห์และออกแบบระบบ. พิมพ์ครั้งที่ 3. กรุงเทพฯ. สำนักพิมพ์มหาวิทยาลัยรามคำแหง 2543.

[6] สัจจะ จรัสรุ่งรวีวร. คู่มือการพัฒนาแอพพลิเคชั่นด้วย Visual Basic 6.0. กรุงเทพฯ. ด่านสุทธาการพิมพ์. 2542.

[7] สัจจะ จรัสรุ่งรวีวร. คู่มือการเขียนโปรแกรมและใช้งาน Visual Basic 6.0. กรุงเทพฯ. ด่านสุทธาการพิมพ์. 2544.

[8] ศุภชัย สมพานิช. Database Programming กับ Visual Basic ฉบับมืออาชีพ. กรุงเทพฯ. บริษัท เอช เอ็น กรุ๊ป จำกัด. 2543.

[9] Rebecca Riordan. การออกแบบระบบฐานข้อมูลเชิงสัมพันธ์. แปลจาก Designing Relational Database Systems. โดย นอ. อโณทัย นอบไทย. กรุงเทพฯ. สำนักพิมพ์สามย่าน. 2544.

ประวัติผู้เขียนโครงงาน

ชื่อ นายวุฒิพงษ์ ปิ่นนาค

ภูมิลำเนา 190 หมู่ 4 ตำบลวังกะพี้ อำเภอเมือง จังหวัดอุตรดิตถ์

ประวัติการศึกษา

- จบระดับมัธยมศึกษาตอนปลายจากโรงเรียนอุตรดิตถ์
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4
  สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
  มหาวิทยาลัยนเรศวร

e-mail : mommam_tao_@hotmail.com

ชื่อ นายอัตถสิทธิ์ เพียรวงศ์

ภูมิลำเนา 36 หมู่ 2 ตำบลแม่ปืม อำเภอเมือง จังหวัดพะเยา

ประวัติการศึกษา

- จบระดับมัธยมศึกษาตอนปลายจากโรงเรียนแม่ใจวิทยาคม
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4
  สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
  มหาวิทยาลัยนเรศวร

e-mail : shadow_v@hotmail.com