บทที่ 4

การออกแบบและพัฒนาโปรแกรม

ในบทนี้จะกล่าวถึงการออกแบบและพัฒนาโปรแกรมคอมพิวเตอร์ ซึ่งได้นำเสนอเป็น 4 ส่วนคือ

4.1 บทนำ

4.2 โครงสร้างของโปรแกรม

4.3 ระบบการทำงานของโปรแกรม

4.4 การพัฒนาโปรแกรม

4.5 เครื่องมือและอุปกรณ์ที่ใช้ในการพัฒนาโปรแกรม

4.1 บทนำ

ปัจจุบัน คอมพิวเตอร์ได้เข้ามามีบทบาทต่อชีวิตประจำวันของคนเราเป็นอย่างมาก โดยเฉพาะการทำงาน เกือบทุกหน่วยงานก็จะใช้คอมพิวเตอร์เข้ามาช่วยในการทำงาน การใช้คอมพิวเตอร์ในปัจจุบันถึงแม้ว่าจะได้รับความนิยมอย่างสูง แต่ก็มีปัญหาอยู่ว่าโปรแกรมคอมพิวเตอร์หรือ Software ส่วนใหญ่เป็นของต่างประเทศและมีราคาแพง

การพัฒนาโปรแกรมคอมพิวเตอร์ขึ้นมาใช้งานในหน่วยงานของตน เพื่อให้เข้ากับระบบงานของตนเองมากที่สุด จึงเป็นสิ่งที่จำเป็น ซึ่งจะทำให้ประหยัดงบประมาณในการซื้อโปรแกรมจากต่างประเทศและยังได้โปรแกรมที่ตรงกับความต้องการของหน่วยงานมากที่สุด

ดังนั้นผู้วิจัยจึงได้คิดพัฒนาโปรแกรม RMS (Rural **R**oad Database **M**anagement **S**ystem for Tambon Administration Organization) ขึ้นมาสำหรับองค์การบริหารส่วนตำบล เพื่อใช้ในการจัดการฐานข้อมูลทางหลวงชนบทที่ได้ก่อสร้างไปแล้วและที่จะก่อสร้างในอนาคต รวมทั้งทางหลวงชนบทที่จะรับถ่ายโอนมาจากหน่วยงานอื่น เช่น กรมทางหลวงชนบท กรมทางหลวง เป็นต้น โดยใช้โปรแกรม RMS เป็นเครื่องมือในการรวบรวมและจัดเก็บข้อมูลทั่วไป และข้อมูลทางกายภาพของทางหลวงชนบท เพื่อจัดทำเป็นทะเบียนประวัติ การจดทะเบียนทางหลวงชนบท การกำกับดูแลทางหลวงชนบทให้เป็นไปตามกฎหมาย เป็นต้น นอกจากนี้ยังสามารถนำฐานข้อมูลทางหลวงชนบทที่ได้ไปเป็นข้อมูลในการศึกษาวิจัยเรื่อง ที่เกี่ยวข้องกับทางหลวงชนบทต่อไปในอนาคตได้ เช่น การศึกษาอุบัติเหตุบนทางหลวงชนบท การประเมินสภาพผิวทางหลวงชนบท เป็นต้น

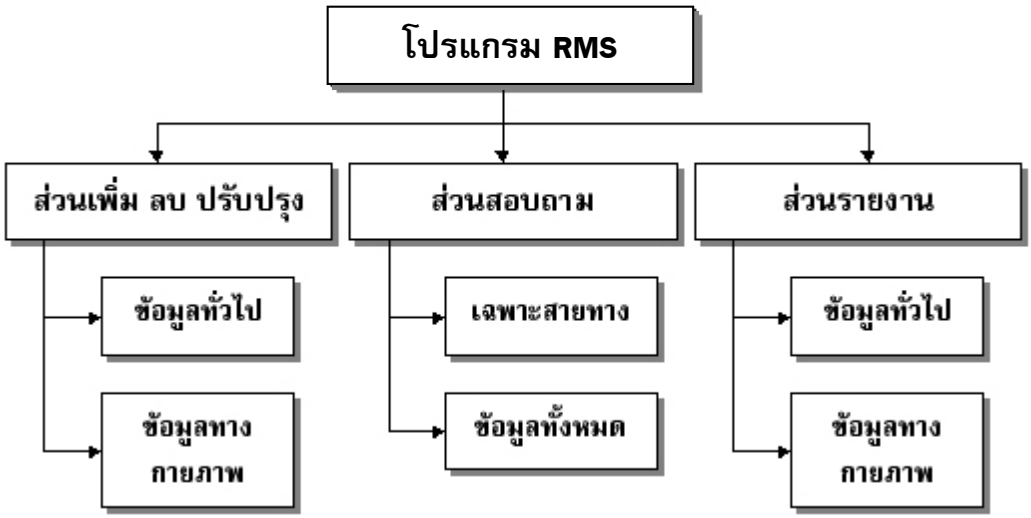
4.2 โครงสร้างของโปรแกรม RMS

โครงสร้างโปรแกรม RMS ประกอบด้วย 3 ส่วนหลักคือ ส่วนเพิ่ม ลบ และ ปรับปรุง ส่วนสอบถาม และส่วนรายงาน โดยแต่ละส่วนประกอบด้วยส่วนย่อยลงไปอีก (ภาพประกอบที่ 4.1)

ส่วนเพิ่ม ลบ และปรับปรุงข้อมูล เป็นส่วนแรกของโปรแกรมก่อนที่จะทำงานในส่วนอื่นๆ ผู้ใช้จะต้องเพิ่มข้อมูลทั่วไปและข้อมูลทางกายภาพของทางหลวงชนบทก่อน หลังจากนั้นจึงจะสามารถทำการลบหรือปรับปรุงข้อมูลต่างๆ ได้

ส่วนสอบถาม เป็นส่วนที่เรียกดูข้อมูลต่างๆ ในฐานข้อมูลที่ผู้ใช้ต้องการ โดยที่ผู้ใช้ไม่จำเป็นต้องรู้จักโครงสร้างของภาษา SQL ซึ่งเป็นภาษาที่ใช้เรียกข้อมูลขึ้นมาแสดง ส่วนสอบถามแบ่งเป็น 2 ส่วนย่อยคือ ส่วนสอบถามข้อมูลเฉพาะสายทาง เป็นส่วนสอบถามที่เรียกดูข้อมูลต่างๆ ของทางหลวงชนบทีละสายทาง และส่วนสอบถามข้อมูลทั้งหมด เป็นส่วนสอบถามที่เรียกดูข้อมูลต่างๆ ของทางหลวงชนบททุกสาย

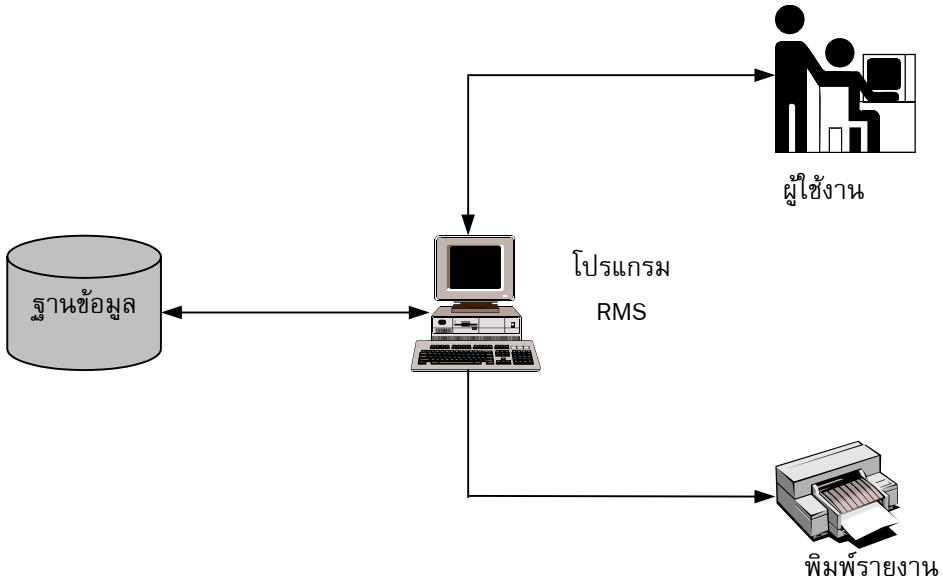
ส่วนสุดท้าย เป็นส่วนรายงานผลออกทางเครื่องพิมพ์ ซึ่งรายละเอียดทั้งสามส่วนจะกล่าวถึงในบทที่ 5 ผลลัพธ์ของโปรแกรม



ภาพประกอบที่ 4.1 โครงสร้างของโปรแกรม RMS

4.3 ระบบการทำงานของโปรแกรม

โปรแกรม RMS จะเป็นตัวกลางในการจัดการฐานข้อมูลทางหลวงชนบท (Rural Road Database Management) ระหว่างผู้ใช้กับฐานข้อมูล โดยโปรแกรม RMS จะทำหน้าที่นำเข้าข้อมูลเพื่อจัดเก็บลงในฐานข้อมูล รวมถึงการลบ/แก้ไขข้อมูลในฐานข้อมูล หลังจากนั้นก็จะทำหน้าที่แสดงผลข้อมูลตามที่ผู้ใช้ต้องการ โดยผ่านทางแบบสอบถาม นอกจากนี้โปรแกรมยังสามารถจัดทำรายงานเกี่ยวกับข้อมูลทั่วไปและข้อมูลทางกายภาพของทางหลวงชนบทออกทางเครื่องพิมพ์ โดยสามารถเลือกพิมพ์รายงานทั้งหมด หรือเลือกพิมพ์รายงานแบบมีเงื่อนไขได้ (ภาพประกอบที่ 4.2)



ภาพประกอบที่ 4.2 ระบบการทำงานของโปรแกรม RMS

4.4 การพัฒนาโปรแกรม

โปรแกรม RMS พัฒนาขึ้นโดยใช้โปรแกรม Microsoft Visual Basic 6.0 ซึ่งเป็นโปรแกรมประเภทเครื่องมือในการพัฒนาโปรแกรม (Software) อยู่ในชุด Microsoft Visual Studio 6.0 โดยบริษัท Microsoft ความสามารถที่เด่นคือ การพัฒนาโปรแกรมกับฐานข้อมูลซึ่งใช้ติดต่อกับฐานข้อมูลได้หลายรูปแบบ เช่น Access, dBase, FoxPro เป็นต้น โดยทั่วไปนิยมใช้กับฐานข้อมูล Microsoft Access

ในการศึกษานี้ใช้โปรแกรม Microsoft Visual Basic 6.0 เพื่อสร้าง User Interface ระหว่างผู้ใช้งานกับฐานข้อมูล และสร้างชุดคำสั่งจัดการฐานข้อมูล ซึ่งชุดคำสั่งที่ใช้ภาษามาตรฐานกลางสำหรับจัดการข้อมูลในฐานข้อมูล โดยเฉพาะอย่างยิ่งฐานข้อมูลประเภท

RDBMS (Relation Database Management System) คือ ภาษา SQL (Structured Query Language) ซึ่งภาษา SQLนี้ ใช้กับฐานข้อมูลเชิงสัมพันธ์ที่สร้างขึ้นโดยโปรแกรม Microsoft Access 97 ได้เป็นอย่างดี

ภาษา SQL ประกอบไปด้วย 3 ส่วนใหญ่ ๆ คือ

1. Data Definition Language (DDL) เป็นกลุ่มคำสั่งที่ใช้สำหรับจัดการโครงสร้างของฐานข้อมูล เช่น การสร้างฐานข้อมูล ปรับปรุงโครงสร้างของฐานข้อมูล เป็นต้น
2. Data Manipulation Language (DML) เป็นกลุ่มคำสั่งที่ใช้สำหรับจัดการข้อมูลในฐานข้อมูล เช่น การแสดงข้อมูลแบบมีเงื่อนไข การเพิ่มข้อมูล การลบข้อมูล การปรับปรุงข้อมูล เป็นต้น
3. Aggregate Function เป็นกลุ่มฟังก์ชันพิเศษของภาษา SQL ที่ทำหน้าที่เฉพาะอย่าง เช่น การหาผลรวม การหาค่าสูงสุด ต่ำสุด เป็นต้น เป็นกลุ่มคำสั่งที่ช่วยให้ไม่ต้องเขียนโค้ดจัดการเอง

การใช้งานภาษา SQL ร่วมกับโปรแกรม Microsoft Visual Basic 6.0 เพื่อจัดการข้อมูลในฐานข้อมูลในการศึกษานี้ จะใช้งานกลุ่มคำสั่ง DML เป็นหลัก ซึ่งจะมีคำสั่งพื้นฐานอยู่ 4 คำสั่ง คือ

1. SELECT เป็นคำสั่งสำหรับการเลือกข้อมูลหรือแสดงเร็คคอร์ดใด ๆ ที่ต้องการจากตารางเดียวหรือหลายตาราง

รูปแบบการใช้งานคำสั่ง SELECT

SELECT [predicate] { * | table.* | [table.]field] [AS alias]] [, [table.]field2 [AS alias2] [, ...]]}
FROM tableexpression [, ...] [IN externaldatabase]
[**WHERE**...]
[**GROUP BY**...]
[**HAVING**...]
[**ORDER BY**...]

โดยที่

predicate	เป็นการจำกัดจำนวนข้อมูลที่จะแสดงในตารางประกอบด้วย ALL, DISTINCT, DISTINCTROW, หรือ TOP เช่น ALL จะแสดงข้อมูลทั้งหมด, DISTINCT จะแสดงข้อมูลโดยข้อมูลที่ซ้ำกันจะแสดงเพียงรายการเดียว เป็นต้น ในกรณีที่ไม่ระบุ predicate จะแสดงข้อมูลเช่นเดียวกับ ALL
*	แสดงข้อมูลทุกฟิลด์ในตารางที่เลือก หรือ
table	ชื่อตารางที่ต้องการดึงข้อมูลมาแสดง (ตารางเดียว หรือหลายตาราง)
field, field2	ชื่อฟิลด์ที่ต้องการดึงข้อมูล ถ้าระบุมากกว่า 1 ฟิลด์ จะแสดงข้อมูลตามลำดับ

alias, alias2	ชื่อคอลัมน์ใหม่ที่ต้องการใช้แทนชื่อคอลัมน์เดิม
tableexpression	ชื่อตารางที่ต้องการดึงข้อมูลมาแสดง (ตารางเดียว หรือหลายตาราง)
externaldatabase	ชื่อฐานข้อมูลของตารางที่ต้องการดึงข้อมูล ในกรณีที่ tableexpression ไม่ได้อยู่ในฐานข้อมูล

การเรียกข้อมูลจากหลายตาราง สามารถทำได้โดยการอาศัยความสัมพันธ์ของคอลัมน์จากทั้งสองตาราง การเชื่อมโยงความสัมพันธ์นี้เรียกว่า JOIN ซึ่งมีหลายรูปแบบ เช่น INNER JOIN, LEFT JOIN เป็นต้น ซึ่งการพัฒนาโปรแกรม RMS จะใช้เฉพาะรูปแบบ INNER JOIN เท่านั้น

ตัวอย่างการใช้คำสั่ง SELECT เช่น การเรียกข้อมูลรหัสทางหลวง (RoadNo) ชื่อทางหลวง (RoadName) ชนิดผิวทาง (TypeSurface) ระยะทาง (Distance) จากตาราง TypeSurface, Tambon และ RoadInventory โดยการเชื่อมโยงความสัมพันธ์แบบ INNER JOIN และเรียงลำดับข้อมูลตามรหัสทางหลวงจากน้อยไปมาก (ภาพประกอบที่ 4.3)

```
SELECT RoadInventory.RoadNo, RoadInventory.RoadName,
TypeSurface.TypeSurface, RoadInventory.Distance
FROM TypeSurface INNER JOIN (Tambon INNER JOIN RoadInventory ON
Tambon.TamID = RoadInventory.TamID) ON
TypeSurface.TypeSurfaceID=RoadInventory.TypeSurfaceID
WHERE Tambon.TamID=MDIForm1.Text1.Text
ORDER BY RoadNo ASC
```

รหัสทางหลวง	ชื่อทางหลวง	ชนิดผิวทาง	ระยะทาง (กม.)
สข.0009	ซอย 21 กาญจนวนิช ม.3	คอนกรีตเสริมเหล็ก	0.100
สข.0039	ซอย 3 นพเก้า ม.5	คอนกรีตเสริมเหล็ก	0.061
สข.0040	ซอยไชยประภาจุฑิศ ม.3	คอนกรีตเสริมเหล็ก	0.202
สข.0042	ซอย19 บ้านคลองเตย	คอนกรีตเสริมเหล็ก	0.727
สข.0043	กาญจนวนิช-ทุ่งโดน (ตอน 2)	ลาดยาง (เคปซีล)	0.850
สข.2045	บ้านคอหงส์ - อบต.คอหงส์	คอนกรีตเสริมเหล็ก	3.490
สข.3025	บ้านทุ่งรี ม.5-8	คอนกรีตเสริมเหล็ก	0.792
สข.3096	บ้านคลองหวะ - บ้านวังมัจฉา	คอนกรีตเสริมเหล็ก	1.580
สข.4097	ถนนกาญจนวนิช - บ้านคลองหวะ	ลาดยาง (เคปซีล)	0.975
สข.4119	บ้านคลองเตย - บ้านคลองเปล	ลาดยาง (เคปซีล)	4.075
สข.4120	ทางเข้าบ้านคลองเตย	ลาดยาง (เคปซีล)	0.875

ภาพประกอบที่ 4.3 ผลลัพธ์การเรียกใช้คำสั่ง SELECT

2. DELETE เป็นคำสั่งที่ใช้สำหรับลบข้อมูลหรือลบเร็คคอร์ดใด ๆ ในตาราง

รูปแบบการใช้งานคำสั่ง DELETE

DELETE [table.*]

FROM Table

WHERE criteria

โดยที่

table ชื่อตารางที่ต้องการลบข้อมูล ซึ่งจะระบุหรือไม่ก็ได้

Table ชื่อตารางที่ต้องการลบข้อมูล

criteria เงื่อนไขในการลบข้อมูล หรือลบเร็คคอร์ดในตารางนั้น ๆ

ตัวอย่างการใช้คำสั่ง DELETE เช่น การลบข้อมูลป้ายจราจรของทางหลวงชนบทสายหนึ่ง โดยลบรายละเอียดทุกอย่างของเร็คคอร์ด (*) จากตาราง TrafficSign ที่มีคีย์หลักของตารางคือ RoadNo= ‘สข.2045’, KmPost= ‘02+350.000’ และ SignName= ‘สามแยก’ หรือเท่ากับค่าอื่นตามที่ผู้ใช้กำหนด (ภาพประกอบที่ 4.4-4.5)

```
DELETE * FROM TrafficSign WHERE RoadNo=TstxtRoadNo.Text
AND KmPost=TstxtKmPost AND SignName=TstxtSignName
```

กิโลเมตรที่	ประเภทป้ายจราจร	ชื่อป้ายจราจร	ขนาดป้ายจราจร (ซม.*ซม.)	สีของสัญลักษณ์บนป้าย
02+150.000	ป้ายเตือน	ทางโค้งข้างหน้า	50 * 50	เหลือง
02+200.000	ป้ายเตือน	เกียร์ต่ำ	50 * 50	เหลือง
02+350.000	ป้ายเตือน	สามแยก	50 * 50	เหลือง
02+400.000	ป้ายเตือน	สามแยก	50 * 50	เหลือง
02+490.000	ป้ายเตือน	เกียร์ต่ำ	50 * 50	เหลือง

ภาพประกอบที่ 4.4 ข้อมูลป้ายจราจรของทางหลวงชนบท

กิโลเมตรที่	ประเภทป้ายจราจร	ชื่อป้ายจราจร	ขนาดป้ายจราจร (ซม.*ซม.)	สีของสัญลักษณ์บนป้าย
02+150.000	ป้ายเตือน	ทางโค้งข้างหน้า	50 * 50	เหลือง
02+200.000	ป้ายเตือน	เกียร์ต่ำ	50 * 50	เหลือง
02+400.000	ป้ายเตือน	สามแยก	50 * 50	เหลือง
02+490.000	ป้ายเตือน	เกียร์ต่ำ	50 * 50	เหลือง

ภาพประกอบที่ 4.5 ข้อมูลป้ายจราจรหลังการใช้คำสั่ง DELETE

3. INSERT INTO เป็นคำสั่งที่ใช้สำหรับเพิ่มข้อมูลหรือเพิ่มเร็คคอร์ดใด ๆ เข้าไปในตาราง

รูปแบบการใช้งานคำสั่ง UPDATE

รูปแบบที่ 1

INSERT INTO target [(field1[, field2[, ...]])] [IN externaldatabase]

SELECT [source.]field1[, field2[, ...]

FROM tableexpression

รูปแบบที่ 2

INSERT INTO target [(field1[, field2[, ...]])]

VALUES (value1[, value2[, ...])

โดยที่

target	ชื่อตารางหรือแบบสอบถามที่ต้องการเพิ่มข้อมูล
field, field2	ชื่อฟิลด์ที่ต้องการเพิ่มข้อมูล
externaldatabase	ชื่อฐานข้อมูลของตารางที่ต้องการเพิ่มข้อมูล ในกรณีที่ target ไม่ได้อยู่ในฐานข้อมูล
source	ชื่อตารางหรือแบบสอบถามที่ต้องการคัดลอก
tableexpression	ชื่อตารางที่ต้องการคัดลอกข้อมูล เพื่อจะนำข้อมูลมาเพิ่มใน target ในกรณีที่เชื่อมโยงหลายตารางให้ใช้ตัวดำเนิน INNER JOIN, LEFT JOIN, หรือ RIGHT JOIN หรือจะใช้แบบสอบถามก็ได้
value, value2	ค่าของฟิลด์ที่จะเพิ่มเข้าไป โดยต้องระบุให้ตรงกับฟิลด์นั้น ๆ ถ้าค่าที่เพิ่มเข้าไปเป็นข้อมูลประเภท Text จะต้องใช้เครื่องหมาย ‘ ’ กำกับไว้เสมอ เช่น ‘Text’ เป็นต้น

ตัวอย่างการใช้คำสั่ง INSERT INTO เช่น การเพิ่มข้อมูลป้ายจราจรของทางหลวงชนบทสายหนึ่งลงในตาราง TrafficSign รายละเอียดที่เพิ่มเข้าไปจะต้องเรียงตามลำดับในฐานข้อมูล (ภาพประกอบที่ 4.6)

INSERT INTO TrafficSign

VALUES (TstxtRoadNo, TstxtKmPost, TstxtSignType, TstxtSignName, TstxtSignSize, TstxtSignColor, TstxtPlateColor, TstxtPlateType, TstxtPostType, TstxtPostColor, TstxtPostSize, TstxtHeigth, TstxtRoL, TstxtOffset, TstxtComment, TstxtUUpdate)

กิโลเมตรที่	ประเภทป้ายจราจร	ชื่อป้ายจราจร	ขนาดป้ายจราจร (ซม.*ซม.)	สีของสัญลักษณ์บนป้าย
02+150.000	ป้ายเตือน	ทางโค้งข้างหน้า	50 * 50	เหลือง
02+200.000	ป้ายเตือน	เกียร์ต่ำ	50 * 50	เหลือง
02+400.000	ป้ายเตือน	สามแยก	50 * 50	เหลือง
02+490.000	ป้ายเตือน	เกียร์ต่ำ	50 * 50	เหลือง
03+150.000	ป้ายเตือน	สี่แยก	50 * 50	เหลือง

ภาพประกอบที่ 4.6 ข้อมูลป้ายจราจรหลังการใช้คำสั่ง INSERT INTO

4. UPDATE เป็นคำสั่งที่ใช้สำหรับแก้ไขข้อมูลหรือแก้ไขเร็คคอร์ดใด ๆ ในตาราง

รูปแบบการใช้งานคำสั่ง UPDATE

UPDATE table

SET newvalue

[WHERE] criteria;

โดยที่

table	ชื่อตารางที่ต้องการแก้ไขข้อมูล
newvalue	ค่าใหม่ที่กำหนดให้แทนที่เร็คคอร์ดเก่าที่ต้องการแก้ไข
criteria	เงื่อนไขในการแก้ไขข้อมูลในตารางนั้น ๆ

ตัวอย่างการใช้คำสั่ง UPDATE เช่น การแก้ไขปรับปรุงข้อมูลป้ายจราจรของทางหลวงชนบทสายหนึ่งลงในตาราง TrafficSign โปรแกรมจะนำข้อมูลที่แก้ไขปรับปรุงใหม่ไปแทนที่ข้อมูลเก่า โดยที่คีย์หลักของตาราง TrafficSign (RoadNo, KmPost และ SignName) จะต้องไม่เปลี่ยนแปลง (ภาพประกอบที่ 4.7)

```
UPDATE TrafficSign SET SignType=TstxtSignType
SignSize=TstxtSignSize, SignColor=TstxtSignColor, PlateColor=TstxtPlateColor,
PlateType=TstxtPlateType, PostType=TstxtPostType, PostColor=TstxtPostColor,
PostSize=TstxtPostSize, Heigth=TstxtHeigth, RoI=TstxtRoL, Offset=TstxtOffset,
Comment=TstxtComment, UUpdate=TstxtUUpdate
WHERE RoadNo=TstxtRoadNo
AND KmPost=TstxtKmPost
AND SignName=TstxtSignName
```

กิโลเมตรที่	ประเภทป้ายจราจร	ชื่อป้ายจราจร	ขนาดป้ายจราจร (ซม.*ซม.)	สีของสัญลักษณ์บนป้าย
02+150.000	ป้ายเตือน	ทางโค้งข้างหน้า	50 * 50	เหลือง
02+200.000	ป้ายเตือน	เกียร์ต่ำ	50 * 50	เหลือง
02+400.000	ป้ายเตือน	สามแยก	50 * 50	เหลือง
02+490.000	ป้ายเตือน	เกียร์ต่ำ	50 * 50	เหลือง
03+150.000	ป้ายเตือน	สี่แยก	55 * 55	เหลือง

ภาพประกอบที่ 4.7 ข้อมูลป้ายจราจรหลังการใช้คำสั่ง UPDATE

ตัวดำเนินการ (Operator) ที่ใช้ในการเปรียบเทียบมีดังนี้

ตัวดำเนินการ	ความหมาย
=	เท่ากับ
<>	ไม่เท่ากับ
<	น้อยกว่า
>	มากกว่า
<=	น้อยกว่าหรือเท่ากับ
>=	มากกว่าหรือเท่ากับ

ตัวดำเนินการ (Operation) ที่ใช้ในการคำนวณมีดังนี้

ตัวดำเนินการ	ความหมาย
-	ลบ
+	บวก
/	หาร
\	หารแต่ผลที่ได้จะเป็นจำนวนเต็มที่มีการปัดเศษ
Mod	หารแต่ผลที่ได้จะเป็นเศษจากการหาร
^	ยกกำลัง
*	คูณ

ตัวดำเนินการตรรกยะ (Logical Operator) ที่นิยมใช้มี 3 ชนิด คือ AND, OR, และ NOT

ตัวอย่างการใช้ตัวดำเนินการ (Operation) ที่ใช้ในการคำนวณ เช่น การคำนวณอายุการใช้งานของทางหลวงชนบทในส่วนสอบถามข้อมูล โปรแกรมเรียกข้อมูลรหัสทางหลวง ชื่อทางหลวง อายุใช้งาน (ปี) อายุใช้งาน (เดือน) อายุใช้งาน (วัน) วันที่ปัจจุบัน วันที่ก่อสร้างแล้วเสร็จ แล้วเรียงลำดับข้อมูลทางหลวงตามอายุการใช้งานจากมากไปน้อย ซึ่งอายุการใช้งานของทางหลวงได้จากนำวันที่ปัจจุบันลบวันที่ก่อสร้างแล้วเสร็จ หลังจากนั้นแปลงเป็นจำนวนปี เดือน และวัน โดยใช้ตัวดำเนินการ -, / และ Mod (ภาพประกอบที่ 4.8)

```
SELECT Constrction.RoadNo, RoadInventory.RoadName,
fix((Val(Now()-Constrction.FinishedDate))/365),
fix(((Val(Now()-Constrction.FinishedDate)) Mod 365)/30),
fix(((Val(Now()-Constrction.FinishedDate)) Mod 365) Mod 30),
Now(),Constrction.FinishedDate
FROM RoadInventory INNER JOIN Constrction ON RoadInventory.RoadNo =
Constrction.RoadNo WHERE RoadInventory.TamID=MDIForm1.Text1.Text
ORDER BY ((Val(Constrction.FinishedDate))/365) DESC,
fix(((Val(Now()-Constrction.FinishedDate)) Mod 365)/30),
fix(((Val(Now()-Constrction.FinishedDate)) Mod 365) Mod 30)
```

โดยที่

Constrction.RoadNo คือ รหัสทางหลวงจากตาราง Construction

RoadInventory.RoadName คือ ชื่อทางหลวงจากตาราง RoadInventory

fix((Val(Now()-Constrction.FinishedDate))/365) คือ อายุใช้งาน (ปี)

fix(((Val(Now()-Constrction.FinishedDate)) Mod 365)/30) คือ อายุใช้งาน (เดือน)

fix(((Val(Now()-Constrction.FinishedDate)) Mod 365) Mod 30) คือ อายุใช้งาน (วัน)

Now() คือ วันที่ปัจจุบัน

Constrction.FinishedDate คือ วันที่ก่อสร้างแล้วเสร็จจากตาราง Construction

รหัสทางหลวง	ชื่อทางหลวง	อายุการใช้งาน (ปี)	อายุการใช้งาน (เดือน)	อายุการใช้งาน (วัน)
สข.3096	บ้านคลองหวะ-บ้านวังมัจฉา	5	8	17
สข.4120	ทางเข้าบ้านคลองเตย	4	9	3
สข.2045	บ้านคอหงส์ - อบต.คอหงส์	4	2	11
สข.0008	ชอยกล้วยไม้ ม.5	1	11	17
สข.4097	ทางเข้าบ้านคลองหวะ	1	5	1
สข.4119	บ้านคลองเตย-บ้านคลองเปล	1	4	2

ภาพประกอบที่ 4.8 ผลลัพธ์การใช้ตัวดำเนินการ (Operation)

กลุ่มฟังก์ชัน Aggregate เป็นฟังก์ชันที่ใช้ในการคำนวณทางคณิตศาสตร์

ชื่อฟังก์ชัน	หน้าที่
AVG()	หาค่าเฉลี่ยของฟิลด์ จากเร็คคอร์ดทั้งหมด
COUNT()	นับจำนวนเร็คคอร์ด
FIRST()	หาค่าแรกในฟิลด์
LAST()	หาค่าสุดท้ายในฟิลด์
MAX()	หาค่ามากที่สุด หรือสูงสุด
MIN()	หาค่าน้อยที่สุด หรือต่ำสุด
SUM()	หาผลรวมทั้งหมดของฟิลด์

นอกจากนี้ยังมีตัวดำเนินการ (Operation) ของ SQL ที่ใช้ในการกำหนดเงื่อนไขของข้อมูลในอนุประโยค WHERE เช่น

BETWEEN...AND... เป็นตัวดำเนินการที่กำหนดเงื่อนไขของคอลัมน์เป็นค่าระหว่างค่าสองค่า

IN เป็นตัวดำเนินการที่ใช้กับเงื่อนไขของคอลัมน์ที่ต้องการระบุเงื่อนไขเป็นกลุ่มของข้อมูล

LIKE เป็นตัวดำเนินการที่ใช้ในการค้นหาข้อมูลของคอลัมน์ที่เก็บข้อมูลประเภทอักษร โดยยังไม่ทราบค่าที่แน่นอน หรือรู้เพียงบางตัวอักษรเท่านั้น

IS NULL เป็นตัวดำเนินการที่ใช้ในการแสดงค่าของคอลัมน์ที่มีค่าเป็นค่าว่างหรือไม่มีค่า

ตัวดำเนินการเหล่านี้ ยังสามารถใช้เป็นเงื่อนไขในเชิงปฏิเสธ โดยใช้คำว่า NOT นำหน้า

นอกจากการใช้ฟังก์ชันที่เกี่ยวกับการรวมแล้ว ยังสามารถแสดงข้อมูลในลักษณะกลุ่มข้อมูล โดยใช้อนุประโยค GROUP BY เพื่อให้จัดกลุ่มตามคอลัมน์ที่กำหนด ให้จัดกลุ่มข้อมูลเฉพาะย่อยลงไป นอกจากนี้ยังใช้อนุประโยค HAVING ร่วมกับ GROUP BY เพื่อต้องการให้แสดงข้อมูลที่ผ่านการจัดกลุ่มโดย GROUP BY เพียงบางส่วน ตามเงื่อนไขที่ระบุใน HAVING ดังนั้น การเรียกดูข้อมูลโดยใช้ HAVING จะต้องมี GROUP BY เสมอ

กลุ่มคำสั่งที่กล่าวข้างต้นเป็นเพียงบางส่วนของกลุ่มคำสั่งที่ใช้ในการจัดการฐานข้อมูลทางหลวงชนบท ซึ่งใช้ในการพัฒนาโปรแกรมคอมพิวเตอร์

4.5 เครื่องมือและอุปกรณ์ที่ใช้ในการพัฒนาโปรแกรม

4.5.1 ด้าน Hardware

1. เครื่องคอมพิวเตอร์ 1 เครื่อง
   - RAM 256 MB
   - Harddisk 2 GB

4.5.2 ด้าน Software

1. ระบบปฏิบัติการ Windows XP
2. Microsoft Access 97 เป็นระบบจัดการฐานข้อมูล
3. Microsoft Visual Basic 6.0 เป็นเครื่องมือหลักในการพัฒนาโปรแกรม
4. Crystal Reports 8.5 เป็นเครื่องมือในการสร้างรายงานจากฐานข้อมูล
5. Setup Factory 5.0 เป็นเครื่องมือในการสร้างตัวติดตั้งโปรแกรม
6. RoboHelp Classic เป็นเครื่องมือในการสร้างส่วนช่วยเหลือ (Help)
7. Adobe Photoshop 7.0 เป็นเครื่องมือในการตกแต่งภาพในโปรแกรม
8. IconArt เป็นเครื่องมือในการสร้างไอคอนต่างๆ ในโปรแกรม