การบังคับและแยกประเภทความถูกต้องของข้อมูล

(ตรวจสอบความสามารถในการบังคับความถูกต้องของข้อมูล โดยระบบจัดการฐานข้อมูล Oracle 7 ในส่วนของ Oracle Developer 2000 กับ Stored Procedure)

Integrity Constraint Classification and Enforcement

(Implement the Enforcement in Oracle 7 By Using Oracle Developer 2000 with Stored Procedure)

โดย

นายสาธิษฐ์ นากกระแสร์

รหัส 38626057

อาจารย์ที่ปรึกษา

รศ.ดร.ศุภมิตร จิตตะยโศธร

รายงานนี้เป็นส่วนหนึ่งของวิชาโครงการพัฒนาระบบงาน

หลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาเทคโนโลยีสารสนเทศ

ภาคเรียนที่ 1 ปีการศึกษา 2540

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ชื่อหัวข้อ การบังคับและแยกประเภทความถูกต้องของข้อมูล (ตรวจสอบความสามารถในการบังคับความถูกต้องของข้อมูล โดยระบบจัดการฐานข้อมูล Oracle 7 ในส่วนของ Oracle Developer 2000 กับ Stored Procedure)

นักศึกษา นายสาธิษฐ์ นากกระแสร์

อาจารย์ที่ปรึกษา รศ.ดร.ศุภมิตร จิตตะยโศธร

ระดับการศึกษา วิทยาศาสตร์มหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ

แขนงวิชา วิทยาการสารสนเทศ

พ.ศ. 2540

บทคัดย่อ

การที่จะทำให้ระบบฐานข้อมูลสามารถทำงานได้อย่างดีมีประสิทธิภาพนั้น จะต้องมีการจัดการที่ดี และสำหรับในส่วนของความคงสภาพของข้อมูลแล้ว ถือว่าเป็นส่วนหนึ่งที่สำคัญ ซึ่งจะต้องให้ความสนใจ โดยถือเป็นส่วนที่ทำให้ข้อมูลคงความถูกต้อง เกิดความตรงกันในทุกส่วนของระบบเมื่อข้อมูลในระบบมีการเปลี่ยนแปลง จากแนวความคิดนี้ ทำให้มีความพยายาม เพื่อหาแนวทางและวิธีในการควบคุมความคงสภาพของข้อมูลขึ้น ซึ่งสอดคล้องกับโครงงานนี้ ที่มีการเน้นในส่วนของการจัดแบ่ง และควบคุมความคงสภาพของข้อมูลขึ้นมาใหม่ โดยมีการศึกษาหาข้อมูลจากตำรา บทความทางวิชาการรวมทั้งเอกสารอ้างอิงของผลิตภัณฑ์ ที่มีการจัดการในด้านฐานข้อมูล พร้อมกับสร้างระบบตัวอย่าง โดยศึกษากฎเกณฑ์ ต่าง ๆ ตามที่ระบบต้องการ เพื่อทำการตรวจสอบและทำการเปรียบเทียบกับกฎความคงสภาพ ที่ได้มีการนำเสนอการจัดแบ่งไว้ จากนั้นก็นำเอากฎเกณฑ์ดังกล่าวมาสร้างเพื่อทำการทดสอบตามที่ระบบได้กำหนด ให้มีการใช้งานได้จริงภายใต้การนำเอา Oracle Developer 2000 ในส่วนของ Stored Procedure และ Database Trigger ตลอดจน Form Developer มาช่วยสร้างระบบตัวอย่างและสนับสนุนการทดสอบดังกล่าวด้วย

Title	Integrity Constraint Classification and Enforcement (Implement the Enforcement in Oracle 7 By Using Oracle Developer 2000 with Stored Procedure)
Student	Mr.Sathit Nakkrasae
Advisor	Assoc.Prof.Dr.Suphamit Chittayasothorn
Level of Study	Master of Science in Information Technology
Major	Information Science
Year	1997

ABSTRACT

It is essential to make best performance in managing database application ,and integrity constraints. Because of consistency and accuracy that we have to consider. So the classification and Enforcement Integrity Constraints ; using the knowledge from the academic journal , books and database product references ; are the main tasks of this article to make integrity constraints of the database system completely and cover for every cases that should have in the system. In addition , we also test these integrity constraints by implementing "The register system" which uses Oracle form , Stored Procedure and Database trigger in Oracle Developer 2000. This system presents checking and comparing the system's rules with the integrity constraints that we have classified and shows that integrity constraints can cover every specification rules of the system. Moreover, it also ensures that system must be always used in consistency and accuracy state.

กิตติกรรมประกาศ

การจัดทำโครงงานในครั้งนี้สำเร็จได้ เพราะได้รับการส่งเสริมและสนับสนุนจากหลายฝ่าย ซึ่งกระผมใคร่ขอขอบพระคุณดังนี้คือ

1. บิดา มารดา ซึ่งเป็นผู้มีพระคุณมาก ได้ให้กำเนิด เลี้ยงดู ส่งเสริมให้ได้รับการศึกษาและสั่งสอนให้กระทำในสิ่งที่ดีมาโดยตลอด เป็นผู้ให้กำลังใจในด้านต่างๆ มากมายซึ่งรวมถึงช่วยสนับสนุนในสิ่งที่ไม่มีในทุก ๆ ด้าน

2. อาจารย์ศุภมิตร จิตตะยโศธร ซึ่งเป็นอาจารย์ที่ปรึกษาโครงงาน ที่คอยให้คำปรึกษาและแนะนำในด้านการศึกษาค้นคว้า แนวทางในการทำงาน ชี้แนะให้เกิดความกระจ่างในปัญหาต่าง ๆ ที่เกิดขึ้นขณะที่มีการทำโครงงาน

3. อาจารย์ทุกท่านที่ให้การศึกษาในเนื้อหาวิชาต่าง ๆ จากที่เรียนมาทั้งหมด ซึ่งได้นำมาเป็นพื้นฐานความรู้ในการทำงานและสนับสนุนการพัฒนาโครงงาน

4. ผู้มีบุญคุณตลอดจนเพื่อน ๆ ที่ให้การสนับสนุนในทุก ๆ ด้านไม่ว่าจะเป็นการจัดหาข้อมูล และเอกสารต่าง ๆ ซึ่งไม่สามารถที่จะกล่าวนามได้หมดในที่นี้

สาธิษฐ์ นากกระแสร์

สารบัญ

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญภาพ

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

การพัฒนาระบบฐานข้อมูลที่มีอยู่ในปัจจุบันนี้ นับว่ามีปรากฏอยู่มากมายและการที่จะทำให้ระบบฐานข้อมูลสามารถทำงานได้อย่างมีประสิทธภาพนั้น ต้องมีการจัดการที่ดีในทุก ๆ ด้านที่มีความเกี่ยวข้อง และสำหรับความคงสภาพของฐานข้อมูลซึ่งถือว่าเป็นส่วนที่สำคัญส่วนหนึ่ง เพราะมันจะเป็นตัวที่คอยตรวจสอบการทำงานต่าง ๆ และคอยตรวจเช็คความคงสภาพของข้อมูลให้มีความถูกต้องอยู่เสมอ จากเหตุผลดังกล่าวจึงก่อให้เกิดแนวความคิดในการพัฒนาปรับปรุงกฎความคงสภาพให้มีความสมบูรณ์ สามารถรองรับความต้องการของระบบในกรณีต่างๆ ให้ได้มากที่สุด

1.2 วัตถุประสงค์ของการพัฒนา

1 ส่งเสริมให้การจัดการฐานข้อมูล มีประสิทธิภาพสูงยิ่งขึ้น

2 เป็นการพัฒนาระบบการจัดการฐานข้อมูลโดยเน้น ในส่วนของความคงสภาพของข้อมูล

3 สร้างความถูกต้องและความตรงกันของข้อมูลในระบบให้มีมากยิ่งขึ้น

4 เป็นแนวทางในการสนับสนุนการพัฒนาระบบฐานข้อมูล รวมทั้งผู้ที่มีความสนใจเพื่อที่จะใช้เป็นข้อมูลอ้างอิงในการพัฒนาส่วนที่เกี่ยวข้องต่อไป

5 ทำการพัฒนา เสนอแนวทางในการจัดแบ่งและการควบคุมความคงสภาพของข้อมูล เพื่อให้ครอบคลุมกับทุกระบบ ตลอดจนสามารถรองรับกฎเกณฑ์ที่อาจจะเกิดขึ้นในอนาคตให้ได้มากที่สุด โดยพยายามที่จะไม่อ้างอิงกับระบบหรือ Model ใด ๆ และสามารถสนับสนุนการทำงานกับทุกระบบ ทุก Model ที่จำเป็นต้องใช้ ตลอดจนนำไปเป็นพื้นฐานหรือแนวทางในการตรวจสอบเปรียบเทียบ และพัฒนางานในส่วนของการบังคับและควบคุมความคงสภาพของข้อมูลในระบบ

1.3 ทฤษฎีที่ใช้ในโครงงาน

ในการจัดแบ่ง ควบคุมความคงสภาพของข้อมูลในโครงงานนี้ ได้นำเอาหลักการและแนวทางพื้นฐานในการจัดทำ ตามที่มีในตำราและบทความวิจัยต่าง ๆ ซึ่งบางส่วนมีการพูดถึงกฎความคงสภาพโดยการอ้างอิงถึงข้อมูลที่เป็น Relational Data Model บางส่วนได้การพูดถึงแนวทางในการพัฒนากฎความคงสภาพตามที่มีในตำรา โดยเสนอวิธีการใหม่มาจัดการเพื่อควบคุมความถูกต้องของระบบให้เกิดประสิทธิภาพเพิ่มขึ้น ตลอดจนเกิดความสมบูรณ์กับระบบ และนอกจากนี้ก็ยังมีเนื้อหาในด้านอื่น ๆ อีก มากมาย

สำหรับตำราที่มีการพูดถึงกฎความคงสภาพของข้อมูล และได้นำมาเป็นพื้นฐานความรู้ในการจัดแบ่งมีดังนี้

1.3.1 An Introduction to Database system แยกกฎความคงสภาพ ออกเป็น 4 กลุ่มคือ

1 Domain Integrity Rule เป็นการกำหนดค่าที่ถูกต้องสำหรับขอบเขต (Domain) ซึ่งในความเป็นจริงแล้วจะหมายถึงการกำหนดค่าที่จะใช้สร้างขอบเขต สำหรับข้อมูลในระบบงาน ซึ่งสามารถสร้างโดยตรงในส่วนของประโยคที่ใช้สร้างขอบเขต เช่น

```
CREATE DOMAIN QTY MUMERIC(9)
 FORALL QTY (QTY>0 AND
                    QTY<=5000 AND
                    MOD(QTY,50) = 0);
```

2 Attribute Integrity Rule เป็นการกำหนดค่าที่ถูกต้องสำหรับ Attribute ซึ่งหมายถึงการกำหนดค่าขอบเขต ให้กับ Attribute

3 Relation Rule เป็นการอ้างถึงค่าที่ถูกต้องของตารางที่กำหนดเท่านั้น ซึ่งจะไม่เกี่ยวข้องกับ ตารางหรือขอบเขต อื่น เช่น

```
CREATE INTEGRITY RULE SR7
FORALL S (IF S.CITY='LONDON' THEN
                    S.STATUS=20)
ON ATTEMPTED VIOLATION REJECT;
```

4 Database rule เป็นความสัมพันธ์ระหว่าง 2 ตารางที่ต่างกัน เช่น

```
CREATE INTEGRITY RULE  C95
FORALL SX ( FORALL SPX
                    (IF SX.STATUS < 20 AND
```

SX.S# = SPX.S# THEN

SPX.QTY<= 500));

1.3.2 Database System Concept แบ่งกฎความคงสภาพของข้อมูล ได้เป็น

1 Domain Constraints เป็นข้อบังคับขั้นพื้นฐานของกฎความคงสภาพของข้อมูล โดยจะมีการตรวจสอบค่าต่าง ๆ อย่างง่าย ๆ เมื่อถูกป้อนเข้ามาในระบบ

2 Referential Integrity เป็นกฎที่ต้องการจะให้ค่าที่ปรากฏในตารางอันหนึ่งสำหรับกลุ่มของ Attributes ที่กำหนด จะต้องมีความถูกต้อง และสามารถปรากฏอยู่ในกลุ่มของ Attributes ในอีกตารางหนึ่งอย่างถูกต้องด้วย

1.3.3 Oracle 7 Server แบ่งชนิดของกฎความคงสภาพของข้อมูล เป็นดังนี้

1 Null เป็นกฎที่ถูกกำหนดบนคอลัมน์(Column)ใดคอลัมน์หนึ่ง ว่าจะอนุญาตหรือไม่อนุญาต ให้มีการ Insert หรือ Update แถว ที่คอลัมน์ในแถวนั้น มีค่าเป็น Null

2 Unique Column Value เป็นกฎที่กำหนดบนคอลัมน์ (หรือ กลุ่มของคอลัมน์) โดยยอมให้มีเพียง การ Insert หรือ Update กับแถว (Row) ที่ค่าของคอลัมน์(หรือ กลุ่มของ คอลัมน์) ใน แถวเป็น Unique แต่จะยอมให้มีค่า เป็น Null ได้

3 Primary Key Values เป็นกฎที่ถูกกำหนดบนคอลัมน์ (หรือกลุ่มของคอลัมน์) เพื่อให้แต่ละแถวในตารางสามารถถูกกำหนดเป็น Unique ตามการกำหนดค่าในคอลัมน์ (หรือกลุ่มของคอลัมน์) ดังกล่าว โดยจะมีค่า เป็น Null ไม่ได้

4 Referential Integrity เป็นกฎที่ถูกกำหนดบนคอลัมน์ (หรือ กลุ่มของคอลัมน์) ว่าในตารางใด ๆ จะยอมให้มีการ Insert หรือ Update กับ แถวได้ ถ้าค่าของคอลัมน์หรือ กลุ่มของคอลัมน์(ค่าที่ขึ้นต่อกัน) ตรงกับค่าใน คอลัมน์ ของตารางที่อ้างถึงกัน(ค่าที่อ้างถึง)

กฎที่เกี่ยวข้องกับ Referential Integrity จะรวมถึง

* Restrict คือ กฎ Referential Integrity ที่ไม่ยอมให้มีการ Update หรือ Deleteข้อมูลที่ถูกอ้างถึง

* Set to Null เมื่อข้อมูลที่มีการอ้างถึง ถูก Update หรือ Delete ข้อมูลที่เกี่ยวข้องและมีผลต่อมันจะถูกกำหนดค่าให้เป็น Null ทั้งหมด

* Set to Default เมื่อข้อมูลที่ถูกอ้างถึงถูก Updated หรือ Deleted ข้อมูลที่เกี่ยวข้องและมีผลต่อมันจะถูกกำหนดให้เป็น ค่า Default ที่กำหนดไว้ทั้งหมด

* Cascade เมื่อข้อมูลที่ถูกอ้างถึงถูก Updated ข้อมูลที่เกี่ยวข้องและมีผลต่อมันทั้งหมดจะถูก Updated ให้สอดคล้องกันและ เมื่อแถว ถูกลบ แถวที่เกี่ยวข้องจะถูกลบด้วย

สำหรับในส่วนของบทความวิจัยรวมทั้งรายละเอียด และการควบคุมความคงสภาพของข้อมูลในส่วนของ Oracle จะได้กล่าวในบทต่อไป

1.4 ประโยชน์ที่คาดว่าจะได้รับ

ในโครงงานนี้ได้มีการศึกษา และนำเสนอการการแบ่งแยก ตลอดจนแนวทางการบังคับความคงสภาพของข้อมูล ดังนั้นประโยชน์ที่คาดว่าจะได้รับจากโครงงานนี้คือ

1 เป็นข้อมูลในการอ้างอิงสำหรับผู้ที่ต้องการพัฒนาระบบฐานข้อมูล เพื่อทำให้การพัฒนาระบบมีความสมบูรณ์โดยเฉพาะในส่วนของกฎความคงสภาพของข้อมูล

2 เป็นข้อมูลพื้นฐานสำหรับผู้ที่ต้องการศึกษาความคงสภาพของข้อมูล และใช้เป็นแนวทางในการพัฒนาต่อไป

3 เป็นการเพิ่มความน่าเชื่อถือให้กับระบบที่จะมีการพัฒนาขึ้น

4 สามารถนำเอาแนวทาง การนำเสนอการบังคับและควบคุมความคงสภาพของข้อมูล ตามที่กำหนดไว้ในทฤษฎี มาช่วยตัดสินใจและช่วยพัฒนาระบบฐานข้อมูลในส่วนของความคงสภาพให้มีความถูกต้องและชัดเจนยิ่งขึ้น

1.5 ขอบเขตของโครงงาน

1 ทำการศึกษาความคงสภาพของข้อมูลที่มีอยู่ในตำรา และในบทความวิจัยต่าง ๆ

2 นำเอาแนวความคิดและเนื้อหาที่ได้จากการศึกษาข้างต้นมาทำการจัดแบ่ง และบังคับความคงสภาพของข้อมูลโดยถือเป็นการนำเสนอในแนวทางใหม่

3 ทำการสร้างระบบตัวอย่าง โดยระบุกฎเกณฑ์ที่ระบบต้องการ ทำการเปรียบเทียบจัดกลุ่มให้เข้ากับความคงสภาพของข้อมูลที่ได้นำเสนอไว้ และแสดงให้เห็นถึงการบังคับ และควบคุมในกฎเกณฑ์ดังกล่าวโดยใช้ Oracle Developer 2000 ในส่วนของ Stored Procedure และ Database Trigger ภายใต้การออกแบบระบบของ Oracle Form Developer 2000

บทที่ 2

บทความวิจัยทางด้าน Integrity Constraints

ก่อนจะมีการจัดแบ่งในส่วนของ ความคงสภาพของข้อมูลและการบังคับควบคุมนั้น ได้มีบทความวิจัยที่ทำการศึกษาและใช้เป็นแนวทางในการจัดแบ่งดังนี้

2.1 Integrity = Validity + Completeness

บทความนี้ได้กล่าวถึง Model ของการควบคุมความคงสภาพของข้อมูลซึ่งได้มีการเสนอการควบคุมความคงสภาพแบบใหม่ ที่ถูกเรียกว่า Validity Constraints และ Completeness Constraints โดยคร่าว ๆ แล้ว ความคงสภาพของข้อมูล 2 อันนี้จะถูกมุ่งเน้นเพื่อให้แน่ใจถึง Validity และ Completeness ของแต่ละคำตอบ โดยความคงสภาพของข้อมูล ทั้ง 2 จะเป็นตัวยืนยันความคงสภาพ ของ Information ของฐานข้อมูล

Validity Constraints จะตระหนักถึงข้อมูลข่าวสารที่เป็นเท็จ ซึ่งต้องไม่เป็นส่วนหนึ่งของ Valid Database อย่างเช่น สมมติว่า Database ต้องการที่มั่นใจต่อ Validity ของ Set ของ " Nonstop flights from A to B" Database ยอมรับ Validity Constraints อันนี้ถ้ามันไม่รวมเอาข้อมูลที่เป็นเท็จซึ่งตรงข้ามกับ Validity Constraints ดังกล่าว

Completeness Constraints จะตระหนักถึงข้อมูลข่าวสาร ที่เป็นจริงที่ต้องเป็นส่วนหนึ่งของฐานข้อมูล ที่สมบูรณ์ อย่างเช่น สมมติว่า มันต้องการที่จะมั่นใจว่า Completeness Constraints ที่เหมาะสมควรจะเป็น " Flights จาก A to B ผ่าน Domestic Carriers" ฐานข้อมูล จะยอมรับ Completeness Constraints นี้ ถ้ามันรวมเอาทุก ๆ สายการบินที่เป็นจริงที่บินจาก A to B ผ่าน Domestic Carrier ไว้ในฐานข้อมูล

นอกจากนี้ Model ของความคงสภาพ อันใหม่นี้ยังถูกออกแบบเพื่อพิสูจน์ ความคงสภาพของคำตอบสำหรับแต่ละคำตอบที่ถูกแสดงออกมาจากการ Query โดยมันจะกำหนดว่า

1 คำตอบต้อง Valid (Information จะต้องถูกต้อง) หรือไม่ก็มีบางส่วนที่ Valid (บางส่วนที่ถูกระบุจะต้องมั่นใจว่าถูกต้อง)

2 คำตอบต้องสมบูรณ์ (Information ที่ระบุต้องมั่นใจว่า จะรวมอยู่ในReal World ที่ปรากฏทั้งหมด) หรือไม่ก็มีบางส่วนที่สมบูรณ์ (บางส่วนที่ถูกระบุจะรวมอยู่ใน Real World ที่ปรากฏทั้งหมด) ผลของมันก็คือฐานข้อมูลกำลังตรวจสอบคุณภาพของคำตอบ

ที่เกิดขึ้นจากการ Query โดยใช้กรอบของความรู้ที่ฐานข้อมูลมีอยู่ (Integrity Constraints)

Validity Constraints ถูกละเมิดเมื่อมี Tuple อยู่ใน View ของ ฐานข้อมูลแต่ไม่อยู่ใน Real World โดย Validity Constraints ที่ฐานข้อมูลมี ณ ขณะนั้นอาจถูกละเมิดได้ใน 2 ทางคือ

V1 ฐานข้อมูลถูก Update โดยการเพิ่ม Tuple เข้าไปใน View ของฐานข้อมูลซึ่งเป็น Tuple ที่เป็น False (ไม่ใช่ View ของ Real World)

V2 Real World เปลี่ยน และ Tuple ถูกลบจาก View ของ Real World ซึ่ง Tuple ดังกล่าวมีปรากฏในฐานข้อมูล

Completeness Constraints ถูกละเมิดเมื่อมี Tuple ที่อยู่ใน View ของ Real World แต่ไม่อยู่ใน View ของฐานข้อมูล การละเมิดมีได้ 2 ทางคือ

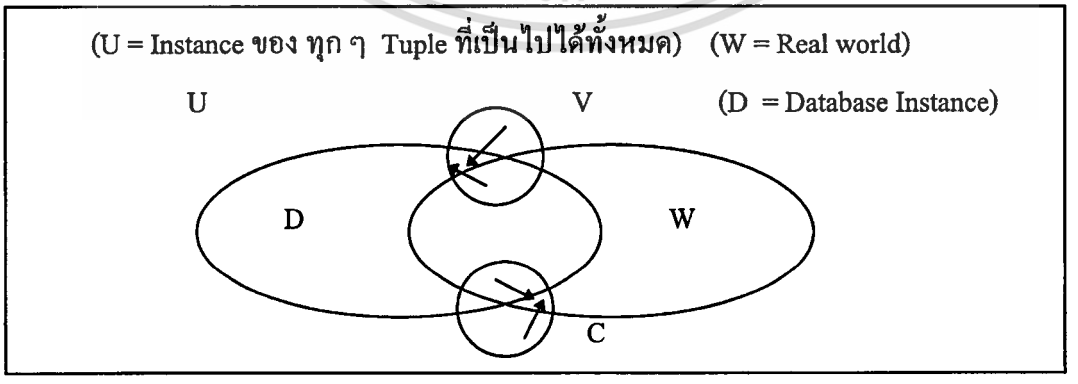
C1 ฐานข้อมูลถูก Update โดยการลบ Tuple ออกจาก View ของฐานข้อมูลซึ่ง Tuple นั้นเป็นจริง(ใน view ของ real world)

C2 Real World เปลี่ยน และ Tuple ถูกเพิ่มเข้าไปใน View ของ Real World ซึ่งไม่ถูกเก็บในฐานข้อมูล (ไม่อยู่ใน View ของ ฐานข้อมูล)

ภาพที่ 1 แสดงการย้ายของ Tuple ที่เกี่ยวข้องกับทั้ง 4 ชนิดของ การละเมิด Integrity ซึ่งทั้ง 4 ชนิดแยกได้ เป็น 2 แบบ คือ การละเมิดที่เกิดจากการเปลี่ยนแปลงของฐานข้อมูล ที่ไม่ได้รับการยอมรับ หรือยืนยัน (V1,C1) และ การละเมิดจากการเปลี่ยนแปลงสถานะปัจจุบันของ Real World (V2,C2)

ภาพที่ 1

Violations of Constraints



ในการแสดงของ Validity Constraints กับกฎเกณฑ์ข้อจำกัดต่อไปนี้คือ View ของ Validity Constraints จะไม่ยอมรวม Tuple ที่เป็นจริง ข้อจำกัดนี้มี 2 ลำดับคือ

1 สถานการณ์ที่ถูกอธิบายใน V1 ซึ่งง่ายที่จะป้องกัน เพียงแค่ดูว่าเมื่อใด ก็ตามที่มีการ Update จะต้องเพิ่ม Tuple เข้าไปใน View ของ Validity Constraints(นั่นคือไม่จำเป็นต้องตรวจสอบว่า Tuple นั้น เป็น เท็จ) ดังนั้นทุก ๆ การ Update ที่เพิ่มTuple เข้าไปใน Views ของ Validity Constraints จะถูก Reject โดยระบบ

2 ถ้าไม่มี Tuple ที่เป็นจริงใด ๆ ใน View ของ Validity Constraints สถานการณ์ใน 2 ก็จะไม่เกิดขึ้น ดังนั้นข้อจำกัดใน Validity Constraints จะช่วยลดภาระให้ DBA จากความต้องการ ที่จะตรวจ Real World สำหรับ Tuple ใน View ของ Validity Constraints ที่กลายเป็นเท็จ และจะถูกลบออกจาก ฐานข้อมูล

สรุป Validity Constraints ที่ไม่เคยระบุ Tuple ที่เป็นจริง จะถูกตรวจสอบโดยตัวระบบเอง : ทุก ๆ ความพยายามที่จะ Update ฐานข้อมูลจะถูก Reject ถ้ามันมีการเพิ่มเข้าไปใน View ของ ฐานข้อมูล

การ Implement ของ Completeness Constraints กับ 2 ข้อจำกัดข้างต้น " View ของ Completeness Constraints ไม่เคยรวม Tuples ที่เป็นเท็จ " มีลำดับของข้อจำกัดคล้ายคลึงกันคือ

1 สถานการณ์ที่อธิบายใน C1 ง่ายต่อการ Delete ซึ่งมีความจำเป็นเพียงแค่ต้องกำหนดว่าเมื่อใดก็ตามที่มีการ Update จะต้องลบ Tuple ออกจาก View ของ Completeness Constraints (นั่นคือมันไม่จำเป็นต้องตรวจสอบว่า Tuple นั้นเป็นจริง) ดังนั้นการ Update ใด ๆ ที่จะส่งผลใน ความไม่เหมาะสมจะถูก Reject ทันที

2 ถ้าไม่เคยมี Tuple ที่เป็นเท็จใน View ของ Completeness Constraints สถานการณ์ 2 ก็จะไม่เกิดขึ้น ดังนั้นข้อจำกัด ของ Completeness Constraints จะช่วยลดภาระให้ DBA จากความต้องการที่จะตรวจสอบ Real World สำหรับ Tuple ใน View ของ Completeness Constraints ที่กลายเป็นจริงและควรจะถูกเพิ่มเข้าในฐานข้อมูล

สรุป Completeness Constraints ที่ไม่เคยกำหนด Tuple ที่เป็นเท็จ จะถูกตรวจสอบโดยตัวระบบเอง ทุกๆ ความพยายาม ที่จะ Update ฐานข้อมูล จะถูก Reject ถ้ามันจะมีการลบออกจาก View ของ Completeness Constraints

จะเห็นแล้วว่า ระบบฐานข้อมูลจะต้องถูกเปิดเผยจำแนกแยกแยะ ซึ่ง Validity และ Completeness Constraints มีชนิดที่ได้แบ่งตามข้อจำกัดดังกล่าวนั้นมันควรจะตรวจสอบ การ Update ไปยัง View ที่สอดคล้องกับ Constraints เหล่านี้ เช่นเดียวกัน มันควรจะจำไว้ด้วยว่าใน บางกรณี สมมติฐานที่ว่า validity Constraints ไม่เคยระบุ Tuple ที่เป็นจริง และ completeness

constraints ไม่เคยระบุ tuple ที่เป็นเท็จ โดยตัวมันเองอาจมีความจำเป็นที่จะต้องถูกละเมิดได้เช่น “ nonstop flight from L to T “อาจถูกใช้ในอนาคตดังนั้น DBA ยังคงต้องรับภาระในการตรวจสอบเช่นเดิม

2.2 Constraints Management in Chimera

บทความนี้กล่าวถึงการป้องกันการละเมิดกฎความคงสภาพ ที่เกิดจากการ Update ซึ่งสามารถใช้วิธีการที่เป็นแบบฉบับและปฏิบัติกันมา ใน 2 วิธีการคือ ทำการตรวจเช็ค กฎความคงสภาพของข้อมูลที่ถูกอ้างถึงหรือมีผลจาก Transaction ก่อนที่จะทำการ Commit หรือทำการแก้ไขการกระทำของ Transaction เมื่อเกิดการละเมิด หรือไม่ก็ให้ DBMS ทำการตรวจเช็คการละเมิดอย่างอิสระและทำการ Roll Back เมื่อพบว่ามี Transaction ใดไม่ถูกต้อง แต่จะเห็นว่าวิธีการอันหลังค่อนข้างจะไม่เหมาะสมเมื่อต้องการควบคุมหรือบังคับกฎความคงสภาพ ที่ซับซ้อนมาก ๆ

การควบคุมกฎความคงสภาพที่มีความซับซ้อนนั้นได้มีการนำเอา Triggers มาช่วยเพื่อสร้างควบคุม (Enforcement) และทำการตรวจเช็ค ซึ่งการควบคุมด้วยวิธีนี้แบ่งออกได้เป็น 2 แบบคือ Immediate และ Deferred

Immediate Trigger เป็น Trigger ที่ถูก Execute เมื่อจบคำสั่ง ใน Transaction ที่มีการ Activated มัน

Deferred Trigger เป็น Trigger ที่ถูก Execute เมื่อ Transaction ที่ทำการ Activated มันมีการทำคำสั่ง Commit หรือ Savepoint

ในการใช้ Trigger ตรวจเช็คการละเมิดกฎความคงสภาพนี้เป็นวิธีที่ถือว่ายืดหยุ่น โดยยอมให้ Transaction มีการแก้ไขแทนที่จะทำการ Abort Transaction ดังกล่าวไป

2.3 Security Constraints Processing During The Update Operation In A Multilevel Secure Database Management System

บทความนี้กล่าวถึงกฎความคงสภาพ ที่สามารถถูกบังคับและควบคุมความถูกต้องของข้อมูลในฐานะที่เป็น Integrity Rule หรือ Derivation Rule หรือ Schema Rule อย่างใดอย่างหนึ่งโดยที่

กฎความคงสภาพที่บังคับและควบคุมความถูกต้องขณะที่ทำการประมวลผล Query เราจะเรียกว่าเป็น Derivation Rule

ส่วนกฎความคงสภาพที่บังคับและควบคุมความถูกต้องของข้อมูลขณะทำการประมวลผล Update เราจะเรียกว่าเป็น Integrity Rule

และสุดท้ายกฎความคงสภาพที่บังคับและควบคุมความถูกต้องของข้อมูล ขณะที่ทำการออกแบบฐานข้อมูลเราจะเรียกว่าเป็น Schema Rule

สำหรับกฎความคงสภาพที่ใช้ควบคุมการทำงานนั้นเราจะแบ่งได้เป็น

1 Simple Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูลโดยทำการแยก Database, Relation , และAttribute ออกมาให้ชัดเจน (ส่วนใหญ่จะทำที่การออกแบบ)

2 Association-base Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูล ตามความสัมพันธ์ระหว่างข้อมูลเช่น Tuple, Attribute, Element และอื่นๆ

3 Content-base Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูลกับส่วนใดส่วนหนึ่ง ซึ่งขึ้นกับบางค่าของข้อมูลในฐานข้อมูลนั้น

4 Event-base Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูลกับส่วนใดส่วนหนึ่งของข้อมูล เมื่อเกิดเหตุการณ์บางอย่างขึ้น ซึ่งมีโอกาสจะละเมิดความถูกต้องของ Database

5 Release-base Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูลเมื่อมีบางส่วนของข้อมูล เกิดการเปลี่ยนแปลงขึ้น ส่วนข้อมูลที่เกี่ยวข้องกับข้อมูลดังกล่าวจะต้องถูกเปลี่ยนแปลงตามไปด้วย

6 Aggregate Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูลที่เกิดจากการรวมกันของข้อมูล ตามความต้องการของระบบ

7 Level-base Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูลในมุมมองระดับความปลอดภัยในการใช้ข้อมูล

8 Fuzzy Constraints เป็นการควบคุมโดยมีการ ให้ค่า Fuzzy กับข้อมูลที่ทำการแบ่งไว้

9 Logical Constraints เป็น การควบคุมโดยมีการกำหนด ค่าของข้อมูลที่มีความเกี่ยวข้องโดยนัย

10 Complex Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูลที่มีการใช้ข้อมูลที่มีความสัมพันธ์มากกว่า 1 กลุ่มในฐานข้อมูล

จากกฎความคงสภาพทั้ง 10 ข้อข้างต้น จะมีข้อพิจารณาคือในส่วนของ ข้อ 8 และ9 จะถือว่าไม่ใช่การจัดแบ่งกฎความคงสภาพจะถือว่าเป็นการกำหนดเงื่อนไขเท่านั้น และสำหรับมุมมองในข้อ 7 นั้นจะถือว่าเป็นส่วนของการกำหนด Security ให้กับข้อมูลในระบบ

จะเห็นว่ากฎความคงสภาพส่วนใหญ่ยกเว้นกฎความคงสภาพบางตัวเช่น Release และ Aggregate Constraints นั้นจะสามารถถูกทำการประมวลผลเพื่อทำการตรวจสอบระหว่าง Operation การ Update ได้ แต่ถือว่ามันเป็นวิธีการที่มีความซับซ้อนอย่างมาก

กฎความคงสภาพที่ดูเหมือนว่าจะเหมาะสมในการจัดการควบคุมระหว่าง Operation ของการออกแบบกลุ่มของ Attribute หรือการรวม Attribute เข้าด้วยกัน คือ Structure-based Constraints ซึ่งเป็น การจัดแบ่งความสัมพันธ์ระหว่าง Attribute โดยความสัมพันธ์ดังกล่าวจะถูกกำหนดโดย Schema และ สามารถจัดการเมื่อ Schema ถูกกำหนดและสร้างขึ้น

อย่างไรก็ตามบางกฎความคงสภาพสามารถที่จะถูกจัดการควบคุมมากกว่า 1 ทาง เช่น Content-base Constraints สามารถจะถูกจัดการควบคุมระหว่างการประมวลผล Query หรือระหว่างการปรับปรุง Database แต่เราก็ไม่มีความจำเป็นที่จะต้องไปจัดการควบคุมกฎความคงสภาพนั้นในหลายแห่ง เช่น ถ้าควบคุมจัดการที่การ Update ฐานข้อมูล แล้วก็ไม่จำเป็นต้องไปควบคุมในระหว่างการทำ Query และจะเห็นได้อีกว่า Operation การ Query ถูกจัดทำและเรียกใช้บ่อยมากกว่าการ Update ซึ่งทำให้เราสามารถลดการทำงานที่กระทำโดย Query Processor ได้มากที่สุดเท่าที่จะเป็นไปได้เพื่อเพิ่มประสิทธิภาพ แต่อย่างไรก็ตามมันอาจจะต้องทำการควบคุมกฎความคงสภาพระหว่าง Query Processing เนื่องจากว่าสภาพความเป็นจริงมันอยู่ในลักษณะ Dynamic ซึ่งทำให้เราไม่สามารถควบคุมและจัดการระหว่าง การออกแบบ ฐานข้อมูลได้หมด รวมทั้งบางระบบก็มีความแตกต่างกัน ในความต้องการในการควบคุม และตรวจเช็คด้วย ซึ่งเป็นความจำเป็นที่จะต้องมีการนำเอา Stored Procedure Trigger เข้ามาช่วยในการตรวจสอบ Constraints ดังกล่าว

2.4 Associative Hardware and Software Technique for Integrity Control

บทความนี้ได้กล่าวถึงวิธีการในการจัดการดูแลและควบคุม กฎความคงสภาพของฐานข้อมูลให้คงอยู่ โดยมีความจำเป็นที่จะต้องกำหนดกลุ่มของกฎความคงสภาพโดย กฎความคงสภาพเหล่านี้ จะเป็นกฎที่เกี่ยวข้องกับ การจัดการฐานข้อมูลที่ถูกใช้เพื่อให้มั่นใจในความถูกต้องทาง Semantic ตลอดจนความถูกต้องของฐานข้อมูล และเกิดความตรงกันของข้อมูลเมื่อมีการ Update

ในระบบฐานข้อมูลระยะแรก ๆ ผู้ใช้จะคุ้นเคยกับการจัดการกับกฎความคงสภาพของฐานข้อมูลในส่วนของ Application ที่เราสร้างขึ้น แต่ในปัจจุบันนี้งานดังกล่าวถือเป็นหน้าที่ของ DBMS โดยกฎความคงสภาพจะถูกบรรจุอยู่ในส่วนของระบบและถูกใช้เพื่อทำการตรวจสอบกับ Operation ที่เกี่ยวกับฐานข้อมูลซึ่งมีความสอดคล้องกับกฎความคงสภาพนั้น และถ้ามี Operation ใดมีการละเมิดกฎความคงสภาพขึ้น จะทำให้ Operation นั้นถูกปฏิเสธการทำงานจากระบบ

ปัญหาใหญ่ในการบังคับความถูกต้องของฐานข้อมูลในระบบทั่วไปคือ ต้องเสียค่าใช้จ่ายสูงในส่วนของการประเมินกฎความคงสภาพของระบบเพราะการประเมินกฎความคงสภาพเหล่านี้ต้องการการเคลื่อนย้ายข้อมูล (ซึ่งไม่เกี่ยวข้องกับ Operation ของฐานข้อมูล) จาก Secondary Storage Device ไปไว้ใน Main Memory การเคลื่อนย้ายข้อมูลระหว่าง Main Memory และ

Secondary Storage Device ถือว่าเป็นงานที่ใช้เวลามาก โดยเฉพาะเมื่อขนาดของข้อมูลมีจำนวนมาก ๆ

อีกปัญหาหนึ่งก็คือในการการบังคับความถูกต้องของข้อมูล คือบางกฎความคงสภาพสามารถทำได้ดีถ้าถูกตรวจสอบหลังจาก Operation ได้ถูกทำไปแล้วเพราะเนื่องจากว่า ความไม่เป็นอิสระของ Database Operation ในส่วนของ Database เราจึงมี 2 วิธีพื้นฐานที่ถูกใช้บังคับ วิธีแรกทำ Operation กับ Original Copy ของ file ที่เกี่ยวข้อง และทำการ Recovery (Back out Database Operation) ถ้าพบว่ากฎความคงสภาพถูกละเมิด อีกวิธีหนึ่งก็คือทำ Operation บน Copy ที่เกี่ยวข้องกับ File และลบ Copy ทิ้งถ้า Operation มีการละเมิดกฎความคงสภาพหรือถ้า Commit ก็จะทำการ Merge ส่วนที่มีการปรับเปลี่ยนเข้ากับ File หลักเพื่อให้การเปลี่ยนแปลงอยู่ในสถานะที่เหมาะสม

กลุ่มของกฎความคงสภาพ ซึ่งกำหนดความถูกต้องของ ฐานข้อมูล (Semantic Accuracy and Consistency) แบ่งแยกได้เป็น 2 ประเภท คือ Assertion และ Trigger

1 Assertion เป็นการกำหนดคุณสมบัติในสถานะของฐานข้อมูล โดยทุก ๆ คุณสมบัติที่เกี่ยวข้องกับฐานข้อมูลต้องถูก Satisfied โดย Operation ของฐานข้อมูล ซึ่งจะไม่ยอมให้มีการทำงานถ้าพบว่า มีการละเมิดคุณสมบัติอันใดอันหนึ่งที่กำหนดไว้

2 Trigger หรือ Alerter เป็นการกำหนดการกระทำ(กำหนดกระบวนการและวิธีการ) ที่จำเป็นสำหรับการ Update File อื่น ที่เกี่ยวข้อง ถ้าส่วนใดส่วนหนึ่งของฐานข้อมูลถูกปรับแปลง ซึ่งส่วนดังกล่าวมีความเกี่ยวข้องกับ Files อื่น ๆ

นอกจากกฎความคงสภาพทั้ง 2 ได้แบ่งแยกไปนั้น ยังมีการมองในส่วนของ การบังคับกฎความคงสภาพ ซึ่งสามารถแบ่งได้เป็น

1 Intratuple Constraints Enforcement เป็นการบังคับและควบคุมตามกฎความคงสภาพซึ่งขึ้นกับการกำหนดเงื่อนไข หรือการกำหนดการกระทำของแต่ละ Tuple

2 Intertuple Constraints Enforcement เป็นการบังคับและควบคุมตามกฎความคงสภาพที่เกิดจากความสัมพันธ์ระหว่าง Tuple

3 Aggregate Constraints Enforcement เป็นการบังคับและควบคุมตามกฎความคงสภาพ ที่กำหนดกับข้อมูลที่เป็นกลุ่มของ Tuple

กฎความคงสภาพของ ฐานข้อมูลสามารถสูญเสียไปโดยสาเหตุ 3 แนวทางคือ

1 มีการควบคุม การทำงานที่เป็นลักษณะ Concurrency ที่ไม่เพียงพอซึ่งจะส่งผลกระทบทำให้ Transaction ถูกรบกวนขณะประมวลผลซึ่งจะก่อให้เกิดปัญหาของ “Lost Update”

2 เกิดความล้มเหลว ในการป้องกัน ความไม่ถูกต้องของข้อมูลซึ่งอาจจะเกิดจาก ไม่มีความรอบคอบ มองปัญหาไม่ครบ มีการ Update ที่ผิด โดยคำสั่ง Query หรือ จาก Application program

3 โดยการ Update Files ที่เกี่ยวข้องอื่น ๆ ซึ่งเป็นเหมือนกับว่าส่วนใดส่วนหนึ่งของฐานข้อมูลถูก Update แต่ข้อมูลอื่น ที่เกี่ยวข้องไม่ได้ ถูก Update ให้สอดคล้องตามไปด้วย

ในการทำงานที่เป็น ลักษณะ Concurrency มีผู้ใช้หลายคน มีการ Query และใช้ฐานข้อมูลร่วมกัน จะมีวิธีป้องกันกฎความคงสภาพ เพื่อให้ฐานข้อมูลคงความถูกต้องตรงกันคือ ในเวลาหนึ่ง ๆ จะมีการยอมให้ผู้ใช้เพียงคนเดียวเท่านั้นที่สามารถทำการ Update ฐานข้อมูลที่ถูก Shared ซึ่งการทำเช่นนี้จะไม่ส่งผลให้ระบบการทำงานเกิด Deadlock ขึ้นได้

ในส่วนความล้มเหลวในการป้องกันความไม่ถูกต้องของข้อมูลที่เกิดจากผู้ที่ไม่มีสิทธิเข้ามาใช้ระบบจะถือว่าเป็นส่วนของ Data Security สำหรับการป้องกันในส่วนของ Semantically Incorrect Data ซึ่งเกิดจากความไม่ระมัดระวัง ไม่มีความรอบคอบหรือ เกิดจากการ Update ที่ไม่เหมาะสมของผู้ที่มีสิทธิ ซึ่งเป็นผลมาจาก Operation ที่ผิดพลาด และการป้องกันจากการ Execution โดยอัตโนมัติของ Trigger ที่จำเป็น เมื่อเงื่อนไขที่เกี่ยวข้องมีความสอดคล้องและถูกอ้างถึง ซึ่งกลไกในการป้องกันและตรวจเช็คสามารถแบ่งออกได้เป็น 5 Phase คือ Modification Phase, Assertion-Validation Phase , Back out Phase, Trigger Phase, Replacement Phase

2.5 Active Integrity Maintenance in Federated Database System

บทความนี้ได้กล่าวถึง Rule Execution ซึ่งประกอบด้วย 2 Phase โดย Phase แรกจะถูกกระตุ้นโดย เหตุการณ์ที่สอดคล้องกัน เงื่อนไขของกฎจะถูกประเมินและถ้าเป็นจริง Phase 2 ซึ่งเป็นการ Execution กฎที่กำหนดการกระทำจะเริ่มทำงาน

ทั้งการประเมินเงื่อนไข และการกระทำจะแสดงในขอบเขตของ Transaction ซึ่ง Transaction เหล่านี้จะเรียกว่า Triggered Transaction ในขณะที่ Transaction ซึ่งทำการกระตุ้นเหตุการณ์ให้เกิด จะเรียกว่า Triggering Transaction

สำหรับในส่วนของกฎความคงสภาพได้มีการกล่าวถึงดังนี้

Attribute Constraints เป็นข้อจำกัดของค่าใน Attribute ที่สามารถจะมีและเป็นไปได้ ซึ่งจะรู้จักในรูปประโยค Check และ Not Null

Key Constraints ถูกใช้ในการออกแบบความเป็นหนึ่งเดียวของ Real-World Entity ซึ่งโดยทั่วไปจะถูกใช้ เพื่อให้ผู้ใช้สามารถที่จะระบุความแตกต่างระหว่าง Object ออกจากกันโดยรู้ว่าเป็น Object ใด

Aggregate Constraints เป็นการตระหนักถึง Object เป็นกลุ่ม เช่น ผลรวมของค่าใช้จ่ายทั้งโครงการจะต้องไม่เกินจำนวนที่ตั้งไว้

Referential Integrity Constraints เป็นการกำหนดว่า Object ใด Object หนึ่งในฐานข้อมูลสามารถอ้างถึงข้อมูลอื่นได้โดยข้อมูลดังกล่าวจะต้องมีปรากฏอยู่ในฐานข้อมูลด้วย

Further Constraints เป็นการวางรูปแบบเอาไว้สำหรับกฎความคงสภาพอื่น ๆ ซึ่งอาจมีขึ้นนอกเหนือจากที่กล่าวมาและเป็นลักษณะเฉพาะของแต่ละงานตลอดจนการบังคับที่มีความซับซ้อนมาก ๆ

2.6 On The Feasibility of Checking Temporal Integrity Constraints

บทความนี้กล่าวถึงกฎความคงสภาพที่จำเป็นของ ทุก ๆ Application ทางด้านฐานข้อมูล โดยบทความนี้แยกกฎความคงสภาพออกเป็น 2 ชนิดนี้คือ

Static Constraints ซึ่งเป็นการอ้างถึงสถานะปัจจุบันของฐานข้อมูล

Temporal (Dynamic) Constraints ซึ่งเป็นอ้างถึงสถานะที่เป็นทั้ง อดีต อนาคต และ ปัจจุบันของข้อมูลในฐานข้อมูล

สิ่งที่จะเน้นคือ Temporal Constraints ซึ่งมีปรากฏอยู่ใน Application ต่าง ๆ มากมาย เช่น Financial Trading, Office Automation เป็นต้น กฎความคงสภาพดังกล่าวนอกจากจะตรวจสอบสถานะของข้อมูลในปัจจุบันแล้ว ยังต้องทำการตรวจเช็คประวัติทั้งหมด ไม่ว่าจะเป็นอดีต หรืออนาคต ซึ่งจะเห็นว่า ส่วนใหญ่แล้วจะเกี่ยวข้องกับเวลา

ข้อแตกต่างอย่างหนึ่งที่เห็นได้ชัด คือการละเมิดกฎความคงสภาพดังกล่าว อาจเกิดขึ้นโดยบางครั้ง ไม่ได้เกิดจากการเปลี่ยนแปลงของข้อมูลในฐานข้อมูลเลย และส่วนใหญ่การละเมิดดังกล่าวจะไม่สามารถทำการ Abort ได้ ซึ่งเราจะเรียกว่าเป็น Real-time Constraints สถานะการณ์เช่นนี้เราจะต้องพยายามทำการตรวจเช็คให้ได้ โดยได้มีการนำเอาการทำงานของ Trigger มาช่วยซึ่งอยู่ในรูปแบบของ

IF *Condition*

Then *Action*

สำหรับวิธีนี้การละเมิดกฎความคงสภาพจะถูกตรวจเช็ค และทำการเก็บไว้ในฐานข้อมูล นอกจากนั้นเรายังใช้บังคับควบคุมการทำงานในส่วนอื่นของระบบ ซึ่งมีความเกี่ยวข้องกับส่วนดังกล่าว เช่น การยืมหนังสือต้องส่งคืนภายใน 7 วัน ถ้าเกิน 7 วันโดยไม่มีการส่งคืนจะถือว่าเป็นการละเมิด Real - time Constraints และถ้ามีการยืมเกิดขึ้นอีกโดยการละเมิดดังกล่าวยังไม่มีการแก้ไขสิทธิในการยืมก็จะถูกตัดไป เป็นต้น

บทที่ 3

การนำเสนอ การแบ่งและควบคุม ความคงสภาพของข้อมูล

จากที่ได้ทำการศึกษาค้นคว้าหาข้อมูล จากบทความวิจัย โครงงานต่าง ๆ ตลอดจนพื้นฐานความรู้ที่ได้จากตำรา และข้อเสนอแนะจากผู้เชี่ยวชาญทางด้านฐานข้อมูล จึงทำให้เกิดแนวความคิดในการจัดแบ่ง รวมทั้งการบังคับกฎความคงสภาพของข้อมูลในแนวทางใหม่ซึ่งเห็นว่าน่าจะเกิดประโยชน์ต่อการพัฒนาระบบ และใช้เป็นแนวทางในการเปรียบเทียบสำหรับผู้ที่จะทำการพัฒนาระบบขึ้นมา ซึ่งแนวทางที่นำเสนอในการจัดแบ่งโดยใช้ข้อมูลความรู้ ตามที่ได้กล่าวข้างต้นมีดังนี้

3.1 A Proposed Constraints classification

Level 0

1 Validity Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูลโดยจะตรวจสอบฐานข้อมูลเพื่อ ไม่ให้ข้อมูลที่ระบบไม่ต้องการไปปรากฏอยู่ในฐานข้อมูล

2 Completeness Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูลโดยจะตรวจสอบฐานข้อมูลเพื่อ ให้ข้อมูลที่ระบบต้องการมีปรากฏอยู่ในฐานข้อมูล

Level 1

1 Static Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูล โดยตรวจสอบที่สถานะที่เป็นปัจจุบันของข้อมูลว่าต้องมีความถูกต้อง

2 Dynamic Constraints (Temporal) เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูล โดยตรวจสอบสถานะของข้อมูลทั้งในอดีต ในปัจจุบัน และในอนาคต ว่าจะต้องมีความถูกต้องเสมอ

Level 2

1 Assertion Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูลในส่วนที่ได้กำหนดเป็น คุณสมบัติต่าง ๆ ของระบบโดยจะต้องสอดคล้องกับ Operation ของฐานข้อมูล

2 Alerter Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูลโดยจะกำหนดการกระทำที่จำเป็น ต่อการปรับปรุงเปลี่ยนแปลงข้อมูล ถ้าส่วนใดส่วนหนึ่งของฐานข้อมูลมีการปรับปรุงเปลี่ยนแปลง

Level 3

1 Real-time-based Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูล ซึ่งขึ้นอยู่กับเวลาที่เปลี่ยนแปลงไป โดยในกรณีนี้การละเมิด Constraints อาจเกิดขึ้นโดยไม่ได้มีการปรับปรุงเปลี่ยนแปลงข้อมูลในฐานข้อมูลเลย

2 Structure-base Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูลโดยมองตามโครงสร้างของ Database Model เช่น Relational Database Model

3 Content-base Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูลกับส่วนใดส่วนหนึ่ง ซึ่งขึ้นกับบางค่าของข้อมูลในฐานข้อมูลนั้น

4 Event-base Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูลกับส่วนใดส่วนหนึ่งของข้อมูล เมื่อเกิดเหตุการณ์บางอย่างขึ้น ซึ่งมีโอกาสจะละเมิดความถูกต้องของข้อมูลในฐานข้อมูล

5 Release-base Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูลเมื่อมีบางส่วนของข้อมูล เกิดการเปลี่ยนแปลงขึ้น ส่วนข้อมูลที่เกี่ยวข้องกับข้อมูลดังกล่าวจะต้องถูกเปลี่ยนแปลงตามไปด้วย

6 Aggregate Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูลที่เกิดจากการรวมกันของข้อมูล ตามความต้องการของระบบ

level 4

1 Single Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูล โดยมีการใช้ข้อมูลที่มีความสัมพันธ์ใน 1 ตาราง

2 Complex (Multiple) Constraints เป็นกฎที่ตรวจสอบความถูกต้องของข้อมูล โดยมีการใช้ข้อมูลที่มีความสัมพันธ์กันมากกว่า 1 ตารางในฐานข้อมูล

จากที่ได้นำเสนอการแบ่งแยกความคงสภาพของข้อมูลนั้นสามารถสรุปเป็นตารางได้ดังนี้

ตารางที่ 1

แสดงการนำเสนอการแยกประเภทของความคงสภาพของข้อมูล

Level	Integrity Constraint Classification					
0	Validity			Completeness		
1	Static			Dynamic		
2	Assertion			Alerter(Trigger)		
3	Real-Time	Structure-Base	Content-Base	Event_Base	Release-Base	Aggregate
4	Single			Complex(Multiple)		

3.2 A Proposed Constraints Enforcement

Level 0 (Event Enforcement)

1 Immediate Enforcement เป็นการบังคับและควบคุมตามกฎความคงสภาพที่มีในระบบ ซึ่งจะถูกเรียกเมื่อ Transaction ที่ Constraints ดังกล่าวเกี่ยวข้องทำงานจบ Transaction

2 Deferred Enforcement เป็นการบังคับและควบคุม ตามกฎความคงสภาพที่มีในระบบซึ่งจะถูกเรียกใช้เมื่อ Transaction ที่ กฎความคงสภาพดังกล่าวเกี่ยวข้อง ทำการ Commit หรือ Savepoint

Level 1 (Relational Database Model Enforcement)

1 Intratuple Constraints Enforcement เป็นการบังคับและควบคุมตามกฎความคงสภาพซึ่งขึ้นกับการกำหนดเงื่อนไข หรือการกำหนดการกระทำของแต่ละ Tuple

2 Intertuple Constraints Enforcement เป็นการบังคับและควบคุมตามกฎความคงสภาพที่เกิดจากความสัมพันธ์ระหว่าง Tuple

3 Aggregate Constraints Enforcement เป็นการบังคับและควบคุม ตามกฎความคงสภาพที่กำหนดกับข้อมูลที่เป็นกลุ่มของ Tuple

จากที่ได้นำเสนอการประเภทการบังคับความคงสภาพของข้อมูลนั้น สามารถสรุปเป็นตารางได้ดังนี้

ตารางที่ 2

แสดงการนำเสนอประเภทการบังคับความคงสภาพของข้อมูล

Level	Integrity Constraint Enforcement		
0	Immediate	Defered	
1	Intratuple	Intertuple	Aggregate

หมายเหตุ

จากที่ได้นำเสนอประเภทการบังคับความคงสภาพของข้อมูลเป็น 2 ระดับนี้ สำหรับใน Level ที่ 1 ในส่วนของ Intertuple กับ Aggregate นั้นสามารถทำการบังคับความคงสภาพของข้อมูลที่เป็นทั้งลักษณะ 1 ตาราง (Relation) หรือมากกว่า 1 ตารางก็ได้

บทที่ 4

หลักการ การทำงานและการใช้งาน ของ Stored Procedure

ก่อนที่จะแสดงรายละเอียดในชนิดและหลักการ ของ Oracle จะได้กล่าวถึงกฎความคงสภาพที่เป็นพื้นฐาน ซึ่งได้มีการกล่าวถึงแล้วบางส่วนในบทนำ โดยส่วนนี้จะแสดงแนวทางการประกาศใช้ จากกฎความคงสภาพที่มีกำหนดไว้เป็นพื้นฐาน ตลอดจนวิธีการที่ใช้สร้างกฎความคงสภาพ ที่มีความซับซ้อนยิ่งขึ้นโดยการใช้ Stored Procedure และ Database Trigger

4.1 Oracle Declaration Constraints

ชนิดของกฎความคงสภาพมีการกำหนดไว้ดังนี้

Primary Key เป็นการทำกับคอลัมน์ โดย กำหนดว่า

- ค่า ของคอลัมน์ ต้องเป็น Unique
- ค่าของมันต้องมีปรากฏจะเป็น NULL ไม่ได้

ใน 1 ตารางสามารถมีค่า Primary Key ได้เพียงหนึ่งค่าเท่านั้น ซึ่งไม่ได้บังคับว่า ทุก ตาราง จะต้องมี แต่ถึงอย่างไรก็ยังถือว่ามันเป็นการยากที่จะทำการสร้าง ตารางโดยที่ไม่มี Key ในการกำหนดการเข้าถึงข้อมูลในตาราง สำหรับ Primary Key นั้นอาจเกิดจากการสร้างจากหลาย ๆ Attribute รวมกันซึ่งเราเรียกว่า Concatenated Key

บาง Application ยอมให้มีการ Update Primary Key ได้แต่บาง Application ก็ไม่ยอม สำหรับ Oracle นั้นยอมให้มีการเปลี่ยนแปลงโดยใช้ SQL Update Statement วิธีการตั้งชื่อ Primary Key ควรจะมีการตั้งชื่อเพื่อสนับสนุนแนวทาง เพื่อลด Debug และเวลาที่ใช้ ในการวิเคราะห์เหตุการณ์ ที่จะเกิดการละเมิดกฎความคงสภาพให้น้อยที่สุด

Not Null Constraints เป็นการป้องกันค่าใน คอลัมน์ จากการ Insert หรือ Update ด้วยค่า Null สำหรับ Not Null Constraints ได้แนะนำให้ทำการตรวจเช็คที่ Client เนื่องจากเหตุผลดังนี้

1 เป็นการลดปัญหาที่จะเกิดขึ้นในเรื่องของ Traffic

2 ถ้ามี Field ใน ตาราง มีการตรวจเช็ค Not Null มากกว่า 1 ตัว ที่ Server จะไม่สามารถระบุได้เลยว่า Field ใด ที่ต้องมีการใส่ค่าและเป็น Null ไม่ได้ แต่ถ้าทำการเช็คที่ Client เราสามารถทำการแสดง Error Message แจ้งได้

Unique Constraints เป็นกฎความคงสภาพที่เหมือนกับ Primary Key ยกเว้นแต่ยอมให้มีค่าเป็น Null ได้ (การให้ชื่อกฎความคงสภาพกำหนดได้ไม่เกิน 30 ตัวอักษร)

Check Constraints เป็นกฎความคงสภาพ ที่กำหนดรายการของค่าที่ไม่ต่อเนื่องในคอลัมน์ซึ่งได้ระบุไว้ โดยรายการของค่าอาจจะเป็น Literally Expressed หรืออาจจะถูกแสดงในรูปของ Mathematical Expression การเช็คจะมีทั้งที่เป็น Boolean types และที่เป็นการเช็คที่มีความซับซ้อน โดยการเช็คแบบที่มีความซับซ้อนนี้อาจจะเลือกใช้ Trigger มาช่วย

Foreign Key Constraints เป็นการแสดงแทน Referential Integrity ซึ่งอ้างถึงค่าของคอลัมน์ ที่มีปรากฏอยู่ในอีก คอลัมน์ ซึ่งคอลัมน์นี้อาจจะอยู่ในตารางเดียวกัน หรือต่างตารางกันก็ได้ ค่าที่ทำหน้าที่เป็นตัวควบคุมจะเป็น Parent Set, Parent Column, Parent Table ส่วนค่าที่ถูกควบคุมจะเป็นสมาชิกของ Child Set , คอลัมน์, หรือตาราง โดยจะมีการแสดงในรูป Parent-Child Relationship หรือใน รูปทั่วไปจะเป็น Master-Detail และ Independent-Dependent Relationship ซึ่งกฎความคงสภาพ ดังกล่าวจะมีกฎเกณฑ์ดังนี้

Child-Insert-Restrict

Child-Update-Restrict

Parent-Update-Restrict

Parent-Delete-Restrict

ส่วนกฎเกณฑ์ที่ไม่ต้องขึ้นกับ Referential Integrity คือ

Parent Insert

Child Delete

สำหรับค่า Foreign Key ใน แถว อาจจะมีคาเป็น Null ได้ถ้าไม่มีการบังคับโดยระบบแต่ถ้าระบบต้องการก็อาจจะกำหนดเป็น Not Null ได้

เมื่อเรากำหนดให้ตารางหนึ่งมี Foreign Key เพื่ออ้างถึงอีกตารางหนึ่งเราอาจจะประกาศว่าเป็น Delete Cascade คือถ้า Parent Row ถูกลบ ข้อมูลใน Row ของ Child ทุกตัวที่อ้างถึง Parent ดังกล่าวจะถูกลบหมด

4.2 PL/SQL

PL/SQL ย่อมาจาก "Procedural Language extension to Structured Query Language"เป็นภาษาที่มีความใกล้เคียงกับ ภาษา SQL แต่จะมีการเพิ่มโครงสร้างของการโปรแกรมเข้าไปด้วย ทำให้การโปรแกรมทำได้อย่างมีประสิทธิภาพซึ่งสามารถเขียนเหมือนกับวิธีของ 3GLโดยรวมการใช้คำสั่ง SQL เพื่อจัดการกับข้อมูลผ่าน Cursor นอกจากนี้ PL/SQL ยังมีประโยชน์อีกมากมายสำหรับการพัฒนาภาษาโปรแกรมในปัจจุบัน และต่อไปในอนาคตด้วย

PL/SQLสามารถสนับสนุนการออกแบบระบบในลักษณะ Object-Oriented และมีกลไกที่มีประสิทธิภาพสำหรับการตรวจเช็ค รองรับเหตุการณ์ที่ไม่ได้เกิดขึ้นเป็นประจำ (Exception Handlers) รวมทั้งการแก้ไขข้อผิดพลาด ที่เกิดขึ้น

PL/SQL เหมาะสมกับโครงสร้างแบบ Client - Server และเนื่องจาก PL/SQL ถูกใช้ทั้งในส่วนของ Database (สำหรับ Stored Procedure และ Database Trigger) และในส่วนของ Application Code (ใช้แสดง Logic ในรูปแบบต่าง ๆ) ซึ่งใช้ในการพัฒนาได้ทั้งในฝั่งของ Client และ Server อีกทั้งยังสามารถย้ายโปรแกรมจากองค์ประกอบของ Configuration หนึ่งไปอีกองค์ประกอบหนึ่งได้ โดยถ้าเห็นว่ามีองค์ประกอบอื่นต้องการใช้ ก็ทำการย้ายองค์ประกอบดังกล่าวไปไว้ยัง Server ซึ่งทั้ง PL/SQL Code และ Program ที่ใช้เรียก Function จะไม่มีการเปลี่ยนแปลงเลย

PL/SQL มีความสามารถในการกำหนดและ Execute PL/SQL Program Units เช่น Procedure , Function และ Package ซึ่งโดยทั่วไปแล้ว PL/SQL จะแยกเป็น

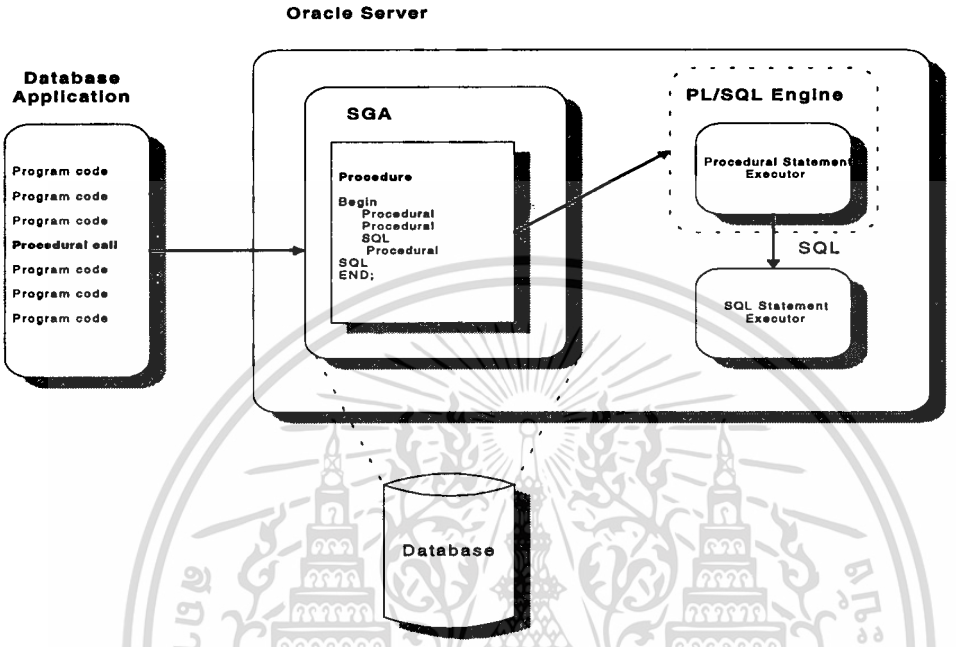
- Anonymous Block เป็น PL/SQL block ที่ปรากฏอยู่ในตัว Application และไม่มีการให้ชื่อหรือไม่ถูกเก็บในฐานข้อมูล

- Stored Procedure เป็น PL/SQL Block ที่ Oracle เก็บไว้ใน Database และถูกเรียกใช้โดยชื่อจาก Application

เมื่อมีการสร้าง Stored Procedure Oracle จะทำการ Parses (ตรวจเช็คไวยากรณ์) หลังจากนั้นก็จะเก็บตัวที่ผ่านการเช็คแล้วไว้ใน Database สำหรับ Oracle นั้นจะยอมให้มีการสร้างและเก็บ Function และ Procedure ในรูปของ Package ซึ่งเป็นกลุ่มของ Procedure และ Function ได้

ภาพที่ 2

PL/SQL Engine และ Oracle Server



จากภาพที่ 2 PL/SQL Engine ซึ่งถือว่าเป็นองค์ประกอบที่สำคัญของผลิตภัณฑ์มากมาย รวมทั้ง Oracle Server ที่ทำการประมวลผล PL/SQL ด้วย Procedure หรือ Function ที่ถูกเก็บใน Database นั้นเมื่อ Application เรียก Procedure ใน Database ตัว Oracle จะดึงเอา Procedure(หรือ Package)ที่ผ่านการ Compile แล้วมาไว้ในส่วนที่ถูก Share ซึ่งเป็นพื้นที่ส่วนรวมของระบบ(SGA (System Global Area)) จากนั้นตัว Execute คำสั่งของ PL/SQL และ SQL จะทำงานร่วมกันเพื่อที่จะประมวลผลคำสั่งใน Procedure

ผลิตภัณฑ์ของ Oracle ที่มีส่วนของ PL/SQL Engine รวมอยู่ด้วยคือ

- Oracle Server
- Oracle Form (ตั้งแต่ Version 3)
- SQL*Memu (ตั้งแต่ Version 5)
- Oracle Report (ตั้งแต่ Version 2)
- Oracle Graphics (ตั้งแต่ Version 2)

การเรียก Stored Procedure สามารถเรียกจาก PL/SQL Block อื่น ซึ่งอาจจะเป็น Anonymous Block หรือ Stored Procedure อื่นเช่นอาจจะเรียกจาก Oracle Form (ตั้งแต่ version 3)เป็นต้น

Anonymous Block ใน Application สามารถส่งผ่านไปยัง Oracle ด้วยเครื่องมือต่างๆ ดังนี้

- Oracle Pre-compilers
- Oracle Call Interfaces (OCIs)
- SQL*Plus
- Server Manager

4.3 Stored Procedure

Stored Procedure เป็น PL/SQL Program Unit โดยที่ Program Unit ที่สามารถใช้ PL/SQL สร้าง มีดังนี้

Stand -Alone Procedure

Stand- Alone Function

Package ซึ่งรวมเอา Procedure และ Function ไว้เป็นกลุ่ม

Trigger

4.3.1 Procedure และ Function

ส่วนของ Procedure และ Function จะประกอบด้วย

Declarative Part

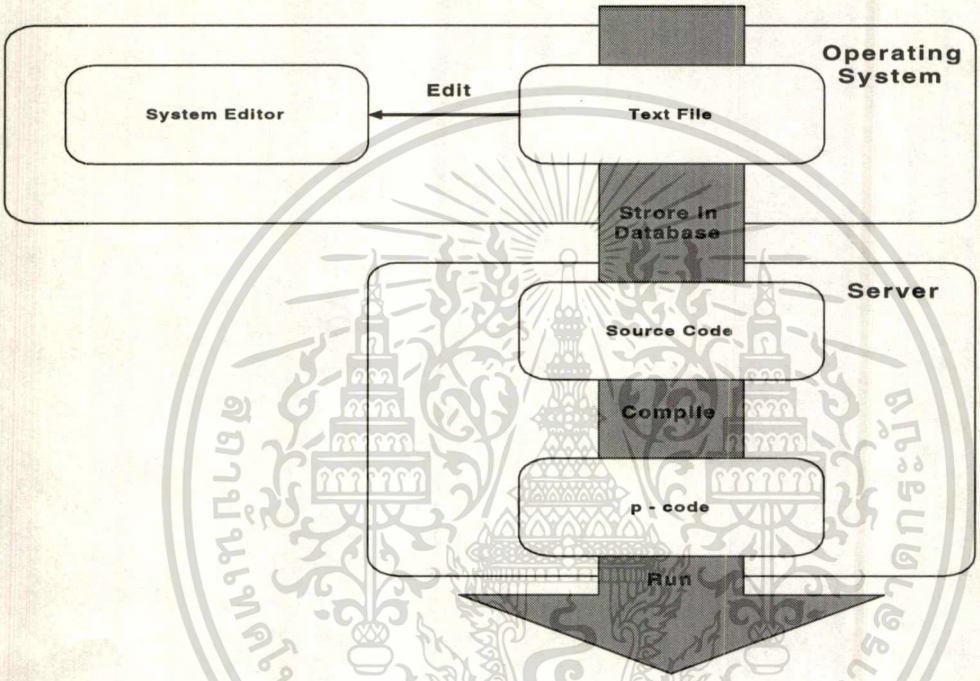
Subprogram Body

Exception Handling

ขั้นตอนการพัฒนา Algorithm ใน Stored Procedure

ภาพที่ 3

ขั้นตอนในการพัฒนา Stored Procedure ในส่วนของ Procedure และ Function



จากภาพที่ 3 แสดงขั้นตอนง่าย ๆ ในการพัฒนา Algorithm ใน Stored Procedure ซึ่งอธิบายได้ดังนี้

1 เขียน Procedure หรือ Function ที่จะสร้างในรูปของ Text File โดยการใช้ Text Editor หรือ Word Processor

2 ขณะที่มีการเรียบเรียงข้อความของ Procedure หรือ Function ที่สร้าง ควรตรวจเช็คข้อความที่อาจเกิด Run Time Error ไว้ล่วงหน้า

3 ทำการ Execute ข้อความใน Procedure หรือ Function ที่สร้างซึ่งจะมีการ Compile ตัว Source Code ไปเป็น P-code และจัดการเก็บทั้งตัว Source Code และ P-code ไว้ใน ฐานข้อมูล โดย SQL*Plus หรือ SQL*DBA

4 การเรียกใช้ Procedure หรือ Functionจะเรียกจาก Oracle Environment

ข้อแนะนำ

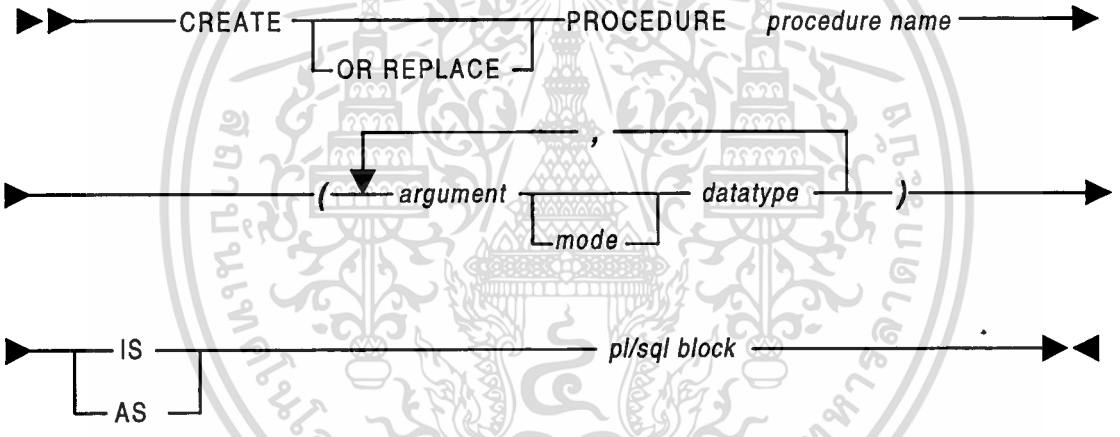
ในขณะ Compile ถ้ามี Error เกิดขึ้น P-code จะไม่ถูกเก็บไว้ในฐานข้อมูล และการเรียกใช้ Procedure ในขณะที่ P-code ไม่ได้ถูกเก็บอยู่ใน ฐานข้อมูล จะเกิด Runtime Error

Syntaxในการสร้าง

Syntax ในการสร้าง Procedure จะมีการประกาศตามรายการของ Argument และกำหนดการทำงาน โดย Standard PL/SQL Block ดังภาพที่ 4

ภาพที่4

Syntax ของการสร้าง Procedure



จากภาพที่4

procedure name คือ ชื่อของ procedure

argument คือ ชื่อของตัวแปร PL/SQL ที่ถูกส่งไปให้ procedure

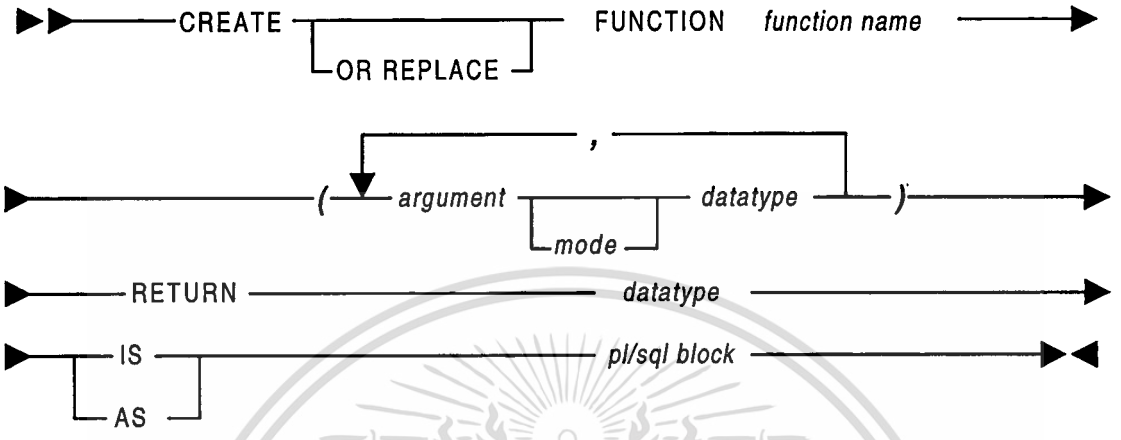
mode คือ การแสดงชนิดของ argument ซึ่งมี IN(default), OUT, IN OUT

datatype คือ ชนิดของข้อมูล ของ argument

pl/sql block คือ ส่วนของ procedural body ที่มีการปฏิบัติการตามที่กำหนดโดย procedure

ภาพที่ 5

Syntax ของการสร้าง Function



จากภาพที่ 5

Argument คือ ชื่อของตัวแปร PL/SQL ที่ถูกส่งไปใน Function

Mode คือ การแสดงชนิดของ argument ปกติจะเป็น IN argument

Datatype คือ ชนิดของข้อมูล ของ argument

RETURN คือ argument ที่ถูกส่งออกมาจาก Function

pl/sql block คือ ส่วนของ procedural body ที่มีการปฏิบัติการตามที่กำหนดโดย function

ความแตกต่างระหว่าง Stored Procedure กับ Oracle Forms Procedure

ตารางที่ 3

แสดงความแตกต่างระหว่าง Stored Procedure กับ Oracle Forms Procedure

Stored Procedure	Oracle Forms Procedure
• ถูกเก็บอยู่ในฐานข้อมูล • เอกสารอ้างอิงจะอยู่ใน Data Dictionary • ถูก Executed จาก Tool ทาง ฐานข้อมูลหรือ Application • ถือเป็น Stored Procedure ในฐานข้อมูลเพียงอย่างเดียว • จะมีการให้สิทธิกับผู้อื่นที่ต้องการใช้ (Grant)จากผู้ที่เป็นเจ้าของ Stored Procedure แต่ถ้าไม่มีการให้สิทธิ ผู้อื่นก็ไม่สามารถใช้ได้ โดยวิธีดังกล่าวถือเป็นวิธีการของ การกำหนดความปลอดภัยของข้อมูล • ถูกเรียกอย่างอิสระจาก Form Procedure	• ถูกเก็บอยู่ใน Oracle Form Application • เอกสารอ้างอิงถูกอ้างอิงใน Oracle Form Application • ถูก Execute จาก Oracle Form Application เท่านั้น • มีการอ้างถึง Oracle Form Application รวมทั้ง Database Stored Procedure ด้วย • จะมีการให้สิทธิการใช้โดยวิธีของ Form-Level Security และการ Copy • ถูกเรียกอย่างอิสระจาก Stored Procedure

ความแตกต่างระหว่าง Procedure กับ Function

Procedure อาจจะมีรายการของตัวแปร (Argument) บรรจุอยู่แต่จะไม่สามารถให้ค่ากลับไปได้ และการเรียกใช้ จะถือเป็นคำสั่งที่สมบูรณ์ใน PL/SQL 1 คำสั่ง

Function อาจจะมีรายการของตัวแปรและจะต้องมีการให้ค่ากลับคืนด้วย ในส่วนของการเรียกใช้จะอยู่เป็นส่วนหนึ่งของ PL/SQL Expression

การจัดการข้อยกเว้นขณะที่มีการทำงานใน Program (Exception Handling)

การจัดการชนิดต่าง ๆ ของ Runtime Exception จะยอมให้ทำการจัดการได้ทั้งในส่วนที่มีการเรียกใช้ หรืออาจจะถูกจัดการภายใน Procedure เลยก็ได้

โดย Exception จะแบ่งเป็น 2 ประเภทคือ Oracle Exception และ User-Defined Exception โดย Oracle Exception จะมีทั้งที่ประกาศไว้ก่อนแล้ว (Predefined Exception) และที่ประกาศขึ้นใช้เองด้วย ตารางต่อไปนี้จะแสดง ชนิดของ Exception ผลที่ต้องการให้เกิด และ วิธีในการทำ

ตารางที่ 4

แสดง Exception Handling

ชนิดของ Exception	ผลที่ต้องการให้เกิด	วิธีในการทำ
• Oracle	• ติดต่อสื่อสารข้อผิดพลาดในลักษณะที่มีการโต้ตอบกัน	• แยก Exception Handling ออกจาก Procedure
• User-Defined	• ติดต่อสื่อสารข้อผิดพลาดในลักษณะที่มีการโต้ตอบกัน	• ใช้ Raise_Application_Error Procedure
• Oracle	• แสดงโดยทำในฐานข้อมูล	• รวม Exception Handling ไว้ใน Procedure
• User-Defined	• แสดงโดยทำในฐานข้อมูล	• รวม Exception Handling ไว้ใน Procedure

ประโยชน์ที่ได้จากการใช้ Procedures และ Functions

นอกจาก Procedure และ function จะช่วยให้การพัฒนา Application ให้เป็นแบบ Module แล้ว ในส่วนของประโยชน์อื่น ๆ ที่มีสามารถจะแยกได้ดังนี้

เพิ่ม Data Security และ Integrity

◆ มีการควบคุมการเข้าถึงโดยอ้อมไปยัง Database Object จากผู้ที่ไม่มีสิทธิเข้าใช้

- งานที่มีความเกี่ยวข้องกันจะถูกรวมเป็น 1 ส่วนโดยมีการรวมเอาตาราง ที่มีความเกี่ยวข้องกันให้มาทำงานร่วมกัน ซึ่งจะทำให้เราทำการตรวจสอบความคงสภาพของข้อมูลได้สะดวกขึ้น

เพิ่มประสิทธิภาพ

- มีการหลีกเลี่ยงการตรวจเช็คไวยากรณ์ของผู้ใช้หลาย ๆ คน โดยมีการกระจาย SQL ที่มีการใช้ร่วมกันออกไป
- มีการหลีกเลี่ยงการตรวจเช็คไวยากรณ์ของ PL/SQL ขณะ Runtime โดยจะทำที่ Compile Time แทน
- ลดปริมาณการเรียกใช้ข้อมูลจากฐานข้อมูล และลด Network Traffic โดยการรวมคำสั่งไว้ด้วยกันใน 1 Transaction

ประหยัด หน่วยความจำ

- มีการจัดเก็บ Code ไว้อันเดียวในฐานข้อมูลซึ่ง เป็นการหลีกเลี่ยงการจัดเก็บที่ซ้ำซ้อนจาก Application ที่แตกต่างกัน
- มีการใช้ SQL ร่วมกันเพื่อหลีกเลี่ยงการใช้ Cursors หลายตัวจาก Application ที่แตกต่างกัน

เพิ่มการบำรุงรักษา(Maintenance)

- สามารถปรับปรุงงานในลักษณะที่เป็น Online โดยที่ไม่มีการขัดจังหวะการทำงานของผู้ใช้คนอื่น
- ในการปรับปรุงงาน 1 งานจะเกิดผลการเปลี่ยนแปลงกับ Application อื่น ๆ ที่เกี่ยวข้องทั้งหมด
- ในการปรับปรุงงาน 1 งานสามารถลดขั้นตอนในการตรวจสอบที่มีความซ้ำซ้อน หลาย ๆ ครั้งลงได้

4.3.2 Package

Package เป็นวิธีที่รวมเอา Procedure และ Function ที่มีวัตถุประสงค์เดียวกัน(Common Object)รวมเข้าไว้ด้วยกัน ซึ่งวัตถุประสงค์เดียวกันนี้อาจจะเป็นตารางในฐานข้อมูลหรือ Entity ที่กำหนดในฐานข้อมูลระดับ Physical

Package จะมีการจัดการใน 3 หน้าที่หลักดังนี้

1 มีการEncapsulation ข้อมูลและ Operations

2 มี่การทำ Information Hiding

3 มีกลไกสำหรับการประกาศ Data Object เช่น Table หรือ Records

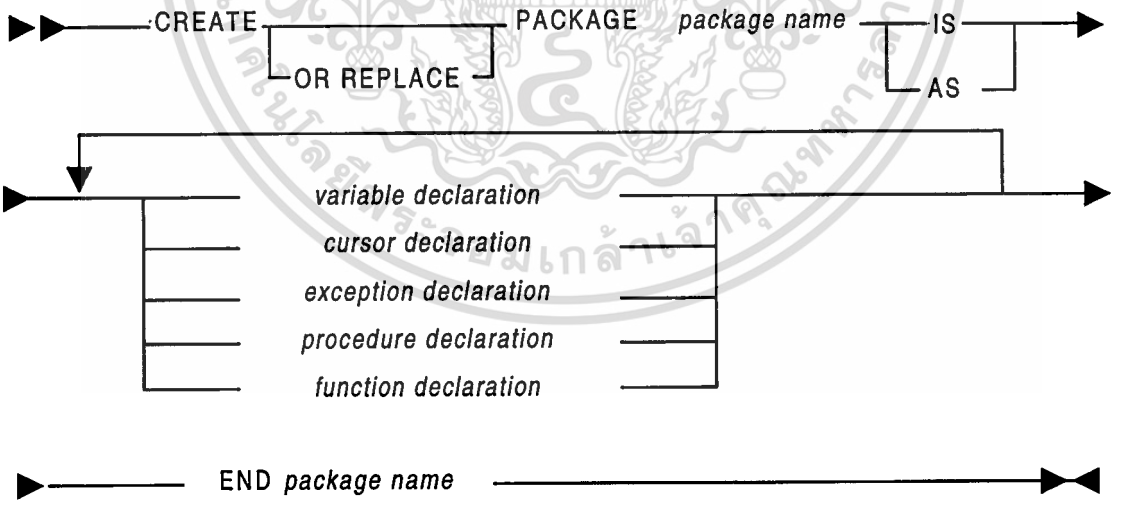
การสร้าง Package จะยอมให้ผู้พัฒนารวมเอา Procedures และ Functions ที่มีแนวทางและจุดมุ่งหมายเดียวกันมารวมไว้ด้วยกันตามความต้องการและตามแต่ที่ผู้พัฒนาจะกำหนด ถึงอย่างไรก็ตามมันก็อาจจะไม่มีความจำเป็นที่จะต้องสร้าง Package ที่มี Procedures และ Functions ที่เกี่ยวข้องกันเช่น อาจสร้าง Package ที่เป็น Text Message หรือ Error Code Number เป็นต้น

Package เป็น หน่วยของโปรแกรมที่ถูก Compiled ถูก Debugged และถูกตรวจสอบเช่นเดียวกับ Procedure หรือ Function เดี่ยว ๆ สำหรับโดยตัว Package เองนั้นมันจะไม่ถูก Executed ซึ่งส่วนที่ถูก Executed จะเป็น โปรแกรมย่อย(Subprogram) ที่บรรจุอยู่ใน Package

Package อาจจะประกอบด้วยเพียงแค่ Package Specification หรือ Package Specification รวมกับ Package Body ของมันก็ได้ โดยเมื่อ Package Specification ถูกสร้างโดยไม่มี Body สิ่งที่ได้ก็จะมีเพียง Data Type และการประกาศตัวแปรเท่านั้น ส่วน Package ที่มีทั้ง Package Specification และ Body สิ่งที่ได้ก็จะมีทั้ง Data Type การประกาศตัวแปรของข้อมูล รวมทั้ง Procedure Operation ด้วย

ภาพที่ 6

Syntax ของการสร้าง Package Specification



จากภาพที่ 6

package name คือ ชื่อของ Package

variable declaration เป็นการประกาศตัวแปรหรือ ค่าคงที่

cursor declaration เป็นการประกาศ cursor

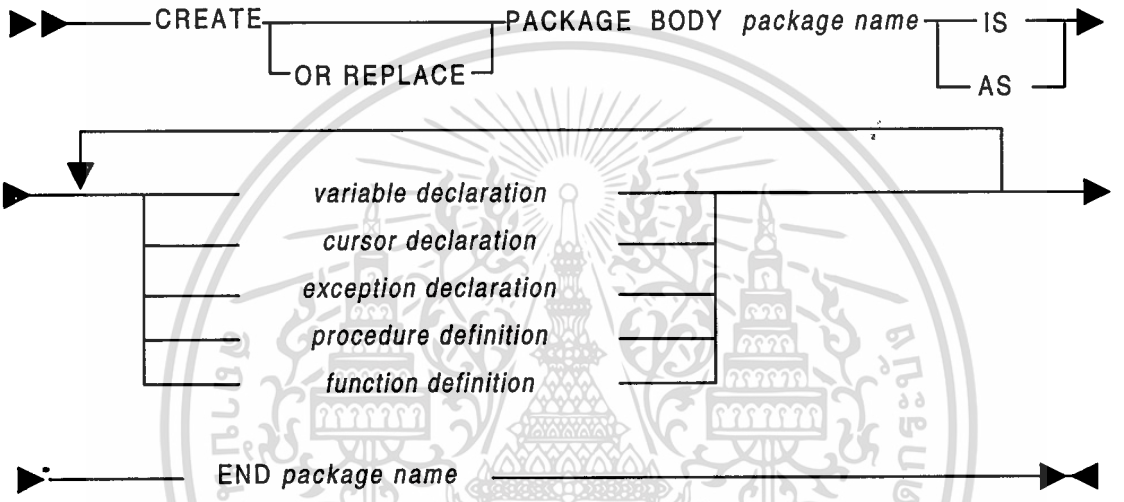
exception declaration เป็นการประกาศ ข้อยกเว้น

procedure declaration เป็นการประกาศการติดต่อกับ procedure

function declaration เป็นการประกาศการติดต่อกับ function

ภาพที่ 7

Syntax ของการสร้าง Package Body



จากภาพที่ 7

package name คือ ชื่อของ Package

variable declaration เป็นการประกาศตัวแปรหรือ ค่าคงที่

cursor declaration เป็นการประกาศ cursor

exception declaration เป็นกวรประกาศ ข้อยกเว้น

procedure definition เป็นการกำหนดการกระทำที่ถูกแสดงโดย procedure

function definition เป็นการกำหนดการกระทำที่ถูกแสดงโดย function

สรุปแล้วจะได้ว่า Package Specificationจะเป็นตัวกำหนดการติดต่อกับผู้ใช้ ส่วน Body จะเป็นส่วนที่บรรจุ Code ที่แสดงการทำงานตามที่ได้กำหนดไว้ใน Specification และสำหรับการ Compileส่วนของ Specification และ Body จะทำแยกจากกัน

ขั้นตอนการพัฒนา Package

1 เขียน Text File ในส่วนของ Package Specification

2 เขียน Text File ในส่วนของ Package Body

3 ทำการ Execute เพื่อ Compile จาก Source Code ไปเป็น P-Code และเก็บไว้ในฐานข้อมูล

4 เรียกใช้ Procedure และ Function ที่ทำการสร้างไว้ใน Package จากส่วนต่าง ๆ ผ่านตัวที่สามารถเรียกใช้ได้ เช่น SQL*Plus เป็นต้น

ประโยชน์ของ Package

นอกจากจะทำให้การพัฒนา Application เป็นลักษณะที่อยู่ในรูปของ Module แล้ว มันยังสามารถเกิดประโยชน์ในส่วนอื่น ๆ ด้วยคือ

เพิ่มการจัดการในการรวมกลุ่ม ของ Procedure และ Function

- มีการจัดกลุ่มของ Procedure และ Function ที่มีความสอดคล้องสัมพันธ์กัน รวมเข้าไว้ด้วยกัน
- เป็นการลดปัญหาในการตั้งชื่อ ที่อาจมีการตั้งชื่อซ้ำซ้อนกันใน Schema เดียวกัน

เพิ่มการจัดการกับส่วนของ Procedure และ Function

- การเปลี่ยนแปลง Body จะไม่มีผลกระทบกับส่วนของ Package Specification

เพิ่ม Security ของ Stored Procedure และ Function

- การให้สิทธิการให้ใช้ใน Function และ Procedure จะทำที่ Package เพียงครั้งเดียว
- มีการปกปิด Source Code จาก ผู้ใช้

เพิ่ม Performance

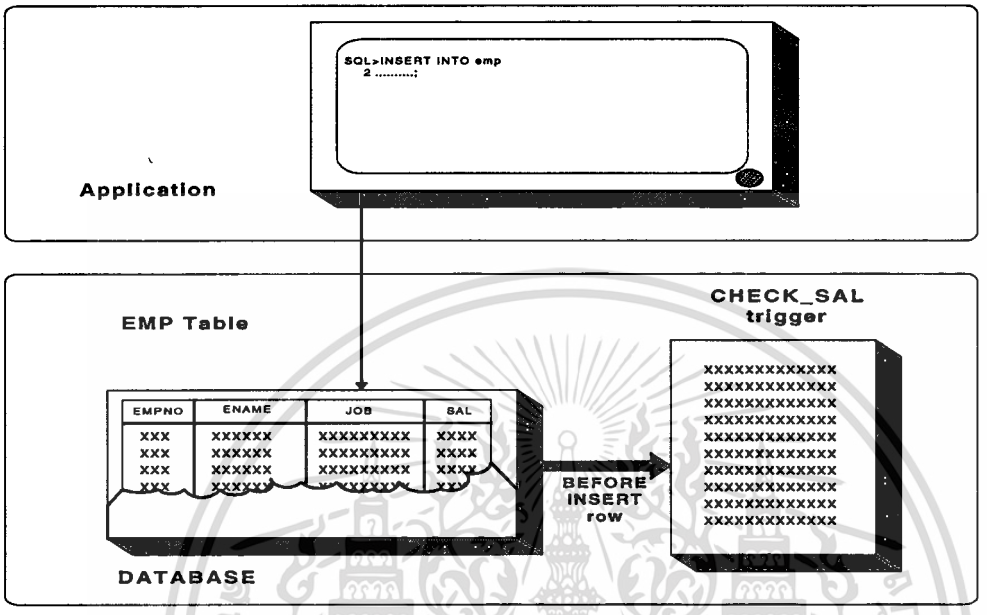
- มีการ Load Package ทั้งหมดมาไว้ใน Memory เมื่อมีการเรียกใช้ครั้งแรก
- ลดการเข้าถึง Disk สำหรับการเรียกที่เกิดขึ้นภายหลัง

4.3.3 Trigger

Trigger เป็น Stored procedure ที่ถูกเรียกเพื่อตอบสนองคำสั่ง SQLในส่วนของการ Insert , Update และ Delete ข้อมูล ในฐานข้อมูล คำสั่ง DML ที่ถูก Executed ภายใต้การตรวจสอบของ Trigger และจะถูก Rolled back ถ้าเกิด Run-time Error ใน Trigger หรือมีการ แจ้ง Error ใน Trigger

ภาพที่ 8

แสดงการเรียกใช้ database trigger ในการตรวจเช็คเมื่อมีการใช้คำสั่ง



สำหรับในระบบฐานข้อมูลใด ๆ สามารถที่จะใช้ Trigger ได้ใน 12 ลักษณะซึ่งใช้ความแตกต่างขณะที่มีการเรียกใช้เป็นตัวแบ่งแยก โดยจะมี 3 ลักษณะ ที่เกี่ยวข้องเมื่อ Trigger เข้ามาควบคุม คือ

- Action
- Level
- Timing

Action จะอ้างถึงคำสั่ง SQL ในการกระตุ้นนั่นคือคำสั่งในการ Insert, Update , Delete ซึ่ง Trigger 1 ตัวสามารถจะถูกออกแบบให้บังคับคำสั่ง SQL ที่รวมกันหลาย ๆ คำสั่งแต่ได้ แต่มันจะเกี่ยวข้องกับตารางได้เพียงตารางเดียวในฐานข้อมูลเท่านั้น

Level จะแบ่งเป็นระดับคำสั่ง กับระดับแถวของ Trigger ซึ่งในระดับของ Trigger จะอยู่ในระดับใดนั้น ก็ดูตรงที่มันมีการบังคับกับคำสั่งของ SQL ทั้งหมด หรือ มีผลกับ SQL โดยบังคับที่ละแถว

จะเห็นความแตกต่างที่ชัดเจนระหว่าง Statement-Level และ Row -Level Trigger คือ ความถี่ที่ถูก Executed โดย Statement -Level Trigger จะถูกบังคับต่อ 1 คำสั่ง SQL ส่วน Row -Level

Trigger จะถูกบังคับต่อ 1 แถว ซึ่งถ้าตารางมี จำนวนแถวมาก ๆ ก็อาจจะมีความจำเป็นที่จะต้องหลีกเลี่ยงการทำ Row - Level Trigger เพื่อเป็นการปรับปรุงประสิทธิภาพในการทำงาน

สำหรับความแตกต่าง ของทั้ง Statement และ Row Level Trigger คือ ระดับของการมองเห็น โดย Row- Level Trigger จะสามารถลงไปจัดการและตรวจสอบได้ในค่าของ คอลัมน์ในแต่ละแถว ได้ขณะที่มันมีการ Update นอกจากนี้ยังมีส่วนของค่า OLD Value และค่า NEW Value เข้ามาเกี่ยวข้องด้วย ซึ่งทั้งสองค่าจะรู้จักกันในฐานะที่เป็น Correlation Value ส่วน Statement -Level Trigger ไม่สามารถที่จะอ่านค่าของคอลัมน์ในแต่ละแถว เมื่อมันมีการ Update ได้ แต่อย่างไรก็ตามมันก็สามารถที่จะทำการตรวจสอบหลังการ Update โดยการตรวจดูค่าในแต่ละ Row ว่ามันมีการละเมิดกฎที่ได้ทำการกำหนดไว้หรือไม่ สำหรับแนวทางในการเลือกใช้ก็ขึ้นอยู่กับขอบเขต ของการบังคับควบคุม ของ Trigger ที่เราต้องการจะทำ

Timing เป็นการแสดงว่า Trigger จะทำการควบคุม ก่อน หรือ หลังจากที่มีการ Execute ใน Statement -Level หรือ Row - Level Trigger

จากลักษณะทั้ง 3 เราจะสามารถสรุปชนิดของการใช้ Trigger ใน 12 ลักษณะดังตารางต่อไปนี้

ตารางที่ 5

แสดงชนิดของการใช้ Trigger ใน 12 ลักษณะ

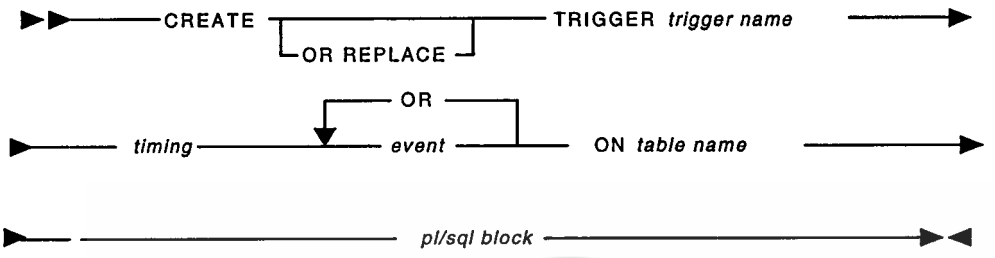
Insert Triggers	Update Triggers	Delete Triggers
• Before Insert Statement	• Before Update Statement	• Before Delete Statement
• Before Insert Row	• Before Update Row	• Before Delete Row
• After Insert Row	• After Update Row	• After Delete Row
• After Insert Statement	• After Update Statement	• After Delete Statement

การสร้าง database trigger มี 2 ลักษณะคือ

1.Statement trigger

ภาพที่ 9

Syntax ของการสร้าง Statement Trigger



จากภาพที่ 9

trigger name คือ ชื่อของตัว trigger

timing คือ การกำหนดหรือระบุเวลาเมื่อ trigger ถูกเรียกโดยมีค่าที่เป็นไปได้ 2 ค่าคือ BEFORE และ AFTER

event คือ การกำหนดหรือการจัดการกับข้อมูลที่เกิดจากการเรียกใช้ trigger โดยมีค่าที่เป็นไปได้ 3 ค่าคือ INSERT , DELETE และ UPDATE

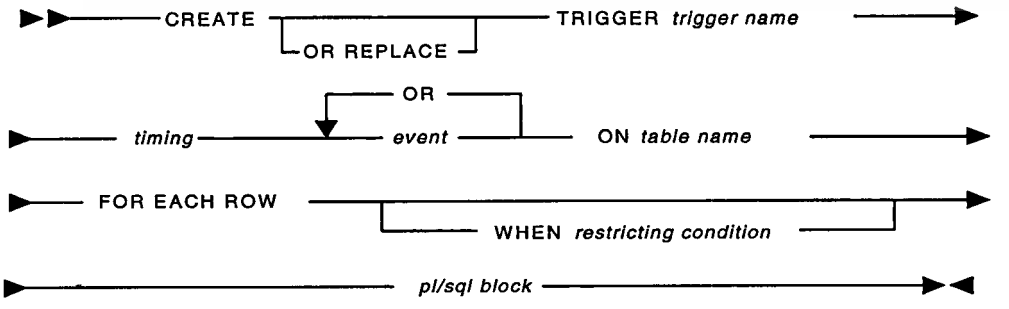
table name คือ การระบุตารางที่เกี่ยวกับ trigger

pl/sql block คือ ส่วนของ trigger body ที่ใช้กำหนดการทำงาน

2. Row Trigger

ภาพที่ 10

Syntax ของการสร้าง Row Trigger



จากภาพที่ 10

trigger name คือ ชื่อของตัว trigger

Timing คือ การกำหนดหรือระบุเวลาเมื่อ trigger ถูกเรียกโดยมีค่าที่เป็นไปได้ 2 ค่าคือ BEFORE และ AFTER

Event คือ การกำหนดหรือการจัดการกับข้อมูลที่เกิดจากการเรียกใช้ trigger โดยมีค่าที่เป็นไปได้ 3 ค่าคือ INSERT , DELETE และ UPDATE

Table name คือ การระบุตารางที่เกี่ยวกับ trigger

Restricting condition คือเงื่อนไขที่ระบุสำหรับแต่ละ row เพื่อกำหนดว่าส่วน trigger body จะถูก execute หรือไม่

pl/sql block คือ ส่วนของ trigger body ที่ใช้กำหนดการทำงาน

ความแตกต่างระหว่าง Statement และ Row Trigger

Statement Trigger

- ป้องกันการทำงานกับข้อมูลที่ไม่ถูกต้อง
- ตรวจสอบการทำงานที่เกี่ยวข้องกับข้อมูลในระบบ
-ควบคุมความปลอดภัยของการทำงานกับข้อมูล
- เริ่มต้นและตั้งค่าใหม่ให้กับ ตัวแปร Global และ Flags ของระบบ

Row Trigger

- ป้องกันความผิดพลาดของข้อมูลใน row
- ตรวจสอบ Row และค่าที่มีผลอันเกิดจากการทำงานกับข้อมูล
- ใช้ Global และ Flags ในการควบคุมการทำงานกับ Data
- มีการสร้างตารางซ้ำ(คัดลอกตาราง)
- มีการคำนวณค่าที่ได้รับเข้ามาก่อนจะใช้ในการตรวจสอบการทำงาน
- มีเปลี่ยนแปลงข้อมูลและแสดง Function โดยทางอ้อม

4.4 Correlation Values

จากที่กล่าวแล้วว่า Statement -Level Trigger ทำการบังคับในแต่ละ คำสั่ง ในขณะที่ Row Trigger บังคับในแต่ละแถวอันเกิดจาก SQL Statement ซึ่งในส่วนของ Statement -Level Trigger จะไม่สามารถรู้ได้ว่าจำนวนแถวมีการเปลี่ยนแปลงเป็นเท่าไรและมีผลกับแถวเป็นจำนวนเท่าใด แต่สำหรับ Row -Level Trigger เราสามารถรู้รายละเอียดในแต่ละ Column ก่อนที่แถวจะมีการเปลี่ยน

แปลง หรือแม้แต่ แถวได้มีการเปลี่ยนแปลงไปแล้วก็ตาม ซึ่งค่าของแต่ละ Column จะถูกระบุด้วย :OLD หรือ :NEW นำหน้าซึ่งเราเรียกว่า Correlation Values โดยที่

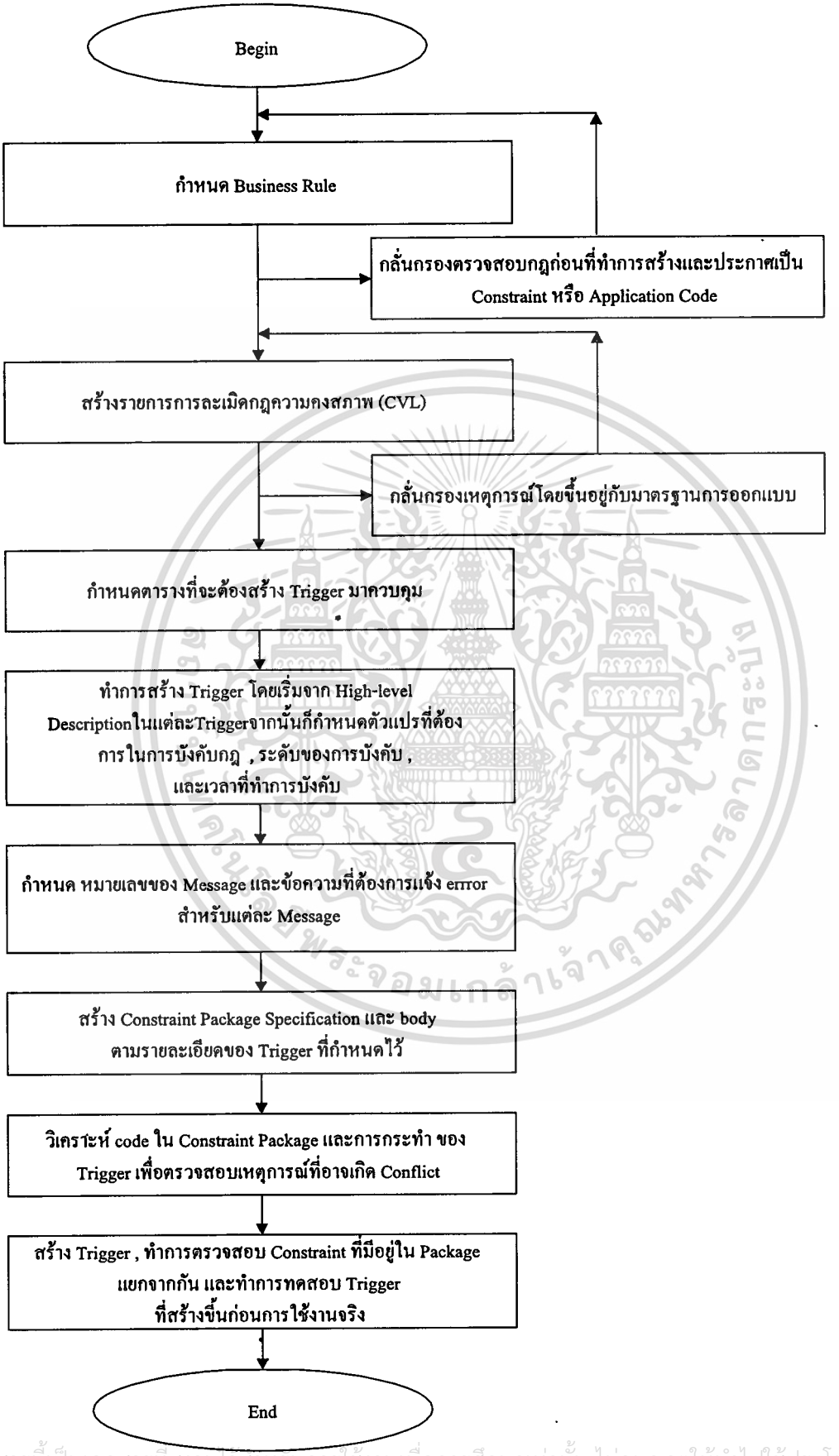
:OLD.Column_name เป็นค่าของ Column ในแถวก่อนที่จะมีการเปลี่ยนแปลงจากคำสั่งการ Update หรือค่าของ Column ก่อนคำสั่งการ Delete ค่านี้จะเป็น Null สำหรับการ Insert Statement

:NEW.Column_name เป็นค่าของ Column ใหม่หลังคำสั่งการ Insert หรือ การ Update ค่านี้จะเป็น Null สำหรับคำสั่งการ Delete

กระบวนการและขั้นตอนในการพัฒนา Trigger

ภาพที่ 11

แสดงกระบวนการและขั้นตอนในการพัฒนา Trigger



ข้อแตกต่างระหว่าง Database Trigger กับ Oracle Form Trigger

ตารางที่ 6

แสดงความแตกต่างระหว่าง Database Trigger กับ Oracle Form Trigger

Database Trigger	Oracle Form Trigger
● ถูก Execute โดยการกระทำจาก เครื่องมือของฐานข้อมูลใด ๆ หรือ Application	● ถูก Execute เพียงแค่ภายใน Oracle Form Application เท่านั้น
● ถูกกระตุ้นโดยคำสั่ง SQL Data Manipulation	● อาจถูกกระตุ้นจาก Field หนึ่งไปอีก Field หนึ่ง หรือโดยการกด Key หรือการกระทำอื่น ๆ ที่มีใน Form
● มีการแยกความแตกต่างไม่เป็น Statement ก็เป็น Row Trigger	● ไม่มีการแยกเป็น Statement หรือ Row
● ถ้าเกิดเหตุการณ์ล้มเหลว(failure) Triggering statement จะทำการ Roll Back เท่านั้น	● ถ้าเกิดเหตุการณ์ล้มเหลว(failure) อันเป็นผลให้ Cursor ไม่ทำงาน หรือทำการ Roll Back กับ Transaction ทั้งหมด
● ถูกควบคุมเป็นอิสระจาก Oracle Form Trigger	● มีการบังคับควบคุมอย่างอิสระจาก Database Trigger
● มีการ Execute ภายใต้ความเป็นเจ้าของ ใน Trigger นั้น ๆ	● มีการ Execute ภายใต้ขอบเขตของ Oracle Form ของแต่ละคน

การใช้ Trigger ในการบังคับ integrity

1.บังคับความถูกต้องของข้อมูล(Data Integrity)

- ◆ บังคับกฎเกณฑ์ความถูกต้องของข้อมูลอื่นๆ ที่ไม่ใช่กฎความถูกต้องพื้นฐาน
- ◆ ตั้งค่า Default ให้กับตัวแปร
- ◆ บังคับ Dynamic Constraints
- ◆ มีการตั้งค่าระหว่าง Enable และ Disable

2. บังคับ Referential Integrity

- ◆ มีการ Update เป็นลำดับขั้น (Cascade Update)

- ตั้งค่า Null ให้กับการ Updateและ Delete
- ตั้งค่า Default ให้กับการ Updateและ Delete
- บังคับ Referential Integrity ใน Distributed System
- มีการตั้งค่าระหว่าง Enable และ Disable

ประโยชน์ของ Database Trigger

ประโยชน์ของ Database Trigger นอกจากจะมีเช่นเดียวกับ Stored Procedure แล้วยังจะมีประโยชน์ทางด้านอื่น ซึ่งรวมถึง

เพิ่ม Data Security

- มีการตรวจเช็คความปลอดภัยของค่าของข้อมูล สำหรับการจัดการกับข้อมูล
- มีการตรวจสอบค่าของการจัดการข้อมูล

เพิ่ม Data Integrity

- มีการบังคับควบคุม Data Integrity Constraints แบบ Dynamic ในระดับของฐานข้อมูล
- มีการบังคับควบคุม Referential Integrity Constraints ที่ซับซ้อนมาก ๆ ทำให้เชื่อมั่นว่า การจัดการข้อมูลที่เกี่ยวข้องกันจะถูกกระทำร่วมกัน

ความแตกต่างที่สำคัญของ Database Trigger กับ Stored Procedure

ตารางที่ 7

แสดงความแตกต่างที่สำคัญของ Database Trigger กับ Stored Procedure(Procedure and Function)

stored procedure	Database Trigger
• มีการเรียกโดยตรงจาก application หรือ procedure	• มีการเรียกใช้โดยอ้อม
• คำสั่ง commit, rollback และ savepoint ถูกให้รวมอยู่ในตัว Procedure ได้	• คำสั่ง commit, rollback และ savepoint จะไม่ถูกรวมอยู่ในตัวของ trigger

สาเหตุที่ Database Trigger ไม่สามารถใส่ คำสั่ง Commit , Rollback และ Savepoint ลงไปได้ นั้นเนื่องจากว่า Database Trigger จะมีการทำงานระหว่าง Insert, Delete และ Update ซึ่งอยู่ในช่วงของการทำงานไม่เสร็จเรียบร้อย

ลักษณะการเก็บ

1 การเก็บของ Database Trigger

- เขียน CREATE TRIGGER Statement เป็น Text file
- Run CREATE TRIGGER Statement จาก Text file

a. ถ้า Compile Trigger สำเร็จ Source Code จะถูกเก็บไว้ใน Database แต่ P-codeจะไม่ถูกเก็บ

b. ถ้า Compile ไม่สำเร็จทั้ง Source Code และ Syntax error จะไม่ถูกเขียนลงใน Data dictionary แต่ error จะมีการแสดงออกมาให้ผู้ใช้ทราบทันที

- ทุกครั้งที่มีการเรียกใช้ Trigger จะมีเริ่มการ Compile ใหม่

2.การเก็บของ Stored Procedure

- เขียน CREATE PROCEDURE Statement เป็น Text file
- Run CREATE PROCEDURE Statement จาก Text file

a. ถ้า Compile Procedure สำเร็จ Source Code และ P-code จะถูกเก็บไว้ใน Database

b. ถ้า Compile ไม่สำเร็จทั้ง Source Code และ Syntax error จะถูกเขียนลงใน Data Dictionary

- เมื่อมีการเรียกใช้ Procedure จะไม่มีการ Compile ใหม่
- การ Recompile สามารถสั่งให้ทำได้

4.5 การจัดการตาราง กฎความคงสภาพ และ Stored Procedure(procedure, function, trigger)

4.5.1 การจัดการตาราง และ กฎความคงสภาพพื้นฐานที่มีใน Oracle นั้นเราสามารถใช้ Alter Table Command ในการจัดการโดยวิธีดังต่อไปนี้

1 ทำการเพิ่มคอลัมน์รวมทั้ง การประกาศ กฎความคงสภาพสำหรับคอลัมน์นั้นด้วย

2 ปรับปรุงคอลัมน์ ที่มีการกำหนดไว้ก่อนหน้านี้ รวมทั้ง กฎความคงสภาพโดยมีการจัดการภายใต้ข้อจำกัดดังนี้

- Datatype ไม่สามารถจะถูกเปลี่ยนแปลงถ้าคอลัมน์มีข้อมูลปรากฏอยู่ ซึ่งข้อมูลอาจจะมีปรากฏอยู่ในคอลัมน์อื่น ได้ การเปลี่ยนแปลงจะทำได้ถ้าในตารางไม่มีข้อมูลหรือใน คอลัมน์นั้นเป็น NULL
- ถ้าคอลัมน์ไม่มีข้อมูลอยู่ Datatype สามารถถูกเปลี่ยนแปลงได้แต่ไม่ควรที่จะทำการเปลี่ยนแปลงกับ ส่วนที่เป็น Unique และ Primary Key เพราะอาจเกิดความไม่ตรงกันกับ Foreign Key ที่อ้างถึงซึ่งอาจเกิดปัญหาในเวลา Run time ได้

3 มีการเปลี่ยนแปลงสถานะของกฎความคงสภาพระหว่าง Disable กับ Enable

ในส่วนการเปลี่ยนแปลง สถานะนี้ถ้า กฎความคงสภาพถูกกำหนดให้เป็น Disable แล้วจะทำให้ฐานข้อมูลสามารถละเมิด กฎความคงสภาพดังกล่าวได้ และเมื่อใดก็ตามที่ทำการ Disable แล้วเราจะไม่สามารถทำการกำหนดให้เป็น Enable ได้ ถ้าข้อมูลที่มีอยู่มีการละเมิดและไม่สอดคล้องกับกฎความคงสภาพดังกล่าว

ในส่วนของการ Disable Primary Key หรือ Unique Constraints จะต้องใช้ทางเลือก Cascade ร่วมด้วย เพราะอาจจะมี Foreign Key จากตารางอื่นอ้างถึงคอลัมน์ดังกล่าวได้ ซึ่งถ้ามีการใช้ ทางเลือกดังกล่าว มันจะทำการ Disable Foreign Key ไปโดยอัตโนมัติด้วย

นอกจากนี้ยังมีการจัดการกับ กฎความคงสภาพอีกรูปแบบหนึ่งคือเมื่อเราต้องการเอา กฎความคงสภาพนั้นออกจาก Data Dictionary ซึ่งเราจะต้องใช้ Drop Command เข้ามาช่วยโดยถ้ากฎความคงสภาพดังกล่าวถูก Drop แล้ว เราจะนำกลับมาใช้ใหม่ไม่ได้ แต่ถ้าต้องการจะต้องทำการสร้างใหม่เท่านั้น

4.5.2 การจัดการ Stored Procedure

4.5.2.1 Procedure และ Function

การจัดการ Procedures และ Functions ที่ถูกเก็บอยู่ในฐานข้อมูลพอสรุปได้ดังนี้

ตารางที่ 8

แสดงการจัดการกับ Procedure และ Function

งาน	คำสั่ง
• สร้าง Procedure หรือ Function ใหม่	• Create Procedure / function
• เปลี่ยนแปลง Procedure หรือ Function ที่มีอยู่	• Create Or Replace Procedure / Function
• ลบ Procedure หรือ Function ที่มีอยู่	• Drop Procedure / Function

4.5.2.2 การจัดการ Package

ตารางที่ 9

แสดงการจัดการกับ Package

งาน	คำสั่ง
• สร้าง Package Specification ใหม่	• Create Package
• สร้าง Package Body ใหม่	• Create Package Body
• เปลี่ยนแปลง Package Specification ที่มีอยู่	• Create Or Replace Package
• เปลี่ยนแปลง Package Body ที่มีอยู่	• Create Or Replace Package Body
• ลบ Package Specification ที่มีอยู่	• Drop Package
• ลบ Package Body ที่มีอยู่	• Drop Package Body

4.5.2.3 การจัดการ Trigger

ตารางที่ 10

แสดงการจัดการกับ Trigger

งาน	คำสั่ง
• สร้าง Trigger ใหม่	• Create Trigger
• เปลี่ยนแปลง Triggerที่มีอยู่	• Create Or Replace Trigger
• ลบ Triggerที่มีอยู่	• Drop Trigger

นอกจากนี้ Trigger ยังสามารถที่จะทำการ Disable และ Enable เช่นเดียวกับ Constraints โดยจะมีทางเลือก All เข้ามาช่วยในกรณีที่จะทำการ Disable หรือ Enable กับ trigger ทั้งหมดของตารางใดตารางหนึ่ง หรืออาจจะทำการ Disable หรือ Enable กับแต่ละ Trigger ก็ได้

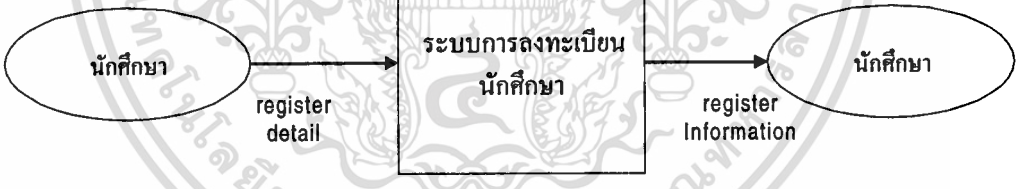
บทที่ 5
ระบบลงทะเบียนนักศึกษา

5.1 การออกแบบ

5.1.1 Context Diagram

ภาพที่ 12

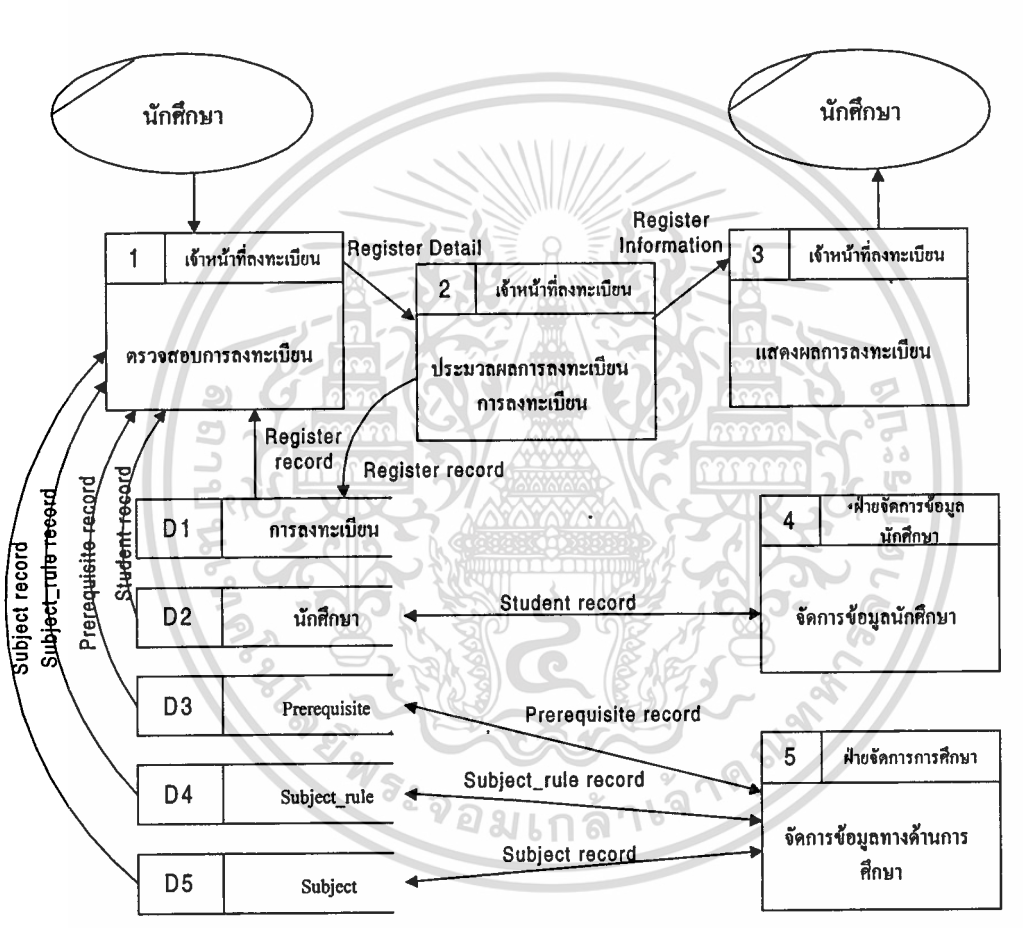
แสดง Context Diagram ของระบบลงทะเบียนนักศึกษา



5.1.2 Data Flow Diagram

ภาพที่ 13

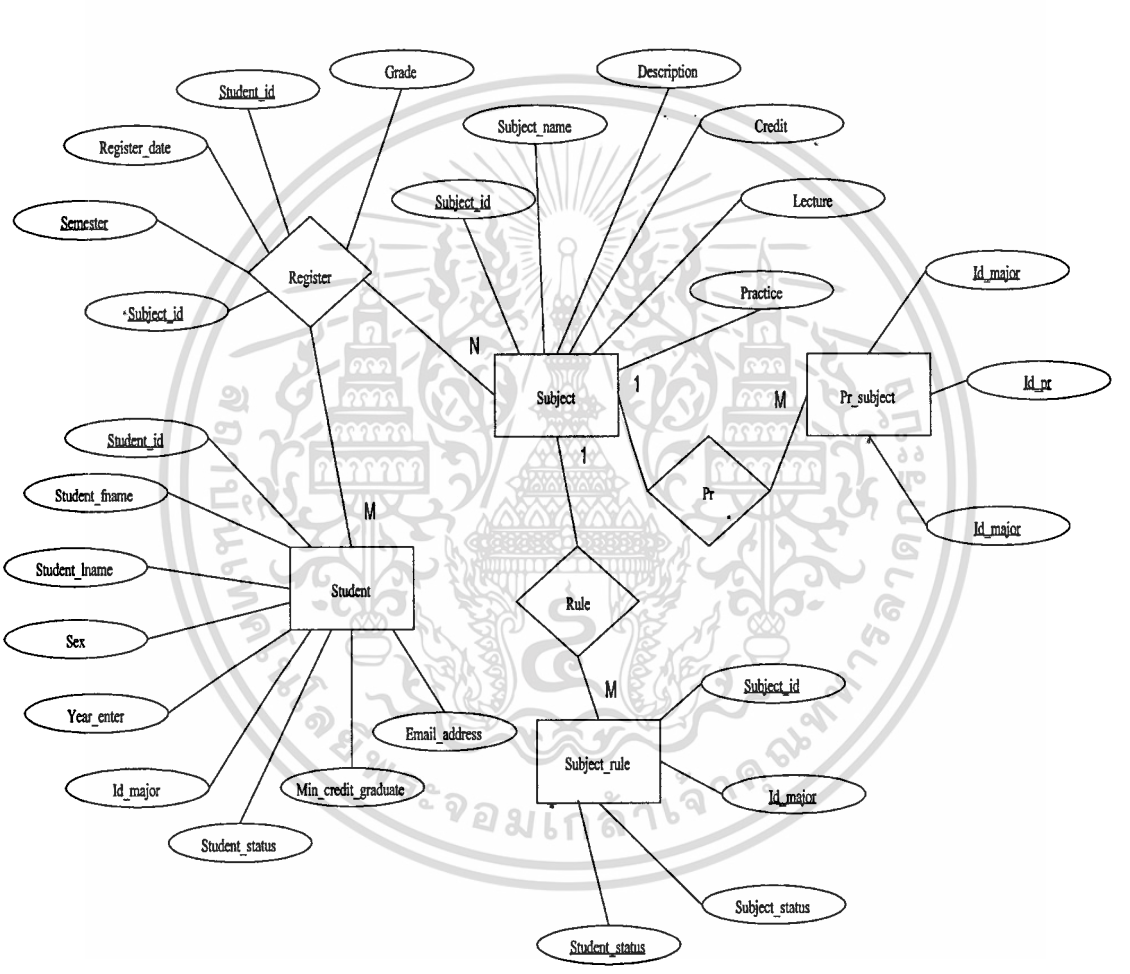
แสดง Data Flow Diagram ของระบบลงทะเบียนนักศึกษา



5.1.3 ER-Diagram

ภาพที่ 14

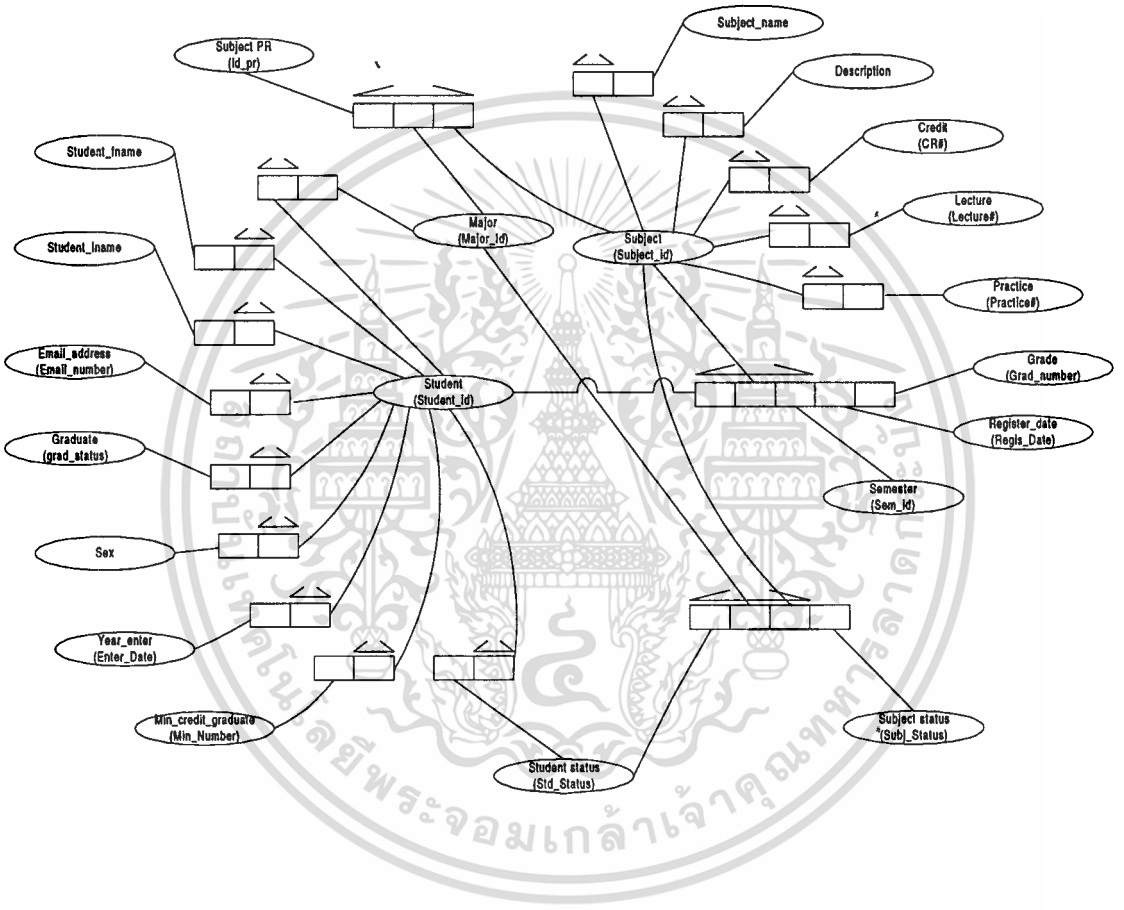
แสดง ER-Diagram ของระบบลงทะเบียนนักศึกษา



5.1.4 NIAM Diagram

ภาพที่ 15

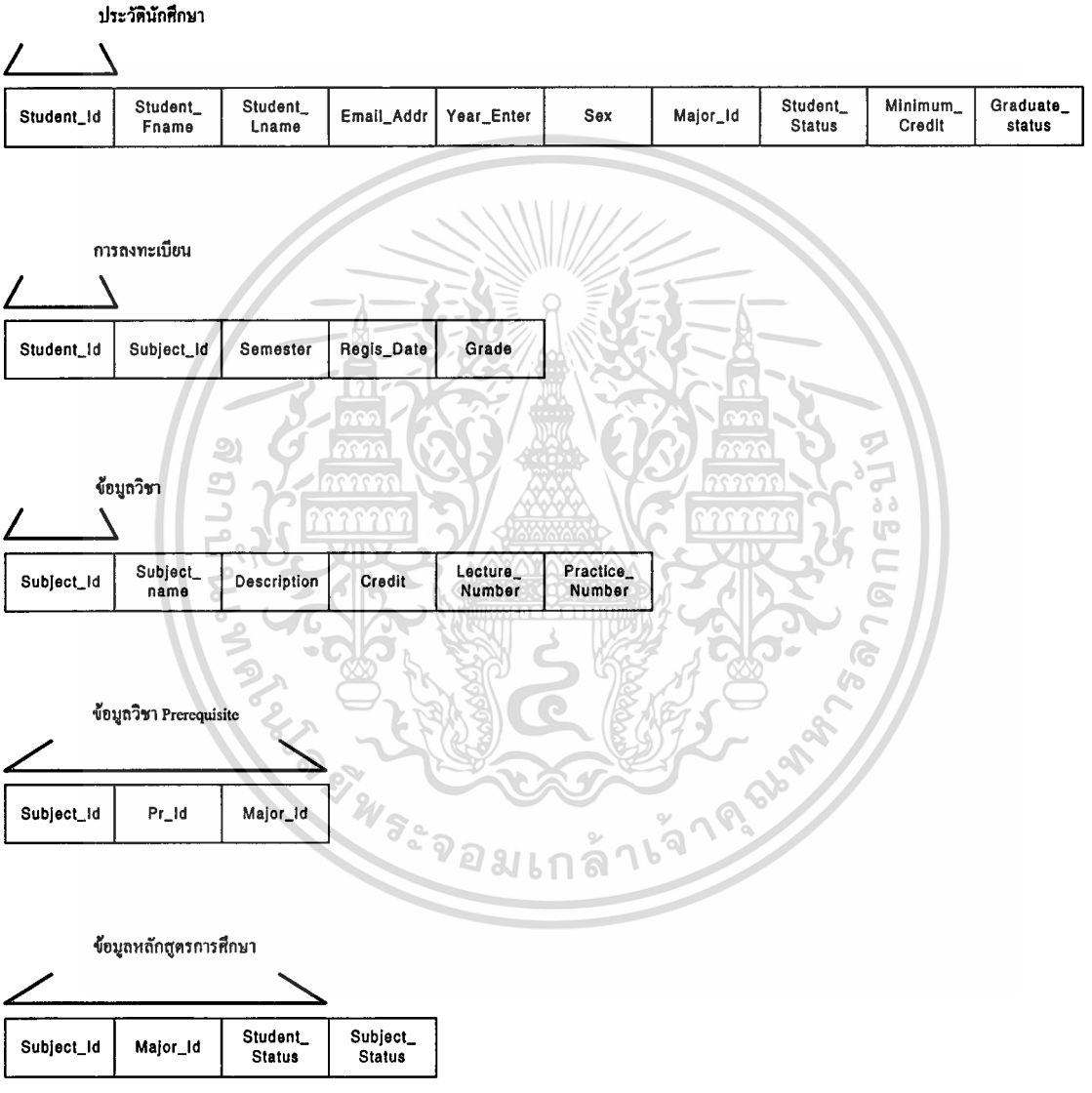
แสดง NIAM Diagram ของระบบลงทะเบียนนักศึกษา



5.1.5 Map จาก NIAM เป็น ตาราง

ภาพที่ 16

แสดงตาราง ของระบบลงทะเบียนนักศึกษา



5.1.6 รายละเอียดของคอลัมน์ในแต่ละตาราง

ตารางการลงทะเบียน

ตารางที่ 11

แสดงรายละเอียดของคอลัมน์ในตารางการลงทะเบียน

คีย์	ฟิลด์	ขนาด	ความหมาย	หมายเหตุ
PK	Semester	varchar(4)	ภาคการศึกษาที่ลง	
	Register_Date	date	วันที่ลงทะเบียน	
PK,FK	Student_Id	varchar(8)	รหัสนักศึกษา	Student
PK,FK	Subject_Id	varchar(8)	รหัสวิชา	Subject
	Grade	Number	ผลการศึกษา	

ตารางนักศึกษา

ตารางที่ 12

แสดงรายละเอียดของคอลัมน์ในตารางนักศึกษา

คีย์	ฟิลด์	ขนาด	ความหมาย	หมายเหตุ
PK	Student_Id	varchar(8)	รหัสนักศึกษา	
	Year_Enter	date	ปีที่เข้า	
	Student_Fname	varchar(40)	ชื่อ	
	Student_Lname	varchar(40)	นามสกุล	
	Sex	varchar(1)	เพศ	
	Email_Address	varchar(50)	ที่ติดต่อทางอิเล็กทรอนิกส์	
	Graduate_Status	char(1)	ระบุการจบการศึกษา	
	Major_Id	char(2)	สาขาวิชา	
	Student_Status	char(1)	แผนการศึกษา	
	Minimum_Credit	number	หน่วยกิตทั้งหมดของหลักสูตร	

ตาราง วิชา

ตารางที่ 13

แสดงรายละเอียดของคอลัมน์ในตารางวิชา

คีย์	ฟิลด์	ขนาด	ความหมาย	หมายเหตุ
PK	Subject_Id	varchar(8)	รหัสวิชา	
	Subject_Name	varchar(50)	ชื่อวิชา	
	Description	varchar(100)	หมายเหตุ	
	Credit	decimal(2,0)	หน่วยกิจ	
	Lecture	varchar(2)	บรรยาย	
	Practice	varchar(2)	ปฏิบัติ	

ตาราง Prerequisite

ตารางที่ 14

แสดงรายละเอียดของคอลัมน์ในตาราง Prerequisite

คีย์	ฟิลด์	ขนาด	ความหมาย	หมายเหตุ
PK,FK	Subject_Id	varchar(8)	รหัสวิชาที่จะลง	Subject
PK	Id_Pr	varchar(8)	รหัสวิชา ที่ต้องผ่านก่อน	Subject
PK	Major_Id	varchar(2)	รหัสแขนงวิชา	

ตาราง Subject_rule

ตารางที่ 15

แสดงรายละเอียดของคอลัมน์ในตาราง Subject_rule

คีย์	ฟิลด์	ขนาด	ความหมาย	หมายเหตุ
PK,FK	Subject_Id	varchar(8)	รหัสสาขาวิชา	Subject
PK	Major_Id	varchar(2)	รหัสแขนงวิชา	
PK	Thesis	varchar(1)	แผนการศึกษา	
	Subject_Status	varchar(1)	ประเภทของวิชา	

5.1.6 Constraints ของระบบ

Constraints ของระบบลงทะเบียนนักศึกษา

* Primary Key
* Check Constraints (Check range ,Check value ของเกรดนักศึกษาที่มีทั้งหมด)
* Not null Constraints
* Unique Constraints
* Foreign Key
* นักศึกษามีระยะเวลาศึกษาได้ไม่เกิน 5 ปี
* นักศึกษาจะลงวิชาต่อไปได้จะต้องผ่านวิชาที่เป็น prerequisite ก่อน
* หน่วยกิตเช็คจบจะถูกนับสะสมก็ต่อเมื่อเกรดที่ได้ต้องไม่ต่ำกว่า C+
* กรณีที่ได้เกรดต่ำกว่า C+ จะลงวิชาเดิมซ้ำได้เฉพาะวิชาที่เป็นวิชาบังคับเท่านั้น
* นักศึกษาจะทำการ Re grade ไม่ได้
* เกรดต่ำกว่า 3.00 แต่เกิน 2.5 ติด Pro
* เกรดต่ำกว่า 2.5 หมดสภาพความเป็นนักศึกษา
* นักศึกษาจะถูกปรับสภาพให้อยู่ในสถานะจบการศึกษาก็ต่อเมื่อ
  1 นักศึกษาต้องลงครบตามหน่วยกิตที่สาขานั้นกำหนดไว้ต่ำสุด
  2 นักศึกษาต้องได้เกรดเฉลี่ยมากกว่าหรือเท่ากับ 3.00
  3 นักศึกษาต้องลงวิชาบังคับตามที่หลักสูตรกำหนด

5.2 การเปรียบเทียบและจัดกลุ่มความคงสภาพของข้อมูลในระบบ

ในหัวข้อนี้จะเกี่ยวข้องกับการเปรียบเทียบและจัดกลุ่ม Constraints ของระบบเข้ากับ Constraints ที่ได้ทำการจัดกลุ่ม ตลอดจน การบังคับและควบคุม Constraints ดังกล่าว

จาก Constraints ของระบบดังกล่าว เมื่อนำมาเปรียบเทียบและจัดกลุ่มกับ Constraints ที่ได้นำเสนอการจัดแบ่งและการบังคับในบทที่ 3 แสดงในรายละเอียดได้ดังนี้

- ◆ Primary Key เป็นตัวที่กำหนดขึ้นเพื่อที่ทำให้เราสามารถที่จะระบุแถวในตารางได้
- ◆ Check Constraints (Check range ,Check value ของเกรดนักศึกษาที่มีทั้งหมด) เป็นการกำหนดขอบเขตของค่าข้อมูลใน คอลัมน์ ตามที่เราต้องการ
- ◆ Not null Constraints เป็นการกำหนดเพื่อตรวจเช็คค่าใน คอลัมน์ว่าจะเป็นค่า Null ไม่ได้ ซึ่งการกำหนดนี้จะทำให้คอลัมน์ดังกล่าวจะต้องมีการให้ค่าเมื่อมีการ ป้อนข้อมูล

- Unique Constraints เป็นการกำหนดเพื่อกำหนดว่า ค่าของคอลัมน์ในแต่ละแถวจะต้องไม่ซ้ำกันแต่จะยอมให้มีค่าเป็น Null ได้
- Foreign Key เป็นการกำหนดการอ้างถึงข้อมูลระหว่างตาราง ที่มีความเกี่ยวข้องสัมพันธ์กัน
- นักศึกษามีระยะเวลาศึกษาได้ไม่เกิน 5 ปี เป็นการกำหนดเงื่อนไขของการเป็นนักศึกษาว่าจะต้องสำเร็จการศึกษา ภายในระยะเวลา 5 ปีไม่เช่นนั้นจะหมดสภาพของการเป็นนักศึกษาไปโดยปริยาย
- นักศึกษาจะลงวิชาต่อไปได้จะต้องผ่านวิชาที่เป็น prerequisite ก่อน เป็นการกำหนดเงื่อนไขในการลงทะเบียน โดยการตรวจสอบจะทำการดูค่าในตาราง Prerequisite ว่าวิชาที่จะลงทะเบียนต่อไปนี้มีวิชาใดเป็น Prerequisite บ้างและถ้ามีก็จะนำเอาวิชา Prerequisite ของวิชาดังกล่าวไปตรวจสอบในวิชาลงทะเบียนว่ามีการลงทะเบียนวิชานี้ผ่านมาแล้วหรือยัง
- หน่วยกิตเช็คจบจะถูกนับสะสมก็ต่อเมื่อเกรดที่ได้ต้องไม่ต่ำกว่า C+ เป็นการกำหนดเพื่อตรวจสอบว่าวิชาที่ลงไปแล้วนั้นเกรดที่ได้มากกว่า หรือเท่ากับ C+ หรือไม่ และถ้าต่ำกว่าจะไม่มีการนับเป็นหน่วยกิตสะสมในการเช็คจบให้ โดยจะต้องทำการลงวิชาซ้ำหรือ ลงวิชาอื่นที่เทียบเท่าวิชาดังกล่าว
- กรณีที่ได้เกรดต่ำกว่า C+ จะลงวิชาเดิมซ้ำได้เฉพาะวิชาที่เป็นวิชาบังคับเท่านั้นเป็นการตรวจสอบต่อเนื่องจาก Constraints ที่ผ่านมา โดยจะดูว่าถ้ามีวิชาใดได้ต่ำกว่า C+ และจำเป็นต้องลงใหม่ในการลงใหม่นั้นจะแยกออกเป็น 2 กรณี คือ

1 ถ้าเป็นวิชาบังคับจะต้องลงซ้ำ

2 ถ้าเป็นวิชาเลือกจะต้องลงวิชาอื่นที่เทียบเท่าวิชาดังกล่าวแทนโดยจะไม่มีการลงซ้ำวิชาเดิม

สำหรับการที่จะรู้ว่าวิชาใดเป็นวิชาบังคับหรือวิชาเลือกนั้นจะได้จากการตรวจสอบจากตาราง หลักสูตรวิชา

- นักศึกษาจะทำการลงซ้ำ (Re grade) ไม่ได้ คือเมื่อมีการลงวิชาใดแล้วโดยผ่านเงื่อนไขที่กำหนดจากข้างต้นทั้งหมดแล้วจะไม่สามารถทำการลงซ้ำได้อีก
- เกรดต่ำกว่า 3.00 แต่เกิน 2.5 ติด Pro เป็นเงื่อนไขในการเตือนให้นักศึกษารู้ว่าจะต้องทำเกรดเพิ่ม เพราะถ้าลงทะเบียนครบตามหลักสูตรโดยที่เกรดยังไม่ถึง 3.00 จะไม่มีสิทธิจบการศึกษา

◆ เกรดต่ำกว่า 2.5 หมดสภาพความเป็นนักศึกษา เป็นเงื่อนไขการตรวจสอบสถานภาพความเป็นนักศึกษาเงื่อนไขหนึ่งว่า เกรดเฉลี่ยตลอดระยะเวลาของการเป็นนักศึกษา ณ เวลาใด ๆ จะต้องไม่ต่ำกว่า 2.5

◆ นักศึกษาจะถูกปรับสภาพให้อยู่ในสถานะจบการศึกษาก็ต่อเมื่อ

1 นักศึกษาต้องลงครบตามหน่วยกิตที่สาขานั้นกำหนดไว้ต่ำสุด เป็นการรวมหน่วยกิตสะสม ที่ได้ทั้งหมดของนักศึกษาที่ลงทะเบียนและผ่านเงื่อนไขต่างๆ ที่กำหนดไว้ข้างต้น โดยจะต้องมีค่ามากกว่าหรือ เท่ากับหน่วยกิตต่ำสุดที่สาขานั้นกำหนดไว้ซึ่งเก็บไว้ในตารางของนักศึกษา

2 นักศึกษาต้องได้เกรดเฉลี่ยมากกว่าหรือเท่ากับ 3.00 เป็นการหาเกรดเฉลี่ยสะสมทั้งหมด โดยการตรวจเช็คในข้อนี้จะต้องผ่านการตรวจสอบจากข้อ 1 ก่อน

3 นักศึกษาต้องลงวิชาบังคับตามที่หลักสูตรกำหนด เป็นการตรวจสอบว่าวิชาบังคับที่สาขานั้นกำหนดไว้ในตารางหลักสูตรมีการลงครบหรือไม่ โดยเงื่อนไขในการนับจะใช้เงื่อนไขจากข้อบังคับข้างต้นประกอบและต้องผ่านการตรวจสอบจากข้อ 1 และ 2 ก่อน

จากรายละเอียดของ Constraints ต่าง ๆ สามารถที่จะสรุปการแบ่งแยก Constraints ดังในตารางต่อไปนี้

ตารางที่ 16

สรุปการเปรียบเทียบกฎความคงสภาพของระบบกับกฎความคงสภาพที่นำเสนอ

Constraints	Level 0	Level 1	Level 2	Level 3	Level 4
1 Primary Key	Completeness	Static	Assertion	Structure-base	Single
2 Check Constraints	Validity	Static	Assertion	Structure-base	Single
3 Not null Constraints	Completeness	Static	Assertion	Structure-base	Single
4 Unique Constraints	Validity	Static	Assertion	Structure-base	Single

5 Foreign Key	Completeness	Static	Assertion	Structure-base	Multiple
6 นักศึกษามีระยะเวลาศึกษาได้ไม่เกิน 5 ปี	Validity	Dynamic	Assertion	Real-time-base	single
7 นักศึกษาจะลงวิชาต่อไปได้จะต้องผ่านวิชาที่เป็น prerequisite ก่อน	Validity	Dynamic	Assertion	Event-base	Multiple
8 หน่วยกิตเช็คจบจะถูกนับสะสมก็ต่อเมื่อเกรดที่ได้ต้องไม่ต่ำกว่า C+	Completeness	Static	Assertion	Content-base Event-base	Multiple
9 กรณีที่ได้เกรดต่ำกว่า C+ จะลงวิชาเดิมซ้ำได้เฉพาะวิชาที่เป็นวิชาบังคับเท่านั้น	Validity	Dynamic	Assertion	Content-base Event-base	Multiple
10 นักศึกษาจะทำการ Regrade ไม่ได้	Validity	Dynamic	Assertion	Event-base	Simple

11 เกรดต่ำกว่า 3.00 แต่เกิน 2.5 ติด Pro	Completeness	Static	Assertion	Aggregate	Simple
12 เกรดต่ำกว่า 2.5 หมดสภาพความเป็นนักศึกษา	Validity	Static	Assertion	Aggregate	Simple
13.1 นักศึกษาต้องลงครบตามหน่วยกิตที่สาขานั้นกำหนดไว้ต่ำสุด	Completeness	Static	Alerter	Aggregate	Multiple
13.2 นักศึกษาต้องได้เกรดเฉลี่ยมากกว่าหรือเท่ากับ 3.00	Completeness	Static	Alerter	Aggregate	Multiple
13.3 นักศึกษาต้องลงวิชาบังคับตามที่หลักสูตรกำหนด	Completeness	Static	Alerter	Aggregate	Multiple

5.3 การสร้างระบบและรายละเอียดการควบคุมความคงสภาพใน ระบบลงทะเบียน

ในขั้นตอนการทำงานของระบบลงทะเบียนนักศึกษา ซึ่งได้มีการนำเอา Integrity Constraints มาใช้ควบคุมความถูกต้อง โดยใช้ Oracle Development 2000 ทั้งในส่วนของ Form Developer , Stored Procedure and Trigger เป็นตัวช่วยในการจัดการนั้นมีดังนี้

การพัฒนาระบบ เพื่อแสดงตัวอย่าง การบังคับและควบคุมความคงสภาพของข้อมูลนั้นได้มีขั้นตอนการทำดังนี้

5.3.1 สร้างตารางที่เกี่ยวข้องกับระบบลงทะเบียนนักศึกษาทั้งหมดโดยได้มีการกำหนด Constraints พื้นฐานที่ Oracle มีประกอบด้วยดังนี้

- Primary Key Constraints กำหนด Key ในการเข้าถึงแถวในแต่ละตาราง
- Not Null Constraints กำหนดเพื่อบังคับคอลัมน์ที่ต้องการว่าจะต้องมีการให้ค่าเมื่อมีการ Insert ข้อมูล
- Unique Constraints กำหนดเพื่อบังคับคอลัมน์ที่ต้องการ ว่าแต่ละแถวในตารางจะต้องมีค่าไม่ซ้ำกันแต่จะยอมให้มีค่าเป็น Null ได้
- Check Constraints เป็นการตรวจสอบค่า และ ขอบเขตค่าของข้อมูลใน คอลัมน์ต่างๆ ตามที่ระบบต้องการ
- Foreign Key เป็นการอ้างถึงข้อมูลอื่นในตารางที่มีความสัมพันธ์กัน

5.3.2 ทำการสร้างหน้าจอการติดต่อและทำงานขึ้นโดยใช้ Form Developer 2000 เข้ามาช่วย ซึ่งมีส่วนการทำงานดังนี้

- หน้าจอหลัก ซึ่งเป็นหน้าจอแรกของการเข้าสู่การทำงานในระบบลงทะเบียน
- หน้าจอรายละเอียดนักศึกษา เป็นหน้าจอที่มีการทำงานเกี่ยวข้องกับข้อมูลของนักศึกษา
- หน้าจอรายละเอียดวิชา เป็นหน้าจอที่มีการทำงานเกี่ยวข้องกับข้อมูลของวิชาหลักสูตร (Subject_Rule) และ การกำหนดวิชาที่ต้องผ่านมาก่อน (Prerequisite)
- หน้าจอการลงทะเบียน เป็นส่วนที่เกี่ยวข้องกับการลงทะเบียนเรียนในวิชาต่างๆ ในแต่ละภาคการศึกษา
- หน้าจอผลการเรียน เป็นส่วนที่เกี่ยวข้องกับการแสดงผลและการลงผลการเรียนที่ได้ในแต่ละภาคการศึกษา
- หน้าจอการตรวจเช็คการจบการศึกษา เป็นหน้าจอที่ทำการเปลี่ยนสถานะสำหรับผู้ที่สำเร็จการศึกษา

5.3.3 การสร้าง เงื่อนไขและกฎเกณฑ์ต่าง ๆ ที่ระบบต้องการเพื่อมาบังคับควบคุมซึ่งมีขั้นตอนดังนี้

- วิเคราะห์ถึงกฎเกณฑ์ที่ระบบต้องการในแต่ละตัว ว่ามีผลต่อข้อมูลในส่วนใดบ้าง และสร้างเป็น Procedure และ Function ในการตรวจสอบตามความต้องการ
- นำเอากฎเกณฑ์ที่มีความเกี่ยวข้องกันมาสร้างรวมไว้เป็น Package เพื่อง่ายต่อการพัฒนาปรับปรุง
- กำหนดว่ากฎเกณฑ์เงื่อนไขต่าง ๆ ที่ระบบมี จะทำการบังคับควบคุมในส่วนข้อมูลที่กำหนดไว้ในลักษณะใดบ้าง เพื่อเป็นแนวทางในการสร้าง Trigger ที่จะมาควบคุมข้อมูลในระบบ โดย Trigger จะมีการเรียกใช้ Procedure และ Function ที่มีอยู่ใน Package ตามที่ได้มีการสร้างไว้แล้ว (ศึกษาแนวทางและขั้นตอนในการพัฒนา Triggerได้จากบทที่ 3)

บทที่ 6
สรุป

สำหรับโครงงานฉบับนี้เนื้อหาโดยรวมแล้วจะเกี่ยวข้องกับ การบังคับและแยกประเภทความคงสภาพของข้อมูลเพื่อให้ข้อมูลในระบบมีความถูกต้องครบถ้วนสมบูรณ์ ซึ่งได้มีการศึกษาและทำการรวบรวมความรู้ที่เกี่ยวข้องจาก บทความวิจัย ตำรา รวมทั้ง คำแนะนำจากผู้เชี่ยวชาญ จากนั้นจึงได้นำเอาความรู้ดังกล่าวมาเป็นแนวทางในการนำเสนอ การบังคับและแยกประเภทของความคงสภาพของข้อมูลขึ้น โดยมีแนวความคิดและได้นำเสนอการแบ่งแยกกฎความคงสภาพของข้อมูลออกเป็นระดับ ได้ทั้งหมด 5 ระดับตั้งแต่ระดับ 0 ซึ่งเป็นระดับสูงสุด ไปจนถึงต่ำสุดคือ ระดับ 4 รวมสำหรับการบังคับความคงสภาพของข้อมูลได้มีการนำเสนอเป็น ระดับเช่นกันโดยแบ่งได้เป็น 2 ระดับคือ ระดับ 0 กับระดับ 1

หลังจากนำเสนอ ในส่วนของการบังคับและแบ่งแยกความคงสภาพของข้อมูลแล้ว ได้มีการนำเอาส่วนของระบบลงทะเบียนนักศึกษามาแสดงเป็นตัวอย่างประกอบ ซึ่งได้ทำการออกแบบตรวจสอบและกำหนดความต้องการของระบบ จากนั้นก็นำเอากฎเกณฑ์ต่างๆ ที่ระบบต้องการ มาเปรียบเทียบและจัดเข้ากลุ่มตามกฎความคงสภาพที่ได้นำเสนอไว้ เพื่อแสดงให้เห็นว่ากฎความคงสภาพที่ได้นำเสนอการแบ่งแยกไว้นั้น สามารถรองรับและครอบคลุมกฎเกณฑ์ต่าง ๆ ของระบบที่มีการจัดการทางด้านฐานข้อมูล รวมทั้งแสดงการบังคับกฎความคงสภาพของระบบโดยใช้ Stored Procedure และ Database Trigger ใน Oracle Developer 2000 มาเป็นเครื่องมือช่วยในการสร้างและทดสอบ

บทที่ 7

ข้อเสนอแนะและแนวทางการพัฒนาในอนาคต

จากการจัดทำโครงงานฉบับนี้ ซึ่งเป็นส่วนที่เกี่ยวข้องกับการแบ่งแยก และบังคับความคงสภาพของข้อมูล จะเห็นได้ว่า ส่วนใหญ่แล้วการบังคับจะเกิดขึ้นเมื่อทำการ Insert, Delete, Update ข้อมูล เพราะเป็นส่วนที่มีผลทำให้ระบบฐานข้อมูลมีการเปลี่ยนแปลง อันจะส่งผลที่อาจเกิดการละเมิดความถูกต้องของระบบได้ แต่ถ้ามองในอีกด้านหนึ่งคือในส่วนของ การ Query เพื่อหาคำตอบต่าง ๆ จะเห็นว่า มักจะมีการมองข้ามสำหรับในส่วนของการตรวจสอบความถูกต้องของผลลัพธ์ที่ได้รับจากการ Query ดังกล่าว ซึ่งในบางครั้งผลลัพธ์ที่ได้อาจมีความไม่สมบูรณ์ถูกต้องและอาจส่งผลทำให้แนวทางในการทำงานกับฐานข้อมูลเกิดความผิดพลาดในเวลาต่อมาได้ ดังนั้นถ้าเรามีการตรวจสอบความถูกต้องเช่นเดียวกับ การตรวจสอบเมื่อเวลามีการเปลี่ยนแปลงข้อมูล การทำงานของระบบก็จะมีการทำงานที่มีประสิทธิภาพเพิ่มขึ้น

อีกกรณีหนึ่งที่เกี่ยวข้องและควรที่จะมีการพัฒนาต่อไปคือ เมื่อระบบมีกฎเกณฑ์และเงื่อนไขเพิ่มขึ้น ซึ่งมีความจำเป็นที่จะต้องนำมาควบคุมกับระบบ เราควรจะมีแนวทางที่เหมาะสมเพียงพอในการตรวจสอบและจัดการ ก่อนที่จะมีการปรับปรุงเพิ่มเติม เพราะในบางครั้งส่วนดังกล่าว อาจเกิดความขัดแย้งกับสิ่งที่มีอยู่ในระบบก่อนหน้านี้ได้

บรรณานุกรม

[1] Amihai, Motro."Integrity = Validity + Completeness." ACM Transaction on Database System ,vol.4,no.4,(December 1989):480-502.

[2] Besser, Brian, Benford John and others. Develop Application with the Procedural Option. n.p.,1993.

[3] Ceri, Stefano, Fraternali Piero and Paraboschi Stefano."Constraints Managemant in Chimera." Milano,Italy. E-mail:{ceri,fraterna,parabosc}@elet.polimit.it.

[4] Chomicki, Jan. "Efficient Checking of Temporal Integrity Constraints Using History Encoding. " Manhattan,Kansas.(May,1995). E-mail:Chomicki@cis.ksu.edu.

[5] Collins, Marie, Ford William and Thuraisingham Bhavani. "Security Constraints Processing During the Update Operation in A Multilevel Secure Database Management System" IEEE, (1991)pp.23-32.

[6] Conrad, Stefan and Turker Can."Active Integrity Maintenance in Federated Database System." Magdeburg,Geramny.E-mail:{conrad|tuerker}@iti.cs.uni-magdeburg.de,1995.

[7] Date, C.J. An Introduction to Database System.6th ed. Reading,Massachustts: Addison-Wesley, 1995.

[8] Feuerstein, Steven. ORACLE PL/SQL Programming. Sebastopol,CA: Oreilly&Associaes,1995.

[9] Korth, Henry F. and Silberschatz Abraham. Database System Concepts. 2nd ed. New York:Prentice Hall, 1991.

[10] Owens, Kevin T.. Building Intelligent Databases with ORACLE PL/SQL, TRIGGER,& STORED PROCEDURE. New Jersey:Prentice Hall, 1996.

[11] Weaver, Philip L. Practical SSADM Version 4. London:Pitman,1993.

[12] Stanley Y.W.Su, Y.C.Hong Nd. "Associative Hardware and Software Technolog for Integrity Control" ACM Transaction on Database System,vol.3,no.3, (Setember 1981): 416-440.

[13] มยุรี ว่องไววิศาล."การออกแบบฐานข้อมูลรวมแบบรีเลชันแนล." รายงานสัมนาทางวิชาการ คณะวิศวกรรมศาสตร์สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง.

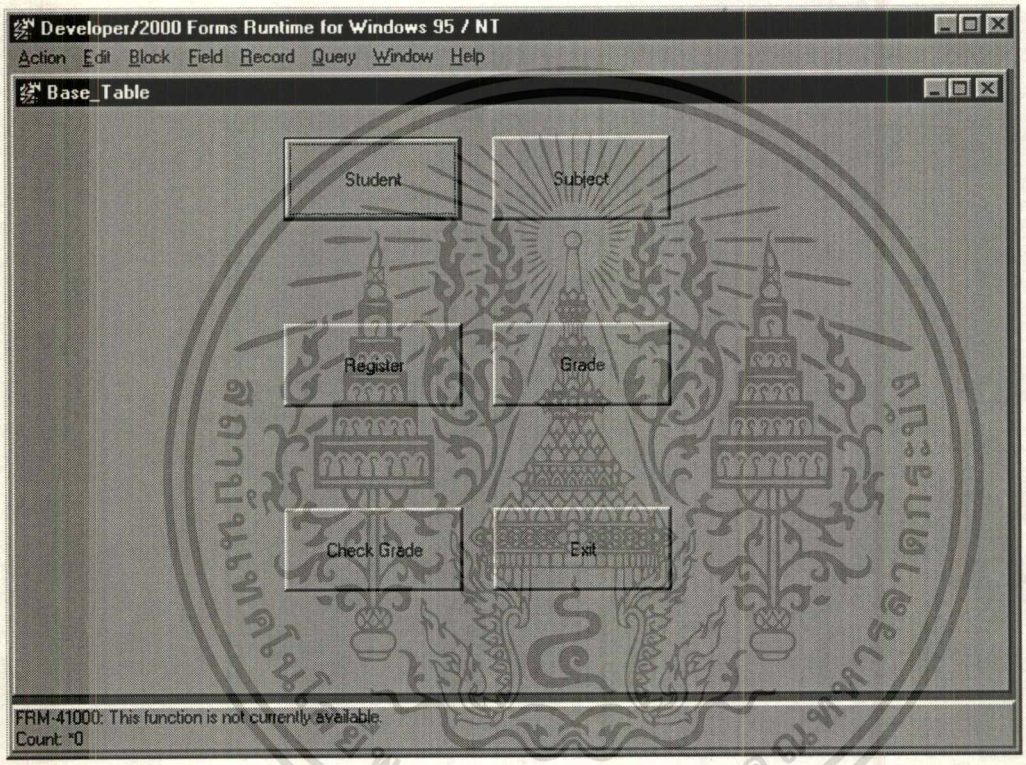
ภาคผนวก

แสดงหน้าจอการใช้งานของระบบลงเบียนนักศึกษา

1 หน้าจอหลัก

ภาพที่ 17

แสดงหน้าจอหลัก ของระบบลงทะเบียนนักศึกษา



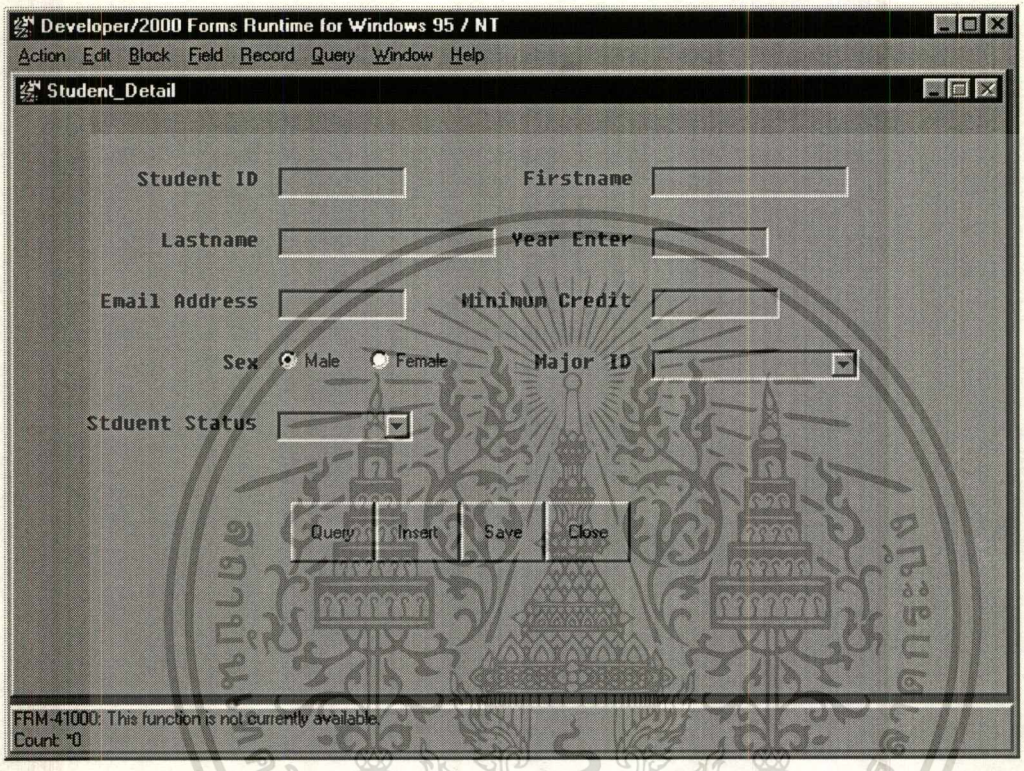
ประกอบด้วยปุ่มในการเลือกรายละเอียดการทำงานดังนี้

- ปุ่มเพื่อแสดงรายละเอียดการทำงานในส่วนของข้อมูล นักศึกษา
- ปุ่มเพื่อแสดงรายละเอียดการทำงานในส่วนของข้อมูล วิชา
- ปุ่มเพื่อแสดงรายละเอียดการทำงานในส่วนของข้อมูลการลงทะเบียน
- ปุ่มเพื่อแสดงรายละเอียดการทำงานในส่วนของข้อมูลการลงเกรด
- ปุ่มเพื่อแสดงรายละเอียดการทำงานในส่วนของข้อมูลการตรวจสอบการจบการศึกษา
- ปุ่มเพื่อออกจากระบบ

2 หน้าจอรายละเอียดนักศึกษา

ภาพที่ 18

แสดงหน้าจอรายละเอียดนักศึกษาในระบบลงทะเบียนนักศึกษา

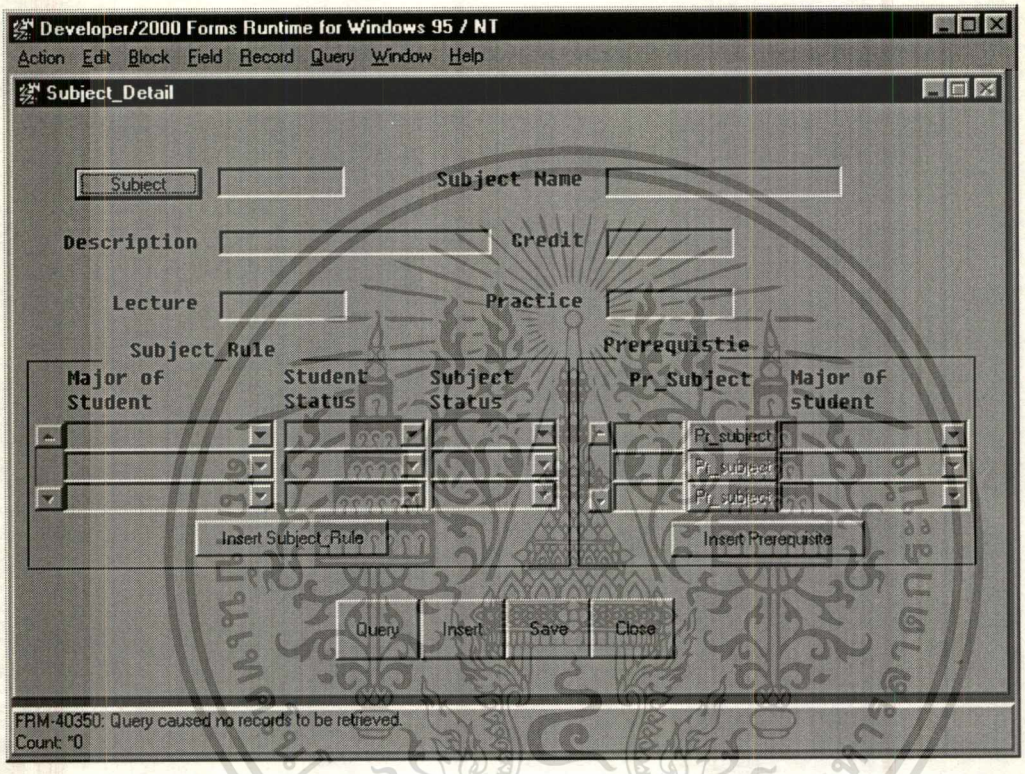


หน้าจอนี้จะจัดการข้อมูลในตาราง Student โดย Operation ต่าง ๆ ที่มีกำหนดไว้

3 หน้าจอรายละเอียดวิชา

ภาพที่ 19

แสดงหน้าจอรายละเอียดรายวิชา หลักสูตร และ Prerequisite ในระบบลงทะเบียนนักศึกษา

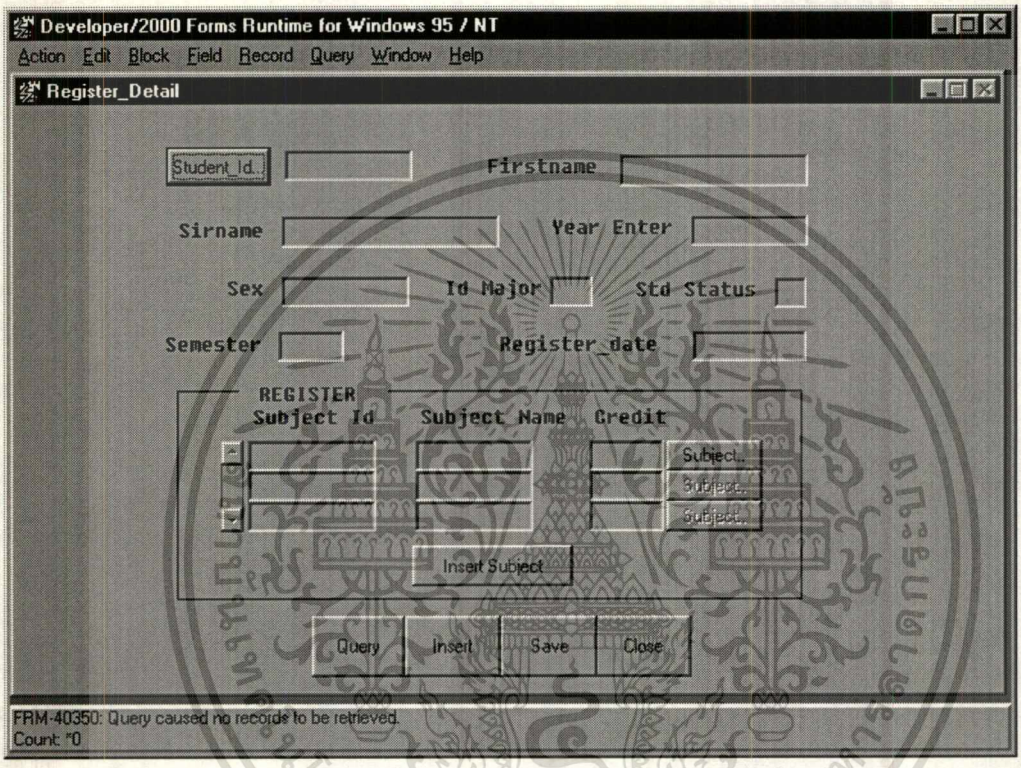


หน้าจอนี้จะมีการออกแบบในลักษณะ Master - Details ซึ่งจะมีการจัดการโดย Operation ต่าง ๆ กับตาราง Subject (เป็น Master) ตาราง Subject_rule (เป็น Detail) และ ตาราง Prerequisite (เป็น Detail)

4 หน้าจอรายละเอียดการลงทะเบียน

ภาพที่ 20

แสดงหน้าจอรายละเอียดการลงทะเบียนเรียนในระบบลงทะเบียนนักศึกษา

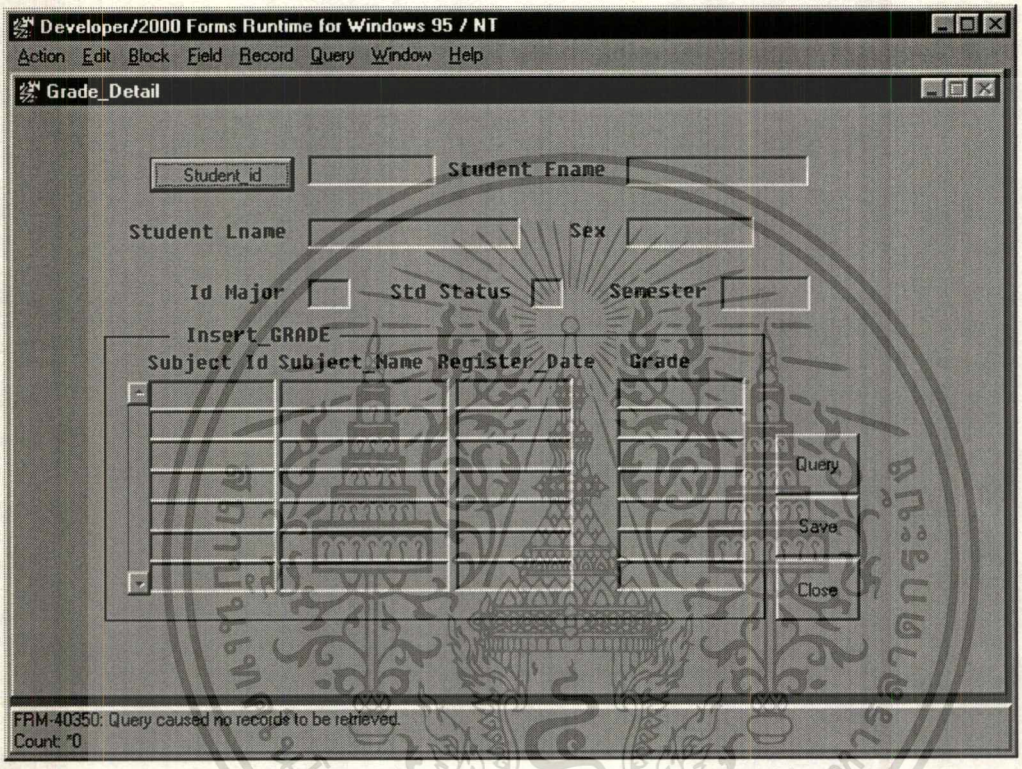


หน้าจอนี้จะมีการออกแบบในลักษณะ Master - Details ซึ่งจะมีการจัดการโดย Operation ต่าง ๆ กับตาราง Student (เป็น Master) และตาราง Student (เป็น Detail)

5 หน้าจอรายละเอียดการลงเกรด

ภาพที่ 21

แสดงหน้าจอรายละเอียดการกรอกเกรดในระบบลงทะเบียนนักศึกษา

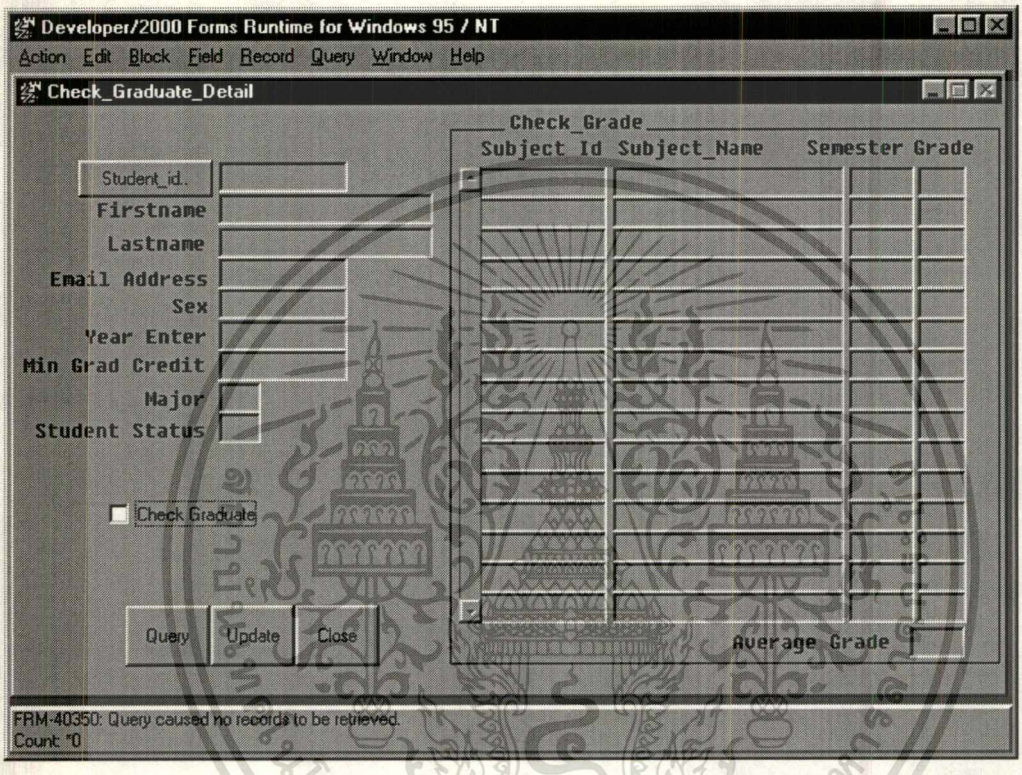


เป็นหน้าจอที่มีการ Update ข้อมูลจากการลงทะเบียนในส่วนของการลงรายละเอียดในส่วนของ เกรดที่ได้ในแต่ละวิชาของภาคการศึกษานั้น ๆ

6 หน้าจอรายละเอียดการตรวจสอบการจบการศึกษา

ภาพที่ 22

แสดงหน้าจอการตรวจสอบการจบการศึกษาของนักศึกษาในระบบลงทะเบียนนักศึกษา



เป็นหน้าจอที่ตรวจเช็คการเรียนของนักศึกษาในตาราง Student(เป็น Master) กับข้อมูลการลงทะเบียนทั้งหมด (เป็น Detail) เพื่อทำการเปลี่ยนสถานภาพเป็นจบการศึกษาถ้ามีการลงทะเบียนครบตามหลักสูตร

ประวัติผู้เขียน

ชื่อผู้เขียน	นายสาธิษฐ์ นากกระแสร์
วันเดือนปีเกิด	16 มีนาคม 2517
สถานที่เกิด	จังหวัดสุรินทร์
วุฒิการศึกษาระดับปริญญาตรี	วท.บ.(วิทยาการคอมพิวเตอร์)
สถานที่สำเร็จการศึกษา	คณะวิทยาศาสตร์ มหาวิทยาลัยรามคำแหง
ปีที่สำเร็จการศึกษา	ปีการศึกษา 2537
การทำงาน	อาจารย์มหาวิทยาลัยรามคำแหง