ปริญญานิพนธ์ปีการศึกษา 2536
ภาควิชา วิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
เรื่อง Database System Environment Studies

ผู้จัดทำ
1.นางสาวชนิกานต์ วัฒนกิจถาวรกุล รหัส 33100069
2.นางสาวธิดาพร ผ่านิ่มมงคล รหัส 33100144
3.นางสาวธีรณี อุจลากุล รหัส 33100145

อาจารย์ที่ปรึกษา

(ผศ.ดร.คุภมิตร จิตตะยโศธร)

การศึกษาสภาวะแวดล้อมของระบบฐานข้อมูล

Database System Environment Studies

โดย นางสาวชนิกานต์ วัฒนกิจถาวรกุล
นางสาวธิดาพร เผ่านิ่มมงคล
นางสาวธีรณี อจลากุล

อาจารย์ที่ปรึกษา ผศ.ดร.ศุภมิตร จิตตะยโศธร

ปีการศึกษา 2536

บทคัดย่อ

ปริญญานิพนธ์นี้นำเสนอหลักการในการพิจารณาเลือกระบบจัดการฐานข้อมูลที่ดีโดยพิจารณาตามกฎ 12 ข้อของคอดด์ และกฎ 16 ข้อจาก RDBMS V.2. โดยพิจารณาเปรียบเทียบระบบจัดการฐานข้อมูล 3 ตัวคือ ออราเคิล อินเกรส และ กุปต้า

นอกจากนี้ปริญญานิพนธ์ยังได้กล่าวถึงความเหมาะสมในการใช้งานระหว่าง ระบบฐานข้อมูลแบบโฮสท์เดี่ยว และแบบไคล์แอ็นต์/เซิร์ฟเวอร์ โดยรวบรวมความรู้จากแหล่งต่างๆมาสรุปลงในระบบผู้เชี่ยวชาญ ระบบดังกล่าวช่วยการตัดสินใจในการเลือกระบบที่เหมาะสมกับงาน

ได้มีการทำการทดสอบระบบจัดการฐานข้อมูล คือ ออราคเคิล และ อินเกรสบนโฮสเดี่ยว และ กุปต้าบนไคล์แอ็นต์/เซิร์ฟเวอร์ เพื่อให้เห็นแนวทางเกี่ยวกับประสิทธิภาพในการประมวลผลการเรียกค้น (query) ต่างๆของระบบจัดการฐานข้อมูลและแพลตฟอร์ม

Abstract

This thesis proposes the way to consider RDBMS based on Codd 12 rules and Codd 16 rules of RDBMS V.2. Three commercial products namely Oracle, Ingres and Gupta were brought to compare.

The thesis also mentions about which RDBMSs and platforms (Single-Host and Client/Server , for example) suit the requirements of the users. All the information collected from many sources was put into a rule-based expert system. This expert system can suggest the right RDBMS for the right job.

Test on RDBMS, Oracle, Ingres on Single-system and Gupta on multiple Client/Server were also done to demonstrate the effiency of RDBMS's query processing.

สารบัญ

บทที่ 1

บทนำ

ระบบจัดการฐานข้อมูล (DBMS:- Database Management System เป็นซอฟแวร์ระบบ (System Software) ที่มีความสำคัญและมีประโยชน์มากซึ่งระบบจัดการฐานข้อมูลในปัจจุบันมีหลายโมเดลไม่ว่าจะเป็น ไฮราคีโมเดล เน็ตเวอร์กโมเดล และระบบจัดการฐานข้อมูลแบบรีเลชั่นนัล โดยเก็บข้อมูลในลักษณะตารางและกำหนดรายละเอียดว่ามีตารางอะไรบ้างและแต่ละตารางมี แอตทริบิวต์ (attribute) หรือ ฟิลด์ (field)ใดบ้างและมีความสัมพันธ์กับตารางใดแบบใดซึ่งข้อมูลเหล่านี้จะเก็บอยู่ใน พจนานุกรมข้อมูล (data dictionary)

E.F.codd ปรมาจารย์ทางด้านระบบฐานข้อมูลแบบรีเลชั่นนัล ได้กล่าวถึงกฎ 12 ข้อในการออกแบบระบบจัดการฐานข้อมูลแบบรีเลชั่นนัล ซึ่งทำให้แนวความคิดโดยทั่วไปทางทฤษฎีของฐานข้อมูลแบบรีเลชั่นนัลค่อนข้างกระจ่างแล้วแต่ยังมีปัญหาโดยส่วนใหญ่ซึ่งเกิดขึ้นกับผู้ขายของระบบจัดการฐานข้อมูลในท้องตลาดส่วนใหญ่ยังมีข้อผิดพลาดอีกมากมายที่เกิดขึ้นในส่วนการออกแบบและสร้างให้เป็นไปตามที่ต้องการ (implement) ดังนั้น E.F.codd ปรมาจารย์ทางด้านระบบฐานข้อมูลเชิงสัมพันธ์ได้กล่าวถึงกฎ 16 ข้อในการออกแบบระบบจัดการฐานข้อมูลแบบรีเลชั่นนัล ซึ่งใช้เป็นแนวทางในการพิจารณาการเลือกระบบฐานข้อมูลแบบรีเลชั่นนัลที่ดีอีกทั้งยังมีทฤษฎีอื่นๆที่เป็นหลักการพื้นฐานของระบบจัดการฐานข้อมูลเช่น concurrency control, integrity, Recovery และ Query optimization เป็นต้นดังนั้นจึงได้ใช้ทั้งกฎ 12 และ 16 ข้อในการออกแบบระบบจัดการฐานข้อมูลแบบรีเลชั่นนัล รวมทั้งทฤษฎีอื่นๆที่เกี่ยวข้องมาเป็นหลักในการพิจารณาเปรียบเทียบระบบจัดการฐานข้อมูลแบบรีเลชั่นนัล และยังได้กล่าวถึงความเหมาะสมในการเลือกใช้งานระหว่างระบบจัดการฐานข้อมูลแบบรีเลชั่นนัล ที่เป็นแบบ Single-host และระบบจัดการฐานข้อมูลแบบรีเลชั่นนัลที่เป็นแบบ Client/Server ด้วยและได้มีการนำข้อมูลที่ได้จากแหล่งต่างๆมาสรุปและนำมาเป็นฐานความรู้ของระบบผู้เชี่ยวชาญชนิดrule-based "PC-PLUS" โดยหลักการในการประมวลผลของระบบผู้เชี่นวชาญนี้ก็คือ backward *engine* ซึ่งระบบนี้จะช่วยตัดสินใจในการเลือกระบบที่เหมาะสมกับงาน

นอกจากนี้ยังได้มีการนำกฎและหลักมาพิจารณาประกอบกับการใช้ชุดของการ query และข้อมูลจากห้องสมุดคณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง โดยแต่ละ query ในชุดของการ query นั้นจะมีเกณฑ์ในการพิจารณาต่างๆกันเช่น query ที่เปรียบเทียบคุณสมบัติ Grouping Feature โดยเงื่อนไขที่ตั้งขึ้นมาคือการมี index และไม่มี index จากนั้นก็ทำการ run query และเปรียบเทียบเวลาที่ใช้ในการประมวลผล ซึ่ง query นี้เป็นการเปรียบเทียบคุณสมบัติของตัวระบบจัดการฐานข้อ

มูลเอง จากนั้นก็ใช้เกณฑ์ดังกล่าวข้างต้นในการเลือกพิจารณาเปรียบเทียบระบบจัดการฐานข้อมูลแบบรีเลชั่นนัล อันได้แก่ INGRES ORACLE และ GUPTA ซึ่งระบบจัดการฐานข้อมูลแต่ละตัวนั้นมีองค์ประกอบสภาวะแวดล้อมที่แตกต่างกัน โดย

- INGRES เป็น INGRES release 6.4 บนเครื่อง HP 827 ซึ่งมีหน่วยความจำ 48 MB และมีเนื้อที่ใน disk ถึง 1.36 GB โดยมีลักษณะเป็น Single-host และมีจำนวนผู้ใช้ที่มากที่สุดถึง 32 คน

- ORACLE เป็น ORACLE version 5 บนเครื่อง LADWFF ซึ่งมีหน่วยความจำ 8 MB และ มี CPU ของโมโตโรล่า โดยมีลักษณะเป็น Single-host

- GUPTA เป็น GUPTA ซึ่งมีหน่วยความจำ 16 M และมี CPU เป็น 486 DX2-66 และมีเนื้อที่ใน disk 240 MB

การทดสอบดังกล่าวนี้ทำให้สามารถเห็นแนวทางการพิจารณาประสิทธิภาพของระบบจัดการฐานข้อมูลแบบรีเลชั่นนัลบนสภาวะแวดล้อมต่างๆ

บทที่ 2

หลักการและทฤษฎีที่เกี่ยวข้อง

2.1 ระบบจัดการฐานข้อมูล (Database Management System)

ระบบจัดการฐานข้อมูล หรือที่นิยมเรียกกันว่า "DBMS" ซึ่งย่อมาจาก Database Management System หมายถึง ซอฟแวร์ระบบ (System Software) ที่มีความสำคัญและมีประโยชน์มากโดยเฉพาะกับองค์กรที่มีระบบงานขนาดใหญ่ จัดการข้อมูลจำนวนมากและมี ผู้ใช้ (user) หลาย ๆ คนใช้พร้อม ๆ กัน และผู้ใช้แต่ละคนนั้นเกี่ยวข้องและสามารถแก้ไขข้อมูลในฐานข้อมูลได้ ซึ่งประโยชน์หลักๆ.ของระบบจัดการฐานข้อมูลก็ต้องกล่าวถึง

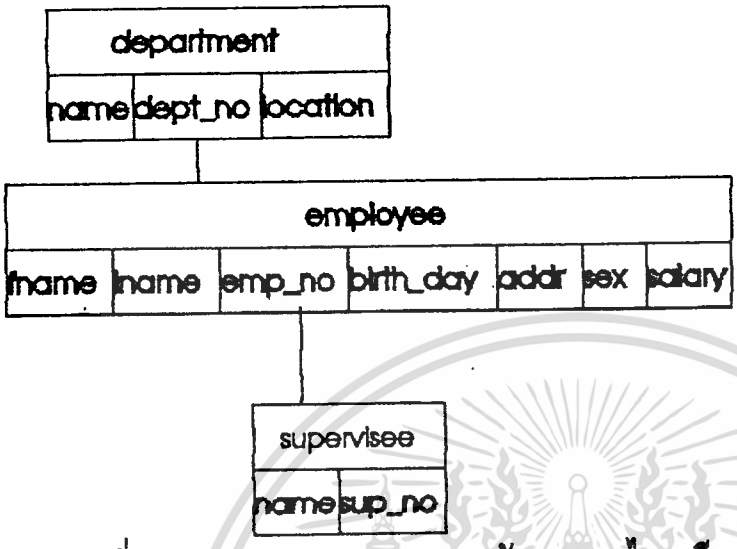
- การประมวลผลกลุ่มของงาน (กลุ่มของงานหรือที่นิยมเรียกกันว่า Transaction)ในแง่ความถูกต้องของข้อมูล (Integrity) การจัดการควบคุมสถานะของข้อมูลเมื่อมีผู้ใช้หลายๆ คนพร้อมกัน (Concurrency Control) และการกู้ข้อมูลคืนให้อยู่ในสถานะที่ถูกต้องเมื่อเกิดความเสียหายกับระบบไม่ว่าในกรณีใด ๆ (Recovery)
- มีส่วนที่ทำหน้าติดต่อกับ ผู้ใช้ หรือที่เรียกว่า Tool ซึ่งมีประสิทธิภาพสูงและมีความเป็นกันเองกับผู้ใช้ เช่น ภาษา SQL หรือภาษาสอบถามเชิงโครงสร้างที่มีอยู่ในระบบจัดการฐานข้อมูลเชิงสัมพันธ์จะทำหน้าที่สร้างองค์ประกอบ (Objects) เช่น ตาราง (Table) ,วิว (view) หรือ index เป็นต้น และผู้ใช้จะเรียกค้นหาข้อมูล (Query) โดยจะกำหนดเงื่อนไขในการค้นหาข้อมูลเช่น ให้แสดงรายชื่อของพนักงานที่มีเงินเดือนมากกว่า 30000 บาทและเป็นเพศหญิงเป็นต้น
- จัดการด้านความปลอดภัยของข้อมูลในระบบฐานข้อมูลเช่น ระบบ password หรือ view เป็นต้น

ฐานข้อมูลเชิงสัมพันธ์ (Relational Database)

ฐานข้อมูลในปัจจุบันมีการจัดสร้างโมเดลของฐานข้อมูลหลายรูปแบบเช่น โมเดลแบบไฮราคี (Hierarchy Model) ; โมเดลแบบเน็ตเวิร์ก (Network Model) และโมเดลเชิงสัมพันธ์ (Relational Model)

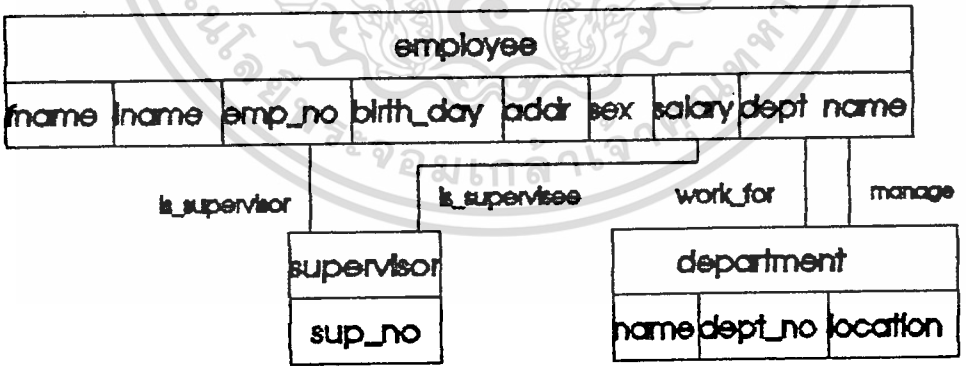
1) โมเดลของฐานข้อมูลแบบไฮราคี เช่น ผลิตภัณฑ์ ของ IBM ที่ชื่อว่า Information Management System (IMS) ซึ่งโครงสร้างของข้อมูล (Data Structure) เป็นแบบต้นไม้ (Tree) โดยมี root (ราก) อยู่บนสุดและไล่ไปตามกิ่งและแขนงของต้นไม้ข้อมูล และความสัมพันธ์ของระเบียนข้อมูล (record) ที่อยู่ในระดับที่สูง

กว่าและต่ำกว่าจะเรียกว่าความสัมพันธ์ parent และ child ซึ่งมีรูปแบบของฐานข้อมูลดังรูป



รูปที่ 2.1 รูปแสดงรูปแบบของฐานข้อมูลแบบไฮราคี

2) ส่วนโมเดลของฐานข้อมูลแบบเน็ตเวิร์กจะเก็บความสัมพันธ์ระหว่างระเบียนข้อมูลเป็นแบบ parent และ child แต่ระเบียนข้อมูลสามารถมีความสัมพันธ์แบบ one-to-many ได้ ซึ่งมีรูปแบบของฐานข้อมูลดังรูป



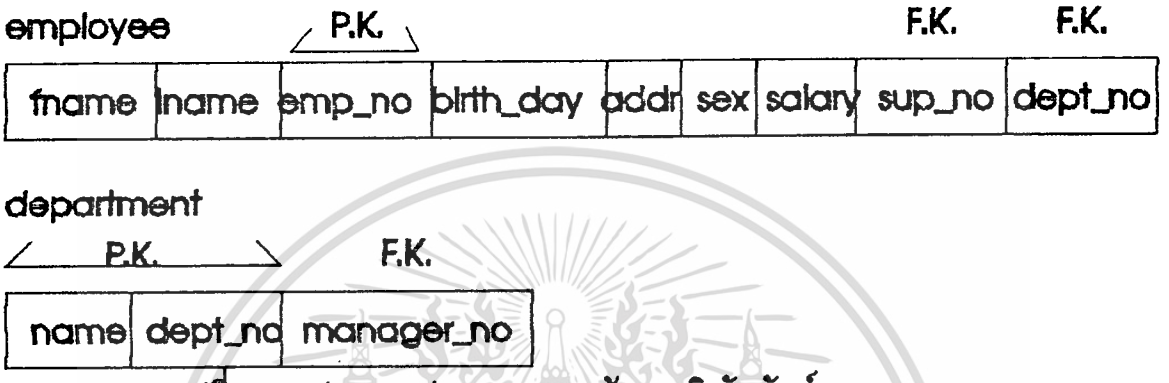
รูปที่ 2.2 รูปแสดงรูปแบบของฐานข้อมูลแบบเน็ตเวอร์ก

แต่ Model ทั้ง 2 แบบยังมีข้อบกพร่องอยู่บางประการเช่น

- โครงสร้างในการจัดเก็บข้อมูลมีความยุ่งยากซับซ้อน
- ในการเข้าถึงข้อมูลที่ data element (element ย่อยในระเบียนข้อมูล)จะต้องดึงข้อมูลทั้งระเบียน
- ในการค้นหาข้อมูลและเข้าถึงข้อมูลมีความยุ่งยาก

3) ส่วนโมเดลของฐานข้อมูลเชิงสัมพันธ์

ดังนั้นประมาณปี คศ.1970-1972 E.F. Codd ได้เสนอแนวความคิดฐานข้อมูลเชิงสัมพันธ์โดยเก็บข้อมูลในลักษณะตารางและกำหนดรายละเอียดว่ามีตารางอะไรบ้างและแต่ละตารางมีแอตทริบิวต์ (attribute) หรือ ฟิลด์ (field)ใดบ้างและมีความสัมพันธ์กับตารางใดแบบใดซึ่งข้อมูลเหล่านี้จะเก็บอยู่ใน พจนานุกรมข้อมูล (data dictionary) ซึ่งมีรูปแบบของฐานข้อมูลดังรูป

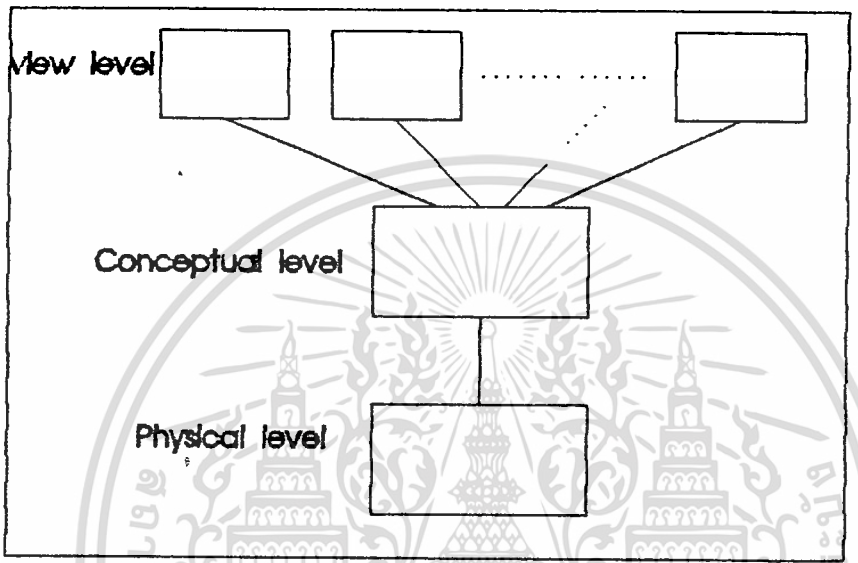


รูปที่ 2.3 รูปแสดงรูปแบบของฐานข้อมูลเชิงสัมพันธ์

ซึ่งจากรูปของฐานข้อมูลจะเห็นได้ว่า สามารถเข้าใจได้ง่ายและสามารถเข้าถึงข้อมูลในระดับ data element ได้โดยไม่ต้องดึงข้อมูลทั้งระเบียนข้อมูล

2.2 สถาปัตยกรรมที่ใช้ในการจัดการฐานข้อมูล

ข้อมูลจำนวนมากจะถูกเก็บอยู่ในฐานข้อมูลซึ่งผู้ใช้จะสามารถใช้งานผ่านระบบจัดการฐานข้อมูล ซึ่งรูปแบบของการแบ่งระดับของฐานข้อมูลและการใช้งานจะแบ่งเป็นหลายระดับดังรูป



รูป 2.4 รูปแสดงมุมมองของการจัดการฐานข้อมูลโดยทั่วไป

ซึ่งประกอบไปด้วย

1) Physical level

เป็น level ล่างสุดซึ่งใน level นี้จะทำหน้าที่อธิบายว่า " มีการจัดการ ข้อมูล อย่างไร" (HOW) ใน level นี้จะมี data structure ที่ค่อนข้างจะยุ่งยากซับซ้อน

2) Conceptual level

เป็น level ที่อยู่ในระดับที่สูงกว่า Physical level ซึ่งทำหน้าที่อธิบายว่า " มีการจัดการข้อมูลใด " (WHAT) ซึ่ง ผู้ใช้ ใน level นี้ไม่จำเป็นต้องเกี่ยวข้องกับโครงสร้างอันยุ่งยากใน Physical level เลยถึงแม้ว่าการ implementation ใน level นี้ต้องเกี่ยวข้องกับ Physical level ก็ตาม ใน level นี้ DBA เป็นผู้ใช้

3) View level

ซึ่งในด้านมุมมองของผู้ใช้นั้นข้อมูลที่ผู้ใช้เรียกค้นขึ้นมาใช้จะปรากฏตาม view ที่กำหนด โดยผู้ใช้ไม่ต้องทราบเลยว่าข้อมูลเหล่านั้นเก็บไว้อย่างไรและถูกเรียกใช้ได้อย่างไรจากฐานข้อมูล ผู้ใช้รู้เพียงแต่ว่าข้อมูลที่เรียกมาใช้เป็นไปตามข้อกำหนด หรือความต้องการของตนเองก็เพียงพอแล้ว

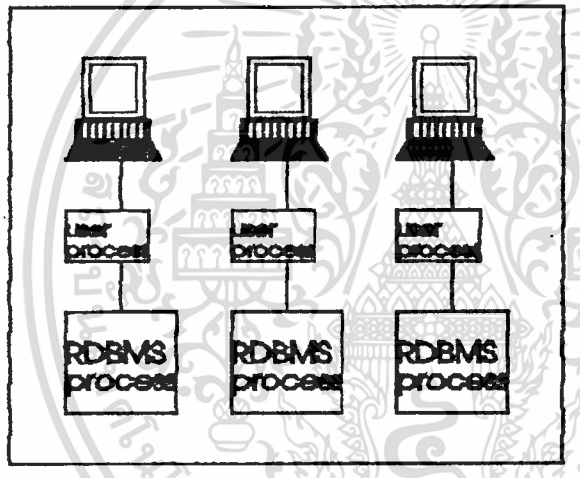
2.3 Process Architecture

จะอธิบายการ implementation ของ multiserver RDBMS ใน Operating system ซึ่งแบ่งออกเป็น 3 model คือ

1) Process Model

เป็นสถาปัตยกรรมที่ support "Single threaded" ("threaded" ในความหมายของ DBMS คือ execution path)

- มีหลักการทำงาน คือ 1 ผู้ใช้ : 1 process server
- สามารถทำ multiprogramming ได้



รูปที่ 2.5 รูปแสดงสถาปัตยกรรมแบบ process model

ข้อดี 1) implement ง่าย

2) RDBMS process มีขนาดเล็ก ดังนั้นสามารถใช้ feature ของ operating system ได้มาก

3) RDBMS process ไม่ถูก swap ออกหมด ถ้า swap RDBMS process ตัวใดตัวหนึ่งออก ผู้ใช้ อื่น ๆ ก็สามารถทำงานต่อได้ด้วย RDBMS process ของ ตนเอง

ข้อเสีย 1) จำนวนของ active users น้อยเพราะถูกจำกัดด้วยจำนวนของ process ซึ่งมีเป็นจำนวนมาก

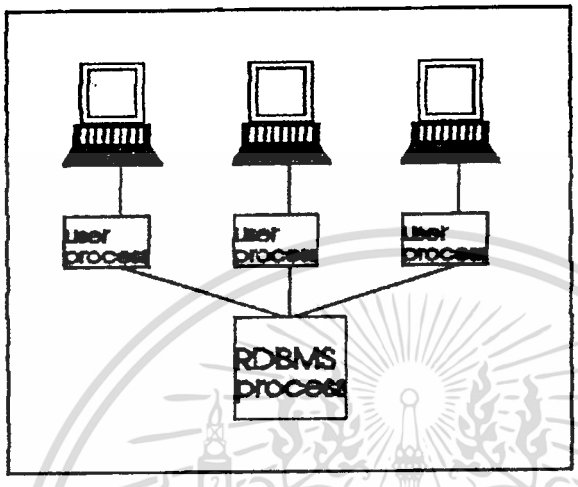
2) เกิด overhead มากเนื่องจาก context switching ของ process มีเป็นจำนวนมาก

2) Single Server Model

- ผู้ใช้ แต่ละคนจะส่ง request message เพื่อให้ single server process

ทำหน้าที่ serve

- ยังถือว่าเป็น " single threaded "เพราะยังถือว่า มี execution path เพียง path เดียว



รูปที่ 2.6 รูปแสดงสถาปัตยกรรมแบบ single server model

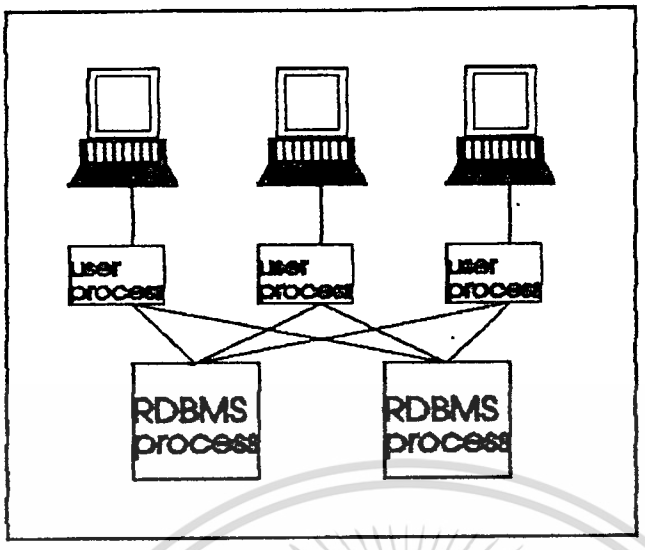
ข้อดี 1) เนื่องจากมีเพียง RDBMS process เดียวทำให้จำนวนผู้ใช้ที่ active มีจำนวนเพิ่มขึ้น

2) จำนวน process ที่ลดลงทำให้ context switching ลดลงดังนั้นทำ" ให้ overhead ลดลงด้วย

ข้อเสีย 1) ทำ parallel processing ไม่ได้

3) Multiple Server Model

- มี RDBMS process หลายตัวคอย serve users ได้หลายคน
- ถือว่าเป็น " multithreaded"



รูปที่ 2.7 รูปแสดงสถาปัตยกรรมแบบ multiple server model

ข้อดี 1) จำนวน process ที่ลดลงทำให้ context switching ลดลงดังนั้นทำให้ overhead ลดลงด้วย

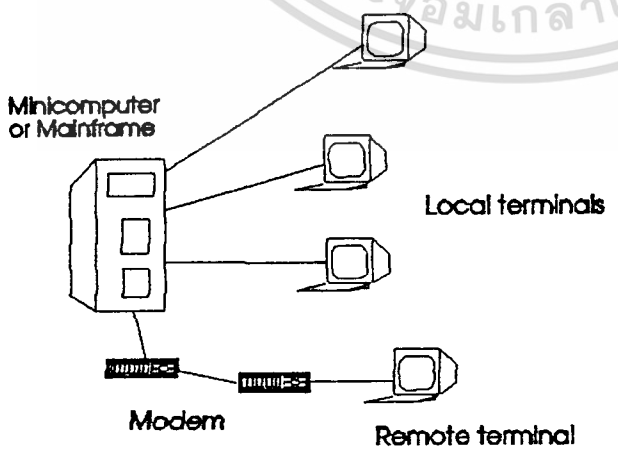
2) สามารถทำ parallel processing ได้

2.4 Computer System Architectures

2.4.1 Centralized Platforms

สำหรับระบบลักษณะนี้ทุกๆโปรแกรมจะรันบนคอมพิวเตอร์หลัก (Host Computer)ทั้งสิ้นรวมทั้งตัวแอปพลิเคชั่นของระบบจัดการฐานข้อมูลที่จะเข้าถึงฐานข้อมูล และ facilities เกี่ยวกับการสื่อสารที่จะทำการส่งและรับข้อมูลจากเทอร์มินัลของผู้ใช้ด้วย ผู้ใช้จะสามารถเข้าถึงระบบได้ทั้งโดยการผ่านการเชื่อมต่อแบบโลคัล (local connect) และการเข้าถึงในระยะไกล (remote access ผ่านโมเด็ม) เทอร์มินัลของระบบแบบนี้จะมีลักษณะเป็นดัมพ์เทอร์มินัลคือ เทอร์มินัลจะไม่มีความสามารถในการประมวลผลด้วยตัวเองเลย ส่วนตัวคอมพิวเตอร์หลักมักจะใช้เมนเฟรมหรือมินิคอมพิวเตอร์เป็นหลัก โดยการประมวลผลข้อมูลทั้งหมดจะถูกกระทำบนคอมพิวเตอร์หลักนี้

โดยส่วนมากในระบบนี้เมื่อผู้ใช้เปิดเทอร์มินัลเพื่อที่จะเข้าถึงระบบหน้าจอแรกที่จะปรากฏมักจะเป็นหน้าจอ login เพื่อให้ผู้ใช้ใส่ชื่อ enter login และ password ซึ่งถ้าถูกต้องผู้ใช้ก็จะ gain access เมื่อมีการ startup แอปพลิเคชั่น หน้าจอที่เหมาะสมต่างๆก็จะถูกส่งผ่านสายมาที่ผู้ใช้เทอร์มินัล และแอปพลิเคชั่นก็จะกระทำการต่างๆตามที่ผู้ใช้กดแป้นพิมพ์หรือใช้เมาส์ ทั้งแอปพลิเคชั่นของผู้ใช้และระบบจัดการฐานข้อมูลต่างก็รันบนคอมพิวเตอร์เดียวกันและจะทำการติดต่อสื่อสารกันโดยผ่าน shared memory ที่ควบคุมและจัดการโดย operating system ของคอมพิวเตอร์หลัก

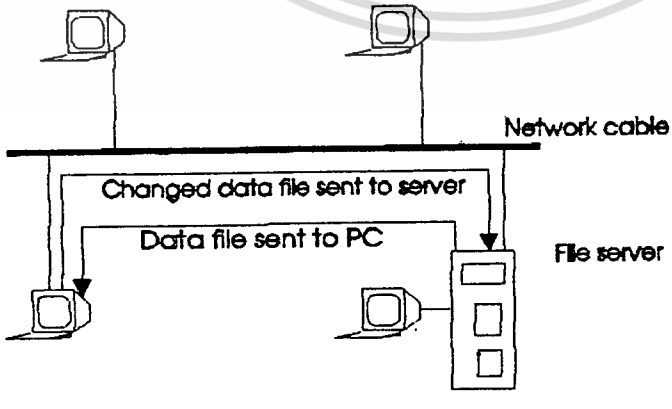


รูป 2.8 ภาพแสดงระบบแบบ Centralize

ระบบแบบ centralize นี้เป็นระบบที่สามารถจะรองรับ ผู้ใช้ จำนวนมากได้ แต่อาจมีค่าใช้จ่ายสูงทั้งด้านการซื้อและการบำรุงรักษา ตัวเมนเฟรม หรือมินิคอมพิวเตอร์ขนาดใหญ่นั้นบางระบบต้องการ facility พิเศษสนับสนุนมากมาย ตัวอย่างเช่น water-cooling systems, climate control systems และรวมทั้งต้องการทีมงานและผู้เขียนโปรแกรมระบบที่มีความเข้าใจระบบสูงอีกด้วยจึงทำให้มักจะต้องมีการจัดการฝึกอบรมพนักงานที่เกี่ยวข้องเสมอๆ

2.4.2 PC-based System

การมีระบบจัดการฐานข้อมูลรันอยู่บน PC PC จะแสดงตัวเป็นทั้งคอมพิวเตอร์หลักและเทอร์มินัลในขณะเดียวกันคือเป็นระบบ stand-alone หรืออาจจะเรียกว่าระบบ PC-based ก็ได้ตัวอย่างเช่น ระบบ LANs ที่มีฐานข้อมูลอยู่บน PC และมีการรัน network operating system พิเศษ เช่น Novell's NetWare หรือ microsoft LAN manager อยู่บน PC เรียกว่าเป็น file server ซึ่งตัว file server นี้จะเป็นตัวจัดการให้ ผู้ใช้สามารถที่จะ share common data files โดยที่การประมวลผลจริงๆยังคงทำอยู่ที่คอมพิวเตอร์ของผู้ใช้ที่รันแอปพลิเคชั่นบนฐานข้อมูล file server จะเพียงแต่ทำการค้นหา file ข้อมูลที่ผู้ใช้ต้องการแล้วส่งผ่านเครือข่ายไปให้คอมพิวเตอร์ของผู้ใช้ หลังจากนั้นข้อมูลจะถูกประมวลผลภายใต้การควบคุมของระบบจัดการฐานข้อมูลบนคอมพิวเตอร์ของผู้ใช้ และการเปลี่ยแปลงข้อมูลต่างๆบนคอมพิวเตอร์ของผู้ใช้จะถูกส่งกลับมาที่ file server ในลักษณะการส่งกลับทั้ง file มาเก็บที่ disk บน file server



รูปที่ 2.9 ภาพแสดงระบบแบบ PC-Based

จากการทำงานในลักษณะนี้ความสามารถของตัว file server จะไม่มีผลต่อความเร็วในการประมวลผลเลยเพราะ ความเร็วต่างๆจะถูกจำกัดด้วยความสามารถของคอมพิวเตอร์ของผู้ใช้ (ที่รันระบบจัดการฐานข้อมูล) ระบบนี้ถ้ามีผู้ใช้มาเข้าถึงพร้อมๆกันหลายๆคน file เดียวกันก็อาจจะถูกส่งไปหลายๆที่ได้ทำให้เพิ่ม traffic และทำให้เครือข่ายช้าลงได้

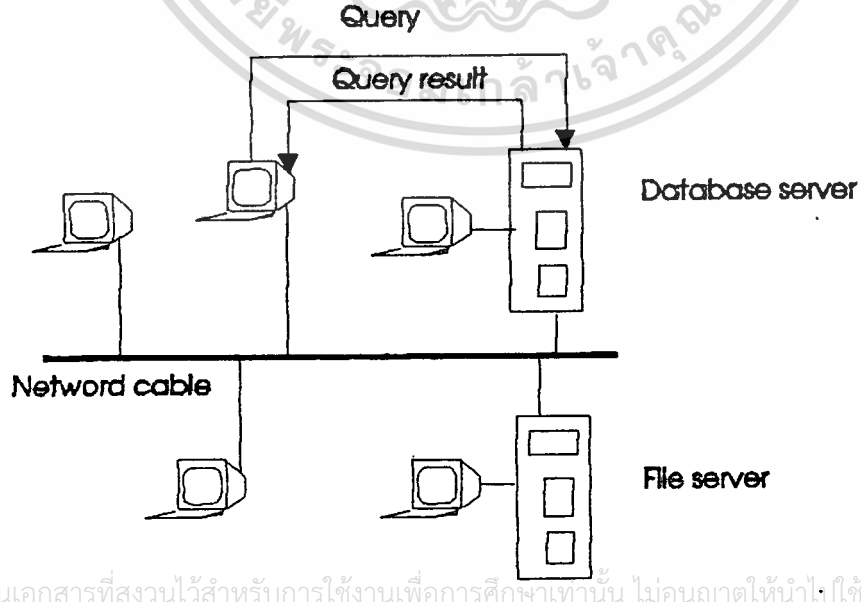
ระบบนี้ performances จะแย่ลงเมื่อจำนวนผู้ใช้เพิ่มมากขึ้นจึงไม่สามารถจะรองรับจำนวนผู้ใช้ได้มากเท่ากับ host-based system และระบบ PC-based นี้ไม่สามารถรับประกัน data integrity ได้เนื่องจากอนุญาติให้มีการทำ direct access กับ data file โดยไม่อยู่ภายใต้การควบคุมของระบบจัดการฐานข้อมูลที่สร้าง data file นั้นขึ้นมา integrity จึงถูกละเมิดได้

2.4.3 Client/Server database

ระบบนี้จะประกอบด้วย

1) client เป็นคอมพิวเตอร์ที่รันแอปพลิเคชั่นของผู้ใช้

2) server เป็นคอมพิวเตอร์ที่รันระบบจัดการฐานข้อมูล (database server) หรือเป็นคอมพิวเตอร์ที่จัดการการจัดสรรทรัพยากร เช่น disk space (file server) หรือ printer (printer server)



รูปที่ 2.10 ภาพแสดงระบบแบบ Client/Server

client จะจัดการทุกๆอย่างเกี่ยวกับหน้าจอและ input/output processing ของผู้ใช้ส่วน server จะจัดการเกี่ยวกับการประมวลผลข้อมูล การเข้าถึง disk ฯลฯ เช่นในกรณีที่ได้รับ query ที่เกี่ยวข้องกับข้อมูลในฐานข้อมูลจากผู้ใช้แอปพลิเคชั่น จะทำการส่ง request นั้นผ่านเครือข่ายไปที่ตัว server และตัว server จะจัดการประมวลผล request นั้นจนได้ผลซึ่งเป็นคำตอบออกมา แล้วส่งข้อมูลที่เป็นคำตอบของ query นั้นกลับไปให้ตัว client ที่เป็นตัว requestจะเห็นได้ชัดว่าสามารถลด traffic ได้อย่างมากเมื่อเทียบกับระบบ PC-based

client มักจะเป็น PC ในขณะที่ server สามารถเป็นได้ตั้งแต่ PC ไปจนถึงเมนเฟรม

การทำ downsizing คือการพยายามใช้work station technologies แทนเมนเฟรมหรือมินิคอมพิวเตอร์ ซึ่งจุดดีก็คือผู้ใช้สามารถที่จะเลือก platform ที่เห็นว่าเหมาะสมกับงานของตนได้

จุดประสงค์ของ client/server คือ

1) การลดค่าใช้จ่ายเพราะ work station มีค่าใช้จ่ายซึ่งถ้ามองโดยรวมจะต่ำกว่าเมนเฟรมมากรวมทั้ง การบำรุงรักษาเมนเฟรมก็มีค่าใช้จ่ายมากกว่า

2) สำหรับCPU intensive job จะให้ throughput ที่สูงขึ้นเพราะการประมวลผล ทำได้ที่ตัว client เลยไม่โหลดงานทุกอย่างมาที่ตัวคอมพิวเตอร์หลัก

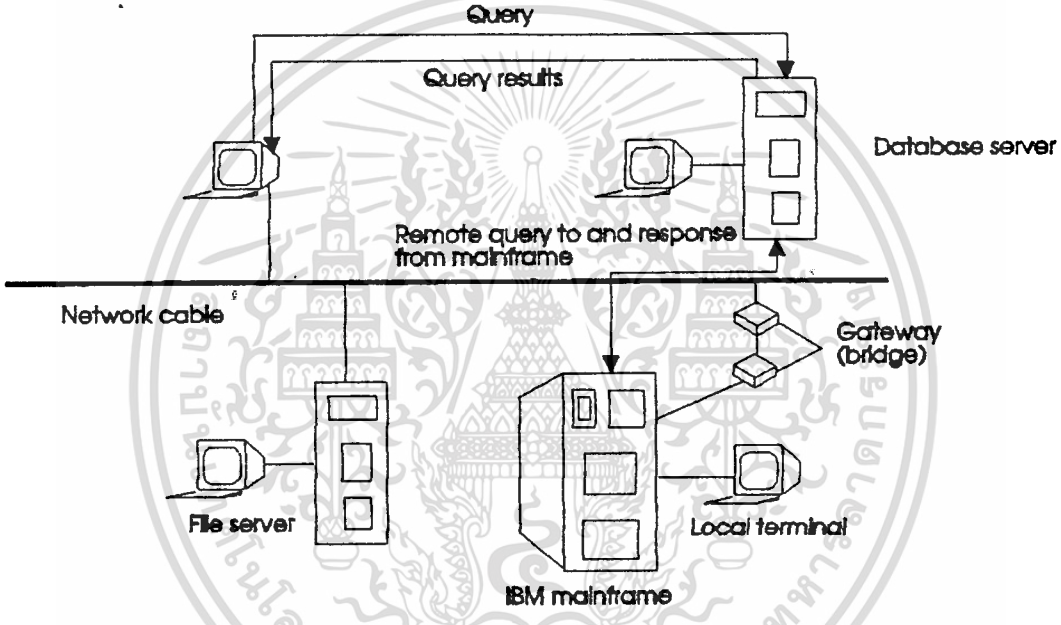
3) แก้ปัญหาเกี่ยวกับการกิน CPU time เพื่อการใช้ Graphic User Interface (GUI) บนระบบการประมวลผลบนคอมพิวเตอร์หลักได้ เพราะสามารถผลักภาระมาให้ตัว client จัดการเองได้

4) เป็นระบบเปิด ที่เปิดโอกาศให้ผู้ใช้เลือก platform ได้เอง

ในปัจจุบัน performance ของ PC ในด้านต่างเพิ่มสูงขึ้นมากทั้ง CPU performanc หนมวยความจำหลักและ storage ซึ่งเป็นอีกเหตุผลที่ทำให้ PC สามารถที่จะรองรับแอปพลิเคชั่นที่ซับซ้อนต่างๆได้และสำหรับระบบที่ผู้ใช้มีความคุ้นเคยกับ PC อยู่แล้ว การใช้ระบบ client/server จะทำให้ลดเวลาในการพัฒนาของแอปพลิเคชั่นลงได้และความเป็นไปได้ในการพัฒนาแอปพลิเคชั่นด้วยตัวเองของผู้ใช้ก็มีมากขึ้นด้วย รวมทั้งจากการที่ผู้ใช้มีความเข้าใจในระบบ การปรึกษาหารือการระบุความต้องการต่อ system specialist จะทำได้ถูกต้องชัดเจนขึ้น แอปพลิเคชั่น ต่างๆที่ถูกพัฒนาขึ้นมาสามารถจะตอบสนองความต้องการได้เป็นอย่างดี

2.4.4 Distributed Processing

ข้อมูลสามารถที่จะถูกใช้ร่วมกันระหว่างหลายๆระบบคอมพิวเตอร์หลักได้โดยผ่านทั้งการเชื่อมต่อโดยตรง (บน network เดียว) และบนการเชื่อมต่อระยะไกล (ผ่านโครงข่ายโทรศัพท์) เป็นการขยายขอบเขตการขยายและแลกเปลี่ยนข้อมูลกัน เช่นระหว่างแผนกที่อยู่ไกลกันทำให้ผู้ใช้สามารถที่จะเข้าถึงข้อมูลข้าม site ได้

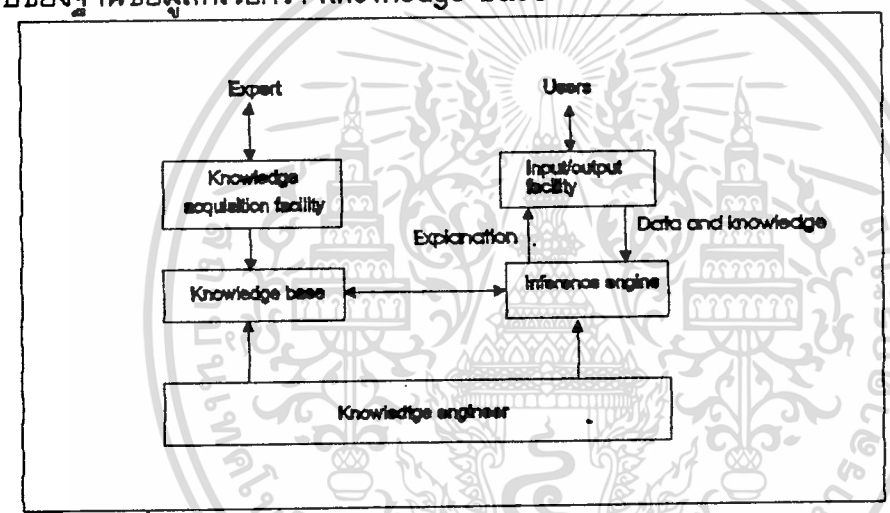


รูปที่ 2.11 ภาพแสดงการประมวลแบบ Distributed

2.5 Expert System

Expert System เป็นระบบที่ถูกออกแบบมาเพื่อทำหน้าที่ในการเป็นผู้เชี่ยวชาญและให้คำปรึกษากับมนุษย์ในเรื่องต่างๆ

Expert System คือโปรแกรมที่สามารถแก้ไขปัญหาได้หลายอย่าง โดยที่ทั่วไปยอมรับแล้วว่าเป็นปัญหาที่แก้ไขได้ยาก ต้องใช้เวลาในการแก้ไขปัญหานาน และปัญหานั้นก็ไม่ใช่ว่าสามารถแก้ได้โดยคนทั่วไป ต้องอาศัยผู้เชี่ยวชาญเฉพาะด้านจริงๆ บางครั้ง expert system จะถูกเรียกว่าเป็นระบบฐานความรู้ (knowledge based system) เนื่องจากการทำงานของระบบจะต้องอาศัยความจริงและกฎของเรื่องนั้นๆมารวบรวมเป็นความรู้แล้วเก็บไว้ในรูปของฐานข้อมูลที่เรียกว่า knowledge base



รูปที่ 2.12 รูปแสดงโครงสร้างของ Expert System

โครงสร้างของ Expert System

1). knowledge engineer มีหน้าที่ในการสร้าง expert system โดยอาศัยการนำความรู้จากผู้เชี่ยวชาญที่เป็นมนุษย์มาสร้างเป็นฐานความรู้ให้กับ expert system โดยอาศัยเครื่องมือที่ใช้ในการพัฒนา expert system

2). input/output facility หรือหน่วย input/output เป็นส่วนที่เชื่อมต่อระหว่างผู้ใช้ระบบกับระบบ

3). inference engine ทำหน้าที่ในการหาเหตุผลจากความรู้ (knowledge) ที่มีอยู่ในฐานความรู้ร่วมกับความจริง (fact) ใหม่ที่ได้รับจากผู้รู้ เพื่อที่จะให้ได้คำตัดสินหรือข้อวินิจฉัยออกมาเป็นคำตอบให้กับผู้ใช้

4). knowledge base คือฐานความรู้ส่วนที่ทำหน้าที่ในการเก็บความรู้ที่อยู่ในรูปของ facts และ rules

5). knowledge acquisition facility เป็นหน่วยดึงความรู้ที่ทำให้ระบบสามารถรับเอาความรู้เพิ่มเติมจาก knowledge engineer หรือการรับความรู้เข้ามาเก็บในฐานความรู้โดยอัตโนมัติ

2.6 หลักการและหน้าที่ของ DBMS

2.6.1 Process Architecture

จะอธิบายการ implementation ของ multiserver RDBMS ใน Operating system ซึ่งแบ่งออกเป็น 3 model คือ

1) **Process Model**

เป็นสถาปัตยกรรมที่ support "Single threaded" ("threaded" ในความหมายของ DBMS คือ execution path)

- มีหลักการทำงาน คือ 1 ผู้ใช้ : 1 process server
- สามารถทำ multiprogramming ได้

2) **Single Server Model**

- ผู้ใช้ แต่ละคนจะส่ง request message เพื่อให้ single server process ทำหน้าที่ serve
- ยังถือว่าเป็น " single threaded "เพราะยังถือว่า มี execution path เพียง path เดียว

3) **Multiple Server Model**

- มี RDBMS process หลายตัวคอย serve users ได้หลายคน
- ถือว่าเป็น " multithreaded"

2.6.2 Data Independence

Data Independence ก็คือการที่เราเปลี่ยน storage structure (ในทาง physical นั้น ข้อมูลถูกเก็บไว้อย่างไร) หรือ การเปลี่ยนยุทธวิธีในการเข้าถึงได้โดยไม่มีผลต่อแอปพลิเคชั่น เช่น สามารถเปลี่ยนจาก index file ไปเป็น hash-addressed file โดยไม่ต้องมีการทำ major modification ในแอปพลิเคชั่นเรา

ความจำเป็นที่ต้องเป็น Data Independent มีอยู่ 2 ประการหลักคือ

1) ในแอปพลิเคชั่นที่ต่างกันอาจมีมุมมอง (view) ที่ต่างกันบนข้อมูลเดียวกัน เช่นมีแอปพลิเคชั่น 2 อันคือ A,B ซึ่งแต่ละอันมี file ของตนเองและใช้ field "customer balance" ทั้งคู่ แต่ A จัดเก็บ value นี้เป็น decimal และ B เก็บเป็น binary ซึ่ง DBMS ควรจะมีความสามารถในการจัดการทำ conversion สำหรับการ represent รูปแบบที่ต่างกันนี้ เช่นสมมติถ้าตัดสินใจที่จะเก็บเป็นแบบ decimal แล้วในการที่ B จะเข้าถึงก็จะต้องมีการทำ conversion
แปลงจาก binary เป็น decimal และ decimal เป็น binary

2) DBA จะต้องมีความอิสระที่จะเปลี่ยน storage structure หรือวิธีในการ เข้าถึงโดยไม่ต้องแก้ไขแอปพลิเคชั่นที่มีอยู่ เช่น อาจมีข้อมูลชนิดใหม่เพิ่มเข้าไปในฐานข้อมูล หรือมีการปรับปรุง standard ใหม่ หรือมีการเปลี่ยนแปลง application priority หรือมีการเพิ่ม storage device ชนิดใหม่ๆเข้าไป และอื่นๆ ซึ่งก็เป็นหน้าที่ของ DBMS ที่จะต้อง detect การเปลี่ยนแปลงนี้แล้วจัดการดูแลให้

Data Independence มี 2 ประเภทคือ

1) Physical Data Independence คือความเป็นอิสระของการเปลี่ยนแปลงดังนี้

- Representation of numeric data เช่นเป็น เปลี่ยนจากการเก็บแบบ decimal เป็นแบบ binary หรือจาก real เป็น complex
- Representation of character data เช่นเป็น EBCDIC ,ASCII
- Unit for numeric data เช่นเปลี่ยนจาก inch -> centimeter
- Data Coding เช่นเปลี่ยนจากเก็บเป็น green,blue,red แทนด้วย 1,2,3 โดย
  1=green,2=blue,3=red
- Structure of stored file เช่น จาก single storage เป็น multiple storage

2) Logical Data Independent คือการเปลี่ยนแปลง, การเพิ่ม column หรือสร้างตารางใหม่จะไม่มีผลต่อ program ที่มีอยู่เดิม

2.6.3 Concurrency Control

หมายถึงการควบคุม transaction ที่ทำงานพร้อม ๆ กันและมีผลกระทบซึ่งกันและกันเพื่อที่จะป้องกันไม่ให้เกิดความไม่ถูกต้องตรงกันของฐานข้อมูลซึ่งจะยึด concept ที่ว่า serializability โดยสามารถใช้กลไกในการควบคุมดังต่อไปนี้

1) **Locking Protocol**

เป็นชุดของกฎซึ่งเป็นสภาวะเมื่อ transaction อาจจะ lock และ unlock แต่ละ data item ใน ฐานข้อมูลซึ่งสามารถแบ่ง lock ได้เป็นหลายประเภทเช่น exclusive lock (read lock), share lock (write lock)โดยจะตรวจสอบว่าสามารถ lock resource ได้หรือไม่จาก compatibility matrix

2) **Timestamp-ordering Protocol**

ทำหน้าที่เลือกลำดับระหว่าง 2 transaction ใดๆ ดังนั้นถ้า timestamp ของ Ti จะเล็กกว่า timestamp ของ Tj schedule จะเป็นไปตามลำดับที่จะทำให้เกิด serializability คือ Ti จะปรากฏก่อน Tj และถ้าไม่เป็นไปตามลำดับนี้ ก็จะ roll back transaction กลับไปที่จุดเดิมจะเห็นว่าลำดับที่ทำให้เกิด serializability จะพิจารณาจาก *timestamp*

3) **Validation Technique**

เป็นวิธีควบคุม concurrency ในกรณีที่ transaction ส่วนใหญ่เป็น transaction ที่ read-only ดังนั้นอัตราการขัดแย้ง ระหว่าง transaction จะต่ำ วิธีนี้จะแบ่ง life time ของ transaction เป็น 3 phase คือ

1) Read phase : ค่าของ data items ต่าง ๆ จะถูกอ่านและเก็บไว้ในตัวแปร Ti

2) Validation phase : ตรวจ constraint ว่าถ้า write แล้วจะขัดแย้งหรือไม่

3) Write phase : การ write ใน phase ก่อนหน้านี้จะ write ลงตัวแปรชั่วคราว ดังนั้นใน phase นี้จะ แก้ไขลงฐานข้อมูลจริงๆหรือไม่ก็ roll back หากไม่เป็น serializability

4) **Multiversion scheme**

แต่ละ operation ของ write(Q) จะสร้าง version ใหม่ๆของ Q และเมื่อมี operation read(Q) ระบบก็จะเลือก versionใด version หนึ่งให้ read โดยจะเลือก version ที่ทำให้เกิดเป็น serializability

5) Multiple Granularity

จะยอมให้ data items มีขนาดต่างๆ และกำหนด hierarchy ของ data items ซึ่งจะแสดง hierarchy ของ data items ให้อยู่ในรูปของ tree

จะยอมให้มีการ lock โดยจะพิจารณาตามลำดับ root-to-leaf ในขณะที่จะ unlock ตามลำดับ leaf-to-root orders

กลไกนี้จะใช้หลักการ lock-base Protocol แต่เพิ่ม Locking Primitive คือมี

- exclusive locks (X-lock)
- shared locks (S-lock)
- intended exclusive locks (IX-lock)
- intended shared locks (IS-lock)
- shared intended exclusive locks (SIX-lock)

และการ locks ของแต่ละ transaction จะทำได้โดยใช้การตรวจสอบตาราง Compatibility matrix ว่าสามารถ lock ได้หรือไม่

2.6.4 Backup and Recovery

เนื่องจากระบบคอมพิวเตอร์ก็เหมือนกับระบบทาง electronic อื่นๆตรงที่มีโอกาสที่จะเกิด failure ได้ซึ่งคำว่า failure นั้นก็มีหลายอย่าง เช่น disk crash,ไฟดับ, software error และอื่นๆ ซึ่งก็ล้วนแล้วแต่มีผลต่อการหายของ information ในระบบฐานข้อมูล ดังนั้นการ recovery จึงเป็นส่วนหนึ่งของ DBMS ที่มีหน้าที่รับผิดชอบต่อการจับว่ามี failure และการกู้ข้อมูลกลับมาให้อยู่ในสภาวะที่ consistent เหมือนเดิมก่อนการเกิด failure

Transaction คือ กลุ่มของงานที่ยอมให้มีการละเมิด integrity constraint ได้และเมื่อเสร็จงานแล้ว integrity constraint ต้องถูกต้องหรือพูดง่ายๆก็คือ transaction คือกลุ่มของ operations ที่แสดง ฟังก์ชั่นเดียวๆในแอปพลิเคชั่นของฐานข้อมูล

Transaction State มีดังนี้คือ

- Active คือ initial state
- Partially commited คือ execute statement สุดท้ายเรียบร้อยแล้ว
- Failed คือ ไม่สามารถดำเนินการ execute อย่างปกติต่อไปได้
- Aborted คือ transaction ถูก roll back และ state ของฐานข้อมูล ถูกกู้กลับมาที่สภาวะก่อนที่จะเริ่มต้น transaction
- Commited คือ การ successful completion

ในการ recovery จะมีวิธีดังนี้

1) Log - Based Recovery แบ่งเป็น
   - Defered Database Modification
   - Immediate Database Modification
2) Shadow Paging
3) ใช้การ Copy Database เป็น backup เก็บเอาไว้

Log - Based Recovery

ในการใช้วิธีนี้จะต้องมีสิ่งที่เกี่ยวข้องคือ Database Log หรือ Log file ซึ่งเป็นที่ๆเอาไว้เก็บสิ่งต่างๆดังนี้

- Transaction name เป็นชื่อ unique name ของ transaction ที่ทำ write operation
- Data Item name เป็นชื่อ unique name ของ data item ที่ถูก write
- Old value คือ ค่าของ data item ก่อนการ write
- New value คือ ค่าของ data item หลังการ write

ชนิดของ log record มีดังนี้

- <Ti,start> เป็น record ที่บอกว่า transaction Ti เริ่มต้น
- <Ti,Xj,V1,V2> หมายถึง Ti กำลังwrite บน data item Xj และ Xj มีค่า V1 ก่อน write และ V2 หลังจาก write แล้ว
- <Ti,commit> หมายถึง Ti commit แล้ว

ซึ่งจะมี Write Ahead Protocol ดังนี้

ก่อนที่จะมีการแก้ไขฐานข้อมูล จะต้องมีการสร้าง log record เก็บเอาไว้แล้วถึงยอมให้มีการเปลี่ยนแปลงใดๆ ดังนั้นเราจึงสามารถที่จะ undo การเปลี่ยนแปลงใดๆ ที่ได้เกิดขึ้นกับฐานข้อมูลได้โดยใช้ค่า old value ที่เก็บไว้

Log - Based Recovery แบ่งเป็น 2 วิธี ดังนี้

1) Defered Database Modification

หลักการของวิธีนี้คือมีเฉพาะ AIJ คือ log record เก็บเฉพาะค่า new value หลังการ write เพียงอย่างเดียวเท่านั้น ไม่มี BIJ (ค่า old value ก่อนการ write)

วิธีนี้ข้อมูลบน disk จะไม่โดนแตะต้องเลยจนกว่าจะ write commit แล้วจึงค่อยถ่ายข้อมูลลง disk

ในการ recovery จะใช้ procedure ดังนี้

- Redo คือ transaction ใดที่ commit ไปแล้วก่อนที่ system จะ crash transaction นั้นก็จะถูก Redo โดยเอาค่า new value ที่เก็บเอาไว้ใน log record ใส่ลงไป ส่วน transaction ใดที่ยังไม่ commit ก็ไม่ต้อง Redo คือไม่ต้องทำอะไรเพราะยังไม่มีการ write ลงบน disk

ข้อดีของ Defered Database Modification

1) ไม่ต้องมี BIJ ทำให้ประหยัดเนื้อที่ใน memory

2) เหมาะกับ transaction ขนาดเล็กๆ เพราะเร็ว คือรอให้มีการ แก้ไข ฐานข้อมูล หลายๆครั้งแล้วก็ค่อย commit ทีนึงแล้วจึงค่อยถ่ายข้อมูลลงบน disk ซึ่งจะทำให้มีข้อดีเพิ่มอีกคือจะ save I/O operation ด้วย

ข้อเสียของ Defered Database Modification

1) ในกรณีที่เป็น transaction ใหญ่ วิธีนี้จะไม่เหมาะเพราะจะเกิดการเปลี่ยนแปลงจำนวนมากซึ่งเราจะต้องรอจนกว่า commit แล้วจึงจะได้ถ่ายข้อมูลลง disk จะไม่ทะยอยถ่าย

2) การ recovery ต้องทำทั้ง transaction เนื่องจากมี AIJ อย่างเดียว

3) ตุนเนื้อที่ใน buffer เยอะ ดังนั้นไม่เหมาะกับ transaction ขนาดใหญ่

2. Immediate Database Modification

หลักการของวิธีนี้คือใน log record จะเก็บทั้ง AIJ และ BIJ คือทั้ง old value และ new value

Procedure ของการ recovery มีดังนี้

- Undo คือ ขณะ crash ถ้า transaction ใดที่ยังไม่ได้ commit ก็จะนำค่า old value ที่เก็บอยู่ใน log record นั้นใส่ลงไป

- Redo คือ ขณะ crash ถ้า transaction ใดที่ commit ไปแล้วก็จะนำค่า new value ที่เก็บใน log record นั้นใส่ลงไป

ข้อดีของ Immediate Database Modification

1) ไม่ต้องเก็บค้างข้อมูลใน buffer เยอะ

2) เหมาะกับ transaction ขนาดใหญ่เพราะถ้าหากมีการ write log record ลงไปแล้วก็จะสามารถทะยอยถ่ายข้อมูลลง disk ไปได้เลย ไม่ต้องรอจน transaction นั้น commit ก็ได้ ทำให้ถ่ายข้อมูลเสร็จเร็วกว่าใช้ Defered Mode ที่ต้องรอจน commit แล้วจึงถ่ายข้อมูลได้ ที่เป็นเช่นนี้เพราะมีการเก็บทั้ง AIJ และ BIJ ทำให้สามารถ undo ข้อมูลของ transaction ที่ยังไม่ได้ commit ได้

ข้อเสียของ Immediate Database Modification

1) เปลือง I/O operation เพราะมีการ write log record บ่อย ทำให้เสียเวลาจึงทำให้ไม่เหมาะกับ transaction ขนาดเล็กๆ

2) เปลืองเนื้อที่ในการเก็บค่า AIJ

ในการ Recovery แบบ Log - Based นี้ยังมีจุดที่สำคัญอีกจุดคือจุด Checkpoint

Checkpoint คือเวลาที่เกิด action ต่อไปนี้

- log record ทั้งหมดที่ยังอยู่ใน main memory จะถ่ายลง stable storage
- ข้อมูลที่ถูกแก้ไขแล้วอยู่ใน buffer block จะถูกถ่ายลง stable storage

Checkpoint time นั้นแล้วแต่ว่าเราจะเลือกว่าให้ถี่แค่ไหนขึ้นกับ volume ของ transaction ถ้าเป็น transaction ขนาดใหญ่ๆก็ไม่ควรให้ถี่นักเพราะต้องเสียเวลาถ่ายข้อมูล
จาก buffer ลง disk ซึ่งมีจุดให้พิจารณาดังนี้

ถ้า Tc ห่าง -> เปลืองเนื้อที่ใน main (ข้อมูลในbuffer ที่ยังไม่ถ่ายลง disk เยอะ)

ถ้า Tc ถี่ -> ตัว Tc อาจเป็น bottle neck คือต้องใช้เวลาในการถ่ายข้อมูลลง

ประโยชน์ของ Checkpoint คือ Recovery time จะเร็ว

Shadow Paging

หลักการของวิธีนี้คือจะมี page table 2 อันคือ

- Current Page Table คือ ตารางที่เปลี่ยนแปลงไปตาม write operation ใน transaction
- Shadow Page Table คือ ตารางที่เก็บรักษาไว้และไม่มีการเปลี่ยนแปลงระหว่าง transaction

Current Page Table จะถูกเก็บเป็น Shadow Page Tabel ในจุดที่ทำการ commit แล้วหลังจากนั้นพอเริ่ม transaction ใหม่ก็จะมี Current Page Table ใหม่ ซึ่งมีสภาวะเริ่มต้นเหมือน Shadow Page Table ทุกอย่าง

ข้อดีของวิธี Shadow Page Table

1) Recovery time จะเร็วมากๆ เพียงแค่ยกเลิก Current Page Table แล้วเปลี่ยนไปใช้ Shadow Page Table เท่านั้น

2) เหมาะกับงานขนาดเล็กๆที่ต้องการ response time ดีๆ เพราะในการ commit ก็เป็นการเขียนข้อมูลลง disk เท่านั้น

งานที่เหมาะกับ Shadow Page Table ได้แก่ งานประเภท OLTP

ข้อเสียของวิธี Shadow Page Table

1) disk กระจัดกระจายเพราะ update แต่ละครั้งต้องสร้าง page table ใหม่ เรื่อยๆอาจทำให้ table กระจัดกระจายไม่อยู่ใน cylinder เดียวกันทำให้ Performance ต่ำลง

2) ไม่เหมาะกับงาน transaction ขนาดใหญ่เพราะการ commit หมายถึงการถ่ายข้อมูลลง disk จริงๆเลย ดังนั้น commit อาจช้ากว่าพวก log based

3) ไม่มี AIJ ดังนั้นการ back up ต้องทำทั้ง volume

2.6.5 Data Integrity

หมายถึง ความถูกต้องของข้อมูลโดยควบคุมความถูกต้องโดย integrity constraint ซึ่งทำให้แน่ใจว่าการเปลี่ยนแปลงฐานข้อมูลนั้นเป็นเพราะผู้ใช้ผู้ที่มีสิทธิ์เปลี่ยนแปลงมิใช่เป็นผลมาจากการไม่มี data consistency ซึ่งเราจะพิจารณาที่ Referential Integrity

Referential Integrity หมายความว่า ความถูกต้องที่ค่าที่ปรากฏในความสัมพันธ์หนึ่งเป็น ชุดของ attribute ก็ปรากฏชุดของ attribute นี้ในความสัมพันธ์ อื่น ๆ แน่นอน

Trigger คือ Procedure ซึ่งถูกกำหนดว่าจะต้อง execute เมื่อเหตุการณ์ที่กำหนดเกิดขึ้นคล้ายกับเป็นผลกระทบที่เกิดจากการเปลี่ยนแปลงฐานข้อมูล

ในการออกแบบ Trigger เราจะต้อง

1) กำหนดสภาวะซึ่งเป็นตัวกำหนดว่าเมื่อเกิดเหตุการณ์นี้จะต้อง execute trigger

2) กำหนดการกระทำที่จะเกิดเมื่อมีการ execute trigger

2.6.6 Optimization Performance

ในเรื่องของการ optimaization จะเป็นการเลือกกลยุทธในการประเมิน relational expression ที่มีประสิทธิภาพซึ่งประเด็นสำคัญของเรื่องนี้จะมุ่งไปที่การทำ Query Optimazation

Query Optimization จะเป็นการเลือกวิธีในการเข้าถึงข้อมูลให้เร็วที่สุดโดยจะมีตัว Query Optimizer เอาไว้เป็นตัวเลือกในการเข้าถึงข้อมูลจริงๆในระดับ physical อาจมองเป็น expert system ตัวนึงเพราะจะใช้กฎต่างๆหาทางเข้าถึงข้อมูลโดย Query Optimizer จะรับ parse tree มาจาก Query Parser ซึ่งทำหน้าที่ตรวจสอบไวยากรณ์ของภาษาฐานข้อมูลแล้วสร้างเป็น algebra tree ส่งมาให้ Query Optimizer

ในการเลือกใช้ DBMS เราต้องพิจารณาประเด็นที่สำคัญ 2 ประการคือ

1) DBMS สามารถจัดการ query แบบ Compile ได้หรือไม่ คือการมีฟังก์ชั่นของ query ที่มีการใช้งานบ่อยๆหลายๆรูปแบบทำเป็นฟังก์ชั่นเก็บไว้โดยถ้ามีรูปแบบของ query ที่เก็บไว้ก็จะสามารถทำการเรียกใช้ query รูปแบบนั้นได้โดยส่งแค่พารามิเตอร์ผ่านเข้าไปเท่านั้น ซึ่งการ Precompiled จะมีผลดีตรงที่เร็วไม่ต้องเสียเวลา Interpret

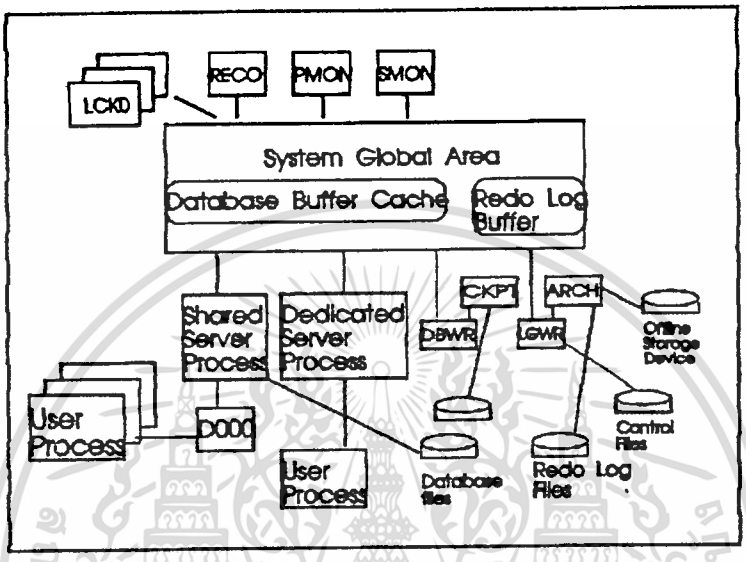
2) DBMS สามารถจัดการ query แบบ Interpret เป็นวิธีการ query แบบธรรมดาที่ทุกๆ DBMS มี คือจะเป็นการรับชุดของ query เข้ามายาวๆแล้วทำการ Interpret ครั้งต่อครั้งไม่มีการเก็บเอาไว้เป็นฟังก์ชั่น ซึ่งวิธีนี้จะมีข้อดีที่ไม่ต้องเปลืองเนื้อที่ในหน่วยความจำเพื่อรักษาฟังก์ชั่นนั้นไว้

บทที่ 3

การวิเคราะห์ระบบจัดการฐานข้อมูล ORACLE Version 7

3.1 System Structure และ Storage Structure ของ ORACLE

3.1.1 System Structure



รูปที่ 3.1 รูปแสดง Memoey Structures และ Process ของ ORACLE

ORACLE สร้างและใช้โครงสร้างของหน่วยความจำในการทำงานต่าง ๆ ให้เสร็จตามต้องการโครงสร้างของหน่วยความจำพื้นฐานที่ใช้กับ ORACLE ประกอบด้วย

1) System Global Area (SGA)

เป็นหน่วยความจำซึ่ง ORACLE ใช้เก็บข้อมูลและข้อมูลเกี่ยวกับการควบคุมส่วนต่างๆของ ORACLE (ORACLE instance หมายถึงองค์ประกอบของ process และหน่วยความจำของบัฟเฟอร์) โดย SGA จะถูกจองไว้เมื่อเริ่มและจะยกเลิกเมื่อส่วนนั้นเลิกการทำงาน information ที่เก็บใน SGA จะแบ่งเป็นหลายชนิด

1.1) Database Buffer Cache

ฐานข้อมูลของบัฟเฟอร์จะเก็บชุดของข้อมูลที่ใช้ล่าสุดและชุดของฐานข้อมูลของบัฟเฟอร์ใน instance จะเรียกว่า " database buffer cache "

ของบัฟเฟอร์ นี้ใช้เก็บข้อมูลที่มีการแก้ไขแล้วยังไม่ได้เก็บลงดิสก์เพื่อลด disk I/O

1.2) Redo Log Buffer

จะเก็บ " redo entries "(ซึ่งหมายถึง log ของการเปลี่ยนแปลงที่เกิดขึ้นในฐานข้อมูล) ลงใน redo log buffers ซึ่งจะถูกเขียนลง online redo log files ซึ่งนำไปใช้ในการทำ recovery

1.3) Share Pool

เป็นการร่วมกันใช้หน่วยความจำคล้ายกับใช้ SQL areas ร่วมกัน

1.4) Cursors

ตัวจัดการ หน่วยความจำด้วยข้อความที่กำหนด

2) Program Global Area

เป็นหน่วยความจำของบัฟเฟอร์ซึ่งเก็บข้อมูลและข้อมูลเกี่ยวกับการควบคุมสำหรับ server process ซึ่งจะถูกสร้างขึ้นเมื่อ server process เริ่มการทำงาน

processes นั้นหมายถึง - " Threaded of control " หรือ

- กลไกควบคุมการทำงานของระบบให้เป็นไปตามลำดับขั้นที่กำหนด

ORACLE ประกอบไปด้วย 2 process คือ

1) User (Client processes)

- จะสร้างและดูแลกระบวนการทำงานของ ผู้ใช้ เพื่อใช้ execute software code ของแอพพลิเคชั่น โปรแกรม (เช่น Pro *C program) หรือ ORACLE tool (เช่น SQL * DBA)

- จะทำหน้าที่จัดการการติดต่อกับ server process จะโดยผ่านโปรแกรมที่เชื่อมการติดต่อ

2) ORACLE processes ประกอบไปด้วย

2.1) Server process

ทำหน้าที่จัดการติดต่อสื่อสารระหว่าง server process และ user process เพื่อทำตาม user process ต้องการ

ORACLE มีข้อกำหนดเกี่ยวกับรูปแบบหลายรูปแบบโดยจะแบ่งตามจำนวนของผู้ใช้ต่อ server process แบ่งเป็น

1) dedicated server configurations

:- server process จะรองรับ user process เพียง 1 user process

2) Multi-threaded server configurations

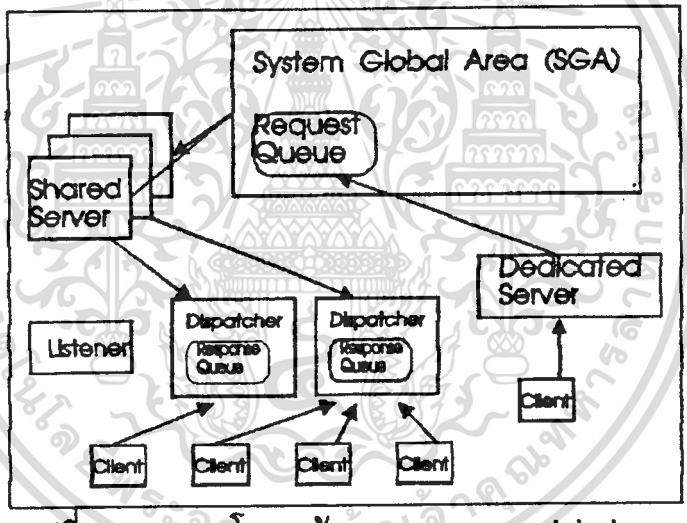
:- user process หลาย process ใช้และร่วมกันใช้server process จำนวนหนึ่งซึ่งเป็นจำนวนน้อย

:- ใช้เพื่อลดจำนวน server process ให้น้อยที่สุดและมีประสิทธิภาพมากที่สุด

:- user process และ server process สามารถวิ่งอยู่บนอุปกรณ์ที่ต่างกัน ซึ่ง user process และ server process ต้องแยกจากกัน

:- ประโยชน์ - ลด overhead จาก context switching เนื่องจาก server process ลดลงเป็นจำนวนที่น้อยและมีประสิทธิภาพและเป็นการขจัดปัญหา bottleneck ของ CPU, I/O, หน่วยความจำ และ operating system

- เพิ่มจำนวนผู้ใช้ต่อ node
- จำนวนของ server process จะเป็นแบบ dynamic กล่าวคือสามารถเปลี่ยนแปลงได้ตามขนาดของงาน โดยไม่มีผลกระทบกระเทือนต่อแอพพลิเคชั่น



รูปที่ 3.2 แสดงโครงสร้างของ system global area

จะใช้ Multi-threaded server configurations เนื่องจากเป็นสถาปัตยกรรมที่มีประสิทธิภาพสูงที่สุดและช่วยแก้ปัญหาที่เกิดจากสถาปัตยกรรมแบบอื่น ๆ

2.2) Background process

จะสร้างสำหรับแต่ละส่วนจะแสดง I/O และ monitor เพื่อเพิ่มความสามารถของ parallelism เพื่อให้มี performance ที่ดีขึ้นและเชื่อถือได้มากขึ้น

แต่ละ ORACLE instance จะมี background process มากมาย ดังต่อไปนี้

- Database Writer (DBWR):- ทำหน้าที่เขียนสิ่งที่แก้ไขใน blocks จาก database buffer cache ลงสู่ดิสก์

- Log Writer (LGWR) :- ทำหน้าที่เขียน redo log entries ลงสู่ disk

- Checkpoint (CKPT) :- เวลาที่กำหนดว่าเป็นเวลาที่เขียนสิ่งที่ถูกแก้ไขใน database ของบัฟเฟอร์ ลงใน data file ซึ่งเป็นหน้าที่ของ DBWR

- System Monitor (SMON) :- ทำหน้าที่แสดง instance recovery ในระบบ multiple instance (ใช้ parallel server)

:- SMON ของ instance ใด instance หนึ่งจะทำหน้าที่ recovery ในขณะที่ instance อื่น ๆ fail

- Process Monitor (PMON):- จะแสดง process recovery เมื่อ user process fail

- Archiver (ARCH) :- จะ copy online redo log files ไปยัง archival storage เมื่อ redo log files เต็ม

- Recover(RECO) :- ใช้ในการแก้ปัญหา distributed transaction ซึ่งปัญหานี้เกิดจากเน็ตเวิร์กหรือระบบเกิดข้อผิดพลาด ใน distributed database

- Dispatcher (Dnnn) :- จะใช้กับ multi-threaded server

:- ใช้สำหรับ communication protocol ต่าง ๆ

:- ทำหน้าที่หาเส้นทางให้ user process ใช้และ share server process

3.1.2 Storage Structure ของ ORACLE

ORACLE มีวิธีในการจัดการการเก็บข้อมูลในตารางหลายวิธี เช่น

Cluster

Cluster หมายถึงกลุ่มของตารางซึ่งใช้ data blocks ร่วมกัน ในลักษณะใช้ common คอลัมน์ร่วมกันโดยตาราง 2 ตารางซึ่งเป็น cluster กันที่ใช้ common คอลัมน์ร่วมกันจะถูกเก็บไว้ใน data blocks เดียวกัน

Cluster key

คือ คอลัมน์หรือชุดของคอลัมน์ซึ่งใช้ตารางแบบ cluster ร่วมกันโดยคอลัมน์ดังกล่าวนี้จะต้องระบุตอนสร้างและคอลัมน์เดียวกันนี้จะอยู่ในตารางซึ่งจะเพิ่มเข้า cluster โดยอัตโนมัติ

- cluster key มีได้ไม่เกิน 16 คอลัมน์ และค่าของ cluster key จะไม่เกิน 1/3 ของเนื้อที่ของข้อมูลใน blocks ของข้อมูล
- หากค่าของข้อมูลใน cluster คอลัมน์ มีการแก้ไขจะทำให้ต้องมีการเปลี่ยนแปลงตามตำแหน่งที่เก็บซึ่งอยู่ในระดับกายภาพ

ประโยชน์ของ cluster

1) ลด Disk I/O และลดเวลาที่ใช้ในการเข้าถึง

2) เก็บตารางที่มีความสัมพันธ์กันและ index data ใน cluster ได้ลดลง

ข้อควรพิจารณา

- cluster จะลดความสามารถของ INSERT statement เพื่อเปรียบเทียบกับเก็บตารางโดยแยก index เป็นของตน่เอง

- ตารางซึ่งมี blocks เป็นจำนวนมากควรเก็บไว้ใน cluster

- ตารางซึ่งเกี่ยวข้องกับ referential integrity และตารางที่มีการเรียกใช้ด้วย SELECT statement และ join ตารางบ่อย ๆจะเหมาะสมกับวิธี cluster เพราะทำให้ลดจำนวน data blocks ที่จะเข้าใช้ในการ queries

- ไม่เหมาะกับกรณีที่คอลัมน์ที่เป็น cluster key ที่มีการแก้ไขบ่อย ๆ

Hashing

- การใช้ hashing จะต้องสร้าง hash cluster และเรียกตารางเข้าไปใน cluster โดย rows ของตารางใน hash cluster จะเก็บและดึงข้อมูลตามผลที่ได้จาก hash function

- " hash function " ใช้ในการสร้างค่าตัวเลขที่มีความกระจายซึ่งค่านี้จะเรียกว่า " hash value "

- key ของ hash cluster อาจเป็นคอลัมน์ หรือชุดของคอลัมน์

- ORACLE จะจัดการเก็บ rows ใน hash cluster โดยใช้ cluster key value ที่ได้จาก hash function

- ในการหาหรือจัดเก็บ rows จะต้องใช้อย่างน้อย I/O 2ครั้ง< มักจะมากกว่านี้ > โดยใช้ I/O 1 ครั้งหรือมากกว่าในการหาและเก็บ key value ใน index และใช้ I/O มากกว่า 1 ครั้งสำหรับ read หรือ write rows ในตารางหรือ cluster

B*-tree index

ทำหน้าที่ดูลเพื่อให้ rows ใดๆมีเวลาที่ใช้ในการเข้าถึงพอๆ กับ branchblocks < blockที่อยู่เหนือกว่า > จะมี index ซึ่งชี้ไปยัง index blocks ที่อยู่ระดับต่ำกว่าและ index blocks ที่อยู่ในระดับต่ำสุด เรียกว่า " leaf blocks " ซึ่งจะเก็บ index ของค่าของข้อมูลซึ่งอยู่ในรูป ROWID (ใช้บอกตำแหน่งจริง ๆ ของ row)

- สำหรับ unique index จะใช้ ROWID 1 ค่าต่อค่าของข้อมูล

- สำหรับ non-unique index :- ROWID จะรวมอยู่ใน key ที่ใช้เรียงลำดับดังนั้นเราจึงใช้ index key และ ROWID ในการ เรียงลำดับ non-unique indexes

ประโยชน์ของการใช้ storage structure แบบ B*-tree

1) leaf blocks ของ tree ทุก ๆ blocks จะอยู่ในระดับความลึกเดียวกันหมดดังนั้น การดึงข้อมูลไม่ว่าอยู่ตำแหน่งใดจึงใช้เวลาพอๆกัน

2) B*-tree index จะอยู่ในสภาพสมดุลเสมอ

3) B*-tree จะทำให้สามารถ queriesได้ในช่วงกว้างรวมทั้งสามารถค้นหาได้อย่างมีประสิทธิภาพ

4) สามารถเพิ่มเติม, เปลี่ยนแปลง, ลบได้อย่างรวดเร็ว

5) เหมาะกับตารางทั้งขนาดเล็กและใหญ่โดยไม่คำนึงถึงขนาดของตารางที่โตขึ้น

3.2 Data Independence ของ ORACLE

Physical Data Independence

ORACLE มีคุณสมบัตินี้คือไม่ว่าจะเปลี่ยนลักษณะการเข้าถึง file ในระดับล่างเป็นแบบใดก็จะแก้เฉพาะในระดับล่างเท่านั้นระดับ application ที่เหนือกว่าไม่ต้องไปทำการเปลี่ยนแปลงใดๆ

Logical Data Independence

ORACLE มีคุณสมบัติข้อนี้คือ DBA สามารถที่จะเปลี่ยนแปลงตารางเพิ่ม column, ตัด column, เพิ่มตาราง สิ่งเหล่านี้จะไม่มีผลต่อการเข้าถึงตารางของผู้ใช้ไม่ต้องมีการแก้ไขโปรแกรมใหม่

3.3 Concurrency Control ของ ORACLE

ORACLE จัดการ Concurrency control โดยใช้

- Locks แบบต่างๆ
- Multi- version consistency model

3.3.1 Locks Mechanism

จะแบ่งออกเป็น 5 ประเภท คือ 1.Data locks (DML locks)
2.Dictionary locks (DDL locks)
3.Internal locks and latches
4.distributed locks
5.Paralell cache management locks (PCM)

หมายเหตุ :- DML หมายถึง Data Manipulation statements ซึ่งจัดการ ข้อมูล ของฐานข้อมูล เช่น querying, ลบ,แก้ไข, แทรกเพิ่มรวมทั้ง lock ตาราง หรือ view

DDL หมายถึง Data Definition statements ซึ่งใช้ ระบุ และ ดูแล objects และ ทำลาย objects

1) **Data locks (DML locks)**

เพื่อป้องกันตาราง ซึ่งมั่นใจได้ว่าข้อมูลในตารางทุกตารางจะถูกต้องตรงกันแม้ว่าจะมีผู้ใช้หลายคนที่เข้าถึงตารางเดียวกัน

แบ่งระดับของ data locks ได้ 2 level คือ 1)Row locks
2)Table locks

1.1) Row locks

เป็น data locks ที่จะทำการ locks เฉพาะ rows ที่ต้องการ โดยจะจัดเป็น " exclusive data locks " ก็ต่อเมื่อ locks เพื่อต้องการที่จะแก้ไขฐานข้อมูลโดยใช้ statement ต่อไปนี้ :- INSERT, UPDATE, DELETE, SELECT พร้อมกับ FOR UPDATE CLAUSE

rows จะถูก locks เพื่อว่าผู้ใช้อื่น ๆ จะได้ไม่สามารถมาแก้ไข rows ที่เข้าถึง อยู่และจะ hold locks จนกว่า transaction ที่ hold locks นั้น commit หรือ roll back

1.2) Table locks

transaction ที่ต้องการแก้ไขตารางด้วย statement ต่อไปนี้ :- INSERT,

UPDATE, DELETE, SELECT พร้อมกับ FOR UPDATE CLAUSE และ LOCK TABLE

table locks มีไว้เพื่อ 1. รักษาระดับของการเข้าถึงตารางแทน transaction
2. ป้องกันการขัดแย้งกับ DDL operations

- Mode ของ table locks แบ่งออกเป็น 5 mode คือ

1) **Row Share Table Locks (RS)**

มีหลักการทำงานคือ ใช้เมื่อ transaction hold table locks ในขณะที่มีการ locks rows ในตาราง และมีเจตนาที่จะแก้ไข ถือว่า locks แบบนี้เข้มงวดน้อยที่สุดและทำให้เกิดระดับของ concurrency มากที่สุดด้วย STATEMENT ต่อไปนี้ "LOCK TABLE table IN ROW SHARE MODE ;"

operations ที่อนุญาตให้เกิด :- จะยอมให้ transaction query, แทรกเพิ่ม, แก้ไข, ลบ หรือ lock rows ในตารางเดียวกันได้
:- จะยอมให้ transaction อื่นสามารถทำงานใน mode row share, row exclusive share, share, share row exclusive mode ได้ในตารางเดียวกัน

operations ที่ห้ามไม่ให้เกิด :- transaction อื่น ๆ จะอยู่ exclusive mode ไม่ได้

2) **Row Exclusive Table Locks (RX)**

มีหลักการทำงานคือใช้เมื่อ transaction hold locks นี้จะต้องเกิดการแก้ไข rows ในตารางแน่นอนด้วย STATEMENT ต่อไปนี้ "LOCK TABLE table IN ROW EXCLUSIVE MODE ;"

operations ที่อนุญาตให้เกิด :- จะยอมให้ transaction อื่นๆ query, แทรกเพิ่ม, แก้ไข, ลบ หรือ lock rows ในตาราง เดียวกันพร้อมๆกันได้
:- จะยอมให้ transaction อื่นๆ locks แบบ row exclusive และ row share mode ใน ตาราง เดียวกับที่อีก transaction locks อยู่ได้

operations ที่ห้ามไม่ให้เกิด :- transaction อื่นๆจะlocks แบบ share mode, share row exclusive, exclusive mode ไม่ได้

3) **Share Table Locks (S)**

ด้วย STATEMENT ต่อไปนี้ "LOCK TABLE table IN SHARE MODE ;"

operations ที่อนุญาตให้เกิด :- หากมี transaction hold locks แบบ share table locks ก็จะยอมให้ transaction อื่นๆ ทำ

ได้เพียงแค่ query และ locks rows ที่ระบุเท่านั้น
ซึ่งจะไม่ยอมให้ transaction อื่นๆมา แก้ไข
:- ดังนั้นจึงจะยอมให้ transaction อื่นๆ locks แบบ
share และ row share table mode ได้

operations ที่ห้ามไม่ให้เกิด :- transaction อื่นๆ จะlocks แบบ row exclusive, share row exclusive, exclusive mode ไม่ได้

4) Share Row Exclusive Table Locks (SRX)

ด้วย STATEMENT ต่อไปนี้ "LOCK TABLE table IN SHARE ROW EXCLUSIVE MODE ; "

operations ที่อนุญาตให้เกิด :- จะมีเพียง transaction เดียวเท่านั้นที่จะ hold locks แบบ share row exclusive table locks
ก็จะยอมให้ transaction อื่นๆ ทำได้เพียงแค่ query และ locks rows ที่ระบุเท่านั้น ห้ามแก้ไข
:- ดังนั้นจึงจะยอมให้ transaction อื่นๆlocks ได้แค่แบบ row share mode

operations ที่ห้ามไม่ให้เกิด :- transaction อื่นๆจะlocks แบบ row exclusive, share, share row exclusive, exclusive mode ไม่ได้

5) Exclusive Table Locks (X)

ด้วย STATEMENT ต่อไปนี้ "LOCK TABLE table IN EXCLUSIVE MODE ; "

operations ที่อนุญาตให้เกิด :- ก็จะยอมให้ transaction อื่นๆ ทำได้เพียงแค่ query
:- ดังนั้นจึงไม่ยอมให้ transaction hold locks ไม่ว่าจะเป็น mode ใด

operations ที่ห้ามไม่ให้เกิด :- transaction อื่นๆจะlocks แบบใดไม่ได้เลย

ตารางต่อไปนี้เป็นหลักการทำงานmode ของ table locks, action ที่จะทำให้เกิด table locks นั้นๆ และ action ที่จะยอมให้เกิดใน transaction อื่นหรือไม่ ในขณะที่มี transaction หนึ่ง hold table locks อยู่

ตารางที่ 3.1 ตารางแสดงหลักการทำงานของกลไกการ lock ของ ORACLE

SQL Statement	Mode of Table Lock				
	RS	RX	S	SRX	X
SELECT...FROM table	NONE	Y	Y	Y	Y	Y
INSERT INTO table	RX	Y	Y	N	N	N
UPDATE table	RX	Y	Y	N	N	N
DELETE FROM table	RX	Y	Y	N	N	N
SELECT FROM table FOR UPDATE OF	RS	Y	Y	Y	Y	N
LOCK TABLE table IN ROW SHARE MODE	RS	Y	Y	Y	Y	N
LOCK TABLE table IN ROW EXCLUSIVE MODE	RX	Y	Y	N	N	N
LOCK TABLE table IN SHARE MODE	S	Y	N	Y	N	N
LOCK TABLE table IN SHARE ROW EXCLUSIVE MODE	SRX	Y	N	N	N	N
LOCK TABLE table IN EXCLUSIVE MODE	X	N	N	N	N	N

จากตารางจะแสดงว่าข้อมูลถูก lock โดยอัตโนมัติไม่ว่าจะในระดับ rows locks และ table locks ว่ามีหรือไม่ ถ้ามีแล้วอยู่ในโหมดใด

จากตารางสามารถสรุปการทำงานของ default locking ได้ 2 แบบคือ

- Default Locking สำหรับการ Queries
- Default Locking สำหรับการ INSERT, UPDATE และ DELETE STATEMENT และพิจารณาเรื่อง Data locks Conversion และ Escalation

Default Locking สำหรับการ Queries

queries นั้นหมายถึง ประเภทของ statement ดังต่อไปนี้

SELECT

INSERT...SELECT...;

UPDATE...;

DELETE...;

แต่ไม่รวม statement ต่อไปนี้ SELECT...FOR UPDATE OF...; เป็นที่น่าสังเกตว่า INSERT, UPDATE และ DELETE STATEMENT ถือว่าเป็น implicit queries ในลักษณะเป็นส่วนหนึ่งของ statement

การ queries เป็น SQL statement ซึ่งอย่างน้อยจะต้องมี statement ใด statement หนึ่งที่ติดต่อกับ statement อื่นๆ ซึ่งเป็นการ read ข้อมูลเท่านั้น และจะส่งผลในลักษณะต่อไปนี้

- การ queries ไม่ต้องการ data locks ดังนั้นในขณะที่ queries นั้น transaction อื่นๆ สามารถ query ตารางนั้นได้ บางครั้งเรียกการ queries แบบนี้ว่า " non-blocking queries "
- การ queries ไม่ต้องรอให้ release data locks

Default Locking สำหรับการ INSERT, UPDATE และ DELETE STATEMENT

จะทำให้

- transaction ซึ่งมี DML statement ต้องการ exclusive row locks บน rows ที่ต้องการจะแก้ไข ดังนั้น transaction อื่นๆ ที่ต้องการแก้ไข rows เดียวกันจะต้องรอจนกว่า transaction ที่ hold locks นั้น released locks โดยการ commit หรือ roll back
- การ queries ใน transaction นั้นจะสามารถดูการเปลี่ยนแปลงที่เกิดจาก DML statement ที่แล้วภายใน transaction เดียวกันได้แต่ไม่สามารถดูการเปลี่ยนแปลงที่ยังไม่ commit ของ transaction อื่นได้
- ในขณะที่ transaction hold share, share row exclusive หรือ exclusive นั้นจะไม่ต้องการ exclusive row locks

Data locks Conversion และ Escalation

ORACLE สามารถ convert table locks ซึ่งมีความเข้มงวดน้อยกว่า -> เป็น locks แบบที่มีความเข้มงวดมากกว่า เช่น ถ้ามี transaction ที่ hold แบบ row share table locks หากต้องแก้ไขตาราง จริง ๆ (อาจด้วย statement " SELECT.. FOR UPDATE..")

ORACLE สามารถเปลี่ยนให้เป็น row exclusive table locks ได้เองโดยอัตโนมัติ

2) Dictionary locks (DDL locks)

จะ protect ส่วนที่เป็นคำนิยามของ statement object ในขณะที่ object นั้นอยู่

ภายใต้ DDL operations ใน transaction เช่น ถ้าผู้ใช้ create procedure แล้ว ผู้ใช้ ไม่จำเป็นต้องจัดการเกี่ยวกับการป้องกันเลย ORACLE จะจัดการ locks ให้เอง โดยจะ locks object ที่ใช้อ้างอิงทั้งหมดไม่ให้ถูกเปลี่ยนแปลงหรือ drop ทิ้งจนกว่า procedure นั้นจะ compile ใหม่เสร็จสมบูรณ์

หากมีการ locks เกิดขึ้นไม่ว่าแบบใด

DDL locks แบ่งเป็น 3 กรณีคือ - exclusive DDL locks

- share DDL locks
- breakable parse locks

2.1) Exclusive DDL locks

- สำหรับ resource ซึ่งต้องการป้องกันการแทรกแซงในระหว่าง DDL operation

เช่น ในระหว่าง ALTER TABLE จะไม่ทำ DROP TABLE operation

- หาก object ที่ถูก hold แบบ exclusive DDL locks แล้วมี transaction อื่นจะขอ hold ก็จะต้องรอจนกว่า exclusive DDL locks นั้นถูก released ก่อน

2.2) Share DDL locks

- สำหรับ resource ซึ่งต้องการป้องกันการแทรกแซง ในระหว่าง DDL operation ซึ่งอาจทำให้เกิด conflict DDL operations เช่นมี operation CREATE PROCEDURE จะทำให้เกิด share DDL locks และ transaction อื่นๆจะสามารถ CREATE PROCEDURE ที่ reference table เดียวกันได้

หมายเหตุ :- หากมีการ locks เกิดขึ้นไม่ว่าแบบใด transaction อื่นจะไม่สามารถ locks แบบ exclusive DDL locksได้ ทำให้มั่นใจได้ว่า definiton ของ object ที่ใช้ reference จะเป็น definition เดียวกัน ตลอดช่วงของ transaction นั้น

2.3) Breakable DDL locks

SQL statement (หรือ PL/SQL program unit) ที่ใช้ร่วมกันจะ hold parse locks เฉพาะ object ที่ถูกอ้างอิงโดย SQL statement ซึ่งเป็นผลทำให้จะไม่มี DDL operations ซึ่งขัดแย้งกันเลย

3) Internal locks and latches

จะปกป้องในส่วนภายในของฐานข้อมูลรวมทั้งโครงสร้างของหน่วยความจำในลักษณะเป็น file

4) Distributed locks

เพื่อให้มั่นใจว่า ข้อมูล และ resource ที่ กระจายอยู่ตาม server ต่างๆ ในระบบ ORACLE Parallel Server จะยังคงถูกต้องตรงกันอยู่

5) **Parallel cache management locks**

เป็น distributed locks ซึ่งครอบคลุม data blocks(ตาราง+index blocks) 1 blocks หรือมากกว่า ภายใน buffer ของ cache

3.3.2 Multi- version consistency model

ORACLE จะสร้างชุดของ ข้อมูล ในขณะที่กำลังทำ queries (read) พร้อมกับมีการแก้ไข (write) ดังนั้นเมื่อเกิดการแก้ไข จะมีการเก็บค่าเดิมของข้อมูลไว้ในฐานข้อมูล และหาก transaction นั้น ยังไม่ commit และมี user query, ORACLE จะ read จาก version ที่จะทำให้ consistency ที่สุด แต่เมื่อtransaction นั้น ยัง commit แล้วค่าของข้อมูล จะถูกเปลี่ยนแปลงอย่างถาวร ซึ่งจะเป็น 2 level คือ

1) Statement-Level Read Consistency

เพื่อ guarantee ว่า ข้อมูลหลังจากการทำ single query จะถูกต้องตรงกับก่อนทำ query

ดังนั้น Statement :- INSERT, DELETE, UPDATE จะต้องทำให้ข้อมูล ก่อนและหลังจากการ execute statement จะถูกต้องตรงกัน

2) Transaction - Level Read Consistency

เพื่อ guarantee ว่าในสภาวะ Multiple-queries ใน transaction ใด transaction หนึ่ง นั้นการ queries ทั้งหมดจะมองเห็น ข้อมูล ที่เป็นค่าตรงกันทุกการ queries ที่เวลาใดเวลาหนึ่ง

3.3.3 Deadlock

ORACLE จะจับว่ามี deadlock และเมื่อจับได้แล้วก็จะแก้ปัญหาโดยอัตโนมัติด้วยการ roll back transaction ที่เป็นเจ้าของ statement ที่ทำให้เกิด deadlock ทันทีเพื่อ released locks ซึ่งขัดแย้งกันอยู่

ORACLE จะจับ deadlock โดยมีวิธีต่างกันแบ่งตามประเภทของ deadlock

1) ใช้วิธี " waits for graph " เพื่อจับ deadlock แบบ local deadlocks

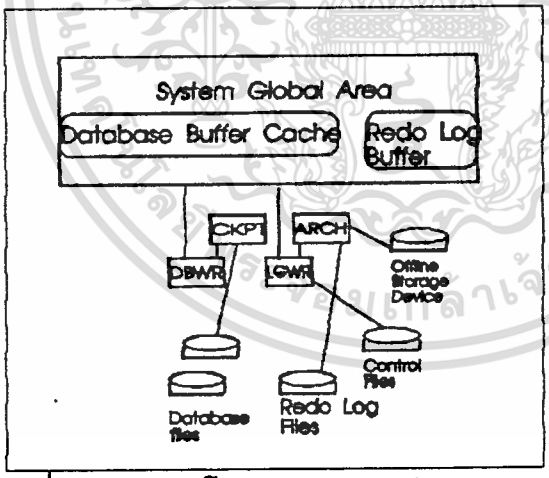
2) ใช้วิธี " time-out " เพื่อจับ deadlock แบบ global deadlocks

3.4 Backup และ Recovery ของ ORACLE

วิธีการ backup และ recovery ใน ORACLE นั้นจะใช้ Log-based Recovery โดยเมื่อผู้ใช้ส่ง commit statement มาแล้ว commit record ก็จะถูกนำลง Redo log buffer ทันทีแต่จะ defer การ write ข้อมูล ลง disk จนกว่าจะมีการเปลี่ยนแปลงใน data buffer มากพอแต่ถึงอย่างไรก็ตามรายละเอียดต่างๆในการเก็บ Log file และ Checkpoint และวิธีในการ recovery ของ ORACLE ก็จะแตกต่างกันกับ INGRES อยู่ไม่น้อย

นอกจากนี้ในการ commit ของ ORACLE ก็ยังคงมีลักษณะที่คล้ายกับ INGRES อีกก็คือ

- GROUP Commit ในเวลาที่มี activity มากๆ LGWR อาจจะ write ไปยัง Redo log file โดยใช้ Group commit เป็นการ save disk I/O คือรวม Redo Entry จากหลายๆผู้ใช้ แล้ว write ลง disk ทีเดียว
- TWO - PHASE Commit ก็คงใช้หลักการเดียวกันกับ INGRES คือ ในระบบ multiuser ถ้า site หนึ่งทำการcommit หรือ rollback อีก site หนึ่งก็ต้อง commit หรือ rollback ตามเช่นกัน



รูปที่ 3.3 แสดงโครงสร้างของหน่วยความจำเฉพาะส่วนที่ทำหน้าที่เกี่ยวกับ backup และ recovery

จากรูปจะแสดงถึงโครงสร้างของหน่วยความจำและ process ของ ORACLE แต่ในเรื่องการ recovery เราจะพิจารณาเพียงส่วนที่เกี่ยวข้องด้วยโดยตรงซึ่งก็ได้แก่

- System Global Area (SGA)
- Database file
- Redo log file

- Control file

และ Process ที่เกี่ยวข้องได้แก่

- DBWR (Database Writer)
- LGWR (Log Writer)
- CKPT (Checkpoint)
- ARCH (Archiving)

SGA (System Global Area) เป็นกลุ่มของ share memory structure ที่ถูก allocate โดย ORACLE ซึ่งจะบรรจุข้อมูล และ control information สำหรับ ORACLE database instance หนึ่งๆ และจะเป็น share memory สำหรับกรณี multiple user ด้วย

สิ่งที่เก็บใน SGA ได้แก่

- database buffer cache
- redo log buffer
- shared pool
- request และ response queues (ถ้าใช้ multithreaded server)
- data dictionary cache
- miscellaneous information อื่นๆ

LGWR (Log Writer) เป็นขบวนการที่ทำการwrite Redo Log Buffer ลง Redo Log File บน disk โดย LGWR process จะ write ทุกๆ Redo Entry (รายละเอียดอยู่ใน Online Redo Log File) ที่อยู่ใน buffer ตั้งแต่ครั้งสุดท้ายที่มัน write

สิ่งที่ LGWR write

- commit record เมื่อผู้ใช้ทำการ process commit transaction
- redo buffer ทุกๆ 3 วินาที
- redo buffer เมื่อ redo log buffer เต็มไป 1 ใน 3
- redo buffer เมื่อ DBWR ทำการ process write modified buffer ไปยัง disk

DBWR (Database Writer) เป็นขบวนการที่ทำการwrite buffer data ลงไปยัง data file ซึ่งจะถูกจัดการโดย buffer cache management เมื่อ buffer ใน buffer cache ถูกแก้ไขแล้วมันจะถูกทำให้เป็น "dirty"และจะถูก "clean" ได้ก็ต่อเมื่อมีการทำ DBWR process เพื่อ write dirty data ลง disk ในขณะที่ buffer ใน cache ถูก fill จากฐานข้อมูล และถูกทำให้ dirty โดย user process ก็จะทำให้จำนวนของ free buffer ลดลง ซึ่งถ้าลดลงมากจน process ของผู้ใช้ไม่สามารถหา buffer ที่ว่างได้ DBWR ก็จะ

จัดการ buffer cache ให้กับ process ของผู้ใช้ซึ่ง ORACLE จะจัดการโดยใช้ LRU Algorithm (Least Recently Used) เพื่อที่จะให้ Most Recently Used คือข้อมูลส่วนที่ถูกเรียกใช้บ่อยๆยังคงอยู่ในหน่วยความจำ

DBWR process จะถูก signal เพื่อ write dirty buffer ลง disk ภายใต้เงื่อนไขต่อไปนี้

- เมื่อถึงเวลา Checkpoint ซึ่ง LGWR จะทำการ signal DBWR
- เมื่อเกิด time out (ทุกๆ 3 วินาที) ซึ่ง DBWR จะ signal ตัวเองโดยจะทำการ search set ของ buffer ตาม LRU Algorithm
- เมื่อ server process พบว่ามีเนื้อที่ว่างใน buffer ไม่พอกับความต้องการ
- เมื่อ server process ทำการย้าย buffer ไปยัง dirty list และพบว่า dirty list นั้นยังไม่ถึงความยาวของ threshold ซึ่ง server process จะทำการ signal DBWR ให้ write

Redo Log Buffer เป็น circular buffer คือเมื่อ LGWR ทำการ write Redo Entry จาก redo log buffer ลงไปยัง redo log file แล้วนั้น server process ก็สามารถ copy entry ใหม่ลงไปทับ entry บน redo log buffer ที่ถูก write ลง disk ไปแล้วได้ โดยปกติ LGWR จะ write อย่างรวดเร็วเพียงพอจนสามารถแน่ใจได้ว่าเนื้อที่ว่างใน buffer จะมีเพียงพอสำหรับentry ใหม่ๆ ไม่ว่าจะมีการเข้าถึงมากเพียงใดก็ตามหรือถ้าเกิดกรณีว่าต้องการเนื้อที่ของ buffer มากกว่าที่มีอยู่ LGWR ก็จะ write redo log entry ก่อนที่ Transaction จะ commit ซึ่ง entry เหล่านี้จะ permanent ก็ต่อเมื่อ transaction นั้น commit ในเวลาต่อมา

Rollback Segment จะถูกใช้สำหรับฟังก์ชั่นจำนวนหนึ่งในการปฏิบัติงานของ ORACLE database หรือพูดง่ายๆก็คือ Rollback Segment ของฐานข้อมูลจะประกอบด้วย Rollback Entry ซึ่งจะมีเอาไว้เก็บ Old value ของข้อมูลโดยข้อมูลนั้นกำลังจะถูกเปลี่ยนแปลงโดย transaction ที่กำลังจะทำงานต่อไปเก็บเอาไว้ใน database space เนื่องจาก Rollback Entry ทำการเปลี่ยนแปลง block ของ data ดังนั้นมันก็จะมีการเก็บการเปลี่ยนแปลงเอาไว้ใน Redo Log file ด้วย ซึ่งในการเก็บครั้งที่ 2 นี้จะมีความสำคัญมากสำหรับ active transaction ที่ยังไม่ได้ commit ในเวลาที่เกิด systemcrash Information ต่างๆ ใน Rollback Segment นี้จะถูกใช้ระหว่างการทำ database recovery เพื่อทำundo ในการเปลี่ยนแปลงที่ uncommitted ดังนั้นถ้าจำเป็นต้องทำ database recovery แล้วหลังจากที่ใช้ Rollback Segment เพื่อทำการย้ายข้อมูลที่ยังไม่ commit ออกจาก data file แล้วฐานข้อมูลก็จะอยู่ในสภาวะที่ consistent

Redo Log File จะบันทึกการเปลี่ยนแปลงทั้งหมดที่ทำต่อฐานข้อมูล ซึ่งจะประกอบด้วย Redo Log File 2 อัน โดย Redo Log File นี้จะแยกออกจาก data file

Redo Log File 2 อัน ได้แก่

- Online Redo Log File
- Archieve Redo Log File

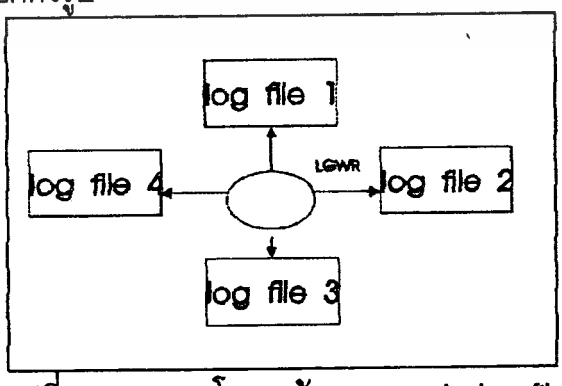
Online Redo Log File จะประกอบด้วย

1) Redo Entry ซึ่ง Redo Entry เหล่านี้จะบันทึกข้อมูลที่สามารถใช้ในการสร้างการเปลี่ยนแปลงทั้งหมดที่กระทำต่อฐานข้อมูลใหม่

2) Rollback Segment ดังนั้น Online Redo Log จะเป็นการป้องกันการ rollback ของข้อมูลด้วย Redo Entry จะเป็น low-level representation ของการเปลี่ยนแปลงในฐานข้อมูล ซึ่งไม่สามารถเขียนหรือทำการเปลี่ยนแปลงใดๆได้

Redo Entry จะถูก buffer ในรูปแบบ circular ใน Redo Log Buffer ของ SGA คือเมื่อ transaction commit แล้ว LGWR ก็จะ write Transaction Redo Entry จาก Redo Log Buffer ไปยัง Redo Log File และจะมีการให้ค่า sequence change number (SCN) ไปด้วยเพื่อชี้ Redo Entry สำหรับแต่ละ Committed transaction แต่อย่างไรก็ตาม Redo Entry ก็สามารถที่จะถูก write ลง Online Redo Log File ได้ก่อนที่ transaction จะ commit ในกรณีที่ Redo Log Buffer เต็ม

โครงสร้างของ Online Redo Log ของฐานข้อมูลจะประกอบด้วย Online Redo Log File มากกว่า 1 อัน และมีการทำงานแบบหมุนเวียนคือเมื่อ Online Redo Log File อันปัจจุบันถูกใส่ข้อมูลไป LGWR ก็จะเริ่ม write ไปยัง Online Redo Log File อันถัดไปและเมื่อเลื่อนไปถึง Redo Log File อันสุดท้าย LGWR ก็จะ return กลับมายัง Redo Log file อันแรกอีกดังรูป



รูปที่ 3.4 แสดงโครงสร้างของ redo log file

Online Redo Log File อันที่สามารถที่จะถูกใส่ข้อมูลได้จะขึ้นอยู่กับว่าการ Archieving ถูก enable หรือไม่

- ถ้า Archieving ถูก disable แล้ว Online Redo Log File ที่จะถูก fill ได้ต้องผ่านการ Checkpoint เพื่อให้การเปลี่ยนแปลงที่อยู่ใน Log File ก่อนหน้านี้ถูก write ลง data files ไปแล้วถึงจะยอมให้มีการใช้ Online Redo Log File อันนั้นใหม่ได้

- ถ้า Archieving ถูก enable แล้ว Online Redo Log File ที่จะถูก fill ได้ต้องผ่านการ Checkpoint แล้วก็ต้องหลังจาก file ใน Online Redo Log File ถูก Archieve แล้วด้วย

Checkpoint เป็น event ที่ DBWR ทำการ write database buffer ที่ถูกเปลี่ยนแปลงทั้งหมดลงใน SGA ทั้ง committed และ uncommitted data ลงไปยัง data file

เหตุผลที่ต้องมี Checkpoint

1) Checkpoint จะทำให้แน่ใจว่า data segment block ใน memory ที่มีการเปลี่ยนแปลงบ่อยๆถูก write ลง data file เป็นระยะๆ เนื่องจากการใช้ LRU Algorithm ในการทำ DBWR จะทำให้ data segment block ที่มีการเปลี่ยนแปลงบ่อยๆจะไม่ถูก write ลง disk เลยจึงทำให้จำเป็นต้องมี Checkpoint

2) เนื่องจากการเปลี่ยนแปลงของฐานข้อมูล ทั้งหมดจนกระทั่งถึง Checkpoint จะถูกบันทึกลงใน data files ทำให้ Redo Log Entry ก่อนหน้า Checkpoint ก็ไม่มีความจำเป็นต้องใช้อีก
ในกรณีที่เกิดจะต้องทำ recovery ดังนั้น Checkpoint จึงมีประโยชน์เพราะว่าจะช่วยให้ recovery เร็วขึ้น

Archieved Redo Log File จะเป็น optional ของ ORACLE ที่อนุญาตให้มีการรวมกลุ่มของ Online Redo Log file แล้วเก็บลงใน Archieved(Offline) Redo Log จะเป็นการเก็บ Redo Log Information อันเก่าๆ จาก Online Redo Log File สำหรับช่วยเพิ่มประสิทธิภาพในการทำ database recovery เพราะในกรณีที่เนื้อที่ที่ allocate ไว้สำหรับเก็บ Online Redo Log File เต็มและต้องการที่จะนำเนื้อที่นั้นมาใช้เก็บการเปลี่ยนแปลงของฐานข้อมูลใหม่ๆ ดังนั้นจึงต้องมีการถ่าย Online Redo Log File เก่ามาเก็บไว้ใน Archieved Redo Log File

การทำ Archieve จะมีประโยชน์ต่อการ backup และ recovery ดังนี้

1) เป็นการทำdatabase backup คือทั้ง Online และ Archieved Redo Log File จะสามารถ guarantee ได้ว่า transaction ที่ commit ไปแล้วทั้งหมดจะสามารถ recover กลับมาได้ในกรณีที่เกิด failure

2) เป็นการทำOnline backup คือเก็บรายละเอียดการเปิดฐานข้อมูล

ต่างๆและรายละเอียดในการใช้งานระบบได้อย่างถาวรถ้าไม่มีการลบ Archieved Redo Log File นี้ทิ้ง

ใน Archieved Redo Log File จะเก็บ copy กลุ่มของ Online Redo Log File และเก็บ Log Sequence Number (SCN) ของกลุ่มไว้ด้วย

ORACLE จะมี mode ให้เลือกว่าต้องการที่จะ ทำNOARCHIEVELOG mode หรือทำ ARCHIEVELOG mode ซึ่งก็ขึ้นอยู่กับกลไกของการ backup และ recovery คือ

- NOARCHIEVELOG mode คือการ disable การทำ Archieved Redo Log File หรืออาจจะพูดได้ว่าเป็น Media Recovery Disabled เนื่องจากจะปกป้องฐานข้อมูลได้เพียงแค่ Instance failure (failure จากไฟดับหรือ operating system crash) ไม่สามารถ protect กรณี Disk (media) failure เช่น disk head crash ได้เพราะในกรณีนี้ข้อมูลใน disk จะหายไปหมดและเมื่อไม่มีการเก็บ Redo Log File อันเก่าๆไว้ก็ไม่สามารถที่จะ recover กลับมาได้

- ARCHIEVELOG mode คือการ enable การทำ Archieved Redo Log File หรือเป็น Media Recovery Enabled จะทำให้สามารถ protect ได้ทั้ง Instance failure และ Disk failure ได้ด้วย

ถ้า enable การ Archieving แล้ว LGWR จะไม่สามารถใช้ Online Redo Log Group นั้นได้จนกว่า Log Group นั้นจะถูก Archieve ไปก่อน ดังนั้นจึง guarantee ว่า Archieved
Redo Log File จะเก็บ copy ของทุกๆกลุ่มตั้งแต่มีการ enable

Control file จะเป็น binary file ซึ่งมีความจำเป็นสำหรับฐานข้อมูล เพื่อที่จะให้เกิดการ start และ operate ที่ successfully. Control File จะถูกแก้ไขอย่างต่อเนื่องโดย ORACLE ระหว่างการใช้งานฐานข้อมูลและแต่ละ Control File จะเกี่ยวข้องกับฐานข้อมูลอันเดียวเท่านั้น

สิ่งที่อยู่ใน Control File ก็คือ information เกี่ยวกับฐานข้อมูลที่เกี่ยวข้องและจะถูก modify โดย ORACLE เท่านั้น DBA ไม่สามารถที่จะเข้าไป edit ได้

Control File จะประกอบด้วยรายละเอียดต่างๆเช่น

- ชื่อของฐานข้อมูล
- Timestamp ของ database creation
- ชื่อ และ ที่ตั้งของฐานข้อมูล และ Online Redo Log File ที่เกี่ยวข้อง
- Current Log Sequence Number
- Checkpoint Information

นอกจากนี้ ORACLE ยังยอมให้ open Control File ที่เหมือนๆกันได้หลายๆอันสำหรับฐานข้อมูลอันเดียวกันเรียกว่า Mirrored Control File โดยการเก็บเอาไว้หลายๆ disk เพื่อเป็นการ backup control file เอาไว้สำหรับใช้ในการ restart กรณีเกิด failure โดยไม่ต้องทำ recovery

ขั้นตอนในการทำ Recovery

ORACLE จะทำขบวนการ Recovery 2 step คือ

Step 1 Rolling Forward จาก Redo Log

Step 2 Rolling Back จาก Rollback Segment

Redo Log and Rolling Forward

การ Roll forward จะเป็นการ reapply การเปลี่ยนแปลงทั้งหมดที่ได้เก็บไว้ใน Redo Log ไปยัง Data File และเนื่องจากข้อมูลในการ Rollback ก็ถูกเก็บเอาไว้ใน Redo Log ด้วย ดังนั้น Roll forward ต้องไปเรียกใช้ Rollback Segment ด้วยซึ่งเป็น step ของการ Rollback

Rolling forward จะทำการจัดการกับ Redo Log File จำนวนมากเท่าที่จำเป็นเพื่อที่จะนำฐานข้อมูลกลับคืนมา โดยจะใช้ทั้ง Online Redo Log File และ Archieve Redo Log File หลังจาก Roll forward แล้วภายในฐานข้อมูลก็จะมีทั้งการเปลี่ยนแปลงที่ commit แล้วทั้งหมดและการเปลี่ยนแปลงที่ยังไม่ commit ที่ถูกเก็บอยู่ใน Redo Log

Rollback Segment and Rolling Back

Rollback Segment จะถูกใช้เพื่อที่จะ undo uncommitted transaction ที่ถูก apply มาก่อนหน้านี้โดย Rolling Forward phase

ใน Roll forward นั้นการเปลี่ยนแปลงใดๆที่ยังไม่ commit จะต้องถูก undo หลังจากที่ Redo Log File ได้ทำการ reapply การเปลี่ยนแปลงที่ทำต่อฐานข้อมูลทั้งหมดแล้วก็จะต้องมีการใช้ rollback segment ด้วยเพื่อที่จะ undo transaction ที่ยังไม่ได้ commit เนื่องจาก transaction ที่ยังไม่ commit นั้นการเปลี่ยนแปลงทั้งหลายก็ยังอยู่ใน Redo log buffer ยังไม่ได้ write ลง Redo Log File ดังนั้นถ้าเกิด failure ตรงนี้ก็จะต้องดึงเอาข้อมูลใน Rollback Segment มาใช้ซึ่ง process นี้เรียกว่า Rolling back

นั้นสามารถใช้ Integrity Constraint เดียวกันได้เลยหากใช้ตารางเดียวกันโดยแอพพลิเคชั่นที่เกี่ยวกับฐานข้อมูลต่าง ๆ ไม่ต้องเขียนส่วนนี้ขึ้นเอง

- Immediate User Feedback

ก่อนที่จะเรียกใช้คำสั่ง SQL จะมีการตรวจสอบความถูกต้องของข้อมูลก่อนโดยใช้ Integrity Constraint ที่เก็บไว้ใน data dictionary

Integrity Constraint สามารถแบ่งได้เป็น 2 แบบคือ

1) Entity Integrity ซึ่งแบ่งเป็น

- DEFAULT Integrity Constraint :- สำหรับคอลัมน์ที่ไม่ได้กำหนดระหว่างที่ insert
- NOT NULL Integrity Constraint :- เป็น Integrity Constraint ซึ่งใช้กำหนดให้คอลัมน์ที่ต้องการในตารางจะต้องมีค่าไม่เป็น ค่า NULL
- UNIQUE Integrity Constraint :- เป็น Integrity Constraint ซึ่งใช้กำหนดเมื่อไม่ต้องการให้ rows 2 rowsใด ๆ ในตารางมีค่าเดียวกันใน คอลัมน์หรือชุดของคอลัมน์ที่เราระบุ
- CHECK Integrity Constraint :- เป็น Integrity Constraint ซึ่งกำหนดคอลัมน์หรือชุดของคอลัมน์ที่ต้องการให้เป็นไปตามเงื่อนไขที่กำหนดโดยตรวจสอบทุก ๆ rows ในตารางและถ้าพบว่าเงื่อนไขนั้นเป็นเท็จ จะระงับ statement นั้นและข้อมูลที่เราจะเปลี่ยนแปลงก็จะถูกงดด้วย

2) Referential Integrity ซึ่งแบ่งเป็น

- PRIMARY KEY Integrity Constraint
  :- แต่ละตารางในฐานข้อมูลจะมี constraint นี้ได้เพียง constraint เดียว
  :- rows ในตารางจะถูกกำหนดระบุและอ้างอิงโดย primary key ซึ่งต้องมีความถูกต้องตาม constraint นี้
- FORIEGN KEY Integrity Constraint
  :- เป็น Integrity Constraint ที่ทำหน้าที่รักษาความสัมพันธ์ระหว่างค่าของ Primary key ใดๆจะต้องมีค่า match กับค่าของ Foriegn key ในอีกตารางหนึ่ง

ซึ่งการ match ก็สามารถแบ่งได้เป็น - match full
- match partial
- match null

- Actions ซึ่ง กำหนด โดย Referential Integrity Constraint

แบ่งเป็น - Update และ Delete Restrict

หมายความว่าค่าของ key ที่เป็นตัวอ้างอิงให้กับ key อื่น ๆ จะถูกเปลี่ยนแปลง หรือ ลบไม่ได้ เช่น primary key จะถูกเปลี่ยนแปลงหรือลบไม่ได้เพราะยังมีค่านั้นปรากฏอยู่เป็น foriegn key ในตารางอื่นๆ

- Delete Cascade

หมายความว่า ถ้า rows ที่มี key ที่เป็นตัวอ้างอิงให้กับ key อื่น ๆ ถูกลบแล้ว ทุก rowsที่มี foriegn key ที่อ้างอิงจาก key นั้นจะต้องถูกลบไปด้วย

3.5.2 Database Trigger

ORACLE ยอมให้ DBA และผู้ใช้สามารถควบคุม integrity ได้โดยไม่จำเป็นต้องประกาศแบบที่เรียกว่า " non-declarative " โดยใช้ database trigger

database trigger หมายถึง database procedure ที่ใช้ควบคุม integrity ซึ่งเก็บไว้และสามารถเรียกใช้ในการ insert,delete หรือ update operations

database trigger นั้นถูกสร้างโดย DBA และผู้ใช้และหากผู้ใช้เป็นผู้ create ก็ควรอยู่ภายใต้ความควบคุมของ DBA

เหตุการณ์ที่ควรใช้ database trigger

- ใช้ referential integrity หากไม่ใช้ Integrity Constraint
  (database trigger ต่างจาก Integrity Constraint ตรงที่ database trigger จะใช้ integrity rules โดยไม่ตรวจสอบข้อมูล load ตาราง ในขณะที่ Integrity Constraint จะมีการ check ก่อน ดังนั้นควรจะใช้ database trigger ก็ต่อเมื่อไม่สามารถใช้ Integrity Constraint ควบคุม)
- ใช้ referential integrity เมื่อ child และ parent ตาราง อยู่ที่ต่าง node กันของระบบ distributed database
- ใช้ business rules ที่ยุ่งยากซับซ้อนและไม่สามารถใช้ Integrity

Constraint

หมายเหตุ :- หาก business rules ที่ควบคุมโดย Integrity Constraint หรือ database trigger เปลี่ยนแปลง DBA ก็สามารถเปลี่ยนแปลงข้อกำหนดของ constraint นั้นที่เก็บไว้ที่ data dictionary ได้เลยดังนั้น แอพพลิเคชั่นที่มาใช้ก็จะรับรู้การเปลี่ยนแปลงนั้นโดยไม่จำเป็นต้องแก้ไข code แต่ถ้าควบคุมโดย code ของแต่ละแอพพลิเคชั่นจะต้องมีการเปลี่ยนแปลง code ของแต่ละแอพพลิเคชั่นซึ่งนับว่ายุ่งยากและเป็นวิธีที่ไม่มีประสิทธิภาพเพียงพอ

3.6 Query Optimization ของ ORACLE

Optimization ของ ORACLE ก็มีจุดมุ่งหมายเหมือนกับใน INGRES ก็คือ Optimizer จะทำการพิจารณา factor ต่างๆ เพื่อหาทางที่มีประสิทธิภาพที่สุดในการ execute SQL Statement โดยใน execution plan สำหรับSQL Statement นั้น Optimizer จะใช้ 2 approach โดย ORACLE มี mode ให้เลือกว่าจะเอาทางใด

Two Approach ของ Optimization ได้แก่

1) Rule-based Approach
2) Cost-based Approach

Rule-based Approach Optimizer จะเลือก execution plan โดยดูจาก Access Path และพิจารณา heuristically ranked operation โดยถ้ามีทาง execution SQL Statement มากกว่า 1 ทางก็จะใช้ operation ของ rank อันดับต่ำว่าซึ่งในกรณีส่วนใหญ่แล้ว operation ของ rank ที่ต่ำกว่าจะ execute ได้เร็วกว่า

ตารางแสดง Access Path

Rank	Access Path
1	Single row by ROWID
2	Single row by cluster join
3	Single row by hash cluster key with unique or primary key
4	Single row by unique or primary key
5	Cluster join
6	Hash cluster key
7	Indexed cluster key
8	Composite index
9	Single-column index
10	Bounded range search on indexed columns
11	Unbounded range search on indexed columns
12	Sort-merge join
13	MAX of MIN of indexed column
14	ORDER BY on index column
15	Full scan table

สำหรับSQL Statement ที่เข้าถึง view นั้น Optimizer มักจะรวม query ของ view เข้าไปใน original statement หรือ original statement เข้าไปใน query ของ view และ Optimize ผลลัพธ์ในการรวม query ของ view เข้าไปใน accessing statement นั้น Optimizer จะแทนชื่อของ view ด้วยชื่อของ base ตารางใน accessing statement และเพิ่ม condition ของ WHERE ใน query จาก view เข้าไปใน acessing statement

4) Choice of optimization approaches

Optimizer จะเลือกว่าจะใช้ Rule-based หรือ Cost-based เพื่อทำการ Optimization

ในการทำงานของOptimizer ในการเลือก approach และ จุดมุ่งหมายจะขึ้นอยู่กับ factor ดังนี้

- OPTIMIZER_MODE initialization parameter
- Statistics ใน data dictionary
- OPTIMIZER_GOAL parameter of the ALTER session command
- Hints in the statement

OPTIMIZER_MODE initialization parameter มี 2 อย่างคือ

1) COST ค่านี้จะทำให้ Optimizer เลือกระหว่าง Rule-based กับ Cost-based คือ ถ้า data dictionary ประกอบด้วย statistic ของตาราง อย่างน้อยที่สุด 1 ตารางแล้ว Optimizer ก็จะใช้ Cost-based ด้วย จุดมุ่งหมาย คือ best throughput และถ้าใน data dictionary ไม่มี statistics ของตารางใดๆเลย Optimizer ก็จะใช้ Rule-based ซึ่งตาม default value ของ parameter นี้จะใช้ COST

2) RULE ค่านี้จะทำให้Optimizer เลือก Rule-based สำหรับทุกๆSQL statement โดยไม่คำนึงถึง statistics เลยว่ามีหรือไม่

OPTIMIZER_GOAL parameter of the ALTER session command

parameter นี้จะทำการล้างOptimization approach และจุดมุ่งหมายใน OPTIMIZER_MODE สำหรับใน SESSION หนึ่งๆ โดยมี 4 อย่างคือ

1) CHOOSE ค่านี้จะทำให้Optimizer เลือกระหว่าง Rule-based และ Cost-based เหมือนกับ COST ใน OPTIMIZER_MODE

2) ALL_ROWS ค่านี้จะทำให้ Optimizer ใช้ Cost-basedสำหรับSQL Statement ใน session ด้วยจุดมุ่งหมาย คือ throughput โดยไม่คำนึงถึงว่ามี statistics หรือไม่

3) FIRST_ROWS ค่านี้จะทำให้Optimizer ใช้ Cost-based สำหรับSQL Statement ใน session ด้วยจุดมุ่งหมาย คือ response time โดยไม่คำนึงถึงว่ามี statistics หรือไม่

4) RULE ค่านี้เหมือนกับ RULE ใน OPTIMIZER_MODE

ค่าเหล่านี้จะมีผลกับการ Optimization ของ SQL Statement ใน Stored procedures และ functions called ในระหว่าง SESSION แต่จะไม่มีผลกับการ Optimization ของ recursive SQL Statement จาก ORACLE ระหว่าง SESSION

5) Choice of access paths

สำหรับแต่ละตารางที่ถูกเข้าถึงโดยstatement นั้น Optimizer จะเลือก access path เพื่อให้ได้ข้อมูลจากตารางโดย ORACLE จะพิจารณาที่ access method และ access path

Access method ได้แก่

1) Full table scan

2) Table access by ROWID

3) Cluster scans

4) Hash scans

5) Index scans

Access path ดูได้ที่ตาราง access path ในเรื่อง Rule-based Approach ก่อนหน้านี้

6) Choice of join orders

สำหรับjoin statement ที่ join table มากกว่า 2 ตารางนั้น Optimizer จะเลือกจับคู่ของตารางที่จะถูก join ก่อนและหลังจากนั้นก็เลือกคู่ join จนได้ผลลัพธ์

7) Choice of join operations

สำหรับ join statement ใดๆ Optimizer จะเลือก operation ที่จะใช้ในการ perform การ join นั้นโดยมี operation ให้เลือกคือ

- nested loops
- sort-merge
- cluster

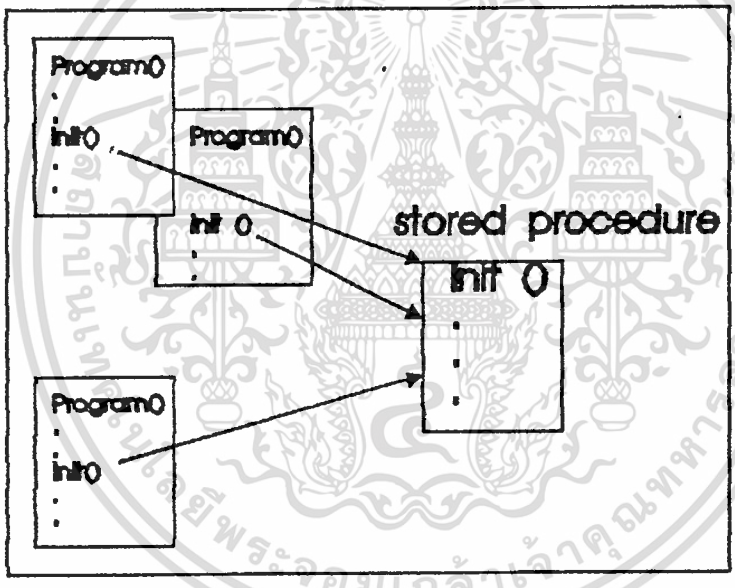
เนื่องจากใน INGRES มี Compiled Database Procedure ซึ่งสิ่งนี้ ORACLE ก็มีเช่นเดียวกันแต่ใช้ชื่อว่า Stored Procedure

Stored Procedure and Function

เป็น object ที่เป็นการ group set ของ SQL และ PL/SQL programming language statement อื่นๆเข้าด้วยกันในทาง logical เพื่อแสดง task เฉพาะขึ้นมา และเก็บ code ที่ compile แล้วนี้ไว้ใน share pool ของ SGA และ compiled version นี้ก็จะอยู่ใน shared pool ตาม LRU algorithm

Procedure และ Function นี้จะถูกสร้างขึ้นใน scheme ของ ผู้ใช้และเก็บมันไว้ในฐานข้อมูลสำหรับใช้งานต่อไปโดยมันจะถูก execute อย่าง interactive โดยใช้ ORACLE tool เช่น SQL*DBA หรืออาจถูกเรียกโดยตรงจาก code ของ database application เช่น SQL*Forms, Precompiler application, procedure อื่นๆ หรือ trigger

จากรูปจะแสดงถึง simple procedure ที่ถูกเก็บในฐานข้อมูลซึ่งจะถูกเรียกโดยหลาย application



รูปที่ 3.5 รูปแสดงหลักการทำงานของ store procedure

นอกจากนี้ procedures, functions, cursors, variables ที่สัมพันธ์กันยังสามารถนำมารวมเข้าด้วยกันได้รวมเป็นหน่วยหนึ่งเรียกว่าpackage และเก็บมันไว้ในฐานข้อมูลเพื่อการใช้งานต่อไป

ประโยชน์ของ Stored Procedure

1) ทางด้าน secerity เนื่องจาก Stored procedure สามารถช่วยควบคุมความปลอดภัยของข้อมูลดังนั้นจึงสามารถจำกัด operation ของฐานข้อมูลสำหรับการเข้าถึงข้อมูลของแต่ละผู้ใช้ โดยผ่านการใช้ Procedure และ Function เพราะในการเรียกใช้ procedure นี้ ORACLE จะมีการตรวจสอบสิทธิ (privilege) ในการใช้ข้อมูลนี้

2) Stored Procedure สามารถเพิ่มประสิทธิภาพของฐานข้อมูลเนื่องจากสามารถลดจำนวนของข้อมูลที่ต้องส่งข้ามเครือข่ายเมื่อเทียบกับการส่ง SQL Statement เข้าไป และก็ยังไม่ต้องมีการ compile ซ้ำอีกในการexecute code นอกจากนี้ถ้าเกิดว่า procedure มีอยู่แล้วใน shared pool ของ SGA ก็ไม่จำเป็นต้องเรียกขึ้นมาจากดิสก์ สามารถ execute ได้ทันที

3) ประหยัดหน่วยความจำเนื่องจาก Stored procedure ใช้ประโยชน์ของ shared memory ดังนั้นก็ต้องการเพียงแค่ copy เดียวนำเข้ามาในหน่วยความจำสำหรับการ execute โดย multiple users

4) Stored Procedure จะช่วยเพิ่มประสิทธิผลเนื่องจากในการออกแบบแอปพลิเคชั่นโดยใช้ชุดของ procedures ทำให้สามารถที่จะเลี่ยงความซ้ำซ้อน เช่น procedure จะถูกเขียนขึ้นมาเพื่อการแก้ไข แทรก ลบ row ออกจากตารางซึ่งแอปพลิเคชั่นใดๆ ก็สามารถเรียกใช้ procedure เหล่านี้ได้โดยไม่ต้องเขียน statement ใหม่และถ้าเกิดว่าวิธีของการจัดการข้อมูลเปลี่ยนไปก็จะต้องการเพียงแค่แก้ไข procedure ใหม่เท่านั้นไม่ต้องแก้ไขทุกๆแอปพลิเคชั่นที่ใช้ procedure นั้น

5) Stored Procedure จะช่วยเพิ่ม integrity และ consistency ของแอปพลิเคชั่นให้ถูกต้องแน่นอนขึ้นคือแต่ละ procedure หรือฟังก์ชั่นต้องการการทดสอบเพียงครั้งเดียวเพื่อที่จะรับประกันว่าได้ผลลัพธ์ถูกต้อง หลังจากนั้นแอปพลิเคชั่นใดๆก็สามารถเรียกใช้ procedure นี้ได้ถ้าเกิดการเปลี่ยนแปลงของโครงสร้างข้อมูลที่ procedure นั้นอ้างถึงก็จะมีเพียง procedure นั้นที่จะต้อง compile ใหม่ ส่วนแอปพลิเคชั่นที่เรียกใช้ไม่เกี่ยว

3.7 Cooperative Development Environment (CDE) (ORACLE Tools)

Coorperative Development Environment (CDE) เป็นเซตรวมเครื่องมือพัฒนาระบบงานคอมพิวเตอร์ ซึ่งจะครอบคลุมทุกขั้นตอนของวงจรการพัฒนาแอปพลิเคชั่นเริ่มตั้งแต่การ re-engineer กระบวนการทำงาน (business process) และคำจำกัดความข้อกำหนดของโปรแกรมใช้งาน (application requirement definition) รวมถึงขั้นตอนการวิเคราะห์ออกแบบพัฒนางานและแก้ไขเปลี่ยนแปลง ใช้ได้ตั้งแต่ single-user จนกระทั่งถึง enterprise-wide
system

CDE ประกอบด้วย เครื่องมือ 3 ประเภท ได้แก่

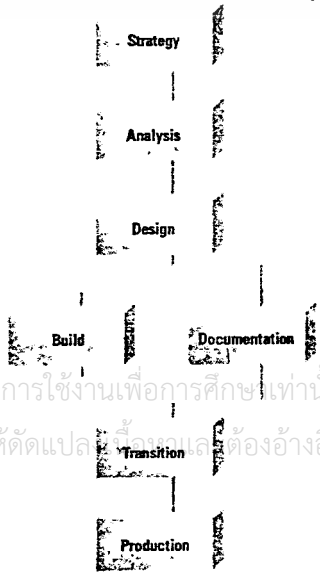
1) CASE
2) Application Development
3) Office Automation and End User

3.7.1 CASE Tools

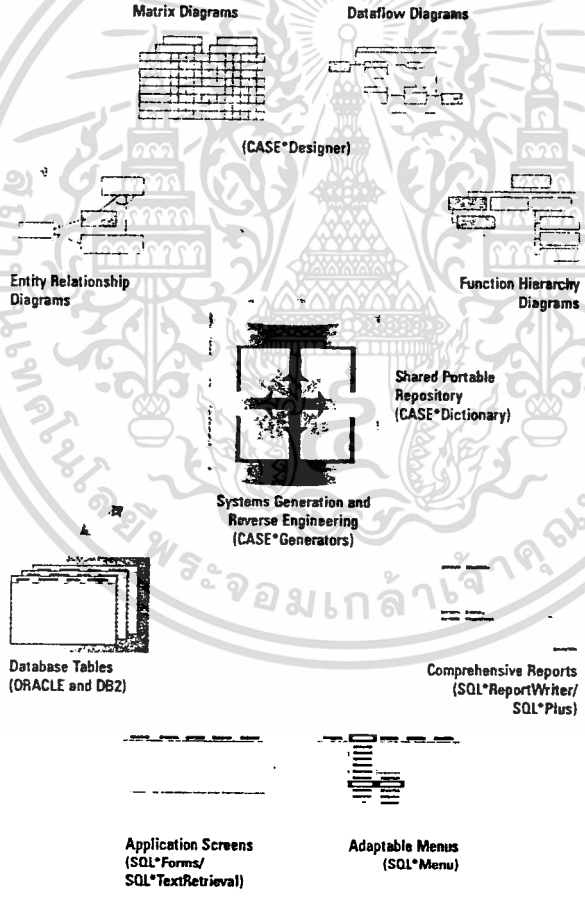
ORACLE มี CASE family ซึ่งจะจัดให้ทั้งเครื่องมือและวิธีการเพื่อใช้ในระบบตลอดวงจรการพัฒนาคือตั้งแต่การกำหนดกลยุทธ ไปจนถึง การได้ผลิตผล
ORACLE/CASE ประกอบด้วย

- CASE method

ซึ่งเป็นวิธีการพัฒนาระบบแบบ Top-down ซึ่งจะระบุขั้นตอนสำคัญๆเพื่อเป็น help guide ในการสร้าง set ของงานเพื่อให้บรรลุจุดมุ่งหมายและผลที่ต้องการ



- CASE Dictionary (Shared Portable Repository)
เป็นที่เก็บข้อมูลสำหรับทุกๆขั้นตอนและข้อมูลเหล่านั้นจะสามารถถูกควบคุมได้โดยง่ายผ่านทาง Forms interface หรือเครื่องมือทางกราฟฟิค CASE Dictionary มี automatic documentation ให้ นั่นคือมีทางเลือกถึง 3 ทาง คือ predefined reports, impact analysis reports และ custom report โดยใช้ SQL Reportwriter สำหรับการสร้างเอกสารของระบบทำให้ผู้พัฒนาสามารถสร้างรายงานเองได้อย่างรวดเร็วและ CASE Dictionary นี้สามารถที่จะรันได้บนหลาย platform ด้วย

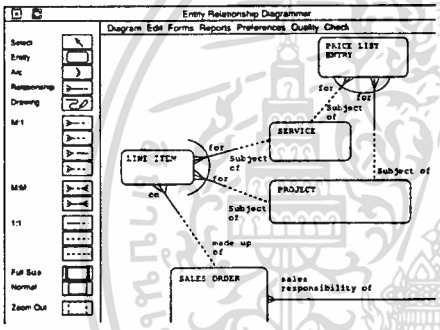


- CASE Designer
เป็นเครื่องมือที่ใช้เพื่อการกำหนดความต้องการของผู้ใช้และระบบ โดยใช้ graphic model ในขณะที่ผู้ใช้วาดรูปลงใน diagrammer ตัวใดตัวหนึ่ง CASE Designer จะทำการแก้ไข

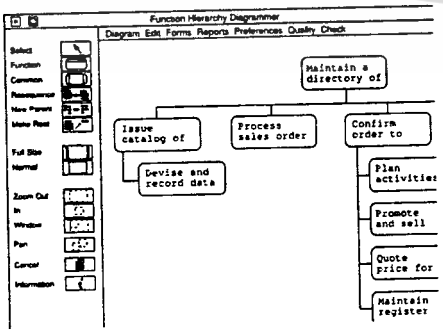
CASE Dictionary ในทันทีโดยไม่มีการสูญเสีย concurrency CASE DEsigner จะสนับสนุนระบบ multiuser คือผู้ใช้หลายๆคนสามารถที่จะเข้าถึง CASE Designer ได้ในเวลาเดียวกันแม้จะเป็นการทำงานบน models ที่เหลื่อมซ้อนกันก็ตามและผู้ใช้ แต่ละคนสามารถที่จะเปิด windows ได้โดยไม่จำกัดเพื่อดูหลายๆพื้นที่ของ model ที่ทำอยู่โดยแต่ละ window จะ executes separate task นอกจากนี้ CASE Designer ยังจะสามารถเคลื่อนย้ายไปรันบนสภาพแวดล้อมทางกราฟฟิคต่างๆได้เช่น X window, DEC window และ Presentation manager เป็นต้น

Diagrammers ของ CASE Designer ประกอบด้วย

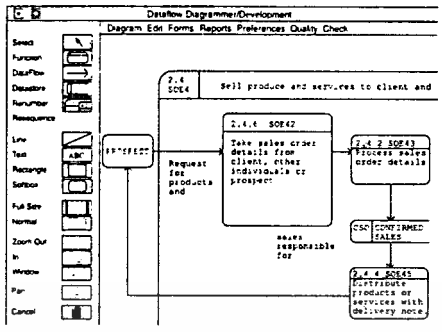
1) Entity Relationship Diagrammer (ER)



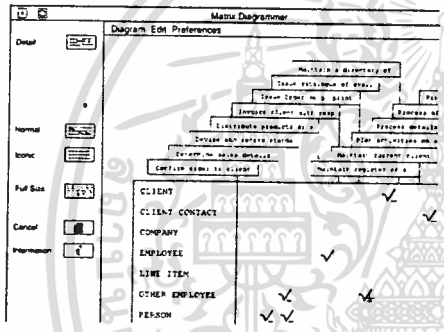
2) Function Hierarchy Diagrammer



3) Dataflow Diagrammer



4) Matrix Diagrammer



- CASE Generator Series

จะใช้เพื่อการสร้าง code สำหรับแอปพลิเคชั่นโดยใช้ข้อมูลที่เก็บอยู่ใน CASE Dictionary มีทั้งสิ้น 4 ตัว

1) CASE Generator for SQL Forms

ใช้ในการสร้างแอปพลิเคชั่นจาก diagram สำหรับ SQL Forms และ แอปพลิเคชั่นที่ถูกสร้างขึ้นนี้จะสามารถรันได้บนทุกๆ platform ที่ ORACLE สนับสนุน output ของ generator ตัวนี้เป็น functional interactive application ที่จะรวมเอา SQL statement ต่างๆเช่น enter, edit, retrieve ข้อมูลจากฐานข้อมูลไว้ด้วยและยังรวม logic ต่างๆที่ใช้ในการบังคับกฎทางธุรกิจที่ถูกกำหนดไว้ใน CASE Dictionary ทำให้เราไม่ต้องเขียน code ซ้ำซากในส่วนของการควบคุมหน้าจอ, การตรวจสอบข้อมูล และ integrity constraint ซึ่งจะถูกบังคับใช้โดยอัตโนมัติและโดยการใช้ SQL Form designer ประกอบทำให้เราสามารถที่จะ override default เช่น เปลี่ยนรูปหน้าจอ เพิ่มความสามารถอื่นๆเข้าในแอปพลิเคชั่นโดยเขียน routines เพิ่มเข้าไป แอปพลิเคชั่นที่ถูกสร้างขึ้นจะ compatible กับ block-mode, character-mode และ bitmap devices

2) CASE Generator for SQL Forms/SQL Menu

CASE Generator สามารถที่จะสร้างทั้ง form layout และ menu layout และการเข้าถึง ฐานข้อมูลได้โดยอัตโนมัติและแอปพลิเคชั่นที่สร้างสามารถจะ regenerate ได้ทุกๆ เวลาที่ความต้องการของผู้ใช้เปลี่ยนไปในการสร้างแอปพลิเคชั่น CASE Generator จะมีการสร้างระบบเมนูเต็มรูปแบบ ให้ด้วยคือมีทั้ง menu item help, menu nevigation, menu acces to form, other menus และมีหลายลักษณะของเมนูให้เช่น full screen, pull down เป็นต้น

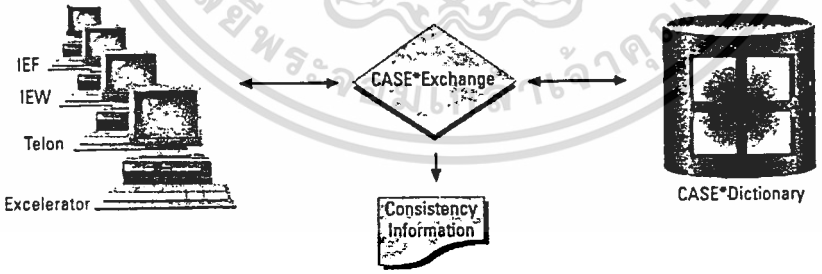
3) CASE Generator Database Tables

ซึ่ง CASE Generator ตัวนี้จะทำการออกแบบรูปแบบของตารางให้กับผู้ใช้โดยจะ map มาจาก diagram ที่ผู้ใช้สร้างขึ้นมา

4) CASE Generator for SQL ReportWriter/SQL plus

เป็น reporting tool ที่ใช้ร่วมกับ CASE Dictionary เพื่อสร้าง layout และ report ตามที่ผู้ใช้ต้องการโดย CASE Generator จะสร้างแอปพลิเคชั่นสำหรับออกรายงานจากรายละเอียดต่างๆที่ถูกเก็บใน CASE Dictionary

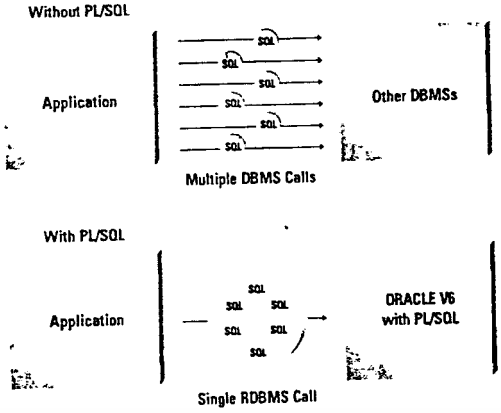
- CASE Exchange เป็น facility ที่ทำให้ข้อมูลสามารถที่จะถูกส่งผ่านระหว่าง non-Oracle CASE tools และ CASE Dictionary ได้



3.7.2 Application Development Tools

1) PL/SQL

เป็น procedural language ที่ขยายมาจาก SQL ทำให้เราสามารถที่จะ execute multistatement หรือ multitransaction PL/SQL procedure โดยใช้คำสั่งเดียว ไปยัง ORACLE จึงทำให้ลดการติดต่อสื่อสารบนเครือข่ายแบบ client/server



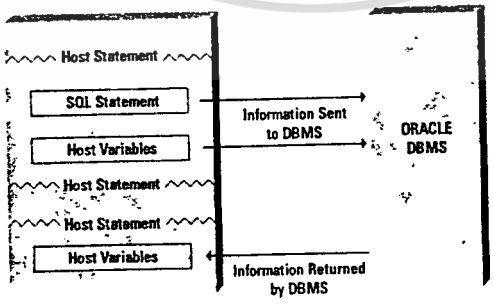
You can execute a multistatement or multitransaction PL/SQL procedure with a single request to ORACLE.

และ PL/SQL สามารถที่จะ execute การคำนวนของข้อมูล และ form control logic ได้เองโดยไม่ต้องส่งคำสั่งไปที่ฐานข้อมูลซึ่งเป็นการเพิ่มประสิทธิภาพของแอปพลิเคชั่นเรา สามารถใช้ PL/SQL เขียนแอปพลิเคชั่นร่วมกับ ORACLE tools อื่นๆได้

ใน PL/SQL procedure สามารถที่จะมี SQL statement มาตรฐาน และ conditional control statement ได้และเมื่อ PL/SQL procedure ถูกกำหนดขึ้นมามันจะสามารถถูกเรียกใช้จาก form อื่นๆด้วย PL/SQL แอปพลิเคชั่นจะสามารถรันได้บนทุกๆ platform ที่รัน ORACLE ด้วย

2) Precompilers

จะอนุญาติให้ฝัง SQL ลงในโปรแกรมหลักที่เขียนด้วย Ada, C, Cobol, Fortran, Pascal PI/1 ได้และสามารถจะควบคุมการติดต่อกับระบบจัดการฐานข้อมูลให้ด้วย



Simply embed SQL in your host program and let the ORACLE Precompilers handle the DBMS interface for you.

ORACLE Precompilers เป็น full ANSI-compatible ซึ่งแอปพลิเคชั่นที่พัฒนาโดยใช้ precompilers สามารถที่จะรันได้ทั้งบน ORACLE และ ANSI-Compatible SQL based DBMS รวมทั้ง DB2 ด้วยและเราสามารถที่จะสร้าง customized application เช่น เราสามารถจะออกแบบแอปพลิเคชั่นให้ดึงข้อมูลจาก ORACLE บนเมนเฟรม และการประมวลข้อมูลบนไมโครคอมพิวเตอร์ รวมทั้งให้แสดงข้อมูลบน graphic work station ได้

3) SQL Form และ SQL Menu

แอปพลิเคชั่นที่เขียนด้วย SQL Form and Menus จะสามาร์ถรันได้บนทุกๆ platformที่ ORACLE สนับสนุน

เราสามารถสร้างแอปพลิเคชั่นโดยไม่ต้องเขียน code แต่ใช้วิธีระบุสเป็คของแอปพลิเคชั่น แทนโดยใช้ menu-driven user interface specification ดังกล่าวร่วมกับข้อมูลที่เก็บใน ORACLE Data Dictionary จะถูกใช้ในการสร้างแอปพลิเคชั่นและสามารถใช้ร่วมกับ PL/SQL routines รวมทั้งสามารถเพิ่ม 3GL routine ลงใน แอปพลิเคชั่นได้ด้วย

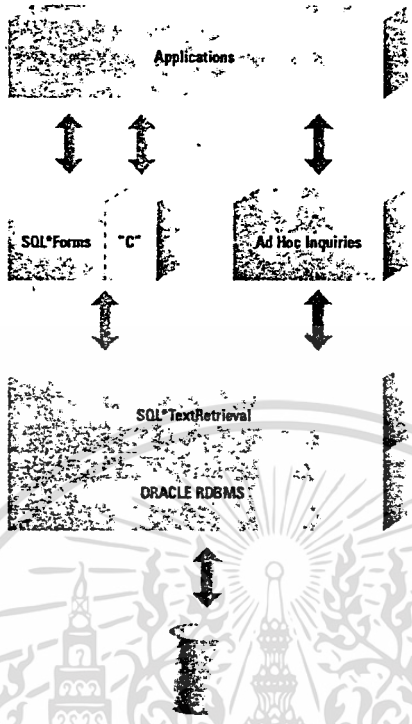
4) SQL TextRetrival

เป็นความสามารถที่สามารถจะเพิ่มเข้าไปใน ORACLE applicaiton ต่างๆได้เป็นการขยาย SQL ผลที่ได้จะเป็น common language สำหรับที่เก็บของข้อมูลและตัวอักษร

SQL TextRetrival จะเก็บข้อมูลทั้งหมดรวมทั้งตัวักษรต้นแบบและindexes ด้วยผู้ใช้จะเข้าถึงข้อมูลทุกตัวได้ภายใต้การควบคุมของ ORACLE ผู้ใช้จะสามารถเข้าถึง SQL TextRetrival software Tools ได้โดยผ่าน SQL Form หรือ 3GL programs

SQL TextRetrival ทำให้การค้นหามีประสิทธิภาพขึ้นโดย query สามารถที่จะรวมเอา words, concepts และ โครงสร้างของฟิลด์ข้อมูลไว้ได้และสามารถที่จะใช้ตัวอักษรจากแหล่งมาตราฐานต่างๆ เช่น external file, screen input เป็นต้นเรายังสามารถจะแก้ไขตัวอักษรโดยใช้ editor ใดๆก็ได้และทำการ import, export ตัวอักษรระหว่างแอปพลิเคชั่น

SQL TextRetrival จะสะดวกในการเคลื่อนย้ายบนหลายๆ platform เช่นเดียวกับเครื่องมืออื่นๆด้วย



5) SQL ReportWriter

เป็นเครื่องมือที่ทำให้ผู้ใช้สามารถที่จะสร้างรายงานได้หลายๆรูปแบบโดยใช้ข้อมูลจากหลายๆฐานข้อมูลทั้ง ORACLE และ non-ORACLE เช่น DB2 โดยไม่ต้องเขียนโปรแกรมเพียงแต่ระบุ report specification ลงใน forms (ซึ่งเป็นรูปแบบ spread sheet) SQL ReportWriter จะมี defaultให้ทั้งตำแหน่ง column, column heading และ display format เราสามารถที่จะ override default ได้ถ้าต้องการ

SQL ReportWriter สามารถใช้เชื่อมกับ ORACLE tools อื่นๆได้เช่นกัน

6) SQL PLUS

ใช้ในการ query,define และ control data ที่เก็บใน ORACLE RDBMS มีความสามารถเช่นเดียวกับ ORACLE/SQL และ PL/SQL

ด้วย syntax ง่ายๆของ SQL PLUS ผู้ใช้สามารถที่จะเข้าถึงข้อมูลได้อย่างรวดเร็วและยัง
สามารถที่จะดึงdata, ทำการคำนวนและสร้าง adhoc report ได้

SQL PLUS เป็น dynamic SQL-based queries คือผู้ใช้จะสามารถเปลี่ยน section ของ query ได้ขณะรันเราสามารถจะเพิ่ม PL/SQL block ลงใน SQL statementได้ (เป็น procedure logic) โดยการใช้ SQL PLUS

7) ORACLE CARD

เป็นเครื่องมือที่ใช้ในการพัฒนาแอปพลิเคชั่นทางกราฟฟิคโดยวิธีง่ายๆและแอปพลิเคชั่น ยังสามารถรันได้บนหลายๆ platforms ทั้ง MS-Windows และ Apple Macintosh systems CARD จะทำให้ผู้ใช้สามารถใช้ GUI (graphic, image) ร่วมกับฐานข้อมูลแบบรีเรชั่นนัลได้

ในการสร้างและควบคุมฐานข้อมูล table,develop application และรวมทั้งการสร้าง report นั้นผู้ใช้เพียงแต่ point และ click mouse เชื่อม screen object (fields) เข้ากับ database object (column) แล้ว ORACLE จะจัดการส่วนที่เหลือเองโดยผู้ใช้ไม่ต้องเขียนโปรแกรมหรือ SQL ใดๆเลยและผู้ใช้ยังสามารถที่จะเก็บและดึงภาพในฐานข้อมูล ได้โดยการ click button ด้วย

ORACLE CARD ประกอบ เครื่องมือ เช่น

- Table Builder ใช้ในการสร้างและแก้ไขตารางในฐานข้อมูลโดยไม่ต้องเขียน SQL เลย
- Stack Builder ใช้พัฒนาแอปพลิเคชั่นโดยการ point และ click mouse และแอปพลิเคชั่น

ที่ได้ผู้ใช้จะสามารถตรวจสอบ แทรก และแก้ไขทั้งตัวอักษรและภาพข้อมูลในตาราง

- Query Builder จะทำ adhoc queries ทางกราฟฟิคจะเก็บข้อมูลลงใน clipboard หรือ simple report ก็ได้
- Programming Environment ใช้ในการสร้างแอปพลิเคชั่นที่ซับซ้อนมากขึ้น

ในการสร้างแอปพลิเคชั่นโดยใช้ ORACLE CARD ผู้ใช้จะเลือกที่จะเขียนโปรแกรมหรือไม่เขียนก็ได้โดยในส่วนของการโปรแกรมจะมี ORACLE CARD's object-oriented programming language ให้เรียกว่า ORACLE Talk ซึ่งจะอนุญาตให้มีการเพิ่ม procedural logic และ 3GL function เช่น C routines เข้าไปได้ ORACLE Talk จะสื่อสารกับฐานข้อมูลของ ORACLE โดยผ่านทาง ORACLE access (interface มาตราฐานของ ORACLE ระหว่าง scripting environment เช่น ORACLE Talk กับ back end) ORACLE access จะทำให้ ผู้ใช้ สามารถที่จะเขียน SQL หรือ PL/SQL statement ลงใน ORACLE Talk script ได้

ORACLE CARD จะมีเครื่องมือทางกราฟฟิคให้มากมายเพื่อให้ผู้ใช้สามารถจะทำการรวมภาพและสีลงในแอปพลิเคชั่นได้

3.7.3 Office Automation and End-user Tools

1) ORACLE Data Query

ถูกออกแบบมาเพื่อให้ผู้ใช้ใช้ในการระบุและรัน ad hoc queries ได้โดยไม่จำเป็นต้องรู้ SQL ในการสร้าง queries ผู้ใช้จะเลือกตารางและ column จาก lists การใช้งานเป็นแบบ step by step โดยใช้วิธี fill in the blank input ของ ผู้ใช้ จะถูกตรวจสอบทุกขั้นตอนถ้าพบข้อผิดพลาดจะแก้ไขในทันที

ORACLE Data Query จะอนุญาตให้มีการสร้าง report หลายๆแบบได้จาก 1 query สามารถคำนวนทางสถิติให้ได้ และข้อมูลสามารถจะถูกเรียงลำดับได้หลายวิธีด้วย

ขณะที่ query ถูก execute อยู่ผลสามารถจะถูกแสดงขึ้นจอ, เขียนลง file, หรือส่งตรงไปยัง printer เลยก็ได้

ORACLE Data Query สามารถจะรันได้บนหลายๆ platforms (รวมทั้ง client/sever configuration) และสนับสนุนทั้ง character-mode และ block-mode terminal

ORACLE Data Query เป็น เครื่องมือแบบ read only ผู้ใช้จะไม่สามารถแทรก,แก้ไข หรือ ลบข้อมูลได้และ DBA จะใช้ resource governor เพื่อจำกัดว่าผู้ใช้แต่ละคนจะสามารถรัน query ได้ยาวเท่าใด

2) ORACLE Mail

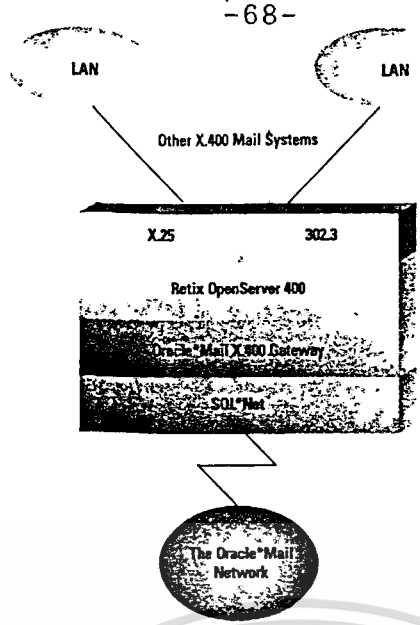
จะสะดวกในการเคลื่อนย้าย ผู้ใช้จะสามารถใช้ระบบ electronic mail เดียวกันได้บนกว่า 80 platforms

Forms, Reports, Batch program หรือแอปพลิเคชั่นสามารถที่จะส่งและรับ mail ซึ่งเป็นข้อมูลประเภทใดก็ได้จากที่ใดก็ได้ใน enterprise-wide system

ORACLE Mail มี communication gateway จึงสามารถติดต่อกับ major Email อื่นๆได้การใช้งาน mail ก็ทำได้ง่ายๆโดยจะมี pulldown menu เพื่อช่วยแนะนำวิธีการใช้ให้

3) ORACLE Mail x.400 GATEWAY

จะทำให้ ORACLE Mail สามารถติดต่อกับ x.400 compatible mail system อื่นๆได้



ORACLE GATEWAY จะใช้ ORACLE SQL Net networking softwareใน การชื่อมกับ ORACLE Mail และ x.400 solution จะใช้ protocal x.25 ในการเชื่อมกับ wide area network (WAN) หรือ Ethenet เพื่อเชื่อมเข้ากับ LAN GATEWAY จะสนับสนุนทั้ง simple message, word-processing document, spread sheet และ graphic image ได้ด้วย

4) SQL QMX

เป็น query manager และ report writer ซึ่งจะรวมเอา browser, query facilities และ report writer ไว้ใน package

ในการใช้ SQL QMX ผู้ใช้สามารถที่จะใช้ function key-driven menus หรือจะใส่ คำสั่งที่ยังไม่สมบูรณ์ก็ได้ SQL QMX จะเติมส่วนที่เหลือให้โดยผู้ใช้ไม่จำเป็นต้องรู้ command syntax สำหรับreport writer ก็จะอยู่ใน fill-in-the-blank formatting panel SQL QMX browser จะช่วยให้ผู้ใช้สามารถหาตารางหรือ view ได้โดยไม่ต้อง สร้าง queries และเมื่อผู้ใช้สร้าง report browser ก็จะให้ผู้ใช้สามารถที่จะ preview report บนหน้าจอก่อนจะ print ได้

ผู้ใช้สามารถที่จะเลือกเขียน SQL statement แทนการใช้ prompt panel ก็ได้ SQL QMX ยังสนับสนุน macro procedure และ substitution variables ด้วย

5) ORACLE Database add-in for lotus 1-2-3

จะอนุญาติให้ผู้ใช้ เลือก, แก้ไข, แทรก และลบข้อมูล ในฐานข้อมูลของ ORACLE ได้โดยตรงจาก 1-2-3 spreadsheet (ไม่ต้องทำ hard copy ออกมาแล้วค่อยไปใช้ เครื่อง มือ อื่นๆช่วย) เมื่อ ผู้ใช้ดึงข้อมูลเข้ามาใน spreadsheet แล้ว ORACLE 1-2-3 จะอนุ

ญาติให้สร้าง what-if analysis, run report และสร้างผลเป็นกราฟได้ในการดึงข้อมูลสามารถทำได้ง่ายๆโดย fill-in-the-blank หรือใช้การกดคีย์เพียงไม่กี่ครั้ง

ในการเข้าถึงฐานข้อมูลผ่าน ORACLE for 1-2-3 integrity ของฐานข้อมูลจะถูกรักษาไว้ตลอดเวลาเพราะในการเข้าถึงข้อมูลในฐานข้อมูลผู้ใช้ที่จะสามารถแก้ไขฐานข้อมูลได้ต้องเป็นผู้ใช้ที่มีสิทธ์แก้ไข privileges เท่านั้น

6) ORACLE for 1-2-3 DATALENS

ORACLE's database driver สำหรับ Lotus 1-2-3 for VAX/VMS เป็น GATEWAY ที่ทำให้ผู้ใช้สามารถใช้ 1-2-3 ในการเข้าถึงข้อมูลในฐานข้อมูลของ ORACLE ใดๆก็ได้ (ทั้งของ remote และ local machine)ใน enterprise-wide information network ORACLE for 1-2-3 DATALENS สนับสนุนทุกๆหน้าที่และคำสั่งของ 1-2-3 ผู้ใช้จึงไม่ต้องใช้ความรู้เกี่ยวกับ SQL เลยในการรวมข้อมูลลงใน 1-2-3 แอปพลิเคชั่น

ORACLE for 1-2-3 DATALENS ให้ผู้ใช้สามารถป้องกัน spreadsheet ข้อมูลได้โดยอาศัย security features ของฐานข้อมูล ORACLE และผู้ใช้จะสามารถให้ access ให้ผู้อื่นได้และข้อมูลเหล่านี้จะ garuntee integrity และ consistency ด้วย

บทที่ 4

การวิเคราะห์ระบบฐานข้อมูลของ INGRES

4.1 Structure storage และ Architecture ของ INGRES

4.1.1 Structure storage ของ INGRES

Structure storage คือการจัดการไฟล์เพื่อใช้ในการเข้าถึงข้อมูลในตารางการที่มี Key จะทำให้สามารถเข้าถึงข้อมูลได้รวดเร็วกว่าไม่มี Key

INGRES มี storage structure 4 แบบ คือ

1) Heap :- เป็นการ เข้าถึง ข้อมูล แบบ sequential

2) Hash :- มีหลักการในการเลือก address โดยใช้ค่าของ key data เป็นตัวกำหนด

3) Isam :- จะจัดเรียงข้อมูลโดยใช้ค่าของ key column สำหรับการเข้าถึงแบบเร็วบนค่าที่แน่นอนและการดึงข้อมูลเป็นช่วง, index เป็นแบบ static และต้องการการแก้ไขทุกๆครั้งที่ตารางมีขนาดใหญ่ขึ้น

4) Btree :- จะจัดเรียงข้อมูลโดยใช้ค่าของ key column สำหรับการเข้าถึงแบบเร็วบนค่าที่แน่นอนและการดึงข้อมูลเป็นช่วง, index เป็นแบบ dynamic และจะโตตามขนาดตารางที่ใหญ่ขึ้น

4.1.1.1 Heap

จะไม่มี key จะเป็นลักษณะการเก็บข้อมูลแบบกองทับ ดังนั้นเมื่อเพิ่มเติม row ใดๆก็จะเก็บ row นั้นไว้ท้ายสุด

Heap เป็น Storage structure ที่ดีหากใช้ในกรณีที่

1) Load ข้อมูลจากตาราง เนื่องจากเป็น storage structure ที่เพิ่มเติมข้อมูลได้เร็วที่สุด เนื่องจากจะเพิ่มเติมข้อมูลเข้าไปที่ท้ายของ heap ดังนั้นจึงไม่มี overhead ที่เกิดจากการคำนวณ page ที่จะต้องใส่ข้อมูลนี้

2) เป็นตารางที่มีขนาดเล็ก (1 ตารางมีอยู่ไม่กี่ page)

3) ดึงข้อมูลทุกๆ row โดยไม่ต้อง sorting

4) ใช้ secondary index กับตารางที่มีขนาดใหญ่และต้องการรักษาพื้นที่ไว้

ข้อเสียของ Heap Storage Structure และวิธีแก้ปัญหา

1) การเข้าถึงตารางใหม่ที่สร้างโดยคำสั่ง " create table as select " ได้นั้น

แก้ปัญหาโดย : ใช้ clause "with structure = type" เมื่อ execute คำสั่ง create

2) Space ที่ deleted rows จะไม่นำมาใช้ใหม่

แก้ปัญหาโดย : ใช้ statement modify เพื่อนำ Space นั้นมาใช้ใหม่

1.3) Select และ แก้ไขช้ามากเพราะ เข้าถึง แบบ sequential

แก้ปัญหาโดย : ถ้าเป็น ตาราง ขนาดเล็กก็ไม่เป็นปัญหาแต่ถ้าเป็น ตาราง ขนาดใหญ่ก็สร้าง secondary index

4.1.1.2 Hash

เป็นโครงสร้าง ที่ใช้ match key value หาก storage structure แบบนี้จะต้องมีการกำหนด key ซึ่งถ้าไม่มีการกำหนดจะใช้ฟิลด์แรกเป็น key

Hash เป็น Storage structure ที่ดีหากใช้ในกรณีที่

1) ดึงข้อมูลโดยใช้ key value ที่กำหนด
2) ต้องการวินิจฉัย ที่ใช้เนื่องจาก storage structure นี้ สามารถคำนวณได้อย่างรวดเร็ว

ข้อเสียของ Hash Storage Structure และวิธีแก้ปัญหา

1) ไม่เหมาะกับรูปแบบที่เป็นการ match เพราะ Performance จะลดลง
แก้ปัญหาโดย : ใช้ Isam, Btree แทน
2) ไม่เหมาะกับการ ดึง ช่วงของค่า (range of values) เพราะ Performance จะลดลง
แก้ปัญหาโดย : ใช้ Isam, Btree แทน
3) ไม่ควรใช้กับส่วนใดของ multi-column key เพราะ Performance จะลดลง
แก้ปัญหาโดย : ใช้ Isam, Btree แทน
4) มี overflow page ใน ตาราง ที่แก้ไขใหม่
แก้ปัญหาโดย : ถ้า key เป็น unique, Algorithm ของ hash อาจจะกระจายข้อมูลได้ไม่ดี
ถ้า column เป็น character column ก็ให้ใช้ Isam แทน
5) มี overflow page หลังจากมีการเพิ่มเติม rows
แก้ปัญหาโดย : แก้ไขใหม่หรือใช้ dynamic structure แบบ Btree

4.1.1.3 Isam

เป็น storage structure ซึ่งใช้ match key value แต่จะสมบูรณ์กว่า hash เนื่องจากสามารถดึงช่วงของ key values ได้อีกทั้งยังทำการ match แบบมีรูปแบบได้และใช้ partial key ทำได้ดีพอๆกับใช้การทำให้ตรงกับค่าที่มีการกำหนดแน่นอน storage structure แบบนี้จะต้องมีการกำหนด key ซึ่งถ้าไม่กำหนดจะใช้ column แรกเป็น key

Isam จะใช้ static index ชี้ไปยัง static numbers ของ page หลักซึ่งเป็น static เช่นเดียวกัน หากต้องการเปลี่ยนแปลงตารางก็ถูกจัดลำดับโดย key ที่กำหนด ดังนั้นก็จะสร้าง sparse index ขึ้นซึ่ง index จะประกอบไปด้วยช่วงของ key และ pointer

ไปยัง index pages หรือไม่ก็ data page

ถ้าจะเติมหลาย rows ให้แก้ไขใหม่เพื่อหลีกเลี่ยง overflow pages ดังนั้นสำหรับ Static table storage structure แบบ Isam อาจจะเหมาะสมกว่า Btree

Isam เป็น Storage structure ที่ดีหากใช้ในกรณีที่

1) เมื่อตารางเป็นแบบ static

2) ดึงข้อมูลเป็นแบบ - Pattern matching , ช่วงของ key values และเป็นส่วนซ้ายสุดของ multi-column เท่านั้น

3) สามารถรองรับการทำงานที่มีผู้ใช้เข้าถึงตารางพร้อม ๆกันหลายๆ คน

ข้อเสียของ Isam Storage Structure

1) ไม่เหมาะกับ ตาราง ที่มีการขยายตัวอย่างรวดเร็วเพราะจะทำให้เกิด overflow page

2) ไม่เหมาะกับ ตาราง ที่มีขนาดใหญ่เกินกว่าจะ modify เพราะจะทำให้เกิด overflow page

3) key เป็นแบบ sequential

4) หากใช้ pattern matching แล้วไม่ระบุ character ซ้ายสุด จะทำให้ต้อง scan

ทั้งตารางเพราะหากไม่ระบุค่ามาก็ไม่สามารถเปลี่ยนแปลง สภาวะของการค้นหาได้

4.1.1.4 Btree

- เป็น storage structure ที่ดีที่สุด สามารถใช้การเข้าถึงโดย key และสามารถรองรับการค้นหาแบบเป็นช่วง และ การ match แบบมี pattern
- index ของ Btree เป็นแบบ dynamics จะเพิ่มตามการเพิ่มขนาดของตาราง
- ใช้ sparse index ชี้ไปยัง page ใน leaf level
- ใน leaf level จะเป็น dense index ซึ่งมี key ชี้ไปยัง row บน data page ในตาราง

โครงสร้างของ Btree table

จะพิจารณาเป็น 4 ส่วนคือ

1) free list header page :- ซึ่งใช้ในการรักษาแนวทาง (track) ของ page ที่ allocate แล้วและไม่ได้ใช้อยู่ในปัจจุบัน

2) index pages 1 pages หรือมากกว่า

3) leaf pages 1 pages หรือมากกว่า

4) data pages 1 pages หรือมากกว่าที่ซึ่ง data users ได้ถูกเก็บจริง ๆ

- index page จะเป็น sparse index ซึ่งจะเก็บ key และ pointer ที่ชี้ไปยัง leaf page
- leaf page จะเป็น dense index เพราะ hold key ไว้และจะชี้ไปยังทุกๆ rows ใน ตาราง
- data page เก็บ data rows และจะไม่มี pointer ที่ page นี้

หมายเหตุ BTree index level คล้ายกับ Isam's index เว้นเสียแต่ Btree's index จะ point ไปที่ index หรือ leaf pages ก่อน แทนที่จะ point ไปที่ data pages เลย

Index growth

ข้อแตกต่างที่สำคัญที่สุดระหว่าง Isam กับ Btree คือ Btree นั้นใช้ dynamic index กล่าวคือ dynamic index จะโตตามการเพิ่มขนาดของตารางในขณะที่ Isam นั้นใช้ static index

- Isam จะไม่มีการจำกัด page สุดท้ายของตารางดังนั้นถ้ามีการเพิ่มเติมข้อมูลที่เป็น key ลงไปใน page นั้นเรื่อย ๆ ก็อาจจะเกิด overflow page ที่ page สุดท้ายได้

ในขณะที่

- Btree นั้นจะมีการจำกัด leaf page ที่เป็น page สุดท้ายไว้หากมีการเพิ่มเติมข้อมูลที่เป็น key จน page นั้นเต็มก็จะเกิดการแบ่งแยกออกเป็น 2 leaf paged ทำให้ไม่เกิด overflow pages และใน index page ก็ใช้หลักการเดียวกัน เพราะฉะนั้นจะเห็นได้ว่า index จะเพิ่มตามขนาดของตาราง

Locking และ Btree tables

โดยปกติแล้ว leaf และ data page จะถูก locks จนกระทั่งจบ transaction หากมีการ locks เกิดขึ้นที่ leaf level ในขณะที่ค้นหา Btree index ก็จะเกิดการ lock ใน index page ตามทางที่จะไป leaf page นั้นและทำการ lock แบบชั่วคราวตามลำดับ level ที่เป็น parent level ของleaf page ที่เราต้องการจะเข้าถึง

Delete Rows: Data Pages

ถ้า rows ที่ถูกลบอยู่บน data page ที่มีลักษณะเป็นแบบ associated, เนื้อที่ว่างนั้นจะถูกนำมาใช้อีกครั้งแต่ถ้ามีหลายๆ rows ที่อยู่บน data page ที่มีลักษณะเป็นแบบ nonassociated และถูก ลบ ก็จะนำแต่ละ page ไปวางไว้ที่ free list

ข้อดี - สามารถใช้กับ ตาราง ที่มีอัตราการโตอย่างรวดเร็ว

- สามารถใช้กับ pattern matching
- สามารถใช้กับการ ดึง key value แบบเป็นช่วง
- สามารถ ดึง โดยใช้เพียงแค่ leftmost ของ multi-column key เท่านั้น

ข้อเสีย - ไม่เหมาะที่จะใช้กับตารางที่มีลักษณะเป็นแบบ relatively static

- ไม่เหมาะที่จะใช้กับตารางที่มีขนาดเล็กและมีคนใช้พร้อม ๆ กันหลาย ๆ คน

- หากจะใช้การ match แบบมีรูปแบบแต่ไม่กำหนด character ตัวซ้ายสุด จะทำให้ต้อง scan ทั้ง ตาราง ซึ่งเป็นผลให้เสียเวลา

- หากจะใช้บางส่วนของ multi-column key แต่ไม่ระบุ column ซ้ายสุด จะทำให้ต้องสร้าง secondary index ที่เป็น column ที่ใช้ค้นหาอยู่

- จะทำให้ลบบ่อยพอๆกับเพิ่มเติมข้อมูล

สรุปเปรียบเทียบกรณีที่เลือกใช้ Isam แทนที่จะใช้ Btree storage structure

1) กรณีที่ใช้ static table (ตารางที่มีขนาดคงที่)

2) ใช้ disk operation ในการเข้า data page น้อยกว่า Btree เพราะ Btree ต้องทำถึง leaf level

3) ไม่ต้อง lock จาก index page ดังนั้น concurrency จะดีกว่า

4) Sequential traversal ของตารางแบบ Isam ที่ไม่เรียงลำดับจะทำได้เร็วกว่า Btree เพราะ Isam มี pointer ที่ชี้ไปยัง data pages โดยตรง

สรุปเปรียบเทียบกรณีที่เลือกใช้ Btree แทนที่จะใช้Isam storage structure

1) ตาราง มีอัตราการเพิ่มขนาดอย่างรวดเร็ว (จนอาจทำให้เกิด overflow ใน Isam storage structure)

2) ใช้เรียงลำดับได้ง่ายเพราะทำการเข้าถึงแบบ sequential

4.2 Architecture ของ INGRES

มีสถาปัตยกรรมแบบ multi server architecture กล่าวคือเป็นแบบ " multithreaded " กล่าวคือยอมให้มี DBMS process server หลาย process server ซึ่งเข้าถึงฐานข้อมูล เดียวกัน ซึ่งมีผลดีคือ

- ลด overhead ต่อ ผู้ใช้:- ลด overhead ของ ผู้ใช้ แต่ละคนให้เหลือแค่ 30-50 K ในขณะที่ แบบ single-threaded จะต้องใช้ถึง 200-400K

- ใช้ประโยชน์ของ multiprogramming ได้อย่างคุ้มค่าเมื่อเทียบกับแบบ process model

- สามารถกำหนดคุณสมบัติซึ่งทำให้ RDBMS สามารถทำงานได้อย่างมีประสิทธิภาพมากที่สุด เช่น ตั้ง priority ให้กับ on-line transaction processing server มากกว่า concurrent decision support หรือ reporting server

4.2 Data Independence ของ INGRES

Physical Data Independence จะพิจารณาตามหัวข้อต่อไปนี้

- Representation of numeric data ใน INGRES ไม่มีคุณสมบัตินี้เพราะ data จะมีการเก็บแบบ binary เท่านั้น DBA ไม่มีสิทธิเลือก
- Representation of character data ใน INGRES สามารถแทนได้ทั้ง 2 แบบ
- Unit for numeric data ใน INGRES ไม่มีคุณสมบัตินี้เพราะไม่มีการเก็บหน่วยของ data จะเก็บเฉพาะ value เฉยๆ
- Data Coding ใน INGRES ไม่สามารถเปลี่ยนเฉยๆแล้วให้ program รู้เองได้ต้องมีการ define บอกเอาไว้ใน program
- Structure of stored file ใน INGRES มีคุณสมบัตินี้

Logical Data Independence

INGRES มีคุณสมบัติข้อนี้คือ DBA สามารถที่จะเปลี่ยนแปลงตาราง เพิ่ม column, ตัด column, เพิ่มตาราง สิ่งเหล่านี้จะไม่มีผลต่อการเข้าถึงตารางของผู้ใช้ไม่ต้องมีการแก้ไข program ใหม่

4.3 Concurrency ของ INGRES

INGRES เป็น concurrency database โดยจะยอมให้ผู้ใช้หลายๆคนเข้าถึงข้อมูลเดียวกันในเวลาเดียวกัน ซึ่งคำนึงถึง

- concurrency หมายถึง การที่ผู้ใช้ทุกคนสามารถเข้าถึงข้อมูลใด ๆในเวลาเดียวกัน
- consistency หมายถึง การที่สามารถแน่ใจได้ว่าการเปลี่ยนแปลงฐานข้อมูลจะทำให้เกิดค่าที่ถูกต้องเหมือนกันและอยู่ในลำดับซึ่งเป็นไปตามโครงสร้างของข้อมูล

INGRES ใช้ระบบ locking ในการจัดการการเข้าถึง resource ซึ่งประกอบไปด้วย databases, ตารางและ page เพื่อรับประกัน consistency ของ ข้อมูล ที่มีการ share กัน

ระบบ locking ของ INGRES จะใช้กลไกที่เรียกว่า "MULTIPLE GRANULARITY" โดยมีการ lock ใน 2 ระดับ คือ

1. logical locks ซึ่งจะ locks อยู่ตราบเท่าที่ transaction ทำงานอยู่ ซึ่ง logical lock นี้จะถูกยกเลิกก็ต่อเมื่อ transaction นั้นถูก commit,roll back หรือ abort
2. physical locks เป็น locks ของระบบ locking ของ INGRES มีหน้าที่จัดการการ เข้าถึง resource ซึ่งอาจเกิดขึ้นในเวลาเดียวกันและ physical lock สามารถ lockและ unlock ภายใน transaction ได้ด้วย

ซึ่งมีหลักการทำงานดังต่อไปนี้

Locks Modes

locks ของ INGRES มีการทำงานอยู่ 6 mode

- Mode of Lock -	คำอธิบาย
- X	- exclusive locks หรือ write locks ซึ่งจะมี transaction
-	- เพียงคนเดียวเท่านั้นที่สามารถ lock resource ได้ผู้ใช้ที่ lock
-	- แบบนี้ เรียกว่า "writer"

-	S	- shared lockหรือ read locksซึ่ง transactionหลายๆtransaction
-		- สามารถ lock resource ได้โดย transaction ที่ lock แบบนี้จะ
-		- ไม่สามารถ แก้ไขฐานข้อมูล ได้ ผู้ใช้ ที่ lock แบบนี้เรียกว่า
-		- " reader "
-	IX,IS	- Intended exclusive, Intend shared locks โดยเมื่อใช้ X,S
-		- lock page ใน ตาราง ใดแล้ว INGRES ก็จะใช้ IX,IS lock
-		- ตาราง เพื่อแสดง locking granularity โดย
-		- IX-lock จะแสดงว่า page ใน ตาราง นี้อาจจะถูกแก้ไขอยู่
-		- IS-lock จะแสดงว่า page ใน ตาราง นี้กำลังถูกอ่านอยู่
-	SIX	- shared intended exclusive lock ซึ่ง buffer manager ของ
-		- INGRES จะใช้ในการแก้ไข pages ใน cache ของ buffer
-		- manager
-	N	- Null locks คือไม่มีการ lock หรือเป็นการรักษาโครงสร้างไว้
-		- เพื่อใช้ต่อไป

level ของการ locks

level ของการ lock จะหมายถึง ขอบเขตของ resource ซึ่งต้องการ lock แบ่งออกเป็น

1) Page : ควบคุมการเข้าถึงในระดับ page

2) Table : ควบคุมการเข้าถึงในระดับตาราง

3) Database : ผู้ใช้ block โดย exclusive block ซึ่งทำให้ไม่สามารถติดต่อกับฐานข้อมูลเองได้ และจะแสดงข้อความแสดงความผิดพลาดเพื่อชี้ว่ามี exclusive lock อยู่

4) Control : เป็น physical locks ที่ใช้จัดการตารางในขณะที่กลไกมีการเปลี่ยนแปลง

DBA จะจัดการเกี่ยวกับข้อกำหนด (configuration ของระบบ locking โดยจะตั้งจำนวนครั้งของการ lock ที่เกิดขึ้น และเมื่อมีการ lock เกิดขึ้น ค่าที่ตั้งไว้จะลดลงทีละ 1

เมื่อใดที่ INGRES จะ lock นั้นขึ้นอยู่กับ mode ของ lock ที่ transaction อื่น ๆ ใช้ lock resource อยู่ซึ่งสามารถ check ได้จากตารางแสดงโหมดของการ lock

Deadlock

Deadlock เป็นสภาวะที่เกิดการรอคอยการ lock อย่างไม่มีกำหนดซึ่งเกิดขึ้นเมื่อ transaction หนึ่งรอ lock ซึ่งอีก transaction หนึ่ง lock อยู่และเป็นเวลาเดียวกับที่ transaction นั้นรอ lock ที่ transaction แรก hold อยู่

เมื่อเกิด Deadlock ระบบ locking ของ INGRES จะยกเลิก transaction ใด transaction หนึ่งเพื่อให้อีก transaction ทำงานต่อไป ส่วน transaction ที่ถูกยกเลิกจะได้รับข้อความที่บอกความผิดพลาด จากนั้นทุกๆ statement ใน transaction ตั้งแต่การ commit ครั้งสุดท้ายจนถึงจุดที่เกิด deadlock จะถูก recovery มา

เนื่องจาก INGRES ใช้การ lock แบบ page-level locking ดังนั้นถ้ามีอย่างน้อย 2 transaction ที่ต้องการจะเข้าถึงฐานข้อมูล และมีผู้ใช้อย่างน้อย 2 คน พยายามจะเปลี่ยนแปลง row แล้วอาจจะเกิด deadlock ใน query เดี่ยวๆได้หาก

1) มีการเข้าถึง pages โดยใช้ path ที่แตกต่างกันในตาราง

หมายความว่า transaction หลาย transaction พยายามจะแก้ไขตารางโดยใช้ access path ที่แตกต่างกัน เช่น ใน ตาราง หนึ่งมี isam structure ซึ่ง index คนละ field กับ ที่ hash index

2) เกิด Locks Escalation

ซึ่งมีสาเหตุมาจากกรณีใดกรณีหนึ่งต่อไปนี้

2.1) convert จาก shared locks เป็น exclusive locks

2.2) long overflow chains :- ตาราง ที่มี overflow pages เกินจะทำให้เกิดปัญหาเพราะ INGRES จะต้องหา overflow pages เหล่านั้น และจะ lock แต่ละ page และเก็บรักษาไว้เพื่อลด overflow chain

2.3) มีการ locks เกินค่า maxlock ของระบบ

2.4) transaction อยู่ใน lock limit

2.5) index ของ Btree เกิดการ split :- กรณีที่มี ผู้ใช้ หลายคน แทรกเพิ่ม ลงบน Btree table เล็กๆ ก็อาจทำให้เกิดได้

INGRES มี feature ในการจัดการ deadlock ด้วย แอปพลิเคชั่น

มีแอปพลิเคชั่นที่ใช้ในการจัดการ check deadlock หลังจากแต่ละ Statement ของ multiple query transaction ถ้าเกิด deadlock เมื่อทำ statement ของ transactionใดทำอยู่ ก็จะ abort transaction นั้นจนกว่าจะไม่เกิด deadlock แอปพลิเคชั่น นี้เขียนด้วย Embeded SQL/FORTRAN

วิธีที่ INGRES จะใช้ในการจัดการ deadlock ในสภาวะที่เป็น heavy concurrent แบ่งเป็น 2 วิธีคือ

1) Table Locks Approach

เพื่อให้เกิด deadlock น้อยที่สุด เมื่อมี concurrent activity เกิดขึ้นมาก ๆใน ตาราง เล็ก หรือ ส่วนหนึ่งใน ตาราง ใหญ่ๆ ซึ่งจะให้ ผู้ใช้ แต่ละคนมีลำดับการ เข้าถึง ตาราง ถึงแม้ว่าเป็นการเสียเวลาแต่ก็สามารถหลีกเลี่ยงสภาวะ deadlock ได้

2) Never Escalation Approach

จะใช้ในกรณีที่ ผู้ใช้ ทำงานในส่วนต่างๆของ ตาราง ถึงแม้ว่าเป็นเพียงการ รัน query หรือ แก้ไขง่ายๆแต่ก็ให้มีการใช้ primary และ secondary index มาก ๆ เพื่อทำให้ ผู้ใช้ สามารถใช้ ตาราง เดียวกันได้โดยไม่มี ผู้ใช้ ใดใช้วิธี locks เพื่อไม่ให้ ผู้ใช้ คนอื่นใช้

หมายเหตุ : หากเกิด deadlock เพราะ Escalation บ่อยๆ อาจจะต้อง

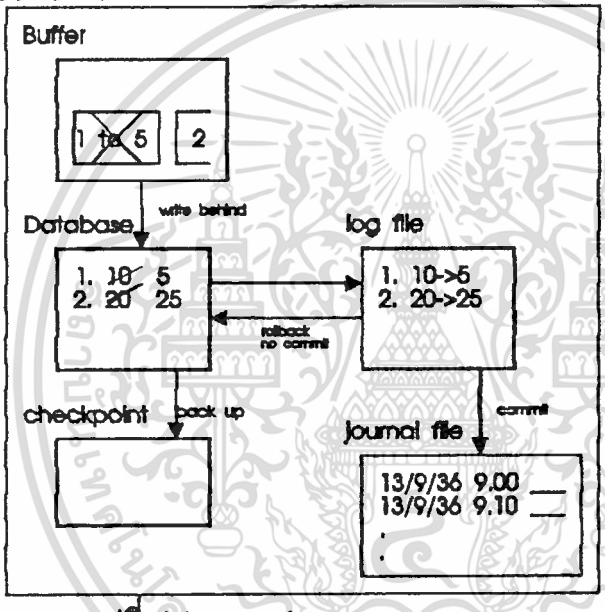
- ตั้งการ locks ให้มี level ของการ locks แบบ "table-level locking" เป็น default สำหรับ ตาราง
- เพิ่มค่า maxlock ของระบบ

4.4 Backup and recovery ของ INGRES

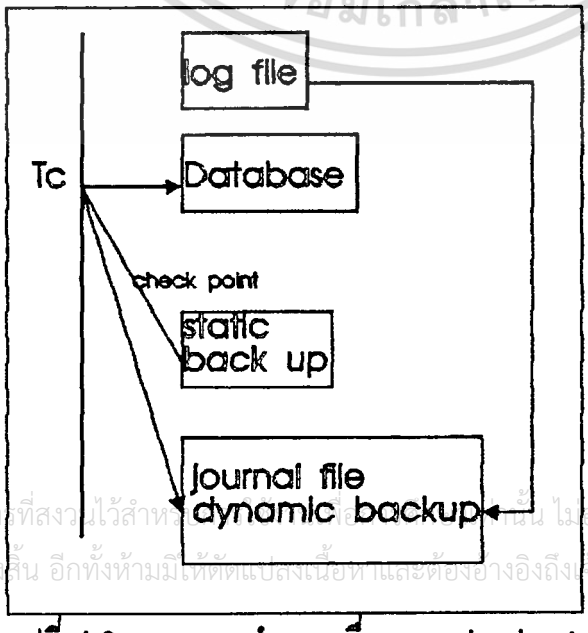
จะมีวิธีต่างๆดังนี้

1) Log - based Recovery ซึ่งในการ recovery แบบนี้ mode ที่ INGRES ใช้ได้แก่ Fast commit ซึ่งก็คือวิธี Defered Mode นั่นเอง ซึ่งในจุดนี้ INGRES ใช้ commercial term ว่า Fast commit ก็คงเนื่องจากจับจุดมุ่งไปที่ transaction ขนาดเล็กๆนั่นเองและ INGRES ก็ยังมองข้อดีของวิธีนี้ตรงที่เป็น I/O Reduction technique อันหนึ่ง

Logging System ของ INGRES เป็นดังนี้



รูปที่ 4.1 แสดง logging system



รูปที่ 4.2 แสดงการทำงานเมื่อเวลา checkpoint

Log file ของ INGRES จะเก็บ log record ซึ่งเป็นการเปลี่ยนแปลงระหว่างการทำ transaction หนึ่งๆตั้งแต่เริ่มต้น transaction จนถึง commit ซึ่ ถ้า transaction commit ก็จะเอาสิ่งที่อยู่ใน log file นั้นเก็บลง Journal file

Journal file เป็น file ที่ INGRES เอาไว้เก็บการเปลี่ยนแปลงทั้งหมดตั้งแต่ Checkpoint ครั้งสุดท้ายซึ่งจะเก็บไล่เป็นวันและเวลาเลยว่าขณะเวลานั้นเกิดการเปลี่ยนแปลงใน ฐานข้อมูลอย่างไรบ้าง ซึ่งข้อมูลใน journal file ก็จะได้รับมาจาก log file

Log file ต่างกับ Journal file ตรงที่ Journal file จะเก็บข้อมูลน้อยกว่า Log file โดยที่ Journal file จะเก็บข้อมูลเฉพาะตารางที่สั่งให้ทำ Journaling เท่านั้น

Fast commit และ Write behind

ในการที่ transaction ทำการแก้ไขฐานข้อมูลนั้น จริงๆแล้ว transaction ไม่ได้ทำการแก้ไขที่ตัวฐานข้อมูลโดยตรงแต่จะทำการแก้ไขไปที่ buffer แทนพร้อมทั้งมีการ write การเปลี่ยนแปลงลง log file ซึ่งหลักการนี้ก็เป็นหลักของ Fast commit และหลังจากที่มีการเปลี่ยนแปลงใน buffer แล้วก็จะต้องมีการ write การเปลี่ยนแปลงนั้นลง ฐานข้อมูลจริงๆ ซึ่งการ write นี้ใน INGRES จะเรียกว่า Write behind

INGRES จะทำการ write behind ก็ต่อเมื่อ Server ว่างพอที่จะทำ

Checkpoint ของ INGRES นั้นจะหมายถึงการ backup ฐานข้อมูลปัจจุบันที่มีอยู่ทั้งอันลงบน binary file ๆ หนึ่งไปยัง stable storage เรียกว่าเป็นการทำ Static backup DBA จะเป็นผู้ run คำสั่ง Checkpoint เท่านั้นซึ่งจะ run เมื่อใดก็แล้วแต่วิจารณญาณของ DBA อาจจะสัปดาห์หนึ่งทำครั้งหนึ่ง หรือ ทำทุกวันก็ได้ซึ่งในการทำ Checkpoint นั้น ผู้ใช้ที่ใช้งานฐานข้อมูลอยู่และยังทำไม่เสร็จก็ยังคงทำต่อไปได้ INGRES จะรอจนผู้ใช้นั้นทำเสร็จและข้อมูลที่อยู่ใน buffer ได้ write behind ลงฐานข้อมูลแล้วจึงจะทำการ backup ส่วนผู้ใช้ที่ไม่ได้ใช้งาน databasetable และต้องการที่จะเข้ามาใช้ในขณะ Checkpoint นั้นจะไม่มีสิทธิเพราะตารางจะโดน lock ไว้ต้องรอให้ Checkpoint เสร็จจึงจะเข้ามาเรียกใช้ตารางได้

Checkpoint จะเป็นการ backup ฐานข้อมูลที่มีอยู่เท่านั้น ถ้าหากฐานข้อมูลนั้นถูกทำลายไป(destroy) จะเป็นการ delete checkpoint ไปด้วย ดังนั้นจะไม่สามารถสร้างฐานข้อมูลนั้นขึ้นมาใหม่จาก checkpoint ได้

Journaling คือการเก็บการเปลี่ยนแปลงของฐานข้อมูลตั้งแต่ Checkpoint ครั้งสุดท้ายลงบน journal file เรียกว่าเป็นการทำ Dynamic backup ซึ่งในการ recover จะสามารถกู้ข้อมูลได้จาก Static backup ที่ Checkpoint ครั้งสุดท้ายมารวมกับ Dynamic backup ที่ได้จากการทำ Journaling หลัง checkpoint ซึ่งจะทำให้สามารถกู้ข้อมูลกลับมาได้อย่างสมบูรณ์

การทำ Journaling นั้นจะต้องบอกเอาไว้ตั้งแต่ตอน สร้างตารางแล้วว่าจะ set ให้ทำที่ตารางใดบ้าง ซึ่งส่วนมากจะทำเฉพาะตารางที่สำคัญๆ มีการเปลี่ยนแปลงบ่อยๆ เช่น ในตารางการฝาก ถอน เงิน ส่วนตารางที่ไม่ได้เปลี่ยนบ่อยๆ เช่น เก็บข้อมูลพนักงานก็ไม่จำเป็นต้อง set journaling ซึ่งตารางที่ไม่ได้มีการทำjournaling ไว้นั้นการกู้ข้อมูลก็จะทำได้ถึงแค่เวลา Checkpoint ครั้งสุดท้ายเท่านั้น

File Storage

การเก็บ file ของ INGRES จะเป็นแบบ multiple file storage คือเก็บตารางหนึ่งเป็นหลายๆ file ซึ่งมีข้อดีในแง่ของการ recovery ก็คือถ้าเกิด file เสียไป file หนึ่งก็จะเป็นความเสียหายเล็กๆ เพราะไม่ได้เสียหายทั้งตารางและ file นั้นเพียง file เดียวที่ต้องทำการกู้กลับมาจาก backup ซึ่งจะช่วยให้ Recovery time เร็วมาก

Feature ในการ commit เพิ่มเติม

1) Group commit หรืออีกชื่อคือ Piggybacked commit คือเป็นการ group commit information จากหลายๆ ผู้ใช้รวมกันเพื่อประโยชน์ในการลด transaction log file bottlenecks และเป็นการลดปริมาณ I/O operation ด้วย

2) Two - Phase commit เป็น feature สำหรับ multiuser คือเมื่อผู้ใช้ในฝั่งหนึ่งทำการcommit แล้วอีกฝั่งหนึ่งก็จะ commit ด้วยหรือถ้าฝั่งหนึ่ง rollback อีกฝั่งก็จะ rollback ตาม

ข้อดีของการ backup แบบ Checkpoint คือ เก็บเป็น file เดียวไม่เปลืองที่

ข้อเสียของการ backup แบบ Checkpoint คือ เนื่องจากเก็บ file เป็น binary จึงเป็น machine dependent ถ้าข้ามเครื่องจะใช้ Checkpoint ไม่ได้

2) Backup โดยการ copy ฐานข้อมูล

จะเป็นการ backup ฐานข้อมูลที่เป็นของตัวเองไม่ใช่ฐานข้อมูลทั้งหมดโดยใช้คำสั่ง copydb ของ INGRES ซึ่งจะเก็บเป็นแบบ text file ซึ่งคนอ่านรู้เรื่อง แต่เปลืองเนื้อที่

วิธีนี้จะมีข้อดีตรงที่เป็นการ backup สำหรับผู้ใช้ที่ไม่ได้เป็น DBA ซึ่งทำให้ผู้ใช้เหล่านั้นสามารถ backup ตาราง, view และ procedure ของตัวเองไว้

3) Backup โดยการ Unloaded และ Reloaded

จะคล้ายกับการ backup แบบ copy และเก็บเป็น text file เช่นกัน

ในการ backup แบบนี้จะเป็นการ copy เก็บทั้ง System table และ User table ทั้งหมดซึ่งใน System table จะประกอบด้วย

ส่วนของ Server ซึ่งเป็นส่วนสำคัญได้แก่

- เก็บว่ามีตารางอะไรบ้าง
- มี column อะไรบ้างและ column ไหนเป็นของตารางไหน
- เก็บรายละเอียดของผู้ใช้
- เก็บ view
- เก็บ secondary index ของแต่ละตาราง
- เก็บ rule ของแต่ละตาราง
- เก็บ database procedure

ส่วนของ Front End เช่น

- มี form อะไรบ้าง
- เก็บ QBF name (Form + ตาราง)
- เก็บ Join dept ที่บอกว่าตารางใด join กับตารางใด

4.5 Data Integrity ของ INGRES

INGRES จะยอมให้เข้าไปหรือแก้ไขข้อมูลได้โดยจะต้องมีการตรวจสอบค่าของ ข้อมูลให้ถูกต้องและเหมาะสมอย่างอัตโนมัติภายใต้ data integrity และกฎต่างๆ

- Integrity อาจจะถูกกำหนดสำหรับ column ที่ต้องการในตารางโดยตารางของ owner ซึ่งก็หมายถึง DBA หรือไม่ก็ private user

INGRES จะยอมให้มีการเก็บ referentail integrity, business rules, database rules, database event alerters ไว้ใน server ซึ่งวิธีนี้จะทำให้สามารถดูแล integrity ของข้อมูลจาก centralized location

การสร้าง integrity statement

ใช้กำหนด integrities ซึ่งทำการตรวจสอบเพื่อให้แน่ใจว่า rows ที่ปรากฏในตารางนั้นเป็นไปตามสภาวะที่กำหนด ซึ่งถ้าไม่เป็นไปตามที่กำหนดก็จะถือว่า reject ไป

เช่น
```
----------------------------------------
- creat integrity on employee is -
-     Salary > 10000;              -
----------------------------------------
```

4.5.1 Database procedure

คือกลุ่มฟังก์ชั่น statement ซึ่งมองว่าคล้ายกับเป็น object หนึ่งของ INGRES ในส่วนที่เป็น database definition

(statement นั้นหมายถึง การ query ฐานข้อมูล 1 หรือมากกว่า เช่น select,แทรก เพิ่ม,ลบ,แก้ไข)

- การ สร้าง procedure มี 2 ประเภทคือ 1) public
  2) private

1) public :- DBA เป็นผู้สร้าง database procedure และ DBA จะเป็นผู้อนุญาตให้ผู้ใช้เรียกใช้ database procedure นี้

2) private :- ผู้ใช้เป็นผู้สร้าง database procedure ซึ่ง database procedure นี้มีผู้ใช้ที่สร้างเท่านั้นที่สามารถเรียกใช้ได้

- object ทั้งหมดที่อ้างอิงใน database procedure จะถูกแก้ไขใหม่ทันทีที่ procedure นั้นถูกสร้าง

- ในการเรียกใช้ database procedure ใช้ได้ทั้งแบบ interactive หรือใช้กับ Embedded SQL

ข้อดี 1) ลดจำนวนครั้งของการติดต่อระหว่างแอปพลิเคชั่นกับ ระบบจัดการฐานข้อมูล

2) DBA สามารถแก้ไขและควบคุมการเข้าถึงข้อมูลได้ในระดับ extra level

- DBA ต้องเป็น owner ของตารางที่อ้างถึงใน database procedure ที่ DBA สร้างขึ้นและ DBA จะเป็นผู้อนุญาติให้ผู้ใช้ใช้ database procedure ดังกล่าวแต่จะมีข้อจำกัดกล่าวคือ จะระบุระดับของการเข้าถึงโดย DBA จะไม่ยอมให้ผู้ใช้ได้รับอนุญาตแบบเต็ม query

3) procedure สามารถใช้ได้หลายแอปพลิเคชั่นใน database procedure ซึ่งทำให้ลดเวลาในการโปรแกรม

หมายเหตุ จะเห็นได้ว่าหากมีการแก้ไข constraint ก็จะไปแก้ที่ database procedure ซึ่งไม่จำเป็นต้องไปตามแก้ที่ แอปพลิเคชั่น ที่มาใช้งาน

4.5.2 กฎ (Rules)

หมายถึง กลไกซึ่งกำหนดขึ้นเองเพื่อให้ทำตาม database procedure เมื่อฐานข้อมูลเปลี่ยนแปลงไปตามที่กำหนดไว้กล่าวคือ

- จะ execute database procedure เมื่อสภาวะที่กำหนดเป็นจริง

- ผู้ที่จะสร้างกฎจะเป็น DBA หรือผู้ใช้ก็ได้แต่ทางที่ดีควรจะเป็น DBA เป็นผู้สร้างและหากผู้ใช้ต้องการจะสร้างกฎสำหรับตารางของตนเองจริง ๆ ก็ควรจะอยู่ภายใต้ความควบคุมของ DBA

กฎสามารถทำให้เกิด Data Integrity ได้ดังนี้

- Referential Integrity

โดยจะดูแลความสัมพันธ์ระหว่าง 2 ตาราง เช่นค่าใน column ของตารางหนึ่งจะต้อง match กับค่าใน column ของอีกตาราง โดยตารางทั้งสองจะมีความสัมพันธ์แบบ parent-child ซึ่ง column ใน parent table จะเรียกว่า "primary key" ในขณะที่ column ใน child table จะเรียกว่า " foreign key"

มีวิธีจัดการ 3 วิธีคือ

1) Reject Method :- จะทำการ roll back statement ทั้งหมดที่เกี่ยวข้องกับกฎ

2) Nullify Method :- จะทำการ set entry ที่ถือว่าเป็น children ของอีก entry ให้มีค่าเป็น NULL

3) Cascade Method :- จะทำการแก้ไข statement ทั้งหมดที่เกี่ยวข้องกับ กฎ

ให้มีค่าตรงกัน

- General Integrity

เป็น Integrity ใดๆซึ่งไม่ใช่ Referential Integrity ทำหน้าที่อธิบายความ สัมพันธ์ระหว่าง original data และ derive data

- General Purpose Rules

เป็น กฎ ที่สามารควบคุม external source ได้ด้วยโดยจะแน่ใจได้ว่าข้อมูลที่กำหนดให้ถูก updaate นั้นไม่ถูกเปลี่ยนแปลงโดยผู้ไม่ได้รับอนุญาต

หมายเหตุ เกี่ยวกับ Entity integrity นั้น

- เนื่องจากการสร้าง integrities บน column ซึ่งมีค่าเป็น NULL ตัว สภาวะ จะพิจารณาถึงความเป็นไปได้ที่จะเกี่ยวข้องกับค่า NULL ด้วยและเนื่องจาก NULL ไม่มีค่าเป็นของตัวเอง ดังนั้นการเปรียบเทียบในสภาวะเงื่อนไขจึงให้ผลเป็น false จึงต้องมีการกำหนด integrity เพื่อให้จัดการกรณีที่ column ที่พิจารณาเป็น NULL สามารถระบุตั้งแต่สร้างตารางได้ว่าข้อมูลที่อยู่ในตารางนั้นต้องมีค่าไม่เป็น NULL

4.5.3 Database Events

จะทำให้แอปพลิเคชั่นหรือระบบจัดการฐานข้อมูลของ INGRES บอกให้แอปพลิเคชั่นอื่นๆ ทำงานเมื่อเป็นไปตามเหตุการณ์ที่เกิดขึ้นซึ่งจับได้จาก database procedure

โดยทั่วๆ ไป event จะมีหลักการทำงานคือ

- แอปพลิเคชั่นหรือระบบจัดการฐานข้อมูลของ INGRES raise events
- ระบบจัดการฐานข้อมูลของ INGRES ให้ monitor แอปพลิเคชั่นที่ register ว่าจะรับ event เท่านั้น
- แอปพลิเคชั่นที่รับ event จะตอบสนอง event ที่เกิดขึ้นแล้วนำไปเป็นเงื่อนไขในการทำงานตามที่ระบุใน Program

ลำดับการใช้ event ใน conjunction ด้วยกฎของ INGRES คือ

1) สร้าง event (สร้าง database events)(สร้าง โดย DBA)

2) raising event (raise dbevent) :- สามารถเรียกใช้จาก interactive หรือ Embeded SQL หรือ database procedure ซึ่งจะทำหน้าที่ส่ง event นี้ไปให้ server รับทราบจากนั้น server ก็จะส่ง dbevent นี้ ไปให้ผู้ใช้ที่ลง register ไว้ว่าต้องการรับ dbevent

3) registering to receive an event (register dbevent)
:- เพื่อระบุว่าแอปพลิเคชั่นนี้ต้องการรับ dbevent

4) receivng an event (get dbevent) :- รับ dbevent

5) dropping an event (drop dbevent):- เฉพาะผู้ใช้ที่เป็นผู้สร้าง dbevent นี้เท่านั้นที่สามารถยกเลิกได้ เมื่อยกเลิก event นี้แล้วก็ไม่สามารถเรียกใช้หรือ ทำให้เกิด dbevent ได้อีก

หลักการตรวจสอบความถูกต้องของข้อมูล (Data Integrity)

- create dbevent A;
- create procedure B

```
begin
    {** check integrity **}
    raise dbevent A;
end;
```

- create rule C

```
after update
execute procedure B;
```

- ทาง application check

```
if (evname = 'A') then
   send mail
endif;
```

4.6 Query Optimization ของINGRES

INGRES's Intelligent Query Optimizer เป็นชื่อที่ทาง INGRES ตั้งเอาไว้เพื่อแสดงความสามารถในการเป็น expert system ของตัว Optimizer ซึ่งจะเป็นตัวที่พิจารณาหากลวิธีที่ดีที่สุดสำหรับการquery แต่ละครั้ง โดยจะวัดจากสถิติในการหาเส้นทางในครั้งก่อนๆ และใช้ heuristic ซึ่งข้อดีของ Query Optimizer นี้คือการเป็น Syntax - Independent คือ INGRES จะทำการ normalize query ให้อยู่ในรูปแบบที่ common ก่อนการทำ Optimization ดังนั้น Performance ของการ query จะไม่ขึ้นอยู่กับลำดับข้อความของ query

เช่นการ Select * From A,B จะมีค่าเท่ากับ Select * From B,A

การ Optimizedb ใน INGRES

Optimizedb จะมีผลกับ speed ของการทำ query processing แต่ไม่ได้บอกว่า query นั้นสามารถ execute ได้หรือไม่ ซึ่งถ้ามีการเก็บค่า statistics ที่ถูกต้องและสมบูรณ์แล้วก็จะทำให้ประสิทธิภาพของการ query ดีขึ้นและมีผลทำให้ performance ของระบบเร็วขึ้น

ถ้าไม่มีการ run Optimizedb แล้ว Optimizer ก็จะสรุปว่า

1) ในการจับคู่ที่เป็น exact match ทั้งหมดในตารางจะมี row return ออกมา 1% ของตาราง ยกเว้นกรณีที่ key หรือ index ถูกระบุให้ unique ซึ่งกรณีนี้จะมีเพียง row เดียวที่ถูก return กลับมาสำหรับ attribute ที่ถูก Index ไว้เช่น

WHERE emp.empno = 275

2) นอกเหนือจากการณีการจับคู่ที่เป็น exact match แล้วกรณีอื่นจะมี row return 10% ของตาราง ดังนั้นถ้ามี qualification ที่ไม่เป็น exact อยู่ 3 อันก็จะมีจำนวนตารางที่ถูก select ดังนี้คือ

1/10 * 1/10 * 1/10 = 1/1000

3) join ทุกอันจะถูกสรุปให้เป็น 1:1 โดยมี base อยู่บน data set ของตารางอันที่เล็กกว่า เช่น เมื่อตาราง1(มี 100 rows) ถูก join กับ ตาราง2(มี 1000 rows) ผลลัพธ์โดยประมาณก็จะได้ 100 rows

4) กรณีที่มีเงื่อนไขบน join ตารางแล้ว จำนวน row ลัพธ์จะมากกว่าหรือเท่ากับ lower bound ของ 10% ของ row ที่มาจากตารางที่เล็กกว่า

สิ่งที่เก็บอยู่ใน Optimizedb

สิ่งที่เก็บไว้สำหรับทำ Query Execution Plan ได้แก่

1) จำนวนของ row ในตาราง ซึ่งโดยปกติค่านี้จะมีอยู่ใน system catalog อยู่แล้ว

2) จำนวนของ unique value ใน column หนึ่งๆ

3) เก็บค่า NULL count ซึ่งแสดงถึงเปอร์เซนต์ของ column value ที่เป็น NULL

4) จำนวนของduplicate value ใน column หนึ่งๆโดยเฉลี่ยซึ่งค่านี้เรียกว่า repetition factor

5) histogram ที่แสดง data distribution ซึ่งจาก histogram เราจะได้ค่า minimum และ maximum สำหรับ column หนึ่งๆ

Information ต่างๆเหล่านี้จะถูกใช้โดย INGRES query optimizer เพื่อคำนวณ cost ของ QEP แต่ละอัน

ชนิดของ statistics

1) Non-sampled statistic คือทุกๆ row บนตารางจะถูกดึงออกมา

2) Sampled statistics คือ subset ของ row จากตารางถูกดึงออกมา

ใน default mode ของ Optimizedb จะเป็นแบบ Non-sampled คือทุกๆ row ของตารางจะถูกใช้ในการ generate statistics ซึ่ง mode นี้จะได้ statistic ที่ถูกต้องมากที่สุดเท่าที่จะเป็นไปได้เพราะจะมีการพิจารณา data ทุกๆค่า

ใน mode ของ Sampled statistics นั้นจะเหมาะกับ base table ที่ใหญ่มากๆ ซึ่งใน Optimizedb นี้จะสามารถระบุเปอร์เซนต์ของ row ที่จะถูก sample ได้และถ้าใช้การ sampling ที่เพียงพอก็จะทำให้ได้ statistics ที่คล้ายคลึงกับการทำ statistics บน full table เลย

นอกจากนี้ทั้ง Non-sampled และ Sampled ก็ยังมี option การทำ sataistics ที่เกี่ยวกับการกระจายของ data ที่ statistic จะ hold ได้คือ

1) Full statistics จะ copy information ทั้งหมดเกี่ยวกับ data distribution ซึ่ง cost จะสูงที่สุดสำหรับแต่ละ column ของตารางจะถูก scan เพียงครั้งเดียวและค่าของ column จะถูกดึงออกมาใน sorted order ดังนั้นต้องมีการสร้าง temporary table ที่มีการ sort data process ของการทำ full statistics สามารถถูกทำให้สั้นลงได้โดยใช้ option ของการ sampling (คือใช้ Sampled statistic) ซึ่งจะทำให้สามารถเลือกได้ว่าจะพิจารณา row ออกมากี่ % ของตารางแล้วทำ full statistics

2) Minmax statistics เป็นวิธีที่ถูกกว่า full statistics โดยปกติจะ require การ scan เพียงครั้งเดียวสำหรับตารางทั้งอัน statistics นี้จะสร้างจาก information ของค่า minimum และ maximum สำหรับแต่ละ column

Minmax ก็มีการใช้ option ของทั้ง Sampled และ Non-sampled ได้ซึ่งถ้าใช้ Non-sampled ก็จะได้ข้อมูลที่ถูกต้องกว่าแต่ใน sampled ก็จะทำให้ process สั้นลงได้ซึ่งก็สามารถเลือก

ได้ว่าจะให้พิจารณา row กี่ % ของตาราง ทั้งหมดแล้วก็เอา row ส่วนที่ดึงออกมานั้นทำ Minmax statistics

นอกจากนี้ทั้ง Full statistics และ Minmax statistics ก็ยังมี option อีกตัวให้เลือกคือ Key Column Statistics ซึ่งเป็นการทำ Full หรือ Minmax บน key หรือ column ที่ถูก index ไว้เท่านั้น

Input Text File Statistics

statistics สามารถอ่านเข้ามาจาก text file ได้โดย input text file ต้องอยู่ใน format ที่ตายตัวซึ่ง file นี้สามารถที่จะถูก edit เพื่อบอกการเปลี่ยนแปลงในการกระจายของ data ตามต้องการก่อนที่จะทำการ Optimizedb ซึ่งจะมีประโยชน์เมื่อ

1) Information กำลังถูกย้ายจาก datbase หนึ่งไปยังอีกอันหนึ่ง (ใช้คำสั่ง copy) และต้องการที่จะย้าย statistics สำหรับตารางนั้นไปด้วย

2) รู้การกระจายของ data ในตารางและต้องการที่จะอ่านหรือใส่ค่าลงไปโดยตรงแทนการให้ Optimizedb ทำการสร้างให้

Query Execution Plan (QEP)

เมื่อ INGRES Optimizer ทำการexecute query นั้นจะมีการ generate QEP ที่แสดงถึงว่า query จะถูก execute อย่างไร เมื่อ QEP ถูก generate ขึ้นมาครั้งหนึ่ง INGRES สามารถใช้มันได้มากกว่า 1 ครั้งเพื่อ execute database query รูปแบบนี้

QEP นี้สามารถ print ออกมาดูได้ว่า query ทำงานอย่างไรและถูกควบคุมจาก INGRES Optimizer

Compiled Database Procedures

ใน INGRES จะมี Compiled Database Procedures ซึ่งเป็น function ที่ถูกเขียนขึ้นมาโดย INGRES/4GL เป็น function ที่ถูก compile เอาไว้เรียบร้อยแล้วและจัดเก็บ transaction เหล่านี้เอาไว้ใน Share memory

ข้อดีของการมี Compiled Database Procedure คือ

1) ลดการใช้ CPU เนื่องจากเป็นการ Compile เก็บเอาไว้

2) transaction ที่ถูก Compile แล้วจะเก็บเอาไว้ใน Share memory ทำให้ลดปริมาณ memory ที่ใช้ลงแทนการเก็บ copy เอาไว้ให้แต่ละผู้ใช้

3) ลด Network Overhead เนื่องจากวิธีนี้จะทำให้การ Compile สะดวกโดยแทนที่จะต้องเขียน SQL ยาวๆเพื่อทำ query ก็สามารถส่งแค่ Procedure name และ Parameter ต่างๆไปให้ Compiled Database Procedure เท่านั้นทำให้ลดการ communication ระหว่าง client/server และ Network traffic ต่างๆลงได้

4) จะเกิดการ Enforce Integrity คือ ผู้ใช้ และ programmer จะไม่ได้

เข้าถึงเข้าไปใน data โดยตรง แต่ใช้การแก้ไขและดึงข้อมูลผ่านทาง Procedure

ส่วนในการทำInterpret INGRES ก็สามารถทำการ Interpret query ได้โดยการส่งชุดของ SQL ซึ่งเป็นการใช้คำสั่งในการแก้ไขและดึงข้อมูลจากฐานข้อมูลขึ้นมาดูโดยตรงได้

นอกจากนี้ใน INGRES จะมี feature หนึ่งที่ใช้ในการจัดการ knowledge ภายใน database server คือ feature ทางด้าน Resource Control

Resource Control Feature ของ INGRES Query Optimizer จะเป็นตัวจำกัดการใช้ query ของผู้ใช้คือจะมีการทำนาย Performance ของระบบโดยการป้องกันไม่ให้เกิด query ที่ return row ออกมาไม่รู้จบจะมีการจำกัด row ที่ยอมให้ return ออกมา เช่น ในกรณีที่ผู้ใช้เกิดพลาดไป Select * ออกมาจากตารางใหญ่โตมีเป็นแสนๆ record ออกมาโดยไม่ได้ตั้งใจ ก็จะเกิดการ return row ออกมาทั้งตาราง โดยไม่หยุด นอกจากการ control ในเรื่อง row แล้วก็ยังมีการ control การใช้ disk I/O ด้วย

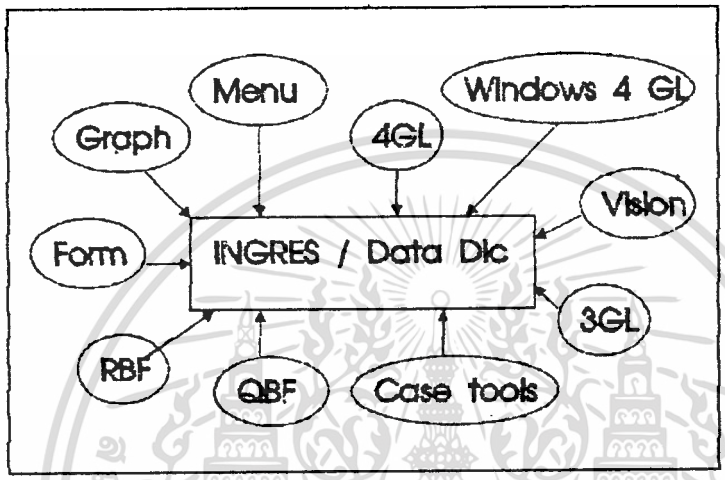
4.7 INGRES Tool set

Tool set จะถูกแบ่งออกเป็น 3 พวกใหญ่ๆ คือ

1) End-user Tools

2) Application Development Tools

3) CASE Tools



รูปที่ 4.3 แสดง tool set ของ INGRES

4.7.1 End-user Tools

INGRES มีเครื่องมือที่ใช้ง่ายๆ สำหรับผู้ใช้ทั่วไป โดยผู้ใช้ไม่จำเป็นต้องเขียโปรแกรม หรือรู้คำสั่ง SQL และเครื่องมือทุกๆตัวจะมีการใช้งานเหมือนกันหมด (consistent user interface) ดังนั้นการเรียนรู้จึงทำเพียงครั้งเดียวก็จะสามารถใช้งาน เครื่องมือ ที่เหลืออื่นๆ ได้ เครื่องมือ ต่างๆ ประกอบด้วย

1) INGRES/Menu เป็น shell ที่ครอบเครื่องมือทั้งหมดของ INGRES เพื่อให้ผู้ใช้เรียกใช้งานได้สะดวกและรวดเร็ว

2) INGRES/Query-By-Forms(QBF) เป็นเครื่องมือช่วยในการทำ query ของผู้ใช้ โดยที่ผู้ใช้ไม่จำเป็นต้องรู้ SQL เงื่อนไขสามารถถูกกำหนดได้อย่างอิสระและเก็บไว้ใช้ในอนาคตได้อีกสำหรับหน้าจอของ INGRES/QBF จะขึ้นมาให้เป็น default แต่ผู้ใช้สามารถตกแต่งใหม่ได้ตามความต้องการ (โดยการใช้ INGRES/Vifred)

3) INGRES/Report-By-Forms(RBF) เป็น เครื่องมือ สำหรับสร้างรายงานที่ง่าย เพราะมีคำสั่งในลักษณะเป็นตัวเลือก (menus) ให้ RBF สามารถทำและจัดรูปแบบรายงานโดยไม่ต้องเขียนโปรแกรมรายงานที่สร้างขึ้นสามารถเก็บและเรียกใช้ได้ในภายหลัง

4) INGRES/Vision-FoRms-Editor(Vifred) คือเครื่องมือที่ใช้ในการตกแต่งหน้าจอ ซึ่งผู้ใช้สามารถสร้างหน้าจอได้ตามที่ต้องการหน้าจอที่สร้างด้วย Vifred สามารถใช้งานได้กับ INGRES/QBF,INGRES/4GL และ 3GL เช่น C,COBOL,FORTRAN เป็นต้น

5) INGRES/Graph คือเครื่องมือในทาง graphics ซึ่งผู้ใช้จะสามารถใช้ในการสร้างกราฟเพื่อแสดงผลได้

4.7.2 Application Devolopment Tools

INGRES มีเครื่องมือดังต่อไปนี้คือ

1) INGRES/3GL
2) INGRES/4GL
3) INGRES/Windows4GL
4) INGRES/Vision

INGRES/Vision

INGRES/Vision เป็น code generator สำหรับสร้าง 4GL application สามารถทำงานได้ง่ายโดยการกำหนดหน้าจอ ความสัมพันธ์ระหว่างตารางและระดับการทำงานระหว่างหน้าจอโดยใช้ visual programming และเราจะสามารถสร้างโครงสร้างของแอปพลิเคชั่นได้ง่ายๆเหมือนกับการสร้าง organization chart INGRES/Vision จะช่วยสร้างโปรแกรมที่สมบูรณ์ที่ประกอบด้วย หน้าจอ (forms),menus,queries,4GL หรือแม้แต่การจัดการข้อผิดพลาดหากทำงานไม่สำเร็จโดยอัตโนมัติ การใช้ INGRES/Vision จึงทำให้

- ลดระยะเวลาที่ต้องใช้ในการพัฒนา แอปพลิเคชั่นใหม่ลงได้อย่างมาก
- สามารถสร้างแอปพลิเคชั่นที่ flexible และ powerful ได้ในเวลาอันสั้น
- การบำรุงรักษาและการพัฒนาแอปพลิเคชั่นทำได้ง่าย

การสร้าง INGRES/Vision application

INGRES/Visionจะประกอบด้วย 2 ส่วนหลักๆคือ Application Flow Diagram และ Visual Query Editor

Application Flow Diagram จะใช้สำหรับการสร้างโครงสร้างของแอปพลิเคชั่นโดยขั้นตอนแรกก็จะต้องระบุ screen layout(frames) ของแอปพลเคชั่นและในการสร้างเฟรมจะต้องระบุชนิดของเฟรมคือพวก menu update append และบอกด้วยว่าตารางของข้อมูลใดที่เฟรมจะเข้าถึงรวมทั้งบอกด้วยว่าถ้าผู้ใช้เลือก menu item ใดแล้วจะให้ทำเฟรมใด Visual Query Editor ใช้ในการระบุข้อมูลที่จะถูกเข้าถึงและ operation ที่จะทำสำหรับแต่ละเฟรม

เมื่อเรากำหนดรายละเอียดต่างๆแล้วINGRES/Vision ก็จะสร้าง code ที่ต้องการสำหรับแอปพลิเคชั่นให้โดยอัตโนมัติและโปรแกรมที่สร้างขึ้นยังสามารถจะเพิ่มเติมและเปลี่ยนแปลงได้ โดยไม่กระทบต่อ code ที่ INGRES/Vision สร้างขึ้นยิ่งไปกว่านั้นผู้ใช้ยังสามารถปรับปรุงเปลี่ยนแปลง template หรือ help file ที่สร้างโดย INGRES/Vision ได้ตามต้องการอีกด้วย การเพิ่มเติมโปรแกรมผู้ใช้สามารถใช้ได้ทั้ง 4GL 3GL หรือเรียกเครื่องมืออื่นๆของ INGRES ก็ได้ INGRES/Vision ยังสนับสนุน 3GL 4GL และ database procedure คือ

- การสร้าง sequential keys โดยอัตโนมัติ
- การส่งผ่านข้อมูลระหว่าง forms
- ตัวแปลทั้ง global และ local
- concerrency control และ recovery และ multiple developers ด้วย

การทำการปรับcode ที่สร้างโดย INGRES/Vision สามารถทำได้ตามที่กล่าวมา ตัวอย่าง code ที่เปลี่ยนได้ เช่น

- แก้ไขเปลี่ยนแปลงข้อความหรือลำดับของ menu item ได้
- เพิ่มทำ menu item เข้าในแอปพลิเคชั่นได้
- แก้ massage ที่ INGRES/Vision จะแสดงในแอปพลิเคชั่นได้
- แก้ไขวิธีการทำ error recovery ได้ เป็นต้น

และแอปพลิเคชั่นต่างๆที่สร้างโดยใช้ INGRES/Vision จะสนับสนุนการใช้เครื่องมือของ INGRES เช่น graph QBF RBF ฯลฯ ด้วย

จึงกล่าวได้ว่า INGRES/Vision สนับสนุนการทำงานของผู้พัฒนา ตั้งแต่การทำ prototype การทดสอบ ใช้งานจริงและบำรุงรักษาทำให้ลดเวลาและขั้นตอนในการทงาน กว่าครึ่งเมื่อเทียบกับ 4GL หรือกว่า 10 เท่าเมื่อเทียบกับ 3GL

INGRES/Window4GL

INGRES/Window4GL เป็น object oriented 4GL สำหรับการพัฒนาแอปพลิเคชั่นทางกราฟฟิค ซึ่งมี

- powerful object oriented 4GL และ debuger
- ความสามารถที่จะใช้ได้กับหลาย platform โดยไม่ต้องมีการแก้ไข code คือมีสภาพแวดล้อมที่สนับสนุนหลาย window manager และหลาย platform
- facilities สำหรับการจัดการแอปพลิเคชั่นที่ซับซ้อนได้

INGRES/Window4GL จึงเหมาะสำหรับการสร้างแอปพลิเคชั่นของฐานข้อมูลแบบ client/server ที่ต้องการทั้ง window toolkit SQL database interface และ

programming language เช่น C C++ การใช้ INGRES/Window4GL นี้จะสามารถประหยัดเวลาในการพัฒนาแอปพลิเคชั่นได้ถึง 3-4 เท่า เมื่อเทียบกับ 3GL-based graphical tools

ส่วนประกอบของ INGRES/Window4GL

1) Visual Editor

ใช้สำหรับการทำหน้าจอ (form), menu, การจัดรูปภาพและ object oriented 4GL โดยผู้เขียนไม่ต้องติดต่อโดยตรงกับ toolkit ของ window manager

2) Interactive Debugger

debugger จะมี เครื่องมือ ที่ผู้ใช้ต้องการอยู่เลยทำให้ผู้ใช้สามารถที่จะดูแอปพลิเคชั่น ขณะที่รันอยู่พร้อมๆกับดู source code ด้วยและอนุญาติให้ระบุ break point ได้มากกว่า 1 แห่งด้วย

3) System Class Library

มี class library ให้มากกว่า 60 class ทำให้ผู้ใช้สามารถที่จะเขียนโปรแกรมแบบออบเจคท์-โอเรียนเต็ด โดยไม่ต้องรู้ซึ้งเกี่ยวกับทฤษฎีของ OOP

Portable Capabilities

แอปพลิเคชั่นที่พัฒนาบน INGRES/Window4GL สามารถที่จะใช้ได้กับหลายๆ OS, hardware platforms และ window managers เช่น Microsoft Windows, Presentation manager, OSF/Motif และ Openlook window managers โดยไม่ต้องเปลี่ยนแปลงโปรแกรมแต่อย่างใด

INGRES/Window4GL เป็น event-based programming คือแอปพลิเคชั่นที่ถูกสร้างขึ้นจาก event blocks เมื่อมี event ใดๆเกิดขึ้น block of code ของ event นั้นจะถูก execute ตัวอย่างของ event เช่น การ click mouse, การเปลี่ยนขนาดของ window, การรับ massage จาก window อื่น เป็นต้น

นอกจากนั้น INGRES/Window4GL application ยังสามารถจะส่งข้อมูล ระหว่าง window ได้ทำให้สามารถมี multiple active window ได้ เช่นถ้า 2 window มีข้อมูลที่เกี่ยวข้องกันการเปลี่ยนแปลงข้อมูลใน window หนึ่งจะทำให้อีก window หนึ่งถูกเปลี่ยนแปลงโดยอัตโนมัติ โดยการใช้ database event คือเมื่อ batch program ทำการ แก้ไขฐานข้อมูลเสร็จเรียบร้อยแล้วจะสามารถบอกแอปพลิเคชั่นว่าการแก้ไขสมบูรณ์แล้วโดยการส่ง database event และหน้าจอจะถูก refresh ด้วยข้อมูลใหม่ที่แก้ไขแล้วโดยอัตโนมัติ

Other Integrated Features

INGRES/Window4GL ได้รวมเอา SQL ไว้ใน 4GL syntax เพราะฉะนั้นการควบคุม data และการควบคุม process สามารถทำได้โดยใช้ INGRES/Window4GL เพียงภาษาเดียว

INGRES/OpenSQLเป็น subset ของ INGRES/SQL ซึ่งสามารถใช้ในการเข้าถึง data ที่อยู่ในฐานข้อมูลได้

INGRES/Gateway รวมกับ INGRES/OpenSQL จะสามารถทำให้ แอปพลิเคชั่นสามารถเข้าถึงข้อมูลของฐานข้อมูลตัวอื่นๆได้ เช่น DB2 IMS Rdb RMS ALLBASE เป็นต้น

INGRES/Window4GL ยังสามารถจะให้ใช้ 3GL routines ร่วมด้วยได้โดยสามารถมี procedure call ที่มีการส่งผ่านและ return parameter ระหว่าง 4GL กับ 3GL routine ได้

Selected Features

1) Object-Oriented 4GL สามารถที่จะ
   - เข้าถึง window elements และ menus ได้ทุกอัน
   - ควบคุม multi window ได้ใน single application
   - ส่งและรับ massage จาก window อื่นๆ
   - access DB2, IMS, Rdb, RMS, ALLBASE ได้โดยผ่านทาง INGRES/OpenSQL
   - share common class definition ระหว่าง object ได้
   - จัดการ data ใน dynamic array ได้
2) Interactive Debugger ซึ่งจะรวมถึง
   - source level display
   - multiple task tracing
   - ความสามารถที่จะ step ทั้ง toward และ backward ได้ตลอด event queues
   - ความสามารถในการแก้ไข source code และ variable content
   - ความสามารถในการที่จะระบุ breakpoint ได้หลายแห่ง
3) Visual Editors อนุญาติให้ users
   - วาง button, bitmap และ element อื่นๆได้เอง
   - ระบุ dialog box

- ใช้ประโยชน์จาก toolkit elements เช่น text, button, check box, radio box list box ฯลฯ

4.7.3 CASE Tools

บริษัท INGRES ใช้ระบบ Open CASE solution นั่นคือการที่ intelligent database ของบริษัทจะสามารถใช้ได้กับ CASE Tools หลายๆตัวทำให้ผู้ใช้สามารถเลือก CASE Tools ที่เหมาะสมกับงานและรวมทั้งสามารถเลือกได้มากกว่าหนึ่ง CASE Tools ด้วย เพราะฉะนั้นการใช้ INGRES ร่วมกับ INGRES's partner CASE Tools จึงมีข้อดี คือ

1) สามารถที่จะเลือก CASE Tools ที่เหมาะสมกับงานได้มากที่สุด
2) เพิ่มประสิทธิภาพของ CASE Tools ด้วย Intelligent database และ INGRES/4GL features
3) Open CASE solution จะให้โอกาศในการเลือกสิ่งที่ดีที่สุดได้มากกว่า single vendor solution
4) ทำให้มั่นใจได้ว่าการลงทุนระยะยาวบน CASE Tools และ training จะทำกับ product ที่เป็นไปตามมาตราฐานของการพัฒนา CASE เท่านั้นเพราะ INGRES ได้ทำงานอย่าใกล้ชิดร่วมกับ major CASE vendors ในการที่จะพัฒนาและระบุมาตราฐานที่จะให้ CASE products ทุกตัวสามารถที่จะรวมเข้ากับ Industry-standard ของการ access data dictionary ได้

บทที่ 5

การวิเคราะห์ระบบฐานข้อมูลของ GUPTA

เนื้อหาในบทนี้จะกล่าวถึงโครงสร้างทั้งทางด้าน physical และ logical ของระบบจัดการฐานข้อมูล GUPTA พร้อมทั้งแสดงเครื่องมือที่ GUPTA มีจัดเตรียมไว้ให้

5.1 Concurrency Control ของ GUPTA

GUPTA เป็นฐานข้อมูลซึ่งจะยอมให้ผู้ใช้หลายๆคนเข้าใช้ข้อมูลเดียวกันในช่วงเวลาเดียวกัน ซึ่งคำนึงถึง

- concurrency หมายถึง การที่ผู้ใช้ทุกคนสามารถเข้าใช้ข้อมูลใด ๆในช่วงเวลาเดียวกัน
- consistency หมายถึง การที่สามารถแน่ใจได้ว่าการเปลี่ยนแปลงฐานข้อมูลจะทำให้เกิดค่าที่ถูกต้องเหมือนกันและอยู่ในลำดับซึ่งเป็นไปตามโครงสร้างของข้อมูล

ซึ่ง GUPTA ใช้ระบบ locking ในการจัดการการใช้ทรัพยากรซึ่งประกอบไปด้วยข้อมูลในฐานข้อมูล ตารางและ page เพื่อรับประกันความถูกต้องตรงกันของข้อมูลที่มีการใช้ร่วมกัน

GUPTA จะป้องกันไม่ให้เกิดกรณีที่ผู้ใช้คนอื่นๆมาอ่านหรือแก้ไขข้อมูลที่มีผู้ใช้คนใดคนหนึ่งกำลังแก้ไขอยู่ซึ่งจะป้องกันกรณี lost update และจะได้มั่นใจได้ว่า ข้อมูลในฐานข้อมูลจะถูกต้องตรงกัน

เมื่อมีการร่วมกันใช้ข้อมูลในฐานข้อมูลก็จะมีการใช้กลไกการ lock เช่นข้อมูลที่เก็บไว้ใน 1K pages ก็จะมีการ lock ที่ page ของตารางฐานและ page ของ index เป็นต้น -

5.2 Recovery ของ GUPTA

5.2.1 การ recovery แบ่งประเภทได้ดังนี้

1) Crash Recovery

ฐานข้อมูล สามารถถูกทำให้เสียหายมาจากหลายวิธี เช่นจาก power failure หรือ ความผิดพลาดจากผู้ใช้ซึ่งนำไปสู่การdown ของ server เมื่อเกิดเหตุการณ์เหล่านี้ขึ้นนั้น SQLBase ก็พยายามที่จะกู้ฐานข้อมูลกลับมาให้อยู่ในสภาวะที่ consistent โดยการทำ Crash Recovery อย่างอัตโนมัติซึ่ง Crash Recovery จะประกอบด้วยการใช้ Transaction Log เพื่อทำการ redo transaction ใดๆที่ commitไปแล้วที่ยังไม่ได้เขียนลงใน ฐานข้อมูล และทำการ undo transaction ใดๆที่ยังไม่ commit ที่ยังคงทำงานอยู่ในขณะที่ server crash

จะมีกรณีที่ SQLBase ไม่สามารถที่จะกู้ฐานข้อมูลกลับคืนมาสู่สภาวะที่ consistent ได้เช่นในกรณีที่ transaction log ถูกทำลายไปด้วยระหว่างที่เกิด media failure

2) Media Recovery

การบำรุงรักษาเป็นส่วนที่จำเป็นในงานของ Database Administrator และเกี่ยวพันกับการเตรียมตัวสำหรับเหตุการณ์ที่จะเกิดขึ้น เช่น disk head crash, operating system crash หรือผู้ใช้ทำการ drop database object ไปโดยไม่ได้ตั้งใจ ซึ่งเราสามารถที่จะ recover จาก media failure และ user error ได้ถ้ามีการทำ backup ฐานข้อมูล และ log file อย่างสม่ำเสมอ ซึ่งเป็นเพียงวิธีเดียวที่จะสามารถป้องกันการสูญหายของข้อมูล

การ backup จะบ่อยแค่ไหนก็ขึ้นอยู่จำนวนข้อมูลที่ต้องการจะป้องกันซึ่งโดยทั่วๆไปจะ

- backup ฐานข้อมูลสัปดาห์ละครั้ง
- backup transaction log files วันละครั้ง

เราสามารถที่จะลดจำนวนการสูญหายของข้อมูลได้โดยการ backup transaction ให้บ่อยครั้งซึ่งการ backup log ทั้งหมดควรจะทำตั้งแต่ backup ฐานข้อมูลครั้งท้ายสุด เพราะฉะนั้นถ้าเกิด media failure ขึ้นก็สามารถ recover ฐานข้อมูล ได้จนถึงจุดที่มีการ backup log อันสุดท้ายไว้

5.2.2 Recovery Parameters

SQLBase parameter สำหรับการ recovery ได้แก่

1) SET RECOVERY ON/OFF

default จะเป็น ON หมายความว่า SQLBase จะ write log แทรกเข้าไปทั้งค่าของฐานข้อมูลทั้งก่อนการเปลี่ยนแปลงและหลังการเปลี่ยนแปลง (before and after image) และก็จะมี log record สำหรับ transaction control (เช่น checkpoint) และ transaction control อีกส่วนหนึ่งก็คือบาง log record ที่บอกถึงการเริ่มต้นและสิ้นสุด transaction (เช่น COMMIT,ROLLBACK) ซึ่ง transaction log เหล่านี้จะยอมให้เกิด transaction rollback, crash recovery และ media recovery ได้

Recovery ควรจะให้เป็น OFF ถ้าคุณสามารถยอมให้เกิดการสูญหายของข้อมูลได้เท่านั้นในกรณีที่เกิด user error, system crash หรือ media failure ถ้า set recovery เป็น OFF แล้วจะมีบาง option ที่จะถูก reset ตามก็คือ autocommit, bulk execute, isolation level, loadversion, cursor context preservation, restriction, rollback, scroll และ timeout

ก่อนที่จะมีการเปลี่ยน Recovery เป็น OFF ควรจะมีการ backup ฐานข้อมูลไว้

2) SET LOGBACKUP ON/OFF (sqlset ด้วย SQLPLBM parameter)

ถ้าหากว่า media recovery มีความจำเป็นกับ site นั้นมากๆ ก็ควรจะ set LOGBACKUP parameterให้เป็น ON เพื่อระบุว่า log จะถูก backup โดย DBA ก่อนที่ SQLBase จะลบมันทิ้ง

โดย dafault แล้ว LOGBACKUP จะไม่ enable และ SQLBase จะลบ log file ทิ้งในทันทีที่ SQLBase ไม่ต้องการที่จะใช้เพื่อทำ transaction rollback หรือ crash recovery ดังนั้นจะไม่มีการสะสม log files และเก็บไว้ใน disk ดังนั้นถ้า LOGBACKUP เป็น OFF ก็จะไม่สามารถ recovery ฐานข้อมูลได้ในกรณีที่อาจเกิดการเสียหายขึ้นกับฐานข้อมูลได้

LOGBACKUP ต้องเป็น ON เพื่อที่จะทำ BACKUP DATABASE หรือ BACKUP LOGS แต่ไม่จำเป็นต้อง ON ถ้าจะทำ BACKUP SNAPSHOT

โดยปกติ default ของ parameter นี้จะเป็น OFF

3) SET CHECKPOINT (sqlset ด้วย SQLPCTI parameter)

checkpoint time interval parameter นี้จะควบคุมความถี่ของการทำ checkpoint operation ซึ่ง default ของ checkpoint จะทำทุกๆนาที ใน checkpoint operation จะสามารถมีผลกับ performance ได้ซึ่งถ้ามีการเพิ่มช่วงเวลาของ checkpoint ก็อาจจะทำให้ performance ดีขึ้นได้ให้จำไว้ว่าถ้าช่วงเวลาของ checkpoint

ยาวขึ้นก็จะทำให้ผลกระทบต่อ performance ลดลงแต่จะทำให้เวลาในการทำ crash recovery เพิ่มขึ้น

5.2.3 การ Backup Database

มี 2 วิธีคือ online และ offline

1) Online Backup เป็นการ copy ฐานข้อมูล (.dbs)file และ log (.log) file ที่ทำกับSQLTalk BACKUP command หรือ API function ในขณะที่กำลังรัน server program อยู่

ข้อดีของการทำ online backup คือ ผู้ใช้สามารถเข้าถึงฐานข้อมูลในขณะที่กำลังทำการ backup อยู่ ซึ่งจะเป็นประโยชน์อย่างมากกับ site ที่ต้องใช้ ฐานข้อมูล ทั้งวันตลอด 24 ชั่วโมง

option ของ online backup ได้แก่

- BACKUP SNAPSHOT (sqlbss)

เป็นการbackup เฉพาะ database file และ log file ที่ถูกต้องการเอาไว้สำหรับ restore ฐานข้อมูลเพื่อที่จะให้กลับเข้าสู่สภาวะ consistent ซึ่งจะรวมถึง log file ที่ active อยู่ในปัจจุบัน คำสั่งนี้เป็น BACKUP command อันเดียวที่ไม่ต้อง set ให้ LOGBACKUP เป็น ON ถ้า LOGBACKUP เป็น ON แล้ว log file ที่ถูกทิ้งไว้ใน directory ของ ฐานข้อมูล ก็จะถูก backup โดย BACKUP LOGS command แล้ว SQLBase ก็จะ ลบ log files เหล่านี้ทิ้งโดยอัตโนมัติ

- BACKUP DATABASE (sqlbdb)

เป็นการ backup data file ซึ่งไม่ควรที่จะ backup ฐานข้อมูล โดยไม่มีการ backup log file ไปด้วย

- BACKUP LOGS (sqlblf)

เป็นการ backup logfile แล้วก็จะลบมันทิ้ง

BACKUP SNAPSHOT จะเป็นวิธีที่ดีที่จะ backup ฐานข้อมูล และ log file เนื่องจากง่ายและเป็นการ backup ที่จะสามารถทำการ recover กลับมาได้ใน step เดียว

BACKUP DATABASE และ BACKUP LOGS จะมีไว้สำหรับ site ต่างๆ ที่มี ฐานข้อมูลใหญ่มากที่ต้องการเพิ่มจำนวนการ backup เช่นต้องการ backup ฐานข้อมูล และ logs ทุกๆวันอาทิตย์ในขณะที่จันทร์ถึงเสาร์จะ backup เฉพาะ logs

2) Offline Backup ข้อดีของ offline backup คือ สามารถที่จะ backup file ได้โดยตรงไปยัง archival media โดยการก่อนที่จะทำการ offline backup จะต้อง shut

down ตัว server ก่อน

เราสามารถทำ offline backup ได้โดยวิธีของ operating system command หรือใช้ utility อื่นๆเช่น โดยการใช้คำสั่ง COPY

5.2.4 การ Restore database

1) การ Restore จาก Online Backup

ผู้ใช้จะไม่สามารถเชื่อมต่อกับฐานข้อมูลได้ในขณะที่ทำการ restoreและ recovery โดยจะทำการ deinstall multi-user ฐานข้อมูล โดย SQLTalk ด้วยคำสั่ง DEINSTALL DATABASE หรือ API sqlded function แล้วทำการ restore และ rollforward หลังจากนั้นก็ทำการ install ฐานข้อมูลด้วย SQLTalk โดยคำสั่ง INSTALL DATABASE หรือ API sqlind function

ถ้าฐานข้อมูลเสียหายก็สามารถที่จะ restore ฐานข้อมูล ที่ backup ไว้ด้วยคำสั่ง SQLTalk RESTORE หลังจากทำการ execute RESTORE SNAPSHOT หรือ sqlrss command แล้วก็ไม่ต้องทำอะไรอีกเพราะคำสั่งนี้จะcopy ทั้ง database file ที่ backup ไว้รวมถึง log files ที่ backup ไว้ด้วย

ถ้าไม่ได้ทำการ backup ด้วย BACKUP SNAPSHOT หรือทำ BACKUP SNAPSHOT
และต้องการที่จะ rollforward จากจุดนั้นไปให้มากที่สุดเท่าที่จะเป็นไปได้ก็สามารถใช้ RESTORE DATABASE command หรือเรียก sqlrdb API function ใน program และเพื่อที่จะ rollforward การเปลี่ยนแปลงที่ถูกกระทำไว้หลังจากการ backup ฐานข้อมูล และจะทำให้ฐานข้อมูลนั้น up-to-date ที่สุดก็ให้ใช้คำสั่ง ROLLFORWARD หรือ sqlrof API function ดังนี้

- ROLLFORWARD TO END เป็นการ rollforward ตลอดทั้ง log file ทั้งหมด (เป็น default) วิธีนี้จะ recover ได้มากที่สุดเท่าที่จะเป็นไปได้
- ROLLFORWARD TO BACKUP เป็นการ rollforward ไปจนถึงสุดของการ backup วิธีนี้จะ recover งานที่ commit แล้วทั้งหมดจนกระทั่งถึงจุดที่ database backup ไว้สมบูรณ์ซึ่งจะเทียบเท่ากับการทำ RESTORE SNAPSHOT
- ROLLFORWARD TO TIME เป็นการ rollforward ไปจนถึงวันและเวลาที่กำหนดไว้

จะต้องมีการทำ backup ของ log files ของ ฐานข้อมูล ทุกอัน และต้องเรียงเป็นลำดับ ถ้าเกิดว่ามี log file อันใดอันนึ่งหายไป จะทำให้ไม่สามารถดำเนินการ

rollforward ต่อไปจากจุดที่ log file หายไปได้ เช่นถ้ามี 1.log, 2.log, 4.log, 5.log ส่วน 3.log หายไปแล้วก็จะทำให้สามารถกู้ ฐานข้อมูล กลับมาได้ถึงแค่ 2.log เท่านั้น

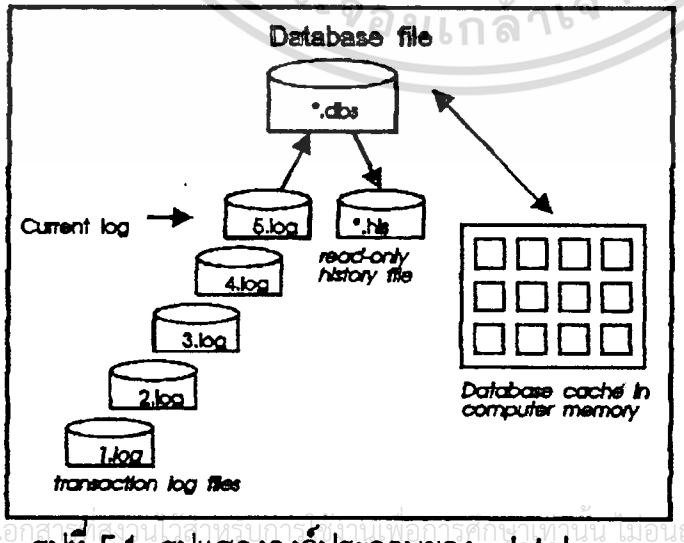
2) การ Restore จาก Offline Backup

การ recover offline backup ได้นั้นจะต้องทำจากวิธีใดวิธีหนึ่งต่อไปนี้คือ

- ถ้าการ backup ประกอบด้วย database file เท่านั้นแล้ว ก็จะ restore ได้โดยการ copy ทับลงไปบน database file ที่เสียหายไป และจะต้องแน่ใจด้วยว่า file ที่ copy กลับมาต้องเป็น .dbs แล้วก็ทำการเชื่อมต่อเข้ากับฐานข้อมูลในวิธีนี้จะไม สามารถกู้การเปลี่ยนแปลงที่เกิดขึ้นหลังจากจุดที่ทำ offline backup ได้

- ถ้าการ backup ประกอบด้วย database file และ log file ตั้งแต่ 1 อันขึ้นไปแล้ว ให้ใช้ SQLTalk RESTORE DATABASE command หรือ sqlrdb API function เพื่อที่จะ restore ฐานข้อมูล แล้วก็ใช้ ROLLFORWARD command เพื่อที่จะจัดการกับ log ทำให้ฐานข้อมูลที่กู้กลับมานั้น up-to-date ที่สุดโดย RESTORE command จะ copy backup ไปที่ database subdirectory และ ROLLFORWAED ก็จะกัดการกับการเปลี่ยนแปลงที่ commit แล้วตั้งแต่จุดที่ทำ offlinebackup ไว้ ถ้าหากว่า SQLBase ไม่สามารถหา log file ที่จะทำการ rollforward ได้ก็สามารถที่จะ restore log file ได้ด้วยการใช้ RESTORE LOGS command หรือไม่ก็ใช้ copy command หรือ utility ต่างๆ แล้วก็ใช้ ROLLFORWARD CONTINUE command โดยตรงหรือไม่ก็เรียก sqlcrf API function เพื่อที่จะจัดการกับ log file

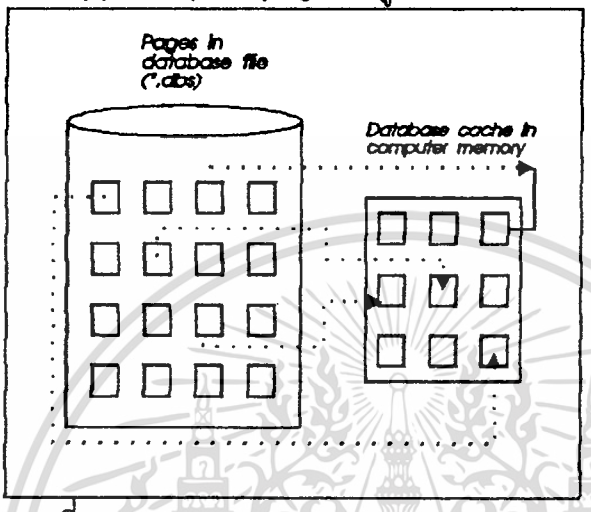
5.2.5 Diagram แสดงองค์ประกอบของ Database



รูปที่ 5.1 รูปแสดงองค์ประกอบของ database

Database cache

SQLBase ใช้ cache เพื่อที่จะ optimize database input และ output ซึ่งส่วนของ cache นี้จะอยู่ในเนื้อที่ของ หน่วยความจำ ของคอมพิวเตอร์ที่เป็น server ซึ่งประกอบด้วย copy ต่างๆของ page ที่ ผู้ใช้ จะ read หรือ write



รูปที่ 5.2 แสดง database cache

1 page = 1024 byte unit ของ ข้อมูล จาก database file ที่ประกอบด้วย rows ของ table data หรือ index data ตั้งแต่ 1 row ขึ้นไป

เมื่อ ผู้ใช้ read หรือ write row หรือ index หนึ่งๆนั้น SQLBase จะตรวจสอบดูว่า page นั้นอยู่ใน cache แล้วหรือยัง ถ้ายัง SQLBase ก็จะทำการ copy จากฐานข้อมูล เข้ามาอยู่ใน cache หรือถ้าอยู่ใน cache อยู่แล้วก็จะใช้ copy ใน cache นั่นเลย ซึ่งการใช้ cache จะทำให้ลดการใช้ disk I/O และก็ยังทำให้ read หรือ write กับหน่วยความจำของคอมพิวเตอร์ได้เร็วกว่า read หรือ write จาก disk file

เมื่อมีการ commit แล้ว SQLBase จะทำการ write commit record นั้นไปยัง transaction log บน disk อย่างไรก็ตาม database page ใน cache ก็จะถูกเขียนกลับลงไปใน.dbs file ตาม LRU algorithm ซึ่งข้อมูลใน transaction log นั้นจะพอเพียงสำหรับการสร้างฐานข้อมูลขึ้นมาใหม่ในกรณีที่อาจเกิดการ crash ดังนั้นจึงไม่จำเป็นที่จะต้องดึง page ใน cache กลับลงไปใน data file ทันทีหลังจาก commit

5.2.6 Checkpoint Time Interval

Gupta จะเรียกว่าเป็น Fuzzy Checkpointing คือจะเป็นวิธีที่ pages ของ cache ในฐานข้อมูลที่ถูกแก้ไขจะถูก mark เอาไว้ทันทีที่ checkpoint หนึ่งและก็จะถูกเขียนใน checkpoint ต่อไป ถ้า page นั้นยังทำการ แก้ไข ไม่เสร็จหรือพูดง่ายๆก็คือ transaction log record จะเก็บ active transaction ที่แต่ละ checkpoint ไว้โดย .

checkpoint จะถูกใช้เพื่อเป็นจุดที่ใช้เริ่มต้นทำการ redo สำหรับ rollforward phase ของการทำ crash recovery

พารามิเตอร์ที่เอาไว้กำหนดช่วงระยะเวลาของ checkpoint จะบอกเอาไว้ว่าบ่อยครั้งแค่ไหนที่จะทำ checkpoint operation ซึ่งสามารถ set ช่วงเวลาได้จาก SQLTalk โดยใช้ SET CHECKPOINT command หรือใช้ sqlset API function ซึ่งตาม default นั้นจะทำ checkpoint ทุกๆนาที ซึ่งช่วงระยะเวลาที่เหมาะสมนั้นก็ขึ้นอยู่กับแอปพลิเคชั่นที่รันบน server เนื่องจาก checkpoint operation นี้จะมีผลกับ performance ของระบบด้วย เพราะฉะนั้นก็อาจจะสามารถเพิ่มช่วงเวลา checkpoint ได้จนกว่าจะได้ performance ที่ดีตามที่ต้องการ

Transaction Log Files

Transaction Log File จะเก็บค่าของทั้งก่อนการเปลี่ยนแปลงและหลังการเปลี่ยนแปลงในฐานข้อมูล (before and after image) เช่นเดียวกับ log record สำหรับ transaction control (เช่น checkpoint) ซึ่ง transaction control ก็จะรวมถึงข้อมูลที่บอกว่า transaction เริ่มต้นเมื่อไรและสิ้นสุดเมื่อไร (เช่น COMMITและROLLBACK)

Transaction Log File จะมีหน้าที่ 3 ประการคือ

1) Rollback

Transaction Log File จะมีข้อมูลที่จำเป็นสำหรับการrollback transaction ซึ่งการ rollback สามารถทำได้โดยจากทั้ง SQLBase หรือไม่ก็ ผู้ใช้

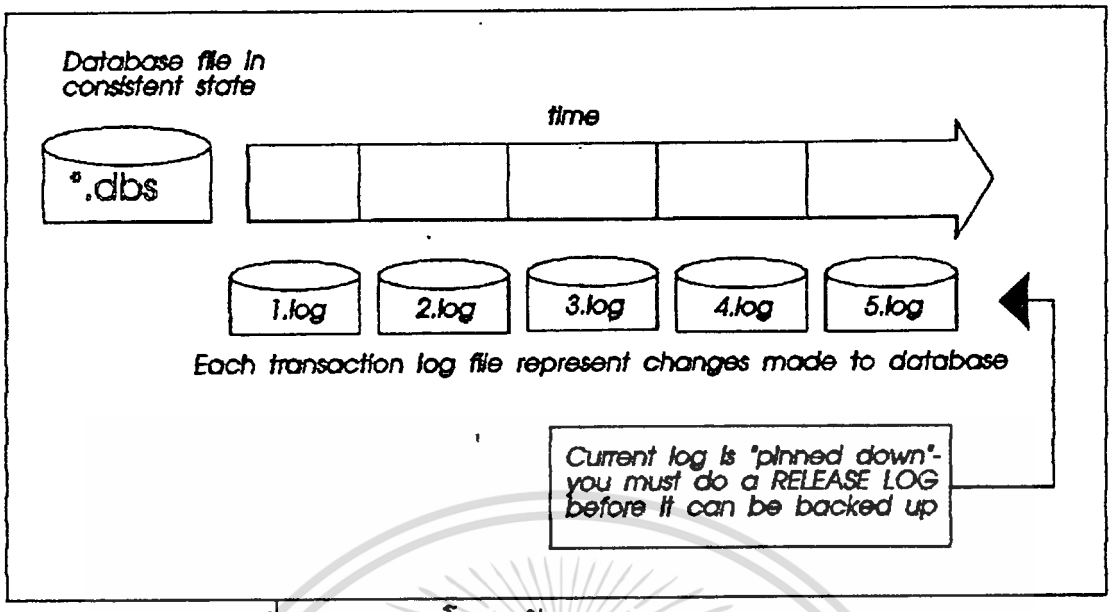
2) Crash Recovery

Transaction Log File จะมีข้อมูลที่จำเป็นสำหรับการนำฐานข้อมูลกลับมาสู่สภาวะที่ consistent หลังเกิดการ crash ซึ่งการ crash เกิดขึ้นได้จากผลของ power failure หรือ operator error โดยฐานข้อมูลจะ crash ถ้าหากว่าเกิดการ shutdown ที่ไม่เหมาะสม เช่นเมื่อ server computer ถูก boot ใหม่หรือถูกปิดโดยไม่ใช้ Escape key เพื่อที่จะหยุดโปรแกรมของ server ก่อน การทำ crash recovery นี้จะทำอย่างอัตโนมัติเมื่อไรก็ตามที่ผู้ใช้เชื่อมต่ออยู่กับฐานข้อมูลที่ถูกทำให้ crash ซึ่งก็ต้องทำแบบ *online*

3) Media Recovery

Transaction Log File จะเก็บในรูปของการ backup ไปกับ ฐานข้อมูล ซึ่งจะมีข้อมูลที่ต้องใช้ในการ restore ฐานข้อมูลนั้น ถ้าเกิดความเสียหายจาก media failure ซึ่ง media failure จะเกิดได้จาก hard drive crash หรือปัญหาต่างๆจากทางด้าน hardware

Log Files



รูปที่ 5.3 แสดงโครงสร้างของ log file

SQLBase จะสร้าง log file อันใหม่เมื่อ log file อันปัจจุบันเต็มถึงขนาดเต็มที่ที่กำหนดไว้

Log file อันแรกจะได้ชื่อว่า 1.log แล้วอันต่อๆไปก็จะเป็น 2.log, 3.log, ... ไปเรื่อยๆจนถึง log file อันมากสุดคือ 99999999 โครงสร้างภายในของ log file จะมี timestamp และค่าอื่นๆซึ่ง SQLBase ใช้เพื่อระบุลำดับ ซึ่งลำดับของ log file นี้จะไม่สามารถเปลี่ยนชื่อหรือเปลี่ยนหมายเลขได้

SQLBase จะใช้ Write Ahead Log (WAL) Protocol เพื่อที่จะทำให้แน่ใจว่า log record สำหรับ page ที่ถูกแก้ไขจะถูกเขียนก่อนที่ page ที่ถูกแก้ไขใน database cache จะถูกเขียนกลับลง disk ซึ่งจะทำให้แน่ใจได้ว่า SQLBase สามารถที่จะ undo การเปลี่ยนแปลงที่ยังไม่ commit ใดๆได้ในกรณีที่เกิดการ crash

5.3 Data Integrity ของ GUPTA

หมายถึงความถูกต้องของข้อมูลที่เก็บไว้ในฐานข้อมูล โดยจะควบคุมความถูกต้องโดยใช้คำสั่ง "CHECK DATABASE" เพื่อตรวจสอบความถูกต้องของฐานข้อมูลดังต่อไปนี้

- ตรวจสอบความถูกต้องของข้อมูลที่เก็บไว้ในฐานข้อมูล table free space และโครงสร้างข้อมูล
- ตรวจสอบ row แต่ละ row ในตารางรวมทั้งนับจำนวน row เหล่านั้นด้วย
- ตรวจสอบ index แต่ละ index ของตารางฐาน
- ตรวจสอบความถูกต้องของ page ของ row และ index แต่ละ page
- เพื่อให้แน่ใจได้ว่า page แต่ละ page ที่อยู่ในฐานข้อมูลจะต้องอยู่ในส่วนที่เป็น page ที่ใช้งานแล้วหรือไม่ก็อยู่ใน list ของ free page

การใช้คำสั่ง SQL > CHECK DATABASE

ในกรณีที่คำสั่ง CHECK DATABASE พบกับปัญหาก็ควรจะแก้ปัญหาโดย

1. restore ฐานข้อมูลโดยใช้ backed-up copy ของฐานข้อมูล
2. ถ้าคำสั่ง CHECK DATABASE แสดงข้อความว่า object ของฐานข้อมูลถูกทำลายไปแล้วก็จำเป็นต้อง drop object นั้น

5.4 Optimizer ของ GUPTA

ใน SQLBase นั้นสามารถเรียกข้อมูลที่ต้องการดูขึ้นมาโดยผ่าน SQL command ได้และ SQLBase ก็จะพิจารณาว่าข้อมูลควรจะถูกเข้าถึงอย่างไรโดยใช้ optimizer ซึ่งใน SQLBase นี้จะเลือกเส้นทางการเข้าถึงโดยพิจารณาจาก index ที่มี catalog ที่เก็บสถิติและองค์ประกอบของ SQL command

ทางเลือกขั้นมูลฐานเช่น

1)ใช้การเข้าถึงแบบ index โดยไม่ต้องอ่านตารางของข้อมูลเลยกรณีนี้จะเป็นการเข้าถึงที่มีประสิทธิภาพที่สุดถ้าข้อมูลที่ต้องการอยู่ใน index ทั้งหมด

2) ใช้การเข้าถึงแบบ index และอ่านตารางของข้อมูลกรณีนี้ตัวกำหนดคุณสมบัติของคำสั่งจะถูก match เข้ากับ index และจะมีเพียง row ที่ตรงคุณสมบัติเท่านั้นที่ถูกอ่านจากตารางซึ่งกรณีนี้ก็คือ SQLBase จะใช้ index แม้ว่าข้อมูลใน index จะไม่ได้ match กับข้อมูลที่ต้องการเรียกซึ่งระบุไว้ในตัวกำหนดคุณสมบัติในคำสั่ง

3) ใช้การ scan table วิธีนี้ก็คือทุก page และทุก row จะถูกอ่าน

นอกจาก option ที่กล่าวมาแล้วก็ยังมีทางเลือกอื่นๆอีกซึ่งถ้า query ที่เกี่ยวข้องกับ table หลายๆอัน การ process ก็จะซับซ้อนขึ้นและจะเกี่ยวพันกับการ sort ภายในและการสร้าง intermediate result tables ซึ่งจะ transparent ต่อ ผู้ใช้

5.5 Gupta Tools

The Gupta SQL system

Gupta เป็นระบบจัดการฐานข้อมูลที่เหมาะสมสำหรับระบบ Client/Server บน PC LANs ประกอบด้วย

1) SQLTalk เป็นเครื่องมือทาง database administration

2) Gupta Quest เป็นการใช้กราฟฟิคเข้าช่วยในการเข้าถึงข้อมูลและออกรายงานสำหรับฐานข้อมูล SQL

3) Gupta SQLWindows เป็น application development tools บน window ของระบบ C/S

สามารถเขียนโปรแกรมแบบโดยใช้ object oriented ได้

4) Gupta SQLBase เป็น SQL DBMS

5) Gupta SQL network เป็น connectivity software

ในส่วนต่อไปจะเป็นการพูดถึงในรายละเอียดเกี่ยวกับ product ต่างๆ

5.5.1 SQLTalk

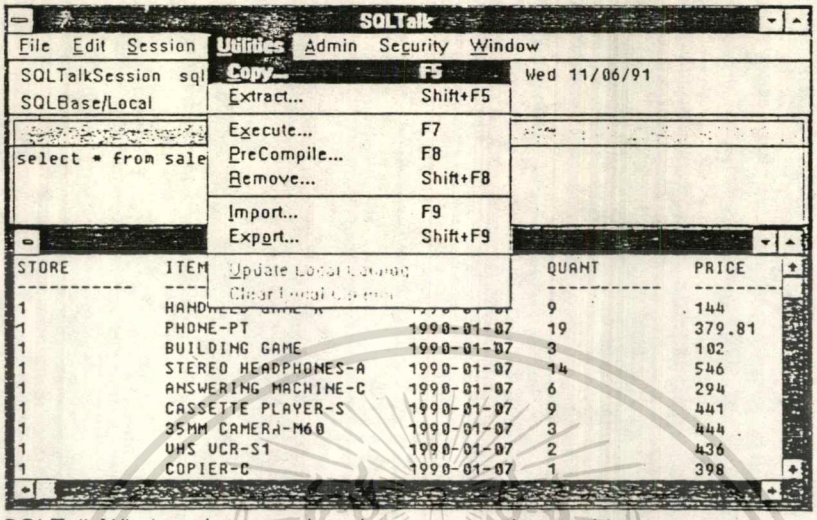
เป็น data manager สำหรับ SQLBase Server และสำหรับฐานข้อมูลที่ Gupta's SQLNetwork support อยู่ SQLTalk จะรับ query (เป็นภาษา SQL) ที่ input window และทำการแสดงผลของ query นั้นออกที่ output window

SQLTalk มี interface 2 แบบด้วยกันคือ

1) SQLTalk/Window เป็น graphical interface ใช้บน micresoft windows

2) SQLTalk/Character เป็น interface แบบ Dos command line

SQLTalk/Windows



SQLTalk/Windows is a complete data manager that enables you to create, query and administer a SQL database.

ทำงานได้โดยใช้การ point-and-click list ของ ฐานข้อมูล,ตาราง และ ผู้ใช้ จะดูได้จาก menus และ dialogs ใน SQLTalk จะประกอบด้วย window 2 windows คือ

- input window สำหรับการ edit SQL statement
- output window สำหรับการแสดงผลของ query

ดังแสดงในรูป

เราสามารถที่จะดูผลของ query ได้ใน farmat หลายๆแบบเช่น SQL SACII DIFF DBF WKS และ raw data และ format เหล่านี้สามารถที่จะถูก import ลงในฐานข้อมูลได้และเนื่องจาก Report/Window สามารถที่จะรับ input จาก source data ใดก็ได้ data file ที่สร้างจาก SQLTalk จึงสามารถที่จะถูก fed ไปที่ Report/Window เพื่อสร้าง report ได้

5.5.2 SQL Quest

เป็น tool ที่ใช้ในการเข้าถึงข้อมูลสำหรับผู้ใช้การ interface กับผู้ใช้ ใช้การ point-and-click บน window ในการใช้ Quest จึงไม่ต้องการความรู้ทาง SQL

Quest ประกอบด้วย

1) Query activity

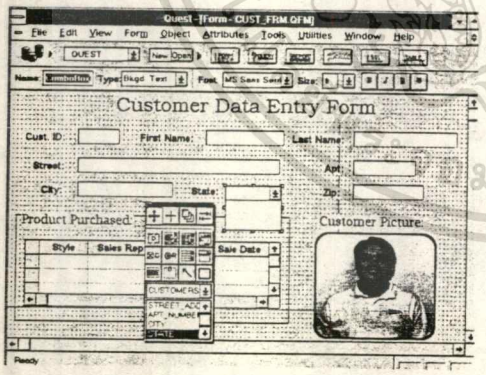
เป็นการเข้าถึงข้อมูลใน SQL database การใช้ Quest ทำให้ผู้ใช้สามารถจะสร้าง database query โดยการ point และการ click เมื่อผู้ใช้ทำการสร้าง query แต่ละ query Quest จะสร้าง SQL code ที่ถูกต้องให้ตัว SQL code นี้จะถูกเก็บซ่อนไว้ไม่แสดงต่อผู้ใช้แต่ถ้าผู้ใช้ต้องการดูก็สามารถที่จะเรียกดูได้ Quest จะ provide การ link ระหว่างตารางที่มีความสัมพันธ์กันได้โดยอัตโนมัติ และหลังจากที่ query ถูก define ขึ้นมาจะสามารถถูกเก็บไว้เพื่อใช้ต่อไปได้

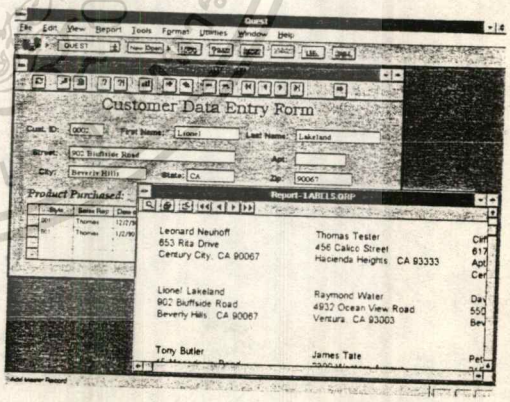
Administration ของระบบสามารถที่จะใช้ Quest ในการสร้าง join formula และ query conditions อื่นๆลงใน query template ได้เพื่อให้ผู้ใช้คนอื่นๆสามารถใช้ template เหล่านี้สำหรับการสร้าง query ของตัวเองได้

2) Form activity

เป็นการเข้าถึงและการจัดการ SQL data โดยผ่าน Form Form activity สามารถที่จะสร้าง form-based data management interface ได้โดยอัตโนมัติ การใช้ Quest From

ผู้ใช้สามารถใช้ data interface ในแบบที่คุ้นเคยและไม่ซับซ้อนได้ผู้ใช้สามารถที่จะ query database ได้โดยใช้ Form และสร้าง form-based report และ data management interface โดยไม่ต้องใช้การเขียนโปรแกรมใดๆเลยเช่น การสร้าง data entry form





การสร้าง data management form จาก SQL Table หรือ view ทำได้โดยใช้ point-and-click painting-tools menus และ dialog boxes user อาจจะสร้าง report โดยใช้ Quest-Form interface หรือ link Form เข้ากับ Quest report ก็ได้

3) Report activity

เป็นการสร้างรายงานที่สามารถเข้าถึงข้มูล SQL ใช้ในการออกแบบและการสร้างรายงานจากแหล่งข้อมูลใดๆก็ได้ที่ Quest support user สามารถที่จะเลือก fonts

graphics และ formatting option อื่นๆได้ ตัวรายงานสามารถที่จะถูก pre-view บนหน้าจอหลังจากเป็นที่พอใจแล้วจึงส่ง output ไปที่ printer

Quest provide reporting function ต่างๆเช่น function ทางสถิติและรวมทั้ง provide formular editor ที่ซับซ้อน (มีกว่า 70 functions) ทำให้ผู้ใช้สามารถควบคุมรูปแบบของ output และการคำนวนต่างๆได้ในหลายระดับของความซับซ้อน

รายงานที่สร้างจาก Quest-report นี้สามารถที่จะถูกรวมเข้าไปในแอปพลิเคชั่นที่สร้างโดย SQLWindows ได้

4) SQL activity

เป็นการ query หรือการใช้คำสั่งเกี่ยวกับ data management โดยใช้ SQL โดยตรงสำหรับผู้ใช้ที่คุ้นเคยกับภาษาSQL อยู่แล้วสามารถที่จะ by pass ส่วน point-and-click มาใช้ภาษา SQL โดยตรงได้เลยอย่างไรก็ตามผลที่ได้จากการ query ยังคงสามารถที่จะถูกส่งผ่านไปยัง Quest-report ได้เช่นกัน

5) Table activity

ใช้ในการสร้าง การ browse และการ edit table ผู้ใช้สามารถใช้ table activity สร้างตารางใหม่จากตาราง หรือ view ที่มีอยู่แล้วก็ได้ และเพื่อการเพิ่ม security และ data integrity Table activity จะถูก install โดยให้มี write privilege กับ local database แต่เป็น read-only privilege สำหรับฐานข้อมูลอื่นๆบน network

6) Catalog activity

ใช้ในการสร้างการ browse และการ modify database Catalog activity จะจัดการเกี่ยวกับ data definitions ใน database user สามารถที่จะ browse ดูได้ว่าแต่ละ table view index ถูกระบุอย่างไรทำให้รู้ถึงสิทธิ์ของตนในการเข้าถึงข้อมูลต่างๆและเช่นเดียวกันกับ Table activity Catalog activity จะถูก install โดยให้มี write privilege สำหรับ local data แต่เป็น read-only privilege สำหรับฐานข้อมูลในระยะไกล

Quest สนับสนุน Dynamic Data EXchange (DDE) และ Object Linking and Embedding (OLE) จึงทำให้ Quest สามารถจะ share และแลกเปลี่ยนข้อมูลกับ window programs อื่นๆได้เช่น Quest สามารถใช้ในการเข้าถึงข้อมูล บน DB2 database และส่งข้อมูลนั้นไปให้เครื่องมือต่างๆ เช่น Excel ได้รวมทั้งการใช้ OLE ทำให้ผู้ใช้สามารถ insert chart ของ excel ลงใน Quest report ได้อีกด้วย

5.5.3 SQLWindows (Graphical C/S Application Development System)

SQLWindow เป็น development system สำหรับการสร้าง mutiuser C/S database application สำหรับ windows window NT และ OS/2 workstations

ลักษณะของ SQLWindows คือ

- เหมาะสำหรับงานการเขียนโปรแกรมที่ต้องการการทำร่วมกันเป็นทีมสามารถที่จะเพิ่ม productivity ได้เนื่องจาก blocks ต่างๆในแอปพลิเคชั่นที่สร้างไว้สามารถนำมาใช้ใหม่ได้

- มีการใช้โปรแกรมแบบ object-oriented ที่แต่ละ object สามารถนำมาใช้ใหม่ได้
- สามารถสร้างแอปพลิเคชั่นได้โดยไม่ต้องรู้จัก SQL และแทบไม่ต้องเขียน code เลยโดยการใช้

QuestWindows

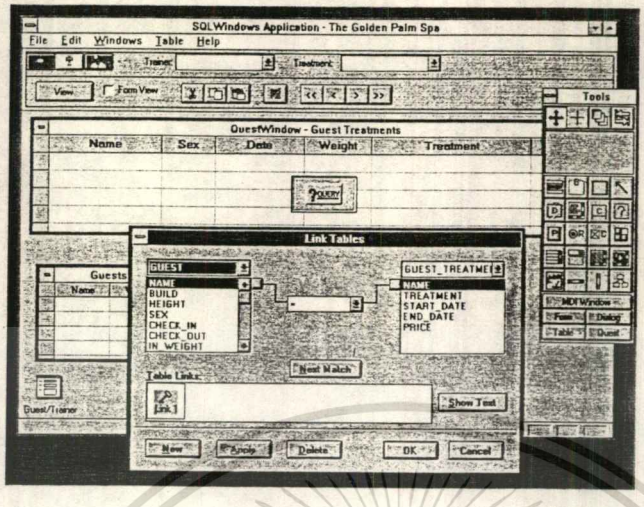
ส่วนประกอบต่างๆของ SQLWindows มีดังต่อไปนี้

1) Application Designer

ใช้ในการออกแบบหน้าจอ forms และ objects ในการเพิ่มหรือ modify objects ต่างๆใน Application Designer programmer สามารถใช้วิธี point-and-click และ drag-and-drop ได้ SQLWindow จะสร้าง code และ documentation ให้โดยอัตโนมัติ object ใน application designer รวมถึงพวก image sound และ vedio สำหรับการสร้าง multimedia applications ด้วย

Application Designer ประกอบด้วย

1) Palette ของ painting และ customizing tools
2) Context-Sensitive Help
3) QuestWindow เป็น graphical data access tools ใช้สำหรับการสร้าง application โดยไม่ต้องเขียน code โดยผู้เขียน program สามารถที่จะรวมเอา QuestWindow เช่น browse และ edit table หรือ form-window object ลงใน application ได้เลยและ code จะถูกสร้างเองโดยอัตโนมัติ QuestWindow มีลักษณะดังรูป



4) TableWindow จะจัดการให้ data retrival, display และ updating มีลักษณะเป็นตารางโดยอัตโนมัติ ตัว TableWindow จะถูกจัดการโดย high-level function และจะช่วยลด programing effort ได้ ตัวอย่าง function เช่น การ load TableWindow ด้วย ข้อมูลจากฐานข้อมูล หรือการจัดการการ update delete และ insert เป็นต้น

2) Application Outliner

จะแสดง overall view ของส่วนประกอบต่างๆของ application เมื่อมี application object ถูกสร้างหรือถูกแก้ไขใน Application designer ตัว Application Outliner ก็จะถูกแก้ไขตามโดยอัตโนมัติ Outliner นี้จะทำให้การทำ document และการทำ navigate แอปพลิเคชั่นขนาดใหญ่สามารถทำได้สะดวกขึ้น

3) SQLWindow Application Language(SAL)

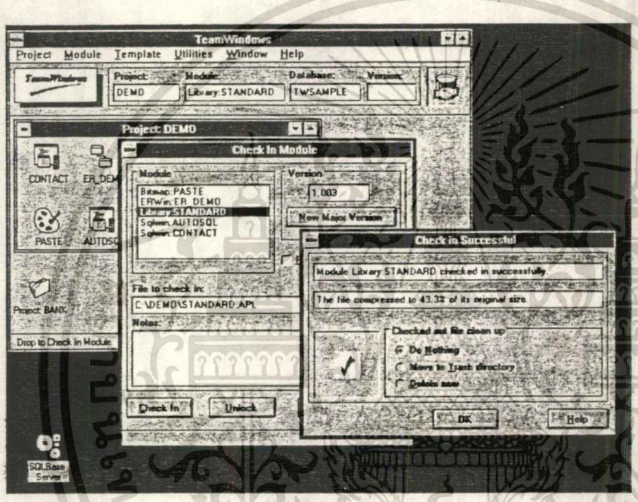
SAL เป็นการ programming แบบ 4GL ซึ่งจะ support

1) 500 SALWindows และ Microsoft Windows function
2) ความสามารถในการสร้างและ share function ใหม่ๆ
3) ทุกๆ Back-End datatype
4) Variables และ Arrays
5) Control Flow
6) การ access ไปยัง externally developed code เช่น C,C++

4) TeamWindows

เป็น Data Dictionary_driven repository สำหรับ ข้อมูลและ reusable application block มี Repository-based tools เพื่อสนับสนุนการเขียนโปรแกรมที่ทำร่วมกันเป็น ทีม

TeamWindows ทำให้เกิดการ sharing overuse components และจะทำ administrative task ต่างๆเช่น security,standard enforcement ฯลฯ รวมทั้ง TeamWindows จะสร้าง up-todate report บนสถานะของทุกๆ phase ของ application project/ลักษณะของ TeamWindows



TeamWindows สามารถที่จะ

1) สร้าง Data Dictionary-Driven Application โดยตัว TeamWindows จะบำรุงรักษา data dictionary กลางของข้อมูลที่เกี่ยวกับโครงสร้างของฐานข้อมูล เช่นพวกตาราง, ชื่อ column,primary และ foreign key,ความสัมพันธ์และพารามิเตอร์ อื่นๆ ซึ่งข้อมูลเกี่ยวกับโครงสร้างเหล่านี้สามารถที่จะถูก import มาจาก CASE tools ที่มีอยู่แล้วได้

2) Modify data dictionary ทุกๆครั้งที่มีการเปลี่ยนแปลงของโครงสร้างของฐานข้อมูล รวมทั้งแก้ไขทุกๆแอปพลิเคชั่นที่เกี่ยวข้องด้วย

3) ควบคุม security ของ source code และ version เมื่อ SQLWindows ถูกพัฒนาสมบูรณ์แล้ว มันจะถูกเก็บใน repository และถูกจัดการโดย TeamWindows source code controller มี check-in และ check-out facility ซึ่งทำให้สมาชิกของกลุ่มสามารถที่จะ share และ modify components ของ application ได้โดยที่ repository security จะคอยปฏิเสธการ เข้าถึง ที่ไม่ถูกต้องของ ผู้ใช้ และทำให้แน่ใจได้ว่าในเวลาใด

เวลาหนึ่งจะมี programmer 1 คนเท่านั้นที่จะสามารถ modify component ใดๆ นอกจากนั้น TeamWindows จะคอยตามและ updates version numbers ของ application modules ตลอดวัฏจักรของการพัฒนา

4) สร้างและดูแลรักษา Template Libraries, TeamWindows จะจัดหา pre-defined templates สำหรับการสร้าง SQLWindows screen และ application โดยใช้ data dictionary information ผู้เขียน program จะสามารถ save และ reuse template เหล่านั้นได้

5) Gupta Quest

ทำให้ ผู้ใช้ สามารถที่จะสร้าง ad hoc report และรวม report นั้นเข้ากับ SQLWindows application ของงานได้อย่างไม่ซับซ้อน รวมทั้ง ผู้ใช้ ยังสามารถใช้ Quest ในการสร้าง form,dialog และ database interfaces อื่นๆสำรับ application ได้อีกด้วย

6) ReportWindows

เป็น grapgical report designer&writer ซึ่งจะมี fonts ,สี และ formatting options อื่นๆให้เลือก Report ที่สร้างจาก Report windows นี้จะ compatible กับ report ที่สร้างจาก Quest

7) GUPTA SQLBase Server for windows

เป็น single user database ที่ทำให้สามารถทดสอบ application และ debug บน stand-alone PC หรือแม้แต่บน laptops

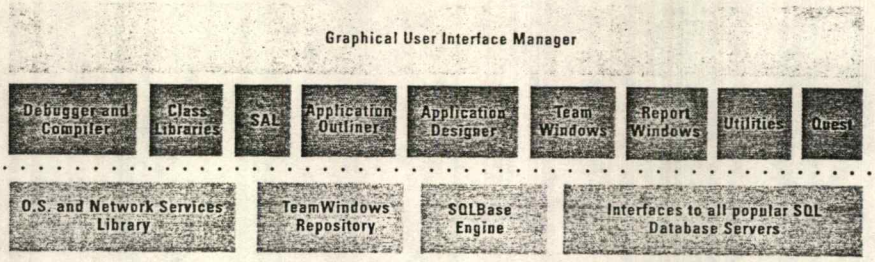
8) Debugger and Compiler

Debugger ใช้ทำการทดสอบโปรแกรมเป็นแบบ multi break point, single stepping, watch values และ code animation

Compiler ใช้ในการ compile code ให้เป็น executable code

Extensive Integration และInterroperability

support Dynamic Data Exchange (DDE), Object linking and Embedding (OLE), Multiple Document Interface (MDI) และ external Dynamic Link libraries (BLL) ทั้งใน Application Designer และ SAL นอกจากนั้น windows message call ต่างๆ ยังสามารถใช้ได้ใน SQL Windows



SQLWindows comprises a complete set of integrated, object-oriented tools and services for developing, debugging and deploying Windows-based client/server applications.

ภาพแสดงส่วนประกอบต่างๆของ SQL Windows

5.5.4 SQL Base

เป็น SQL DBMS มีทั้งสำหรับ DOS, WIndows, OS/2, Netware(NLM) และ SUN UNIX

5.5.5 SQL Network

เป็น connectivity software ระหว่าง graphical PC (Client) กับ LAN database และ SQL database SQL Network นี้จะ support IBM DB2, OS/2 Database Manager และ AS/400, ORACLE, GUPTA SQL Base Server, Microsoft and Sybase SQL server, Informix, Cincom Supra และ HP ALLBASE/SQL

บทที่ 6

กฎของ E.F. Codd ในการพิจารณาเลือกระบบฐานข้อมูล

6.1 กฎ 12 ข้อของ Codd ในการวิเคราะห์ระบบจัดการฐานข้อมูล

E.F. Codd ได้เสนอกฎเกณฑ์ไว้ทั้งหมด 12 ข้อเพื่อใช้เป็นมาตราฐานเบื้องต้นในการเปรียบเทียบและวิเคราะห์ software กฎทั้ง 12 ข้อนี้ได้สรุปถึงปัจจัยต่างๆที่ RDBMS จำเป็นต้องมีดังนี้

6.1.1 นิยามของกฎของ Codd 12 ข้อ

กฎข้อที่ 1 The Information Rule

ข้อมูลต่างๆในระบบฐานข้อมูลจะถูก represent ในระดับ logical ไว้ด้วยวิธีการเดียวเท่านั้นคือ การใช้ตารางโดยอย่างน้อยจะต้องประกอบด้วยสิ่งต่อไปนี้

- ชื่อตาราง
- ชื่อ column
- column ใดเป็น index หรือ key
- ชนิดของข้อมูลในแต่ละ column
- ขอบเขตของค่าข้อมูลในแต่ละ column (domain)

กฎข้อที่ 2 Guaranteed Access Rule

ผู้ใช้ต้องสามารถเข้าถึงข้อมูลทุกตัวในตารางได้ด้วยการระบุชื่อตาราง ค่าของ primary key และชื่อ column ที่ต้องการ และถ้าไม่มีการเปลี่ยนแปลงข้อมูลในตารางแล้วละก็คำสั่งเดิมที่ใช้ในการเข้าถึงข้อมูลจะได้ผลออกมาเพียงค่าเดียว และเหมือนเดิมทุกครั้ง

กฎข้อที่ 3 Systematic Treatment of Missing Information

มีการใช้ค่า null เพื่อแสดงว่าระบบนี้ไม่มีข้อมูลในส่วนนั้น

กฎข้อที่ 4 Dynamic on-line catalog must be based on the relational model

กฎข้อนี้เน้นคุณสมบัติที่สามารถให้ผู้ใช้เรียกดูและ แก้ไขโครงสร้างข้อมูลต่างๆ ใน catalog ของระบบได้ด้วยภาษาและวิธีเดียวกับการเรียกดูข้อมูลของระบบ

กฎข้อที่ 5 Comprehensive data sublanguage rule

ระบบ RDBMS ควรมีภาษาอย่างน้อย 1 ภาษาที่สามารถทำการต่อไปนี้ได้

1. นิยามโครงสร้างระบบข้อมูล
2. นิยาม view
3. เปลี่ยนแปลงแก้ไขข้อมูล ทั้งด้วยวิธีการใช้คำสั่งแบบ interactive และด้วยการเขียนโปรแกรม
4. ควบคุม integrity ทั้ง entity integrity และ referential integrity
5. การให้สิทธิในการใช้งาน
6. ขอบเขตของ transaction คือการกำหนดว่าการทำงานกับข้อมูลในหน่วยใดหน่วยหนึ่งนั้นจะเริ่มต้น และสิ้นสุดตรงไหนทั้งนี้ก็เพราะหากระบบเกิด fail ระหว่างกลาง transaction ระบบจะได้สามารถ redo ได้

กฎข้อที่ 6 View updating rule

DBMS จะต้องสามารถตัดสินได้ว่าผู้ใช้สามารถเพิ่มเติมหรือลบ tuple หรือแก้ไขข้อมูลใน column ใดๆโดยกระทำการผ่าน view ได้หรือไม่ คือการกระทำดังกล่าวจะทำให้มีการสูญเสียความถูกต้องเกิดขึ้นกับตารางจริงหรือไม่ซึ่งเป็นส่วนที่ DBMS จะต้องควบคุมได้

กฎข้อที่ 7 High-Level insert , update and delete

DBMS ควรจะมีภาษาที่สามารถให้ผู้ใช้เพิ่ม ลด หรือแก้ไขข้อมูลในหลายๆ แถวหรือหลายๆ column ได้ด้วยการออกคำสั่งเพียงคำสั่งเดียว

กฎข้อที่ 8 Physical data independence

คือผู้ใช้ไม่จำเป็นต้องรับรู้เกี่ยวกับการจัดเก็บข้อมูลจริงรวมทั้งการเปลี่ยนแปลงวิธีการจัดเก็บเหล่านี้จะไม่กระทบกระเทือนกับการใช้งานที่ทำไว้แต่ดั้งเดิมด้วย

กฎข้อที่ 9 Logical data independence

การเปลี่ยนแปลงข้อมูลในระดับ logical (การเปลี่ยนแปลงโครงสร้างข้อมูล เช่น การเพิ่มเติม column เข้าในตาราง หรือการสร้างตารางใหม่) ไม่มีผลต่อคำสั่งและโปรแกรมที่เขียนขึ้นไว้ก่อน

กฎข้อที่ 10 Integrity independence

DBMS ควรจะสามารถจัดเก็บข้อมูลเกี่ยวกับการควบคุม integrity ไว้ใน catalog ของระบบโดยให้เป็นอิสระจากโปรแกรม

กฎข้อที่ 11 Distribution independence

ผู้ใช้ระบบไม่ต้องสนใจว่า ข้อมูลจะถูกจัดเก็บอย่างไรแม้จะถูกโยกย้ายไปยัง computer อื่นๆที่เชื่อมโยงไว้ก็ตาม (หลักการของ Distributed Data Base System)

กฎข้อที่ 12 Nonsubversion rule

integrity ต่างๆที่ถูกระบุไว้ใน catalog จะต้องสามารถใช้ควบคุมความถูกต้องของข้อมูลในระบบได้ตลอดเวลา ไม่ว่าผู้ใช้จะเข้าถึงข้อมูลด้วยเครื่องมือใด

6.1.2 การวิเคราะห์ระบบจัดการฐานข้อมูลตามกฎ 12 ข้อของ Codd

กฎข้อที่ 1 The Information rule

เนื่องจากระบบจัดการฐานข้อมูลทั้งสามเก็บข้อมูล(ในระดับ logical)เป็นตารางจึงกล่าวได้ว่า INGRES GUPTA และ ORACLE มีคุณสมบัติตามกฎ

กฎข้อที่ 2 Guaranteed Access rule

การเข้าถึง ข้อมูลในฐานข้อมูลของระบบจัดการฐานข้อมูลทั้งสาม สามารถทำได้โดยการระบุ ชื่อตาราง ค่า primary key และชื่อ column ที่ต้องการเท่านั้นและการเข้าถึงตารางด้วยคำสั่งเดิมใดๆก็ตามจะได้ผลออกมาเหมือนกันทุกครั้ง (ถ้าตารางไม่ได้ถูกแก้ไข) ซึ่งเป็นไปตามกฎข้อที่ 2

กฎข้อที่ 3 Systematic Treatment of Missing Information

ORACLE โดย default สำหรับ cell ที่ยังไม่มีข้อมูลใดๆ ORACLE ให้มีการใส่ค่า null ลงไป ยกเว้นเฉพาะ column ที่มีการระบุเป็น NOT NULL ซึ่ง column ดังกล่าวจะต้องมีค่าเสมอเป็น null ไม่ได้ ORACLE จึงถูกต้องตามกฎข้อที่ 3

INGRES มีการใช้ค่า null เพื่อแทนข้อมูลซึ่งยังไม่มีในตาราง โดยจะมีการเพิ่ม byte พิเศษเข้าไป 1 byte เพื่อแสดงว่าเป็น null ถ้าพบ byte นี้อยู่หน้าข้อมูลใด ก็จะถือว่าข้อมูลใน cell นั้นเป็น null ทันที INGRES จึงมีคุณสมบัติถูกต้องตามกฎข้อที่ 3

GUPTA ค่า NULL เป็นค่าที่แสดงว่า cell นั้นไม่มีค่าใดทุกๆชนิดของข้อมูล สามารถมีค่าเป็น NULL ได้ค่า NULL ไม่ได้มีค่าเท่ากับ 0 หรือ blank และค่า NULL ไม่สามารถจะนำไปเปรียบเทียบกับค่าอื่นๆได้ GUPTA จึงมีคุณสมบัติตามข้อนี้

กฎข้อที่ 4 Dynamic on-line catalog must be based on the relational model

ORACLE system catalog ของ ORACLE เช่นเดียวกับข้อมูลทั่วไปคือจะถูกจัดเก็บในลักษณะเป็น relational table และการ เข้าถึง system catalog ก็สามารถทำได้เหมือนการ เข้าถึงข้อมูลทั่วไป (แต่การแก้ไขต้องได้รับอนุญาตจาก DBA ก่อนไม่ใช่ทุกคนสามารถแก้ไข system catalogได้)
จึงจัดว่า ORACLE ถูกต้องตามกฎข้อที่ 4

INGRES system catalog ของ INGRES จะเก็บรายละเอียดที่เกี่ยวกับ ผู้ใช้และ รายละเอียดของตารางต่างๆ รวมทั้งเก็บกฎที่ใช้ในการควบคุมข้อมูลในตารางที่ ผู้ใช้สร้างขึ้นอีกด้วย system catalog ของ INGRES ถูกจัดเก็บในลักษณะเป็นตารางและ

สามารถ เข้าถึงโดยใช้ query language ได้(แต่การที่ ผู้ใช้ จะแก้ไข system catalog จะต้องได้รับอนุญาติจาก DBA ก่อน) INGRES จึงถูกต้องตามกฎข้อที่ 4

GUPTA system catalog ของ GUPTA เก็บรายละเอียดเกี่ยวกับตารางทุกๆ ตารางในฐานข้อมูลในลักษณะเป็นตารางและผู้ใช้สามารถที่จะเข้าถึงได้โดยการใช้ query เช่นเดียวกับการเข้าถึงตารางอื่นๆจึงจัดว่ามีคุณสมบัติข้อนี้เช่นกัน

กฎข้อที่ 5 Comprehensive data sublanguage rule

ORACLE มี ORACLE/SQL ซึ่งเป็นภาษาที่มีความสามารถตามกฎข้อที่ 5 ORACLE จึงถูกต้องตามกฎข้อนี้

INGRES มีINGRES/SQL ซึ่งเป็นภาษาที่มีความสามารถตามกฎข้อที่ 5 จึงกล่าวได้ว่า INGRES ถูกต้องตามกฎข้อนี้ นอกจากนั้น INGRES/QBF (แม้จะไม่ใช่ภาษา)ก็เป็นเครื่องมือที่มีความสามารถที่จะอนุโลมตามกฎข้อที่ 5 ได้อีกด้วย

GUPTA มี SQLTalk ทั้งสำหรับดอสและสำหรับวินโดส์ซึ่งมีความสามารตามกฎข้อนี้

กฎข้อที่ 6 View Update rule

ORACLE สำหรับ ORACLE ผู้ใช้ สามารถที่จะทำการแทรก แก้ไข หรือ ลบผ่าน view ได้
โดยมีกฎข้อบังคับดังต่อไปนี้คือ
1. ถ้า view ที่สร้างขึ้นถูกระบุโดยมี join operation, Set หรือ distinct operation, GROUP BY clause หรือ group function row ใน view นั้นจะไม่สามารถแทรก แก้ไข หรือ ลบได้
2. ถ้า view ถูกระบุขึ้นโดยมี with check option row ใน view นั้นจะไม่อนุญาติให้มีการ แทรก หรือ แก้ไขได้
3. ถ้า view ถูกระบุขึ้นโดยมี NOT NULL column (column ที่ต้องมีค่าเสมอเป็น null ไม่ได้) ที่ไม่มี default clause ของ base table จะไม่สามารถแทรก row ได้โดยใช้ view
4. ถ้า view ถูกสร้างโดยใช้ expression เช่น DECODE(deptno.,10,'sales',...) row จะไม่สามารถถูกแทรกหรือแก้ไขโดยใช้ view ได้

โดยสรุปก็คือ ORACLE สามารถควบคุมการแก้ไขผ่าน view ให้ถูกต้องได้โดยจะไม่อนุญาติให้มีการแก้ไขตามกรณีตามที่กล่าวมาข้างต้น ORACLE ระบบจัดการฐานข้อมูลจึงถูกต้องตามกฎข้อที่ 6 แม้จะไม่สมบูรณ์นัก

INGRES กฎการแก้ไข view ของ INGRES มีดังต่อไปนี้คือ INGRES จะไม่สนับสนุน การแก้ไขบน view ที่มี base table มากกว่า 1 table และไม่สนับสนุนการแก้ไข column ที่ source ไม่ได้เป็นชื่อ column name ธรรมดาๆเช่น column ที่เป็น set function หรือเป็น computation ในการทำการแก้ไขบน view จะมี with check option เป็นตัวควบคุมอยู่ถ้า with check option มีอยู่ใน column ใด system จะไม่อนุญาติให้มีการแก้ไขหรือแทรกใดๆบน column นั้น การแก้ไขบน view จะได้รับอนุญาติก็ต่อเมื่อเราสามารถรับประกันได้ว่าผลของการแก้ไขบน view นั้นจะให้ผลเหมือนกับการแก้ไขบน base table จริงๆ และการแทรกหรือแก้ไขดังกล่าวจะกระทำได้โดยใช้คำสั่ง SQL เท่านั้น INGRES จะไม่อนุญาติให้ทำโดยใช้ INGRES/QBFจากที่กล่าวมาจึงกล่าวได้ว่า INGRES มีคุณสมบัติตามกฎข้อที่ 6 แต่ไม่สมบูรณ์คือระบบจัดการฐานข้อมูลสามารถที่จะควบคุมความถูกต้องในการแก้ไข ข้อมูล บน view ได้แต่เฉพาะการแก้ไขบางกรณีตามที่กล่าวมาเท่านั้น

GUPTA ในการสร้าง view ผู้สร้างจะต้องมีสิทธิ์ในการใช้คำสั่ง SELECT INSERT UPDATE และ DELETE บน column ของ base table ที่จะนำมาสร้าง view การจะแก้ไขผ่าน view ได้นั้นจะทำได้ก็ต่อเมื่อ view นั้นสร้างจาก base table ตารางเดียวและ column ของ view ไม่สามารถจะนำมาจาก function หรือ expression ทางคณิตศาตร์ได้ ในการจะแทรกหรือแก้ไขผ่าน view ได้นั้นคำสั่งจะต้องถูกตรวจสอบโดย with check option ก่อน จึงพอสรุปได้ว่า GUPTA มีคุณสมบัติข้อนี้แต่ไม่สมบูรณ์

กฎข้อที่ 7 High-level insert,update and delete

ORACLE มีภาษาหลายตัวที่กระทำการตามกฎข้อนี้ได้เช่น ORACLE /SQL PLUS,PL/SQL,ORACLE/SQL เป็นต้น

INGRES มีINGRES/SQL ซึ่งเป็นภาษาที่มีความสามารถตามกฎข้อที่ 7

GUPTA มี SQLTalk ที่มีคุณสมบัติตามกฎข้อนี้

กฎข้อที่ 8 Physical data independence

ORACLE การจัดเก็บข้อมูลทางกายภาพ หรือแม้แต่การเปลี่ยนแปลงวิธีการจัดเก็บทางกายภาพใดๆของ ORACLE Data Dictionary จะไม่มีผลกระทบต่อ แอปพลิเคชั่นและการใช้งานที่ทำไว้ของ ผู้ใช้เลย จึงจัดว่า ORACLE ระบบจัดการฐานข้อมูล ถูกต้องตามกฎข้อนี้

INGRES ระบบจัดการฐานข้อมูลของ INGRES การเก็บข้อมูลทาง physical จะแยกออกจากโปรแกรมของผู้ใช้โดยเด็ดขาดนั่นคือวิธีการจัดเก็บข้อมูลและการเปลี่ยนแปลง

วิธีการจัดเก็บใดๆจะไม่มีผลต่อ code หรือ โปรแกรมที่ผู้ใช้เขียนไว้เลยจึงกล่าวได้ว่า INGRES มี physical data independence

GUPTA สำหรับ GUPTA การเก็บข้อมูลทาง physical จะแยกออกจากโปรแกรมของผู้ใช้การเปลี่ยนแปลงใดๆจึงไม่มีผลต่อแอปพลิเคชั่น GUPTA จึงมีคุณสมบัติข้อนี้

กฎข้อที่ 9 Logical data independence

ORACLE ระบบจัดการฐานข้อมูลของ ORACLE มีคุณสมบัติตามกฎข้อนี้เพราะแอปพลิเคชั่นของผู้ใช้จะไม่มีผลกระทบใดๆเลยเมื่อมีการแก้ไขเปลี่ยนแปลงโครงสร้างข้อมูลเช่นเพิ่ม column ในตารางหรือเพิ่มตารางใหม่ถึงแม้จะกระทบกระเทือนถ้าการเปลี่ยนแปลงนั้นเป็นการลบ column, drop ตารางหรือ split ตารางที่ผู้ใช้ใช้งานในโปรแกรมในระบบจัดการฐานข้อมูลของ INGRES การเปลี่ยนแปลง การเพิ่ม column หรือแม้แต่การสร้างตารางเพิ่มใหม่จะไม่มีผลต่อโปรแกรมที่ผู้ใช้เขียนไว้เช่น หากผู้ใช้มองเห็นอยู่ 5 ตาราง หากจะมีการสร้างตารางใหม่เพิ่มอีก 3 ตารางการเข้าถึงตารางของผู้ใช้ก็ยังเหมือนเดิมไม่ต้องมีการแก้ไขโปรแกรมใหม่ ทั้งนี้ยกเว้นกรณีที่การแก้ไข column เช่นการตัด column ที่มีการอ้างถึงในโปรแกรมของผู้ใช้ทิ้ง หรือการนำตารางที่ถูกอ้างถึงของผู้ใช้มา split เป็น 2 ตารางหรือ drop ตารางนั้นทิ้งซึ่งเหตุการณ์เหล่านี้จะมีผลต่อโปรแกรมที่เขียนไว้ก่อนอย่างแน่นอน และจะต้องแก้ code และ compile โปรแกรมใหม่ แต่อย่างไรก็ตามเรายังมองว่า INGRES มีคุณสมบัติตามกฎข้อที่ 9

GUPTA การเปลี่ยนแปลง การเพิ่ม column หรือแม้แต่การสร้างตารางเพิ่มใหม่จะไม่มีผลต่อโปรแกรมที่ผู้ใช้เขียนไว้ยกเว้นการเปลี่ยนแปลงนั้นเป็นการลบ การ drop ตาราง หรือการ split ตารางที่เกี่ยวข้องกับแอปพลิเคชั่นเท่านั้นจึงจะมีผลกระทบจึงสามารถมองว่า GUPTA มีคุณสมบัติข้อนี้

กฎข้อที่ 10 Integrity Independence

ORACLE จะอนุญาติให้มีการระบุและบังคับหลายๆชนิดของ data integrity rules ได้ซึ่งกฎเหล่านี้ระบุได้โดย integrity constraint ซึ่งเป็น declarative method ของการระบุกฎสำหรับ column ของตาราง

integrity constraint ที่ ORACLE สนับสนุน มีดังต่อไปนี้คือ

1. NOT NULL สำหรับกฎที่เกี่ยวกับ null ใน column
2. UNIQUE สำหรับกฎที่เกี่ยวกับค่าใน unique column
3. PRIMARY KEY สำหรับกฎที่เกี่ยวกับ primary identification values (entity integrity)

4. FOREIGN KEY คือกฎที่เกี่ยวกับ referencial integrity

5. CHECK สำหรับกฎที่เกี่ยวกับ complex integrity rule

การใช้ declarative integrity constraint จะไม่ต้องมีการเขียนโปรแกรมเลยใช้เพียง SQL เท่านั้น

สำหรับintegrity constraint นอกเหนือจากนี้คือ non-declarative integrity จะต้อง ระบุ โดยใช้ database trigger ซึ่งก็คือการเขียน stored procedures เช่น ถ้า child กับ parent table อยู่คนละ node กันบน distributed database referencial integrity จะไม่สามารถถูกบังคับใช้โดยใช้ declarative integrity ได้ต้องเขียน database trigger จัดเป็น non-declarative integrity

ทั้งนี้ทั้ง declarative และ non-declarative integrity constraint จะถูกเก็บแยกออกจาก แอปพลิเคชั่นของผู้ใช้เพราะฉะนั้นเวลาผู้ใช้เขียนแอปพลิเคชั่นจะไม่ต้องเขียนส่วน integrity นี้ใหม่ทุกๆครั้งแต่ ORACLE จะสามารถบังคับใช้กฎต่างๆให้ได้เลย จึงจัดว่ามี integrity independence ตามกฎข้อที่ 10 นี้

INGRES เราสามารถที่จะเขียนกฎต่างๆเพื่อใช้ในการควบคุมความถูกต้องของข้อมูลได้โดยที่กฎเกณฑ์เหล่านี้จะถูกเขียนเป็น database procedure (เขียนโดย DBA หรือผู้ใช้) แล้วถูกเก็บลงใน system catalog เมื่อผู้ใช้เขียนโปรแกรมอะไรก็ตามก็จะไม่ต้องเขียนส่วนตรวจสอบความถูกต้องในนี้นั่นก็คือ integrity ถูกเก็บโดยเป็นอิสระจากโปรแกรม จึงเป็นไปตามกฎข้อที่ 10 แต่ทั้งนี้ INGRES จะไม่ได้ provide integrity ใดๆมาให้ (ทั้ง entity integrity และ referential integrity) ผู้ใช้หรือ DBA จะต้องเขียนเอง

GUPTA มิได้มีการกล่าวถึงคุณสมบัติในข้อนี้

กฎข้อที่ 11 Distribution independence

ORACLE ระบบจัดการฐานข้อมูล ORACLE มีคุณสมบัติตามกฎข้อนี้เพราะในการทำงานบนแอปพลิเคชั่นใดๆผู้ใช้จะไม่ต้องสนใจว่าข้อมูลถูกเก็บอยู่ที่ไหนบนเครื่องใดในเครือข่ายแม้จะมีการโยกย้ายข้อมูลไปยังคอมพิวเตอร์เครื่องอื่นๆก็ยังคงไม่มีผลต่อแอปพลิเคชั่นของผู้ใช้

INGRES ในระบบจัดการฐานข้อมูลของ INGRES ไม่ว่าข้อมูลจะถูกจัดเก็บหรือโยกย้ายไปอย่างไรก็ตามแม้แต่จะถูกย้ายไปจัดเก็บยังเครื่องอื่นๆในระบบ โปรแกรมที่ใช้อยู่ก็จะไม่ต้องแก้ไขหรือ compile ใหม่เลยจึงจัดว่ามี distribution independence ตามกฎข้อที่ 11

GUPTA มิได้มีการกล่าวถึงคุณสมบัติในข้อนี้

กฎข้อที่ 12 Nonsubversion rule

ORACLE เนื่องจาก integrity ที่ระบุไว้สามารถจะควบคุมความถูกต้องของข้อมูลในระบบได้ตลอดเวลาไม่ว่าผู้ใช้จะใช้เครื่องมือใดในการเข้าถึงข้อมูลจึงถือว่า ORACLE ถูกต้องตามกฎข้อนี้

INGRES เนื่องจาก integrity ใน system catalog จะสามารถควบคุมความถูกต้องของระบบได้ตลอดเวลาจึงถือว่าถูกต้องตามกฎ

GUPTA ในการตรวจสอบ integrity ของ GUPTA จะกระทำเมื่อผู้ใช้รันคำสั่ง check database เท่านั้นจึงไม่สามารถควบคุมความถูกต้องของข้อมูลได้ตลอดเวลาจึงสรุปได้ว่า GUPTA ไม่มีคุณสมบัติข้อนี้

6.2 กฎ 16 ข้อของการออกแบบระบบจัดการฐานข้อมูลของ Codd

E.F.codd ปรมาจารย์ทางด้านระบบฐานข้อมูลเชิงสัมพันธ์ได้กล่าวถึงกฎ 16 ข้อในการออกแบบระบบจัดการฐานข้อมูลเชิงสัมพันธ์ซึ่งโดยทั่วไปแนวความคิดทางทฤษฎีของ relational model ค่อนข้างกระจ่างแล้วแต่ ผู้ค้าของระบบจัดการฐานข้อมูลส่วนใหญ่ยังมีข้อผิดพลาดอีกมากมายที่เกิดขึ้นในส่วนการ implementation (ออกแบบและสร้างให้เป็นไปตามที่ต้องการ) ดังนั้นกฎ 16 ข้อของการออกแบบระบบจัดการฐานข้อมูลเชิงสัมพันธ์ที่จะกล่าวถึงนี้จะเน้นในด้านการ implementation ดังต่อไปนี้

กฎข้อที่1. Non-violation of any Fundamental Law of Mathematics

(ไม่ควรเกิดกรณีที่ขัดแย้งกับกฎทางคณิตศาสตร์)

Codd คิดว่าทั้งผลจากการทำงานของระบบจัดการฐานข้อมูลเชิงสัมพันธ์และภาษาที่ใช้ซึ่งจะต้องเป็นภาษาประเภทภาษา relational เช่น ภาษา SQL เป็นต้น ไม่ควรขัดแย้งกับกฎทางคณิตศาสตร์ เพียงแค่พิจารณาจากกฎการสลับที่ได้ของลอจิก AND ซึ่งจะพบว่ากรณีที่ระบบจัดการฐานข้อมูลเชิงสัมพันธ์ในท้องตลาดปัจจุบันขาดคุณสมบัติตามกฎข้อนี้คือ กรณีที่เงื่อนไขเป็น where Y and X แล้วหากมีการเรียกค้นหาข้อมูล Query1 มีเงื่อนไข where Y AND X และการค้นหาข้อมูล Query2 มีเงื่อนไข where X AND Y แล้วมักจะได้ผลไม่เหมือนกันซึ่งจะเห็นได้ว่าขัดแย้งกับกฎการสลับที่ได้ของคณิตศาสตร์

เมื่อพิจารณาผลการทดสอบ พบว่าทั้ง INGRES, ORACLE และ GUPTA มีคุณสมบัติข้อนี้เนื่องจาก ผลที่ได้จากการทดสอบ query โดยเงื่อนไข Where ที่ต่างกันแต่ได้ผลลัพธ์เดียวกัน จะให้ response time ที่ต่างกัน

กฎข้อที่ 2 Under-the-covers Representation and Access

(ระบบจัดการฐานข้อมูลจะมีหน้าที่จัดการเกี่ยวกับการจัดเก็บและการเข้าถึงข้อมูลระดับล่าง)

Codd คิดว่า สิ่งที่เกี่ยวกับการเข้าถึงข้อมูลระดับล่างไม่ว่าจะเป็นวิธีการเก็บรักษาข้อมูลและวิธีการเข้าถึงข้อมูลระดับล่างเป็นหน้าที่ของ ระบบจัดการฐานข้อมูล โดยที่ ผู้ใช้ไม่จำเป็นต้องรับรู้ เช่นในกรณี query ที่ได้ result เหมือนกันแต่ ใช้วิธี query และวิธีการเลือกเส้นทางที่ต่างกันควรจะได้ response time เดียวกัน

เมื่อพิจารณาผลการทดสอบ :- INGRES ORACLE และ GUPTA

ในกรณี query ที่ได้ result เดียวกันโดย query แบบ JOIN และแบบ SUBQUERY นั้น

- ในสภาวะที่มี ผู้ใช้ เพียงคนเดียวนั้น query เดียวกันแต่ใช้วิธีการ JOIN และวิธีการ SUBQUERY นั้นใช้เวลาประมวลผลใกล้เคียงกันมากแต่

- ในสภาวะที่มี ผู้ใช้ 2 และ 3 คน

เมื่อทดสอบ query โดยใช้วิธีการ JOIN จะใช้เวลาประมวลผลต่างจากเมื่อทดสอบ query โดยใช้วิธีการ SUBQUERY ซึ่งผลต่างของเวลาในการประมวลผลจะเพิ่มขึ้นตามจำนวนของผู้ใช้ ดังนั้นจึงสรุปได้ว่า INGRES ,GUPTA และ ORACLE ไม่มีคุณสมบัติข้อนี้

กฎข้อที่ 3 Sharp Boundary

(ระบบจัดการฐานข้อมูลจะต้องแยกระหว่างส่วนที่จัดการเกี่ยวกับ Performance และ ส่วนที่จัดการเกี่ยวกับ Semantic)

ระบบจัดการฐานข้อมูลจะต้องแยกระหว่างส่วนที่จัดการเกี่ยวกับ Performance เช่น Index และ ส่วนที่จัดการเกี่ยวกับ Semantic เช่น Primary key,foreign key ฯลฯ กรณีที่ระบบจัดการฐานข้อมูลในท้องตลาดปัจจุบันขาดคุณสมบัติตามกฎข้อนี้คือ ถ้าเราต้องการตารางที่มี primary key ก็ไม่ควรต้องสร้าง unique index แต่ยังมี ระบบจัดการฐานข้อมูลบางระบบที่มีคุณสมบัติข้อนี้ซึ่งระบบเหล่านั้นจะใช้ primary integrity เป็นตัวกำหนด primary key แทน

เมื่อพิจารณาผลการทดสอบ :-

INGRES

ถือว่า INGRES มีคุณสมบัติข้อนี้โดยใช้เครื่องมือช่วยในการกำหนด PRIMARY KEY ขณะสร้างตารางส่วนการสร้าง index นั้นก็ช่วยให้ response time ในการทำ Grouping Feature ได้ดีขึ้น

ORACLE

ถือว่า ORACLE มีคุณสมบัติข้อนี้โดยใช้ " PRIMARY KEY INTEGRITY CONSTRAINT " ส่วนการ สร้าง index นั้นก็ช่วยให้ response time ในการทำ Grouping Featureได้ดีขึ้น

GUPTA

ถือว่า GUPTA มีคุณสมบัติข้อนี้โดยใช้ คำสั่ง PRIMARY KEY(column-name) ในขณะที่สร้างตาราง ส่วนการสร้าง index นั้นก็ช่วยให้ response time ในการทำ Grouping Featureได้ดีขึ้น

กฎข้อที่ 4 Concurrency Independence

(ระบบจัดการฐานข้อมูลจะควบคุมการใช้ข้อมูลในฐานข้อมูลร่วมกันของหลายผู้ใช้ในช่วงเวลาเดียวกัน)

ระบบจัดการฐานข้อมูลจะจัดการควบคุมการเรียกใช้และเข้าถึงข้อมูลในฐานข้อมูลในสภาวะแบบ Multiuser กล่าวคือ มีผู้ใช้มากกว่า 1 คน และ ผู้ใช้เหล่านั้น แก้ไขข้อมูลในตารางเดียวกัน ในช่วงเวลาเดียวกันจำเป็นต้องมี Concurrency Control ซึ่ง INGRES ใช้กระบวนการ locks โดยประกันได้ว่าในขณะที่ Transaction ใด Transaction หนึ่ง แก้ไข ข้อมูลใดอยู่จะไม่มี Transaction อื่นใดสามารถเข้าถึง ข้อมูลเหล่านั้นได้ไม่ว่าชนิดของการเข้าถึงนั้นจะเป็น read หรือ write ก็ตามจนกว่า Transaction ที่ แก้ไข ข้อมูลนั้นจะ commit แล้วเพื่อรักษาสภาวะความถูกต้องโดยไม่ยอมให้ข้อมูลและการประมวลผลกลุ่มของงานต้องขัดแย้งกันและทำให้ข้อมูลในฐานข้อมูลไม่ถูกต้องโดยที่ผู้ใช้ไม่จำเป็นต้องรับรู้เลยซึ่งระบบจัดการฐานข้อมูล ในท้องตลาดมักจะมีคุณสมบัติตามกฎข้อนี้โดยวิธีที่นิยมใช้คือใช้กลไกการ Lock (Lock Mechanism)

เมื่อพิจารณาผลการทดสอบ :-

INGRES

เมื่อพิจารณาเวลาที่ใช้ในการประมวล query และหลักการทำงานแล้วพบว่า INGRES มีคุณสมบัติข้อนี้ โดยจะใช้ระบบ locking แบบ "multiple granularity "ซึ่งมี level การ lock ทั้งหมด 3 levelได้แก่ 1.page level
2.table level
3.database level

ORACLE

เมื่อพิจารณาเวลาที่ใช้ในการประมวล query และหลักการทำงานแล้วพบว่า ORACLE มีคุณสมบัติข้อนี้ โดยจะใช้ "locking mechanism" ซึ่งแบ่งเป็น 3 ประเภท ได้แก่

1. Data locks
2. Dictionary locks
3. Distributed locks
4. Internal locks and Latches
5. Parallel cache management locks

GUPTA

เมื่อพิจารณาเวลาที่ใช้ในการประมวล query และหลักการทำงานแล้วพบว่า GUPTA มีคุณสมบัติข้อนี้ โดยจะใช้ "locking mechanism"

กฎข้อที่ 5 Protection against unauthorized Long-term Locking

(ระบบจัดการฐานข้อมูลต้องมีการจัดการป้องกันการ Lock ในระยะเวลาที่นานเกินไป)

ระบบจัดการฐานข้อมูลต้องมีการกำหนดระยะเวลาที่ผู้ใช้แต่ละคนจะสามารถ Lock ทรัพยากรข้อมูลซึ่งระบบจัดการฐานข้อมูลในท้องตลาดบางระบบมีคุณสมบัติข้อนี้กล่าวคือจะใช้วิธีการกำหนดตัวแปรชื่อ "timeout" ซึ่งจะกำหนดระยะเวลาที่ผู้ใช้จะสามารถเรียกใช้และเข้าถึงข้อมูลได้

เมื่อพิจารณาผลการทดสอบ :-

INGRES

เมื่อพิจารณาแล้วพบว่า INGRES มีคุณสมบัติข้อนี้ โดยจะใช้ parameter " time out "

ORACLE

เมื่อพิจารณาแล้วพบว่า ORACLE มีคุณสมบัติข้อนี้โดยจะใช้ "Transaction Processing option " (TPO)

GUPTA

เมื่อพิจารณาแล้วพบว่า GUPTA ไม่มีคุณสมบัติข้อนี้

กฎข้อที่ 6 Orthogonality in DBMS Design

(แนวทางการออกแบบระบบจัดการฐานข้อมูลต้องอยู่ในแนวทางเดียวกัน)

ระบบจัดการฐานข้อมูลในท้องตลาดปัจจุบันนี้ยังไม่ชัดเจนในคุณสมบัติตามกฎข้อนี้

เมื่อพิจารณาผลการทดสอบ พบว่าทั้ง INGRES, ORACLE และ GUPTA มี

กฎข้อที่ 7 Damain-based Index

Codd กล่าวไว้ว่า การสร้าง index นั้นควรจะสร้างจาก domain เลยแต่กรณีที่ระบบจัดการฐานข้อมูลในท้องตลาดปัจจุบันขาดคุณสมบัติตามกฎข้อนี้คือยังคงต้องสร้าง index ที่ column อยู่

เมื่อพิจารณาผลการทดสอบ พบว่าทั้ง INGRES, ORACLE และ GUPTA ไม่มีคุณสมบัติข้อนี้เนื่องจากระบบจัดการฐานข้อมูลในปัจจุบันยังใช้การ สร้าง index บน attribute

กฎข้อที่ 8 Database Statistics

(ระบบจัดการฐานข้อมูลต้องมีข้อมูลทางสถิติของฐานข้อมูล)

Codd กล่าวไว้ว่าควรมีการเก็บข้อมูลทางสถิติของข้อมูลที่เก็บอยู่ในฐานข้อมูลและเก็บข้อมูลทางสถิติเหล่านี้ใน catalog ของระบบเพื่อช่วยในการทำ Query Optimization (การเลือกวิธีในการ retrieve และจัดการเก็บข้อมูลในฐานข้อมูลให้มีประสิทธิภาพมากที่สุด)

เมื่อพิจารณาผลการทดสอบ

INGRES

เมื่อพิจารณาเวลาที่ใช้ในการประมวล query และหลักการทำงานแล้วพบว่า INGRES มีคุณสมบัติข้อนี้ โดยจะเก็บข้อมูล

1.จำนวน row ใน table

2.จำนวน unique value ใน column

3.NULL count

4.จำนวน duplicate value ใน column

5.Histogram data แสดง distribution

ORACLE

เมื่อพิจารณาเวลาที่ใช้ในการประมวล queryและหลักการทำงานแล้วพบว่า ORACLE มีคุณสมบัติข้อนี้ โดยจะทำ Query optimization ด้วย mode การทำงาน

1.rule-based approach พิจารณาจาก เส้นทางในการเข้าถึง

2.cost-based approach พิจารณาจาก cost ของแต่ละ execution plan ของแต่ละ SQL statement ซึ่ง cost พิจารณาจากข้อมูลทางสถิติ

GUPTA

เมื่อพิจารณาเวลาที่ใช้ในการประมวล query และหลักการทำงานแล้วพบว่า GUPTA มีคุณสมบัติข้อนี้ โดยจะมีข้อมูลเกี่ยวกับ ข้อมูล,indexes,ตาราง และ catalog statistics เป็นต้น

กฎข้อที่ 9 Interrogation of Statistics

(ต้องสามารถเรียกดูข้อมูลทางสถิติได้)

ผู้บริหารฐานข้อมูล (DBA) จะต้องสามารถเรียกดูข้อมูลทางสถิติที่เก็บไว้ใน catalog ของระบบได้

เมื่อพิจารณาผลการทดสอบ เมื่อพิจารณาหลักการทำงานแล้วพบว่าทั้งINGRES ORACLE และ GUPTA ต่างก็มีคุณสมบัติข้อนี้

กฎข้อที่ 10 Changing Storage Representation and Access Options

(ระบบจัดการฐานข้อมูลสามารถเปลี่ยนแปลงการเก็บข้อมูลและวิธีการเข้าถึงข้อมูลได้)

ระบบจัดการฐานข้อมูลสามารถเปลี่ยนแปลงการเก็บข้อมูลและวิธีการเข้าถึงข้อมูลได้โดยที่ application ที่วิ่งอยู่ไม่จำเป็นต้องรับรู้หรือเปลี่ยนแปลง application ตาม เมื่อพิจารณาผลการทดสอบ

เมื่อพิจารณาหลักการทำงานแล้วพบว่าทั้ง INGRES ,ORACLE และ GUPTA ต่างก็มีคุณสมบัติข้อนี้โดย

INGRES มี structure storage 4 แบบคือ heap isam hash B-tree

ORACLE มี structure storage 3 แบบคือ cluster hash B**-tree

กฎข้อที่ 11 Automatic Protectiion in case of malfunctions

(ถ้า transaction ใด transaction หนึ่ง ผิดพลาด แล้ว ฐานข้อมูล ต้องไม่เสียหาย แต่ transaction อาจต้องมีผลกระทบบ้าง)

หมายถึง ฐานข้อมูลควรอยู่ใน consistence state ไม่ว่า transaction จะผิดพลาด หรือ ถูกต้องสมบูรณ์

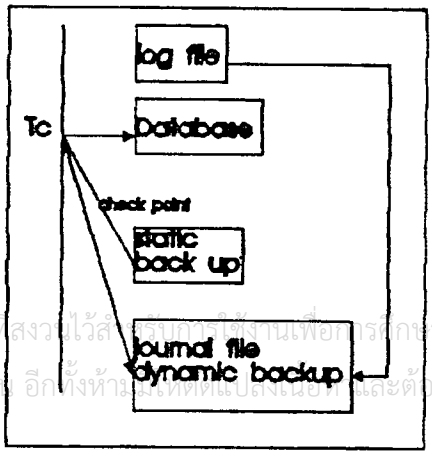
เมื่อพิจารณาผลการทดสอบพบว่าทั้ง INGRES, ORACLE และ GUPTA ต่างก็มีคุณสมบัติข้อนี้

กฎข้อที่ 12 Automatic recovery in case of malfunctions

(ถ้าเกิดการเสียหายกับ transaction แล้ว ระบบจัดการฐานข้อมูล ต้องมีความสามารถที่จะ recovery transaction คืนได้)

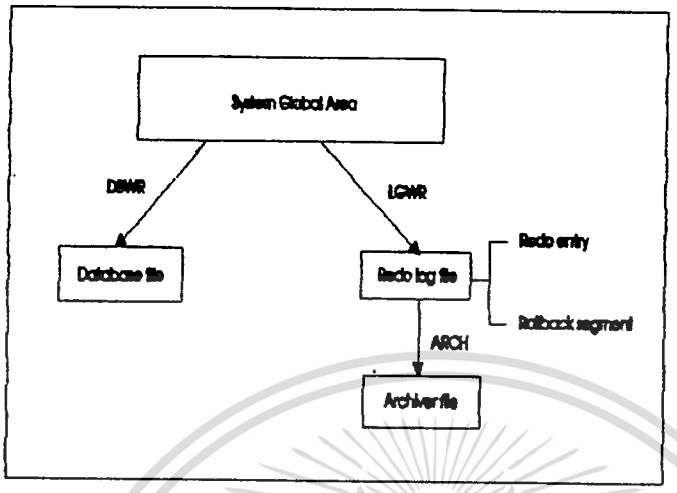
เมื่อพิจารณาผลการทดสอบพบว่า

INGRES พิจารณาที่ check point (tc)



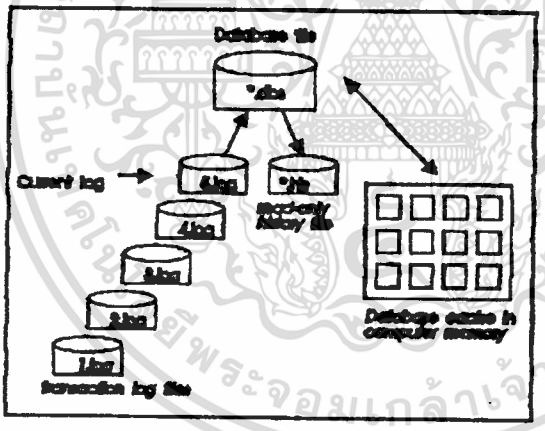
การ recovery จะทำจาก static backup + dynamic backup

ORACLE



การ recovery จะทำจาก online redo log file และ archieve redo log file

GUPTA



การ recovery จะทำจาก database file และ log file ที่ได้ทำการ backup ไว้

กฎข้อที่ 13 Atomic execution of relational command

(1 คำสั่งrelational ต้องเป็น atomic ใน sense ของ ผู้ใช้ คือ ถ้า ผิดพลาด ต้องผิดพลาด ทั้ง command)

เมื่อพิจารณาผลการทดสอบพบว่า INGRES,ORACLE และ GUPTA มีคุณสมบัติข้อนี้

กฎข้อที่ 14 Atomic archieving

- archieve คือ ที่เก็บของที่ไม่ค่อยได้ใช้

- เช่น เมื่อถึงเวลาใดเวลาหนึ่งจะมีการ ดัมพ์ฐานข้อมูล ลงใน archeive file เก็บเอาไว้

เมื่อพิจารณาผลการทดสอบพบว่า

INGRES มี โดยการทำ static backup และ dynamic backup ซึ่งเรียกว่า archiver process

ORACLE มี โดยใช้ขบวนการที่เรียกว่า ARCH จะเป็นการเก็บกลุ่มของ redo log file เก่าๆลงใน archieved redo log file เพื่อเพิ่มที่ว่างให้กับ online redo log file

GUPTA มีการ backup ฐานข้อมูลทั้งแบบ online backup และ offline backup และสามารถตั้ง parameter LOGBACKUP ให้เป็น ON ซึ่งจะทำให้มีการ backup log file ให้

กฎข้อที่ 15 Avoiding Cartesian product

- cartesian product ไม่มีความหมาย เป็น intermediate result เฉยๆ
- ไม่ควรมีการ generate cartesian product ลงใน ฐานข้อมูล และไม่ควรให้มี final result เป็น cartesian product เด็ดขาด

เมื่อพิจารณาผลการทดสอบพบว่า INGRES,ORACLE และ GUPTA ต่างก็ขาดคุณสมบัติในข้อนี้

กฎข้อที่ 16 Responsibility for decryption and encryption

(decrytion และ encryption ควรเป็นเรื่องของใต้ ตาราง ที่ ผู้ใช้ ไม่ต้องทำ)

เมื่อพิจารณาผลการทดสอบพบว่า INGRES,ORACLE และ GUPTA ต่างก็มีคุณสมบัติในข้อนี้

บทที่ 7

ระบบผู้เชี่ยวชาญ 'การเลือก platform'

งานปริญญานิพนธ์นี้ได้มีการนำเสนอการเลือก platform ในลักษณะเป็น ระบบผู้เชี่ยวชาญ ซึ่งเมื่อผู้ใช้ได้ทำการตอบคำถามต่างๆจนครบแล้วระบบผู้เชี่ยวชาญจะให้คำปรึกษาว่าในลักษณะงานของผู้ใช้ควรที่จะใช้ platform แบบใดเช่นควรจะเป็น Client/Server, Host-based หรือแม้แต่ PC-based LAN สำหรับระบบผู้เชี่ยวชาญตัวนี้เขียนขึ้นโดยใช้ PC-Plus expert system shell

7.1 Platform ต่างๆที่มีการนำเสนอใน expert system

1. stand alone PC

คือ PC ธรรมดาที่ไม่มีการต่อเข้ากับเครือข่ายใดๆซึ่งเหมาะสำหรับกรณีที่ไม่มีความต้องการ multiple users

2. multiuser system with terminals

เป็นการรัน multiuser operating system บน PC ที่มีความสามารถสูง หรือ RISC workstation และต่อ PC เพิ่มเข้าไปเป็นเทอมินัลมีลักษณะคล้ายเป็น centralize system ขนาดเล็ก

3. multi PC-based DBMS

ระบบนี้จะมีการรันระบบจัดการฐานข้อมูลบน PC นั่นคือตัว PC ที่รันระบบจัดการฐานข้อมูล นี้จะทำตัวเป็นทั้งคอมพิวเตอร์หลัก และเทอมินัลเลยและจะมี PC ต่างหากที่รัน network OS พิเศษเรียกว่าเป็น file server ทำให้มีการใช้ข้อมูลร่วมกันได้

4. limited Client/Server DBMS บน PC-server

เป็น Clien/Server ที่มีข้อจำกัด ลักษณะของระบบคือมีการเพิ่ม server function ลงใน PC-based database เป็นระบบที่สามารถจะรองรับ multiple frontend แต่ในจำนวนที่ไม่มากนักการประมวลผลส่วนมากจะทำอยู่ที่ตัว Client ตัว server สามารถที่จะ provide เฉพาะบาง DBMS function เท่านั้นส่วนมากคือพวก function ที่เกี่ยวกับการเก็บข้อมูลข้อเสียเปรียบคือ server ไม่สามารถที่จะกันผู้ใช้จากการเข้าถึงข้อมูลโดยไม่อยู่ภายใต้การดูแลของ DBMS sever ที่รันบน server นั้นได้

5. Client/Server บน RISC หรือ PC system

ในระบบนี้ตัวระบบจัดการฐานข้อมูลจะรันอยู่บน server และดูแลการ access ทุกอย่างคือ ผู้ใช้จะเข้าถึงข้อมูลได้จะต้องผ่านระบบจัดการฐานข้อมูลบน server เสมอการประมวลผลของระบบจัดการฐานข้อมูลจะทำอยู่บน server และเป็นตัวจัดการการ query

การแก้ไขข้อมูล และ รายงานผล สำหรับข้อมูลก็จะเก็บอยู่บน server(1 ตัวหรือมากกว่าก็ได้)

6. Client/Server DBMS gateway

เป็นระบบที่มักจดเป็นการใช้ร่วมกันระหว่าง PC-based system กับระบบจัดการฐานข้อมูลมารันบนเมนเฟรมหรือมินิคอมพิเตอร์โดย gateway จะสร้าง bridge ระหว่าง user front-end application และระบบจัดการฐานข้อมูลที่รันอยู่บนระบบ(ที่ไม่ได้เป็น Client/Server) รวมทั้ง gateway จะทำการแปลง query ฯลฯ ไปเป็น procedures และ calls ที่ระบบจัดการฐานข้อมูลจะสามารถประมวลผลได้

7. software Client/Server gateway

คือในกรณีที่ระบบจัดการฐานข้อมูลมี Client/Server option แล้วและรวมทั้ง vendor ใช้ protocol ที่ front-end platform support อยู่จึงสามารถที่จะ install vendor gateway software ลงบนคอมพิวเตอร์หลักได้เลย

8. hardware Client/Server bridge/gateway

คือกรณีที่ต้องมีการเพิ่ม hardware และ software gateway ซึ่งจะทำหน้าที่ translate protocol ทั้ง Client/Server และ network protocol

7.2 วิธีการปรึกษาระบบผู้เชี่ยวชาญ

เมื่อผู้ใช้รันโปรแกรม ระบบผู้เชี่ยวชาญจะถามคำถามต่อผู้ใช้โดยผู้ใช้จะต้องเลือกตอบ 'yes' หรือ 'no' เมื่อได้คำตอบแล้วระบบผู้เชี่ยวชาญก็จะถามคำถามต่อซึ่งจะเป็นคำถามใดนั้นขึ้นอยู่กับคำตอบของผู้ใช้และเมื่อระบบผู้เชี่ยวชาญได้รับข้อมูลเพียงพอในการตัดสินใจแล้วก็จะให้คำตอบที่คิดว่าเหมาะสมที่สุดออกมาทั้งนี้การที่จะได้คำตอบเป็น platform ที่เหมาะสมกับลักษณะงานและความต้องการจริงๆแล้วผู้ใช้จะต้องตอบคำถามต่างๆของระบบผู้เชี่ยวชาญตามความเป็นจริงเสมอ

7.3 ตัวอย่างการใช้ระบบผู้เชี่ยวชาญ

คำถามของ ระบบผู้เชี่ยวชาญ	คำตอบของ ผู้ใช้
1. ต้องการ multiple ผู้ใช้ หรือไม่	'y'
2. จะมีผู้ใช้ใช้งานพร้อมๆกันมากกว่า 20 คนหรือไม่	'y'
3. มี multiple Tx ที่จะแก้ไขข้อมูลหรือไม่	'y'
4. เดิมใช้ multiusers PC-based DBMS หรือไม่	'n'
5. ใช้ minicomputer หรือ mainframe อยู่แล้วหรือไม่	'y'

5. ใช้ minicomputer หรือ mainframe อยู่แล้วหรือไม่ 'y'
6. ต้องการจะ downsize โดยใช้ระบบใหม่แทนระบบเก่าเลยหรือไม่ 'n'
7. ระบบจัดการฐานข้อมูล มี C/S option สำหรับ software อยู่หรือไม่ 'n'

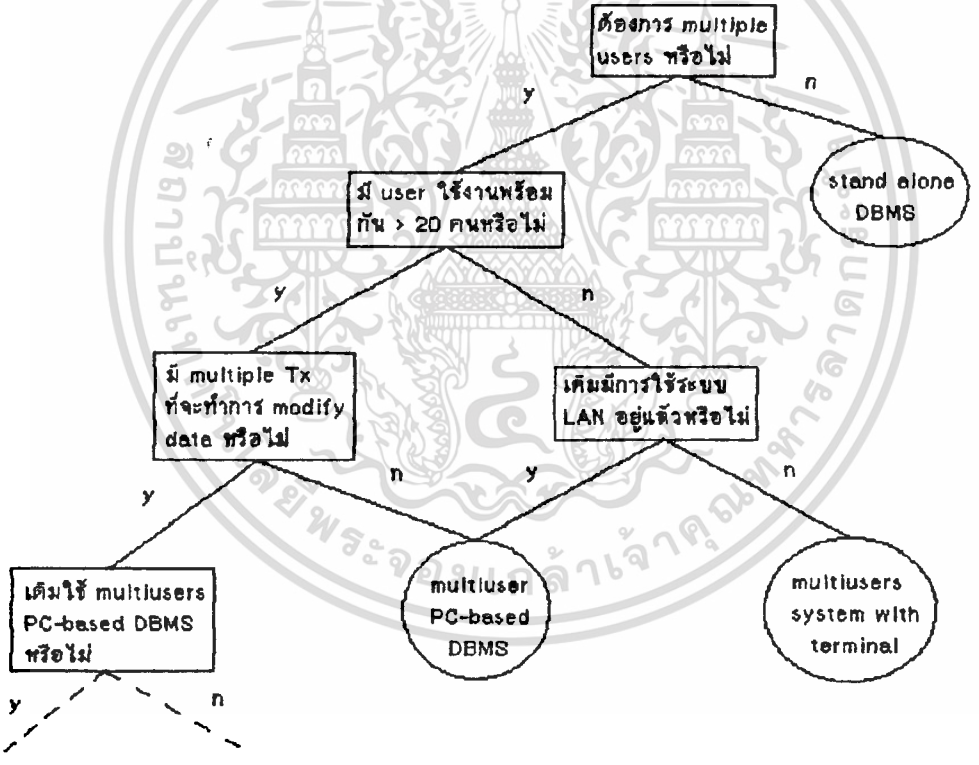
หลังจากหมดชุดคำถามนี้แล้วผู้ใช้จะได้รับคำตอบเป็นระบบ C/S DBMS gateway

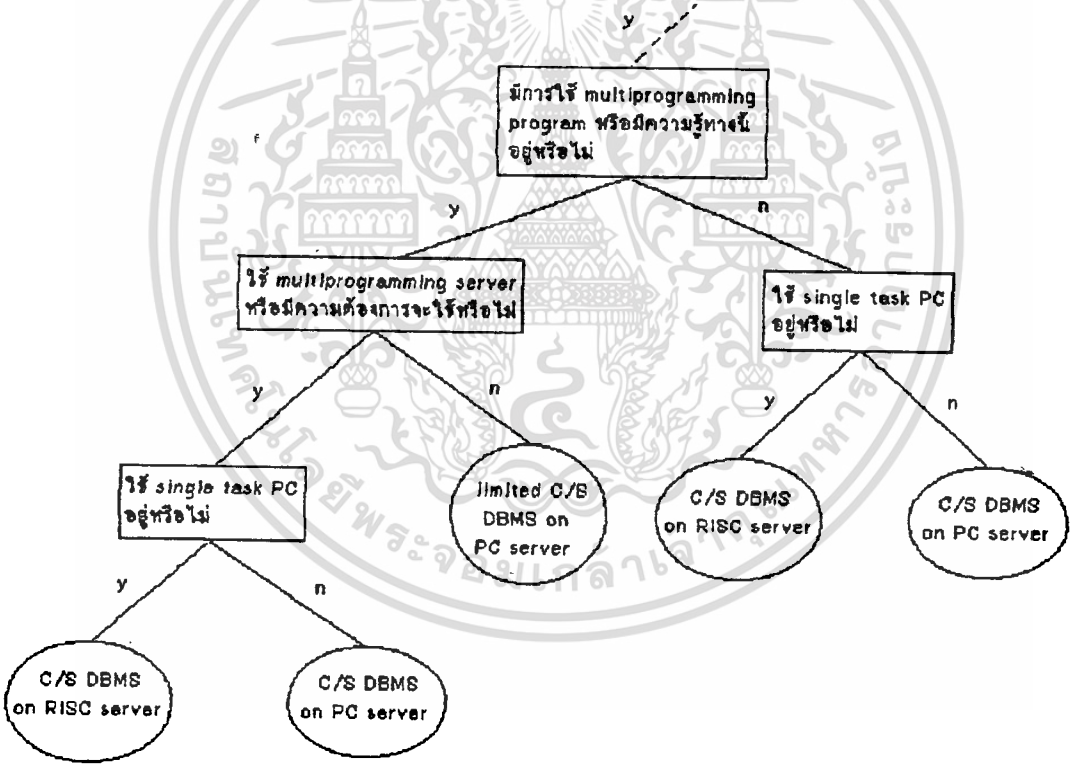
จากตัวอย่างจะเห็นว่าผู้ใช้ต้องการระบบที่ support multiuser จำนวนมากพอสม ควรสามารถดูแลการแก้ไขข้อมูลจากหลายๆ transaction ได้รวมทั้งทางผู้ใช้มี mini หรือ mainframe อยู่แล้วและไม่ต้องการจะ downsize โดยใช้ระบบใหม่แทนเลยตัวระบบจัดการฐานข้อมูลที่ใช้อยู่ก็ไม่มี Client/Server option เพราะฉะนั้น platform หรือระบบที่เหมาะสมกับความต้องการตามลักษณะดังกล่าวมากที่สุดคือ C/S DBMS gateway เพราะการใช้ gateway จะทำให้ระบบ PC-based system สามารถที่จะทำงานร่วมกับระบบจัดการฐานข้อมูลที่รันบน mainframe ได้คือ front-end สามารถจะติดต่อกับระบบที่ไม่ได้เป็น C/S อยู่ได้ทำให้ไม่ต้องมีการ downsize ระบบทั้งหมด

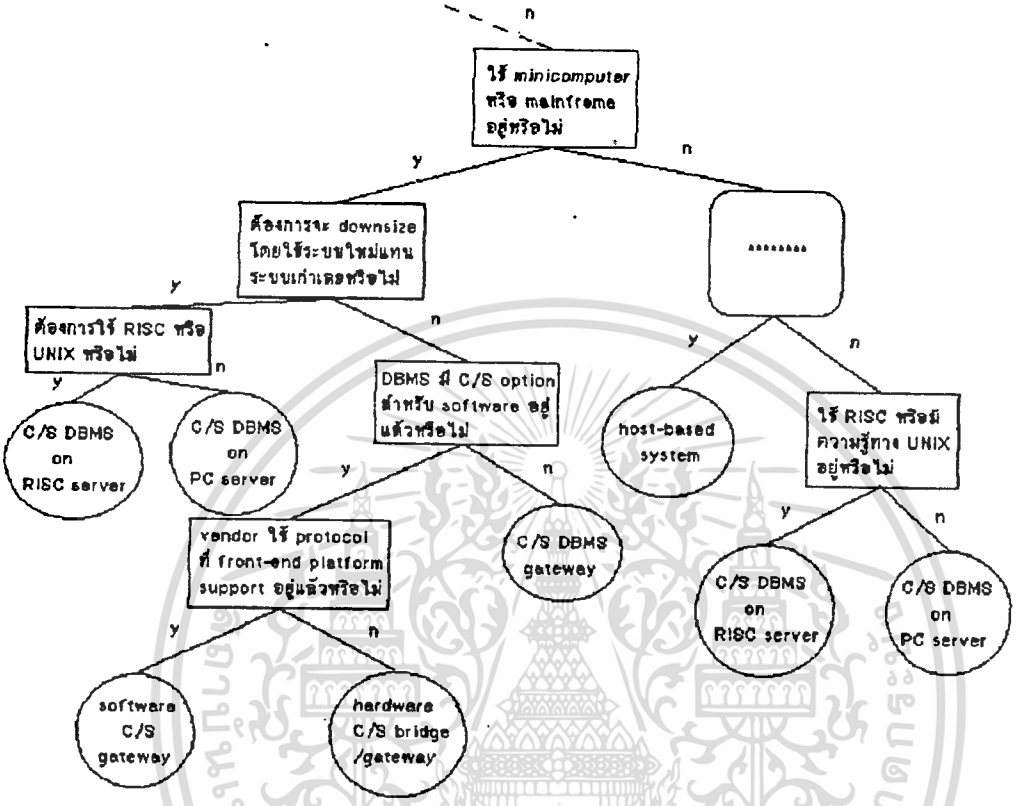
เหตุที่ผู้ใช้ไม่ต้องการที่ตะ downsize ระบบทั้งหมดเลยอาจเกิดจากข้อมูลที่ใช้มีมากเกินที่ PC หรือ mini ขนาดเล็กจะรองรับได้หรืออาจเพราะได้ลงทุนไปมากแล้วในการเขียนแอปพลิเคชั่นและไม่อยากจะทำการเขียนใหม่ เพราะฉะนั้นจึงใช้วิธี provide C/S functionality บนระบบที่มีอยู่แล้ว

7.4 Decision Tree

ลักษณะของข้อมูลในระบบผู้เชี่ยวชาญตัวนี้เป็น decision tree ซึ่งมีรูปแบบดังต่อไปนี้







1. users มีความคุ้นเคยกับระบบ PC อยู่แล้ว
2. ต้องการ expandability ของ terminal
3. ต้องการให้ end user พัฒนา application
4. ต้องการ remote users
5. ต้องการ graphic interface กับ users (ต้องการ user friendly)
6. ต้องการให้ user สามารถเลือก hardware และ software ให้เหมาะสมกับ application และความต้องการได้
7. มี initial cost จำกัด
8. งานส่วนมากเป็น CPU intensive application

บทที่ 8

การทดสอบระบบจัดการฐานข้อมูลแบบรีเลชันนัล

8.1 การทดสอบประสิทธิภาพของ DBMS

1. ข้อมูลที่นำมาใช้ในการทดสอบนี้เป็นข้อมูลรายการหนังสือจากห้องสมุด ซึ่งในการทดสอบนี้ใช้ตาราง 2 ตารางและได้ตัดเอาเฉพาะบาง column มาใช้ดังนี้

ตารางที่ 1

p_access_no	p_status_cd	p_publish_plc	p_be_yr	p_ad_yr	p_call_no

ตารางที่ 1 มีจำนวน record ทั้งหมดที่นำมาใช้ทดสอบ 22275 record

ตารางที่ 2

p_call_no	title	p_publisher	p_length	p_apage_no	p_lang	author_name	author_surname

ตารางที่ 2 มีจำนวน record ทั้งหมดที่นำมาใช้ทดสอบ 12901 record

2. ค่า distinct values ของแต่ละ column ในตารางเป็นดังนี้

Table print_lib

column	distinct value
p_access_no	22275
p_status_cd	1
p_publish_plc	1559
p_be_yr	60
p_ad_yr	93
p_call_no	12901

Table p_call_lib

column	distinct value
p_call_no	12901
title	11215
p_length	31
p_apage_no	1087
p_lang	13
author_name	5084
author_surname	6449

3. ในการทดสอบนั้นจะแบ่งการวัดประสิทธิภาพออกเป็น 3 ส่วนพิจารณาคือ

1) การ LOOK UP จะเป็นการ scan ตารางในรูปแบบต่างๆ ทั้งแบบ scan ดูข้อมูลทั้ง ตาราง และ scan แบบมีเงื่อนไข

2) การ INSERT จะเป็นการ insert ข้อมูล

3) การ UPDATE จะเป็นการทดสอบการแก้ไขข้อมูลและการลบข้อมูลออกจากตารางโดยการทดสอบจะมุ่งประเด็นในการพิจารณาที่ระดับการ lock ไม่ได้มุ่งที่เวลา

4. ในขั้นตอนการทดสอบจะเริ่มต้นด้วยการ create ตารางต่อด้วยการ copy ข้อมูลของตารางที่ 1 และ 2 ลงไปและขั้นสุดท้ายก็เป็นการเขียน query เพื่อทดสอบดูเวลาและประสิทธิภาพในด้านต่างๆ

5. ในการทดสอบนั้นจะทดสอบแบบมีตั้งแต่ 1 active user จนถึง 3 active user (ยกเว้น GUPTA ที่สามารถหา resource มาทดสอบได้มากสุดแค่ 2 active user)เพื่อเป็นการทดสอบ concurrency

Query และ จุดประสงค์ในการทดสอบ

1. Subquery

จุดประสงค์เป็นการใช้ Subquery กับตารางตารางเดียว ออกแบบเพื่อทดสอบประสิทธิภาพของระบบจัดการฐานข้อมูลในการทำlookup ตารางกับ records จำนวนมากๆ การทำ subquery เป็นการเพิ่มจำนวน records ที่เกี่ยวข้องให้มากๆขึ้น ระบบจัดการฐานข้อมูลจะต้องไล่ตารางมากกว่า 1 รอบเพื่อหาผลของ subquery แต่ละชั้น

```
Query :-  SELECT title, author_name, author_surname
          FROM   p_call_lib
          WHERE  p_call_no NOT LIKE 'QA%'
           AND author_name IN (
                  SELECT author_name
                  FROM   p_call_lib
                  WHERE  p_call_no IN (
                         SELECT p_call_no
                         FROM   p_call_lib
                         WHERE     p_call_no LIKE 'QA%'
                                  )
                                 }
          AND author_surname in (
```

```
                                    SELECT author_surname
                                    FROM   p_call_lib
                                    WHERE  p_call_no LIKE 'QA%'
                                    }
        ORDER BY title;
```

query นี้จะแสดง title author_name และ author_surname ที่เขียนหนังสือในหมวด QA% เขียนหนังสืออะไรในหมวดอื่นบ้าง

2. Semijoin (Join และ Subquery)

Semijoin คือ query ที่เงื่อนไขอยู่ในตารางหนึ่งแล้วได้ output อยู่อีกตารางหนึ่งโดยใช้ query แบบ join และ subquery

จุดประสงค์เพื่อทดสอบ response time ของ query ที่ได้ผลลัพธ์เดียวกันโดย query แบบ JOIN และแบบ SUBQUERY

Query_join :-

```
SELECT title
FROM p_call_lib,print_lib
WHERE p_call_lib.p_call_no = print_lib.p_call_no
    AND p_call_no LIKE 'QA%';
```

ซึ่งผลที่ได้จาก query ชุดนี้คือ

- การแสดงค่าใน column "title" จากตาราง p_call_lib โดยมีเงื่อนไขคือ ให้มีค่าของ column "p_call_no" ของตาราง p_call_lib เท่ากับ ค่าของ column "p_call_no" ของตาราง print_lib โดยเฉพาะ record ที่ค่าใน column "p_call_no" มีค่าขึ้นต้นด้วยอักษร 'QA'เช่น QA 76.6 D45 ,QA 73 B087 เป็นต้น ซึ่งเป็นการใช้วิธีการ JOIN ระหว่าง 2 ตาราง

Query_sub :-

```
SELECT title
FROM p_call_lib
WHERE p_call_no =
        ( SELECT p_call_no
          FROM  print_lib
          WHERE p_call_no LIKE 'QA%');
```

ซึ่งผลที่ได้จาก query ชุดนี้คือ

- การแสดงค่าใน column "title" จากตาราง p_call_lib โดยมีเงื่อนไขคือ

ให้มีค่าของ column "p_call_no" ของตาราง p_call_lib เท่ากับค่าของ column "p_call_no" ของตาราง print_lib โดยเฉพาะ record ที่ค่าใน column "p_call_no" มีค่าขึ้นต้นด้วยอักษร 'QA'เช่น QA 76.6 D45 ,QA 73 B087 เป็นต้น ซึ่งเป็นการใช้เงื่อนไขโดยใช้ SUBQUERY

3. Built_in function

1. จุดประสงค์ในการทดสอบคือ เป็นการทดสอบประสิทธิภาพในเรื่องการประมวลผล function ทางคณิตศาสตร์ ซึ่งจะเป็นการรวมเอาขบวนการเปรียบเทียบค่า integer และการ sequential search เข้ามาเกี่ยวข้องด้วย

```
Query 1:-    SELECT MAX(p_ad_yr)
             FROM   print_lib;
```

query นี้จะเป็นการหาว่าหนังสือเล่มใดพิมพ์หลังสุดคือการค่า maximum ของ p_ad_yr จากตาราง print_lib

2. จุดประสงค์ในการทดสอบคือ เป็นการทดสอบประสิทธิภาพในเรื่องการประมวลผล function ทางคณิตศาสตร์อีกแบบซึ่งจะเป็นรวมเอาขบวนการของการคำนวณค่า integer และการ sequential scan เข้ามาเกี่ยวข้องด้วย

```
Query 2:-    SELECT SUM(p_ad_yr)
             FROM   print_lib;
```

query นี้จะเป็นการทดสอบการหาค่าผลรวมทั้ง column โดยใช้ column p_ad_yr เป็น column ทดสอบ

4. Grouping Feature

จุดประสงค์เพื่อทดสอบคุณสมบัติ Grouping Feature ด้วยเงื่อนไขการมี index และการไม่มี index

```
Query :-
        { CREAT INDEX point_author
          ON p_call_lib.author_name }
        ( ในกรณีที่ทดสอบโดยมีเงื่อนไขคือมี index )

        SELECT author_name,COUNT(*)
        FROM   p_call_lib
        GROUP BY author_name;
```

ซึ่งผลที่ได้จาก query ชุดนี้คือ การแสดงค่าใน column "author_name"และ "จำนวนหนังสือของ author_name นั้น" จากตาราง p_call_lib

5. Syntax-Independent

จุดประสงค์ในการทดสอบคือ เป็นการทดสอบคุณสมบัติในการ query ว่าลำดับของเงื่อนไขใน WHERE clause จะมีผลอะไรกับเวลาที่ได้จากการ query หรือไม่ ก็คือเป็นการทดสอบดูว่า DBMS มี query optimization ที่จะทำหน้าที่ในการเลือกเส้นทางที่ดีที่สุดในการ access โดยไม่สนใจในลำดับของการ query หรือไม่ และ query optimizer จะทำการแปลง query ให้ไปอยู่ในรูปแบบกลางรูปแบบหนึ่งแล้วค่อยทำการ query หรือไม่

```
Query 1:-   SELECT *
            FROM   print_lib
            WHERE  p_ad_yr = 1985
                   AND p_publish_plc LIKE 'LONDON';
```

และ

```
Query 2:-   SELECT *
            FROM   print_lib
            WHERE  p_publish_plc LIKE 'LONDON'
                   AND p_ad_yr = 1985;
```

query นี้จะเป็นการหาว่ามีหนังสือเล่มใดบ้างที่พิมพ์ที่ LONDON ในปี 1985 โดย

- query ที่ 1 จะหาหนังสือทั้งหมดที่พิมพ์ในปี 1985 ก่อนแล้วค่อยพิจารณาต่อว่าเล่มใดพิมพ์ที่ LONDON

- query ที่ 2 จะหาหนังสือทั้งหมดที่พิมพ์ใน LONDON ก่อนแล้วค่อยพิจารณาต่อว่าเล่มใดพิมพ์ในปี 1985

6. Subquery with test for Existence

1. จุดประสงค์ เป็นการทำ subquery with test for existence เพื่อการทดสอบ speed ในการ look up ของระบบจัดการฐานข้อมูลโดยร่วมกับการใช้ exists แทนการใช้ฟังก์ชั่น IN ใน subquery ในข้อที่แล้วและเป็นการทำ subquery โดยใช้ 2 ตารางด้วย

```
Query :-
        SELECT DISTINCT title, author_name
        FROM   p_call_lib
        WHERE  EXISTS (
                SELECT *
```

```
                    FROM   print_lib
                    WHERE  p_ad_yr > 1980
                   )
        ORDER BY title;
```

query นี้จะแสดง ชื่อหนังสือ, ชื่อผู้แต่งสำหรับหนังสือที่แต่งหลังปี ค.ศ. 1980

2. จุดประสงค์ เป็นการทำ subquery with test for existence เช่นเดียวกับข้อที่แล้วแต่เป็นการ lookup ที่เมื่อทำเสร็จแล้วจะได้ผลออกมาเป็นเซ็ตว่าง เมื่อทำตามเงื่อนไขไปแล้วจะไม่พบกลุ่ม records ใดๆ ที่มีคุณสมบัติตามเงื่อนไขที่กำหนดเป็นการใช้ NOT EXIST

Query :-

```
SELECT title, author_name, author_surname
FROM   p_call_lib
WHERE  EXISTS (
                SELECT *
                FROM   print_lib
                WHERE  p_call_no LIKE 'POM'
               )
ORDER BY title;
```

query นี้จะแสดงชื่อหนังสือและชื่อผู้แต่งสำหรับหนังสือที่มี p_call_no เป็น 'POM"

7. Create view และ scan ข้อมูลจาก view

- Create view

Query:-

```
CREATE VIEW recent AS
SELECT *
FROM   print_lib;
```

การสร้าง view เป็นความสามารถอย่างหนึ่งของ DBMS จะเป็นการสร้าง image ตารางขึ้นมาซึ่ง view ที่ได้จะไม่ใช่ตารางที่มีอยู่จริง

จะเป็นการสร้าง view ที่ simple ที่สุด ไม่มีการ join ระหว่างตารางเนื่องจากเวลาทดสอบไม่ต้องการที่จะให้เวลาในการ join มามีผลกระทบกับการ create view นี้

query นี้จะเป็นการสร้าง view ของตาราง print_lib ขึ้นมา

- Scan ข้อมูลจากตารางจริงเปรียบเทียบกับ view

จุดประสงค์ในการทดสอบคือ เป็นการทดสอบคุณสมบัติในการ query ว่าเวลาที่ได้จากการ scan จากตารางจริงจะต่างจากการ scan จาก view อย่างไรซึ่งในการทดสอบจะใช้การ scan แบบไม่มีเงื่อนไขใดๆ เนื่องจากไม่ต้องการให้เกิดผลกระทบจากเงื่อนไขใน query นั้นและดูเวลาที่ได้ออกมาเปรียบเทียบกันว่าต่างกันหรือไม่ อย่างไร เพราะ view เป็นการสร้างมโนทัศน์ไม่ใช่ตารางที่มีอยู่จริงเหมือนการ copy ตาราง

Query:- SELECT *
FROM print_lib;
และ
SELECT *
FROM recent;

query นี้จะเป็นการ scan ค่าทั้งหมดออกมาจาก ตาราง print_lib

8. Concurrency Control

จุดประสงค์ เพื่อทดสอบคุณสมบัติ Concurrency Control เมื่อมีการแก้ไข

Query1 :-

```
UPDATE print_lib
SET    p_publish_plc = 'DK.'
WHERE  p_access_no LIKE 'QA%';
```

ซึ่งผลที่ได้จาก query ชุดนี้คือ

- การแก้ไขค่าใน column "p_publish_plc" ให้มีค่าเท่ากับ 'DK.'
เฉพาะ record ที่ ค่าใน column "p_access_no" มีค่าขึ้นต้นด้วยอักษร 'QA'
เช่น QA 76.6 D45 , QA 73 B087 เป็นต้น

Query2 :-

```
UPDATE print_lib
SET    p_publish_plc = 'DK.'
WHERE  p_access_no LIKE 'QC%';
```

ซึ่งผลที่ได้จาก query ชุดนี้คือ

- การแก้ไขค่าใน column "p_publish_plc" ให้มีค่าเท่ากับ 'DK.'
เฉพาะ record ที่ ค่าใน column "p_access_no" มีค่าขึ้นต้นด้วยอักษร 'QC'
เช่น QC 2340 D45 , QC 7583 K7 เป็นต้น

9. Delete

จุดประสงค์ในการทดสอบคือ เป็นการทดสอบที่มุ่งประเด็นการทดสอบไปที่การ lock ว่า DBMS มีการ lock ระดับใด

การ delete ชุดข้อมูล

```
Query1:-    DELETE FROM print_lib
            WHERE  p_call_no LIKE 'QA%';
Query2:-    DELETE FROM print_lib
            WHERE  p_call_no LIKE 'QC%';
Query3:-    DELETE FROM print_lib
            WHERE  p_call_no LIKE 'TK%';
```

เป็นการทดสอบที่มุ่งประเด็นการทดสอบไปที่การ lock ว่า DBMS มีการ lock ระดับใดโดยแบ่งการทดสอบดังนี้

- ทดสอบทีละ active user โดยใช้ query ข้างต้นดูว่า delete เงื่อนไขที่ p_call_no = 'QA%' หรือ 'QC%' หรือ 'TK%' ทีละ query ว่าได้เวลาเป็นอย่างไร (ในการ delete แต่ละ query ต้องใช้จากตารางที่สมบูรณ์เท่านั้น จะนำตารางที่เคยถูก delete ไปแล้วมา delete ซ้ำไม่ได้เพราะจะได้เวลาที่คลาดเคลื่อนไปมาก)

- ทดสอบ 2 active user โดยให้ delete ตาราง เดียวกันคือ print_lib โดย active user ที่ 1 จะ delete โดยใช้ query ที่ 1 และ active user ที่ 2 จะใช้ query ที่ 2 และพิจารณาจากเวลาที่ได้ว่ามีการ lock ระดับใด

- ทดสอบ 3 active user โดยให้ delete ตาราง เดียวกันคือ print_lib โดย active user ที่ 1 จะ delete โดยใช้ query ที่ 1 และ active user ที่ 2 จะใช้ query ที่ 2 และ active user ที่ 3 จะใช้ query ที่ 3 และพิจารณาจากเวลาที่ได้ว่ามีการ lock ระดับใด

8.2 การทดสอบ ORACLE

ในสภาพแวดล้อมที่ทำการทดสอบจะประกอบไปด้วย

- ORACLE version 5 แบบ Single-host
- บนเครื่อง NIXDROF TARGON model 5 มี RAM 8 M.
- CPU ของ Motorola 68030
- มี dumb terminal เป็น PC

ซึ่งขั้นตอนในการทดสอบ performance ของ ORACLE นั้นจะมีลำดับขั้นตอนดังนี้

1. การสร้างฐานข้อมูล

ทำได้โดยการใช้คำสั่ง "CREATE DATABASE "

```
SQL>CREATE DATABASE test
```

2. การสร้างตารางของฐานข้อมูลที่มีอยู่

สร้างตาราง "print_lib" ซึ่งมี column ต่างๆ ดังนี้

- p_access_no CHAR(15)
- p_status_cd CHAR(2)
- p_publish_plc CHAR(70)
- p_be_yr INT4
- p_ad_yr INT4
- p_call_no CHAR(21)

และตาราง "p_call_lib"

- p_call_no CHAR(21)
- title CHAR(60)
- p_publish CHAR(4)
- p_length CHAR(6)
- p_apage_no CHAR(70)
- p_lang CHAR(8)
- author_name CHAR(20)
- author_surname CHAR(30)

โดยใช้คำสั่ง SQL>CREATE TABLE print_lib

```
(
        p_access_no     CHAR(15)
```

```
                p_status_cd      CHAR(2)
                p_publish_plc    CHAR(70)
                p_be_yr          INT4
                p_ad_yr          INT4
                p_call_no        CHAR(21)
              )
```

เป็นต้น

3.การใส่ข้อมูลลงในตาราง

โดยการเขียนโปรแกรม Embedded SQL โดยใช้ PRO * C เป็น compiler ซึ่งตัวอย่างของ program การใส่ data ลงในตาราง p_call_lib จะเป็นดังต่อไปนี้

```
/******* EXAMPLE OF LOAD TEXT FILE TO TABLE IN DATABASE
**********/
#include <stdio.h>
#include <string.h>
#include <time.h>
EXEC SQL BEGIN DECLARE SECTION;
var char p_call_no[21];
var char title[60];
var char p_publish[70];
var char p_length[4];
var char p_apage_no[6];
var char p_lang[8];
var char author_name[20];
var char author_surname[30];
var char uid[20];
var char pwd[20];
EXEC SQL END SECTION;
EXEC SQL INCLUDE SQLCA;
main()
{
  FILE *fp;
  char txt[59];
```

```
int  i=0;
strcpy (uid.arr,"pom");
uid.len = strlen(uid.arr);
strcpy (pwd.arr,"pom");
pwd.len=strlen(pwd.arr);
EXEC SQL WHENEVER SQLERR GOTO errt;
EXEC SQL CONNECT:uid IDENTIFIED BY:pwd;
printf("CONNECTED TO ORACLE AS USER :%s \n",uid.arr);
if((fp=fopen("data1.doc","r"))==NULL)
{
  printf("FILE CANNOT BE OPENED \n");
  exit(0);
}
fgets(txt, ,fp);
do
{
  strncpy(p_call_no.arr,txt,15);
  strncpy(title.arr,&txt[16],60);
  strncpy(p_length.arr,&txt[77],4);
  strncpy(p_apage_no.arr,&txt[82],6);
  strncpy(p_publish.arr,&txt[89],70);
  strncpy(p_lang.arr,&txt[160],8);
  strncpy(author_name.arr,&txt[160],20);
  strncpy(author_surname.arr,&txt[183],30);
  p_call_no.len      = strlen(p_call_no.arr);
  title.len          = strlen(title.arr);
  p_length.len       = strlen(p_length.arr);
  p_apage_no.len     = strlen(p_apage_no.arr);
  p_publish.len      = strlen(p_publish.arr);
  p_lang.len         = strlen(p_lang.arr);
  author_name.len    = strlen(author_name.arr);
  author_surname.len = strlen(author_surname.arr);
```

```
    EXEC SQL INSERT INTO P_CALL_LIB
(P_CALL_NO,TITLE,P_PUBLISH,
                                    P_LENGTH,P_APAGE_NO,P_LANG,
                                    AUTHOR_NAME,AUTHOR_SURNAME)
            VALUES(:p_call_no,:title,:p_publish,:p_length,:p_apage_no,
                    :p_lang,:author_name,:author_surname);
    EXEC SQL COMMIT WORK;
    i++;
    fgets(txt,,fp);
  } while (!feof(fp));
fclose(fp);
EXEC SQL WHENEVER SQLERR CONTINUE;
EXEC SQL COMMIT WORK RELEASE;
exit(1);
printf("\n %70s",sqlca.sqlerrm.sqlrmc);
EXEC SQL ROLLBACK WORK RELEASE;
exit(1);
}
```

4. การทำ query

4.1 รัน query ที่ SQL prompt เช่น

```
SQL>select * from p_call_lib;
```

4.2 เขียน query ใส่ file ซึ่งต้องมีนามสกุล .SQL และนำมา รัน ภายหลัง เช่น เขียน query ลงใน file "testjoin.sql" และ รัน โดย

```
SQL>@testjoin
```

5.การจับเวลาของ query

โดยการ set time on ที่ SQL prompt และ set term off เพื่อไม่ให้มีการแสดงผลออกหน้าจอเพื่อให้เวลาที่ออกมาถูกต้อง

8.3 Query และ ผลการทดสอบของ ORACLE

1. Subquery 3 ชั้น

Query :-
```
SELECT title, author_name, author_surname
FROM   p_call_lib
WHERE  p_call_no NOT LIKE 'QA%'
       AND author_name IN (
                SELECT author_name
                FROM   p_call_lib
                WHERE  p_call_no in (
                       SELECT p_cll_no
                       FROM   p_call_lib
                       WHERE    p_call_no LIKE 'QA%'
                              )
                             }
        AND author_surname in (
                SELECT author_surname
                FROM   p_call_lib
                WHERE  p_call_no LIKE 'QA%'
                              )
ORDER BY title;
```

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	259	434	451	758	707	675

วิเคราะห์และสรุปผลการทดสอบ

เนื่องจาก query นี้เป็นการ lookup ตารางโดยมีการค้นหาตารางมากกว่า 1 รอบ

เพราะใช้ subquery lock mode ในตารางเป็น S-lock mode การทำงานพร้อมกันหลายๆ active user ทำให้ CPU time ต้องถูก share จึงได้ response time ที่สูงขึ้น

2. JOIN and SUBQUERY

Query_join :-

```
SELECT title
FROM p_call_lib,print_lib
WHERE p_call_lib.p_call_no = print_lib.p_call_no
    AND p_call_no LIKE 'QA%';
```

Query_sub :-

```
SELECT title
FROM p_call_lib
WHERE p_call_no =
        ( SELECT p_call_no
          FROM  print_lib
          WHERE p_call_no LIKE 'QA%');
```

ผลที่การทดสอบ

number of active user	1		2				3					
	active user 1		active user 1		active user 2		active user 1		active user 2		active user 3	
	sub query	join	sub query	join	sub query	join	sub query	join	sub query	join	sub query	join
time (sec.)	72	72	180	269	190	264	286	381	288	431	288	430

วิเคราะห์ผลการทดสอบ

- ในสภาวะที่มีผู้ใช้เพียงคนเดียวนั้น

เมื่อทดสอบ query_join โดยใช้วิธีการ JOIN ใช้เวลาประมวลผลประมาณ 72 วินาที

เมื่อทดสอบ query_sub โดยใช้วิธีการ SUBQUERYใช้เวลาประมวลผลประมาณ 72 วินาที ซึ่งเห็นได้ว่าเท่ากัน ดังนั้นในสภาวะที่มีผู้ใช้เพียงคนเดียวนั้นในกรณีquery เพื่อให้ได้ผลลัพธ์เดียวกันแต่ใช้วิธีการ JOIN และวิธีการ SUBQUERY นั้นใช้เวลาประมวลผลเท่ากัน

- ในสภาวะที่มีผู้ใช้ 2 และ 3 คน

เมื่อทดสอบ query_join โดยใช้วิธีการ JOIN จะใช้เวลาประมวลผลมากกว่าเมื่อทดสอบ query_sub โดยใช้วิธีการ SUBQUERY ซึ่งผลต่างของเวลาในการประมวลผลจะเพิ่มขึ้น

สรุปผลการทดสอบ

- ตามหลักของการออกแบบ DBMS ของ Codd นั้นในกรณี query ที่ได้ผลลัพธ์เหมือนกันแต่ใช้วิธี query และวิธีการเลือกเส้นทางที่ ต่างกัน ควรจะได้ response time เดียวกันซึ่ง ORACLE ไม่มีคุณสมบัติข้อนี้

3. Built_in function

Query1:- SELECT MAX(p_ad_yr)
FROM print_lib;

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	29	35	35	38	38	38

Query2:- SELECT SUM(p_ad_yr)
FROM print_lib;

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	31	33	33	47	47	47

วิเคราะห์ผลการทดสอบ และ สรุปผลการทดสอบ

เมื่อมีหลาย active user ขึ้นนั้น เวลาในการประมวลผล table ก็จะเพิ่มขึ้นตามไปด้วยและเวลาในการคำนวณทั้ง 2 แบบก็ไม่ต่างกันมากนัก แต่ถ้านำไปเปรียบเทียบกับ INGRES แล้วจะพบว่าผลที่ได้นั้น INGRES จะค่อนข้างดีกว่ามาก

4. Grouping Feature

Query :-

(ในกรณีที่ทดสอบโดยมีเงื่อนไขคือมี index)

```
{ CREAT INDEX point_author
  ON p_call_lib.author_name }

SELECT author_name,COUNT(*)
FROM   p_call_lib
GROUP BY author_name;
```

ผลที่การทดสอบ

number of active user	1		2				3					
	active user 1		active user 1		active user 2		active user 1		active user 2		active user 3	
	with no index	index	with no index	index	with no index	index	with no index	index	with no index	index	with no index	index
time (sec.)	63	40	83	60	85	63	122	78	126	80	126	82

วิเคราะห์และสรุปผลการทดสอบ

จากผลการทดสอบพบว่า เวลาที่ใช้ในการประมวล query ในกรณีที่มี index ชี้ไปที่ column "author_name" น้อยกว่า เวลาที่ใช้ในการประมวล query กรณีที่ไม่มี index ชี้ไปที่ column "author_name"

- จะเห็นได้ว่า เมื่อทดสอบ query นี้ในสภาวะผู้ใช้ 2,3 คนนั้น เวลาที่ใช้ในการประมวล query จะใช้เวลามากกว่า การทดสอบ query นี้ในสภาวะผู้ใช้คนเดียว

5. Syntax-Independent

```
Query1:-    SELECT *
            FROM   print_lib
            WHERE  p_ad_yr = 1985
```

AND p_publish_plc LIKE 'LONDON';

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	36	42	40	60	61	64

Query2:-

```
SELECT *
FROM   print_lib
WHERE  p_publish_plc LIKE 'LONDON'
       AND p_ad_yr = 1985;
```

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	30	31	31	39	42	44

วิเคราะห์ผลการทดสอบและสรุปผลการทดสอบ

ยิ่งเพิ่มจำนวนของ active user เข้าไปจะพบว่าผลต่างของเวลาจะยิ่งเพิ่มขึ้นอย่างเห็นได้ชัด ซึ่งทำให้สามารถสรุปได้ว่า ORACLE ขาดคุณสมบัติในด้าน syntax-independent หมายถึงลำดับของเงื่อนไขใน query มีผลต่อการทำ query แสดงว่า query optimizer ของ ORACLE ทำหน้าที่ทางด้านนี้ได้ไม่สมบูรณ์

6. Subquery with test for existence

Query1 :-

```
SELECT DISTINCT title, author_name
FROM   p_call_lib
WHERE  EXISTS (
                SELECT *
```

```
                    FROM   print_lib
                    WHERE  p_ad_yr > 1980
                    )
    ORDER BY title;
```

Query2 :- (Subquery with test for existence โดยผลออกมาเป็นเซ็ตว่าง)

```
    SELECT title, author_name, author_surname
    FROM   p_call_lib
    WHERE   EXISTS (
                    SELECT *
                    FROM   print_lib
                    WHERE  p_call_no LIKE 'POM'
                    )
    ORDER BY title;
```

Query3 :- (Subquery with test for not existence โดยผลออกมาเป็นเซ็ตว่าง)

```
    SELECT title, author_name, author_surname
    FROM   p_call_lib
    WHERE   NOT EXISTS (
                    SELECT *
                    FROM   print_lib
                    WHERE  p_call_no NOT LIKE 'POM'
                    )
    ORDER BY title;
```

ผลการทดสอบ

Query1 :-

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	142	242	251	364	448	447

Query2:-

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	139	202	185	283	288	287

Query3:-

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	38	43	44	66	68	71

วิเคราะห์และสรุปผลการทดสอบ

การทดสอบ query ชุดนี้ออกแบบขึ้นเพื่อทดสอบการ lookup ตารางโดยใช้ฟังก์ชั่น exists และฟังก์ชั่น not exists ซึ่งlock-mode ในตารางจึงเป็น s-lock จากผลการทดลองพบว่ายิ่งผลลัพธ์ของ query มีจำนวน row ยิ่งมาก response time จะยิ่งสูง และจะสังเกตได้ว่า การใช้ฟังก์ชั่น not exists จะได้ response time ที่ต่ำกว่าการใช้ฟังก์ชั่น exists มาก(ใน query เดียวกัน)จึงคาดได้ว่า ORACLE ใช้การค้นหาแบบ index ในฟังก์ชั่น not exists และใช้การค้นหาแบบ scan ในฟังก์ชั่น exists เช่นเดียวกับ INGRES

ข้อสังเกต response time ของ INGRES ในทุกๆ query จะต่ำกว่าของ ORACLE มากทั้งนี้เป็นเพราะ INGRES และ ORACLE ที่นำมาทดสอบนั้น run อยู่บนคนละ platform กันโดย INGRES ใช้ platform ที่มีประสิทธิภาพสูงกว่า

7. Create view และ scan ข้อมูลจาก view

1. Create view

Query:-

```
CREATE VIEW recent AS
SELECT *
FROM   print_lib;
```

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	2	2	3	3	4	4

วิเคราะห์ผลการทดสอบ

เนื่องจากการ create view ไม่ได้สร้างตารางขึ้นมาจริง ดังนั้นจึงพบว่าเวลาที่ใช้ในการ create จึงน้อยมากเมื่อเทียบกับ query แบบอื่น

2. Scan ข้อมูลจาก table จริง เปรียบเทียบกับ view

Query จาก table จริง :-

```
SELECT *
FROM   print_lib;
```

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	168	334	330	500	506	505

Query จาก view :-

```
SELECT *
FROM   recent;
```

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	169	331	332	512	514	520

วิเคราะห์ผลการทดสอบ และ สรุปผลการทดสอบ

เวลาที่ได้จากการ scan table จริงกับ scan จาก view นั้นถ้าเป็น active user เดียวจะพบว่าใกล้เคียงกันเนื่องจาก view ที่สร้างนั้น simple และเหมือนกับ table จริงทุกประการแต่เมื่อเพิ่ม active user เข้าไปก็จะเห็นว่ามีความแตกต่างกันโดย table ที่ได้จาก view นั้นจะใช้เวลาในการ scan มากกว่า table จริงอย่างแท้จริงเนื่องจากใน query นี้ไม่มีการใส่เงื่อนไขใดๆเข้าไปที่อาจจะก่อให้เกิดความคลาดเคลื่อนได้

8. Update

Query1 :-

```
UPDATE print_lib
SET    p_publish_plc = 'DK.'
WHERE  p_access_no LIKE 'QA%';
```

Query2 :-

```
UPDATE print_lib
SET    p_publish_plc = 'DK.'
WHERE  p_access_no LIKE 'QC%';
```

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	43	121	78	241	206	127

วิเคราะห์ผลการทดสอบ

- จะเห็นได้ว่าเมื่อทดสอบ query นี้ในสภาวะผู้ใช้ 2 คนนั้นเมื่อผู้ใช้คนหนึ่งทำการแก้ไขข้อมูลนั้น ผู้ใช้อีกคนก็ไม่สามารถแก้ไขข้อมูลได้เห็นได้จากเวลาซึ่ง
  - หากทดสอบ query1 โดยมีผู้ใช้เพียงคนเดียวใช้เวลาประมาณ 43 วินาที
  - หากทดสอบ query2 โดยมีผู้ใช้คนที่ 2 ใช้เวลาประมาณ 78 วินาทีและ ทดสอบ query1 โดยมีผู้ใช้คนที่ 1 ซึ่งต้องใช้เวลา 43 วินาที แต่รวมเวลาที่ถูก locksในขณะที่ผู้ใช้คนที่ 2 แก้ไขจึงใช้เวลารวม 78+43=121วินาที

สรุปผลการทดสอบ

- เมื่อพิจารณาเวลาที่ใช้ในการประมวล query พบว่าในสภาวะแบบ Multiuser กล่าวคือผู้ใช้มีมากกว่า 1 คนและผู้ใช้เหล่านั้นแก้ไขข้อมูลในตารางเดียวกัน ในช่วงเวลาเดียวกันจำเป็นต้องมี Concurrency Control ซึ่ง ORACLE ใช้กระบวนการ locks โดยประกันได้ว่าในขณะที่ Transaction ใด Transaction หนึ่งแก้ไขข้อมูลใดอยู่จะไม่มี Transaction อื่นใดสามารถเข้าใช้ข้อมูลเหล่านั้นได้ไม่ว่าชนิดของการเข้าใช้นั้นจะเป็น read หรือ write ก็ตามจนกว่า Transaction ที่แก้ไขข้อมูลนั้นจะ commit แล้ว

9. *Delete*

การ delete ชุดข้อมูล

```
Query1:-    DELETE FROM print_lib
            WHERE  p_call_no LIKE 'QA%';
Query2:-    DELETE FROM print_lib
            WHERE  p_call_no LIKE 'QC%';
Query3:-    DELETE FROM print_lib
            WHERE  p_call_no LIKE 'TK%';
```

ผลการทดสอบ

number of active user	1			2		3		
	active user 1	active user 1	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
	delete 'QA%'	delete 'QC%'	delete 'TK%'	delete 'QA%'	delete 'QC%'	delete 'TK%'	delete 'QA%'	delete 'QC%'
time (sec.)	40	74	124	108	75	124	238	205

วิเคราะห์ผลการทดสอบ

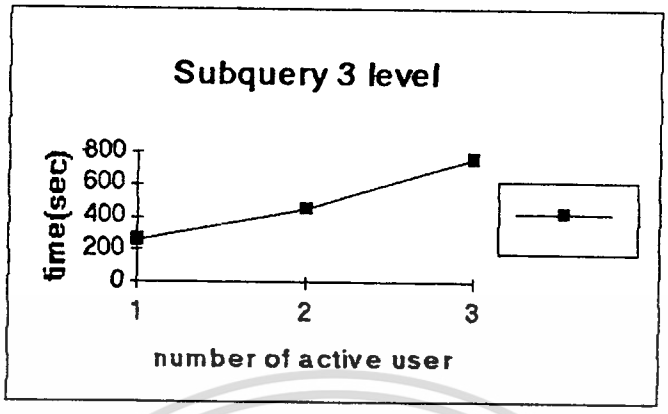
จากการทดสอบที่ใช้ 2 active user และให้ active user ที่ 1 ทำก่อนแล้วตามด้วย active user ที่ 2 ไปติดๆกัน พิจารณาจากผลของเวลาจะพบว่า active user ที่ 1 จะได้ทำก่อนจนเสร็จแล้วจึงตามด้วย active user ที่ 2 หรือถ้าเป็น 3 active user ก็จะได้ว่า active user ที่ 1 จะได้ทำก่อนตามด้วย active user ที่ 2 และ active user ที่ 3 จะได้ทำเป็นคนสุดท้าย

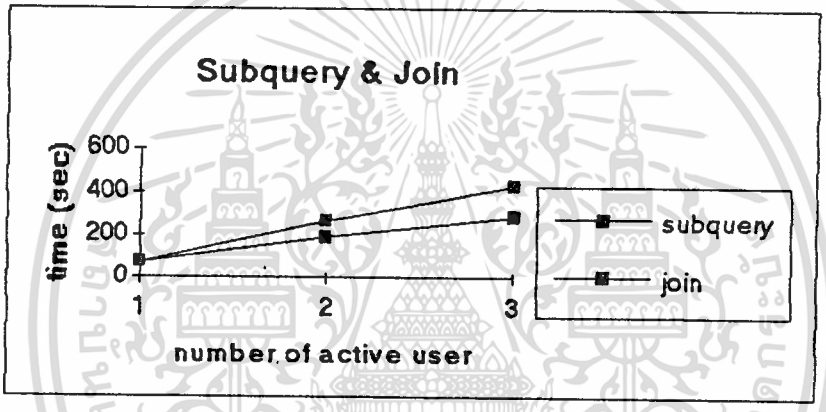
สรุปผลการทดสอบ

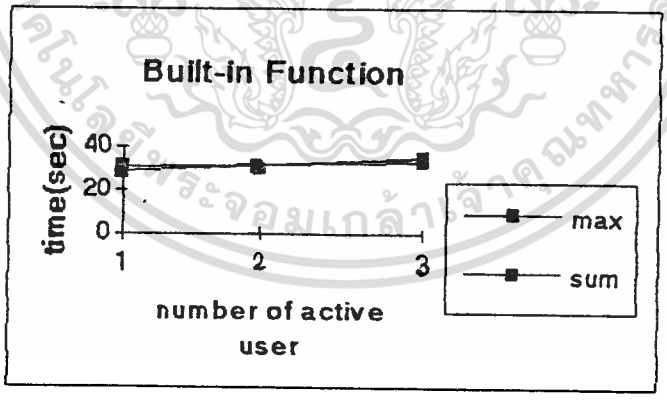
จากผลการทดสอบทำให้สามารถสรุปได้ว่าในการ access ตารางเดียวกัน

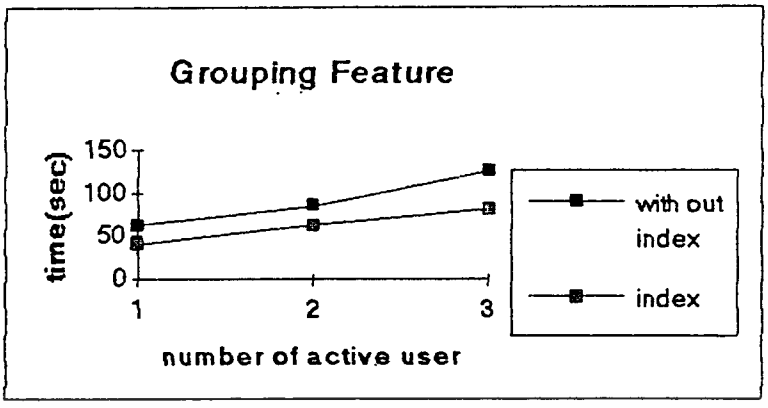
พร้อมๆกันจากหลายๆ user นั้น ถ้าเป็นการ delete แล้ว ORACLE จะทำการ lock ระดับ table เอาไว้ไม่ให้ access พร้อมกันหลายๆ user โดยจะ lock table ให้ user ที่ได้เข้าไปก่อนได้ทำจนเสร็จแล้วจึงจะ release lock ให้ table อื่นได้ทำต่อไป

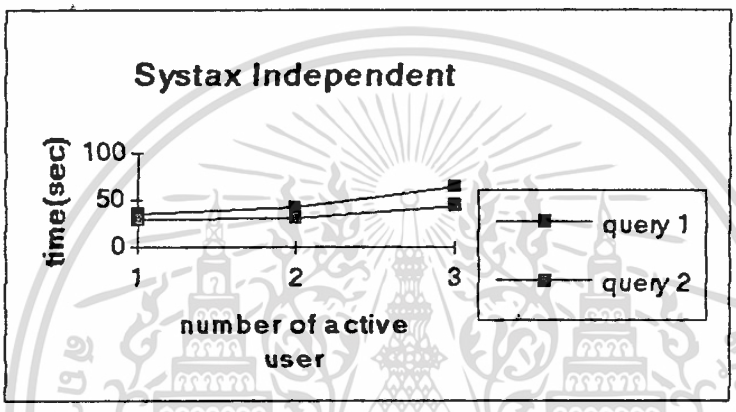
8.4 กราฟแสดงผลการทดสอบของ ORACLE V.5





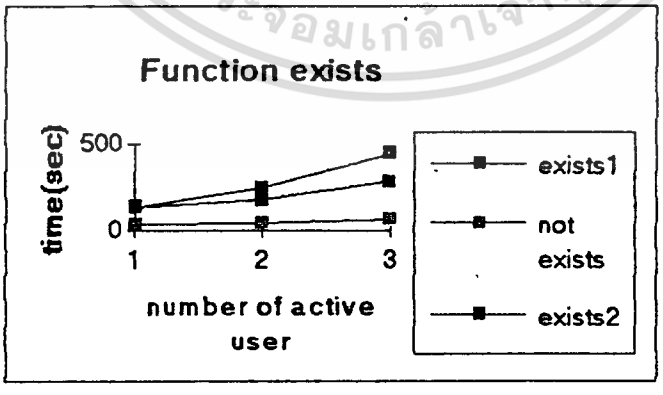






Query 1: เป็น query ซึ่งมีเงื่อนไข where p_ad_yr = 1985 and p_publish_plc = LONDON

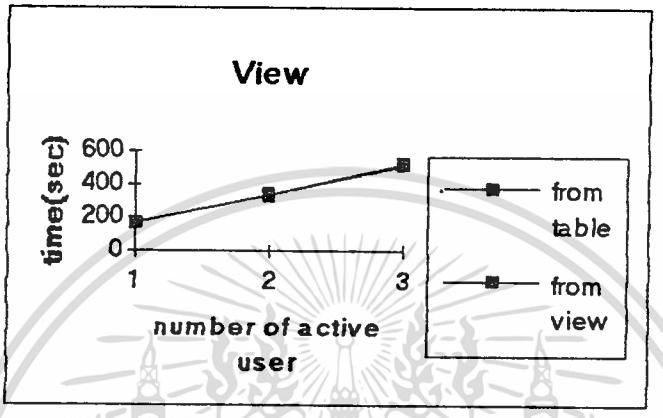
Query 2: เป็น query ซึ่งมีเงื่อนไข where p_publish_plc = LONDON and p_ad_yr = 1985

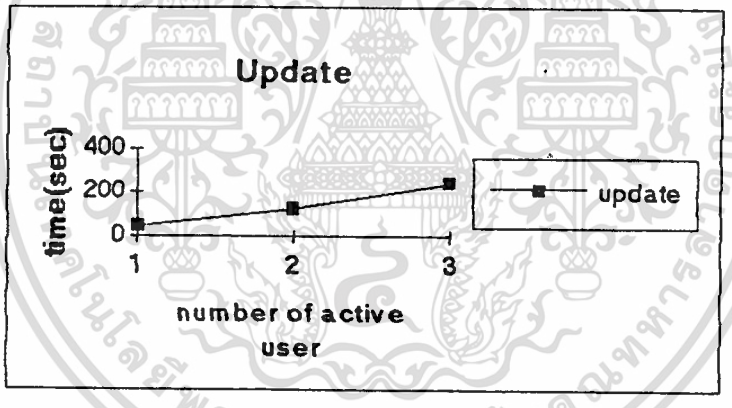


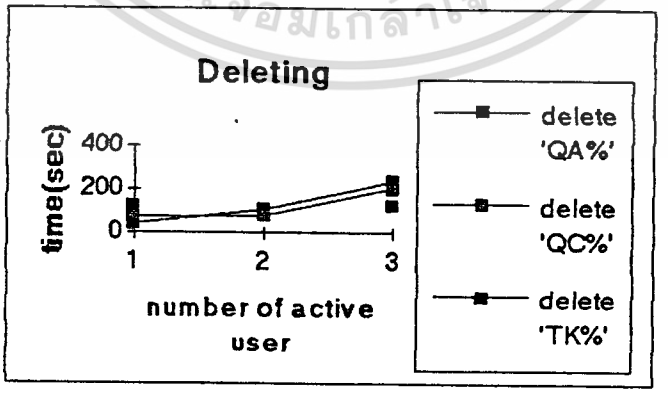
exist1 :- เป็นการทำ subquery โดยใช้ function exist

exist2 :- เป็นการทำ subquery โดยใช้ function exist แล้วได้ผลลัพธ์เป็น set ว่าง

exist3 :- เป็นการทำ subquery โดยใช้ function not exist แล้วได้ผลลัพธ์เป็น set ว่าง







8.5 การทดสอบ INGRES

ในระบบที่ทดสอบนี้สภาพแวดล้อมที่ใช้ทดสอบเป็นดังนี้

- เป็นระบบแบบ host base ซึ่งใช้เครื่อง HP 9000/827 เป็น host และมี terminal ต่อออกมาเป็น dump terminal
- ระบบจัดการฐานข้อมูลที่ใช้ทดสอบคือ INGRES Release 6.4
- host มีหน่วยความจำ 48 M
- Operating system เป็น HP UX9
- เนื้อที่ hard disk 1.36 GB
- Floating Point Co-processor
- host เชื่อมต่อ user ได้ 32 คน
- 24-port MUX

1. การสร้างฐานข้อมูลทำได้โดยการใช้คำสั่ง

   prompt> createdb dbname

2. การสร้างตารางของ database ที่มีอยู่ ทำได้โดย
   - prompt> ingmenu dbname
   - หลังจากเข้ามาใน menu ของ ingres แล้วก็เลือก TABLES
   - ใช้คำสั่ง Create เพื่อสร้างตารางใหม่
   - กำหนดชื่อตารางและ field รวมทั้ง data type ลงไป
3. การใส่ข้อมูลลงในตารางทำได้โดย
   - เลือกคำสั่ง Query ใน ingmenu
   - เลือกคำสั่ง Append ถ้าต้องการจะใส่ record ใหม่ลงในตารางหรือ
   - เลือกคำสั่ง Retrieve ถ้าต้องการดูค่าข้อมูลในตารางเท่านั้น
   - เลือกคำสั่ง Update ถ้าต้องการแก้ไขข้อมูลในตาราง
4. การ copy ข้อมูลจาก file ที่เก็บข้อมูลของตารางที่มีอยู่แล้วที่ต้องการจะนำมาลง ทำได้โดยการใช้คำสั่ง copy ตารางของ ISQL ซึ่งมีขั้นตอนดังนี้
   - create ตารางขึ้นมาเพื่อตั้งชื่อตารางและกำหนด field ก่อน
   - เข้า ISQL เพื่อใช้ SQL statement ทำการ copy ตารางดังนี้

```
COPY TABLE (tablename)
        { fieldname1 = datatype1,
          fieldname2 = datatype2,
                :          :
          fieldnameN = datatypeN nl}
```

FROM (filename);

5. การเข้า ISQL เพื่อใช้ SQL statement การใช้คำสั่ง SQL ในการทดสอบต่างๆวิธีหนึ่งที่สะดวกและเป็น tool ที่ INGRES มีเตรียมไว้ให้สามารถเข้าไปใช้ได้ดังนี้

- เรียก ingmenu แล้วเลือกไปที่ Queries หรือเข้าจาก prompt โดยตรงก็ได้โดยใช้คำสั่ง

prompt> ISQL dbname

- หลังจากนั้นหน้าจอก็จะขึ้นเป็นจอว่างๆให้ enter SQL statement ลงไป

6. การ run SQL statement สามารถใช้ได้ 2 คำสั่งคือ Go และ Complete

- Go มีลักษณะคือเมื่อ INGRES ทำการประมวลผลได้หน้าจอหนึ่งแล้วก็จะ return output ออกมาทางหน้าจอให้เลย ไม่ต้องรอจนประมวลผลเสร็จสมบูรณ์

- Complete มีลักษณะคือ INGRES จะ run และรอจนการประมวลผลทุกอย่างเสร็จสมบูรณ์มีการ return ผลกลับมาที่ INGRES แล้วเรียบร้อย จึงจะแสดง output ออกหน้าจอมา

7. การจับเวลาของ INGRES จะดูค่าของเวลาและรายละเอียดจากผลที่ INGRES เก็บอยู่ใน dbmsinfo ซึ่งค่าใน dbmsinfo จะมีค่าของสภาวะของ INGRES ในขณะหนึ่งขณะใดไว้ ซึ่งในการวัดผลก็ทำได้จากการ Select ค่าที่เก็บใน dbmsinfo ก่อนและหลังการ query นำมาลบกันออกมาซึ่งจะทำให้ได้เวลาที่ INGRES ใช้ไป

8.6 Query และ ผลการทดสอบของ INGRES

1. Subquery 3 ชั้น

```
Query :- SELECT title, author_name, author_surname
         FROM   p_call_lib
         WHERE  p_call_no NOT LIKE 'QA%'
             AND author_name IN (
                     SELECT author_name
                     FROM   p_call_lib
                     WHERE  p_call_no in (
                             SELECT p_cll_no
                             FROM   p_call_lib
                             WHERE    p_call_no LIKE 'QA%'
                                    )
                                  )
             AND author_surname IN (
                     SELECT author_surname
                     FROM   p_call_lib
                     WHERE  p_call_no LIKE 'QA%'
                                  )
         ORDER BY title;
```

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	35	75	75	102	101	101

วิเคราะห์และสรุปผลการทดสอบ

เนื่องจาก query นี้เป็นการค้นหาเพียงอย่างเดียวโดยไม่มีการแก้ไขเปลี่ยนแปลงข้อมูลในตาราง การ lock ตารางจึงอยู่ใน mode share lock (Slock) การ Share จึงเกิด

ขึ้นได้เสมอ ซึ่งจาผลการทดลองจะเห็นได้ว่า ในการทดสอบการทำงานแบบ concurrency แต่ละจอจะใช้เวลาในการประมวลผลใกล้เคียงกันมากนั่นคือสามารถทำงานไปได้พร้อมกันไม่ต้องมี transaction ใดหยุดรอและการที่ 2 active user ใช้เวลามากกว่า 1 active user ก็สรุปได้ว่าเพราะ CPU ต้อง share เวลาให้หลาย transaction ดังนั้น response time จึงช้าลง

2. Compare between JOIN and SUBQUERY

Query_join :-

```
SELECT title
FROM p_call_lib,print_lib
WHERE p_call_lib.p_call_no = print_lib.p_call_no
        AND p_call_no LIKE 'QA%';
```

Query_sub :-

```
SELECT title
FROM p_call_lib
WHERE p_call_no =
        ( SELECT p_call_no
          FROM  print_lib
          WHERE p_call_no LIKE 'QA%');
```

ผลการทดสอบ

number of active user	1		2				3					
	active user 1		active user 1		active user 2		active user 1		active user 2		active user 3	
	sub query	join	sub query	join	sub query	join	sub query	join	sub query	join	sub query	join
time (sec.)	40	39	49	45	49	45	95	62	96	62	100	58

วิเคราะห์ผลการทดสอบ

- ในสภาวะที่มีผู้ใช้เพียงคนเดียวนั้น

เมื่อทดสอบ query_join โดยใช้วิธีการ JOIN ใช้เวลาประมวลผลประมาณ 39 วินาที เมื่อทดสอบ query_sub โดยใช้วิธีการ SUBQUERYใช้เวลาประมวลผลประมาณ 40 วินาทีซึ่งเห็นได้ว่าใกล้เคียงกันมาก ดังนั้นในสภาวะที่มีผู้ใช้เพียงคนเดียวนั้น query เพื่อ

ให้ได้ ผลลัพธ์ เดียวกันแต่ใช้วิธีการ JOIN และวิธีการ SUBQUERY นั้นใช้เวลาประมวลผลใกล้เคียงกันมาก

- ในสภาวะที่มีผู้ใช้ 2 และ 3 คน

เมื่อทดสอบ query_join โดยใช้วิธีการ JOIN จะใช้เวลาประมวลผลน้อยกว่าเมื่อทดสอบ query_sub โดยใช้วิธีการ SUBQUERY ซึ่งผลต่างของเวลาในการประมวลผลจะเพิ่มขึ้นตามจำนวนของผู้ใช้

สรุปผลการทดสอบ

- ตามหลักของการออกแบบ DBMS ของ Codd นั้นในกรณี query ที่ได้ผลลัพธ์เหมือนกันแต่ใช้วิธี query และวิธีการเลือกเส้นทางที่ ต่างกัน ควรจะได้ response time เดียวกันซึ่ง INGRES ไม่มีคุณสมบัติข้อนี้

3. **Built_in function**

Query1:- SELECT MAX(p_ad_yr)
FROM print_lib;

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	2	3	3	4	5	4

วิเคราะห์ผลการทดสอบ

เมื่อมีหลาย active user ขึ้นนั้น เวลาในการประมวลผลตารางก็เพิ่มมากขึ้นด้วย แต่ในอัตราส่วนที่น้อยมากและจะเห็นว่าช่วงเวลาของการ scan ตารางและเปรียบเทียบค่าจำนวน 20,000 กว่า record ของ INGRES นั้นใช้เวลาน้อยมาก

Query2:- SELECT SUM(p_ad_yr)
FROM print_lib;

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	2	3	3	3	3	2

วิเคราะห์ผลการทดสอบ

เมื่อมีหลาย active user ขึ้นนั้นเวลาในการประมวลผลตารางก็แทบจะไม่เพิ่มขึ้นหรือเพิ่มก็ด้วยอัตราส่วนที่น้อยมากไม่ถึงวินาทีในการ scan ตารางและคำนวณค่า sum จำนวน 20,000 กว่า record

สรุปผลการทดสอบ

การทดสอบประสิทธิภาพในเรื่องการประมวลผลฟังก์ชั่นทางคณิตศาสตร์ทั้งแบบเปรียบเทียบและหาค่าผลรวมนี้ใช้เวลาน้อยมากและเวลาจากการคำนวณทั้ง 2 แบบนี้ค่อนข้างจะใกล้เคียงกันมากและใช้เวลาน้อยเมื่อเทียบกับการ scan ใน query อื่นซึ่งทำให้คาดว่าใน function ที่นำมาทดสอบ 2 แบบนี้อาจจะมีการใช้ค่า statistics ที่เก็บใน catalog มาช่วยในการประมวลผลด้วย

4. Grouping Feature

Query :-

(ในกรณีที่ทดสอบโดยมีเงื่อนไขคือมี index)

```
{ CREAT INDEX point_author
  ON p_call_lib.author_name }

SELECT author_name,COUNT(*)
FROM   p_call_lib
GROUP BY author_name;
```

ผลการทดสอบ

number of active user	1		2				3					
	active user 1		active user 1		active user 2		active user 1		active user 2		active user 3	
	with no index	index	with no index	index	with no index	index	with no index	index	with no index	index	with no index	index
time (sec.)	20	5	25	11	29	11	42	16	43	16	41	16

วิเคราะห์และสรุปผลการทดสอบ

- จากผลการทดสอบพบว่า เวลาที่ใช้ในการประมวล query ในกรณีที่มี index ชี้ไปที่ column "author_name" น้อยกว่า เวลาที่ใช้ในการประมวล query กรณีที่ไม่มี index ชี้ไปที่ column "author_name"
- จะเห็นได้ว่า เมื่อทดสอบ query นี้ในสภาวะที่มีผู้ใช้ 2,3 คนนั้นเวลาที่ใช้ในการ

ประมวล query จะใช้เวลามากกว่า การทดสอบ query นี้ในสภาวะผู้ใช้คนเดียว

5. Syntax-Independent

Query1:-
```
SELECT *
FROM   print_lib
WHERE  p_ad_yr = 1985
       AND p_publish_plc LIKE 'LONDON';
```

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	24	32	32	107	107	107

Query2:-
```
SELECT *
FROM   print_lib
WHERE  p_publish_plc LIKE 'LONDON'
       AND p_ad_yr = 1985;
```

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	24	26	26	90	90	90

วิเคราะห์ผลการทดสอบและสรุปผลการทดสอบ

จากเวลาที่ได้จาก query นี้จะพบว่า ถ้าทดสอบเพียงแค่ active user เดียว นั้นเวลาที่ได้จาก query ทั้ง 2 แบบจะไม่แตกต่างกันเลย แต่เมื่อเพิ่มจำนวน active user ขึ้นจะพบว่าเวลาจาก query 2 แบบที่สลับตำแหน่งของเงื่อนไขในการquery นั้นจะได้ผลของเวลาออกมาไม่เท่ากัน ยิ่งเพิ่มจำนวนของ active user เข้าไปจะพบว่าผลต่างของเวลาจะยิ่งเพิ่มขึ้นอย่างเห็นได้ชัด ซึ่งทำให้สามารถสรุปได้ว่า INGRES ขาดคุณสมบัติ

ในด้าน syntax-independent หมายถึงลำดับของเงื่อนไขในquery มีผลต่อการทำ query แสดงว่า query optimizer ของ INGRES ทำหน้าที่ทางด้านนี้ได้ไม่สมบูรณ์

6. Subquery with test for existence

Query1 :-

```
SELECT DISTINCT title, author_name
FROM   p_call_lib
WHERE  EXISTS (
                SELECT *
                FROM   print_lib
                WHERE  p_ad_yr > 1980
               )
ORDER BY title;
```

Query2 :- (Subquery with test for existence โดยผลออกมาเป็นเซ็ตว่าง)

```
SELECT title, author_name, author_surname
FROM   p_call_lib
WHERE  EXISTS (
                SELECT *
                FROM   print_lib
                WHERE  p_call_no LIKE 'POM'
               )
ORDER BY title;
```

Query3 :- (Subquery with test for not existence โดยผลออกมาเป็นเซ็ตว่าง)

```
SELECT title, author_name, author_surname
FROM   p_call_lib
WHERE  NOT EXISTS (
                SELECT *
                FROM   print_lib
                WHERE  p_call_no NOT LIKE 'POM'
               )
ORDER BY title;
```

ผลการทดสอบ

Query1:-

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	28	49	49	89	82	82

Query2:-

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	<1	<1	<1	<1	<1	<1

Query3:-

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	26	36	56	58	58	59

วิเคราะห์และสรุปผลการทดสอบ

การทดสอบ query ชุดนี้ออกแบบขึ้นเพื่อทดสอบการ search ตารางของระบบจัดการฐานข้อมูลโดยใช้ฟังก์ชั่น exists ทั้งแบบที่ค้นหาแล้วพบข้อมูลกับค้นหาแล้วได้ผลเป็นเซ็ตว่าง transaction ทั้งหมดเป็นการ lookup เพราะฉะนั้น lock-mode ในตารางจึงเป็นs-lock จากผลการทดลองจะเห็นว่า ใน transaction ที่ hit ratio ต่ำๆ จะได้ response time ที่ต่ำด้วยใน query แรกจะใช้เวลาค่อนข้างมากเมื่อเทียบกับ 2 query หลังเป็นเพราะจำนวนrow ที่เป็น set ของคำตอบมีมากจึงใช้เวลามากสำหรับ 2 query เป็น

query ที่ให้คำตอบเหมือนกันคือเป็นเซ็ตว่างแต่จะสังเกตได้ว่า เวลาจะต่างกันมาก คือ การใช้ฟังก์ชั่น not exists จะใช้เวลาน้อยคือไม่ถึง 1 วินาที ส่วน query ที่ใช้ฟังก์ชั่น exists แม้จะเป็น query ที่ให้ผลลัพธ์เดียวกันแต่ใช้เวลามากกว่ามาก จึงคาดได้ว่า ใน INGRES ฟังก์ชั่น not exists จะใช้ index ในการค้นหาในขณะที่ฟังก์ชั่น exists ใช้การค้นหาแบบ Scan

7. Create view และ scan ข้อมูลจาก view

1. Create view

Query:-
```
CREATE VIEW recent AS
SELECT *
FROM   print_lib;
```

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	1	1	1	1	1	2

วิเคราะห์ผลการทดสอบ

เนื่องจากการ create view ไม่ได้สร้างตารางขึ้นมาจริง ดังนั้นจึงพบว่าเวลาที่ใช้ในการ create จึงน้อยมาก

2. Scan ข้อมูลจากตารางจริงเปรียบเทียบกับ view

Query จากตารางจริง :-
```
SELECT *
FROM   print_lib;
```

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	75	124	124	99	99	99

Query จาก view :-

```
SELECT *
FROM   recent;
```

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	75	128	128	135	135	190

วิเคราะห์ผลการทดสอบ และ สรุปผลการทดสอบ

เวลาที่ได้จากการ scan ตารางจริงกับ scan จาก view นั้นถ้าเป็น active user เดียวจะพบว่าใกล้เคียงกันเนื่องจาก view ที่สร้างนั้น simple และเหมือนกับตารางจริงทุกประการแต่เมื่อเพิ่ม active user เข้าไปก็จะเห็นว่ามีความแตกต่างกันโดยตารางที่ได้จาก view นั้นจะใช้เวลาในการ scan มากกว่าตารางจริงอย่างแท้จริงเนื่องจากใน query นี้ไม่มีการใส่เงื่อนไขใดๆเข้าไปที่อาจจะก่อให้เกิดความคลาดเคลื่อนได้

8. Update

Query1 :-

```
UPDATE print_lib
SET    p_publish_plc = 'DK.'
WHERE  p_access_no LIKE 'QA%';
```

Query2 :-

```
UPDATE print_lib
SET    p_publish_plc = 'DK.'
WHERE  p_access_no LIKE 'QC%';
```

ผลการทดสอบ

number of active user	1	2		3		
	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
time (sec.)	31	47	16	31	59	47

วิเคราะห์ผลการทดสอบ

- จะเห็นได้ว่าเมื่อทดสอบ query นี้ในสภาวะ 2 ผู้ใช้นั้นเมื่อผู้ใช้คนหนึ่งทำการแก้ไขข้อมูลนั้น ผู้ใช้อีกคนก็ไม่สามารถแก้ไขข้อมูลได้เห็นได้จากเวลาซึ่ง
  - หากทดสอบ query1 โดยมีผู้ใช้เพียงคนเดียวใช้เวลาประมาณ 31 วินาที
  - หากทดสอบ query2 โดยมีผู้ใช้คนที่ 2 ใช้เวลาประมาณ 16 วินาทีและทดสอบ query1 โดยมีผู้ใช้คนที่ 1 ซึ่งต้องใช้เวลา 31 วินาที แต่รวมเวลาที่ถูก locks ในขณะที่ผู้ใช้คนที่2 แก้ไขจึงใช้เวลารวม 16+31=47 วินาที

สรุปผลการทดสอบ

- เมื่อพิจารณาเวลาที่ใช้ในการประมวล query พบว่าในสภาวะแบบ Multiuser กล่าวคือมีผู้ใช้มากกว่า 1 คนและผู้ใช้เหล่านั้นแก้ไขข้อมูลในตารางเดียวกันในเวลาพร้อม ๆ กันจำเป็นต้องมี Concurrency Control ซึ่ง INGRES ใช้กระบวนการ locks โดยประกันได้ว่าในขณะที่ Transaction ใด Transaction หนึ่งแก้ไขข้อมูลใดอยู่จะไม่มี Transaction อื่นใดสามารถเข้าใช้ข้อมูลเหล่านั้นได้ไม่ว่าชนิดของการ access นั้นจะเป็น read หรือ write ก็ตาม จนกว่า Transaction ที่แก้ไขข้อมูลนั้นจะ commit แล้ว

9. Delete

การ delete ชุดข้อมูล

```
Query1:-   DELETE FROM print_lib
           WHERE  p_call_no LIKE 'QA%';
Query2:-   DELETE FROM print_lib
           WHERE  p_call_no LIKE 'QC%';
Query3:-   DELETE FROM print_lib
           WHERE  p_call_no LIKE 'TK%';
```

ผลการทดสอบ

number of active user	1			2		3		
	active user 1	active user 1	active user 1	active user 1	active user 2	active user 1	active user 2	active user 3
	delete 'QA%'	delete 'QC%'	delete 'TK%'	delete 'QA%'	delete 'QC%'	delete 'TK%'	delete 'QA%'	delete 'QC%'
time (sec.)	6	13	17	5	17	20	27	38

วิเคราะห์ผลการทดสอบ

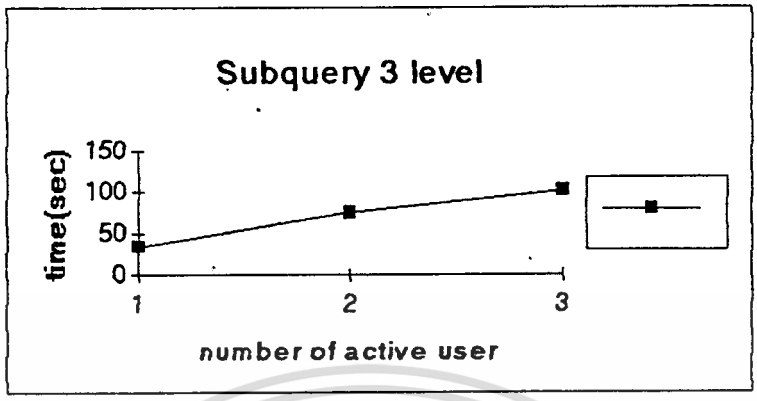
จากการทดสอบที่ใช้ 2 active user และให้ active user ที่ 1 ทำก่อนแล้วตามด้วย active user ที่ 2 ไปติดๆกัน พิจารณาจากผลของเวลาจะพบว่า active user ที่ 1 จะได้ทำก่อนจนเสร็จซึ่งใช้เวลาใกล้เคียงกับการdelete QA โดยมีแค่ active user เดียวแล้วหลังจากนั้น active user ที่ 2 จึงจะได้ทำเมื่อนำเวลาจาก active user ที่ 2 มาหักลบกับ active user ที่ 1 จะได้ 12 วินาที ซึ่งก็ใกล้เคียงกับเวลา 13 วินาทีที่ได้จากการ delete QC ในตารางแรก

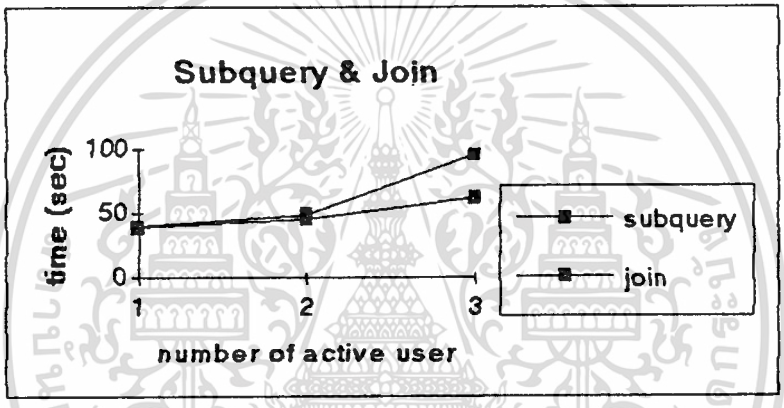
เมื่อเพิ่มจำนวน user เป็น 3 active users ก็จะเห็นได้ว่าการทำงานนั้นจะเริ่มด้วยการ delete TK ก่อนแล้วตามด้วย QA แล้วค่อยทำ QC

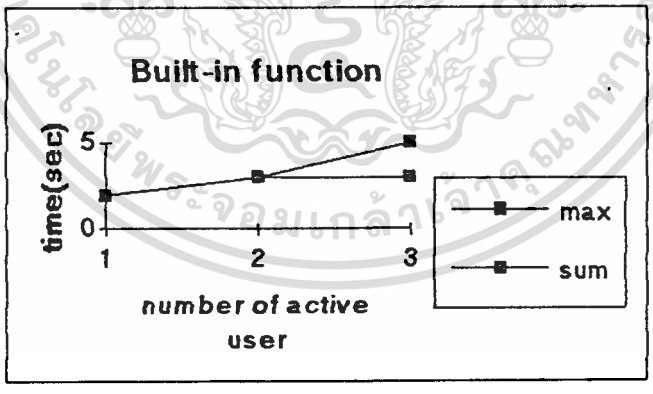
สรุปผลการทดสอบ

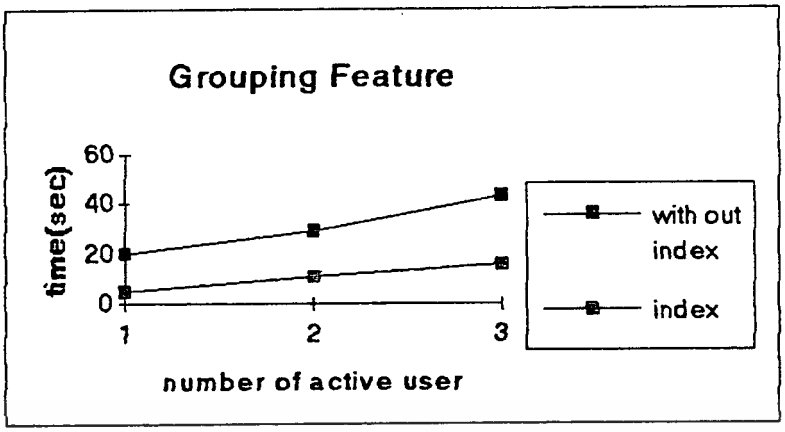
จากผลการทดสอบทำให้สามารถสรุปได้ว่าในการaccess ตารางเดียวกันพร้อมๆกันจากหลายๆ user นั้นถ้าเป็นการ delete แล้ว INGRES จะทำการ lock ระดับตารางเอาไว้ไม่ให้ access พร้อมกันหลายๆ user โดยจะ lock ตารางให้ user ที่ได้เข้าไปก่อนได้ทำจนเสร็จแล้วจึงจะ release lock ให้ตารางอื่นได้ทำต่อไป

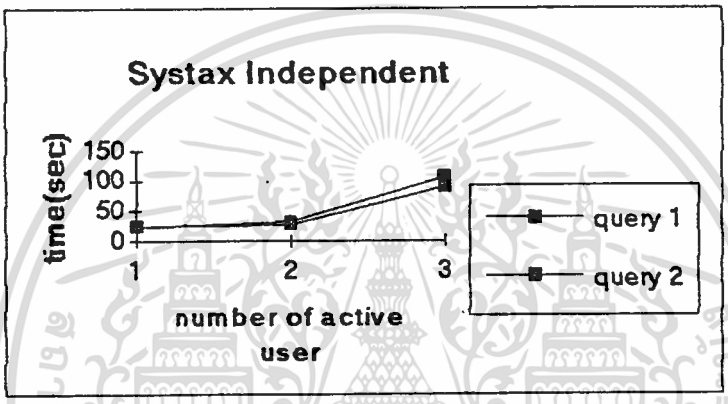
8.7 กราฟแสดงผลการทดสอบของ INGRES R.6.4





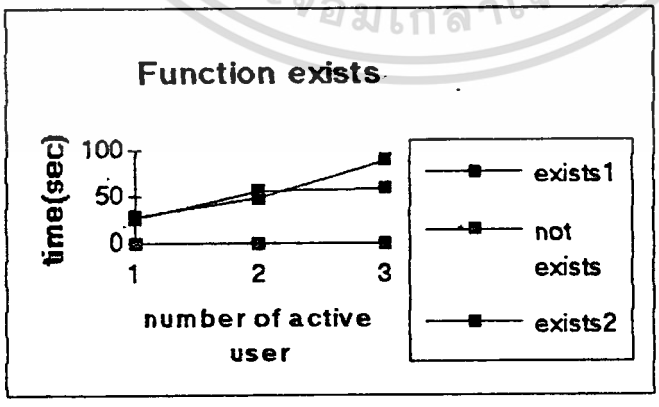






Query 1: เป็น query ซึ่งมีเงื่อนไข where p_ad_yr = 1985 and p_publish_plc = LONDON

Query 2: เป็น query ซึ่งมีเงื่อนไข where p_publish_plc = LONDON and p_ad_yr = 1985



exist1 :- เป็นการทำ subquery โดยใช้ function exist

8.8 การทดสอบ GUPTA

สภาวะแวดล้อมของระบบที่นำมาทดสอบเป็นดังนี้

- เป็นสถาปัตยกรรมในการประมวลผลข้อมูลแบบ client/server
- ขนาดของหน่วยความจำของ server = 16 M
- ความจุของ hard disk ของ server = 240 M

ระบบจัดการฐานข้อมูล GUPTA มี SQLTalk ซึ่งเป็นหน้าจอสำหรับรับคำสั่งภาษา SQL SQLTalk มีทั้งที่ใช้บนสภาพแวดล้อมแบบวินโดวส์และบนดอสในการทดสอบครั้งนี้ใช้ SQLTalk/Windows ขั้นตอนแรกคือการสร้างตารางและ load ข้อมูลลงไปโดยใช้คำสั่ง

```
create table print_lib
  (
      p_access_no CHAR(15)
      p_status_cd CHAR(2)
      p_publish_plc CHAR(70)
      p_be_yr  INT
      p_ad_yr  INT
      p_call_no CHAR(21)
  );

create table p_call_lib
  (
       p_call_no CHAR(21)
       title  CHAR(60)
      p_publisher  CHAR(4)
      p_length CHAR(6)
      p_apage_no CHAR(70)
      p_lang  CHAR(8)
      author_name  CHAR(20)
      author_surname CHAR(30)
  );
```

และ load ข้อมูลลงตารางโดยใช้คำสั่ง

```
insert into print_lib values(:1,:2,:3,:4,:5,:6)
  \
   file ข้อมูล
  /
```

บน input windows ของ SQLTalk

จากนั้นก็ทำการทดสอบจับเวลา query ต่างๆโดยการใส่คำสั่ง set time on; ลงใน input windows ซึ่ง time เป็นค่าของเวลาที่ใช้ในการรัน query โดยค่า time นี้จะไม่รวมเวลาในการแสดงผลบน output windows เมื่อเสร็จสิ้นการประมวลผลแล้วจะมีการ return ค่าเวลากลับมาเป็นเวลาในหน่วยวินาที

8.9 Query และ ผลการทดสอบของ GUPTA

1. Subquery 3 ชั้น

```
Query :- SELECT title, author_name, author_surname
         FROM   p_call_lib
         WHERE  p_call_no NOT LIKE 'QA%'
              AND author_name IN (
                   SELECT author_name
                   FROM   p_call_lib
                   WHERE  p_call_no in (
                        SELECT p_cll_no
                        FROM   p_call_lib
                        WHERE     p_call_no LIKE 'QA%'
                                  }
                                  )
              AND author_surname in (
                        SELECT author_surname
                        FROM   p_call_lib
                        WHERE  p_call_no LIKE 'QA%'
                                  )
         ORDER BY title;
```

2. Compared between JOIN and SUBQUERY

```
Query_join :-
         SELECT title
         FROM p_call_lib,print_lib
         WHERE p_call_lib.p_call_no = print_lib.p_call_no
              AND p_call_no LIKE 'QA%';
Query_sub :-
         SELECT title
         FROM p_call_lib
         WHERE p_call_no =
                   ( SELECT p_call_no
```

```
FROM  print_lib
WHERE p_call_no LIKE 'QA%');
```

ผลที่การทดสอบ

number of active user	1		2			
	active user 1		active user 1		active user 2	
	sub query	join	sub query	join	sub query	join
time (sec.)		89.80		121.60		121.66

วิเคราะห์และสรุปผลการทดสอบ

- เมื่อทดสอบ query_join โดยใช้วิธีการ JOIN ใช้เวลาประมวลผลประมาณ 89.80 วินาที แต่เมื่อทดสอบ query_sub โดยใช้วิธีการ SUBQUERYใช้ปรากฏว่าไม่สามารถทำการทดสอบได้เพราะเมื่อ run query แล้วพบว่าไม่ได้ผลลัพธ์จึงทำให้ไม่ทราบเวลาที่ใช้ในการประมวลผล ซึ่งคาดว่า ลักษณะการประมวลผลนั้นเป็นลักษณะการค้นหาข้อมูลอย่างไม่มีจุดจบจึงทำให้ไม่มีผลลัพธ์แสดงให้เห็น

3. **Built_in function**

```
Query1:-    SELECT MAX(p_ad_yr)
            FROM   print_lib;
```

ผลการทดสอบ

number of active user	1	2	
	active user 1	active user 1	active user 2
time (sec.)	12.85	16.69	16.58

วิเคราะห์ผลการทดสอบ

เมื่อมีหลาย active user ขึ้นนั้น เวลาในการประมวลผลตารางก็เพิ่มมากขึ้นด้วย

2.
```
SELECT SUM(p_ad_yr)
FROM   print_lib;
```

ผลการทดสอบ

number of active user	1	2	
	active user 1	active user 1	active user 2
time (sec.)	14.06	18.84	14.33

วิเคราะห์ผลการทดสอบ

เมื่อมีหลาย active user ขึ้นนั้นเวลาในการประมวลผลตารางก็พิ่มขึ้น

สรุปผลการทดสอบ

การทดสอบประสิทธิภาพในเรื่องการประมวลผลฟังก์ชั่นทางคณิตศาสตร์ทั้งแบบเปรียบเทียบและหาค่าผลรวมนี้ใช้เวลาน้อยเมื่อเทียบกับ query แบบอื่นๆ และเวลาจากการคำนวณทั้ง 2 แบบนี้ค่อนข้างจะใกล้เคียงกัน

4. Grouping Feature

Query :-

(ในกรณีที่ทดสอบโดยมีเงื่อนไขคือมี index)

```
{  CREAT INDEX point_author
   ON p_call_lib.author_name  }

SELECT author_name,COUNT(*)
FROM   p_call_lib
GROUP BY author_name;
```

ผลที่การทดสอบ

number of active user	1		2			
	active user 1		active user 1		active user 2	
	with no Index	Index	with no Index	Index	with no Index	Index
time (sec.)	39.72	0.38	63.83	0.94	63.33	1.05

วิเคราะห์และสรุปผลการทดสอบ

- จากผลการทดสอบพบว่า เวลาที่ใช้ในการประมวล query ในกรณีที่มี index ชี้ไปที่ column "author_name" น้อยกว่า เวลาที่ใช้ในการประมวล query กรณีที่ไม่มี index ชี้ไปที่ column "author_name"
- จะเห็นได้ว่า เมื่อทดสอบ query นี้ในสภาวะผู้ใช้ 2 คนนั้น เวลาที่ใช้ในการประมวล query จะใช้เวลามากกว่า การทดสอบ query นี้ในสภาวะผู้ใช้คนเดียว

5. *Syntax-Independent*

Query1:-
```
SELECT *
FROM   print_lib
WHERE  p_ad_yr = 1985
       AND p_publish_plc LIKE 'LONDON';
```

ผลการทดสอบ

number of active user	1	2	
	active user 1	active user 1	active user 2
time (sec.)	10.38	14.06	13.76

Query2:-
```
SELECT *
FROM   print_lib
WHERE  p_publish_plc LIKE 'LONDON'
       AND p_ad_yr = 1985;
```

ผลการทดสอบ

number of active user	1	2	
	active user 1	active user 1	active user 2
time (sec.)	9.62	13.02	12.91

วิเคราะห์ผลการทดสอบและสรุปผลการทดสอบ

จากเวลาที่ได้จาก query นี้จะพบว่า ถ้าทดสอบเพียงแค่ active user เดียว นั้น เวลาที่ได้จาก query ทั้ง 2 แบบจะไม่แตกต่างกันเลย แต่เมื่อเพิ่มจำนวน active user ขึ้นจะพบว่าเวลาจาก query 2 แบบที่สลับตำแหน่งของเงื่อนไขในการquery นั้นจะได้ผล ของเวลาออกมาไม่เท่ากัน ยิ่งเพิ่มจำนวนของ active user เข้าไปจะพบว่าผลต่างของเวลา จะยิ่งเพิ่มขึ้นอย่างเห็นได้ชัด ซึ่งทำให้สามารถสรุปได้ว่า GUPTA ขาดคุณสมบัติ ในด้าน syntax-independent

6. Create view และ scan ข้อมูลจาก view

1. Create view

```
CREATE VIEW recent AS
SELECT *
FROM   print_lib;
```

ผลการทดสอบ

number of active user	1	2	
	active user 1	active user 1	active user 2
time (sec.)	0.48	0.55	0.75

วิเคราะห์ผลการทดสอบ

เนื่องจากการ create view ไม่ได้สร้างตารางขึ้นมาจริง ดังนั้นจึงพบว่าเวลาที่ ใช้ในการ create จึงน้อยมาก

2. Scan ข้อมูลจากตารางจริง เปรียบเทียบกับ view

```
SELECT *
FROM   print_lib;
```

ผลการทดสอบ

number of active user	1	2	
	active user 1	active user 1	active user 2
time (sec.)	892.0	843.6	880

```
SELECT *
FROM   recent;
```

ผลการทดสอบ

number of active user	1	2	
	active user 1	active user 1	active user 2
time (sec.)	894.2	860.2	876.6

วิเคราะห์ผลการทดสอบ และ สรุปผลการทดสอบ

เวลาที่ได้จากการ scanตารางจริงกับ scan จาก view นั้นถ้าเป็น active user เดียวจะพบว่าใกล้เคียงกันเนื่องจาก view ที่สร้างนั้น simple และเหมือนกับตารางจริงทุกประการแต่เมื่อเพิ่ม active user เข้าไปก็จะเห็นว่ามีความแตกต่างกันโดยตารางที่ได้จาก view นั้นจะใช้เวลาในการ scan มากกว่าตารางจริงอย่างแท้จริงเนื่องจากในquery นี้ไม่มีการใส่เงื่อนไขใดๆเข้าไปที่อาจจะก่อให้เกิดความคลาดเคลื่อนได้

7. Subquery with test for existence

Query1 :-

```
SELECT DISTINCT title, author_name
FROM   p_call_lib
WHERE  EXISTS (
                SELECT *
                FROM   print_lib
                WHERE  p_ad_yr > 1980
              )
ORDER BY title;
```

Query2 :- (Subquery with test for existence โดยผลออกมาเป็นเซ็ตว่าง)

```
SELECT title, author_name, author_surname
FROM   p_call_lib
WHERE  EXISTS (
                SELECT *
```

```
                FROM   print_lib
                WHERE  p_call_no LIKE 'POM'
               )
ORDER BY title;
```

Query3 :- (Subquery with test for not existence โดยผลออกมาเป็นเซ็ตว่าง)

```
SELECT title, author_name, author_surname
FROM   p_call_lib
WHERE  NOT EXISTS (
                SELECT *
                FROM   print_lib
                WHERE  p_call_no NOT LIKE 'POM'
               )
ORDER BY title;
```

วิเคราะห์และสรุปผลการทดสอบ

- เมื่อทดสอบ query โดยใช้วิธีการ SUBQUERYใช้ปรากฏว่า ไม่สามารถทำการทดสอบได้เพราะเมื่อ run query แล้วพบว่าไม่ได้ผลลัพธ์จึงทำให้ไม่ทราบเวลาที่ใช้ในการประมวลผลซึ่งคาดว่า ลักษณะการประมวลผลนั้นเป็นลักษณะการค้นหาข้อมูลอย่างไม่มีจุดจบจึงทำให้ไม่มีผลลัพธ์แสดงให้เห็น

8. Update

Query1 :-

```
UPDATE print_lib
SET    p_publish_plc = 'DK.'
WHERE  p_access_no LIKE 'QA%';
```

Query2 :-

```
UPDATE print_lib
SET    p_publish_plc = 'DK.'
WHERE  p_access_no LIKE 'QC%';
```

ผลการทดสอบ

number of active user	1	2	
	active user 1	active user 1	active user 2
time (sec.)	47.29	47.74	-

วิเคราะห์ผลการทดสอบ

- จะเห็นได้ว่าเมื่อทดสอบ query นี้ในสภาวะผู้ใช้ 2 คนนั้นเมื่อผู้ใช้คนหนึ่งทำการแก้ไขข้อมูลนั้นผู้ใช้อีกคนก็ไม่สามารถแก้ไขข้อมูลได้โดยทดสอบ query1 โดยมีผู้ใช้คนที่ 1 ซึ่งต้องใช้เวลา 47.29 วินาทีในขณะที่ผู้ใช้คนที่ 2 จะไม่สามารถแก้ไขข้อมูลได้เลย โดยจะแสดงข้อความ "system deadlock"

สรุปผลการทดสอบ

- เมื่อพิจารณาเวลาที่ใช้ในการประมวล queryพบว่าในสภาวะแบบMultiuser กล่าวคือ ผู้ใช้มีมากกว่า 1 คนและผู้ใช้เหล่านั้นแก้ไขข้อมูลในตารางเดียวกัน ในช่วงเวลาเดียวกัน จำเป็นต้องมี Concurrency Control ซึ่ง GUPTA ใช้กระบวนการ locks โดยประกันได้ว่าในขณะที่ Transaction ใด Transaction หนึ่งแก้ไขข้อมูลใดอยู่จะไม่มี Transaction อื่นใดสามารถเข้าใช้ข้อมูลเหล่านั้นได้ไม่ว่าชนิดของการเข้าใช้นั้นจะเป็น read หรือ write ก็ตามจนกว่า Transaction ที่แก้ไขข้อมูลนั้นจะ commit แล้ว

9. Delete

การ delete ชุดข้อมูล Query :-

```
1.  DELETE FROM print_lib
    WHERE  p_call_no LIKE 'QC%';
2.  DELETE FROM print_lib
    WHERE  p_call_no LIKE 'TK%';
```

ผลการทดสอบ

number of active user	1		2	
	active user 1	active user 1	active user 1	active user 2
	delete 'TK%'	delete 'QC%'	delete 'TK%'	delete 'QC%'
time (sec.)	33.94	93.21	-	93.21

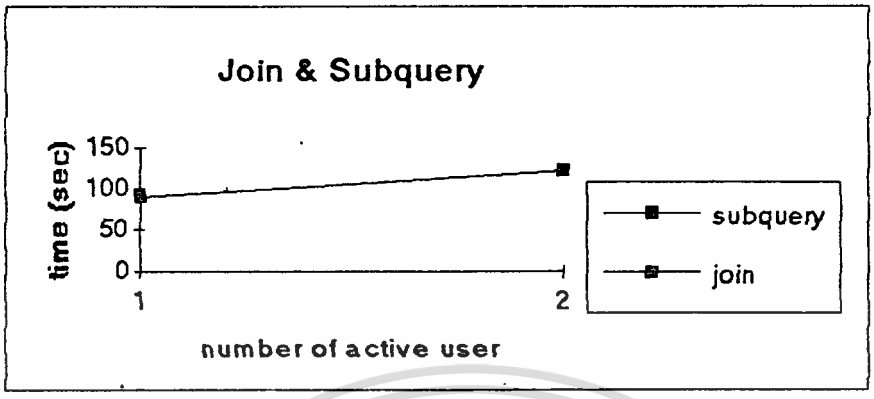
วิเคราะห์ผลการทดสอบ

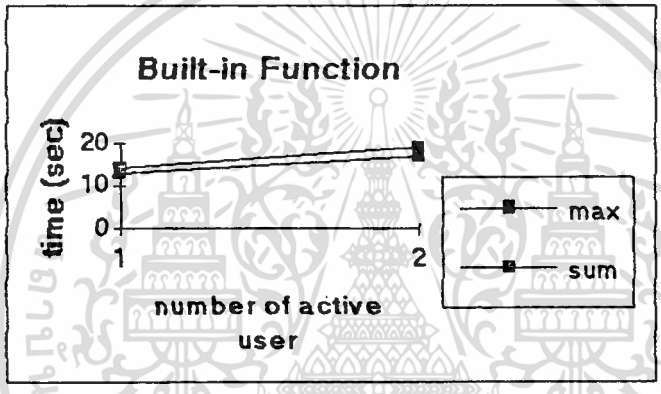
จากการทดสอบที่ใช้ 2 active user และให้ active user ที่ 1 ทำก่อนแล้วตามด้วย active user ที่ 2 ไปติดๆกัน พิจารณาจากผลของเวลาจะพบว่า active user ที่ 1 จะได้ทำจนเสร็จโดยที่ user อีกคนจะไม่ได้ run query เลยโดย GUPTA จะแสดงข้อความ "system deadlock"

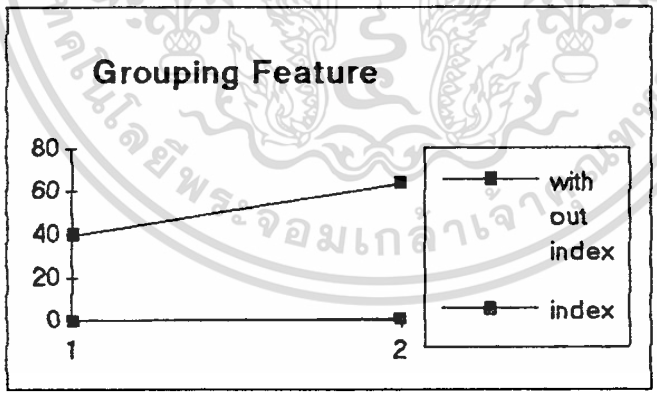
สรุปผลการทดสอบ

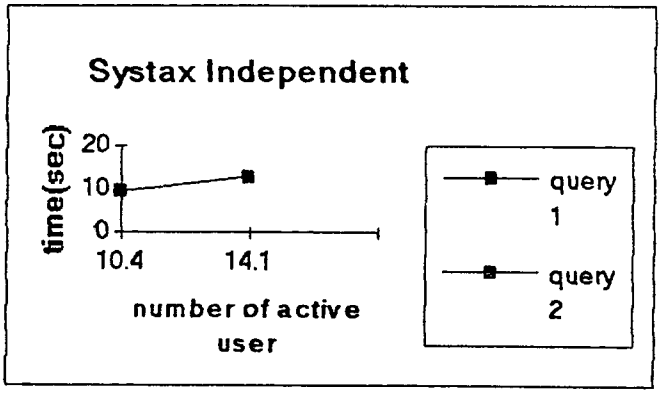
จากผลการทดสอบทำให้สามารถสรุปได้ว่าในการaccess ตารางเดียวกันพร้อมๆกันจากหลายๆ user นั้นถ้าเป็นการ delete แล้ว INGRES จะทำการ lock ระดับตารางเอาไว้ไม่ให้ access พร้อมกันหลายๆ user โดยจะ lock ตารางให้ user ที่ได้เข้าไปก่อนได้ทำจนเสร็จแล้วจึงจะ release lock ให้ตารางอื่นได้ทำต่อไป

8.10 กราฟแสดงผลการทดสอบของ GUPTA V.5



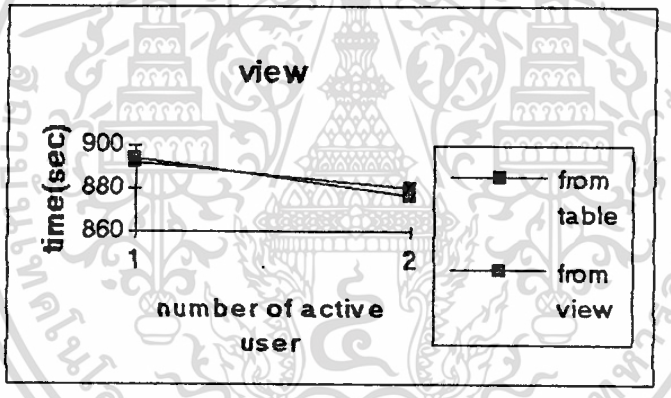


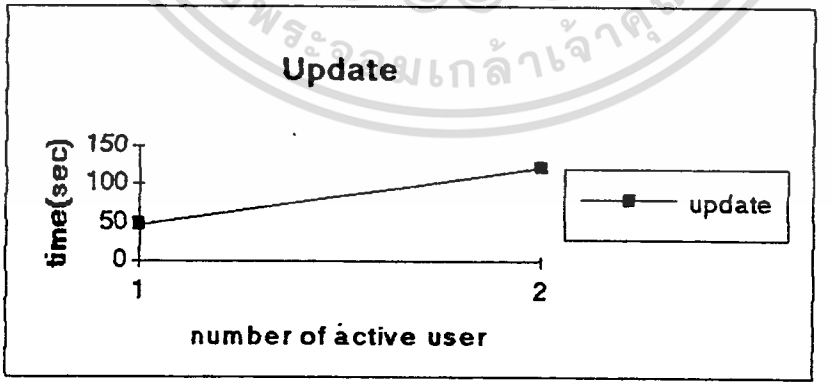


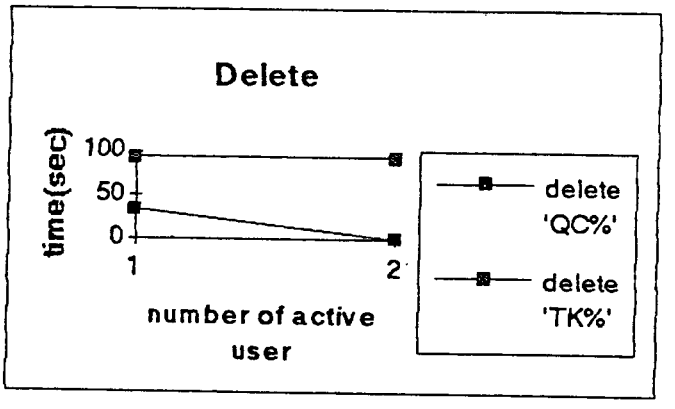


Query 1: เป็น query ซึ่งมีเงื่อนไข where p_ad_yr = 1985 and p_publish_plc = LONDON

Query 2: เป็น query ซึ่งมีเงื่อนไข where p_publish_plc = LONDON and p_ad_yr = 1985







บทที่ 9

บทสรุป

ในปริญญานิพนธ์ฉบับนี้ได้มีการรวบรวมข้อมูลที่ได้จากการศึกษาจากแหล่งต่างๆ เกี่ยวกับ database system environment ทางผู้จัดทำได้ทำการศึกษาเกี่ยวกับประเภทของ platform ว่ามีอะไรบ้างๆและแต่ละประเภทหมาะสมกับงานในลักษณะใด หลัง จากรวบรวมข้อมูลแล้วได้มีการสรุปรวบรวมข้อมูลเหล่านั้นลงในระบบผู้เชี่ยวชาญ โดยใช้ PC-plus expert system shell ซึ่งระบบผู้เชี่ยวชาญดังกล่าวจะให้คำปรึกษาสำหรับผู้ใช้เกี่ยวกับการเลือก platform ได้โดยใช้ลักษณะการถามชุดของคำถามจากผู้ใช้และรวบรวมคำตอบมาทำการตัดสินใจ

นอกจากนั้นผู้จัดทำยังได้ทำการศึกษา DBMS ชั้นนำในท้องตลาดปัจจุบันคือ ORACLE INGRES และ GUPTA ถึงลักษณะต่างทั้งทางด้านกายภาพ การใช้งาน และการจัดเก็บข้อมูลโดย หัวข้อที่ได้ทำการศึกษามีดังต่อไปนี้

1. โครงสร้างของระบบ
2. data independence
3. การทำ concurency control
4. การ back และการ recovery
5. data integrity
6. qury optimization
7. เครื่องมือต่างที่ DBMS จัดหาไว้ให้เช่น CASE Tool, Application development tool ฯลฯ

ซึ่งข้อมูลที่รวบรวมมานั้นผู้จัดได้ทำการศึกษาจาก manual ของผลิตภัณฑ์และพูดคุยกับบริษัทที่เป็นตัวแทนจะหน่ายผลิตภัณฑ์ทั้ง 3 หลังจากการศึกษาตามหัวข้อดังกล่าวแล้วทางผู้จัดทำได้สรุปคุณสมบัติต่างๆของระบบจัดการฐานข้อมูลโดยยึดตามกฎ 12 ข้อของ Codd และ กฎ 16 ข้อจาก RDBMS V.2

ผู้จัดทำได้ทำการทดสอบประสิทธิภาพในด้านต่างๆของระบบจัดการฐานข้อมูลทั้งสามคือ

1. ORACLE บน single-Host platform
2. INGRES บน single-Host platform
3. GUPTA บน Client/Server

โดยใช้ benchmark ซึ่งเป็นชุดของ query ที่ทางผู้จัดทำเขียนขึ้นทำการทดสอบร่วมกับข้อมูลซึ่งเป็นตารางของห้องสมุดคือ ตาราง print_lib (มี 22275 records)และตาราง p_call_lib (มี 12901 records)ในการทดสอบได้มีการเก็บข้อมูลเกี่ยวกับเวลาที่แต่ละระบบ

ละระบบจัดการฐานข้อมูลใช้ในการประมวลผล query ต่างๆและนำมาวิเคราะห์และสรุปผล ได้มีการแสดงผลของการทดสอบเป็นตารางและกราฟ เนื่องจากการทดสอบระบบจัดการฐานข้อมูลทั้งสามนั้นเป็นการทดสอบบนคนละ platform ซึ่งมีประสิทธิภาพต่างกันค่อนข้างมาก ผลที่ได้จึงไม่ได้นำมาแสดงบนตารางหรือกราฟเดียวกันเพื่อการเปรียบเทียบให้เห็นชัดเจนทั้งนี้เพราะผู้จัดทำเห็นว่าจะเป็นการไม่ยุติธรรมต่อบาง products ปัญหาที่เกิดขึ้นในการทำการทดสอบคือผู้จัดทำไม่สามารถหาplatform ที่เหมือนหรือให้เคียงกันเพื่อทำการทดสอบระบบจัดการฐานข้อมูลได้จึงไม่สามารถใช้ข้อมูลเวลาที่ได้มาในการตัดสินประสิทธิ์ภาพในเชิงเปรียบเทียบกันระหว่างระบบจัดการฐานข้อมูลทั้ง 3 ได้

หนังสืออ้างอิง

1. ASK/INGRES Technical Communication,"INGRES Database Administrator's Guide(UNIX)"
2. OARCLE SYSTEMS(Thailand),"ORACLE Database Administrator's Guide"
3. GUPTA Technologies, Incorporated, "SQL Base Version 5.0 Database Administrator Guide"
4. Valduriez, Patrick, "Analysis and comparison of relational database systems "
5. Robert V.Head; " A Guide to Package System "
6. Shaku Atre, " Database Structure Techniques for Design, Performance and Management second edition "
7. " Computer and the Planning Process Magazine "
8. Joe Salemi, " PC Magazine Guide to Client/Server Database "
9. Henry F.Korth,Abraham Silberschatz," Database System Concepts "
10. C.J.Date," An Introduction to Database Systems "

กิตติกรรมประกาศ

ขอขอบคุณบริษัท ไมโครเอกซ์ บริษัท ORACLE SYSTEMS THAILAND และบริษัท จักรวาลเทเลคอมมิวนิเคชั่น ที่ให้คำปรึกษาและข้อมูลต่างๆ เกี่ยวกับผลิตภัณฑ์

ขอขอบคุณธนาคารไทยพาณิชย์ จำกัด (สำนักงานใหญ่) ที่เอื้อเฟื้อเครื่องมือที่ใช้ในการทดสอบ GUPTA DBMS

ขอขอบคุณฝ่ายระบบสารสนเทศ คณะวิศวกรรมศาสตร์สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังที่เอื้อเฟื้อข้อมูลห้องสมุดที่ใช้ในการทดสอบ

ขอขอบคุณ ผศ. ดร. ศุภมิตร จิตตะยโศธร ที่ให้คำปรึกษาและแนะนำเกี่ยวกับปริญญานิพนธ์ฉบับนี้