บทที่ 1
บทนำ

1.1 ที่มาและปัญหา

ในปัจจุบันฐานข้อมูล (Database) ได้เข้ามามีบทบาทต่อองค์กรทั้งเล็กและใหญ่เพราะสามารถที่จะช่วยเพิ่มประสิทธิภาพให้แก่องค์กรได้ เมื่อมีการสอบถาม (Query) ข้อมูลเพื่อที่จะต้องการผลลัพธ์จากฐานข้อมูล ระบบจัดการฐานข้อมูลจะทำหน้าที่เป็นตัวกลางในการจัดการการสอบถามนั้น ระบบจัดการฐานข้อมูลมีมากมายให้เลือกใช้ในปัจจุบัน แต่ละตัวก็มีเทคนิคที่แตกต่างกันไปเล็กน้อย แต่โดยรวมแล้วก็มีการทำงานที่คล้ายๆ กันคือ ขั้นตอนแรก เมื่อมีการสอบถามเข้ามาเป็นภาษาของการสอบถาม เช่น ภาษาเอสคิวแอล (SQL Languages) ระบบจัดการฐานข้อมูลจะทำการส่งตัวแปรให้เช็คไวยากรณ์ของภาษาว่าภาษาการสอบถามที่เข้ามามีความถูกต้องหรือไม่ และตารางที่จะทำการสอบถามมีอยู่ในฐานข้อมูลหรือไม่ ต่อจากนั้นจะทำการแปลงภาษาการสอบถามเป็นพีชคณิตเชิงสัมพันธ์ และจะทำการส่งต่อให้แก่ ออพติไมเซอร์ (Optimizer) ออพติไมเซอร์จะทำการสร้างแผนประมวลผล (Execution Plan) หลายๆ แผนแล้วออพติไมเซอร์จะทำการเลือกแผนประมวลผลที่ดีที่สุดเพื่อทำการส่งต่อให้ทำตัวสืบค้นข้อมูลจากฐานข้อมูลและได้เป็นผลลัพธ์ที่ต้องการออกมา

โดยทั่วไปออพติไมเซอร์จะทำการสร้างและพิจารณาแผนประมวลผลของแต่ละการสอบถาม เพื่อทำการเลือกแผนประมวลผลที่มีประสิทธิภาพที่ดีที่สุด เพื่อให้การสอบถามสามารถถูกประมวลผลได้อย่างรวดเร็ว และถูกต้องตรงกับความต้องการของผู้ใช้ ทั้งนี้การประเมินจำนวนแถว (Cardinality) ของแต่ละความสัมพันธ์ (Relation) ที่เกี่ยวข้องกับการสอบถามนั้นถือได้ว่าเป็นกระบวนการที่สำคัญที่สุดกระบวนการหนึ่งในการสร้างและพิจารณาแผนประมวลผล เนื่องจากจำนวนแถวนั้นสามารถนำไปใช้ในการจัดการแผนประมวลผลได้ เช่น การเรียงลำดับเงื่อนไขในการประมวลผลการสอบถามหนึ่งๆ ซึ่งถ้าการประเมินจำนวนแถวนั้นมีการใช้เทคนิคที่มีประสิทธิภาพแล้ว แผนประมวลผลที่เป็นผลลัพธ์จากออพติไมเซอร์ก็มีแนวโน้มที่จะเป็นแผนที่มีประสิทธิภาพยิ่งขึ้น

ในปัจจุบันออพติไมเซอร์ใช้สถิติโดยรวมของข้อมูลแต่ละความสัมพันธ์ คือ วิธีฮิสโตแกรม (Histogram) [1] ในการหาจำนวนแถวซึ่งเมื่อมีเงื่อนไขหลายเงื่อนไขในการสอบถาม ออพติไมเซอร์อาศัยค่าผลรวมแบบง่ายๆ โดยการประมาณในการประเมินจำนวนแถวเพื่อที่จะใช้เวลาน้อยที่สุดเท่าที่จะสามารถเป็นไปได้ แต่ความเร็วนั้นอาจต้องแลกกับการที่ต้องเสียประสิทธิภาพของแผนประมวลผล ซึ่งเทคนิคกประเมินจำนวนแถวที่ใช้เวลาน้อยมักจะใช้การประมาณค่าจำนวนแถว ซึ่งไม่ใช่ค่าจำนวนแถวที่เป็นค่าจริง ซึ่งเป็นเทคนิคที่ใช้เวลาน้อยแต่ไม่มีประสิทธิภาพ

```
Student(ID,Name,Major,Gender)
Course(Code,Title,Instrutor,Textbook)
Enrolment(StID,CrsID,Grade)
```

รูปที่ 1.1 แสดงสกีมา (Schema) ของฐานข้อมูลตัวหนึ่ง

พิจารณารูปที่ 1.1 ซึ่งแสดงสกีมาของฐานข้อมูลหนึ่ง ซึ่งเป็นฐานข้อมูลที่ทำการเก็บการลงทะเบียนเรียนวิชาของนักศึกษาคณะวิศวกรรมศาสตร์ โดยจากรูปจะประกอบไปด้วยตาราง 3 ตาราง ซึ่งประกอบไปด้วย

ตาราง Student จะเป็นตารางที่เก็บประวัติของนักศึกษาซึ่งประกอบไปด้วยแอตทริบิวต์ (Attributes)

ID ทำหน้าที่เก็บ รหัสนักศึกษาของแต่ละคน

Name ทำหน้าที่เก็บ ชื่อของนักศึกษา

Major ทำหน้าที่เก็บ สาขาวิชาของนักศึกษาคนนั้น

Gender ทำหน้าที่เก็บ เพศของนักศึกษาคนนั้น

ตาราง Course เป็นตารางที่เก็บข้อมูลของวิชาที่ทำการเปิดสอนในคณะวิศวกรรมศาสตร์ซึ่งประกอบไปด้วยแอตทริบิวต์

Code ทำหน้าที่เก็บ รหัสวิชาที่ทำการเปิดสอน

Title ทำหน้าที่เก็บ ชื่อวิชาที่ทำการเปิดสอน

Instructor ทำหน้าที่เก็บ ชื่ออาจารย์ผู้ทำการสอนวิชานั้น

Textbook ทำหน้าที่เก็บ ชื่อหนังสือที่ใช้ในวิชานั้น

ตาราง Enrolment เป็นตารางที่เก็บผลการเรียนของนักศึกษา ว่าได้เกรดในวิชาที่ลงเรียนเท่าใด

StID ทำหน้าที่เก็บ รหัสนักศึกษาของแต่ละคน

CrsID ทำหน้าที่เก็บรหัสวิชาที่ทำการเปิดสอน

Grade ทำหน้าที่เก็บ ผลการเรียนของนักศึกษา

ตารางที่ 1.1 แสดงข้อมูลของตาราง Student

ID	Name	Major	Gender
001	Ponjet	CPE	Male
002	Patis	CPE	Male
003	Bantueng	EE	Male
004	Pruet	ME	Male
005	Tasinee	CPE	Female

ตารางที่ 1.2 แสดงข้อมูลของตาราง Enrolment

StID	CrsID	Grade
001	101	A
002	101	B
003	101	A
004	101	B
002	203	A
005	101	A

ตารางที่ 1.3 แสดงข้อมูลของตาราง Course

Code	Title	Instructor	Textbook
101	Basic Computer	Arnan	Introduction Computer
102	Programming Language	Sutasinee	C Programming
203	OOP	Narathip	OOP Using Java6

```
SELECT title,major
FROM student,course,enrolment
WHERE enrolment.StID = student.ID
AND  enrolment.CrsID = course.ID
AND enrolment.Grade = 'A'
```

รูปที่ 1.2 แสดงการสอบถามข้อมูล

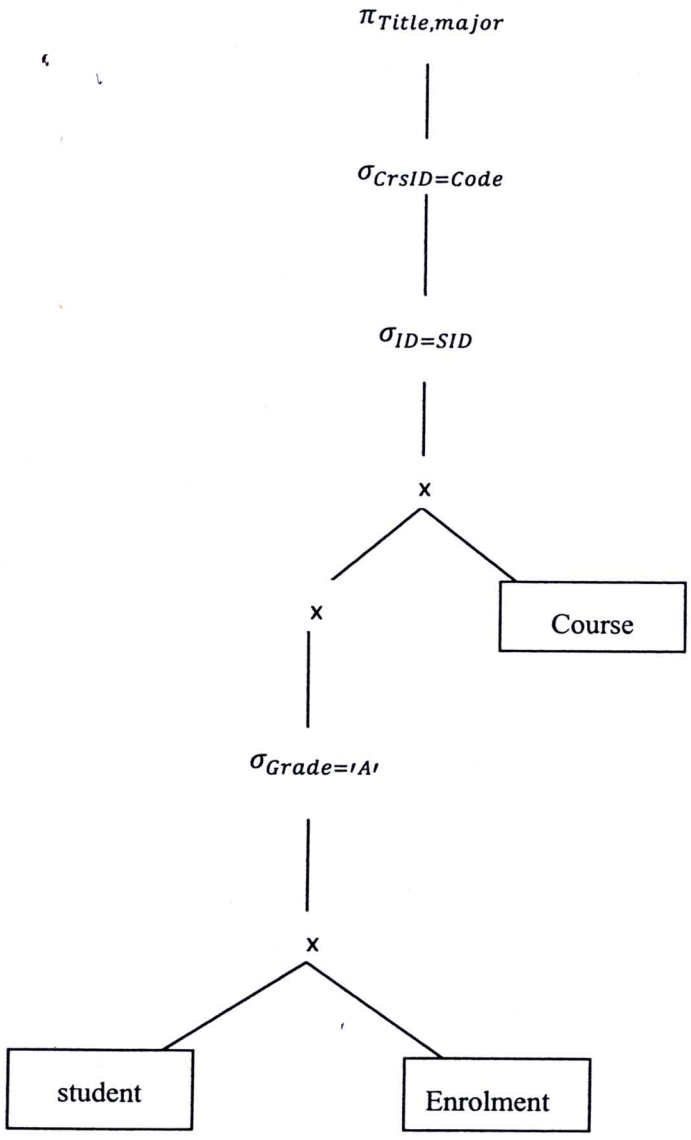
ตารางที่ 1.1 ตารางที่ 1.2 และตารางที่ 1.3 แสดงตัวอย่างข้อมูลจากสกีมาในรูปที่ 1.1 และจากรูปที่ 1.2 แสดงุการสอบถามข้อมูลชุดหนึ่ง โดยสิ่งที่ต้องการในการสอบถามข้อมูลคือ ชื่อวิชาและชื่อสาขาวิชาของนักศึกษาที่ได้เกรด A จากข้อมูลจะสังเกตได้ว่าการที่จะได้ผลลัพธ์ของการสอบถามข้อมูลนี้ จะมีความสัมพันธ์กันทั้ง 3 ตาราง

$$\pi_{Titile,major}(\sigma_{Grade='A'}(student\ x\ Enrolment\ x\ Course)$$

รูปที่ 1.3 แสดงลำดับการทำงานของการสอบถาม

จากรูปที่ 1.2 เมื่อมีการสอบถามข้อมูลออพติไมเซอร์จะทำการวิเคราะห์ว่าจะทำตามลำดับเงื่อนไขใดก่อนหลัง ซึ่งลำดับการสอบถามข้อมูลสามารถทำได้หลายวิธี ในที่นี้สมมุติให้ออพติไมเซอร์ทำการสอบถามข้อมูลแต่ละเงื่อนไขตามลำดับพีชคณิตเชิงสัมพันธ์ในรูปที่ 1.3 ถ้าสมมุติให้มีนักศึกษาทั้งหมดในคณะวิศวกรรมศาสตร์ 10,000 คน และจำนวนที่นักศึกษาลงเรียนทั้งหมดเป็น 100,000 คน และมีวิชาที่ทำการเปิดสอนทั้งหมด 2,000 วิชา จากรูปที่ 1.4 ออพติไมเซอร์จะทำการเปรียบเทียบระหว่าง Student x Enrolment หรือ Enrolment x Course ว่าจะทำการจอย (Join) ตารางใดก่อน โดยทำการประมาณจำนวนแถวจากการพิจารณาจากข้อมูล Student x Enrolment ซึ่งจะเท่ากับ 10,000 x 100,000 = 1,000,000,000 แถว และ Enrolment x Course เท่ากับ 100,000 x 2,000 = 200,000,000 แถว เมื่อออพติไมเซอร์พิจารณาแล้วจึงจะทำการจอยระหว่าง Enrolment x Course ก่อน แต่เมื่อพิจารณาจากรูปที่ 1.4 การประมาณอาจมีการคลาดเคลื่อน เช่น อาจมีนักศึกษาที่ได้เกรด A มีจำนวน 200 คน ซึ่งจะส่งผลให้การจอยกันระหว่าง Student x Enrolment ข้อมูลที่แท้จริงจะเป็น 200 x 100,000 = 20,000,000 เมื่อเป็นเช่นนี้แล้วสิ่งที่ออพติไมเซอร์จะต้องทำก่อนจะต้องเป็นการ

จอยกันระหว่าง Student x Enrolment ไม่ใช่ระหว่าง Enrolment x Course จะเห็นได้ว่าจากที่ยกตัวอย่างมาข้อเสียของการประมาณจำนวนแถวบางครั้งการประมาณจำนวนแถวสามารถเกิดข้อผิดพลาดได้ ซึ่งจากตัวอย่างได้ประมาณผิดพลาดไปถึง 50 เท่า และสาเหตุนี้นี่เองที่ทำให้แผนประมวลผลที่ได้มาไม่ประสิทธิภาพ



รูปที่ 1.4 แสดงการประมาณค่าจำนวนแถว

จากปัญหาที่กล่าวมามีการพัฒนาวิธีการแก้ปัญหาวิธีหนึ่งคือ การคำนวณจำนวนแถวแบบเอ็กแซกท์ (Exact Cardinality) ซึ่งเทคนิคนี้ก็เป็นอีกวิธีหนึ่งในการประเมินจำนวนแถว โดยจะทำการนับจำนวนแถวทีละเงื่อนไข ซึ่งค่าจำนวนแถวที่ได้จะเป็นค่าจำนวนแถวที่เป็นค่าจริงๆ ของเงื่อนไขนั้น ดังนั้นเมื่อเป็นค่าจำนวนแถวที่เป็นค่าจริงแล้ว การทำแผนประมวลที่ได้ย่อมมีประสิทธิภาพมากกว่าการประมาณจำนวนแถว แต่เนื่องจากเทคนิคนี้ใช้เวลานานกว่าการประมาณค่า โดยค่าประสิทธิภาพจะต้องถูกใช้ไปในการทำแต่ละเงื่อนไขซึ่งต้องใช้ทรัพยากรมาก ในกรณีที่ข้อมูลจากฐานข้อมูลน้อยๆ อาจจะไม่มีปัญหาแต่ถ้าเป็นฐานข้อมูลที่เป็นขนาดใหญ่แล้ว อาจจะใช้เวลานานหลายชั่งโมง ซึ่งก็เป็นสิ่งที่ออพติไมเซอร์ก็ไม่ต้องการเช่นเดียวกัน

ในงานวิจัย [1] ผู้เขียนได้นำเสนอแนวคิดที่จะใช้เทคนิคการคำนวณจำนวนแถวแบบเอ็กแซกท์ ที่มีประสิทธิภาพและใช้เวลาน้อยกว่าการประมาณค่า โดยในที่นี้จะอธิบายหลักการของเทคนิคดังกล่าวด้วยตัวอย่างต่อไปนี้

ในฐานข้อมูลหนึ่ง ประกอบไปด้วยตารางต่างๆ แต่มีการสอบถามข้อมูลซึ่งเกี่ยวข้องกับ 3 ตาราง

```
Lineitem(linekey,orderkey,shipdate,receipdate,discount)
Orders(orderkey,custkey,orderpriority)
Customer(custkey,mktsegment)
```

รูปที่ 1.5 แสดงสกีมา (Schema) ของการสั่งสินค้า

จากรูปที่ 1.5 แสดงสกีมาของการสั่งสินค้า โดยมีรายละเอียดของแต่ละตารางดังนี้

ตาราง Lineitem เป็นตารางที่จะเก็บรายละเอียดในการสั่งสินค้าสินค้า ซึ่งประกอบไปด้วยแอตทริบิวต์

linekey ทำหน้าที่เก็บรหัสของรายละเอียดสินค้า

orderkey ทำหน้าที่เก็บรหัสของใบสั่งสินค้า

shipdate ทำหน้าที่เก็บวันที่ที่สินค้าถูกส่ง

receiptdate ทำหน้าที่เก็บวันที่จะได้รับของของลูกค้า

discount ทำหน้าที่เก็บส่วนลดที่ให้แก่ลูกค้า

ตาราง Orders จะเก็บการสั่งสินค้าของลูกค้าแต่ละคน ซึ่งประกอบไปด้วยแอตทริบิวต์

orderkey ทำหน้าที่เก็บรหัสของใบสั่งสินค้า

custkey ทำหน้าที่เก็บรหัสของลูกค้า

orderpriority ทำหน้าที่เก็บ ความสำคัญของสินค้าชนิดนั้น

ตาราง Customer จะเก็บข้อมูลของลูกค้า ซึ่งประกอบไปด้วยแอตทริบิวต์

custkey ทำหน้าที่เก็บรหัสของการรายละเอียดสินค้า

mktsegment ทำหน้าที่เก็บประเภทของสินค้าชนิดนั้น

พิจารณาการสอบถามข้อมูลดังรูปที่ 1.6

```
SELECT……. FROM  Lineitem,Orders,Customer
WHERE l_orderkey = o_orderkey
AND   o_custkey = c_custkey
AND   l_shipdate > '2008-01-01'
AND   l_receiptdate < '2008-02-01'
AND   l_discount < 0.05
AND   o_orderpriority = 'HIGH'
AND   c_mktsegment = 'AUTOMOBILE'
```

รูปที่ 1.6 แสดงการสอบถามข้อมูล

รูปที่ 1.6 แสดงการสอบถามข้อมูลชุดหนึ่ง สิ่งที่ผู้สอบถามต้องการเกี่ยวข้องกับสินค้าประเภทรถยนต์ (Automobile) โดยมีลำดับความสำคัญมาก มีส่วนลดน้อยกว่า 5 เปอร์เซ็นต์ และเป็นสินค้าที่

มีการส่งและรับ ภายในเดือนมกราคมถึงเดือนกุมภาพันธ์ ปี 2008 ซึ่งมีการทำดัชนี (Index) สำหรับแอททริบิวต์ shipdate และ receiptdate ในตาราง lineitem เมื่อมีการสอบถามดังกล่าวแล้วออพติไมเซอร์จะทำการประเมินจำนวนแถวในรูปที่ 1.6 ตามลำดับ ดังแสดงในตารางที่ 1.4

อย่างไรก็ตามถ้าจะทำการประเมินการคำนวณจำนวนแถวแบบเอ็กแซกท์ เพื่อที่จะได้แผนประมวลผลที่มีประสิทธิภาพ การประเมินเงื่อนไขเพื่อที่จะหาค่าจำนวนแถวที่เป็นค่าจริงทีละข้อเป็นการประเมินจำนวนแถวที่สิ้นเปลืองทรัพยากรและใช้เวลานาน ออพติไมเซอร์จึงทำการประเมินจำนวนแถว โดยใช้การประมาณค่าเพื่อที่จะประหยัดเวลาแต่สิ่งที่ตามมาก็คือแผนประมวลผลที่ไม่มีประสิทธิภาพดังที่ได้เคยกล่าวมาแล้ว ดังแสดงในตารางที่ 4 โดย n คือ จำนวนข้อมูลทั้งหมดของตารางนั้นและ s คือ ข้อมูลที่ตรงกับเงื่อนไขที่ทำการสอบถาม

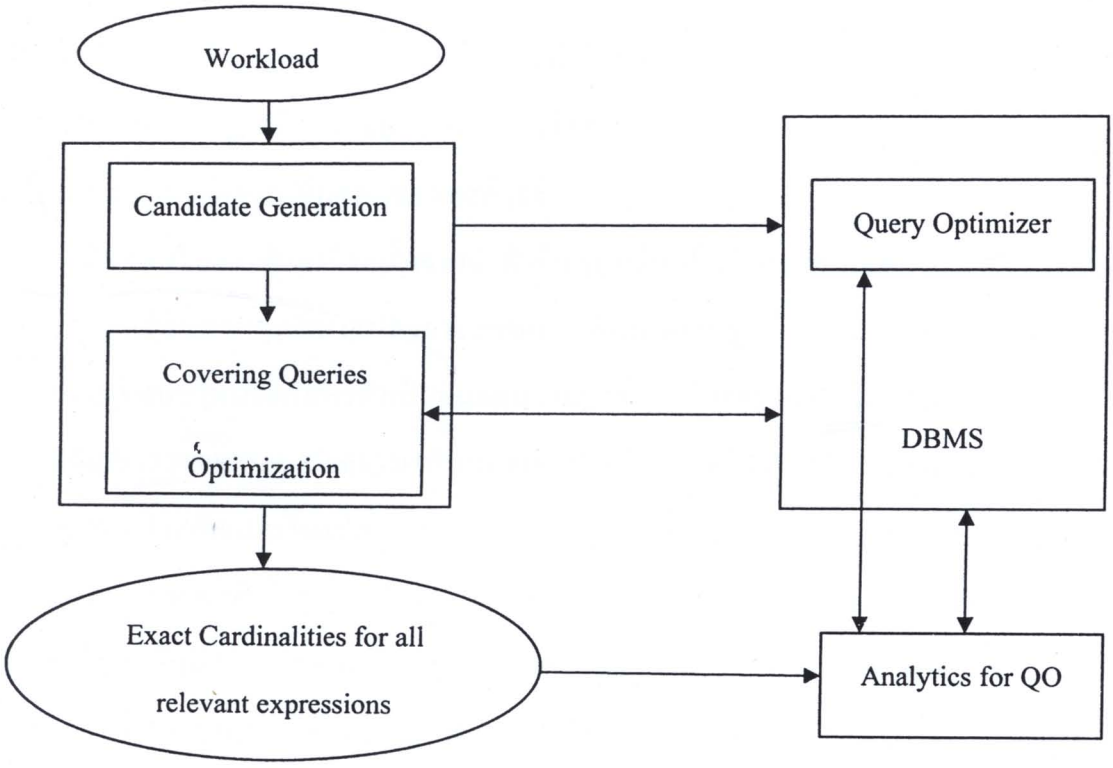
ตารางที่ 1.4 แสดงลำดับการทำงานของออพติไมเซอร์

Expression	Cost
1. (l_shipdate> '2008-01-01')	O(log n +s)
2. (l_shipdate<'2008-02-01'	O(log n +s)
3. (l_shipdate>'2008-01-01' AND l_receipdate<2008-02-01')	O(log n+s)
4. (l_shipdate>'2008-01-01' AND l_receipdate<'2008-02-01' AND l_discount<0.05)	O(log n +s)
5. (o_orderpriority='HIGH')	O(n)
6. (c_mktsegment='AUTOMOBILE')	O(n)
7. (Customer ⋈orders)	$O(n^2)$
8. (Orders ⋈ Lineitem)	$O(n^2)$
9. (Lineitem ⋈ Order ⋈ Customer)	$O(n^2)$

เทคนิคที่สำคัญในงานวิจัย [1] ได้นำเสนอคือ เมื่อมีการประมวลผล (Executed) เงื่อนไขหนึ่ง แล้วจะทำการคำนวณจำนวนแถวอื่นๆ ที่มีความสัมพันธ์กับเงื่อนไขที่กำลังทำการคำนวณอยู่พร้อมกันไปด้วย

โดยเทคนิคทั้งหมดที่ผู้เขียนงานวิจัย [1] นำเสนอรูปแบบโครงสร้างของการทำงานทั้งหมดแสดงในรูปที่ 1.7 โดยผู้เขียนได้ทำการใช้ต้นแบบ (Prototype) ที่มีอยู่ใน Microsoft SQL Server นำมาปรับปรุงแก้ไขโค้ด (Code) เพื่อให้ได้ผลตามที่ต้องการ โดยเทคนิคนี้มีอินพุต (Input) คือ เวิร์คโหลด (Workload), ดังแสดงในรูปที่ 7 ซึ่งเวิร์คโหลดนั้นเป็นเซ็ตของการสอบถามข้อมูลทั้งหมดในขณะใดขณะหนึ่ง โดยจะทำการนำการสอบถามข้อมูลทั้งหมดมาหาความสำคัญ ซึ่งจะทำการผ่านขั้นตอนวิธี (Algorithm) 2 ขั้นตอนวิธี ซึ่งก็คือ แคนดิเดตเจเนเรชัน (Candidate Generation) และ คอเวอร์ริงคิวรีออพติไมเซชัน (Covering Queries Optimization) เมื่อผ่านขั้นตอนวิธีทั้ง 2 แล้ว สิ่งที่ได้จะเป็นเอาต์พุต (Output) คือ จำนวนแถวของทุกเงื่อนไขที่มีความสัมพันธ์กัน เพื่อนำไปใช้ในการออพติไมซ์ต่อไป

จากรูปที่ 1.7 แสดงโครงสร้างของการทำงานของการประเมินจำนวนแถวแบบเอ็กแซกท์ โดยขั้นตอนแรกจะมีอินพุตที่เป็นชุดของการสอบถามข้อมูลคือ เวิร์คโหลด โดยขั้นตอนวิธีจะทำการหาความสัมพันธ์ของเงื่อนไขทุกเงื่อนไขที่มีอยู่ในเวิร์คโหลดว่า มีการใช้ตารางร่วมกันและมีการจอยเงื่อนไขที่คล้ายกัน จากนั้นนำมาจัดกลุ่มโดยเรียกกลุ่มที่เหมือนกันนี้ว่า ซิกเนเจอร์ (Signature) ต่อจากนั้นจะทำการส่งซิกเนเจอร์ ต่อไปยังแคนดิเดตเจเนเรชันดังรูปที่ 1.7 โดยแคนดิเดตเจเนเรชันจะทำการนำซิกเนเจอร์ที่ได้จากเวิร์คโหลดมาทำเป็นเคสคิวรี (Case Query) เพื่อที่จะสามารถทำการนับจำนวนแถวแบบหลายๆ เงื่อนไขได้ ต่อจากนั้นจะเป็นขั้นตอนของคอเวอร์ริงคิวรีออพติไมเซชัน โดยจะนำซิกเนเจอร์แต่ละซิกเนเจอร์มาหาค่าประสิทธิภาพว่าซิกเนเจอร์ใดมีค่าประสิทธิภาพมาก ถ้ามีประสิทธิภาพมากก็จะอยู่ข้างบนเพื่อที่จะนำมาประมวลผลก่อน ซึ่งจะช่วยให้การคำนวณแถวแบบเอ็กแซกท์ใช้เวลาที่น้อยลง เมื่อเสร็จสิ้นจากขั้นตอนนี้แล้วจะนำผลที่ได้ไปหาค่าจำนวนแถวแบบเอ็กแซกท์ตามรูปที่ 1.7 และนำไปประมวลผลเพื่อที่จะได้แผนประมวลผลออกมา



รูปที่ 1.7 แสดงโครงสร้างการหาค่าจำนวนแถวแบบเอ็กแซกท์

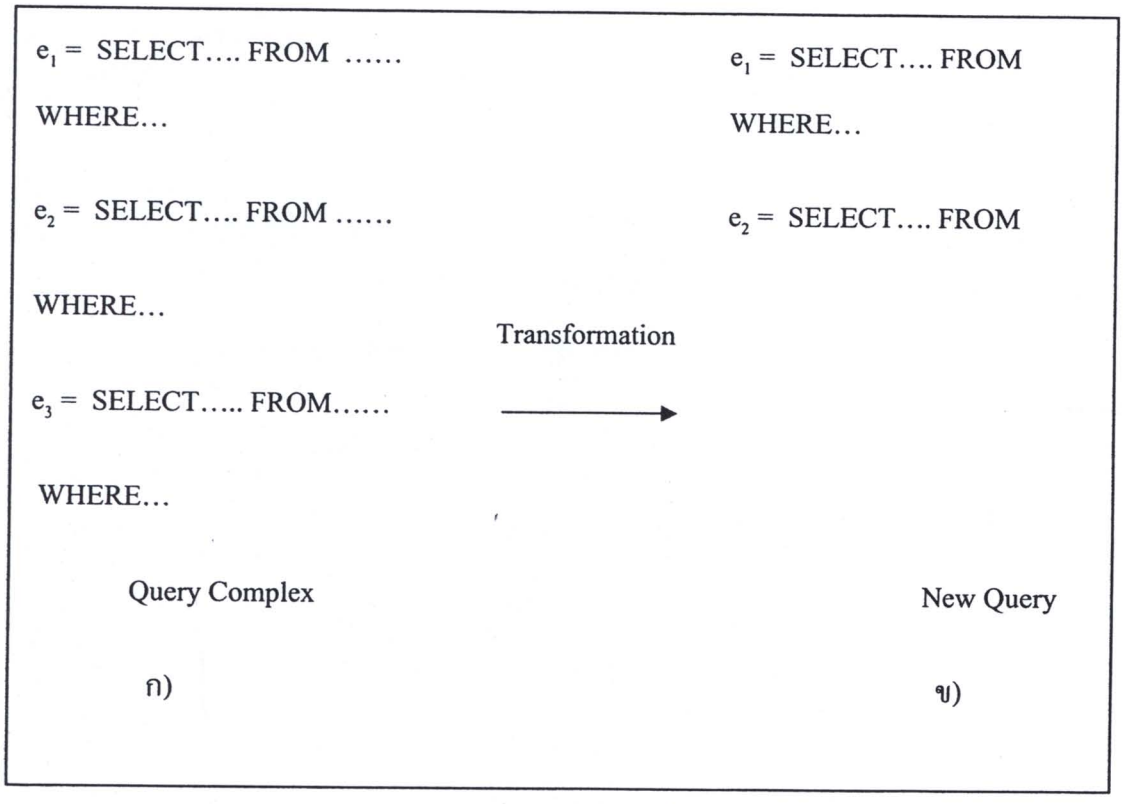
1.2 แนวทางการแก้ปัญหา

จากปัญหาที่กล่าวมาข้างต้นเกี่ยวกับการทำแผนประมวลผลของออพติไมเซอร์ สิ่งที่จะช่วยให้แผนประมวลผลออกมามีประสิทธิภาพจำเป็นต้องใช้จำนวนแถวแบบเอ็กแซกท์ จากวิธีข้างต้นจะช่วยให้การประเมินจำนวนแถวแบบเอ็กแซกท์เป็นไปอย่างรวดเร็วและมีประสิทธิภาพ โดยมีขั้นตอนสำคัญอยู่ที่การหาความสัมพันธ์ของทุกๆ เงื่อนไขที่มีอยู่ในเวิร์คโหลดที่มีความสัมพันธ์เกี่ยวข้องกัน และนำเงื่อนไขเหล่านั้นที่มีความสัมพันธ์กันมาทำเคสดิวรี ซึ่งในส่วนของเคสดิวรีเป็นส่วนหนึ่งที่ช่วยเพิ่มประสิทธิภาพในการทำกระบวนการประเมินจำนวนแถวแบบเอ็กแซกท์เช่นกัน

ในงานวิจัยนี้จะทำการศึกษาการประยุกต์ใช้การแปลงการสอบถาม (Query Transformation) มาใช้ในการประเมินจำนวนแถวแบบเอ็กแซกท์ โดยพิจารณาว่าเมื่อทำการแปลงการสอบถามแล้วได้เคสดิวรีที่แตกต่างออกไปจากที่ไม่ได้มีการแปลงแบบสอบถามแล้ว เวลาและประสิทธิภาพของแผนประมวลผลที่ออกมาแตกต่างไปจากเดิมอย่างไร เพราะใน

งานวิจัย [1] ที่ได้ทำการศึกษามานั้นในตัวอินพุตที่อยู่ในเวิรโหลดไม่ได้มีการแปลงการสอบถามก่อนที่จะมีการทำเคสดิวรีแต่อย่างใด จึงเกิดข้อสงสัยว่าถ้าชุดการสอบถามข้อมูลที่มีการสอบถามข้อมูลที่มีความซับซ้อนมากๆ แล้วเคสดิวรีที่ได้ก็จะซับซ้อนตามไปด้วยทำให้เวลาในการประมวลผลของขั้นตอนของเคสดิวรีช้า

ขั้นตอนในการศึกษาก็จะเริ่มจากนำตัวอินพุตที่มีอยู่ในเวิร์คโหลดโดยนำการสอบถามที่มีความซับซ้อนมากๆ นำมาแปลงการสอบถาม ดังแสดงในรูปที่ 8 เมื่อได้ผลการแปลงการสอบถามแบบใหม่จึงทำการดำเนินขั้นตอนตามรูปที่ 1.7 โดยจะทำการวัดประสิทธิภาพโดยใช้ทีพีซีเอช (TPC-H Benchmark) มาทำการวัดผลประสิทธิภาพโดยจะทำการวัดผลแบบการแปลงแบบสอบถามกับแบบไม่แปลงแบบสอบถาม นอกจากนั้นยังจะทำการวัดประสิทธิภาพของการสอบถามแบบเคสดิวรีและการคำนวณจำนวนแถวแบบเอ็กแซกท์ เมื่อได้ผลลัพธ์สุดท้ายแล้ว จะสรุปได้ว่าเมื่อมีการสอบถามมาในรูปแบบใดรูปแบบหนึ่งที่ได้ทำการทดลองมานั้น สามารถบอกได้ว่าการสอบถามข้อมูลแบบใดสมควรที่จะทำการแปลงการสอบถามหรือไม่สมควรแปลงแบบสอบถาม



รูปที่ 1.8 แสดงการแปลงการสอบถาม

1.3 สรุปสาระสำคัญจากเอกสารที่เกี่ยวข้อง

Surajit และคณะ [1] ได้นำเสนอวิธีการคำนวณจำนวนแถวแบบเอ็กแซกท์ โดยทำการสร้างโปรแกรมเพิ่มเข้าไปในต้นแบบของ Microsoft SQL Server เพื่อที่จะเพิ่มความสามารถให้ออพติไมเซอร์ทำงานได้ดีขึ้น

Ahmed และคณะ [2] ได้นำเสนอรูปแบบการแปลงแบบสอบถาม ซึ่งสามารถทำให้การสอบถามข้อมูลที่มีความซับซ้อนมากๆ ให้อยู่ในรูปแบบที่ง่ายต่อการประมวลผล

Graefe [3] ได้นำเสนอวิธีการขั้นพื้นฐานของออพติไมเซอร์ว่ามีวิธีการทำงานอย่างไร ซึ่งได้บอกเทคนิคที่จะช่วยให้ออพติไมเซอร์สามารถทำงานได้ดีขึ้น

Loannidis และคณะ [4] ได้เสนอสาเหตุของความผิดพลาด (Error) ของข้อมูลที่เกิดจากการจอยกันมากๆ โดยความผิดพลาดจะเพิ่มขึ้นแบบเอ็กโพเนนเชียล (Exponential) กับตารางที่เพิ่มมากขึ้น

Tamal [5] ได้นำเสนอปัญหาเซ็ตคอเวอร์ (Set Cover Problem) โดยเซ็ตคอเวอร์จะเป็นขั้นตอนวิธีที่ใช้ในการคำนวณเซ็ตที่เล็กที่สุด ที่ครอบคลุมอินพุตทั้งหมด

1.4 วัตถุประสงค์ของการวิจัย

เพื่อศึกษาการประยุกต์ใช้การแปลงการสอบถามในการคำนวณจำนวนแถวแบบเอ็กแซกท์

1.5 ประโยชน์ที่ได้รับจากการศึกษา เชิงทฤษฎี และ/หรือ เชิงประยุกต์

สามารถนำแนวทางการแปลงการสอบถามในงานวิจัยนี้ ไปใช้ในการปรับปรุงประสิทธิภาพการสอบถามทีละมากกว่าหนึ่งการสอบถามได้

1.6 ขอบเขตการทำวิจัย

ใช้ฐานข้อมูล TPC-H Benchmark เพื่อศึกษาลักษณะเฉพาะของการแปลงการสอบถามที่สามารถปรับปรุงประสิทธิภาพของการคำนวณจำนวนแถว โดยประสิทธิภาพจะวัดจากเวลาในการแปลงการสอบถาม เวลาในการคำนวณการสอบถามแบบเคสคิวรี และเวลาในการคำนวณจำนวนแถวแบบเอ็กแซกท์และนำการแปลงแบบสอบถามเทคนิคต่างๆ ที่จะนำมาศึกษาได้แก่ เทคนิค

Subquery Unnesting, Group-By and Distinct View Merging, Join Predicate Pushdown, Join factorization

1.7 ระเบียบวิธีวิจัย

1.7.1 ศึกษาทฤษฎีที่เกี่ยวข้องกับการแปลงการสอบถาม

1.7.2 พัฒนาโมดุลสำหรับการแปลงการสอบถาม

1.7.3 พัฒนาโมดุลสำหรับการคำนวณจำนวนแถวแบบเอ็กแซกท์

1.7.4 ใช้ฐานข้อมูล TPC-H Benchmark เพื่อศึกษาลักษณะเฉพาะของการแปลงสอบถามที่สามารถปรับปรุงประสิทธิภาพของการคำนวณจำนวนแถว โดยประสิทธิภาพจะวัดจากเวลาในการแปลงการสอบถาม เวลาในการคำนวณการสอบถามแบบเคสดิวรีและเวลาในการคำนวณจำนวนแถวแบบเอ็กแซกท์

1.7.5 สรุปผลการทดลอง วิเคราะห์ผลจากการทดลอง จัดทำวิทยานิพนธ์ฉบับสมบูรณ์

1.8 เครื่องมือในการพัฒนา

ในงานวิจัยนี้ มีเครื่องมือในการพัฒนา ดังต่อไปนี้

1.8.1 ฮาร์ดแวร์

-1.81 GHz AMD Turion(tm) 64x2 Dual 960 MB of RAM

1.8.2 ซอฟต์แวร์

- Microsoft Window XP
- Sql Server 2008 release 2
- TPC-H Benchmark (Database)