บทที่ 3

การประยุกต์การแปลงการสอบถามสำหรับการคำนวณจำนวนแถวแบบเอ็กแซกท์

3.1 การประยุกต์การแปลงการสอบถามสำหรับการคำนวณจำนวนแถวแบบเอ็กแซกท์

ในงานวิทยานิพนธ์ฉบับนี้ได้ศึกษางานวิจัย [1] เพื่อนำมาประยุกต์ใช้กับการแปลงการสอบถาม ดังนั้นจึงข่อแสดงวิธีการคำนวณจำนวนแถวแบบเอ็กแซกท์ว่ามีวิธีการอย่างไร ก่อนที่จะทำการแสดงวิธีการประยุกต์ต่อไป อย่างที่ได้กล่าวมาแล้วว่าในปัจจุบันได้มีระบบจัดการฐานข้อมูลมากมายให้เลือกใช้ แต่ละระบบจัดการฐานข้อมูลต่างก็มีจุดเด่นและจุดด้อยแตกต่างกันไป แต่โดยทั่วไปแล้วระบบจัดการฐานข้อมูลจะมีการจัดการสอบถามที่คล้ายๆ กัน ขั้นตอนทั่วไปของการจัดการของระบบจัดการฐานข้อมูลคือ เมื่อมีการสอบถามเข้ามา ขั้นตอนแรกจะเป็นการตรวจภาษาที่เข้ามาว่ามีความถูกต้องตรงตามกับหลักไวยากรณ์และตารางที่จะทำการสอบถามมีอยู่ในฐานข้อมูลหรือไม่ ต่อจากนั้นจะทำการแปลงภาษาการสอบถามเป็นพีชคณิตเชิงสัมพันธ์ และจะทำการส่งต่อให้แก่ออพติไมเซอร์ (Optimizer) ซึ่งออพติไมเซอร์จะทำการสร้างแผนประมวลผล (Execution Plan) หลายๆ แผนแล้วออพติไมเซอร์จะทำการเลือกแผนประมวลผลที่ดีที่สุดเพื่อทำการส่งต่อให้ทำตัวสืบค้นข้อมูลจากฐานข้อมูลและได้เป็นผลลัพธ์ที่ต้องการออกมา ในส่วนวิทยานิพนธ์นี้จะทำการศึกษาในขั้นตอนของออพติไมเซอร์ในระบบจัดการฐานข้อมูลการคำนวณจำนวนแถวแบบเอ็กแซกท์ในงานวิจัย [1] มีขั้นตอนหลายขั้นตอนด้วยกัน ซึ่งจะขอยกตัวอย่างดังต่อไปนี้

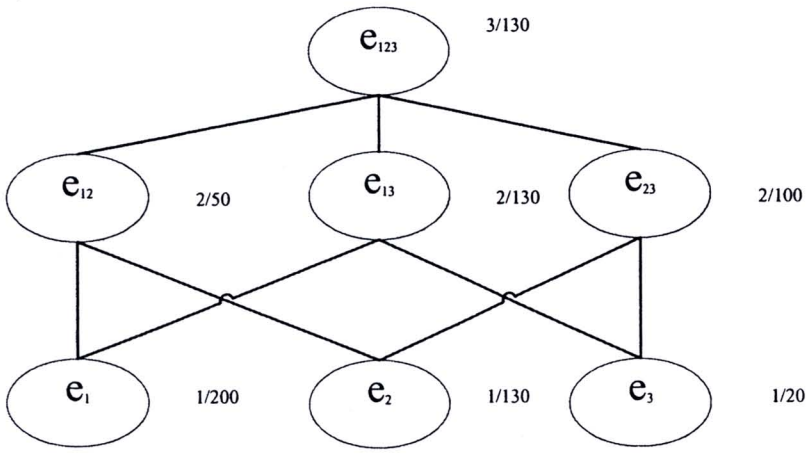
ในขั้นตอนเริ่มต้นของการคำนวณจำนวนแถวแบบเอ็กแซกท์มีอินพุตคือ เวิร์คโหลด ซึ่งในเวิร์คโหลดจะประกอบไปด้วยชุดของการสอบถามข้อมูลหลายๆ การสอบถามอยู่ในเวิร์คโหลด ที่ซึ่งเราเลือกมาในเวลาขณะนั้น ซึ่งมาจากการสอบถามข้อมูลที่มาจากผู้ใช้ ในงานวิจัย [1] ได้ทำการเพิ่มประสิทธิภาพให้กับเวิร์คโหลดซึ่งก็คือ การจัดกลุ่มการสอบถามข้อมูลที่มีอยู่ในเวิร์คโหลดทั้งหมด โดยหลักการในการจัดกลุ่มของการสอบถามที่เพิ่มเข้ามาในเวิร์คโหลดมีดังนี้คือ ในเวิร์คโหลดจะมีขั้นตอนการหาความสัมพันธ์ของเงื่อนไขทุกเงื่อนไขที่มีอยู่ในเวิร์คโหลดว่า มีการใช้ตารางร่วมกันและมีการจอย (Join) เงื่อนไขที่คล้ายกัน จากรูปที่ 3.1 ในเวิร์คโหลดมีเงื่อนไขที่มี

ความสัมพันธ์กันคือเงื่อนไขที่ 1, 2 และ 3 เพราะมีการใช้ตาราง orders เหมือนกันและมีรูปแบบการจอยพรีดิเคต (Join Predicate) ที่คล้ายกันและเงื่อนไขที่ 4 และ5 มีความสัมพันธ์กัน เนื่องจากมีการตาราง lineitem และตาราง orders เหมือนกันและมีรูปแบบการจอยพรีดิเคตที่คล้ายกันเช่นกัน ซึ่งนี่เป็นขั้นตอนแรกที่ทำการเพิ่มประสิทธิภาพให้แก่ออพติไมเซอร์ ขั้นตอนต่อไปเป็นการทำการการจัดกลุ่มการสอบถาม โดยจะนำการสอบถามที่มีความสัมพันธ์กัน ดังที่กล่าวข้างต้นนำอยู่มาในกลุ่มเดียวกัน และจะเรียกกลุ่มการสอบถามที่ความสัมพันธ์กันนี้ว่า ซิกเนเจอร์ (Signature) โดยจากรูปที่ 3.1 จะได้เงื่อนไขที่1, 2 และ 3 เป็นซิกเนเจอร์ {lineitem,orders} และเงื่อนไขที่ 4 และ5 เป็นซิกเนเจอร์ {orders}

เมื่อได้การสอบถามข้อมูลที่ทำการจัดกลุ่มที่เรียกว่า ซิกเนเจอร์ ต่อจากนั้นจะนำแต่ละซิกเนเจอร์นั้นมาทำการเพิ่มประสิทธิภาพให้แก่ออพติไมเซอร์อีกวิธีหนึ่งซึ่งเรียกว่า แคนดิเดตเจเนเรชัน (Candidate Gene ration) โดยจากรูปที่ e_{12} จะหมายถึงเคสดิวรีที่เกิดจากเงื่อนไขที่ 1 และเงื่อนไขที่ 2 โดยการเลือกมีขั้นตอนวิธี (Algorithm) ดังที่เคยกล่าวมาแล้วในบทที่ 2 ในกรณีที่มีเงื่อนไขที่มีซิกเนเจอร์เดียวกันมากๆ จากรูปที่ 3.1 เงื่อนไขที่ 1 เงื่อนไขที่ 2 และเงื่อนไขที่ 3 มีซิกเนเจอร์เดียวกันคือ {lineitem,orders} นำมาทำการแคนดิเดตเจเนเรชัน ได้ดังรูปที่ 3.2 ในกรณีที่มีเงื่อนไข 3 เงื่อนที่จะนำมาพิจารณาในการเลือกแคนดิเดตเจเนเรชันมีกรณีที่เป็นไปได้ทั้งหมดคือ e_{123}, $e_{12}+ e_3$, $e_{13}+ e_2$, $e_{23}+ e_1$ และ $e_1+ e_2+ e_3$ พิจารณาจากรูป 3.2 การเลือกแคนดิเดตเจเนเรชั่นจะทำการเลือกกรณี $e_{12}+ e_3$ ไปทำเคสดิวรีดังต่อไปนี้ คือหลังจากที่ทำการจัดกลุ่มซิกเนเจอร์ แล้วจะนำเงื่อนไข 1,2 และ 3 ที่มีซิกเนเจอร์เดียวกันคือ {lineitem,orders} ไปทำการเลือกว่าจะนำเงื่อนไขใดไปทำเคสดิวรี (Case Query) ซึ่งจะกล่าวถึงในขั้นตอนต่อไปสำหรับการทำเคสดิวรี นำกลุ่มการสอบถามข้อมูลที่มีซิกเนเจอร์เดียวกันไปคำนวณค่าประสิทธิภาพ ดังรูปที่ 3.2 จากรูปมีเงื่อนไขที่ 1 เงื่อนไขที่ 2 และเงื่อนไขที่ 3 ที่จะนำมาทำเคสดิวรี ซึ่งประกอบไปด้วยตัวเลข 2 ชุดในแต่ละเงื่อนไข โดยตัวเลขชุดแรกเกิดจากจำนวนเงื่อนไขที่จะนำมาทำเคสดิวรี และตัวเลขชุดที่ 2 คือค่าประสิทธิภาพของเงื่อนไขนั้น โดยคำนวณจากการประมาณค่าโดยออพติไมเซอร์

```
q1:   SELECT l1.l_discount, o1.o_custkey
FROM lineitem l1, orders o1
WHERE l1.l_orderkey=o1.o_orderkey
      AND o_orderdate > '1990-01-01'
      AND l1.l_extendedprice > (SELECT AVG(l_extendedprice)
            FROM lineitem l2
            WHERE l2.l_suppkey=l1.l_suppkey)
      AND l1.l_suppkey IN (SELECT ps_suppkey
                  FROM partsupp ps, part p
                  WHERE ps.ps_partkey=p.p_partkey
                  AND ps.ps_supplycost > 300)
q2:   SELECT l1.l_discount, o1.o_custkey
FROM lineitem l1, orders o1
WHERE l1.l_orderkey=o1.o_orderkey
      AND o_orderdate > '1990-01-01'
      AND l1.l_extendedprice > (SELECT AVG(l_extendedprice)
            FROM lineitem l2
            WHERE l2.l_suppkey=l1.l_suppkey)
      AND l1.l_suppkey IN (SELECT ps_suppkey
                  FROM partsupp ps, part p
                  WHERE ps.ps_partkey=p.p_partkey
                  AND ps.ps_supplycost > 500)
q3:   SELECT l1.l_discount, o1.o_custkey
FROM lineitem l1, orders o1
WHERE l1.l_orderkey=o1.o_orderkey
      AND o_orderdate > '1990-01-01'
      AND l1.l_extendedprice > (SELECT AVG(l_extendedprice)
            FROM lineitem l2
            WHERE l2.l_suppkey=l1.l_suppkey)
      AND l1.l_suppkey IN (SELECT ps_suppkey
                  FROM partsupp ps, part p
                  WHERE ps.ps_partkey=p.p_partkey
                  AND ps.ps_supplycost > 1200)
q4:   SELECT o_custkey
FROM orders
WHERE o_totalprice > 3000
      AND o_orderdate < '1995-01-01'
q5:   SELECT o_custkey
FROM orders
WHERE o_totalprice > 2000
      AND o_orderdate < '1997-01-01'
```

รูปที่ 3.1 แสดงเวิร์คโหลด



รูปที่ 3.2 แสดงการเลือกแคนดิเดตเจเนเรชัน

เมื่อทำขั้นตอนแคนดิเดตเจเนเรชั่นเรียบร้อยแล้ว ขั้นตอนต่อไปจะเป็นการนำเงื่อนไขที่ได้ผ่านการเลือกจากขั้นตอนแคนดิเดตเจเนเรชั่น ที่มีประสิทธิภาพที่ดีที่สุดนำมาทำเคสคิวรี ซึ่งเหตุผลที่มีการทำเคสคิวรีนั้นเนื่องจากเมื่อเรานำกลุ่มการสอบถามหลายๆ การสอบถามที่มีซิกเนเจอร์เดียวกัน นำมารวมกลุ่มกันไว้เป็นเงื่อนไขเดียวกัน จะทำการประมวลผลของระบบฐานข้อมูลมองการสอบถามนั้นเป็นหนึ่งการสอบถามทำให้ระบบจัดการฐานข้อมูลทำการประมวลผลการสอบถามที่ได้ทำเคสคิวรีเพียงแค่ครั้งเดียวเท่านั้น ทำให้การทำงานของระบบจัดการฐานข้อมูลใช้เวลาและประสิทธิภาพที่ดีกว่าการประมวลผลการสอบถามหลายๆ การสอบถาม จากตัวอย่างข้างต้น กรณีที่แคนดิเดตเจเนเรชั่นทำการเลือกคือ e_{12}+ e_3 พิจารณารูปที่ 3.3 แสดงการคำนวณจำนวนแถวของซิกเนเจอร์ {lineitem,orders} โดยมีการทำเคสคิวรีที่เงื่อนไข 1 และเงื่อนไขที่ 2 แต่เงื่อนไขที่ 3 ไม่นำทำเคสคิวรี

```
e12:  SELECT SUM(a) as card1, SUM(b) as card2
FROM (
      SELECT
      a = CASE when l1.l_extendedprice > (SELECT
      AVG(l_extendedprice)
      FROM lineitem l2
      WHERE l2.l_suppkey=l1.l_suppkey)
            AND l1.l_suppkey IN (SELECT ps_suppkey
                  FROM partsupp ps, part p
                  WHERE ps.ps_partkey=p.p_partkey
                  AND ps.ps_supplycost > 300) then 1 else
                  0 end,
      b = CASE when l1.l_extendedprice > (SELECT
      AVG(l_extendedprice)
      FROM lineitem l2
      WHERE l2.l_suppkey=l1.l_suppkey)
            AND l1.l_suppkey IN (SELECT ps_suppkey
                  FROM partsupp ps, part p
                  WHERE ps.ps_partkey=p.p_partkey
                        AND ps.ps_supplycost > 500) then 1
                        else 0 end
      FROM lineitem l1, orders o1
      WHERE l1.l_orderkey=o1.o_orderkey
            AND o_orderdate > '1990-01-01'
            AND l1.l_extendedprice > (SELECT
            AVG(l_extendedprice)
                  FROM lineitem l2
                  WHERE l2.l_suppkey=l1.l_suppkey)
                  AND l1.l_suppkey IN (SELECT ps_suppkey
                  FROM partsupp ps, part p
                  WHERE ps.ps_partkey=p.p_partkey
                        AND ps.ps_supplycost > 300)
e3:   SELECT SUM(a) as card1
FROM (
      SELECT
      a = CASE when l1.l_extendedprice > (SELECT
      AVG(l_extendedprice)
      FROM lineitem l2
      WHERE l2.l_suppkey=l1.l_suppkey)
            AND l1.l_suppkey IN (SELECT ps_suppkey
                  FROM partsupp ps, part p
                  WHERE ps.ps_partkey=p.p_partkey
                  AND ps.ps_supplycost > 1200) then 1 else
                  0 end,
                  FROM lineitem l1, orders o1
      WHERE l1.l_orderkey=o1.o_orderkey
            AND o_orderdate > '1990-01-01'
            AND l1.l_extendedprice > (SELECT
            AVG(l_extendedprice)
                  FROM lineitem l2
                  WHERE l2.l_suppkey=l1.l_suppkey)
                  AND l1.l_suppkey IN (SELECT ps_suppkey
                  FROM partsupp ps, part p
                  WHERE ps.ps_partkey=p.p_partkey
                        AND ps.ps_supplycost > 1200)
```

รูปที่ 3.3 แสดงเคสคิวรี

นอกเหนือจากการทำเคสคิวรีที่ช่วยเพิ่มประสิทธิภาพในการคำนวณจำนวนแถวแบบเอ็กแซกท์แล้ว ในงานวิจัย [1] ยังทำการใช้ขั้นตอนวิธีที่มีชื่อว่า คอเวอร์ริงคิวริงออพติไมเซชัน (Covering Queries Optimization) ดังที่เคยกล่าวมาแล้วในบทที่ 2 เช่นกัน

3.2 การแปลงการสอบถามสำหรับการคำนวณจำนวนแถวแบบเอ็กแซกท์

จากตัวอย่างที่ได้แสดงการคำนวณจำนวนแถวแบบเอ็กแซกท์ข้างต้น ในวิทยานิพนธ์ฉบับนี้จึงต้องการศึกษาถึงประสิทธิภาพ เมื่อมีการนำการแปลงการสอบถามเข้ามาใช้ในการคำนวณจำนวนแถวแบบเอ็กแซกท์ โดยรูปที่ 3.4 แสดงการแปลงการสอบถามในเวิร์คโหลดจากรูปที่ 3.1 เมื่อมีการแปลงการสอบถามแล้ว ค่าประสิทธิภาพของการสอบถามย่อมต้องเปลี่ยนแปลงไปด้วย ทำให้ขั้นตอนของงานวิจัยข้างต้นได้เปลี่ยนแปลงไปด้วย เพราะงานวิจัย [] ได้ใช้ค่าประสิทธิภาพเป็นตัวแปรสำคัญในการพิจารณา โดยเมื่อแปลงการสอบถามอาจจะทำให้ขั้นตอนของแคนดิเดตเจเนเรชันเปลี่ยนแปลงไป ก็จะทำให้การเลือกทำเคสคิวรีเปลี่ยนแปลงตามไปด้วย

โดยในปัจจุบันเทคนิคการแปลงการสอบถามนั้นมีหลายรูปแบบ แต่ในวิทยานิพนธ์ฉบับนี้จะเลือกใช้เทคนิคการแปลงการสอบถามแบบคอสต์เบส (Cost-based Transformation) ซึ่งการแปลงการสอบถามรูปแบบนี้จะพิจารณาจากค่าประสิทธิภาพว่าแบบแปลงการสอบถามหรือแบบไม่แปลงการสอบถามแบบไหนที่ดีกว่ากัน ซึ่งในเทคนิคการแปลงการสอบถามแบบคอสต์เบสนี้ ได้มีเทคนิคย่อยๆ อีกหลายเทคนิค โดยในวิทยานิพนธ์ฉบับนี้ได้ใช้เทคนิคดังต่อไปนี้คือ Subquery Unnesting, Group-by Distinct view Merging, Join Predicate Pushdown และ Join Factorization

```
q1:   SELECT l1.l_discount, o1.o_custkey
      FROM lineitem l1, orders o1, (SELECT
AVG(l_extendedprice)
                        avg exprice, l_suppkey
                        FROM lineitem l2
                        GROUP BY l_suppkey)
      WHERE l1.l_orderkey=o1.o_orderkey
      AND o_orderdate > '1990-01-01'
      AND l1.l_suppkey=v.l_suppkey
      AND l1.l_extendedprice > v.avg_exprice
      AND l1.l_suppkey IN (   SELECT ps_suppkey
                                    FROM partsupp ps, part
                        p
                                    WHERE
                        ps.ps_partkey=p.p_partkey
                        AND ps.ps_supplycost > 300)
q2:   SELECT l1.l_discount, o1.o_custkey
FROM lineitem l1, orders o1, (SELECT AVG(l_extendedprice)
                        avg exprice, l_suppkey
                        FROM lineitem l2
                        GROUP BY l_suppkey)
WHERE l1.l_orderkey=o1.o_orderkey
      AND o_orderdate > '1990-01-01'
      AND l1.l_suppkey=v.l_suppkey
      AND l1.l_extendedprice > v.avg_exprice
      AND l1.l_suppkey IN (   SELECT ps_suppkey
                        FROM partsupp ps, part p
                        WHERE ps.ps_partkey=p.p_partkey
                              AND ps.ps_supplycost > 500)
q3:   SELECT l1.l_discount, o1.o_custkey
FROM lineitem l1, orders o1, (SELECT AVG(l_extendedprice)
                        Avg_exprice, l_suppkey
                        FROM lineitem l2
                        GROUP BY l_suppkey)
WHERE l1.l_orderkey=o1.o_orderkey
      AND o_orderdate > '1990-01-01'
      AND l1.l_suppkey=v.l_suppkey
      AND l1.l_extendedprice > v.avg_exprice
      AND l1.l_suppkey IN (   SELECT ps_suppkey
                        FROM partsupp ps, part p
                        WHERE ps.ps_partkey=p.p_partkey
                        AND ps.ps_supplycost > 1200)

q4:   SELECT o_custkey
FROM orders
WHERE o_totalprice > 3000
      AND o_orderdate < '1995-01-01'
q5:   SELECT o_custkey
FROM orders
WHERE o_totalprice > 2000
      AND o_orderdate < '1997-01-01'
```

รูปที่ 3.4 แสดงการแปลงการสอบถามจากเวิร์คโหลด

3.2.1 เทคนิค Subquery Unnesting

เทคนิค Subquery Unnesting นี้ เป็นเทคนิคที่ใช้เมื่อมีการสอบถามย่อยซ้อนกันหลายๆ ชั้น ดังแสดงในรูปที่ 3.5 การสอบถามดังกล่าวต้องการที่จะทราบถึงส่วนลดของลูกค้าและรหัสของลูกค้าโดยที่มีการสั่งสินค้าตั้งแต่วันที่ 1 มกราคม ค.ศ. 1990 เป็นต้นไปและมีส่วนเพิ่มของราคาสินค้าต้องมากกว่าค่าเฉลี่ยของส่วนเพิ่มของราคาสินค้าโดยที่มีการค่าใช้จ่ายแก่ผู้ผลิตมากกว่า 300 ขึ้นไป จากรูปที่ 3.5 จะเห็นได้ว่ามีการสอบถามย่อยซ้อนอยู่ในการสอบถามหลักหลายชั้นและจากพรีดิเคตของการสอบถามดังรูปได้มีการสอบถามย่อยที่มีความสัมพันธ์กับการสอบถามย่อยของการสอบถามย่อยอื่นด้วย คือ l1.l_suppkey ได้มีความสัมพันธ์ในการหาค่าเฉลี่ยของ l_extendedprice ในการสอบถามย่อย การสอบถามรูปแบบนี้จะทำให้การพิจารณาของออพติไมเซอร์ซับซ้อนมาก เพราะเป็นเรื่องยากที่จะทำการเมิร์จ (Merge) กันระหว่างการสอบถามย่อยสองการสอบถามที่มีความสัมพันธ์กัน การแปลงการสอบถามในรูปแบบนี้จะทำการนำการสอบถามย่อยที่มีการคำนวณแบบจัดกลุ่ม (Aggregate) นำไปไว้ในวิว (view) แล้วทำการจัดกลุ่ม (Group-by) ซึ่งจะทำให้การสอบถามจะทำการคำนวณค่าเฉลี่ยเพียงแค่ครั้งเดียวเท่านั้น ก่อนที่จะนำไปเมิร์จกันเพื่อเปรียบเทียบค่าในพรีดิเคต โดยการแปลงการสอบถามเทคนิค Subquery Unnesting ได้แสดงในรูปที่ 3.6

```
SELECT l1.l_discount, o1.o_custkey
FROM lineitem l1, orders o1
WHERE l1.l_orderkey=o1.o_orderkey
    AND o_orderdate > '1990-01-01'
            AND l1.l_extendedprice> (SELECT
                AVG(l_extendedprice)
                      FROM lineitem l2
                      WHERE
                   l2.l_suppkey=l1.l_suppkey)
    AND l1.l_suppkey IN (SELECT ps_suppkey
                 FROM partsupp ps, part p
                 WHERE ps.ps_partkey=p.p_partkey
                   AND ps.ps_supplycost > 300)
```

รูปที่ 3.5 แสดงการสอบถามแบบ Unnesting

```
SELECT l1.l_discount, o1.o_custkey
      FROM lineitem l1, orders o1,
                  (SELECT AVG(l_extendedprice)
                                        Avg_exprice,
                                        l_suppkey
                        FROM lineitem l2
                              GROUP BY l_suppkey)
                WHERE l1.l_orderkey=o1.o_orderkey
                AND o_orderdate > '1990-01-01'
                AND l1.l_suppkey=v.l_suppkey
                AND l1.l_extendedprice > v.avg_exprice
                AND l1.l_suppkey IN (SELECT ps_suppkey
                            FROM partsupp ps, part p
                                  WHERE
                                  ps.ps_partkey=p.p_partkey
                                  AND ps.ps_supplycost > 300)
```

รูปที่ 3.6 แสดงการแปลงการสอบถามแบบ Unnesting

3.2.2 เทคนิค Group-by and Distinct View Merging

เทคนิคการแปลงการสอบถามรูปแบบนี้จะเป็นการสอบถามที่มีการสอบถามย่อยที่มี Group-by หรือ Distinct อยู่ในวิว โดยการแปลงการสอบถามจะทำการนำการสอบถามย่อยนั้นไปเมิร์จกันนอกการสอบถามย่อย ซึ่งการแปลงการสอบถามในรูปแบบนี้ทำให้ออพติไมเซอร์นั้นมีทางเลือกในการที่จะทำการจอยตารางเข้าด้วยกัน ดังที่ได้แสดงในรูปที่ 3.6 เป็นการสอบถามในรูปแบบ Group-by and Distinct View Merging โดยมี Group-by อยู่ในวิว ซึ่งการสอบถามเป็นการสอบถามที่ต้องการผลลัพธ์เหมือนกับเทคนิค Subquery Unnesting พิจารณารูปที่ 3.7 เป็นการแปลงการสอบถามของรูปแบบ Group-by and Distinct View Merging เมื่อทำการแปลงการสอบถามจะทำให้การคำนวณค่าเฉลี่ยนั้นเป็นขั้นตอนหลังจากที่ออพติไมเซอร์ได้ทำการจอยตารางที่ต้องใช้ในการพิจารณา

```
SELECT l1.l_discount, o1.o_custkey
       FROM lineitem l1, orders o1, lineitem l2
       WHERE l1.l_orderkey=o1.o_orderkey
            AND o_orderdate > '1990-01-01'
            AND l2.l_suppkey=l1.l_suppkey
      AND l1.l_suppkey IN (SELECT ps_suppkey
                           FROM partsupp ps, part p
                           WHERE ps.ps_partkey=p.p_partkey
                                AND ps.ps_supplycost > 300)
            GROUP BY l2.l_suppkey, l1.l_suppkey,
            o1.o_custkey, l1.l_discount,
            l1.l_extendedprice
      HAVING l1.l_extendedprice > AVG(l2.l_extendedprice)
```

รูปที่ 3.7 แสดงการแปลงการสอบถามรูปแบบ Group-by and Distinct View Merging

3.2.3 เทคนิค Join Predicate Pushdown

เทคนิค Join Predicate Pushdown นี้เป็นเทคนิคที่ทำการแปลงการสอบถามในรูปแบบที่มีการสอบถามย่อยที่มีพรีดิเคตอยู่ข้างในวิวและมีความสัมพันธ์กับพรีดิเคตที่เป็นการสอบถามหลัก โดยในรูปที่ 3.8 แสดงการสอบถามในรูปแบบที่กล่าวถึงข้างต้น ซึ่งการสอบถามดังกล่าวต้องการที่ทราบถึง ชื่อของลูกค้า, วันที่ลูกค้าทำการสั่งสินค้าและวันที่จะทำการส่งสินค้า โดยต้องเป็นสินค้าที่มีผู้ผลิตอยู่ในประเทศอังกฤษ ซึ่งโดยทั่วไปแล้วการสอบถามรูปแบบนี้อาจจะทำให้ออพติไมเซอร์ทำงานซ้ำซ้อนขึ้นได้ การแปลงการสอบถามของรูปแบบ Join Predicate Pushdown ได้แสดงในรูปที่ 3.9 โดยวิธีแปลงการสอบถามคือจะนำพรีดิเคตจากการสอบถามย่อยที่มีความสัมพันธ์กับพรีดิเคตของการสอบถามหลักมาไว้ในวิว โดยวิธีนี้จะทำให้ออพติไมเซอร์ตัดการทำงานของ Distinct ออกไปและทำให้รูปแบบการจอยของออพติไมเซอร์เปลี่ยนแปลงตามไปด้วย

```
SELECT c.c_name, o.o_orderdate, l.l_shipdate
FROM   customer   c,   lineitem   l,   partsupp   ps,orders
o,(SELECT distinct s.s_suppkey,
      FROM nation n, supplier s
      WHERE n.n_nationkey=s.s_nationkey
            AND n.n_nationkey = 'united kingdom') v
      WHERE v.s_suppkey=ps_suppkey
                        AND c.c_custkey=o.o_custkey
                        AND l.l_orderkey=o.o_orderkey
                        AND l.l_shipdate > '1995-01-01'
```

รูปที่ 3.8 แสดงการสอบถามแบบ Join Predicate Pushdown

```
SELECT c.c_name, o.o_orderdate, l.l_shipdate
                  FROM customer c, lineitem l, partsupp
                  ps, orders o, (SELECT distinct
                  s.s_suppkey,
                        FROM nation n, supplier s,
                        partsupp ps
                        WHERE n.n_nationkey=s.s_nationkey
                          AND ps.ps_suppkey=s.s_suppkey
                                  AND n.n_nationkey =
                                 'united kingdom') v
 WHERE c.c_custkey=o.o_custkey
           AND l.l_orderkey=o.o_orderkey
     AND l.l_shipdate > '1995-01-01'
```

รูปที่ 3.9 แสดงการแปลงการสอบถามแบบ Join Predicate Pushdown

3.2.4 เทคนิค Join Factorization

เทคนิค Join Factorization เป็นเทคนิคที่มีการสอบถามเกี่ยวกับ Union และ Union All โดยการสอบถามที่จะทำการแปลงเป็นการสอบถาม 2 การสอบถามที่มีการเรียกใช้ตารางที่เกือบจะเรียกใช้ตารางเหมือนกันทุกตาราง โดยการแปลงการสอบถามจะทำการนำการสอบถามทั้ง 2 การสอบถามที่มีการเรียกใช้ตารางเหมือนกันนำมารวมกันเป็นหนึ่งการสอบถาม และนำการสอบถามที่มีการเรียกใช้ตารางเหมือนนำไปเป็นการสอบถามย่อยและทำการยูเนียนกันในวิวแทน ซึ่งการแปลงการสอบถามรูปแบบนี้ช่วยให้ลดการทำงานของออพติไมเซอร์ลงได้ จากรูปที่ 3.10 แสดงการสอบถามที่ไม่ได้แปลงการสอบถาม ซึ่งการสอบถามดังกล่าวต้องการที่ทราบถึง วันที่ขนส่งสินค้า

และรหัสของผู้ผลิตสินค้า โดยที่ต้องมีส่วนเพิ่มของราคาสินค้ามากกว่า 5000 ขึ้นไป และมีค่าใช้จ่ายผู้ผลิตมากกว่า 600 ขึ้นไป โดยออพติไมเซอร์จะทำการประมวลผลการสอบถามที่หนึ่งก่อนและค่อยทำการประมวลผลการสอบถามที่สองหลังจากนั้นจึงนำการสอบถามทั้งสองมายูเนียนกัน แต่จากรูปมีการเรียกใช้ตารางที่คล้ายกันจึงทำให้ออพติไมเซอร์ทำการเรียกใช้ตารางที่ซ้ำซ้อน การแปลงการสอบถามรูปแบบนี้ช่วยให้การเรียกใช้ตารางของออพติไมเซอร์เรียกใช้ตารางเพียงแค่ครั้งเดียว ซึ่งจะช่วยให้ประสิทธิภาพของออพติไมเซอร์ดีขึ้น โดยการแปลงการสอบถามรูปแบบนี้ได้แสดงในรูปที่ 3.11

```
SELECT l.l_shipdate, s.s_suppkey
 FROM supplier s, lineitem l, customer c
       WHERE l.l_suppkey=s.s_suupkey
   AND s.s_nationkey=c.c_nationkey
         AND l.l_extendedprice > 5000

UNION ALL

SELECT l.l_shipdate, ps.s_suppkey
        FROM partsupp ps, supplier s, lineitem l,
                    customer
      WHERE l.l_partkey=ps.ps_partkey
 AND ps.ps_suppkey=s.s_suppkey
    AND s.s_nationkey=c.c_nationkey
       AND ps.ps_supplycost > 600
```

รูปที่ 3.10 แสดงการสอบถามแบบ Join Factorization

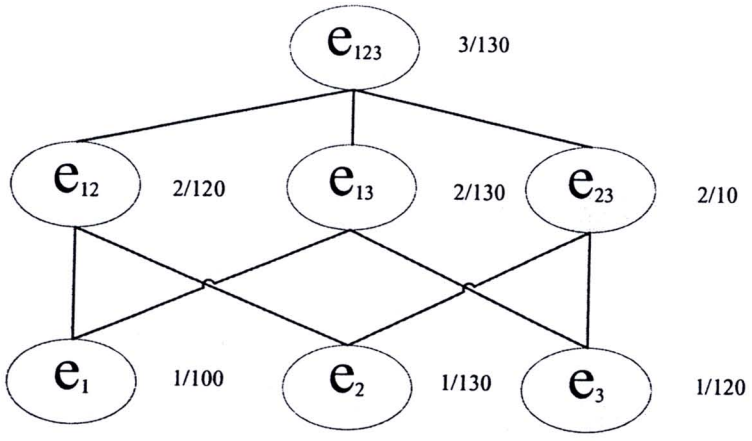
```
SELECT v.l_shipdate, s.s_suppkey
FROM supplier s, customer c,
            (SELECT l.l_shipdate, l.l_suppkey
             FROM lineitem l
             WHERE l.l_extendedprice > 5000

             UNION ALL

             SELECT l.l_shipdate, ps.ps_suppkey
             FROM partsupp ps, lineitem l
             WHERE l.l_partkey=ps.ps_partkey
               AND ps.ps_supplycost > 600)
WHERE s.s_suppkey=l.l_suppkey
  AND s.s_nationkey=c.c_nationkey
```

รูปที่ 3.11 แสดงการแปลงการสอบถามแบบ Join Factorization

เมื่อทำการแปลงการสอบถามแล้วค่าประสิทธิภาพของการสอบถามย่อมต้องเปลี่ยนแปลงไป อย่างที่เคยกล่าวข้างต้น ซึ่งเมื่อทำการแปลงการสอบถามแล้วอาจจะมีประสิทธิภาพที่ดีขึ้นหรือแย่ลงนั้น เป็นสิ่งที่จะทำการศึกษาในวิทยานิพนธ์ฉบับนี้ เมื่อทำการแปลงการสอบถาม ดังแสดงในรูปที่ 3.6 แล้วขั้นตอนในการเลือกแคนดิเดตเจเนเรชันย่อมต้องเปลี่ยนแปลงไปดังแสดงในรูปที่ 3.12 ค่าประสิทธิภาพที่เปลี่ยนแปลงไปย่อมต้องส่งผลในการเลือกการสอบถามไปทำเคสคิวรี



รูปที่ 3.12 แสดงแคนดิเดตเจเนเรชัน

รูปที่ 3.12 จะทำการเลือกการสอบถามที่จะนำมาทำเคสคิวรีคือ e_{23}+ e_1 เพราะมีค่าประสิทธิภาพที่ดีสุดใน 5 กรณี ทำให้เคสคิวรีของการคำนวณจำนวนแถวแบบเอ็กแซกท์เปลี่ยนแปลงไปเช่นกัน จากการเปลี่ยนแปลงดังกล่าวจะนำไปทดลองเพื่อเปรียบเทียบเวลาของการแปลงการสอบถามกับไม่แปลงการสอบถามว่าเทคนิคแบบใดที่เหมาะสมกว่าในการคำนวณจำนวนแถวแบบเอ็กแซกท์