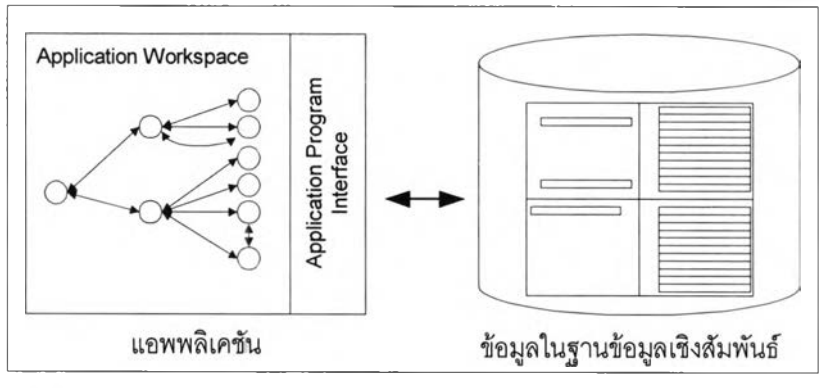
บทที่ 2

แนวคิดและทฤษฎีที่เกี่ยวข้อง

2.1 การพัฒนาแอพพลิเคชันเชิงวัตถุในปัจจุบัน [1]

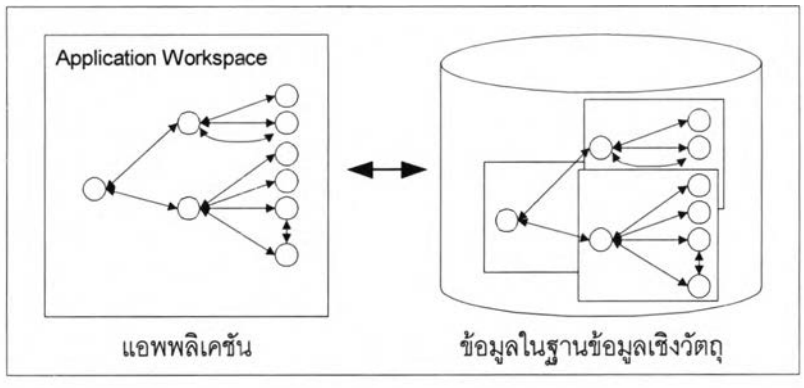
ข้อมูลในฐานข้อมูลเชิงสัมพันธ์อยู่ในรูปของแถวข้อมูลในตาราง แต่ข้อมูลบนพื้นที่การทำงานของแอพพลิเคชันจะต้องอยู่ในรูปของอ็อบเจคท์ ซึ่งเป็นคนละประเภทกัน ฉะนั้นการนำข้อมูลจากฐานข้อมูลเข้าสู่พื้นที่การทำงานของแอพพลิเคชันในรูปของอ็อบเจคท์ หรือการนำอ็อบเจคท์จัดเก็บบนฐานข้อมูล จะต้องมีการแปลงระหว่างข้อมูลในระบบจัดการฐานข้อมูลเชิงสัมพันธ์กับอ็อบเจคท์บนพื้นที่การทำงานของแอพพลิเคชัน



รูปที่ 2.1 อ็อบเจคท์ใน Application Workspace และการจัดเก็บข้อมูลบน RDBMS

รูปที่ 2.1 แสดงการนำข้อมูลในฐานข้อมูลเชิงสัมพันธ์มาใช้ในพื้นที่การทำงานของแอพพลิเคชัน (Application Workspace) ซึ่งจะต้องมีส่วนของการแปลงข้อมูลระหว่างข้อมูลในฐานข้อมูลเชิงสัมพันธ์เป็นอ็อบเจคท์เรียกว่า Application Program Interface

อย่างไรก็ตาม เมื่อระบบการจัดเก็บข้อมูลสามารถจัดเก็บอ็อบเจคท์ในพื้นที่การทำงานของแอพพลิเคชันขึ้นบนฐานข้อมูลได้โดยตรง ขั้นตอนการพัฒนาแอพพลิเคชันจะลดไป 1 ขั้นตอน ดังแสดงในรูปที่ 2.2 เนื่องจากไม่จำเป็นต้องมี Application Program Interface



รูปที่ 2.2 อ็อบเจคท์ใน Application Workspace และการจัดเก็บข้อมูลบน OODBMS

รูปที่ 2.2 แสดงอ็อบเจคท์บนพื้นที่การทำงานของแอพพลิเคชันและอ็อบเจคท์ในฐานข้อมูลเชิงวัตถุ จะเห็นว่าแอพพลิเคชันสามารถนำอ็อบเจคท์ในฐานข้อมูลมาใช้ได้โดยตรงและทำนองเดียวกันการจัดเก็บสามารถจัดเก็บได้โดยตรง

เพื่อให้การพัฒนาแอพพลิเคชันเชิงวัตถุง่ายและสะดวกมากขึ้น ฐานข้อมูลที่ใช้ควรเป็นฐานข้อมูลเชิงวัตถุ จึงมีความจำเป็นที่จะทำการแปลงฐานข้อมูลเชิงสัมพันธ์เป็นฐานข้อมูลเชิงวัตถุ ทฤษฎีและแนวคิดที่เกี่ยวข้องมีดังต่อไปนี้

- แผนแบบเชิงวัตถุ (Object Model)
- แผนแบบเชิงสัมพันธ์ (Relational Model)
- การจัดรูปแบบบรรทัดฐาน (Normalization)
- การแปลงแผนแบบเชิงวัตถุเป็นแผนแบบเชิงสัมพันธ์
- ระบบจัดการฐานข้อมูล (DBMS Concepts)
- ความสัมพันธ์ระหว่างระบบจัดการฐานข้อมูลกับออบเจคท์ (DBMS and Objects)

2.2 **แผนแบบเชิงวัตถุ** (Object Model) [1]

แผนแบบเชิงวัตถุเป็นแนวคิดในการนิยามกระบวนการ (Process) และเอนทิตี(Entity) ต่างๆให้เป็นแผนแบบเชิงวัตถุประกอบด้วยส่วนประกอบดังนี้

หลักนามธรรมข้อมูล (Data Abstraction)

หลักนามธรรมข้อมูลเป็นวิธีการในการกลั่นกรองข้อมูลเพื่อนำส่วนที่สำคัญและจำเป็นมาใช้ หรือกล่าวอีกนัยหนึ่งคือกระบวนการในการทำแผนแบบข้อมูลโดยสรุปตามความจำเป็นในการใช้งานของแอพพลิเคชัน ประกอบด้วยส่วนประกอบย่อยดังต่อไปนี้

อินแสตนท์ (Object Instance)

ในการทำแผนแบบเชิงวัตถุจะได้เอนทิตี (Entity) ขึ้นมาจำนวนหนึ่ง แต่ละเอนทิตีเรียกว่า Object Instance

รหัส (Object Identification – OID)

OID ใช้ในการอ้างอิงอินแสตนท์ โดยที่ OID ไม่ขึ้นกับข้อมูลในอ็อบเจคท์ OID ถูกสร้างขึ้นมาโดย Object System และ OID จะคงอยู่ตลอดอายุของอ็อบเจคท์

ความสัมพันธ์ (Relationship)

ความสัมพันธ์ระหว่างอ็อบเจคท์ใช้ OID ในการกำหนดความสัมพันธ์

คลาส (Classes)

อินแสตนท์ที่มีคุณลักษณะเหมือนกันสามารถนำมาจัดกลุ่มเป็นกลุ่มเดียวกันเรียกว่าคลาส

การห่อหุ้ม (Encapsulation)

การห่อหุ้ม หมายถึงแนวคิดในการรวมวิธีการหรือพฤติกรรมเข้ากับอินแสตนท์ซึ่งกำหนดโดยคลาส หรือพูดอีกนัยหนึ่งการห่อหุ้มคือการห่อรวมโค้ดคำสั่งเข้ากับข้อมูลประกอบเป็นสิ่งที่เรียกว่าอ็อบเจคท์

การสืบทอดคุณสมบัติ (Inheritance)

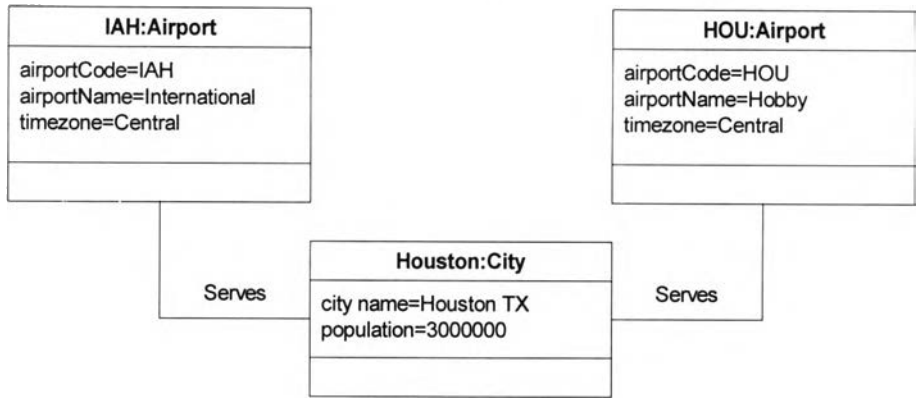
การนิยามคลาสขึ้นมาใหม่โดยที่คลาสที่นิยามใหม่มีคุณลักษณะบางประการเหมือนกับคลาสที่นิยามมาก่อนหน้านี้แล้วทุกประการ คลาสที่นิยามขึ้นมาใหม่นี้สามารถนำเอาคุณลักษณะของคลาสที่มีอยู่แล้วนั้นมาใช้ แล้วนิยามเพิ่มคุณลักษณะอื่นที่จำเป็น กระบวนการนี้เรียกว่า Inheritance

2.3 แนวคิดในการทำแผนแบบเชิงวัตถุ

แนวคิดในการแปลงแผนแบบเชิงสัมพันธ์เป็นแผนแบบเชิงวัตถุใช้ 2 แนวคิดคือ แนวคิดการเชื่อมโยงและความเกี่ยวพัน (Link and Association Concepts) และแนวคิดการทำให้มีลักษณะทั่วไป (Generalization Concepts)

2.3.1 แนวคิดการเชื่อมโยงและความเกี่ยวพัน (Link and Association Concepts) [2]

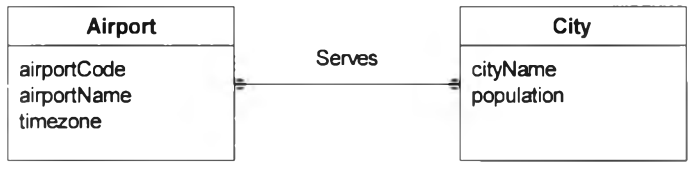
เป็นแนวคิดในการทำแผนแบบเชิงวัตถุ (Object models) แบบหนึ่งโดยอาศัยความสัมพันธ์ในลักษณะการเชื่อมโยงระหว่างอ็อบเจคท์ กล่าวคือถ้าอ็อบเจคท์ 2 อ็อบเจคท์ มีความสัมพันธ์กันเรียกว่ามีการเชื่อมโยง (Link) ระหว่างกัน การเชื่อมโยงที่มีความสัมพันธ์และโครงสร้างทำนองเดียวกันนำมาสร้างความสัมพันธ์และโครงสร้างสำหรับการเชื่อมโยงเหล่านั้นเรียกว่า ความเกี่ยวพัน (Association) หรือกล่าวอีกนัยหนึ่ง Link คือ Instance ของ Association นั่นเอง



รูปที่ 2.3 ความสัมพันธ์ระหว่างอ็อบเจคท์เรียกว่า Link

รูปที่ 2.3 สนามบิน IAH และสนามบิน HOU ต่างก็ Serve เมือง Houston หรือพูดอีกนัยหนึ่งอ็อบเจคท์ IAH Servers อ็อบเจคท์ Houston และอ็อบเจคท์ HOU Servers อ็อบเจคท์ Houston

จากรูปที่ 2.3 แสดงความสัมพันธ์ระหว่างอ็อบเจคท์ (IAH กับ Houston และ HOU กับ Houston) นำมาสร้างเป็นความสัมพันธ์ระหว่างคลาสดังรูปที่ 2.4 จะได้ความสัมพันธ์ระหว่างคลาส Airport กับคลาส City เป็นคลาส Airport Servers คลาส City



รูปที่ 2.4 ความสัมพันธ์ระหว่างคลาสเรียกว่า Association

2.3.2 แนวคิดการทำให้มีลักษณะทั่วไป (Generalization Concepts) [2]

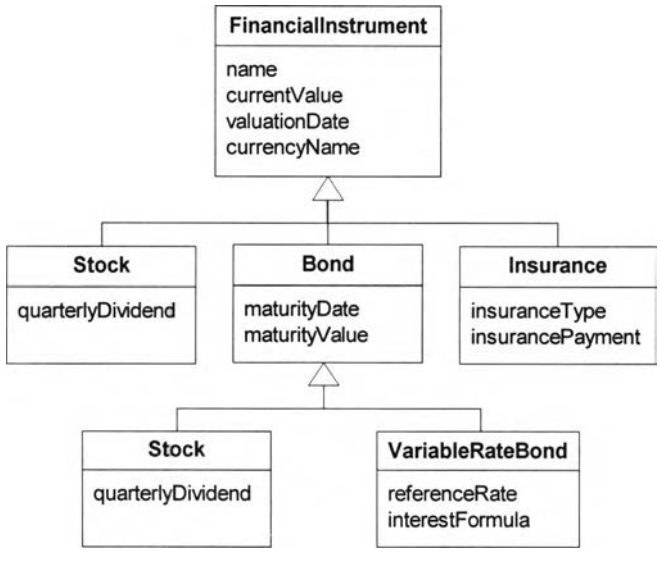
การทำให้มีลักษณะทั่วไป (Generalization)

การทำให้มีลักษณะทั่วไป (Generalization) เป็นความสัมพันธ์กันระหว่างคลาสพ่อ (Parent Class หรือ Superclass) กับคลาสลูก(Child Class หรือ Subclass) ตั้งแต่ 1 คลาสขึ้นไป การจัดจำแนกคลาสตามแนวคิดการทำให้มีลักษณะทั่วไป นี้จะจัดแยกคลาสตามความเหมือนและความแตกต่างกันของคลาสเหล่านั้น คลาสพ่อจะประกอบด้วยคุณลักษณะสามัญทั่วไปที่คลาสลูกทุกคลาสต้องมี และคลาสลูกจะประกอบด้วยคุณลักษณะ

ทุกประการที่คลาสพ่อมีรวมเข้ากับคุณลักษณะบางประการที่ทำให้คลาสลูกแตกต่างจากคลาสพ่อ

การสืบทอดคุณสมบัติ (Inheritance)

การทำให้มีลักษณะทั่วไป (Generalization) ทำให้กลไกการสืบทอดคุณสมบัติเกิดมีขึ้นได้ โดยที่คลาสลูกจะมีการสืบทอดคุณลักษณะต่างๆของคลาสพ่อทุกประการ เช่นแอททริบิวท์ (Attribute) ทุกแอททริบิวท์ โอเปอร์เรชัน(Operation)ทุกโอเปอร์เรชันของคลาสพ่อจะถูกสืบทอดเป็นของคลาสลูกด้วย



รูปที่ 2.5 Generalization จัดจำแนกคลาสตามความเหมือนกันและความแตกต่างกัน

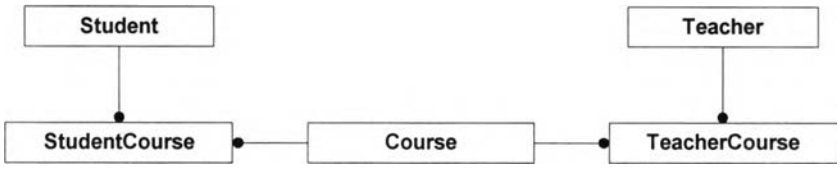
รูปที่ 2.5 แสดงการจัดแยกจำแนกคลาสเป็นลำดับชั้นเนื่องจากกระบวนการทำให้มีลักษณะทั่วไป(Generalization)

2.4 แผนแบบเชิงสัมพันธ์ (Relational Model) [1]

แนวคิดพื้นฐานของแผนแบบเชิงสัมพันธ์คือความสัมพันธ์หรือตาราง (Relation or Table) ซึ่งกำหนดคุณลักษณะของข้อมูลในรูปของตาราง 2 มิติประกอบด้วยคอลัมน์หรือฟิลด์ (Columns หรือ Fileds) เป็นแอททริบิวท์ (Attribute) ต่างๆ เช่น ชื่อ อายุ และอื่นๆเป็นต้น แต่ละทูเปิล (Tuple) หรือโรว์ (Row) ของข้อมูลจะแสดงข้อมูลหนึ่งรายการ

ข้อมูลแต่ละรายการ จะต้องมีคอลัมน์หรือฟิลด์ที่แสดงความเป็นหนึ่งเดียวของรายการข้อมูลในตารางนั้นๆเรียกว่า Unique Identification หรือคีย์ (Key) สำหรับข้อมูลรายการนั้น นอกจากนี้แผนแบบเชิงสัมพันธ์ยังมีแนวคิดเรื่องคีย์นอก (Foreign Keys) ซึ่งเป็นคีย์หลัก (Primary Key) ของตารางอื่นที่ช่วยในการเชื่อมความสัมพันธ์ระหว่างตาราง

ความสัมพันธ์ระหว่างเอนทิตี (Entity) ของแผนแบบเชิงสัมพันธ์จำแนกเป็น 3 ลักษณะคือ ONE-TO-ONE ONE-TO-MANY และ MANY-TO-MANY



รูปที่ 2.6 แสดงแผนแบบเชิงสัมพันธ์

จากรูปที่ 2.6 แสดงส่วนหนึ่งของแผนแบบเชิงสัมพันธ์ของการเรียนการสอนในมหาวิทยาลัย Student มีความสัมพันธ์กับ StudentCourse แบบ ONE-TO-MANY

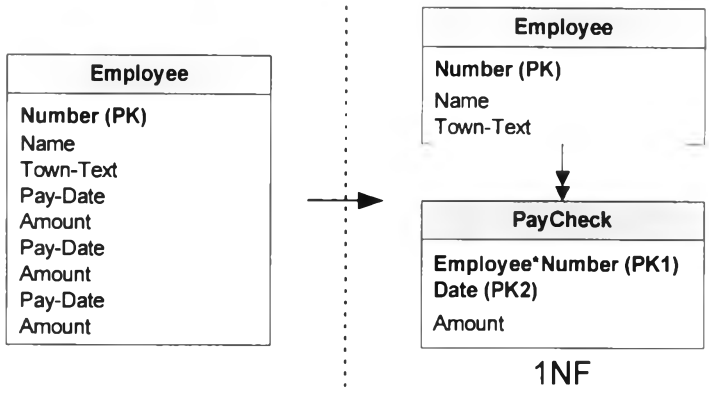
2.5 การทำให้เป็นบรรทัดฐาน (Normalization) [3]

การจัดแผนแบบข้อมูลให้อยู่ในรูปแบบบรรทัดฐาน (Normal Form) เรียกว่า การทำให้เป็นบรรทัดฐาน (Normalization) วัตถุประสงค์คือเพื่อความถูกต้องของข้อมูล(Correctness) ความคงตัวของข้อมูล(Consistentcy) ลดความซ้ำซ้อนของข้อมูล(Nonredundancy) และความมีเสถียรภาพของข้อมูล(Stability)

ผลลัพธ์การทำให้เป็นบรรทัดฐานเรียกว่ารูปแบบบรรทัดฐาน (Normal Form) กฎการทำให้เป็นบรรทัดฐานมีดังนี้

2.5.1 รูปแบบบรรทัดฐานแบบที่หนึ่ง (First Normal Form - 1NF)

เอนทิตีจะอยู่ในรูปของรูปแบบบรรทัดฐานแบบที่หนึ่งจะต้องไม่มีแอททริบิวท์ที่ซ้ำซ้อนกัน



รูปที่ 2.7 การจัดรูปแบบบรรทัดฐานแบบที่หนึ่ง

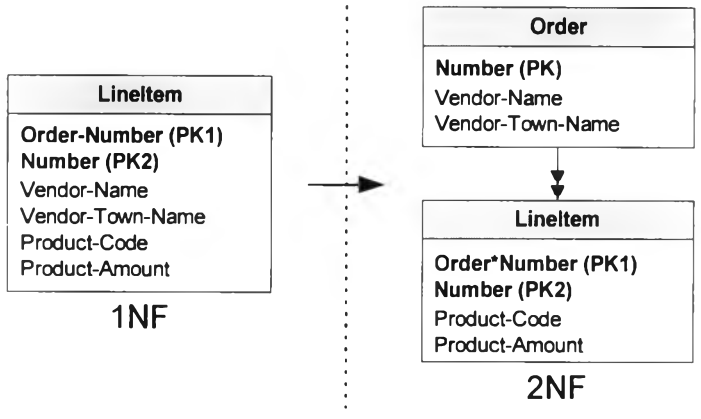
รูปที่ 2.7 เอนทิตี Employee ด้านซ้ายมือมีแอททริบิวท์ Pay-Date และ Amount 3 ชุด เมื่อผ่านการจัดรูปแบบบรรทัดฐานแบบที่หนึ่งแล้วจะได้เอนทิตี Employee และเอนทิตี PayCheck ดังในรูปด้านขวามือ

2.5.2 **รูปแบบบรรทัดฐานแบบที่สอง** (Second Normal Form – 2NF)

เอนทิตีที่อยู่ในรูปแบบบรรทัดฐานแบบที่สอง เมื่อเอนทิตีนั้นอยู่ในรูปแบบบรรทัดฐานแบบที่หนึ่ง และทุกแอททริบิวท์ที่ไม่ใช่แอททริบิวท์ซึ่งประกอบเป็นคีย์หลักจะต้องขึ้นกับคีย์หลักทั้งหมด

รูปที่ 2.8 ด้านซ้ายมือเอนทิตี LineItem คีย์หลักประกอบด้วยแอททริบิวท์ Order_Number กับ Number แอททริบิวท์ Product-Code และแอททริบิวท์ Product-Amount ขึ้นกับคีย์หลักทั้งหมดในขณะที่แอททริบิวท์ Vendor-Name และแอททริบิวท์ Vendor-Town-Name ขึ้นกับแอททริบิวท์ Order-Number เท่านั้น เมื่อผ่านการจัดรูปแบบบรรทัดฐานแบบที่สองแล้วจะได้เอนทิตี 2 เอนทิตีคือ

- Order มีแอททริบิวท์ Number Vendor-Name และ Vendor-Town-Name โดยที่แอททริบิวท์ Number เป็นคีย์หลัก
- LineItem มีแอททริบิวท์ 4 แอททริบิวท์คือ Order*Number Number Product-Code และ Product-Amount โดยที่แอททริบิวท์ Order*Number และแอททริบิวท์ Number ประกอบเป็นคีย์หลัก

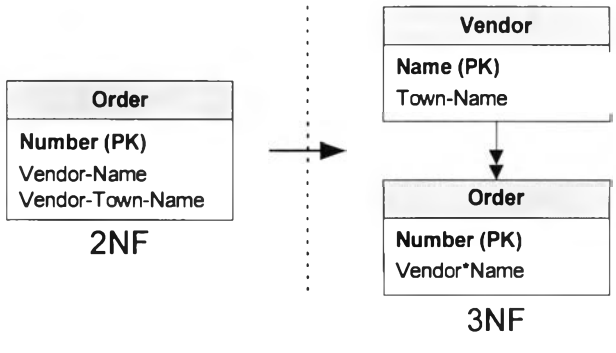


รูปที่ 2.8 การจัดรูปแบบบรรทัดฐานแบบที่สอง

2.5.3 **รูปแบบบรรทัดฐานแบบที่สาม** (Third Normal Form - 3NF)

เอนทิตีที่อยู่ในรูปแบบบรรทัดฐานแบบที่สาม เมื่อเอนทิตีนั้นอยู่ในรูปแบบบรรทัดฐานแบบที่สอง และทุกแอททริบิวท์ที่ไม่ใช่แอททริบิวท์ที่ประกอบเป็นคีย์หลักจะต้องไม่ขึ้นแก่กันและกัน

รูปแบบบรรทัดฐานแบบที่สองนำมาจัดเป็นรูปแบบบรรทัดฐานแบบที่สาม โดยการแยกแอททริบิวท์ที่ขึ้นกับแอททริบิวท์อื่นซึ่งไม่ใช่แอททริบิวท์ที่ประกอบเป็นคีย์หลัก

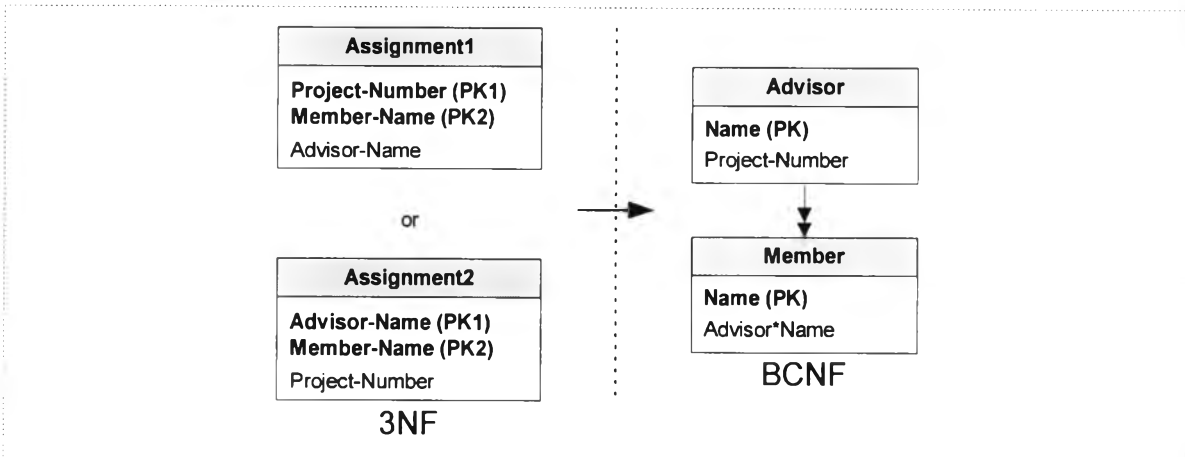


รูปที่ 2.9 การจัดรูปแบบบรรทัดฐานแบบที่สาม

รูปที่ 2.9 เอนทิตี Order ประกอบด้วยแอททริบิวท์ Number Vendor-Name และ Vendor-Town-Name โดยมีแอททริบิวท์ Number เป็นคีย์หลัก แอททริบิวท์ Vendor-Town-Name ขึ้นกับแอททริบิวท์ Vendor-Name โดยตรง ฉะนั้นจากรูปด้านซ้ายมือเมื่อผ่านการจัดรูปแบบบรรทัดฐานแบบที่สามแล้วจะได้เอนทิตี 2 เอนทิตีที่สัมพันธ์กันคือ Vendor และ Order โดยที่เอนทิตี Vendor มีแอททริบิวท์ Name เป็นคีย์หลัก เอนทิตี Order มีแอททริบิวท์ Number เป็นคีย์หลัก

2.5.4 **รูปแบบบรรทัดฐาน** Boyce/Codd (Boyce/Codd Normal Form - BCNF)

เอนทิตีจะอยู่ในรูปของรูปแบบบรรทัดฐาน Boyce/Codd ก็ต่อเมื่อเอนทิตีนั้นอยู่ในรูปแบบบรรทัดฐานที่สามและแอททริบิวท์แต่ละแอททริบิวท์ขึ้นกับคีย์หลักทั้งหมด



รูปที่ 2.10 การจัดรูปแบบบรรทัดฐาน Boyce/Codd

รูปที่ 2.10 เอนทิตี Assignment1 อยู่ในรูปของรูปแบบบรรทัดฐานแบบที่สาม โดยที่ Advisor-Name ขึ้นกับคีย์หลักทั้งหมด (Project-Name กับ Member-Name) ในขณะที่ Assignment2 ไม่อยู่ในรูปแบบบรรทัดฐานแบบที่สามเนื่องจาก Project-Number ขึ้นกับ Advisor-Name เท่านั้น

Project-Number	Member-Name	Advisor-Name
101	A	Brown
101	B	Brown
101	C	Graham
492	A	Thomas
492	D	Thomas
492	B	Andrews

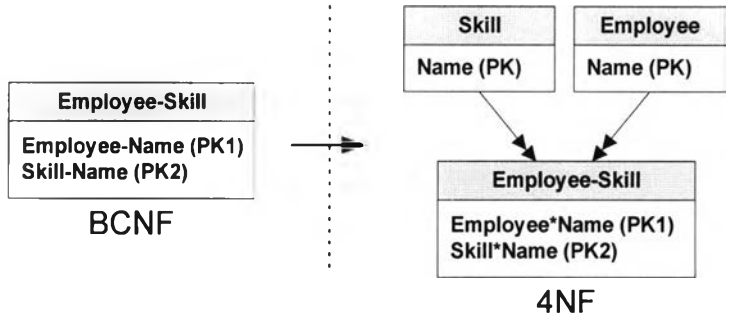
ตารางที่ 2.1 รายการข้อมูลสำหรับเอนทิตี Assignment1 หรือ Assignment2

ตารางที่ 2.1 รายการข้อมูลสำหรับเอนทิตี Assignment1 หรือ Assignment2 จะเห็นว่า ถ้า Brown เปลี่ยนชื่อเป็น Browning จะต้องแก้ไขข้อมูล 2 รายการ และถ้า C ลาออกจากโครงการ 101 ข้อมูลที่บอกว่า Graham เป็นที่ปรึกษาโครงการหมายเลข 101 จะสูญหายไปด้วย

การแก้ไขเพื่อลดความซ้ำซ้อนเหล่านี้ทำได้โดยการแยกเอนทิตี Assignment1 หรือ Assignment2 เป็น 2 เอนทีตีคือ Advisor และ Member ดังรูปที่ 2.10 ด้านขวามือ

2.5.5 **รูปแบบบรรทัดฐานแบบที่สี่** (Fourth Normal Form - 4NF)

เมื่อเอนทิตีซึ่งมีแอททริบิวท์หลายตัวประกอบกันเป็นคีย์หลัก รูปแบบบรรทัดฐานแบบที่สี่เป็นการจำแนกและแยกแอททริบิวท์ซึ่งประกอบกันเป็นคีย์หลัก และเป็นอิสระต่อกันเป็นเอนทิตีใหม่เรียกว่าเอนทิตีพ่อ (Parent Entity) ส่วนเอนทิตีดั้งเดิมเป็นเอนทิตีลูก

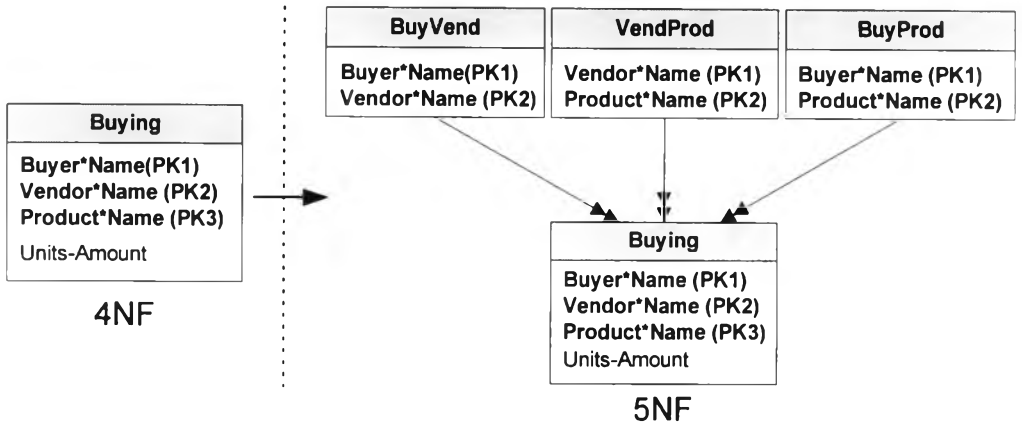


รูปที่ 2.11 การจัดรูปแบบบรรทัดฐานแบบที่สี่

รูปที่ 2.11 เอนทีตี Employee-Skill คีย์หลักประกอบด้วยแอททริบิวท์ Employee-Name และ Skill-Name เมื่อผ่านการทำให้เป็นรูปแบบบรรทัดฐานแบบที่สี่แล้วจะได้เอนทิตีใหม่ 2 ตัวเป็นเอนทิตีพ่อดังรูปด้านขวามือ

2.5.6 **รูปแบบบรรทัดฐานแบบที่ห้า** (Fifth Normal Form - 5NF)

เมื่อเอนทิตีซึ่งมีแอททริบิวท์ตั้งแต่ 3 ตัวขึ้นไปประกอบเป็นคีย์หลักและแต่ละคู่ของแอททริบิวท์หล่านั้นขึ้นแก่กันและกันในวนลักษณะเป็นวงกลม ทำให้สามารถแยกเป็นเอนทิตีย่อยออกมาเป็น 3 เอนทิตีหรือมากกว่า การแยกเอนทิตีเป็นเอนทิตีย่อยเหล่านี้เรียกว่าการทำให้อยู่ในรูปแบบบรรทัดฐานแบบที่ห้า



รูปที่ 2.12 การจัดรูปแบบบรรทัดฐานแบบที่ห้า

รูปที่ 2.12 เอนทิตี Buying คีย์หลักประกอบด้วยแอททริบิวท์ Buyer*Name Vendor*Name และ Product*Name คู่ของแอททริบิวท์ Buyer*Name กับ Vendor*Name ต่างก็ขึ้นแก่กันและกัน คู่ของแอททริบิวท์ Vendor*Name กับ Product*Name ต่างก็ขึ้นแก่กันและกัน และคู่ของแอททริบิวท์ Product*Name กับ Vendor*Name ต่างก็ขึ้นแก่กันและกัน จะเห็นว่าการขึ้นแก่กันและกันของคู่แอททริบิวท์เหล่านั้นมีลักษณะวนเป็นวงกลม กรณีเช่นนี้สามารถนำเอนทิตีนี้มาผ่านการทำรูปแบบบรรทัดฐานแบบที่ห้า โดยการสร้างเอนทิตีใหม่ 3 ตัว เป็นเอนทิตีพ่อดังรูปด้านขวามือ

การแปลงแผนแบบเชิงสัมพันธ์เป็นแผนแบบเชิงวัตถุ แผนแบบเชิงสัมพันธ์จะต้องผ่านการทำให้เป็นบรรทัดฐานก่อน อย่างน้อยต้องอยู่ในรูปแบบบรรทัดฐานแบบที่สาม มิฉะนั้นแผนแบบเชิงวัตถุที่ได้อาจจะไม่ถูกต้องมากนัก

2.6 **การแปลงแผนแบบเชิงวัตถุเป็นแผนแบบเชิงสัมพันธ์** [2]

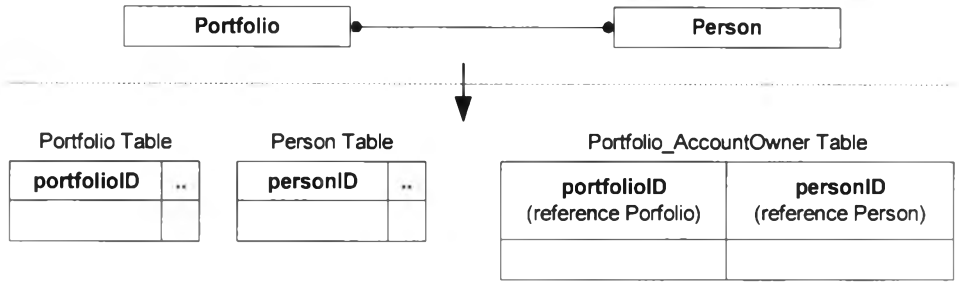
การแปลงแผนแบบเชิงสัมพันธ์เป็นแผนแบบเชิงวัตถุ ไม่มีวิธีการซึ่งสามารถกระทำได้โดยตรง ดังนั้นจึงอาศัยแนวคิดในการแปลงแผนแบบเชิงวัตถุเป็นแผนแบบเชิงสัมพันธ์ โดยใช้วิธีการคิดย้อนกลับ การแปลงแผนแบบเชิงวัตถุเป็นแผนแบบเชิงสัมพันธ์ จำแนกเป็น 2 กรณีดังนี้

2.6.1 **แปลงความเกี่ยวพันระหว่างคลาส** (Association)

การแปลงความเกี่ยวพันระหว่างคลาสเป็นตารางในแผนแบบเชิงสัมพันธ์จำแนกเป็น 4 กรณีคือ

2.6.1.1 แปลงความเกี่ยวพันระหว่างคลาสแบบที่หนึ่ง

คลาส 2 คลาสมีความสัมพันธ์กันแบบ MANY-TO-MANY แปลงเป็นแผนแบบเชิงสัมพันธ์โดยกำหนดให้ 1 คลาสเป็น 1 ตารางและสร้างตารางใหม่ 1 ตารางโดยที่ตารางนี้มีคีย์หลักซึ่งประกอบด้วยคีย์นอก (Foreign Key) ชี้ไปยังตาราง 2 ตารางซึ่งได้จากการแปลงโดยตรงจากคลาสทั้งสอง

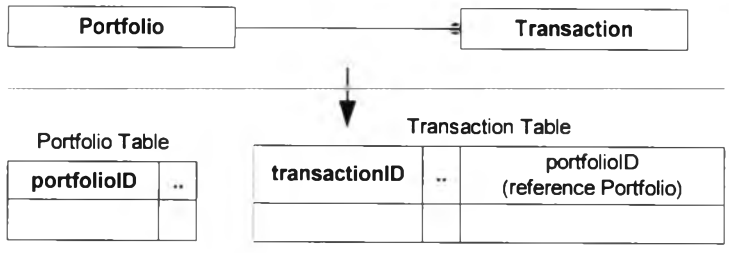


รูปที่ 2.13 การแปลงความเกี่ยวพันระหว่างคลาสแบบที่หนึ่งเป็นแผนแบบเชิงสัมพันธ์

รูปที่ 2.13 คลาส Portfolio กับคลาส Person มีความสัมพันธ์กันแบบ MANY-TO-MANY เมื่อแปลงเป็นแผนแบบเชิงสัมพันธ์แล้วจะได้ตาราง 3 ตารางโดยที่ตาราง Porfolio และตาราง Person ได้จากการแปลงจากคลาส Portfolio และคลาส Person ตามลำดับ และตาราง Portfolio_AccountOwner เป็นตารางซึ่งเกิดขึ้นใหม่มีคีย์หลักซึ่งประกอบด้วยคีย์นอกชี้ไปยังตาราง Portfolio และตาราง Person

2.6.1.2 แปลงความเกี่ยวพันแบบที่สอง

คลาส 2 คลาสมีความสัมพันธ์กันแบบ ONE-TO-MANY แปลงเป็นตารางในแผนแบบเชิงสัมพันธ์โดยกำหนดให้ 1 คลาสเป็น 1 ตารางและกำหนด คีย์นอกให้กับตารางที่แปลงจากคลาสฝั่ง MANY ชี้ไปยังตารางที่แปลงจากคลาสฝั่ง ONE

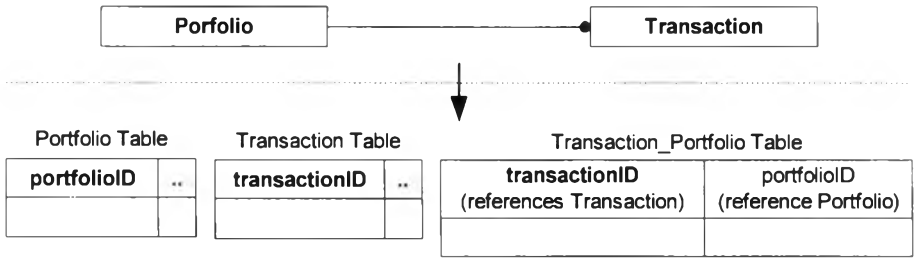


รูปที่ 2.14 การแปลงความเกี่ยวพันระหว่างคลาสแบบที่สองเป็นแผนแบบเชิงสัมพันธ์

รูปที่ 2.14 คลาส Portfolio มีความสัมพันธ์กับคลาส Transaction แบบ ONE-TO-MANY แปลงเป็นตารางในแผนแบบเชิงสัมพันธ์โดยกำหนดให้คลาส Portfolio เป็นตาราง Portfolio และคลาส Transaction เป็นตาราง Transaction โดยที่ตาราง Transaction มีคีย์นอกชี้ไปยังตาราง Portfolio

2.6.1.3 แปลงความเกี่ยวพันแบบที่สาม

คลาส 2 คลาสมีความสัมพันธ์กันแบบ ONE-TO-MANY แปลงเป็นตารางในแผนแบบเชิงสัมพันธ์โดยกำหนดให้ 1 คลาสเป็น 1 ตารางและสร้างตารางใหม่ 1 ตารางเพื่อแสดงความสัมพันธ์ระหว่างตารางทั้งสอง ตารางใหม่นี้ประกอบด้วยคีย์นอก 2 ตัวโดยที่คีย์หลักเป็นคีย์นอกชี้ไปยังตารางซึ่งแปลงจากคลาสฝั่ง MANY

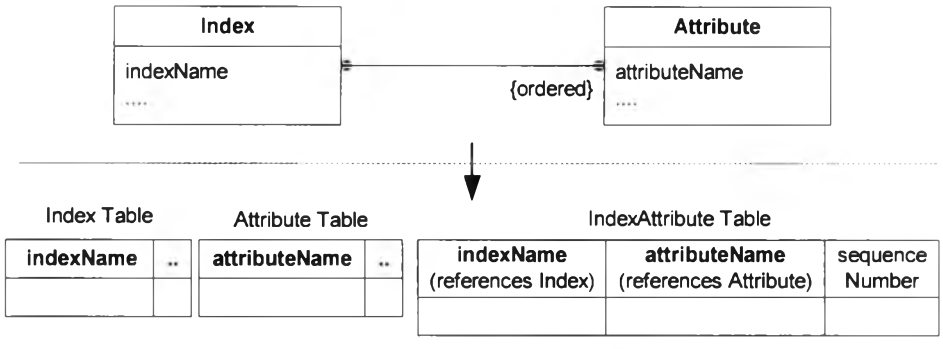


รูปที่ 2.15 การแปลงความเกี่ยวพันระหว่างคลาสแบบที่สามเป็นแผนแบบเชิงสัมพันธ์

รูปที่ 2.15 คลาส Portfolio มีความสัมพันธ์กับคลาส Transaction แบบ ONE-TO-MANY แปลงเป็นแผนแบบเชิงสัมพันธ์โดยกำหนดให้คลาส Portfolio เป็นตาราง Portfolio คลาส Transaction เป็นตาราง Transaction และสร้างตาราง Transaction_Portfolio เพื่อแสดงความสัมพันธ์ระหว่างตาราง Portfolio กับตาราง Transaction โดยที่ตาราง Transaction_Portfolio มีคีย์หลัก transactionID เป็นคีย์นอกชี้ไปยังตาราง Transaction และมี portfolioID เป็นคีย์นอกชี้ไปยังตาราง Portfolio

2.6.1.4 การแปลงความเกี่ยวพันระหว่างคลาสแบบที่สี่

คลาส 2 คลาสมีความสัมพันธ์กันแบบ MANY-TO-MANY และความเกี่ยวพันระหว่างคลาสทั้งสองเป็นความเกี่ยวพันแบบตามลำดับ (Ordered Association) แปลงเป็นตารางในแผนแบบเชิงสัมพันธ์โดยกำหนดให้ 1 คลาสเป็น 1 ตารางและสร้างตารางใหม่ 1 ตารางเพื่อแสดงความสัมพันธ์ระหว่างตารางทั้งสอง ตารางใหม่นี้ประกอบด้วยคีย์นอก 2 ตัวซึ่งประกอบกันเป็นคีย์หลักและคอลัมน์แสดงลำดับที่ของความสัมพันธ์



รูปที่ 2.16 การแปลงความเกี่ยวพันระหว่างคลาสแบบที่สี่เป็นแผนแบบเชิงสัมพันธ์

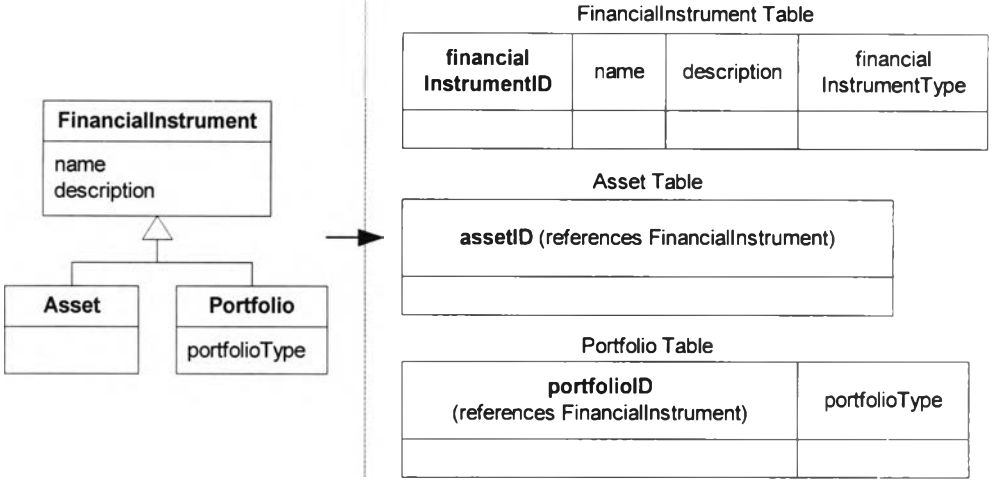
รูปที่ 2.16 คลาส Index กับคลาส Attribute มีความสัมพันธ์ต่อกันแบบ MANY-TO-MANY และเป็นความเกี่ยวพันกันแบบตามลำดับ แปลงเป็นแผนแบบเชิงสัมพันธ์โดยกำหนดให้คลาส Index เป็นตาราง Index คลาส Attribute เป็นตาราง Attribute และสร้างตาราง IndexAttribute เพื่อแสดงความสัมพันธ์ระหว่างตาราง Index กับตาราง Attribute มีคีย์หลักซึ่งประกอบด้วยคีย์นอกชี้ไปยังตาราง Index และตาราง Attribute และมีคอลัมน์ sequenceNumber เพื่อแสดงลำดับของความเกี่ยวพันระหว่างตาราง Index กับตาราง Attribute

2.6.2 การแปลงการสืบทอดคุณสมบัติระหว่างคลาส (Inheritance)

การแปลงการสืบทอดคุณสมบัติระหว่างคลาสเป็นแผนแบบเชิงสัมพันธ์จำแนกเป็น 4 กรณีคือ

2.6.2.1 แปลงการสืบทอดคุณสมบัติระหว่างคลาสแบบที่หนึ่ง

แปลงโดยกำหนดให้ 1 คลาสเป็น 1 ตารางในแผนแบบเชิงสัมพันธ์ การแปลงแบบนี้ตารางที่แปลงจากคลาสพ่อจะต้องมีคอลัมน์ที่กำหนดว่าข้อมูลรายการนั้นเป็นของตารางลูกตารางไหน (คอลัมน์นี้มีชื่อเรียกว่า Discriminator)

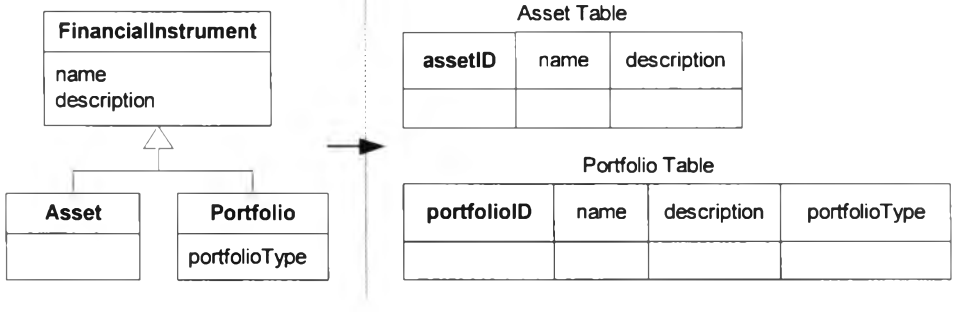


รูปที่ 2.17 การแปลงการสืบทอดคุณสมบัติระหว่างคลาสแบบที่หนึ่ง

รูปที่ 2.17 คลาส FinancialInstrument มีการสืบทอดคุณสมบัติเป็นคลาส Asset และ Portfolio แปลงเป็นตารางในแผนแบบเชิงสัมพันธ์โดยตรง 1 คลาสเป็น 1 ตารางยกเว้นตาราง FinancialInstrument มีคอลัมน์ financialInstrumentType เพื่อใช้ในการจำแนกว่ารายการข้อมูลนั้นๆเป็นของตารางลูกตารางใด

2.6.2.2 แปลงการสืบทอดคุณสมบัติระหว่างคลาสแบบที่สอง

แปลงเป็นแผนแบบเชิงสัมพันธ์โดยการผลักแอททริบิวท์ของคลาสพ่อเป็นแอททริบิวท์ของคลาสลูกทุกคลาส

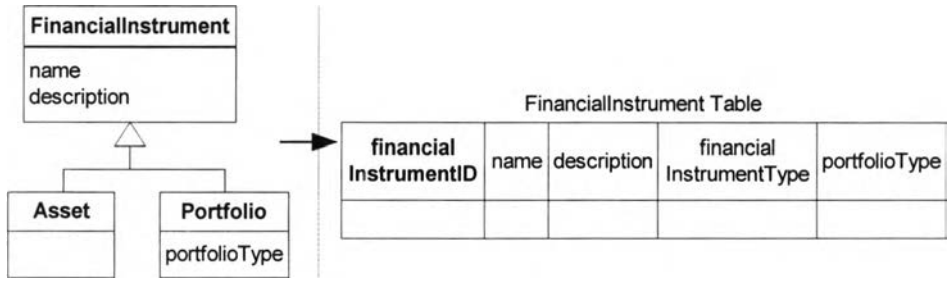


รูปที่ 2.18 การแปลงการสืบทอดคุณสมบัติระหว่างคลาสแบบที่สอง

รูปที่ 2.18 คลาส FinancialInstrument มีการสืบทอดคุณสมบัติเป็นคลาส Asset และ Portfolio แปลงเป็นแผนแบบเชิงสัมพันธ์โดยแปลงเฉพาะคลาส Asset เป็นตาราง Asset และคลาส Portfolio เป็นตาราง Portfolio โดยที่ทุกคอลัมน์ของตาราง Asset ได้จากแอททริบิวท์ของคลาส Asset และแอททริบิวท์ของคลาส FinancialInstrument ทำนองเดียวกันทุกคอลัมน์ของตาราง Portfolio ได้จากแอททริบิวท์ของคลาส Portfolio และแอททริบิวท์ของคลาส FinancialInstrument

2.6.2.3 **แปลงการสืบทอดคุณสมบัติระหว่างคลาสแบบที่สาม**

แปลงเป็นแผนแบบเชิงสัมพันธ์ โดยการผลักแอททริบิวท์ของคลาสลูกทุกคลาสขึ้นเป็นแอททริบิวท์ของคลาสพ่อ

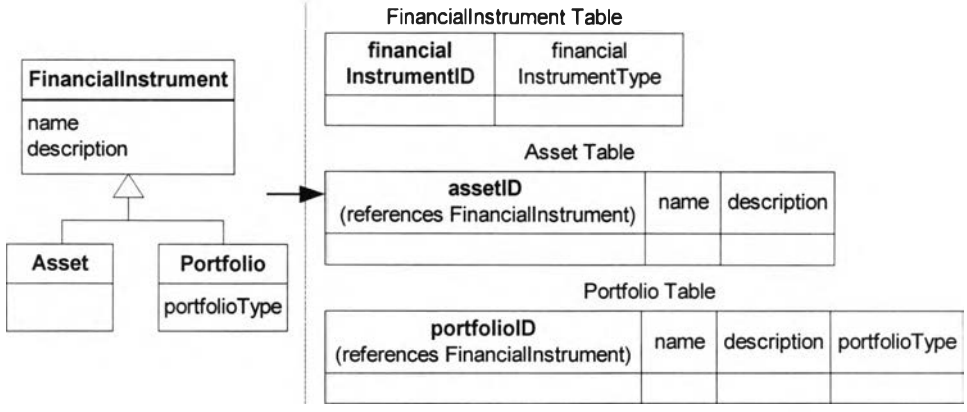


รูปที่ 2.19 การแปลงการสืบทอดคุณสมบัติระหว่างคลาสแบบที่สาม

รูปที่ 2.19 คลาส FinancialInstrument มีการสืบทอดคุณสมบัติเป็นคลาส Asset และ Portfolio แปลงเป็นแผนแบบเชิงสัมพันธ์โดยสร้างเฉพาะตาราง FinancialInstrument เท่านั้น โดยที่ประกอบด้วยคอลัมน์ซึ่งได้จากแอททริบิวท์ของคลาส FinancialInstrument และทุกแอททริบิวท์ของคลาสลูก Asset และ Portfolio

2.6.2.4 **แปลงการสืบทอดคุณสมบัติระหว่างคลาสแบบที่สี่**

เป็นการแปลงแบบผสมกล่าวคือแปลง 1 คลาสเป็น 1 ตาราง โดยการผลักแอททริบิวท์ทุกตัวของคลาสพ่อ ลงมาเป็นแอททริบิวท์ของคลาสลูก ยกเว้นแอททริบิวท์ที่เป็นคีย์หลักและแอททริบิวท์ซึ่งใช้ในการจำแนกคลาสลูก (Discriminator)



ปที่ 2.20 การแปลงการสืบทอดคุณสมบัติระหว่างคลาสแบบที่สี่

รูปที่ 2.20 คลาส FinancialInstrument มีการสืบทอดคุณสมบัติเป็นคลาส Asset และ Portfolio แปลงเป็นแผนแบบเชิงสัมพันธ์โดยกำหนดให้ 1 คลาสเป็น 1 ตาราง โดยที่ตาราง Asset ประกอบด้วยคอลัมน์ซึ่งได้จากแอททริบิวท์ของคลาส Asset และคอลัมน์ซึ่งได้จากแอททริบิวท์ของคลาส FinancialInstrument ทำนองเดียวกันตาราง Portfolio ประกอบด้วยคอลัมน์ซึ่งได้จากแอททริบิวท์ของคลาส Portfolio และคอลัมน์ซึ่งได้จากแอททริบิวท์ของคลาส FinancialInstrument

2.7 แนวคิดระบบจัดการฐานข้อมูล (DBMS Concepts) [1]

แนวคิดเบื้องต้นของระบบจัดการฐานข้อมูล คือผู้ใช้งานสามารถใช้ข้อมูลร่วมกันได้อย่างถูกต้องและมีประสิทธิภาพ คุณสมบัติของระบบจัดการฐานข้อมูลนี้เรียกว่ามีคุณสมบัติ ACID (Atomicity, Consistency, Isolation, Durability)

- อะตอมมิกซิตี (Atomicity) คือคุณสมบัติของระบบจัดการฐานข้อมูลที่ต้องกระทำงานทั้งหมดในหนึ่งชุดของการทำงานให้เสร็จสิ้น และถ้าไม่สามารถกระทำได้หมดก็จะไม่กระทำเลย

- ความคงตัว (Consistency) คือคุณสมบัติของระบบจัดการฐานข้อมูล ในการเปลี่ยนสถานะของฐานข้อมูล จากสถานะคงตัว (Consistent) หนึ่ง ไปยังสถานะคงตัวอีกสถานะหนึ่ง

- ไอโซเลชัน (Isolation) คือคุณสมบัติที่ระบบจัดการฐานข้อมูลจะต้องจัดการเพื่อป้องกันการเกิดความขัดแย้งกันระหว่างทรานแซกชันเมื่อมีการทำงานพร้อมกัน

- ทนทาน (Durability) คือคุณสมบัติของระบบจัดการฐานข้อมูลที่ผลลัพธ์ของความสำเร็จสมบูรณ์ (Committed) ของการทำงานของทรานแซกชันจะถูกบันทึกถาวรในฐานข้อมูล และต้องไม่สูญหายเมื่อเกิดความ ผิดพลาดจากการทำงานในภายหลัง

2.8 ระบบจัดการฐานข้อมูลกับออบเจคท์ (DBMS and Objects) [1]

คุณสมบัติ ACID ทำให้ระบบจัดการฐานข้อมูลมีความน่าเชื่อถือและความปลอดภัยในการทำงาน สนองตอบการใช้งานของผู้ใช้งานหลายคนได้อย่างมีประสิทธิภาพ

แผนแบบเชิงวัตถุเมื่อเพิ่มคุณสมบัติ ACID เข้าไปจะทำให้อ็อบเจคท์มีคุณสมบัติที่เรียกว่า Persistent หรือกล่าวอีกนัยหนึ่งเมื่อสามารถจัดเก็บอ็อบเจคท์บนระบบจัดการฐานข้อมูลซึ่งมีคุณสมบัติ ACID แล้วอ็อบเจคท์เหล่านั้นเรียกว่า Persistent Object กล่าวคืออ็อบเจ็คท์ก่อนที่จะถูกบันทึกลงฐานข้อมูลมีคุณลักษณะเช่นไร เมื่อบันทึกแล้วและอ่านกลับมาใช้ใหม่ คุณลักษณะต่างๆ ก็ยังคงถูกต้องครบถ้วนเหมือนก่อนที่จะมีการบันทึกลงฐานข้อมูล

ระบบจัดการฐานข้อมูลที่มีคุณสมบัติ ACID และสามารถจัดเก็บอ็อบเจคท์ได้โดยตรงและเมื่อต้องการอ่านนำไปใช้สามารถอ่านไปใช้ได้โดยตรง เรียกระบบจัดการนี้ว่าระบบจัดการฐานข้อมูลเชิงวัตถุ (OODBMS)