บทที่ 5

การทดสอบโปรแกรม

การทดสอบโปรแกรมจะทดสอบผลการทำงานของโปรแกรม ว่าเป็นไปตามความคาดหมายหรือให้ผลลัพธ์เป็นไปตามที่ต้องการหรือไม่

โปรแกรมทำหน้าที่แปลงฐานข้อมูลเชิงสัมพันธ์เป็นฐานข้อมูลเชิงวัตถุ โดยใช้ฐานข้อมูลบนระบบจัดการเชิงสัมพันธ์ Microsoft SQL Server และระบบจัดการฐานข้อมูลเชิงวัตถุ POET เป็นกรณีตัวอย่างทดสอบ

ตัวอย่างข้อมูลที่ใช้ทดสอบเป็นข้อมูลสมมุติ ประกอบด้วยตารางทั้งหมด 18 ตารางในฐานข้อมูลชื่อ Invoice ดังนี้

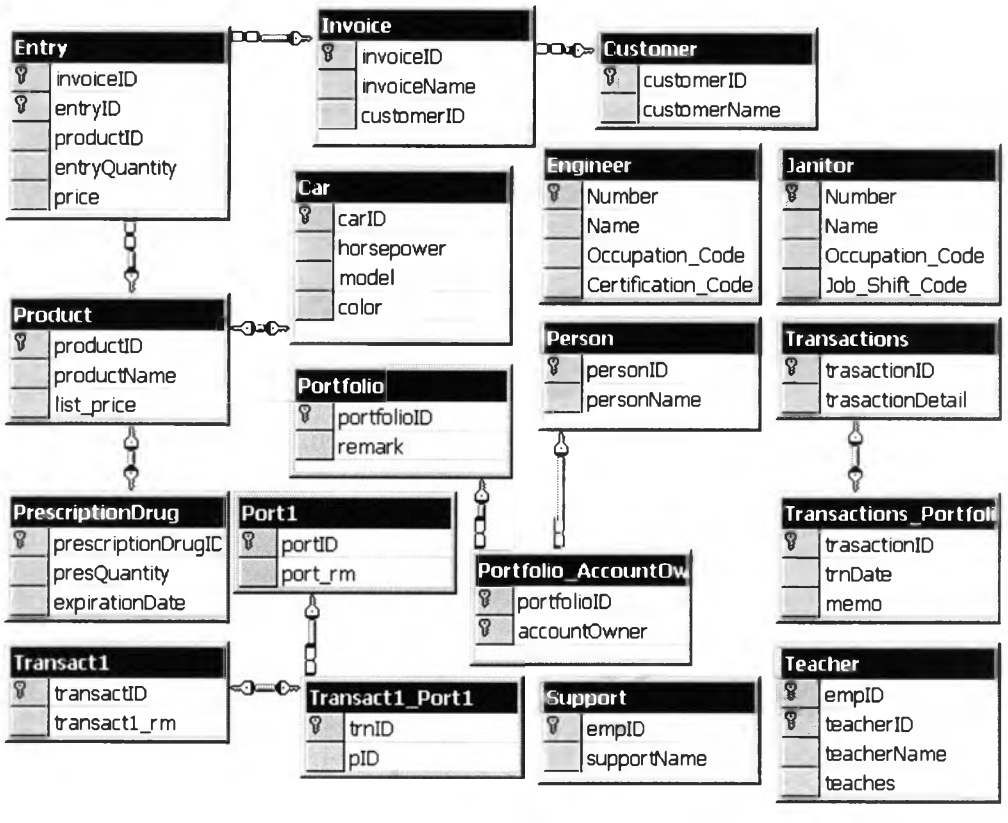
1. Customer
2. Invoice
3. Entry
4. Car
5. PrescriptionDrug
6. Product
7. Janitor
8. Engineer
9. Teacher
10. Support
11. Transaction
12. Transaction_Portfolio
13. Person
14. Portfolio
15. Portfolio_AccountOwner
16. Port1

17. Transact1

18. Transact1_Port1

การทดสอบแบ่งเป็น 2 กรณีคือ

1. การแปลงโครงสร้างในแต่ละกรณีแปลงได้ถูกต้องหรือไม่
2. เมื่อมีการนำเข้าข้อมูลแล้วข้อมูลในทั้งสองฐานข้อมูลสอดคล้องกันหรือไม่



รูปที่ 5.1 ฐานข้อมูลบน Microsoft SQL Server

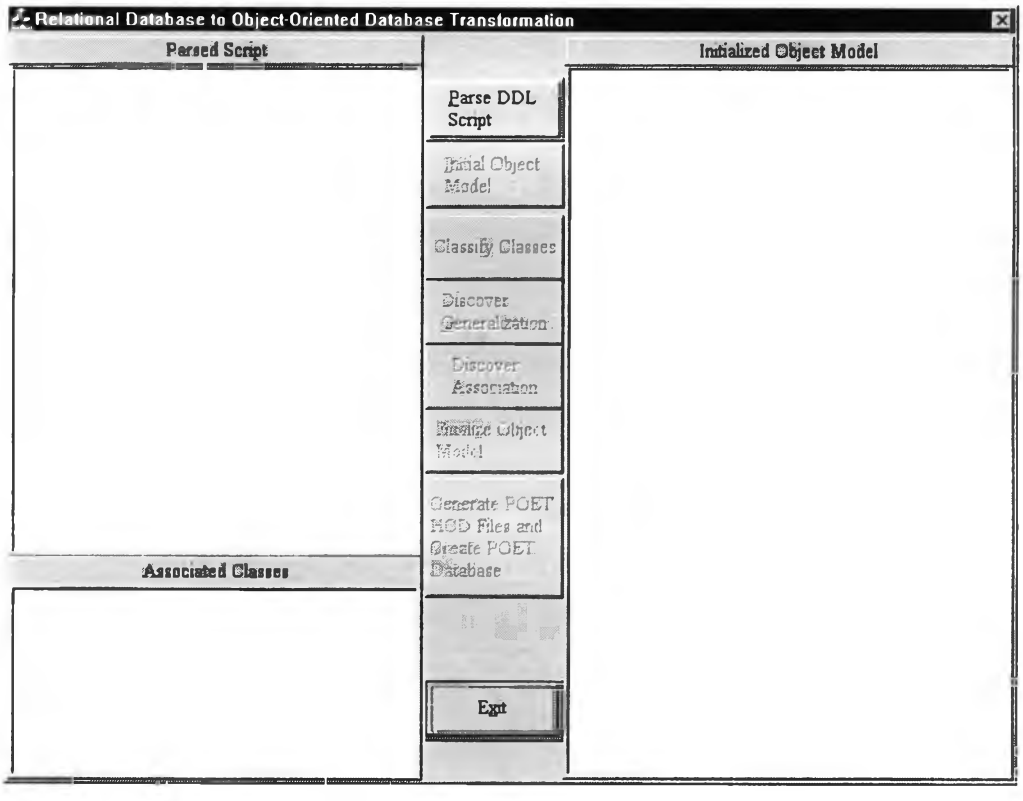
รูปที่ 5.1 แสดงความสัมพันธ์ระหว่างตารางต่างๆในฐานข้อมูล Invoice บน Microsoft SQL Server

5.1 ทดสอบผลการแปลงโครงสร้าง

แบ่งเป็น 2 กรณีคือ ทดสอบการมีลักษณะทั่วไป (Generalization) และทดสอบความเกี่ยวพันระหว่างคลาส (Association)

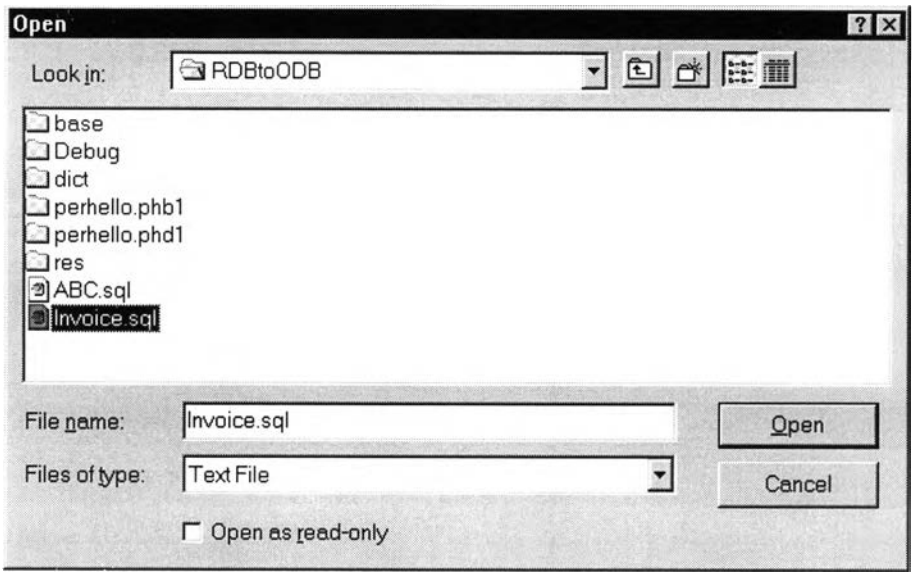
ก่อนทำการทดสอบต้องเตรียมสคริพท์สำหรับเป็นข้อมูลเข้าและ Data Source สำหรับฐานข้อมูล Invoice โดยกำหนดชื่อแฟ้มสคริพท์เป็นชื่อฐานข้อมูลนามสกุล sql ในที่นี้จะได้ชื่อ Invoice.sql วิธีการเตรียมแสดงในภาคผนวก ข. และภาคผนวก ก. ตามลำดับ

เมื่อสั่งโปรแกรมแปลงฐานข้อมูลฯทำงาน จะได้วินโดวส์ดังรูปที่ 5.2



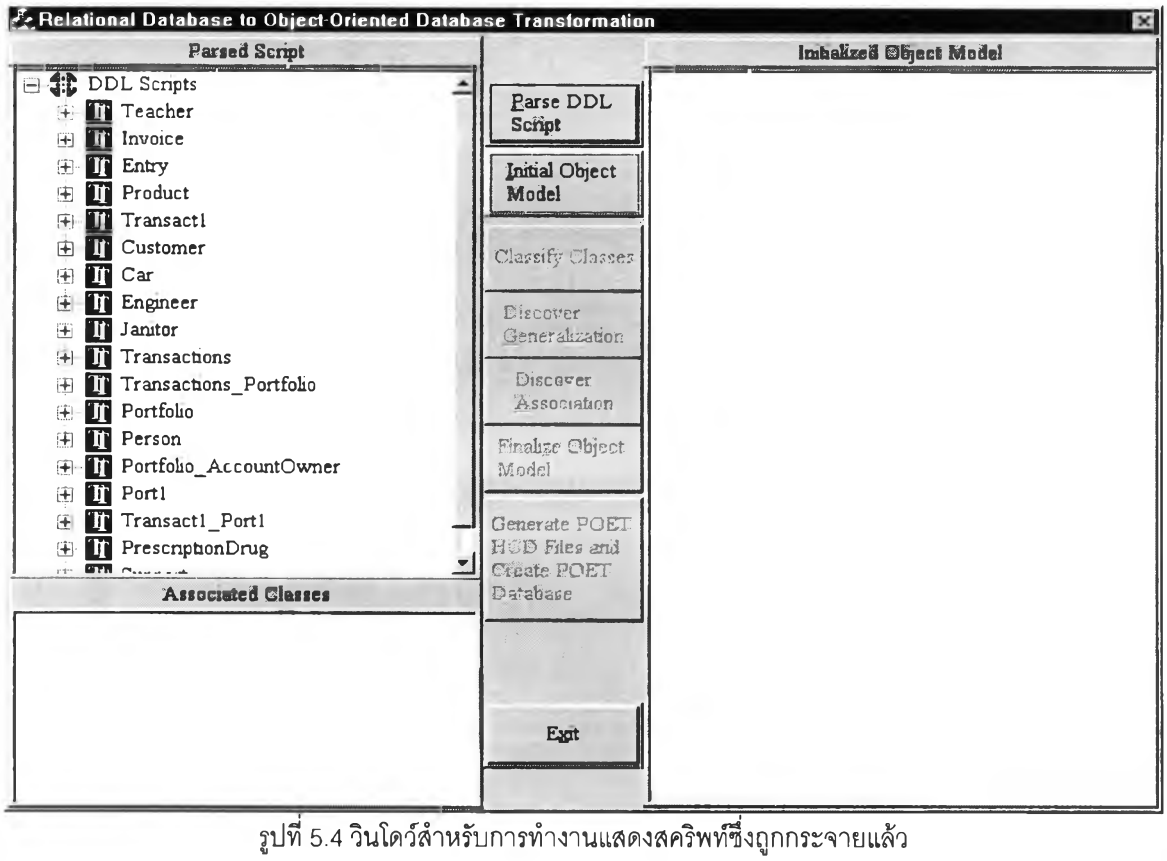
รูปที่ 5.2 วินโดว์การทำงานของโปรแกรมแปลงฐานข้อมูลเชิงสัมพันธ์เป็นฐานข้อมูลเชิงวัตถุ

รูปที่ 5.2 แสดงวินโดวส์วินโดว์การทำงานของโปรแกรมแปลงฐานข้อมูลเชิงสัมพันธ์เป็นฐานข้อมูลเชิงวัตถุ การปรากฏครั้งแรกของวินโดวส์จะสามารถเลือกปุ่มคำสั่ง 2 ปุ่มคือ Exit เพื่อออกจากโปรแกรมหรือ Parse DDL Script เพื่อเปิดแฟ้มสคริพท์ให้โปรแกรมทำงานต่อไป



รูปที่ 5.3 วินโดว์สำหรับการเปิดแฟ้มสคริพท์

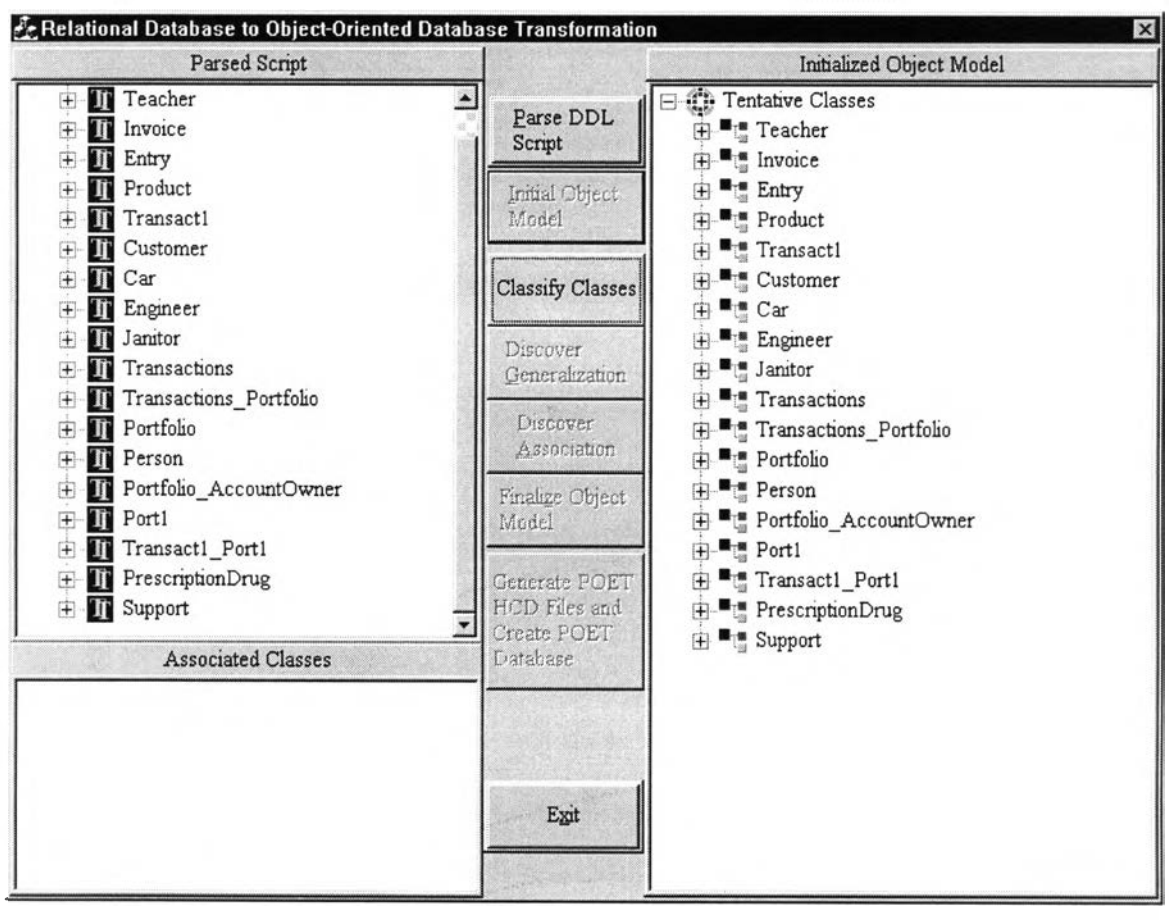
เมื่อเลือกแฟ้มสคริพท์แล้ว โปรแกรมจะทำการกระจายสคริพท์แยกเป็นรายละเอียดที่จำเป็นในการทำงานในขั้นตอนต่อไป



รูปที่ 5.4 วินโดว์สำหรับการทำงานแสดงสคริพท์ซึ่งถูกกระจายแล้ว

รูปที่ 5.4 แสดงผลการทำงานของโปรแกรมหลังจากกระจายสคริพท์แล้ว หลังจากนี้สามารถสั่งให้โปรแกรมทำงานในขั้นตอนต่อไปหรือย้อนกลับไปเริ่มต้นใหม่

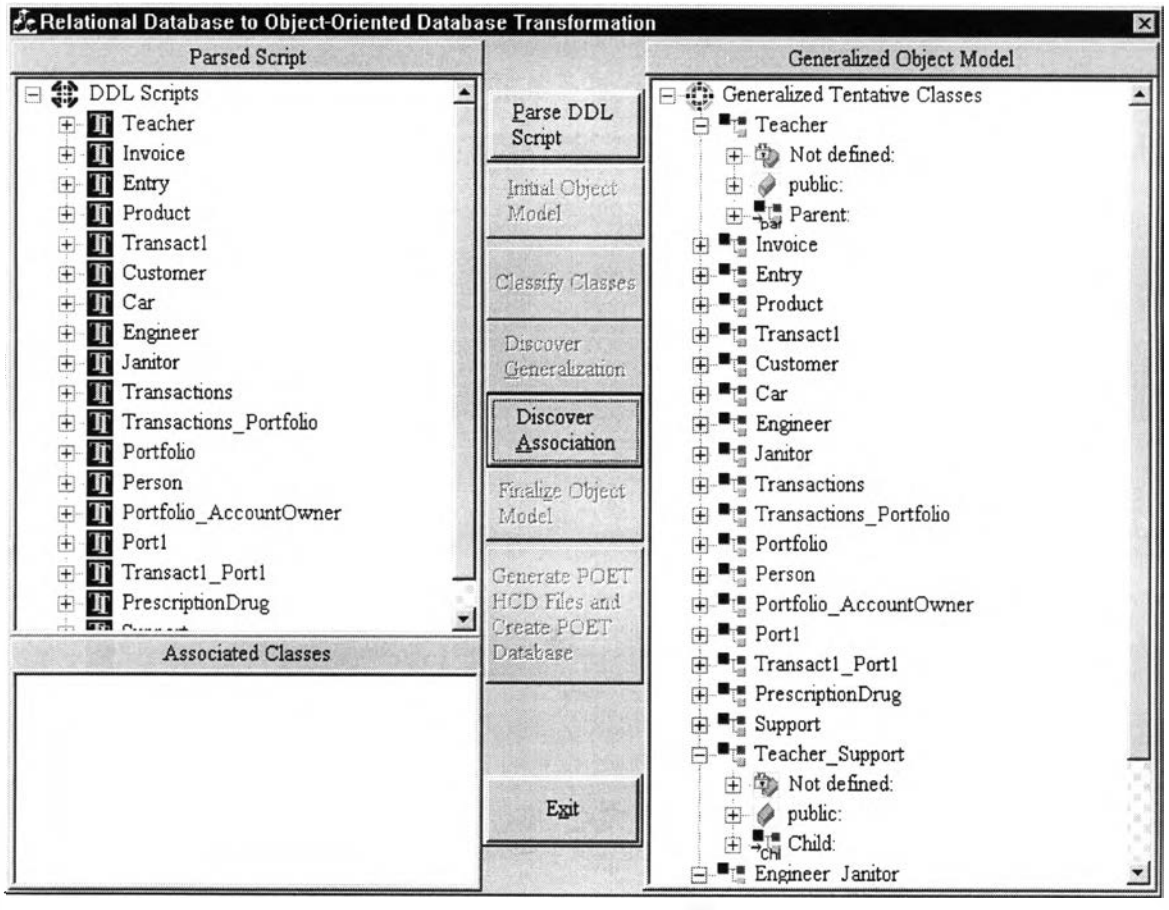
เมื่อเลือกปุ่มคำสั่ง Initial Object Model โปรแกรมจะเริ่มแปลงจากสคริพท์ที่กระจายแล้วในรูปด้านซ้ายมือเป็นแผนแบบเชิงวัตถุเบื้องต้น ได้ผลลัพธ์แสดงในรูปด้านขวาดังรูปที่ 5.5



รูปที่ 5.5 วินโดว์สำหรับการทำงานหลังการเลือกคำสั่ง Initial Object Model

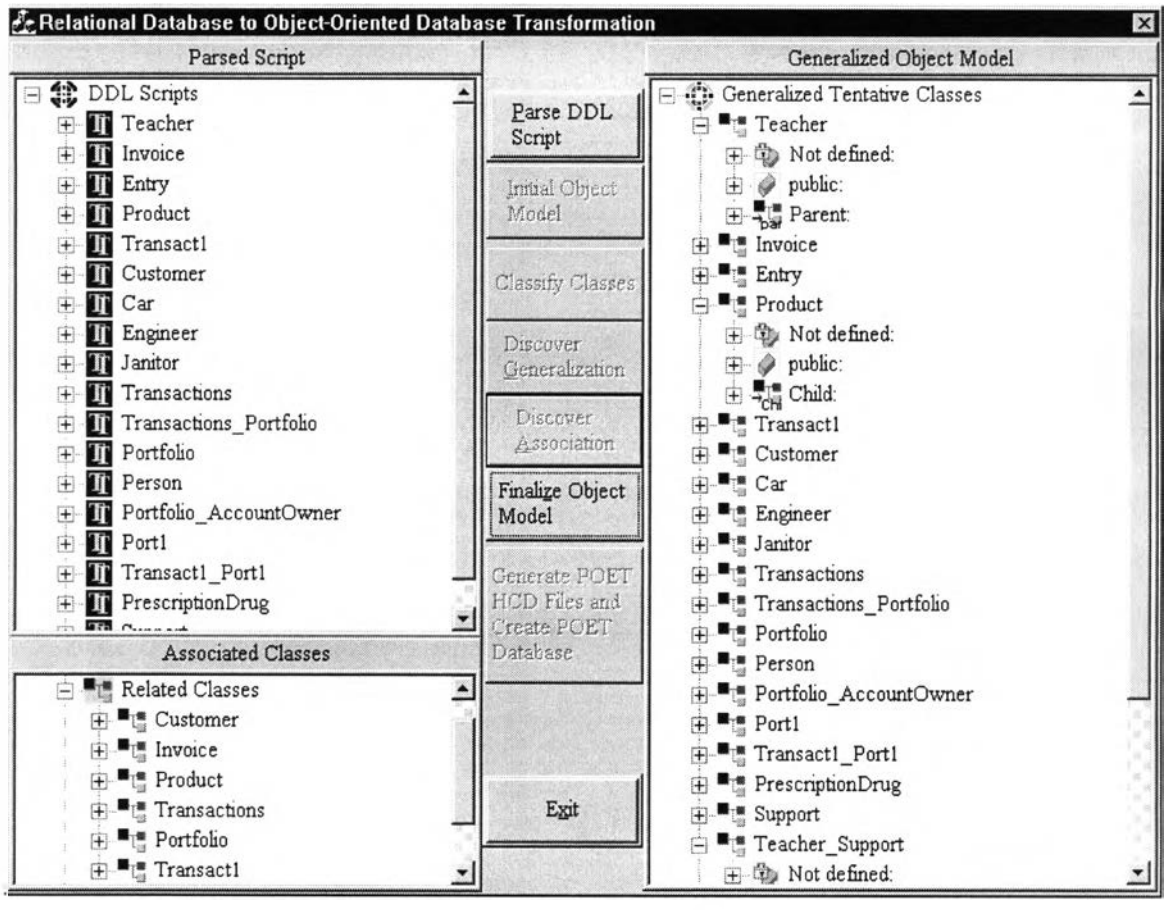
รูปที่ 5.5 รูปด้านขวามือแสดงแผนแบบเชิงวัตถุเบื้องต้น ขั้นตอนนี้จะเป็นการแปลง 1 ตาราง เป็น 1 คลาส ขั้นตอนต่อไปสำหรับการทำงานของโปรแกรมคือการจำแนกคลาสเป็นคลาสประเภทต่างๆ โดยการเลือกปุ่มคำสั่ง Classify Classes

หลังการเลือกปุ่มคำสั่งโปรแกรมยังไม่แสดงผลอะไรให้เห็น ขั้นตอนต่อไปเลือกปุ่มคำสั่ง Discover Generalization เพื่อหาการสืบทอดคุณสมบัติระหว่างคลาส



รูปที่ 5.6 วินโดว์สำหรับการทำงานหลังการเลือกคำสั่ง Discover Generalization

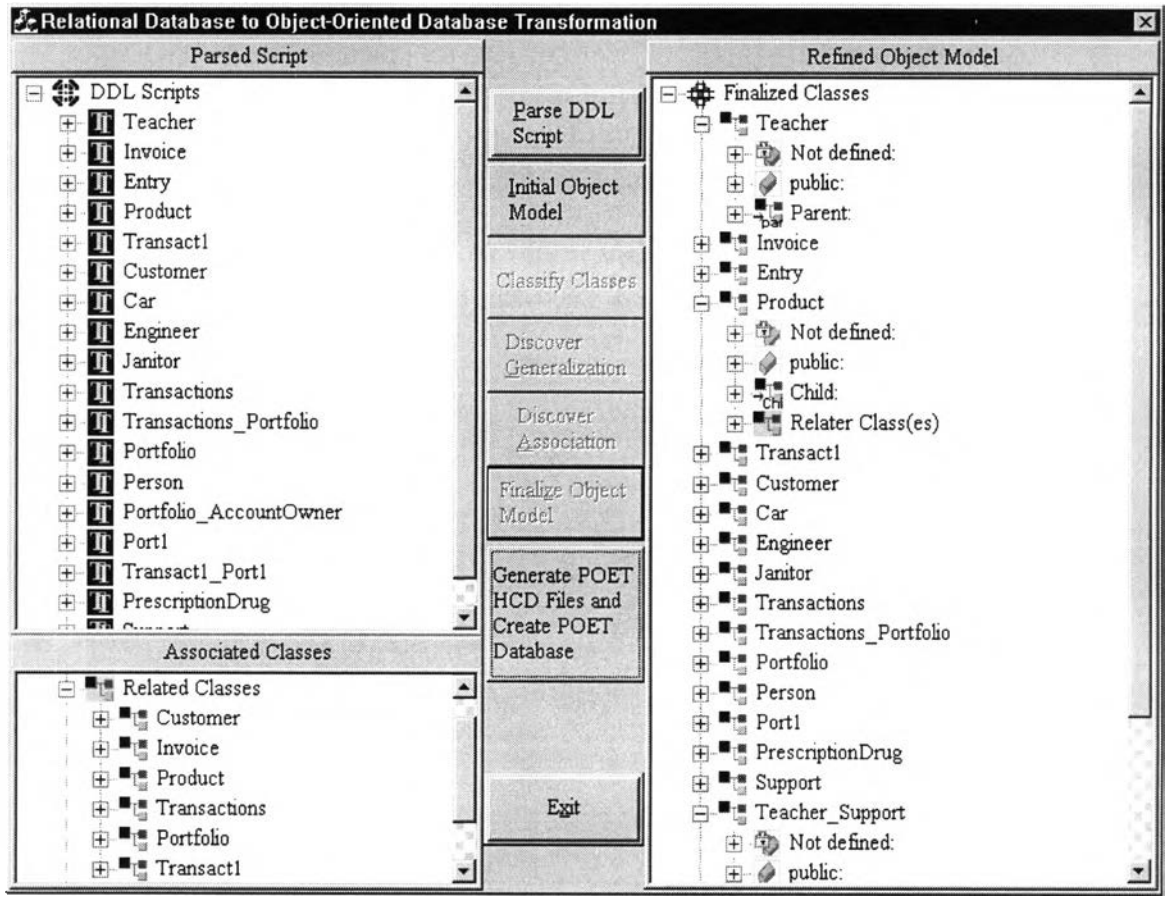
รูปที่ 5.6 ผลการทำงานของโปรแกรมหลังเลือกคำสั่ง Discover Generalization รูปด้านขวามือ ถ้าคลาสใดมีการสืบทอดคุณสมบัติเป็นคลาสอื่นจะขยายออกมาให้เห็นว่ามีคลาสอะไรเป็นคลาสลูก ถัดไปเลือกปุ่มคำสั่ง Discover Association



รูปที่ 5.7 วินโดว์สำหรับการทำงานหลังการเลือกคำสั่ง Discover Association

รูปที่ 5.7 ผลการทำงานของโปรแกรมหลังจากเลือก Discover Association รูปด้ายซ้ายล่างจะแสดงคลาสผู้ถูกอ้างอิงและคลาสผู้อ้างอิง พร้อมความสัมพันธ์ระหว่างคลาสเหล่านั้น

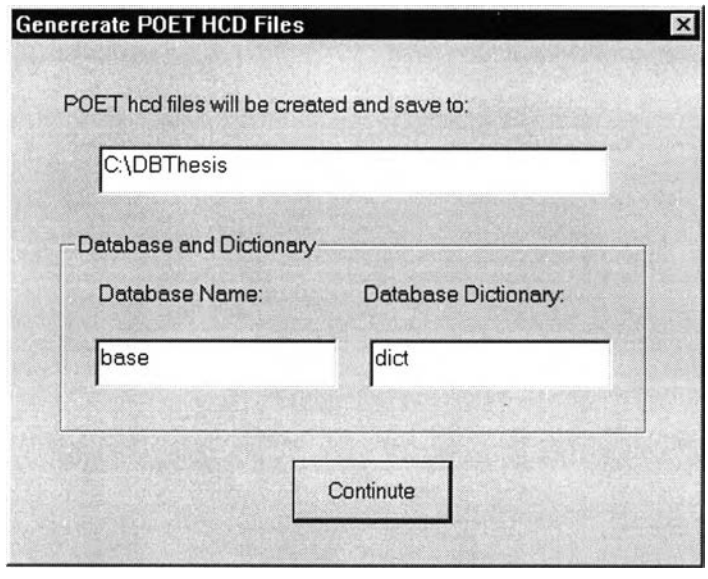
ต่อไปโปรแกรมจะรอให้เลือกปุ่มคำสั่ง Finalize Object Model



รูปที่ 5.8 วินโดว์สำหรับการทำงานหลังการเลือกคำสั่ง Finalize Object Model

รูปที่ 5.8 แสดงผลการทำงานของโปรแกรมหลังเลือกปุ่มคำสั่ง Finalize Object Model รูปด้านขวามือแสดงคลาสต่างๆที่ถูกปรับสมบูรณ์แล้ว

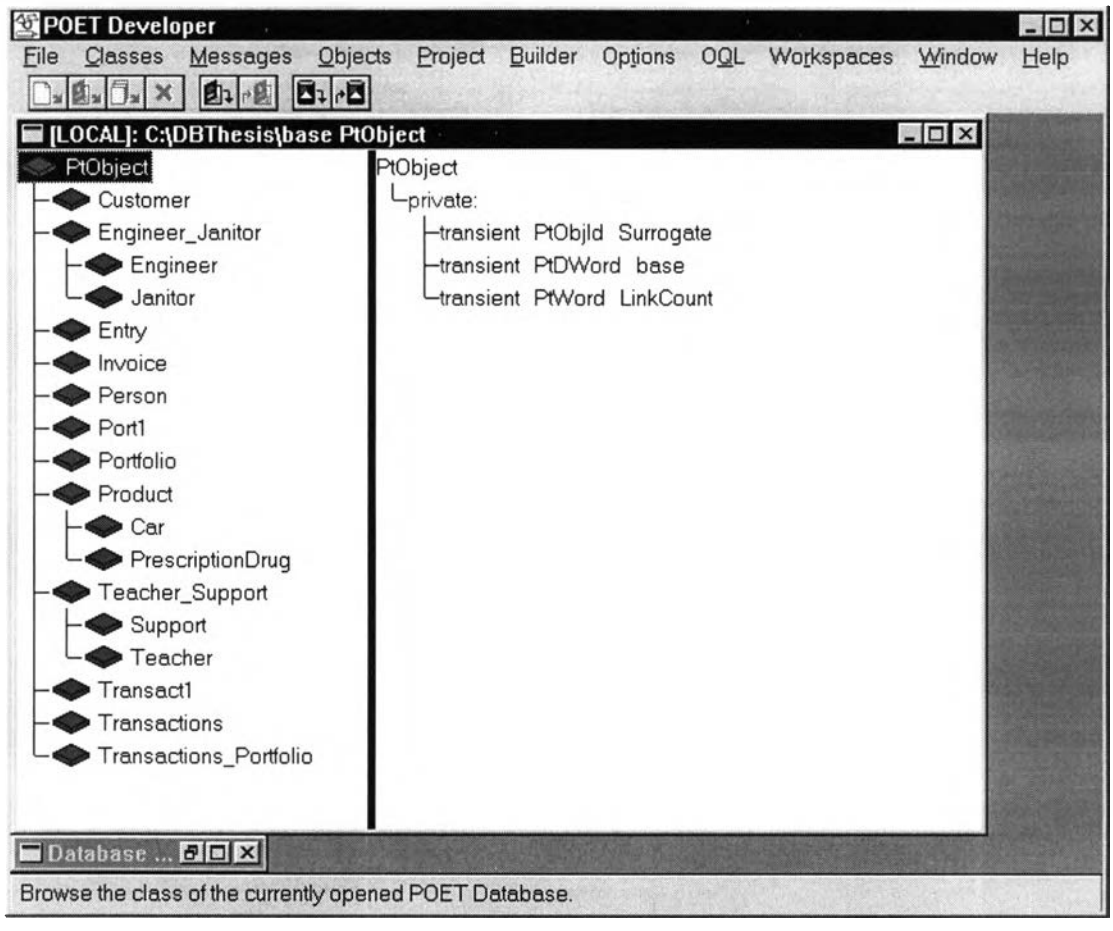
ต่อไปเป็นขั้นตอนสุดท้ายซึ่งจะมีการสร้างแฟ้มข้อความ POET C++ Class Definition และแฟ้มอื่นๆที่เกี่ยวข้องรวมทั้งการสร้างฐานข้อมูลบนระบบจัดการฐานข้อมูล POET



รูปที่ 5.9 วินโดว์สำหรับให้กำหนดโฟลด์เดอร์ใช้เก็บแฟ้ม HCD และกำหนดชื่อฐานข้อมูล

รูปที่ 5.9 วินโดวส์สำหรับให้ผู้ใช้กำหนดตำแหน่ง สำหรับเก็บแฟ้มต่างๆที่ได้ รวมทั้งชื่อฐานข้อมูลพร้อมชื่อพจนานุกรมฐานข้อมูล เมื่อเลือกปุ่มคำสั่ง Continue แล้วโปรแกรมจะทำการสร้างแฟ้ม HCD และแฟ้มอื่นๆที่เกี่ยวข้อง สุดท้ายสร้างฐานข้อมูลเชิงวัตถุบนระบบจัดการฐานข้อมูลเชิงวัตถุ POET จากแฟ้มต่างๆที่ได้เหล่านั้น

หลังการสร้างฐานข้อมูลเรียบร้อยแล้ว จะใช้โปรแกรม POET Developer ซึ่งเป็นโปรแกรมที่มาพร้อมกับฐานข้อมูลระบบจัดการฐานข้อมูล POET ในการทดสอบกรณีต่างๆ ต่อไป



รูปที่ 5.10 วินโดว์การทำงานของโปรแกรม POET Developer

5.1.1 **ทดสอบการมีลักษณะทั่วไป** (Generalization)

ทดสอบการมีลักษณะทั่วไปหรือทดสอบการสืบทอดคุณสมบัติ แบ่งเป็น 3 กรณีย่อยคือ

- กรณีตารางที่มีคอลัมน์ซ้ำกัน
- กรณีตารางหนึ่งมีคีย์หลักเป็นซับเซ็ตของคีย์หลักอีกตารางหนึ่ง
- กรณีตาราง 2 ตารางมีคีย์หลักเป็นคีย์นอกชี้ไปยังตารางอื่น

5.1.1.1 **กรณีตารางที่มีคอลัมน์ซ้ำกัน**

เมื่อแปลงเป็นฐานข้อมูลเชิงวัตถุแล้วจะต้องได้คลาสใหม่เกิดขึ้น และคลาสนี้เป็นคลาสพ่อของคลาสที่มีแอททริบิวท์ซ้ำกัน

ตารางที่ใช้ทดสอบในที่นี้ชื่อ Janitor และ Engineer

ตาราง Janitor

ชื่อคอลัมน์	ประเภทข้อมูล	ขนาด
Number	int	4
Name	char	50
Occupation_Code	char	1
Job_Shift_Code	char	1

ตาราง Engineer

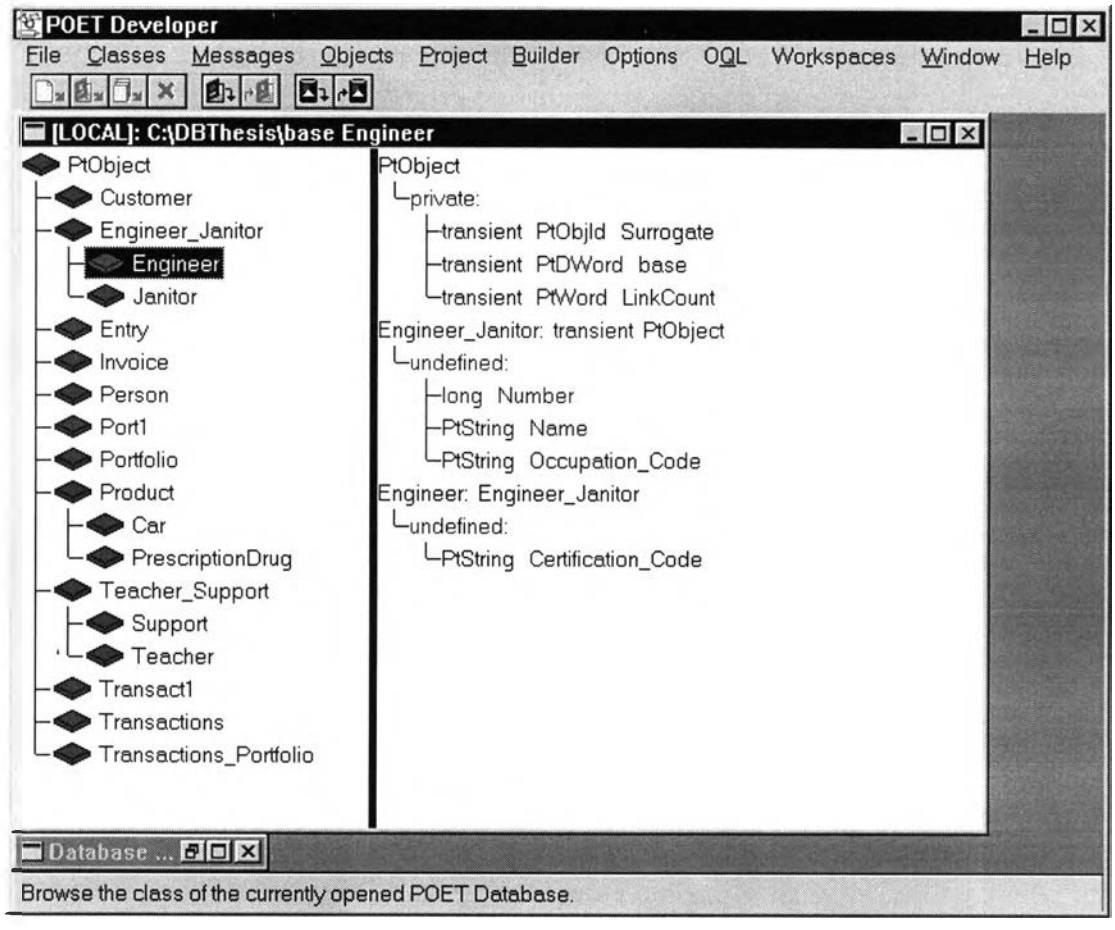
ชื่อคอลัมน์	ประเภทข้อมูล	ขนาด
Number	Int	4
Name	Char	50
Occupation_Code	Char	1
Certification_Code	Char	1

ตารางทั้งสองมีแอททริบิวท์ Number Name และ Occupation_Code ซ้ำกัน ตรวจสอบผลจากการทำงานของโปรแกรมแปลงฐานข้อมูล โดยใช้โปรแกรม POET Developer ดังแสดงในรูปที่ 5.11

ฐานข้อมูลที่ได้ คลาส Engineer_Janitor เป็นคลาสพ่อมี Number Name และ Occupation_Code เป็นแอททริบิวท์ คลาส Engineer และคลาส Janitor เป็นคลาสลูก

คลาส Engineer มีแอททริบิวท์ Certification_Code

คลาส Janitor มีแอททริบิวท์ Job_Shift_Code



รูปที่ 5.11 คลาสซึ่งมีแอททริบิวท์ซ้ำกันแสดงรายละเอียดคลาสลูกด้านขวา

รูปที่ 5.11 คลาส Engineer_Janitor เป็นคลาสพ่อมี Number Name และ Occupation_Code เป็นแอททริบิวท์ โดยที่คลาส Engineer และคลาส Janitor เป็นคลาสลูก รูปด้านขวาแสดงรายละเอียดการสืบทอดคุณสมบัติ

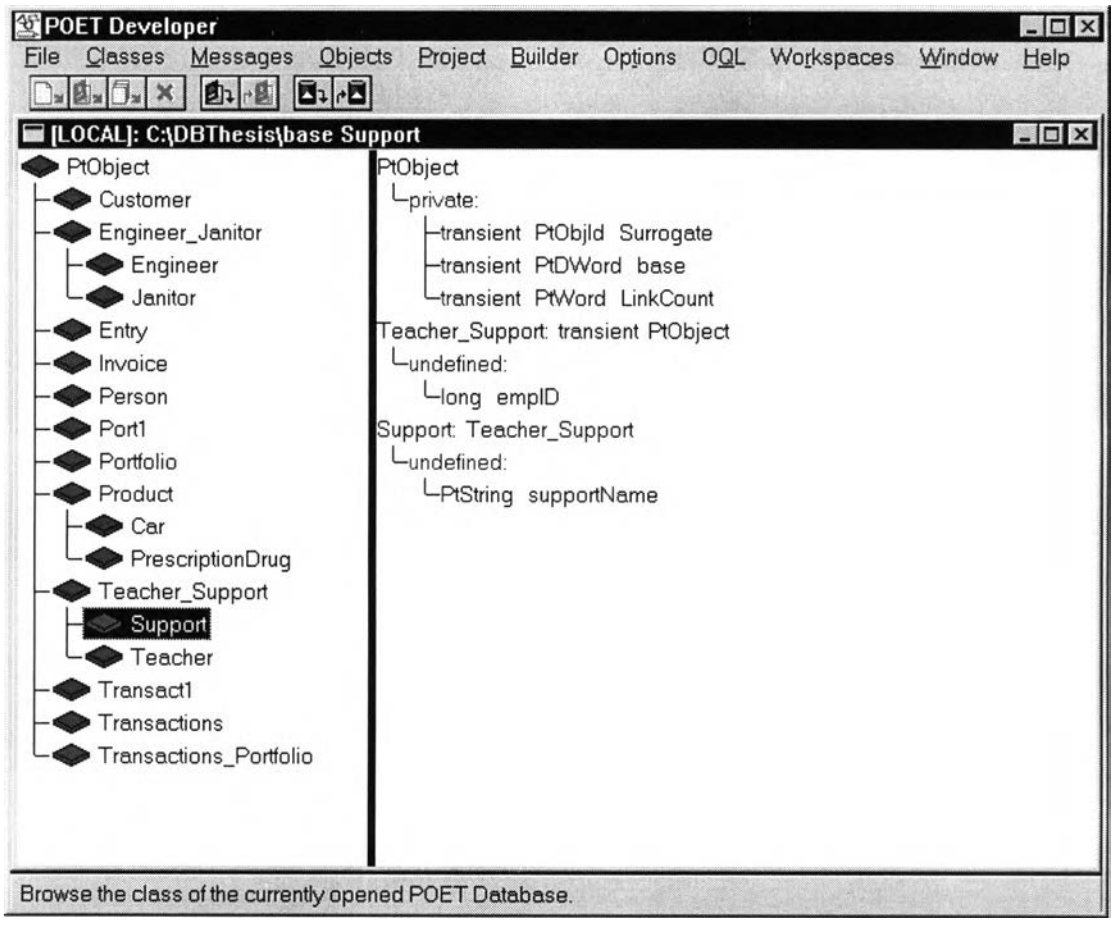
สรุปผลการทดสอบกรณีนี้ถูกต้อง

5.1.1.2 กรณีตารางหนึ่งมีคีย์หลักเป็นซับเซ็ตของคีย์หลักอีกตารางหนึ่ง

เมื่อแปลงเป็นฐานข้อมูลเชิงวัตถุแล้วจะต้องได้คลาสใหม่เกิดขึ้น และคลาสนี้เป็นคลาสพ่อ

ตารางที่ใช้ทดสอบในที่นี้ชื่อ Teacher และ Support โดยที่ตาราง Support มี empID เป็นคีย์หลักและเป็นซับเซ็ตกับคีย์หลักของ Teacher ซึ่งมี empID และ teacherID เป็นคีย์หลัก

ถ้าการแปลงถูกต้องจะต้องได้คลาสใหม่เป็นคลาสพ่อของคลาส Teacher และคลาส Support จากการตรวจสอบด้วย POET Developer ได้ผลดังแสดงในรูปที่ 5.12



รูปที่ 5.12 คลาสซึ่งมีคีย์หลักเป็นซับเซ็ตคีย์หลักของคลาสอื่น

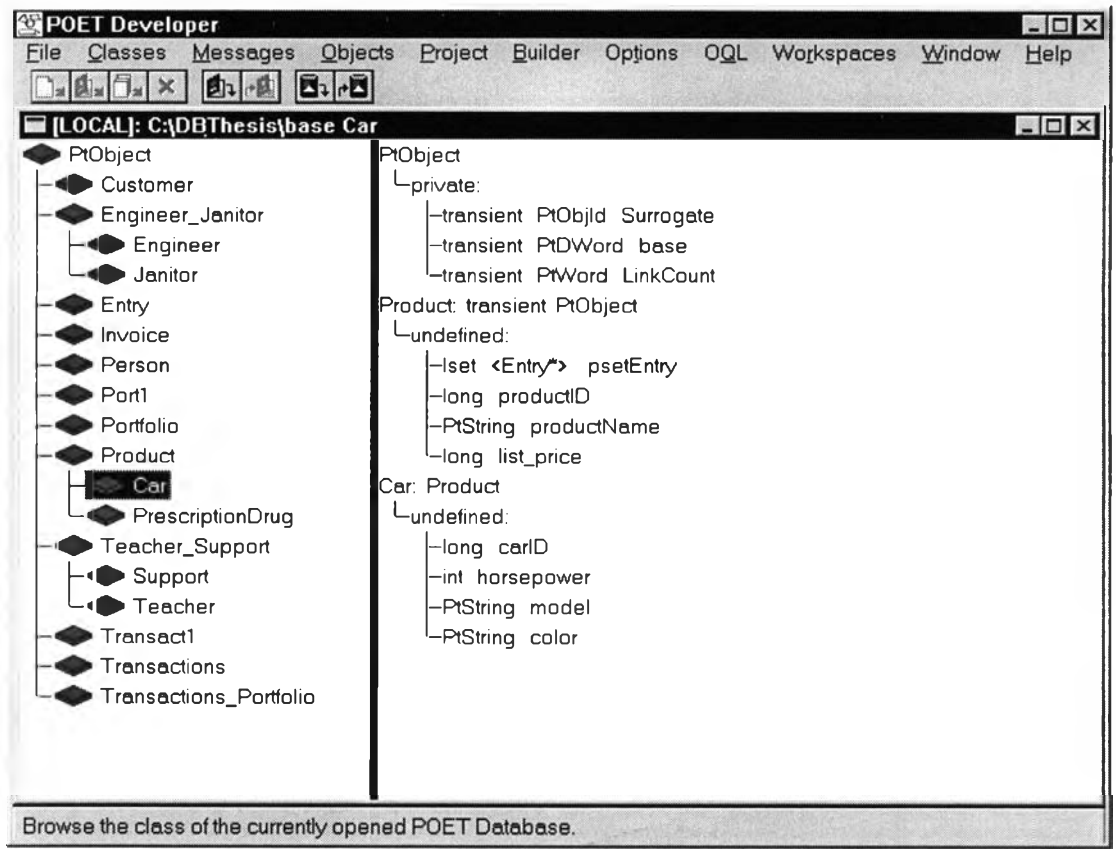
รูปที่ 5.12 ด้านขวามือแสดงรายละเอียดคลาสลูก Support ซึ่งมี Teacher_Support เป็นคลาสพ่อ และด้านซ้ายแสดงคลาส Teacher_Support มีคลาสลูก Teacher และ Support

สรุปผลกรณีตารางซึ่งมีคีย์หลักเป็นซับเซ็ตของคีย์หลักอีกตารางหนึ่ง โปรแกรมแปลงฐานข้อมูล แปลงถูกต้อง

5.1.1.3 กรณีตาราง 2 ตารางมีคีย์หลักเป็นคีย์นอกชี้ไปยังตารางอื่น

เมื่อแปลงเป็นฐานข้อมูลเชิงวัตถุแล้ว คลาสที่ถูกอ้างอิงเป็นคลาสพ่อ และคลาสผู้อ้างอิงเป็นคลาสลูก

ตารางที่ใช้ทดสอบในที่นี้ชื่อ Product Car และ PrescriptionDrug โดยที่ตาราง Car และตาราง PrescriptionDrug ต่างก็มีคีย์นอกซึ่งเป็นคีย์หลักด้วยชี้ไปยังตาราง Product



รูปที่ 5.13 คลาสผู้อ้างอิง 2 คลาสต่างก็มีคีย์นอกซึ่งเป็นคีย์หลักด้วยชี้ไปยังคลาสที่สาม

รูปที่ 5.13 ด้านขวามือแสดงรายละเอียดคลาส Car โดยแสดงว่ามีแอททริบิวท์อะไรบ้าง และยังแสดงด้วยว่ามีคลาส Product เป็นคลาสพ่อ รูปด้านซ้ายมือแสดงให้ทราบว่าคลาส Product มีคลาสลูก Car และ PrescriptionDrug

สรุปได้ว่าสำหรับกรณีตาราง 2 ตารางคีย์หลักเป็นคีย์นอกชี้ไปยังตารางอื่น โปรแกรมแปลงฐานข้อมูล แปลงถูกต้อง

5.1.2 **ทดสอบความเกี่ยวพัน** (Association)

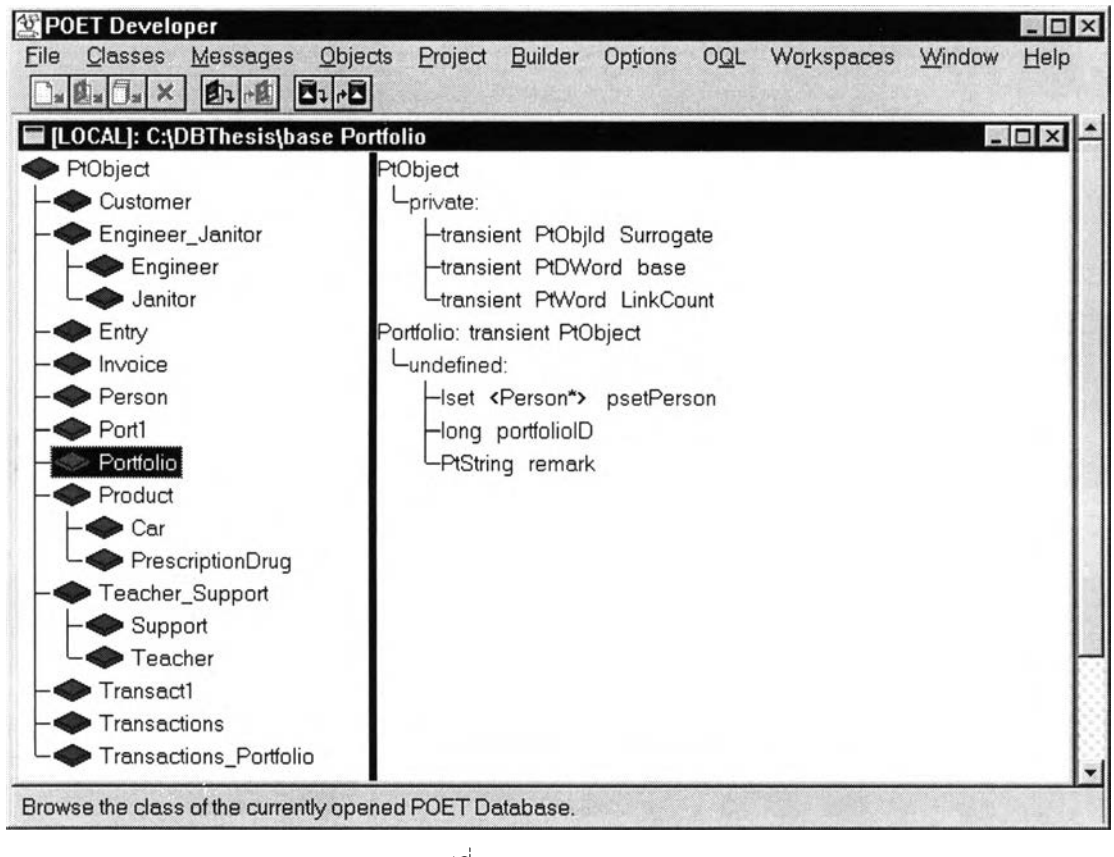
ทดสอบความเกี่ยวพัน แบ่งเป็น 4 กรณีย่อยคือ

- ตารางซึ่งมีคีย์นอก 2 ตัว โดยที่ตารางนี้มีคีย์หลักประกอบด้วยคีย์นอก 2 ตัว และตารางนี้มีคอลัมน์ที่เป็นคีย์นอกเท่านั้น
- ตารางซึ่งมีคีย์นอก 2 ตัว โดยที่ตารางนี้มีคีย์หลักเป็นคีย์นอกตัวใดตัวหนึ่ง และตารางนี้มีคอลัมน์ที่เป็นคีย์นอกเท่านั้น
- ตารางซึ่งมีคีย์นอก 1 ตัวและคีย์นอกเป็นคีย์หลักด้วย
- ตารางซึ่งคีย์นอกไม่เป็นส่วนใดส่วนหนึ่งกับคีย์หลัก

5.1.2.1 ตารางซึ่งมีคีย์นอก 2 ตัว โดยที่ตารางนี้มีคีย์หลักประกอบด้วยคีย์นอก 2 ตัว และตารางนี้มีคอลัมน์ที่เป็นคีย์นอกเท่านั้น

เมื่อแปลงเป็นฐานข้อมูลเชิงวัตถุแล้ว คลาสที่ถูกอ้างอิงจะมีความเกี่ยวพันกันแบบ "MANY-TO-MANY" โดยแสดงในรูปของแอททริบิวท์ ซึ่งเป็นเซ็ตของพอยท์เตอร์ชี้ไปยังคลาสฝั่งตรงข้าม โดยที่ตารางผู้อ้างอิงจะไม่ถูกแปลงเป็นคลาสด้วย

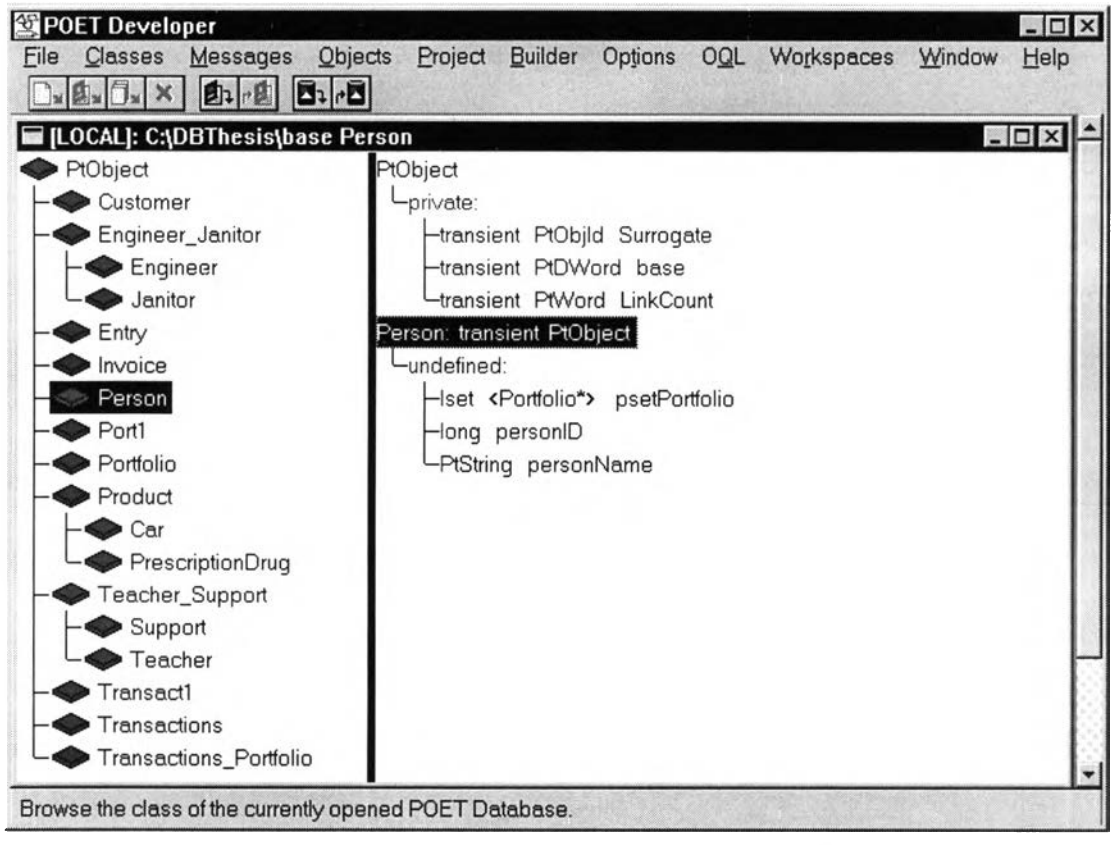
ตารางที่ใช้ทดสอบในที่นี้ชื่อ Person Portfolio และ Portfolio_AccountOwner โดยที่ตาราง Portfolio_AccountOwner มีคีย์นอก 2 ตัวชี้ไปยังตาราง Person และ Portfolio โดยที่ตาราง Portfolio_AccountOwner มีคอลัมน์ที่ประกอบเป็นคีย์นอกเท่านั้นและคีย์หลักประกอบด้วยคีย์นอกทั้ง 2 ตัว



รูปที่ 5.14 คลาส Portfolio

รูปที่ 5.14 คลาส Portfolio มีแอททริบิวท์ประกอบด้วยแอททริบิวท์ซึ่งแปลงจากคอลัมน์ในตาราง Portfolio และแอททริบิวท์ซึ่งแสดงความเกี่ยวพันกับคลาส Person ในรูปของเซ็ตของพอยท์เตอร์ชี้ไปยัง Person

lset <Person *> psetPerson



รูปที่ 5.15 คลาส Person

รูปที่ 5.15 คลาส Person มีแอททริบิวท์ประกอบด้วยแอททริบิวท์ซึ่งแปลงจากคอลัมน์ในตาราง Person และแอททริบิวท์ซึ่งแสดงความเกี่ยวพันกับคลาส Portfolio ในรูปของเซ็ตของพอยท์เตอร์ชี้ไปยัง Portfolio

lset <Portfolio *> psetPortfolio

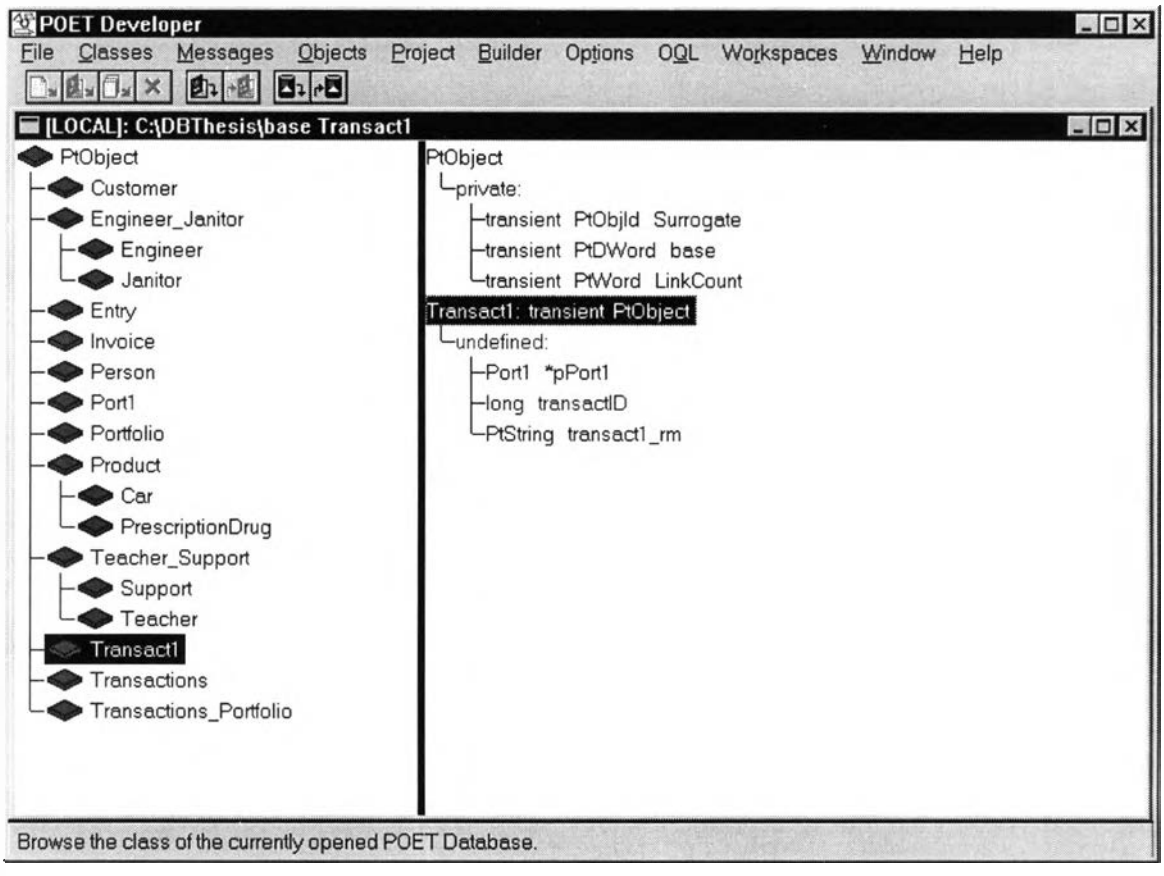
และจะเห็นว่าตาราง Portfolio_AccountOwner ไม่ได้ถูกแปลงเป็นคลาสด้วย

สรุปได้ว่าตารางซึ่งมีคีย์นอก 2 ตัว โดยที่ตารางนี้มีคีย์หลักประกอบด้วยคีย์นอก 2 ตัว และตารางนี้มีคอลัมน์ที่เป็นคีย์นอกเท่านั้น โปรแกรมแปลงฐานข้อมูล แปลงได้ถูกต้อง

5.1.2.2 ตารางซึ่งมีคีย์นอก 2 ตัว โดยที่ตารางนี้มีคีย์หลักเป็นคีย์นอกตัวใดตัวหนึ่ง และตารางนี้มีคอลัมน์ที่เป็นคีย์นอกเท่านั้น

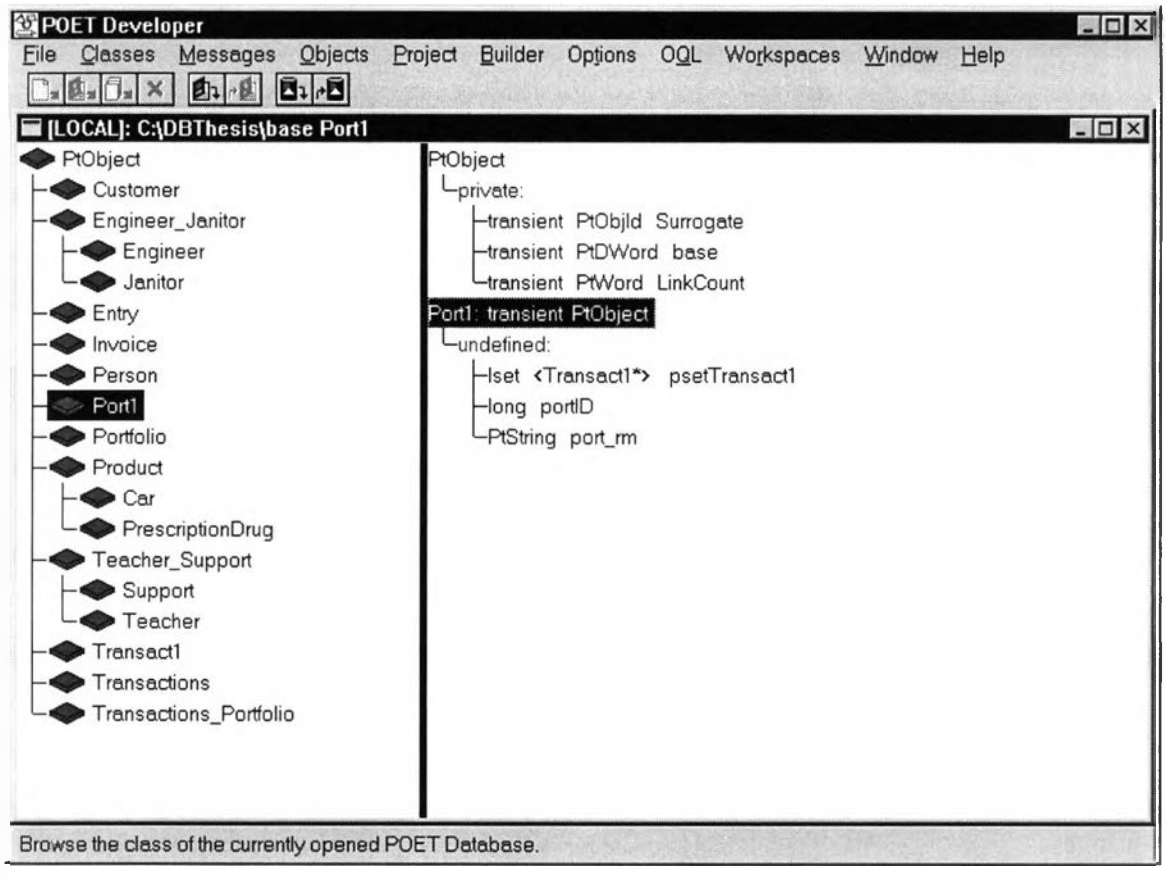
เมื่อแปลงเป็นฐานข้อมูลเชิงวัตถุแล้ว คลาสที่ถูกอ้างอิงโดยคีย์หลักจะมีความเกี่ยวพันกับคลาสที่ถูกอ้างอิงอีกตัวแบบ "MANY-TO-ONE" และตารางผู้อ้างอิงจะไม่ถูกแปลงเป็นคลาสด้วย

ตารางที่ใช้ทดสอบในที่นี้ชื่อ Transact1 Port1 และ Transact1_Port1 โดยที่ตาราง Transact1_Port1 มีคีย์นอก 2 ตัว คีย์นอกตัวที่ 1 เป็นคีย์หลักด้วยชี้ไปยังตาราง Transact1 คีย์นอกตัวที่ 2 ชี้ไปยังตาราง Port1



รูปที่ 5.16 คลาส Transact1

รูปที่ 5.16 คลาส Transact1 แปลงมาจากตาราง Transact1 ทุกแอททริบิวท์แปลงมาจากคอลัมน์ในตาราง Transact1 ยกเว้นแอททริบิวท์ pPort1 เป็นพอยท์เตอร์ชี้ไปยังคลาส Port1



รูปที่ 5.17 คลาส Port1

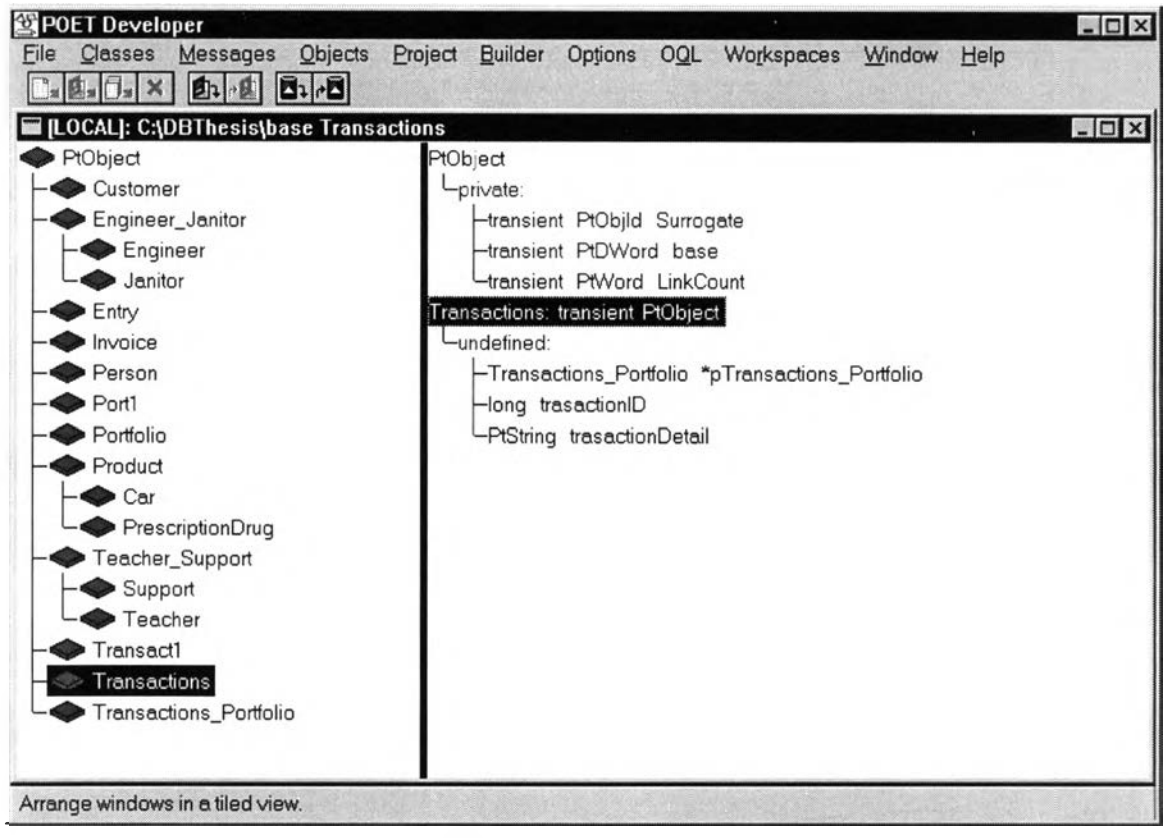
รูปที่ 5.17 คลาส Port1 แปลงมาจากตาราง Port1 ทุกแอททริบิวท์แปลงมาจากคอลัมน์ในตาราง Port1 ยกเว้นแอททริบิวท์ psetTransact1 เป็นเซ็ตของพอยท์เตอร์ชี้ไปยังคลาส Transact1

จากรูปที่ 5.16 และรูปที่ 5.17 สำหรับกรณีนี้ สรุปได้ว่าโปรแกรมแปลงฐานข้อมูล แปลงได้ถูกต้อง

5.1.2.3 ตารางซึ่งมีคีย์นอก 1 ตัวและคีย์นอกเป็นคีย์หลักด้วย

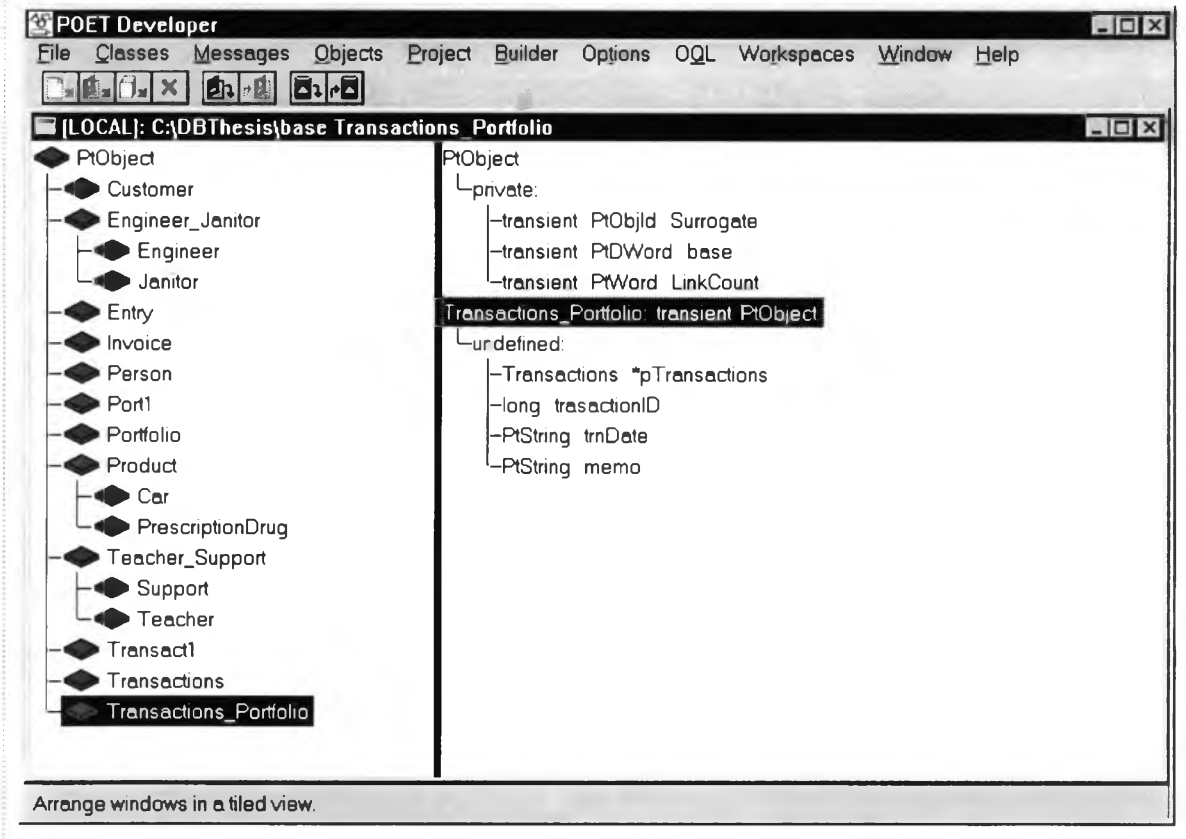
เมื่อแปลงเป็นฐานข้อมูลเชิงวัตถุแล้ว คลาสผู้อ้างอิงและคลาสผู้ถูกอ้างอิงมีความสัมพันธ์กันแบบ "ONE-TO-ONE" แสดงด้วยพอยท์เตอร์ชี้ไปยังคลาสฝั่งตรงข้าม

ตารางที่ใช้ทดสอบในกรณีนี้คือ Transactions และ Transactions_Portfolio โดยที่ตาราง Transactions_Portfolio มีคีย์นอก 1 ตัวซึ่งเป็นคีย์หลักด้วยชี้ไปยังตาราง Transactions



รูปที่ 5.18 คลาส Transactions

รูปที่ 5.18 คลาส Transactions แปลงมาจากตาราง Transactions ทุกแอททริบิวท์แปลงมาจากคอลัมน์ในตาราง Transactions ยกเว้นแอททริบิวท์ pTransactions_Portfolio เป็นพอยท์เตอร์ชี้ไปยังคลาส Transactions_Portfolio



รูปที่ 5.19 คลาส Transactions_Portfolio

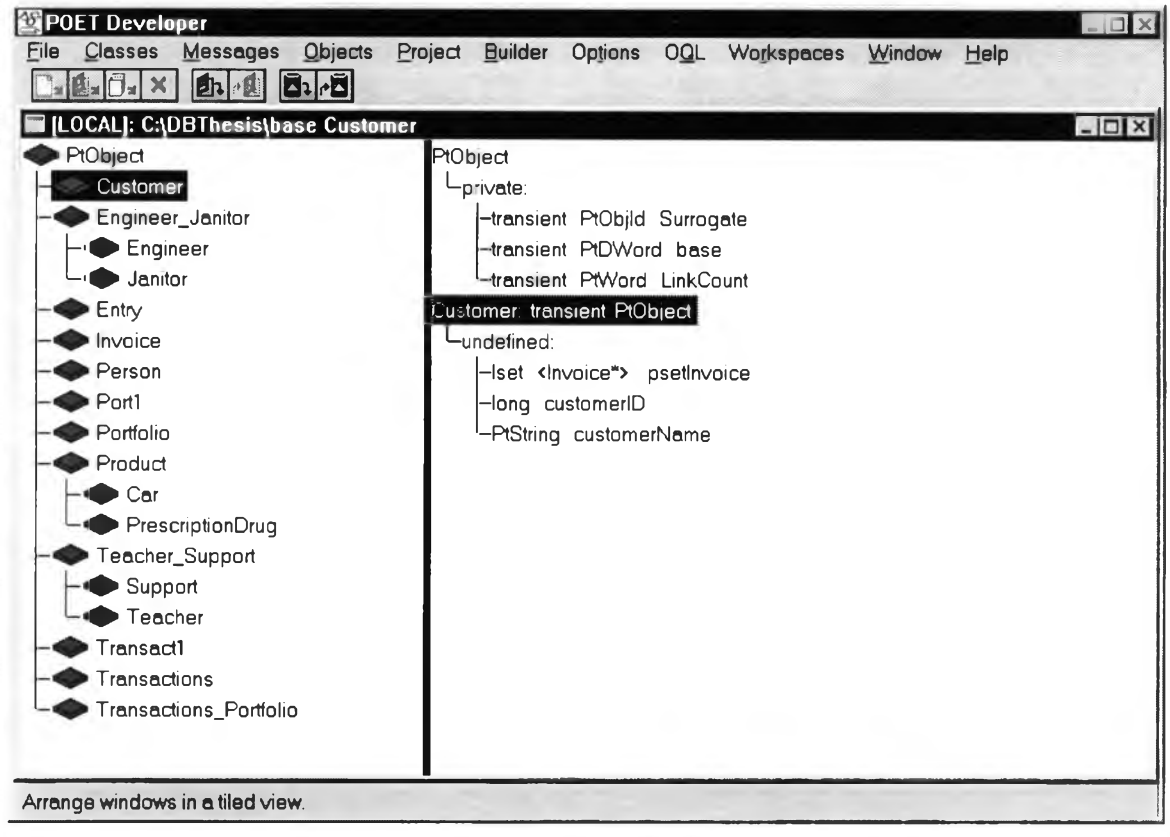
รูปที่ 5.19 คลาส Transactions_Portfolio แปลงมาจากตาราง Transactions_Portfolio ทุกแอททริบิวท์แปลงมาจากคอลัมน์ในตาราง Transactions_Portfolio ยกเว้นแอททริบิวท์ pTransactions เป็นพอยท์เตอร์ชี้ไปยังคลาส Transactions

จากรูปที่ 5.18 และรูปที่ 5.19 สำหรับกรณีนี้ สรุปได้ว่าโปรแกรมแปลงฐานข้อมูล แปลงได้ถูกต้อง

5.1.2.4 ตารางซึ่งคีย์นอกไม่เป็นส่วนใดส่วนหนึ่งกับคีย์หลัก

เมื่อแปลงเป็นฐานข้อมูลเชิงวัตถุแล้ว ความสัมพันธ์ระหว่างคลาสผู้อ้างอิงกับคลาสผู้ถูกอ้างอิงขึ้นกับผลการวิเคราะห์ข้อมูลในตารางที่เกี่ยวข้อง

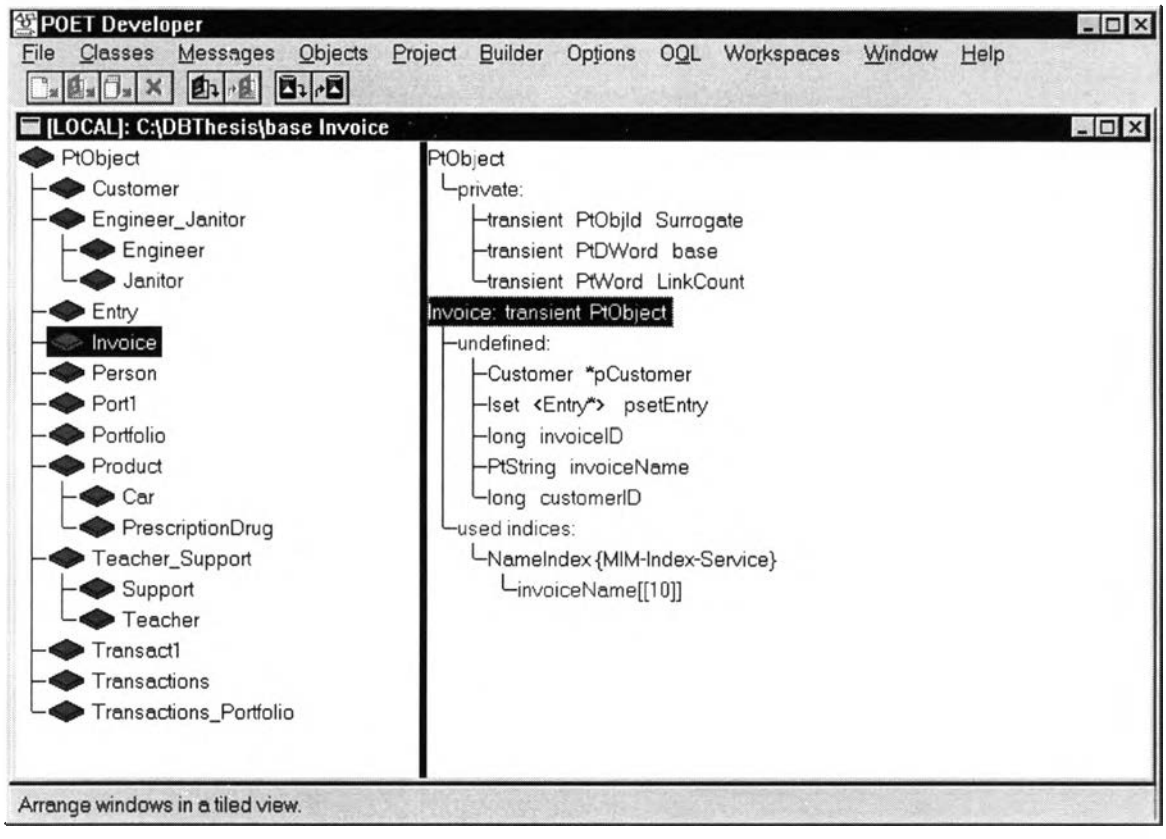
ตารางที่ใช้ทดสอบในกรณีนี้คือ Invocie Customer Entry และ Product โดยที่ตาราง Invoice มีคีย์นอกชี้ไปยังตาราง Customer และตาราง Entry อ้างอิงตาราง Product และตาราง Invoice



รูปที่ 5.20 คลาส Customer

รูปที่ 5.20 คลาส Customer แปลงมาจากตาราง Customer ทุกแอททริบิวท์แปลงมาจากคอลัมน์ในตาราง Customer ยกเว้นแอททริบิวท์ psetInvoice

Invoice มีความสัมพันธ์กับ Customer แบบ "MANY-TO-ONE" ดังนั้นจึงแสดงด้วยเซ็ตของพอยท์เตอร์ชี้ไปยังคลาส Invoice

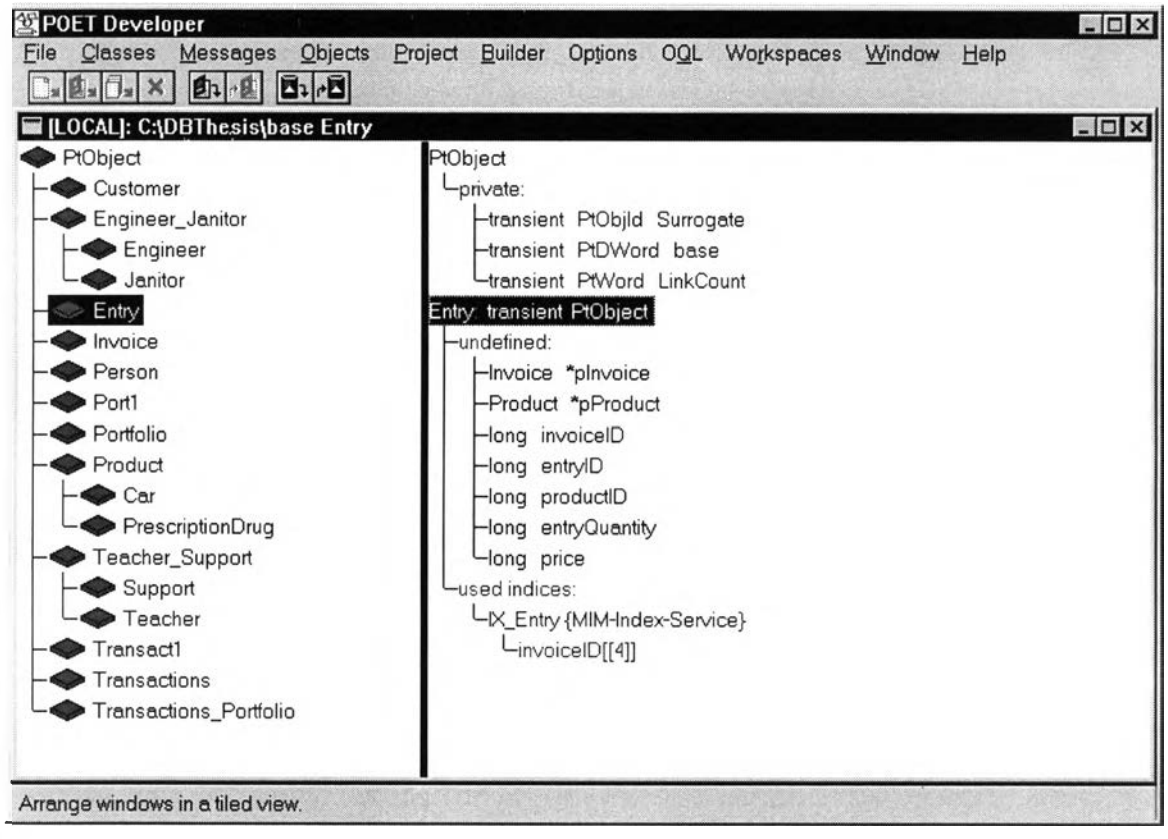


รูปที่ 5.21 คลาส Invoice

รูปที่ 5.21 คลาส Invoice แปลงมาจากตาราง Invoice ทุกแอททริบิวท์แปลงมาจากคอลัมน์ในตาราง Invoice ยกเว้นแอททริบิวท์ pCustomer และ psetEntry

Customer มีความสัมพันธ์กับ Invoice แบบ “ONE-TO-MANY” ดังนั้นจึงแสดงด้วยพอยท์เตอร์ชี้ไปยังคลาส Customer

Entry มีความสัมพันธ์กับ Invoice แบบ “MANY-TO-ONE” ดังนั้นจึงแสดงด้วยเซ็ตของพอยท์เตอร์ชี้ไปยังคลาส Entry

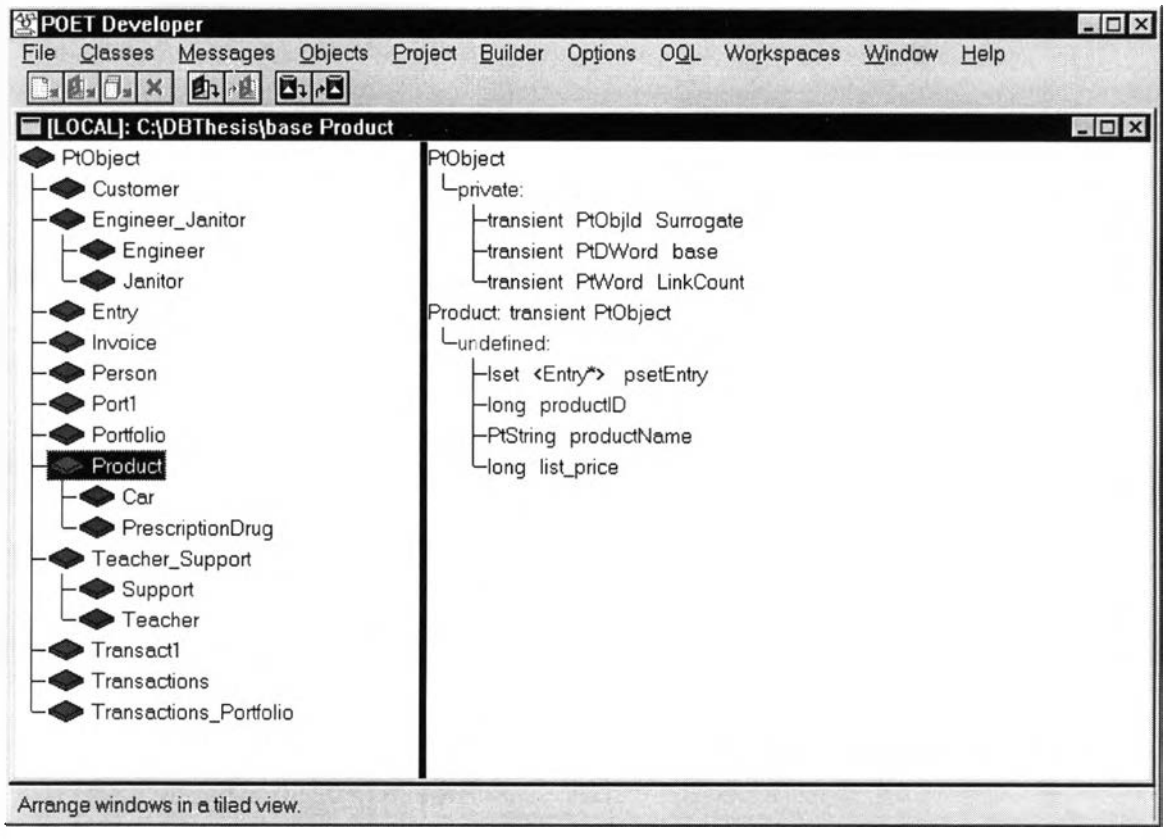


รูปที่ 5.22 คลาส Entry

รูปที่ 5.22 คลาส Entry แปลงมาจากตาราง Entry ทุกแอททริบิวท์แปลงมาจากคอลัมน์ในตาราง Entry ยกเว้นแอททริบิวท์ pInvoice และ pProduct

Invoice มีความสัมพันธ์กับ Entry แบบ "ONE-TO-MANY" ดังนั้นจึงแสดงด้วยพอยท์เตอร์ชี้ไปยังคลาส Invoice

Product มีความสัมพันธ์กับ Entry แบบ "ONE-TO-MANY" ดังนั้นจึงแสดงด้วยพอยท์เตอร์ชี้ไปยังคลาส Product



รูปที่ 5.23 คลาส Product

รูปที่ 5.23 คลาส Product แปลงมาจากตาราง Product ทุกแอททริบิวท์แปลงมาจากคอลัมน์ในตาราง Product ยกเว้นแอททริบิวท์ psetEntry

Entry มีความสัมพันธ์กับ Product แบบ "MANY-TO-ONE" ดังนั้นจึงแสดงด้วยเซ็ตของพอยท์เตอร์ชี้ไปยังคลาส Entry

จากรูปที่ 5.20 รูปที่ 5.21 รูปที่ 5.22 และรูปที่ 5.23 สรุปได้ว่าโปรแกรมแปลงฐานข้อมูลเชิงสัมพันธ์เป็นฐานข้อมูลเชิงวัตถุ แปลงได้ถูกต้อง

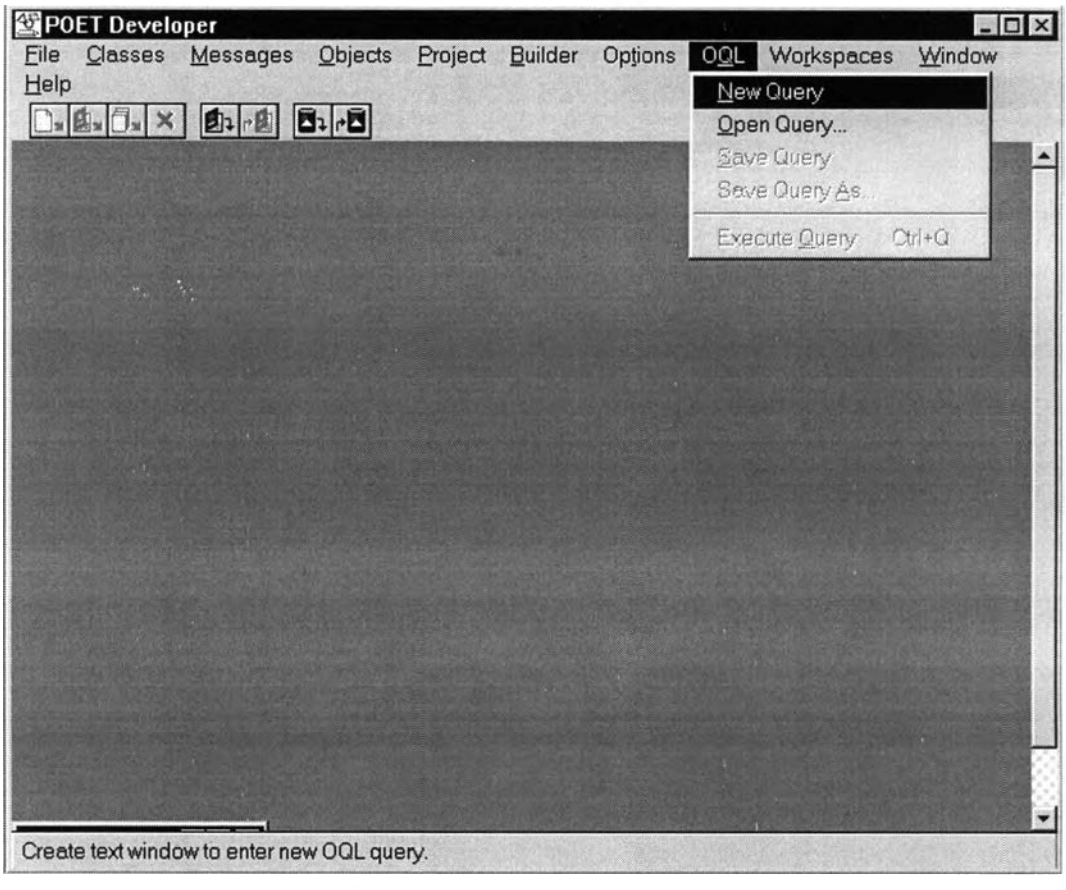
จากทั้ง 2 การสืบทอดคุณสมบัติและ กรณีความเกี่ยวพันธ์ (Association) สรุปได้ว่าโปรแกรมแปลงฐานข้อมูลเชิงสัมพันธ์เป็นฐานข้อมูลเชิงวัตถุ แปลงได้ถูกต้อง

5.2 ทดสอบการนำเข้าข้อมูลเข้าฐานข้อมูลเชิงวัตถุ

นำเข้าข้อมูลจากฐานข้อมูลบน Microsoft SQL Server ไปฐานข้อมูลบนระบบจัดการฐานข้อมูล POET โดยใช้โปรแกรม SQL Enterprise Manager แล้วทดสอบสอบถามข้อมูลจากฐานข้อมูลทั้งสองฐานข้อมูลเปรียบเทียบกัน (วิธีนำเข้าข้อมูลแสดงในภาคผนวก ค.)

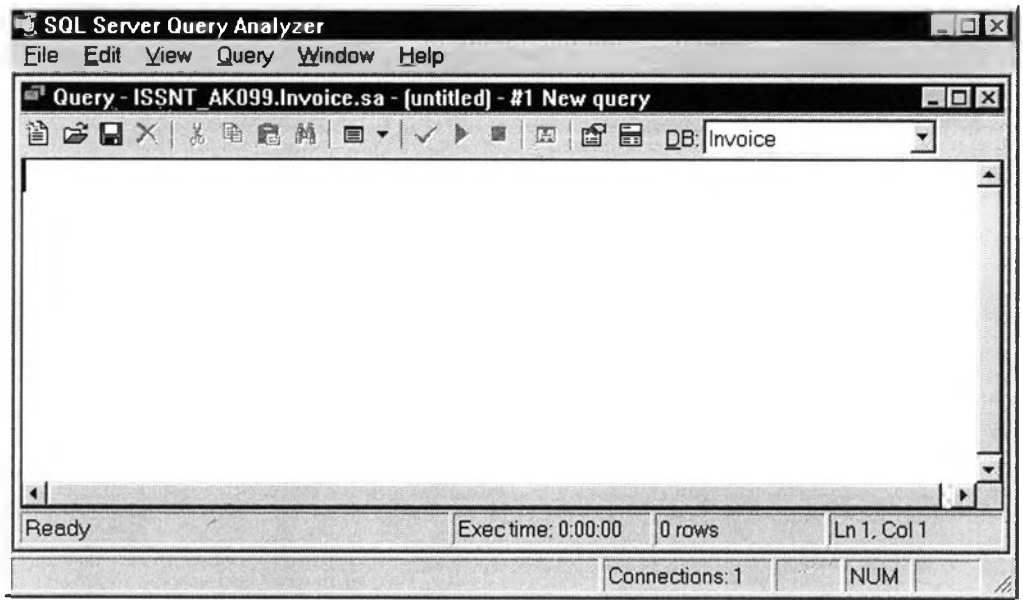
สอบถามข้อมูลบนระบบจัดการฐานข้อมูล POET ใช้ภาษา OQL (Object Query Language) เป็นเครื่องมือในการสอบถาม โดยจะใช้ภาษา OQL บนโปรแกรม POET Developer

สอบถามข้อมูลบนระบบจัดการฐานข้อมูล MS SQL Server ใช้โปรแกรม SQL Server Query Analyzer เป็นเครื่องมือในการสอบถาม โดยจะใช้ภาษา SQL (Structure Query Language) บนโปรแกรม SQL Server Query Analyzer



รูปที่ 5.24 การใช้ OQL บน POET Developer

รูปที่ 5.24 แสดงการเปิดฐานข้อมูลชื่อ base จากนั้นเปิดเมนู OQL เลือกเมนูย่อย New Query จะได้วินโดว์สำหรับการป้อนคำสั่ง OQL เพื่อสอบถามข้อมูลในฐานข้อมูล base บนระบบจัดการฐานข้อมูล POET



รูปที่ 5.25 การสอบถามข้อมูลโดยใช้คำสั่งภาษา SQL บน SQL Server Query Analyzer

รูปที่ 5.25 แสดงการเรียกใช้ SQL บนโปรแกรม SQL Server Query Analyzer

OQL หรือ Object Query Language เป็นภาษามาตรฐานสำหรับการจัดการข้อมูลบนระบบจัดการฐานข้อมูลเชิงวัตถุซึ่งกำหนดโดย ODMG (Object Database Management Group) รูปแบบของภาษาถูกกำหนดขึ้นโดยอาศัยพื้นฐานมาจากภาษา SQL

Persistent Class เมื่อผ่านโปรแกรม PTXX Schema Compiler แล้วจะมีการสร้างคลาสคลาสหนึ่งเพื่อเก็บรายการของ Persistent Class นั้นๆไว้สำหรับการทำ Query เช่นถ้ามีการสอบถามอ็อบเจ็คท์สำหรับคลาส Car แทนที่จะสอบถามจากคลาส Car โดยตรงแต่จะต้องสอบถามจาก CarExtent แทน

ตารางที่ 5.3 เปรียบเทียบการใช้คำสั่ง SQL กับ OQL

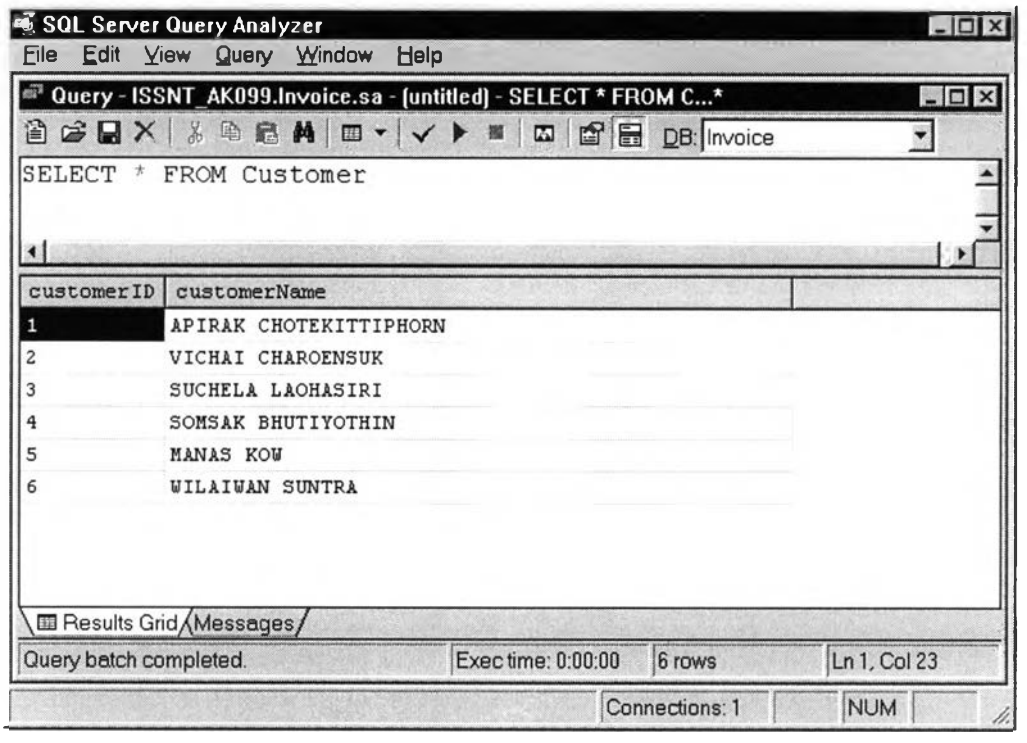
การสอบถามข้อมูลในฐานข้อมูลเชิงสัมพันธ์ (MS SQL Server)	การสอบถามข้อมูลในฐานข้อมูลเชิงวัตถุ (POET)
SELECT * From Product	SELECT * FROM ProductExtent
SELECT Product.name FROM Product	SELECT Product.name FROM ProductExtent
SELECT * FROM Product P ORDER BY P.productID ASC	SELECT * FROM ProductExtent AS P ORDER BY P.productID ASC
SELECT * FROM Product P WHERE p.list_price > 1000000 ORDER BY P.productID ASC	SELECT * FROM ProductExtent AS P WHERE P.list_price > 1000000 ORDER BY P.productID ASC

ตารางที่ 5.1 เปรียบเทียบคำสั่ง SQL กับคำสั่ง OQL

การตรวจสอบดูว่าการแม็บจากฐานข้อมูลเชิงสัมพันธ์เป็นฐานข้อมูลเชิงวัตถุ ถูกต้องหรือไม่ ทดสอบโดยการใช้คำสั่ง Select เพื่อดูว่าจำนวนข้อมูลและรายละเอียดข้อมูลในแต่ละตารางเหล่านั้น ถูกแปลงเป็นอ็อบเจคท์และรายละเอียดในอ็อบเจคท์ได้ถูกต้องครบถ้วนโดยที่ไม่มีข้อมูลสูญหายไป

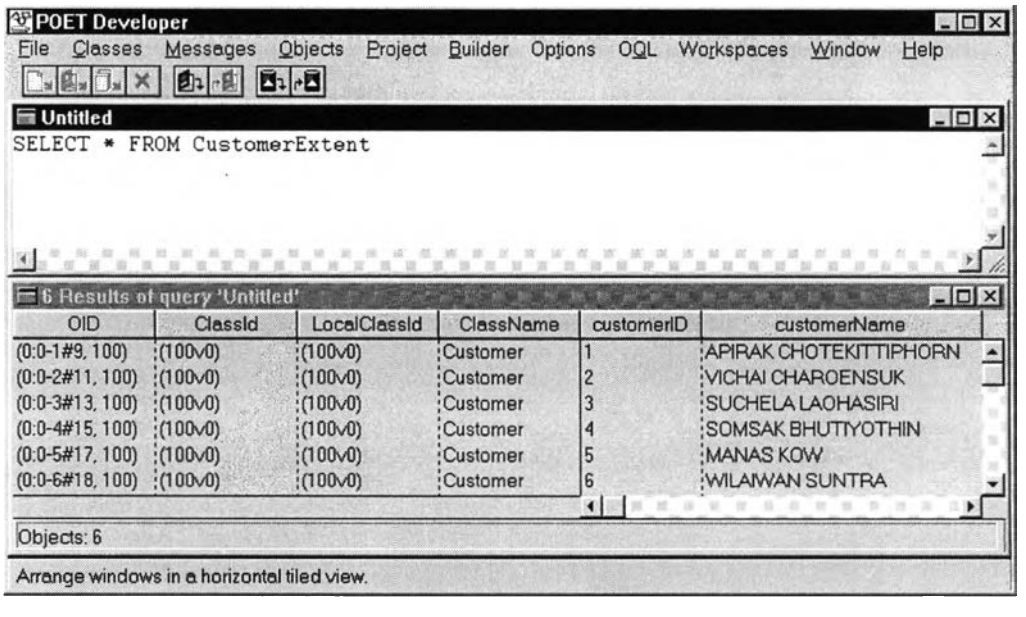
จากรูปที่ 5.1 และรูปที่ 5.10 ตารางต่างๆในฐานข้อมูล Invoice บนระบบจัดการฐานข้อมูล MS SQL Server ถูกแม็บเป็นคลาสต่างๆในฐานข้อมูล base บนระบบจัดการฐานข้อมูล POET

5.2.1 เปรียบเทียบข้อมูลในตาราง Customer กับอ็อบเจคท์สำหรับคลาส Customer



รูปที่ 5.26 รายการข้อมูลทั้งหมดในตาราง Customer

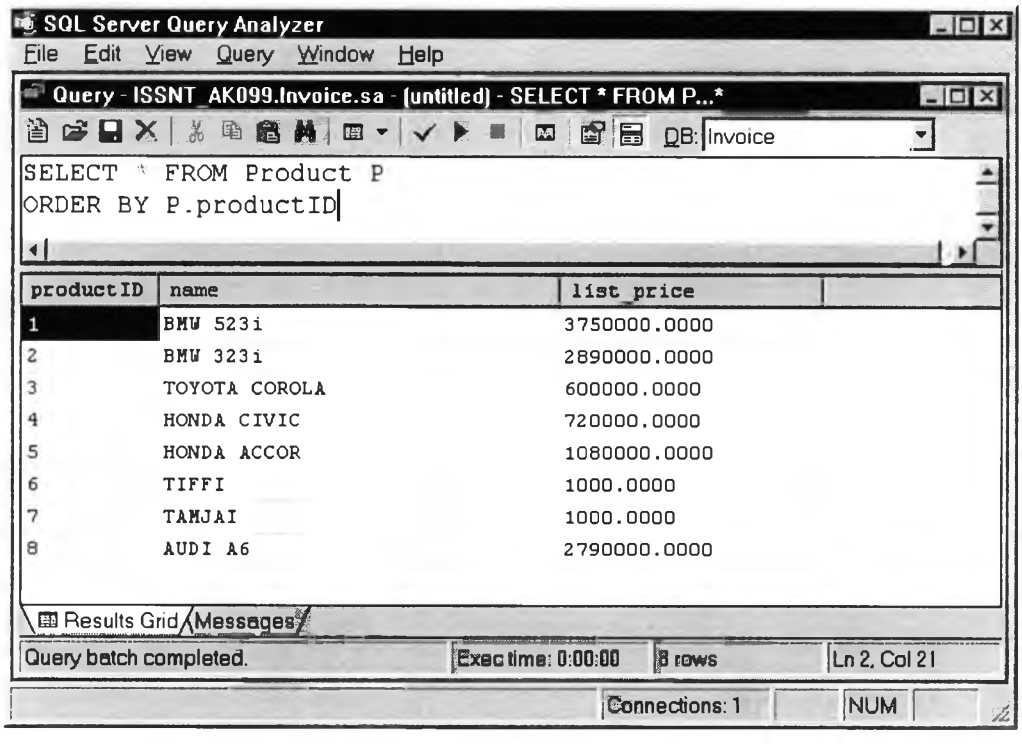
รูปที่ 5.26 โดยใช้คำสั่ง SELECT * FROM Customer ได้จำนวนข้อมูลทั้งหมด 6 รายการ



รูปที่ 5.27 อ็อบเจคท์ทั้งหมดสำหรับคลาส Customer

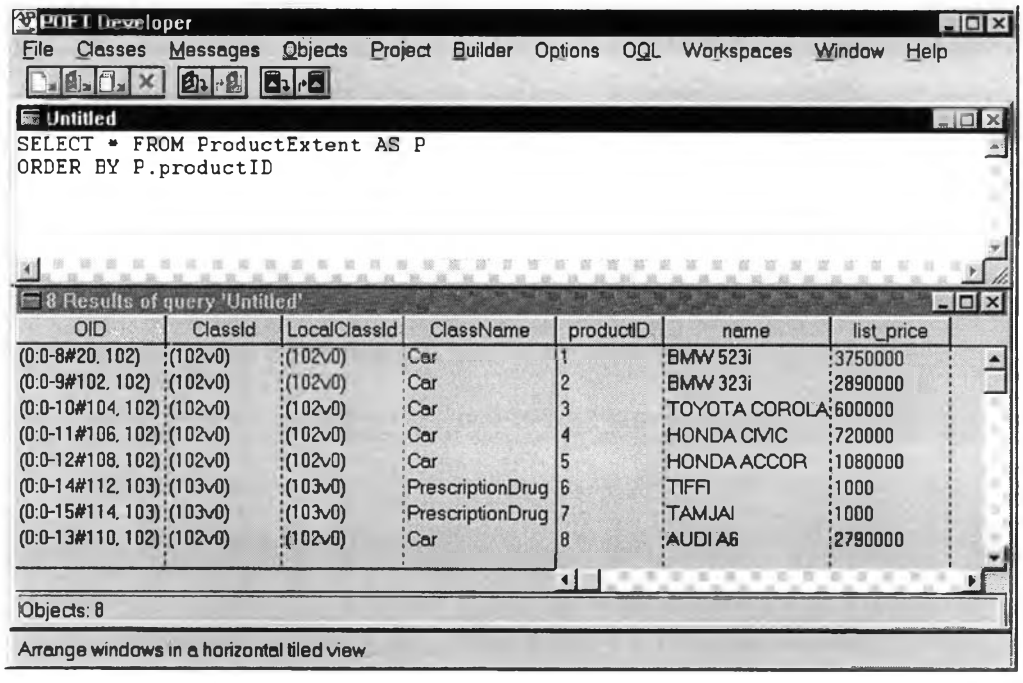
จากรูปที่ 5.26 และรูปที่ 5.27 สรุปได้ว่าการแม็บจากตารางและข้อมูลในตาราง Customer เป็นคลาสและอ็อบเจคท์สำหรับคลาส Customer ถูกต้อง

5.2 2 เปรียบเทียบข้อมูลในตาราง Product กับอ็อบเจคท์สำหรับคลาส Product



รูปที่ 5.28 ข้อมูลทั้งหมดสำหรับตาราง Product

รูปที่ 5.28 โดยใช้คำสั่ง SELECT * FROM Product P ORDER BY P.productID ได้จำนวนข้อมูลทั้งหมด 8 รายการ

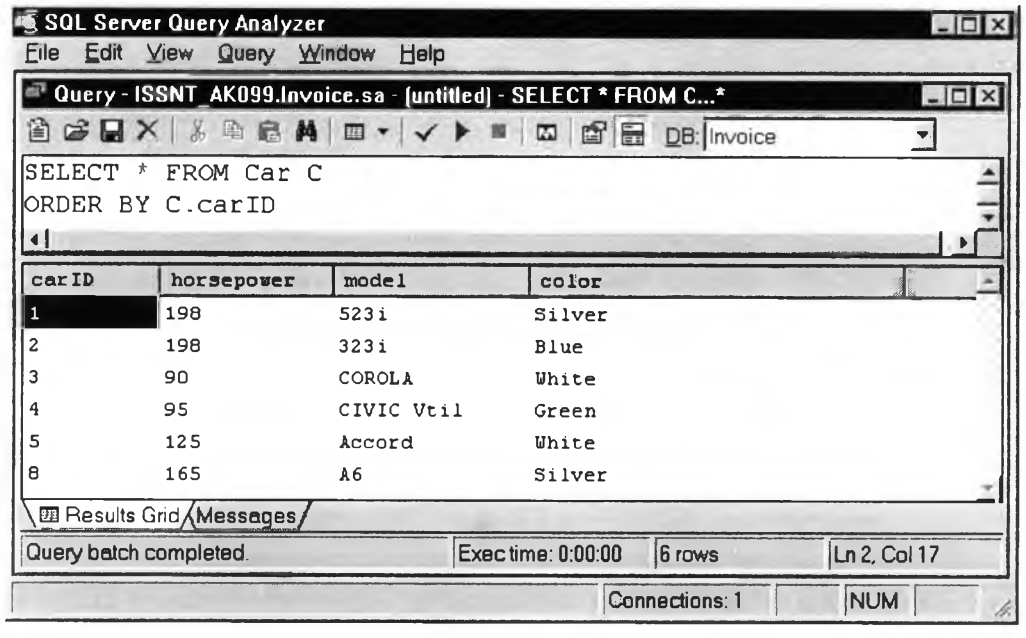


รูปที่ 5.29 อ็อบเจ็คทั้งหมดสำหรับคลาส Product

จากรูปที่ 5.29 ใช้คำสั่ง SELECT * FROM ProductExtent AS P ORDER BY P.productID ได้จำนวนข้อมูลทั้งหมด 8 รายการเช่นกัน

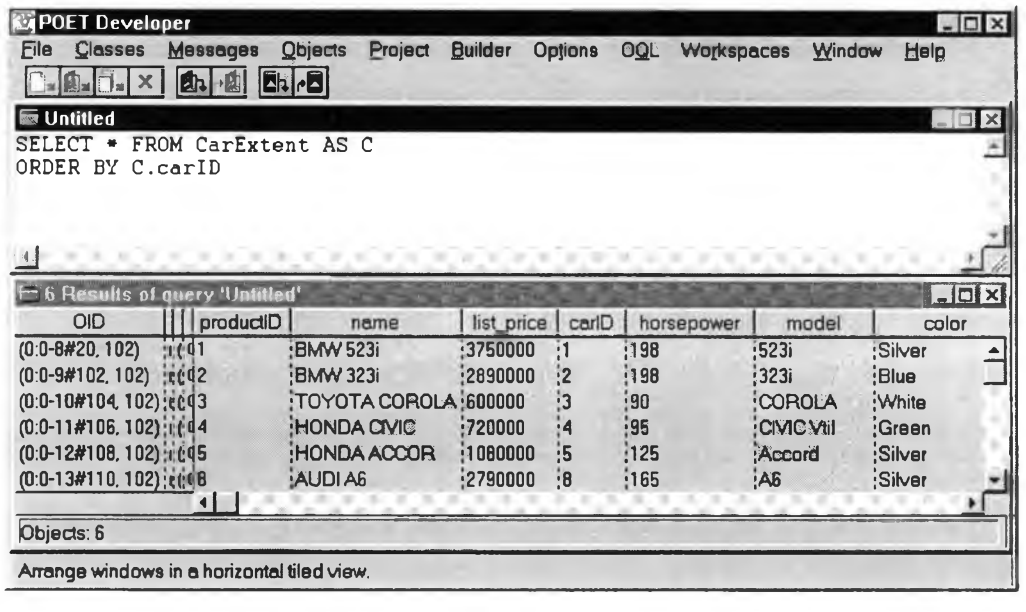
จากรูปที่ 5.28 และรูปที่ 5.29 สรุปได้ว่าการแม็บจากตารางและข้อมูลในตาราง Product เป็นคลาสและอ็อบเจคท์สำหรับคลาส Product ถูกต้อง

5.2.3 เปรียบเทียบข้อมูลในตาราง Car กับอ็อบเจคท์สำหรับคลาส Car



รูปที่ 5.30 ข้อมูลทั้งหมดสำหรับตาราง Car

รูปที่ 5.30 โดยใช้คำสั่ง SELECT * FROM Car C ORDER BY C.carID ได้จำนวนข้อมูลทั้งหมด 6 รายการ



รูปที่ 5.31 อ็อบเจคท์ทั้งหมดสำหรับคลาส Car

รูปที่ 5.31 ใช้คำสั่ง SELECT * FROM CarExtent C ORDER BY C.carID ได้จำนวนข้อมูลทั้งหมด 6 รายการเช่นกัน

อ็อบเจคท์ต่างๆในคลาส Car แสดงรายละเอียดแอททริบิวท์ซึ่งสืบทอดคุณสมบัติมาจากคลาส Product ด้วย

จากรูปที่ 5.30 และรูปที่ 5.31 สรุปได้ว่าการแม็บจากตารางและข้อมูลในตาราง Car เป็นคลาสและอ็อบเจคท์สำหรับคลาส Car ถูกต้อง