

# บทที่ 3

## การออกแบบฐานข้อมูล

วิชา การออกแบบและพัฒนาซอฟต์แวร์  
อ เพียรทิพย์ ศรีสุธรรม

# เนื้อหา

- ❖ การออกแบบฐานข้อมูลระดับตรรกะ (แปลง ER-เป็นแบบจำลองฐานข้อมูลเชิงสัมพันธ์)
- ❖ กระบวนการปรับบรรทัดฐาน (The Normalization Process)
- ❖ การออกแบบฐานข้อมูลระดับกายภาพ (Data Dictionary)

# การออกแบบฐานข้อมูลระดับตรรกะ

## (Logical Database Design)

(การแปลง แบบจำลอง E-R เป็น แบบจำลองฐานข้อมูลเชิงสัมพันธ์)

ER-Model



แบบจำลองฐานข้อมูลเชิงสัมพันธ์



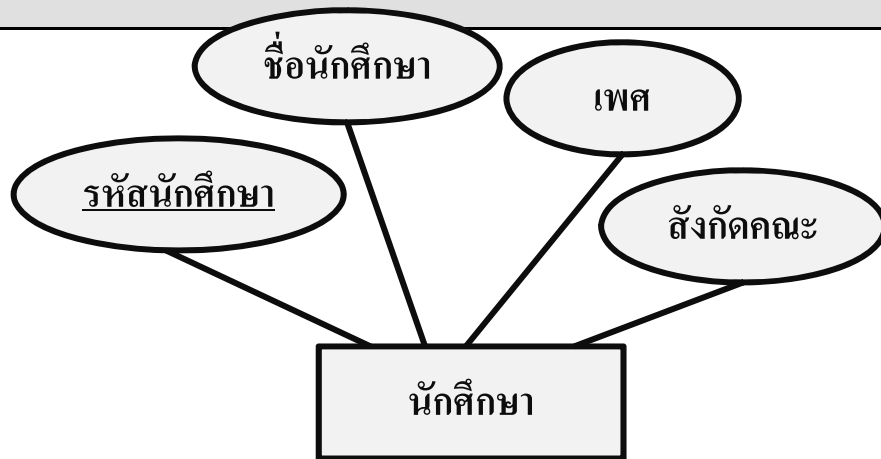
# การแปลงรูป ER-Diagram เป็น Relation

- ❑ 1.การแปลงเอนทิตีที่ปกติและแอททริบิวต์ของเอนทิตีที่ปกติ
- ❑ 2.การแปลงความสัมพันธ์ระหว่างเอนทิตี

# แอททริบิวต์แบบปกติ (Simple or Atomic Attribute)

- ❖ เอนติตีปกติแต่ละเอนติตีในแผนภาพ E-R จะถูกแปลงเป็นรีเลชัน
- ❖ ชื่อของรีเลชันจะเหมือนกับชื่อเอนติตี
- ❖ แต่ละแอททริบิวต์จะแปลงเป็นคอลัมน์ของรีเลชัน
- ❖ แอททริบิวต์ที่เป็นตัวชี้เฉพาะ (ที่ขีดเส้นใต้) จะถูกแปลงเป็นคีย์หลักในรีเลชัน

องค์ประกอบในแผนภาพ E-R



องค์ประกอบในโมเดลฐานข้อมูลเชิงสัมพันธ์

## นักศึกษา

<u>รหัส นักศึกษา</u>	ชื่อนักศึกษา	เพศ	สังกัด คณะ
--------------------------	--------------	-----	---------------

# การแปลงความสัมพันธ์(Relationship) เป็น Relation (ตาราง)

## one to one

ไม่มีการสร้าง relation ใหม่ แต่จะเป็นการปรับปรุงคูรีเลชันที่มีความสัมพันธ์กันโดยการ **เพิ่มเติมฟิลด์ที่ทำหน้าที่เป็นคีย์นอก**

## one to many

ไม่มีการสร้าง relation ใหม่ แต่จะเป็นการปรับปรุงคูรีเลชันที่มีความสัมพันธ์กันโดยการ **เพิ่มเติมฟิลด์ที่ทำหน้าที่เป็นคีย์นอก**

## many to many

**จะต้องสร้าง Relation** อันใหม่ในกรณีที่เป็น many to many

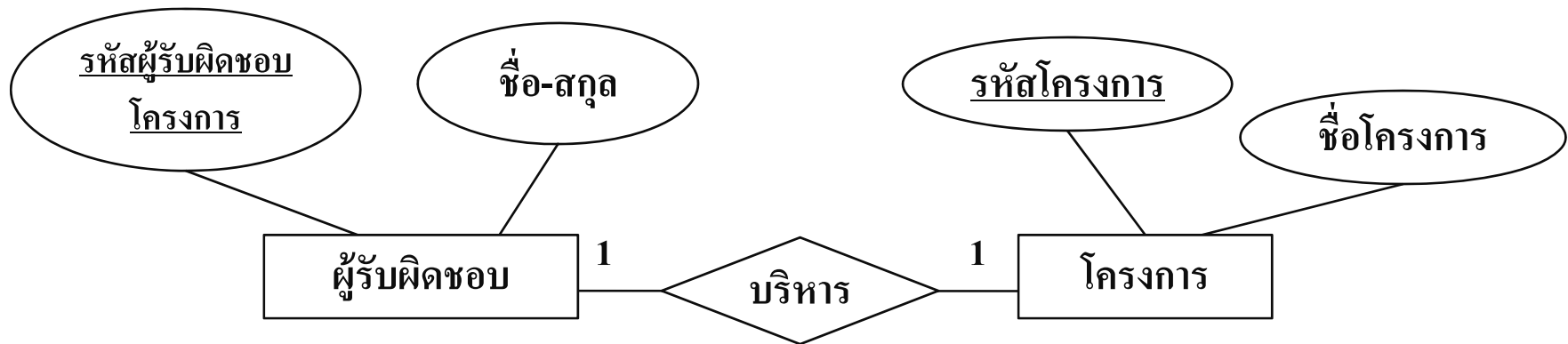
## การแปลงความสัมพันธ์(Relationship) เป็น Relation (ตาราง)

การแปลงความสัมพันธ์ระหว่าง 2 เ็นติติ (แบบเงื่อนไขของเวลาเข้ามาเกี่ยวข้อง)

ความสัมพันธ์ทั้ง 3 ชนิด (1:1, 1:M, M:N) เมื่อแปลงแล้วจะมีการสร้าง relation ใหม่ 1 รีเลชัน เกิดจากความสัมพันธ์ ที่มีเงื่อนไขเวลาเข้ามาเกี่ยวข้อง

# การแปลง one-to-one Relationship เป็น Relation (ตาราง)

- แปลง Entity ที่สัมพันธ์กันเป็น Relation
- เอา Primary Key ของ Entity ฝั่งใดฝั่งหนึ่งเป็น Foreign Key ของ Entity ฝั่งตรงกันข้าม



คีย์นอก



ผู้รับผิดชอบ (รหัสผู้รับผิดชอบโครงการ, ชื่อ-สกุล)  
โครงการ(รหัสโครงการ,ชื่อ, รหัสผู้รับผิดชอบโครงการ)

หรือ

ผู้รับผิดชอบ(รหัสผู้รับผิดชอบโครงการ, ชื่อ-สกุล,รหัสโครงการ)  
โครงการ(รหัสโครงการ,ชื่อ)



คีย์นอก



# วิธีที่ 1

## การแปลง one-to-one Relationship เป็น Relation (ตาราง)

ผู้รับผิดชอบ

รหัส ผู้รับผิดชอบ	ชื่อ-สกุล	รหัสโครงการ
01	Chanchai	1
02	Kittisak	2
03	Wittaya	3

โครงการ

รหัสโครงการ	ชื่อ
1	โครงการสร้างเขื่อน
2	โครงการขุดลอกคูคลอง
3	โครงการสร้างฝายทดน้ำ

ผู้รับผิดชอบ

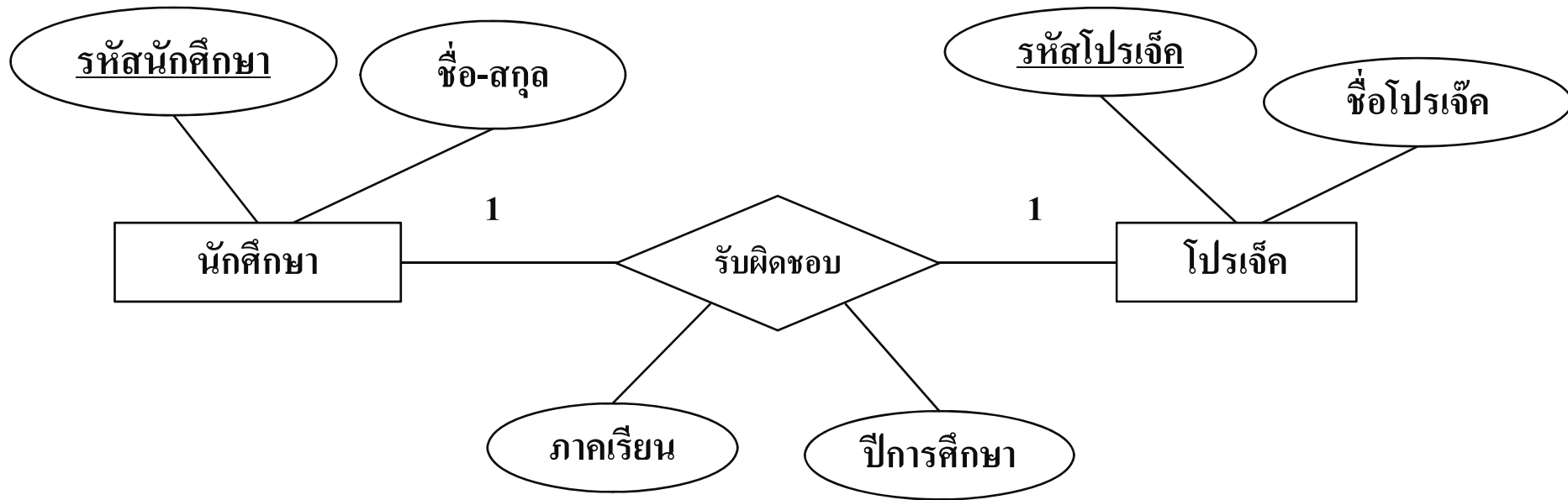
รหัส ผู้รับผิดชอบ	ชื่อ-สกุล
01	Chanchai
02	Kittisak
03	Wittaya

โครงการ

รหัส โครงการ	ชื่อ	รหัส ผู้รับผิดชอบ
1	โครงการสร้างเขื่อน	01
2	โครงการขุดลอกคูคลอง	02
3	โครงการสร้างฝายทดน้ำ	03

# วิธีที่ 2

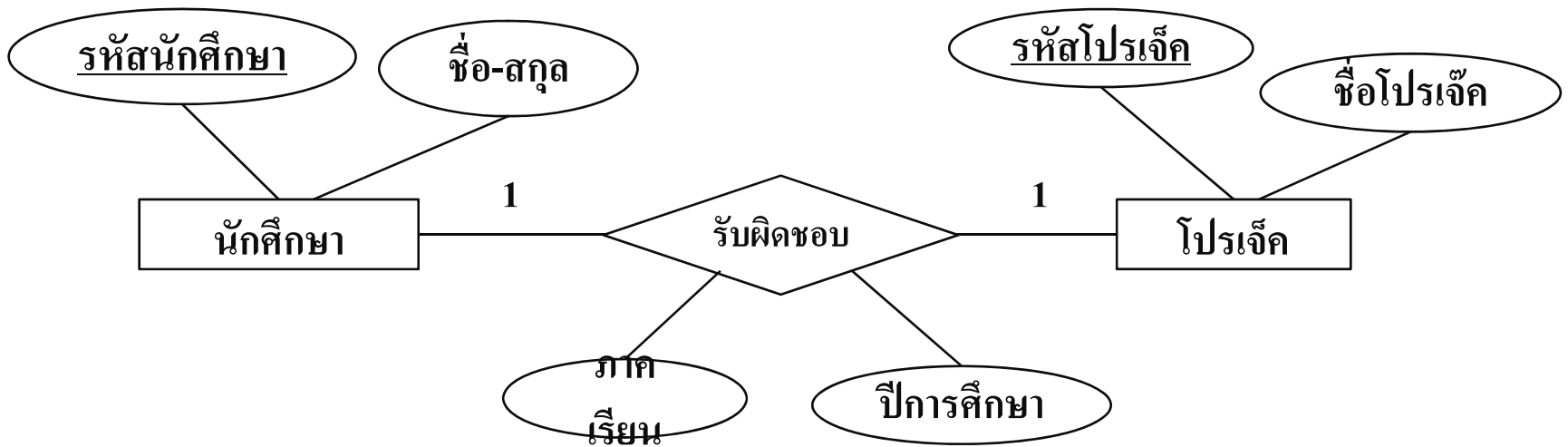
# การแปลง one-to-one Relationship เป็น Relation (ตาราง) ที่มีเวลาเข้ามาเกี่ยวข้อง



- ✓ จากตัวอย่าง เป็น ER แสดงความสัมพันธ์ ระหว่างนักศึกษากับโปรเจ็คที่รับผิดชอบ
- ✓ เพื่อให้ตรวจสอบได้ว่า นักศึกษาคนไหน ทำโปรเจ็คอะไรเมื่อไหร่ ดังนั้น ต้องระบุด้วยว่า ทำโปรเจ็คภาคเรียนใด ปีการศึกษาใด

# การแปลง one-to-one Relationship เป็น Relation (ตาราง) ที่มีเวลาเข้ามาเกี่ยวข้อง

- ✓ การแปลงความสัมพันธ์ แบบ 1:1 ที่มีเงื่อนไขเรื่องเวลาเข้ามาด้วย จะต้องแปลงเป็นรีเลชันใหม่เกิดขึ้น ในตัวอย่างคือ รีเลชัน การรับผิดชอบโปรเจ็ค
- ✓ จากตัวอย่างจะได้จะได้ทั้งหมด 3 รีเลชันคือ
  - นักศึกษา
  - โปรเจ็ค
  - **การรับผิดชอบโปรเจ็ค** (แปลงมาจาก 1:1 ที่มีเงื่อนไขเวลาเข้ามาเกี่ยวข้องด้วย)



แอททริบิวต์ที่จะเป็น คีย์หลัก (Primary Key) ของรีเลชัน **การรับผิดชอบโปรเจกต์** คือ

### ทางเลือกที่ 1

ใช้ คีย์หลักของ รีเลชัน นักศึกษา และ คีย์หลักของรีเลชันโปรเจกต์ ผลที่ได้คือ  
การรับผิดชอบโปรเจกต์(**รหัสนักศึกษา,รหัสโปรเจกต์**,ภาคเรียน,ปีการศึกษา)

### ทางเลือกที่ 2

**สร้างคีย์หลักขึ้นมาใหม่** แต่นำคีย์หลักของ รีเลชัน นักศึกษา และ คีย์หลักของรีเลชันโปรเจกต์ มาเป็นคีย์นอก

การรับผิดชอบโปรเจกต์(**เลขที่โปรเจกต์**,**รหัสนักศึกษา,รหัสโปรเจกต์**,ภาคเรียน,ปีการศึกษา)

## นักศึกษา

รหัสนักศึกษา	ชื่อ สกุล
49001	นายปราโมทย์ สีนั่นคง
49002	นายคงเดช มีถาวร
49003	นายปรีชา กิจตรง

## โปรเจ็ค

รหัสโปรเจ็ค	ชื่อโปรเจ็ค
P01	ระบบหอพักออนไลน์
P02	ระบบขายออนไลน์
P03	ระบบประวัติบุคลากร
P04	ระบบเช่ารถออนไลน์

การรับผิดชอบโปรเจ็ค

คีย์หลักของตาราง และ เป็นคีย์นอก ด้วย

รหัสนักศึกษา	รหัสโปรเจ็ค	ภาคเรียน	ปีการศึกษา
49001	P01	2	2551
49002	P02	2	2552
49003	P03	2	2553
49001	P04	1	2552

นักศึกษา

รหัสนักศึกษา	ชื่อ สกุล
49001	นายปราโมทย์ สิ้นมันคง
49002	นายคงเดช มีถาวร
49003	นายปรีชา กิจตรง

โปรเจ็ค

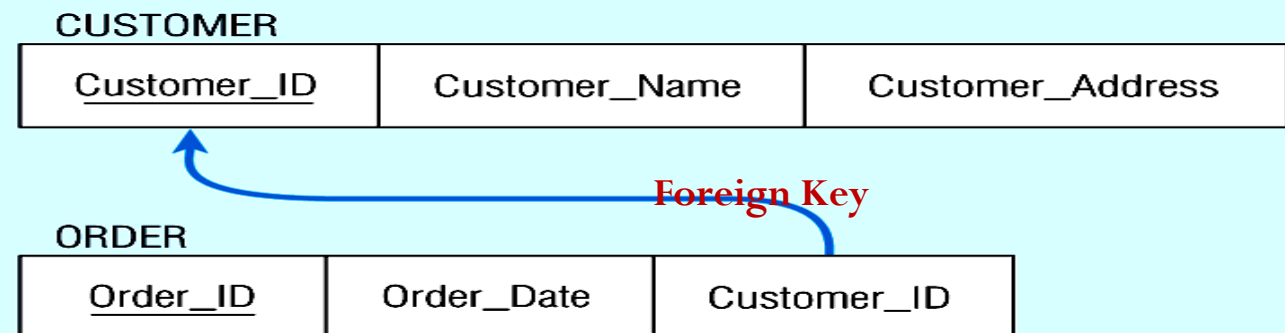
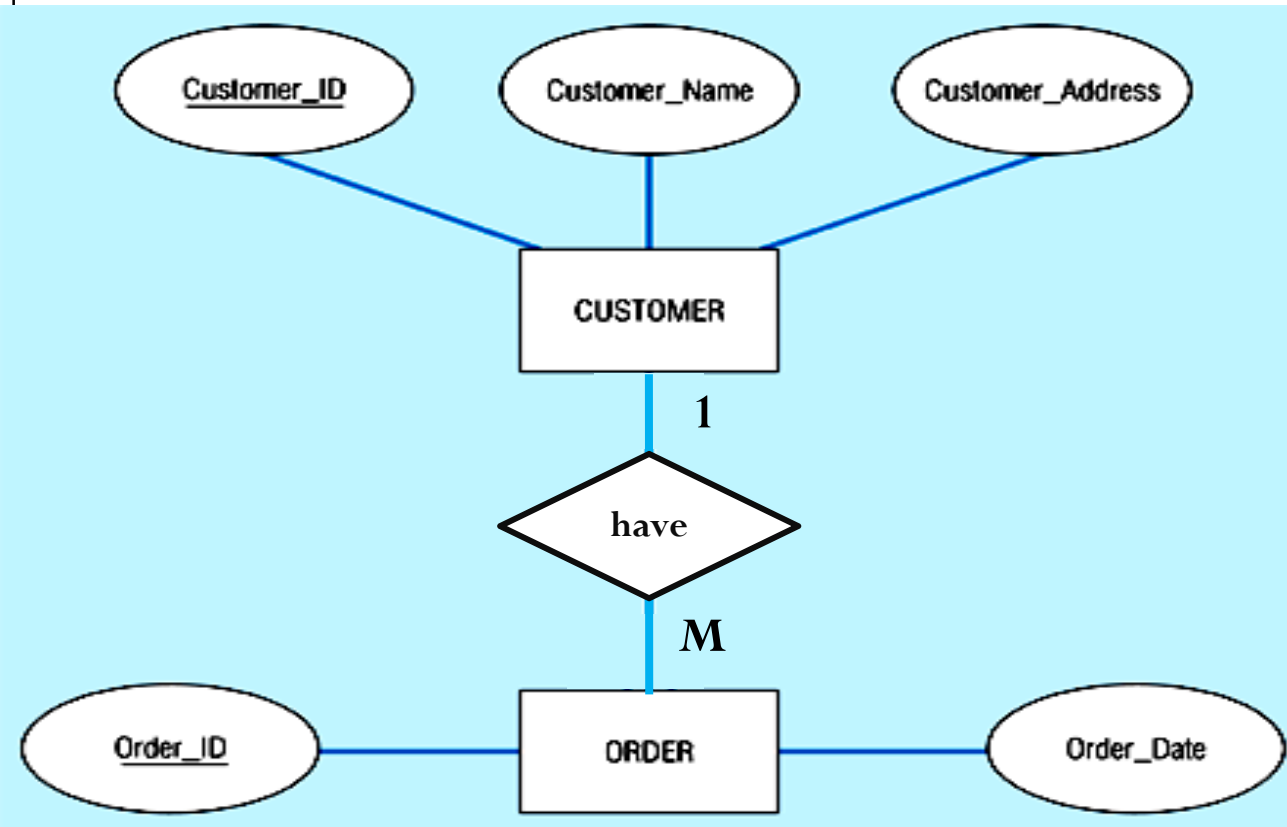
รหัสโปรเจ็ค	ชื่อโปรเจ็ค
P01	ระบบหอพักออนไลน์
P02	ระบบขายออนไลน์
P03	ระบบประวัติบุคลากร
P04	ระบบเช่ารถออนไลน์

การรับผิดชอบโปรเจ็ค

รหัส	รหัสนักศึกษา	รหัสโปรเจ็ค	ภาคเรียน	ปีการศึกษา
1	49001	P01	2	2551
2	49002	P02	2	2552
3	49003	P03	2	2553
4	49001	P04	1	2552

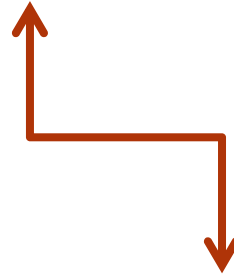
## การแปลง one-to-many Relationship เป็น Relation (ตาราง)

นำ Primary Key ของ Entity ฝั่ง One มาเป็น Foreign Key ให้ Relation ของ Entity ฝั่ง Many



## CUSTOMER

<u>Customer ID</u>	Customer_Name	Customer_Address
01	นายธนภัทร ศรีสุธรรม	กรุงเทพ
02	นางรัชฎาพร ศรีแก้ว	นนทบุรี
03	นายเอกชัย ปานมาก	สมุทรปราการ

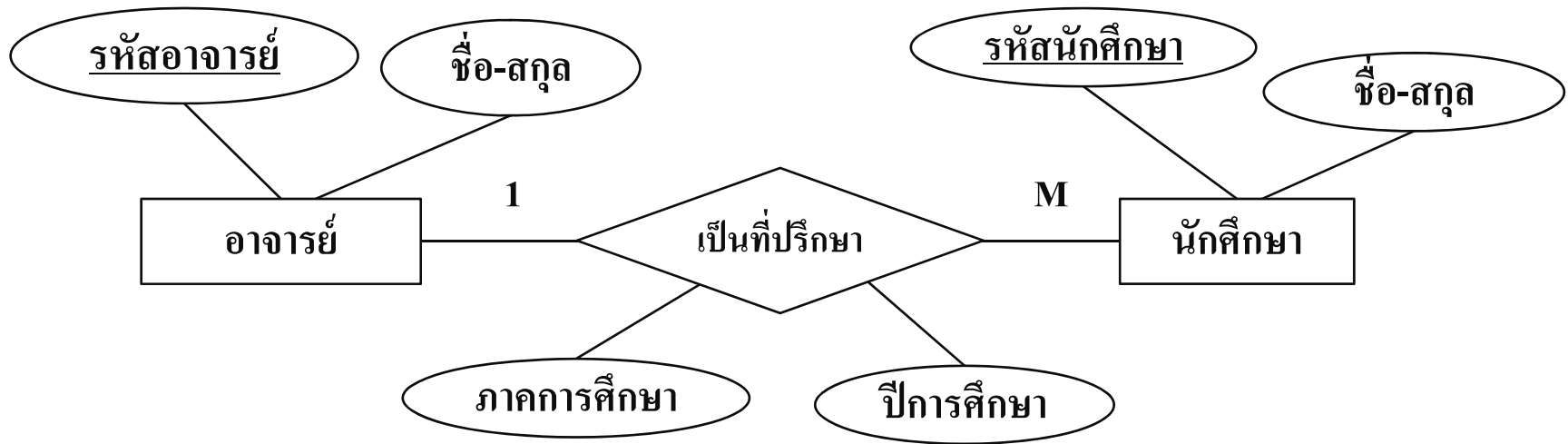


## ORDER

<u>Order Id</u>	Order_Date	Customer_Id
OD01	11/01/53	01
OD02	12/01/53	01
OD03	12/03/53	02



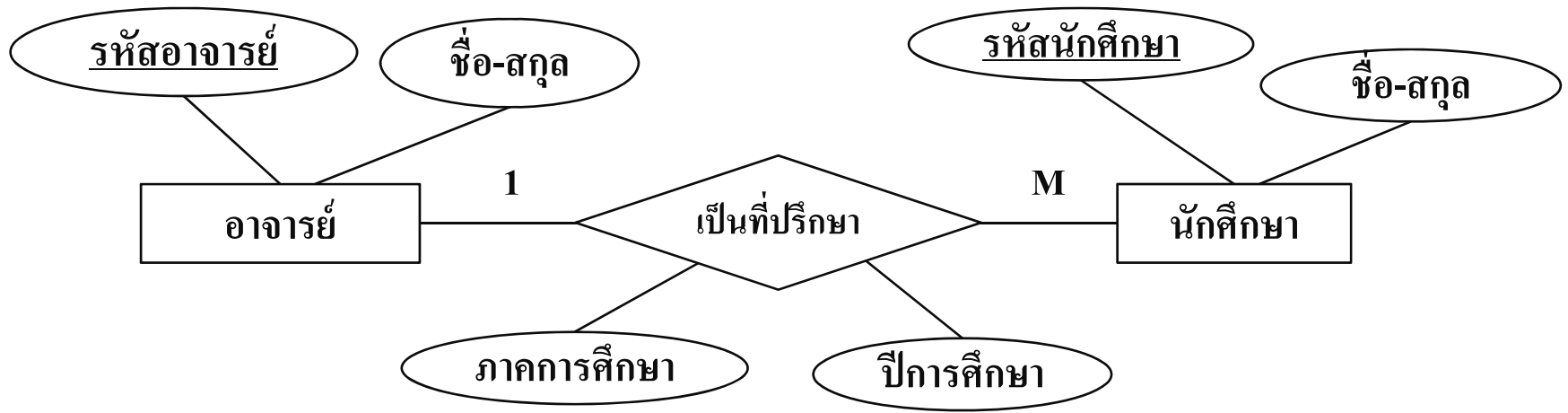
## การแปลง one-to-many Relationship เป็น Relation (ตาราง) มีเวลาเกี่ยวข้องกับ



- ✓ จากตัวอย่าง เป็น ER แสดงความสัมพันธ์ ระหว่างอาจารย์กับนักศึกษา
- ✓ เพื่อให้ตรวจสอบได้ว่า ในภาคเรียนและปีการศึกษาใดๆ อาจารย์แต่ละคนเป็น ที่ปรึกษานักศึกษาคนไหนบ้าง

# การแปลง one-to-one Relationship เป็น Relation (ตาราง) ที่มีเวลาเข้ามาเกี่ยวข้อง

- ✓ การแปลงความสัมพันธ์ แบบ 1:M ที่มีเงื่อนไขเรื่องเวลาเข้ามาด้วย จะต้องแปลงเป็นรีเลชันใหม่เกิดขึ้น ในตัวอย่างคือ รีเลชัน การรับผิดชอบโปรเจ็ค
- ✓ จากตัวอย่างจะได้จะได้ทั้งหมด 3 รีเลชันคือ
  - นักศึกษา
  - อาจารย์
  - **การเป็นที่ปรึกษา** (แปลงมาจาก 1:M ที่มีเงื่อนไขเวลาเข้ามาเกี่ยวข้องด้วย)



แอททริบิวต์ที่จะเป็น คีย์หลัก (Primary Key) ของรีเลชัน **การเป็นที่ปรึกษา** คือ **ทางเลือกที่ 1**

ใช้ คีย์หลักของ รีเลชัน อาจารย์ (คือ รหัสอาจารย์) และ คีย์หลักของรีเลชันนักศึกษา (คือ รหัสนักศึกษา) มาเป็นคีย์หลัก ในขณะเดียวกัน ก็ทำหน้าที่เป็นคีย์นอกด้วย ผลที่ได้  
การเป็นที่ปรึกษา(**รหัสอาจารย์,รหัสนักศึกษา**,ภาคการศึกษา,ปีการศึกษา)

## ทางเลือกที่ 2

สร้างคีย์หลักขึ้นมาใหม่ ในที่นี้ตั้งชื่อให้เป็น รหัส แต่นำคีย์หลักของ รีเลชัน อาจารย์ และ คีย์หลักของรีเลชันนักศึกษา มาเป็นคีย์นอก

การเป็นที่ปรึกษา(**รหัส**,รหัสอาจารย์,รหัสนักศึกษา,ภาคการศึกษา,ปีการศึกษา)

(เพิ่มเติม)

อาจารย์

รหัสอาจารย์	ชื่อ สกุล
T001	นางสาวอมสิน น่ารัก
T002	นางสาวน้ำค้าง สวยดี
T003	นางสาวหนึ่ง สวยมาก

นักศึกษา

รหัสนักศึกษา	ชื่อ สกุล
49001	นายปราโมทย์ สิ้นมั่นคง
49002	นายคงเดช มีถาวร
49003	นายปรีชา กิจตรง
50001	นายเอก ใจดี

การเป็นทีปรีกษา

คีย์หลัก

รหัสอาจารย์	รหัสนักศึกษา	ภาคเรียน	ปีการศึกษา
T001	49001	2	2551
T002	49002	2	2552
T003	49003	2	2553
T001	50001	1	2554

คีย์นอก

คีย์นอก

(เพิ่มเติม)

อาจารย์

รหัสอาจารย์	ชื่อ สกุล
T001	นางสาวอมสิน น่ารัก
T002	นางสาวน้ำค้าง สวยดี
T003	นางสาวหนึ่ง สวยมาก

นักศึกษา

รหัสนักศึกษา	ชื่อ สกุล
49001	นายปราโมทย์ สิ้นมันคง
49002	นายคงเดช มีถาวร
49003	นายปรีชา กิจตรง
50001	นายเอก ใจดี

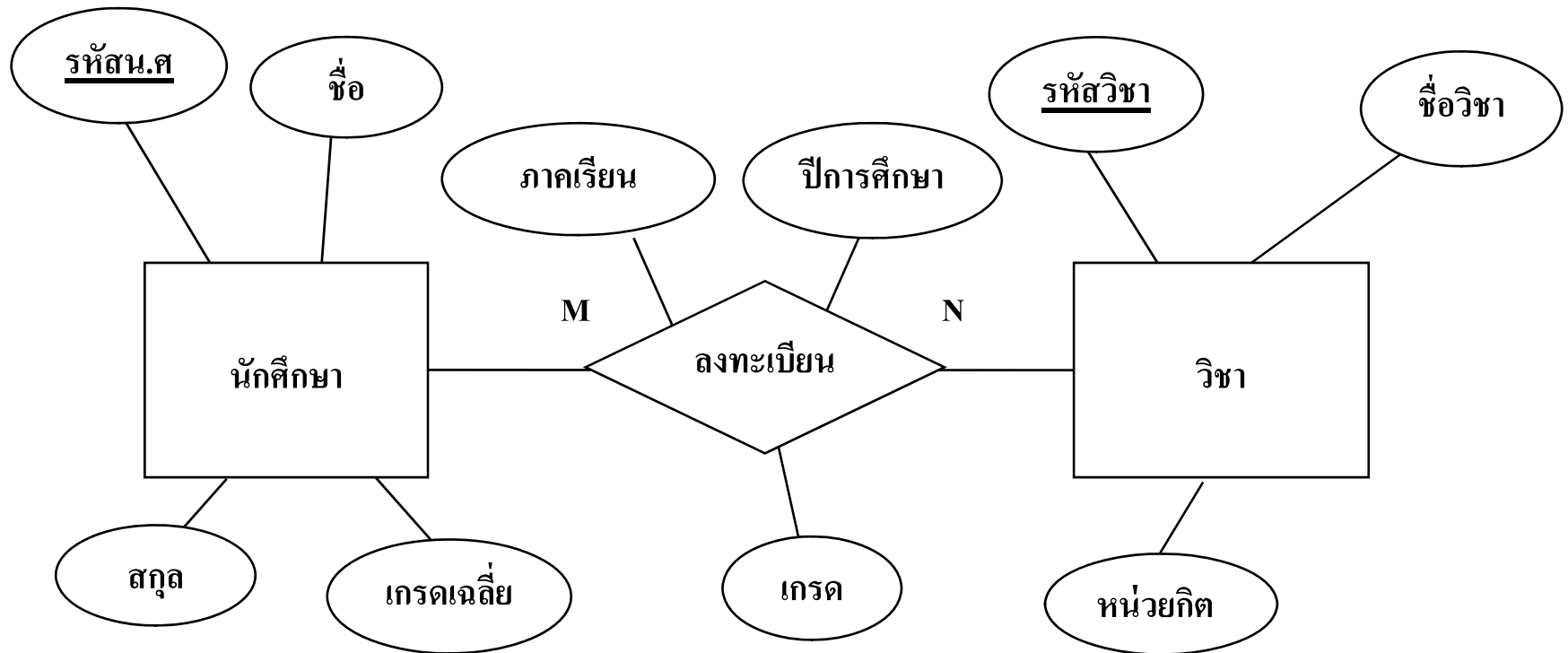
การเป็นที่ปรึกษา

รหัส	รหัสนักศึกษา	รหัสโปรเจกต์	ภาคเรียน	ปีการศึกษา
1	T001	49001	2	2551
2	T002	49002	2	2552
3	T003	49003	2	2553
4	T001	50001	1	2552

# การแปลง many-to-many Relationship เป็น Relation (ตาราง)

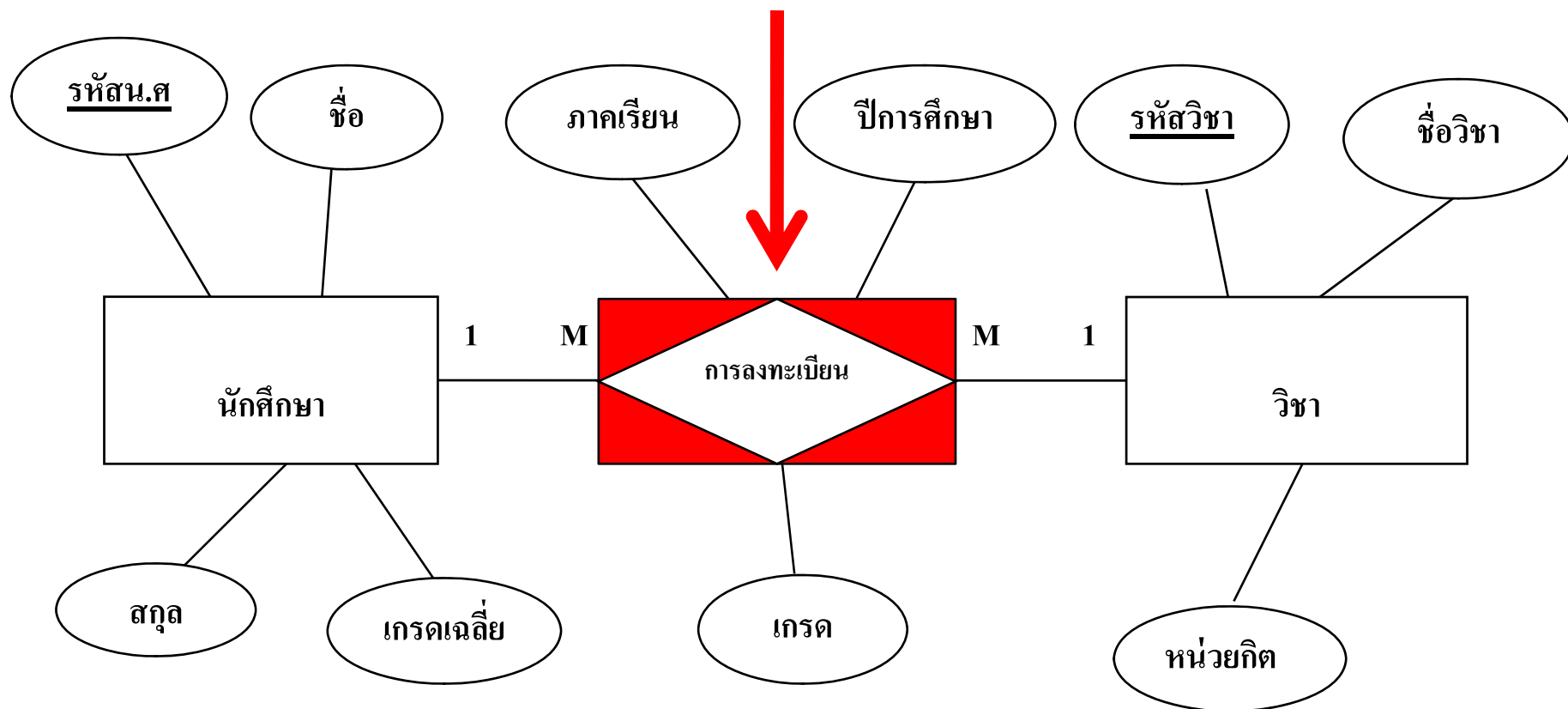
❑ สามารถแปลงความสัมพันธ์ M:N ให้อยู่ในรูปของเอนทิตีเชิงสัมพันธ์ (Associative Entity) ได้

❑ จาก ER-Diagram การลงทะเบียนของนักศึกษา ความสัมพันธ์ “ลงทะเบียน” สามารถแปลงให้อยู่ในรูป เอนทิตีเชิงสัมพันธ์ ได้ดังรูปต่อไป





❑ ความสัมพันธ์ **ลงทะเบียน** → เปลี่ยนเป็นเอนทิตีเชิงสัมพันธ์ **การลงทะเบียน**



# การแปลง many-to-many Relationship เป็น Relation (ตาราง)

- ❑ แปลงความสัมพันธ์แบบ many-to-many ให้เป็น Relation ใหม่
- ❑ นำคีย์หลักของ เอนติตี้ทั้ง 2 เอนติตี้ที่มีความสัมพันธ์กัน มาใส่เป็น คีย์หลักของ Relation ที่เกิดจากความสัมพันธ์แบบ many to many (M:N)
- ❑ ในกรณีที่ Relation ที่ได้จากการแปลงความสัมพันธ์ (M:N) มีแอททริบิวต์ที่จำเป็นต้องนำมาเป็นประกอบของคีย์หลักได้ เพื่อให้บอกความแตกต่างของแต่ละแถวได้ ต้องนำมาเป็นส่วนหนึ่งของคีย์หลักด้วย
- ❑ ส่วนเอนติตี้อื่นๆ ก็แปลงตามปกติเหมือนขั้นตอนก่อนหน้านี้ที่ได้กล่าวมาแล้ว

จากตัวอย่าง จะได้ 3 relation คือ

1) นักศึกษา (รหัสสน.ศ., ชื่อ, สกุล, เกรดเฉลี่ย)

2) วิชา (รหัสวิชา, ชื่อวิชา , หน่วยกิต)

3) การลงทะเบียน(รหัสนักศึกษา,รหัสวิชา,ภาคเรียน,ปีการศึกษา,เกรด)

แต่ต้องพิจารณาเงื่อนไขที่ต้องการด้วย เช่น ถ้าให้นักศึกษาสามารถลงทะเบียนวิชาเดิมซ้ำได้เพราะไม่ผ่าน ต้องกำหนด ภาคเรียนกับปีการศึกษาเป็นส่วนหนึ่งของคีย์หลักด้วย ดังนั้นจะได้ผลลัพธ์ดังนี้

“การลงทะเบียน(รหัสนักศึกษา,รหัสวิชา,ภาคเรียน,ปีการศึกษา,เกรด)”



## ได้ผลลัพธ์ทั้งหมด 3 relations

1) นักศึกษา (รหัสนักศึกษา, ชื่อ, สกุล, เกรดเฉลี่ย)

2) วิชา (รหัสวิชา, ชื่อวิชา, หน่วยกิต)

**Composite primary key (คีย์หลักที่ประกอบด้วยแอททริบิวต์หลายตัว)**

3) การลงทะเบียน (รหัสนักศึกษา, รหัสวิชา, ภาคเรียน, ปีการศึกษา, เกรด) ← Relation ที่เกิดจากความสัมพันธ์แบบ many to many

Foreign key

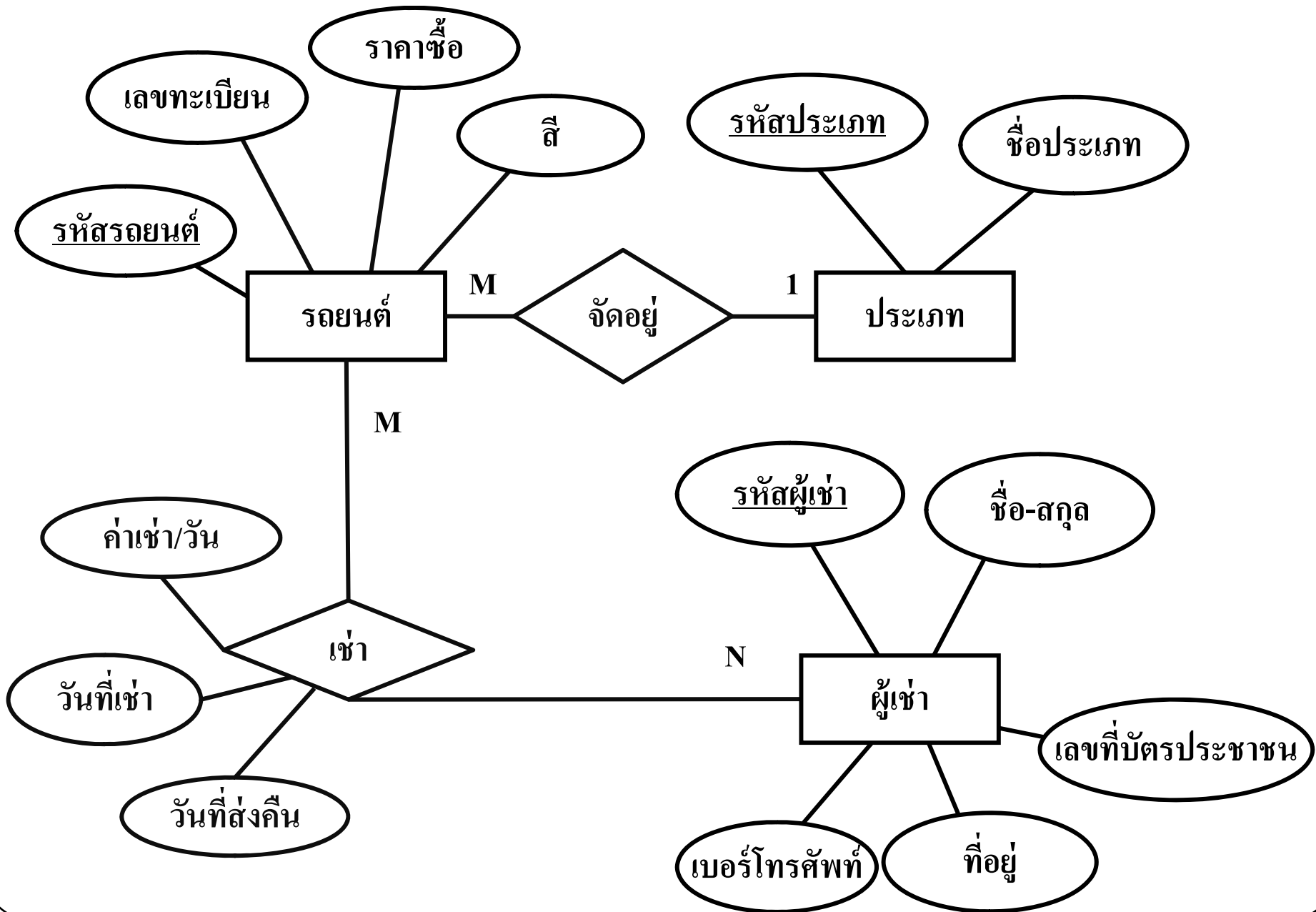
คีย์นอก

Foreign key

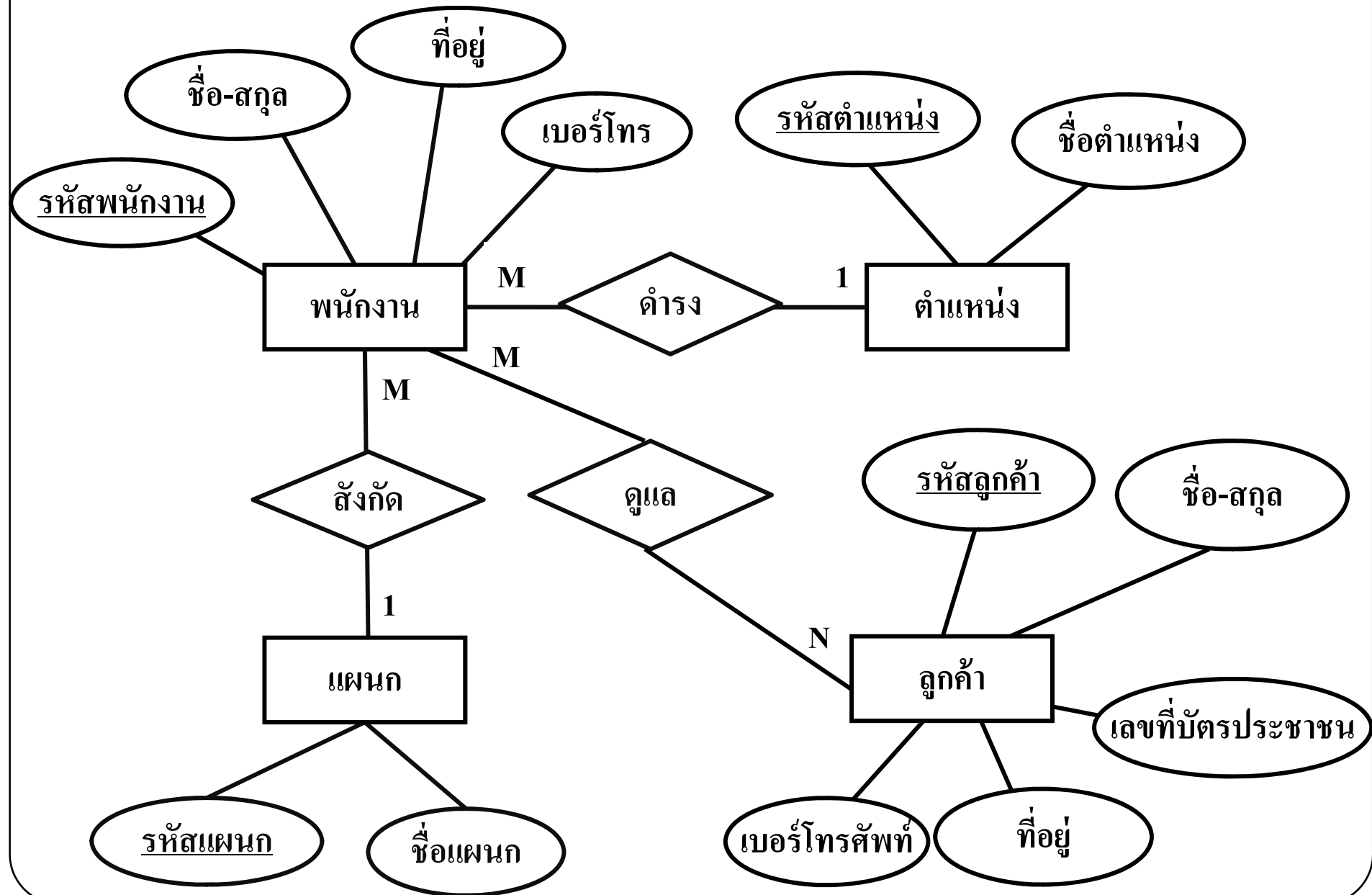
คีย์นอก

จงแปลงแบบจำลอง E-R ต่อไปนี้ เป็นแบบจำลองฐานข้อมูลเชิงสัมพันธ์

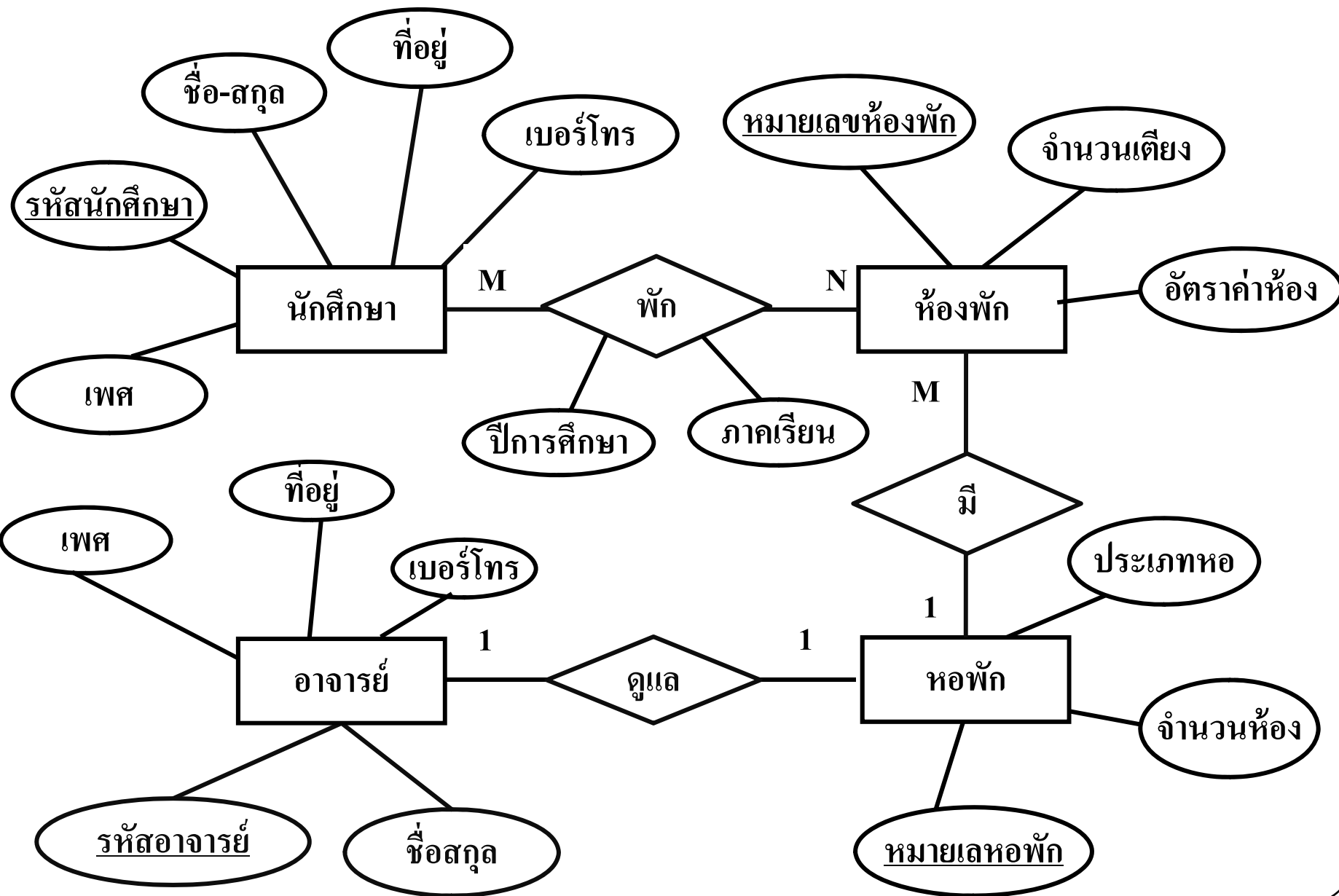
# แบบจำลอง E-R ระบบเช่ารถยนต์



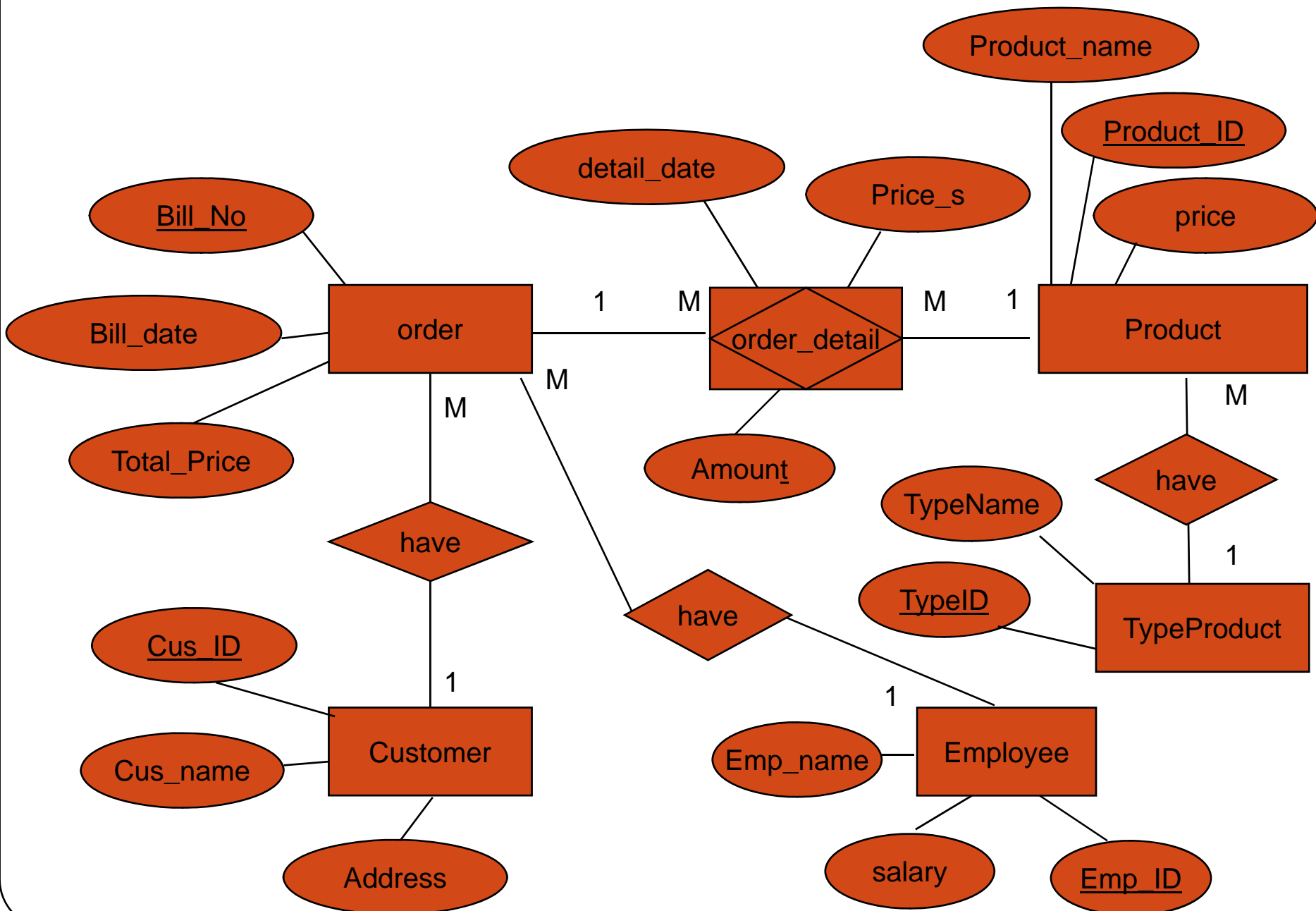
# แบบจำลอง E-R ระบบบุคลากร



# แบบจำลอง E-R ระบบหอพักนักศึกษา



# ER-Diagramของระบบการขายสินค้า



# กระบวนการปรับบรรทัดฐาน (Normalization)

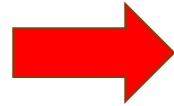
- ความหมายและจุดประสงค์ของการนอร์มัลไลเซชัน
- ฟังก์ชันการขึ้นต่อกัน (Function Dependencies)
- กระบวนการนอร์มัลไลเซชัน

## ความหมายและจุดประสงค์ของการนอร์มัลไลเซชัน

- นอร์มัลไลเซชัน เป็นทฤษฎีที่ผู้ออกแบบฐานข้อมูลจะต้องนำมาใช้ในการแปลงข้อมูลที่อยู่ในรูปแบบที่ซับซ้อน ให้อยู่ในรูปแบบที่ง่ายต่อการนำไปใช้งานและก่อให้เกิดปัญหาน้อยที่สุด
- ในบทนี้จะสอนกระบวนการนอร์มัลไลเซชัน ใน 3 ระดับด้วยกันคือ
  - ☐ นอร์มัลไลเซชันระดับที่ 1 หรือเรียกว่า 1NF
  - ☐ นอร์มัลไลเซชันระดับที่ 2 หรือเรียกว่า 2NF
  - ☐ นอร์มัลไลเซชันระดับที่ 3 หรือเรียกว่า 3NF



วิเคราะห์ความ  
ต้องการของผู้ใช้



E-R Diagram



รีเลชันที่มีรูปแบบไม่เป็นบรรทัดฐาน  
(Unnormalized relation)

1 NF

2 NF

3 NF

บอยด์ คอตต์

4 NF

กระบวนการปรับบรรทัด  
ฐาน (Normalization)

รีเลชันที่มีเป็นบรรทัดฐาน  
(Normalized relation)

## วัตถุประสงค์ของนอร์มัลไลส์ (Normalization)

- ❑ ลดความซ้ำซ้อนของข้อมูล เมื่อลดความซ้ำซ้อนก็ทำให้ลดเนื้อหาที่ใช้ในการจัดเก็บข้อมูล
- ❑ ลดปัญหาความไม่ถูกต้องของข้อมูล เมื่อข้อมูลไม่เกิดความซ้ำซ้อนทำให้การปรับปรุงข้อมูลสามารถทำได้จากแหล่งข้อมูลเพียงแหล่งเดียว
- ❑ ลดความผิดพลาดที่อาจเกิดจากการปรับปรุงข้อมูล (update anomalies) ซึ่งประกอบด้วย

## ความซ้ำซ้อนและข้อผิดพลาดจากการปรับปรุงข้อมูล

- แนวคิดหลักอันสำคัญของการออกแบบฐานข้อมูลเชิงสัมพันธ์ คือ การออกแบบให้มีการเก็บข้อมูลซ้ำซ้อนน้อยที่สุด
- เพื่อประหยัดเนื้อที่ในการเก็บข้อมูลและลดปัญหาที่จะเกิดดังตัวอย่างต่อไปนี้

## ตัวอย่างการออกแบบฐานข้อมูลที่ดี

### Employee (พนักงาน)

รหัสพนักงาน	ชื่อ-สกุล	ตำแหน่ง	เงินเดือน	รหัสสาขา
SG21	ชูชาติ สุขศรี	ผู้จัดการ	30000	B005
SG37	ศิริ ควงเด่น	ผู้ช่วย	20000	B003
SG14	ควงใจ มีสุข	เลขานุการ	20000	B003
SG09	อัจฉรา เปี้ยวแก้ว	ผู้จัดการ	30000	B007

### Branch (สาขา)

รหัสสาขา	ที่อยู่
B005	เชียงใหม่
B003	กรุงเทพ
B007	พิษณุโลก

## ตัวอย่างการออกแบบฐานข้อมูลที่จะมีปัญหาของความซ้ำซ้อนตามมา

### Employee\_Branch (รวมรายละเอียดของพนักงานไว้ด้วยกันกับรายละเอียดของสาขา)

รหัสพนักงาน	ชื่อ-สกุล	ตำแหน่ง	เงินเดือน	รหัสสาขา	ที่อยู่
SG21	ชูชาติ สุขศรี	ผู้จัดการ	30000	B005	เชียงใหม่
SG37	ศิริ ควงเด่น	ผู้ช่วย	20000	B003	กรุงเทพ
SG14	ควงใจ มีสุข	เลขานุการ	20000	B003	กรุงเทพ
SG09	อัจฉรา เปี้ยวแก้ว	ผู้จัดการ	30000	B007	พิษณุโลก

## ตัวอย่างปัญหาความซ้ำซ้อนในข้อมูล รีเลชัน EMPLOYEE\_BRANCE

<u>รหัสพนักงาน</u>	<u>ชื่อ-สกุล</u>	<u>ตำแหน่ง</u>	<u>เงินเดือน</u>	<u>รหัสสาขา</u>	<u>ที่อยู่</u>
SG21	ชูชาติ สุขศิริ	ผู้จัดการ	30000	B005	เชียงใหม่
SG37	ศิริ ดวงเด่น	ผู้ช่วย	20000	B003	กรุงเทพ
SG14	ดวงใจ มีสุข	เลขานุการ	20000	B003	กรุงเทพ
SG09	อัจฉรา เขียวแก้ว	ผู้จัดการ	30000	B007	พิษณุโลก

### ความผิดพลาดจากการเพิ่ม

○ถ้าต้องการเพิ่มพนักงานใหม่ ที่อยู่สาขา B005

○จะต้องกรอก B005 และที่อยู่สาขา คือ เชียงใหม่ เพิ่มอีก

## ตัวอย่างปัญหาความซ้ำซ้อนในข้อมูล รีเลชัน EMPLOYEE\_BRANCE

รหัสพนักงาน	ชื่อ-สกุล	ตำแหน่ง	เงินเดือน	รหัสสาขา	ที่อยู่
SG21	ชูชาติ สุขศิริ	ผู้จัดการ	30000	B005	เชียงใหม่
SG37	ศิริ ควงเด่น	ผู้ช่วย	20000	B003	กรุงเทพ
SG14	ควงใจ มีสุข	เลขานุการ	20000	B003	กรุงเทพ
SG09	อัจฉรา เขียวแก้ว	ผู้จัดการ	30000	B007	พิษณุโลก

### ความผิดพลาดจากการเพิ่ม

- ☐ ถ้าต้องการเพิ่มสาขา จะมีปัญหาคือ ตารางนี้มีทั้งข้อมูลพนักงานและข้อมูลสาขาอยู่รวมกัน
- ☐ หากจะเพิ่มเฉพาะ รหัสสาขา และ ที่อยู่ ก็ไม่ได้เพราะ รหัสพนักงาน จะมีค่าว่างไม่ได้เพราะเป็น Primary Key ของตาราง
- ☐ ดังนั้นจะบันทึกได้ก็ต่อเมื่อมีพนักงานแล้ว

## ความผิดพลาดจากการลบข้อมูล

- ❑ ถ้าลบข้อมูลหนึ่งแล้วส่งผลกระทบกับข้อมูลอื่น ที่ต้องถูกลบตาม
  - ❑ เช่น พนักงานรหัส SG21 ลาออก ก็ลบแถวนั้นออก
  - ❑ ข้อมูลสาขา B005 ก็จะหายไปด้วย

## ข้อผิดพลาดจากการเปลี่ยนแปลง

- ❑ ในกรณีที่ต้องการเปลี่ยนแปลงข้อมูลบางตัวของสาขา
  - ❑ เช่น เปลี่ยนที่อยู่ของ B003 ก็ต้องเปลี่ยนหลายที่
  - ❑ ถ้าหากมีพนักงานสังกัดสาขานี้หลายที่ก็ต้องไปตามแก้ทุก ๆ ที่

ดังนั้นเราควรแยกตาราง Employee\_Branch ออกเป็นสองตาราง คือ ตารางพนักงาน และตารางสาขา

## พนักงาน

<u>รหัสพนักงาน</u>	ชื่อ-สกุล	ตำแหน่ง	เงินเดือน	รหัสสาขา
SG21	ชูชาติ สุขศรี	ผู้จัดการ	30000	B005
SG37	ศิริ ควงเด่น	ผู้ช่วย	20000	B003
SG14	ควงใจ มีสุข	เลขานุการ	20000	B003
SG09	อัจฉรา เขียวแก้ว	ผู้จัดการ	30000	B007

## สาขา

<u>รหัสสาขา</u>	ที่อยู่
B005	เชียงใหม่
B003	กรุงเทพ
B007	พิษณุโลก



## ฟังก์ชันการขึ้นต่อกัน (Functional Dependency : FD)

- ถ้าให้  $X$  และ  $Y$  เป็น Attribute ใน Relation ใดๆ แทนด้วย  $R(X,Y)$  Attribute  $Y$  เป็น จะถูกเรียกว่ามีฟังก์ชันการขึ้นต่อกันกับแอททริบิวต์  $X$  ก็ต่อเมื่อ แต่ละค่าที่ไม่ซ้ำกันของแอททริบิวต์  $X$  มีข้อมูลของ  $Y$  ที่เกี่ยวข้องกับ  $X$  เพียง 1 ค่า
- เขียนแทนด้วย สัญลักษณ์  $X \rightarrow Y$

# ตัวอย่าง

<u>EmployeeNo</u>	Name	Position
S01	ฉัตรชัย มีสมบัติ	Manager
S02	เอกชัย ใจดี	Manager Assistant
S03	มนีรัตน์ เจริญสุข	Manager
S04	ขวัญชัย ใจเพชร	Manager Assistant
S05	มานพ เกตุแก้ว	Staff
S06	ดวงกมล ทิพย์เทพ	Staff

EmployeeNo  Position

ตัวอย่าง A) employeeNo



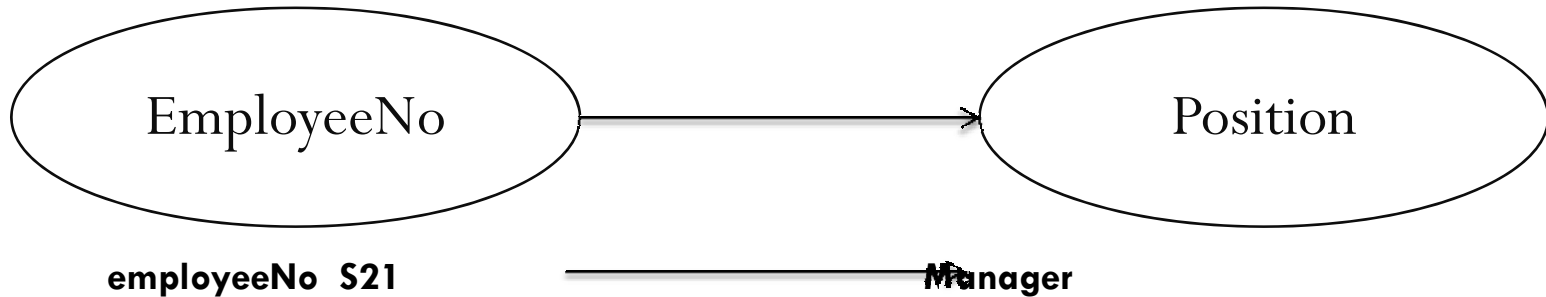
Position

B) Position (not)

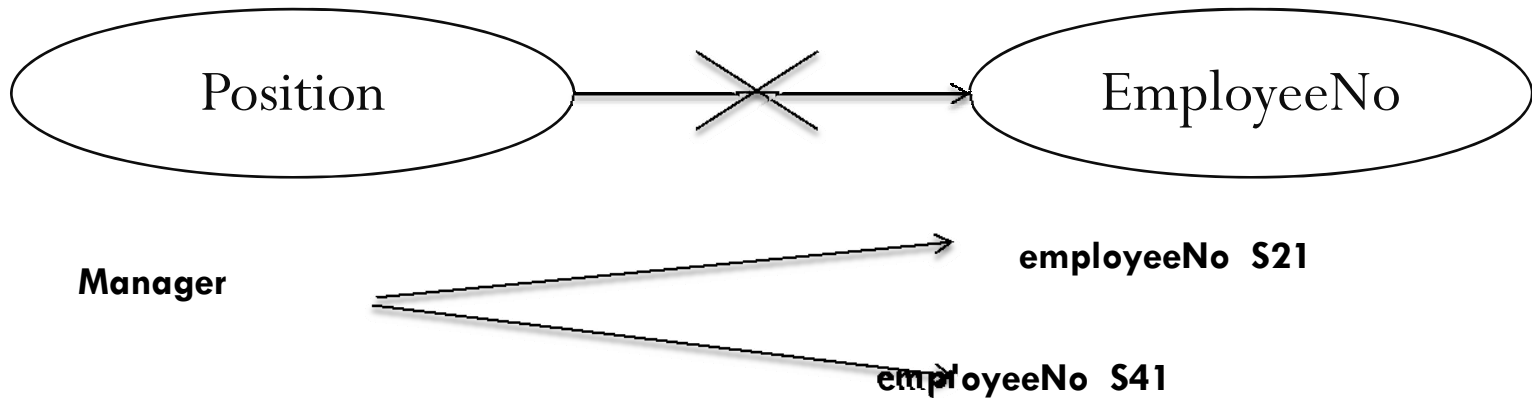


employeeNo

**A**



**B**



## ชนิดของฟังก์ชันการขึ้นต่อกัน ( Functional Dependency :FD)

- 1) Complete dependencies การขึ้นต่อกันอย่างสมบูรณ์
- แอททริบิวต์ที่ไม่ใช่คีย์หลัก ขึ้นต่อ แอททริบิวต์หรือกลุ่มของแอททริบิวต์ที่เป็นคีย์หลัก
- ตัวอย่าง ตารางที่มีแอททริบิวต์ค่าเดียวทำหน้าที่เป็นคีย์หลัก คือ หมายเลขบัตรประชาชน

<u>หมายเลขบัตรประชาชน</u>	ชื่อเจ้าของบัตร
3440100634931	กนกวรรณ พ่วงพงษ์
3437283420343	ชาติชาย เตชะวงศ์
2938742039485	กิงกาญ เดชาทรัพย์

หมายเลขบัตรประชาชน → ชื่อเจ้าของบัตร

## ชนิดของฟังก์ชันการขึ้นต่อกัน ( Functional Dependency :FD)

- ตัวอย่าง ตารางที่มีแอทริบิวต์หลายตัวขึ้นกับคีย์หลักตัวเดียว

หมายเลขบัตรประชาชน	ชื่อเจ้าของบัตร	วันเกิด	วันที่ทำบัตร
3440100634931	กนกวรรณ พ่วงพงษ์	27/03/2520	28/04/2553
3437283420343	ชาติชาย เตชะวงศ์	23/06/2522	25/02/2553
2938742039485	กิงกาญ เดชาทรัพย์	21/04/2525	19/0125/52

หมายเลขบัตรประชาชน  $\longrightarrow$  ชื่อเจ้าของบัตร,วันเกิด,วันที่ทำบัตร

## ชนิดของฟังก์ชันการขึ้นต่อกัน ( Functional Dependency :FD)

□ ตัวอย่าง ตารางที่มีแอทริบิวหลายตัวรวมกันเป็นคีย์หลัก คือ รหัสนักศึกษา รหัสวิชา

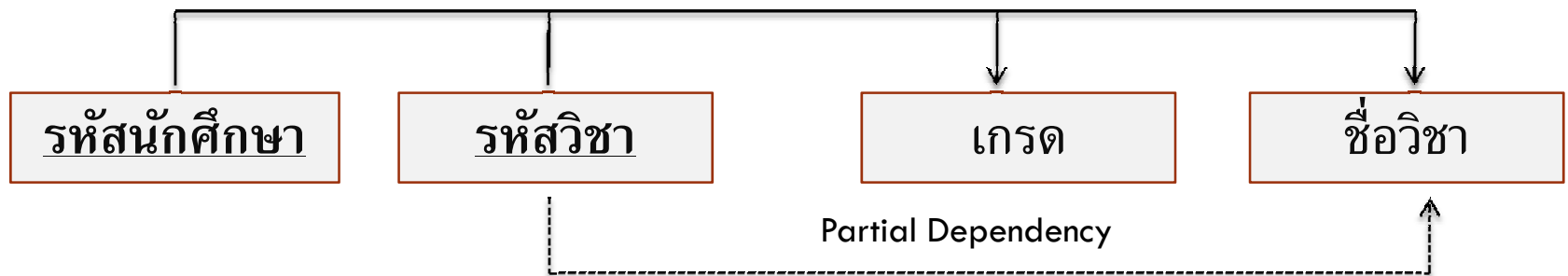
ตารางการลงทะเบียน

<u>รหัสนักศึกษา</u>	<u>รหัสวิชา</u>	เกรด
520014001	S001	A
520014001	S002	B
520014002	S001	C
520014002	S002	A

รหัสนักศึกษา,รหัสวิชา  $\longrightarrow$  เกรด

# ชนิดของฟังก์ชันการขึ้นต่อกัน ( Functional Dependency :FD)

- **2) Partial Dependency** (การขึ้นต่อกันบางส่วน)
- เกิดขึ้นเมื่อกีย์หลักประกอบด้วยหลาย Attribute รวมกัน
- เมื่อแอตทริบิวต์บางส่วนของคีย์หลัก สามารถไประบุค่าแอตทริบิวต์ตัวอื่น ๆ ที่ไม่ใช่คีย์หลักของรีเลชันได้



รหัสนักศึกษา, รหัสวิชา  $\Rightarrow$  เกรด , ชื่อวิชา  
รหัสวิชา  $\Rightarrow$  ชื่อวิชา

## ตัวอย่างฟังก์ชันการขึ้นต่อกันแบบ Partial

<u>รหัสนักศึกษา</u>	<u>รหัสวิชา</u>	เกรด	ชื่อวิชา
534267001	F01	A	การเขียนโปรแกรม
534267001	F02	B	การออกแบบฐานข้อมูล
534267002	F01	D	การเขียนโปรแกรม
534267002	F02	A	การออกแบบฐานข้อมูล
534267003	F01	A	การเขียนโปรแกรม
534267003	F02	C	การออกแบบฐานข้อมูล



# ชนิดของฟังก์ชันการขึ้นต่อกัน (Functional Dependency : FD)

- 3 Transitive Dependency เกิดขึ้นเมื่อ Attribute ที่ไม่ใช่ Primary Key ไปขึ้นอยู่กับ Attribute อื่นที่ไม่ใช่ Primary Key ในรีเลชันนั้น ๆ

เลขประจำตัว	ชื่อ สกุล	ที่อยู่	ตำแหน่ง	รถประจำตำแหน่ง
01	ฉัตรชัย มีสมบัติ	กรุงเทพ	ผู้จัดการ	BMW
02	เอกชัย ใจดี	นนทบุรี	ผู้ช่วยผู้จัดการ	Honda
03	มนีรัตน์ เจริญสุข	เชียงใหม่	ผู้จัดการ	BMW
04	ขวัญชัย ใจเพชร	ราชบุรี	ผู้ช่วยผู้จัดการ	Honda

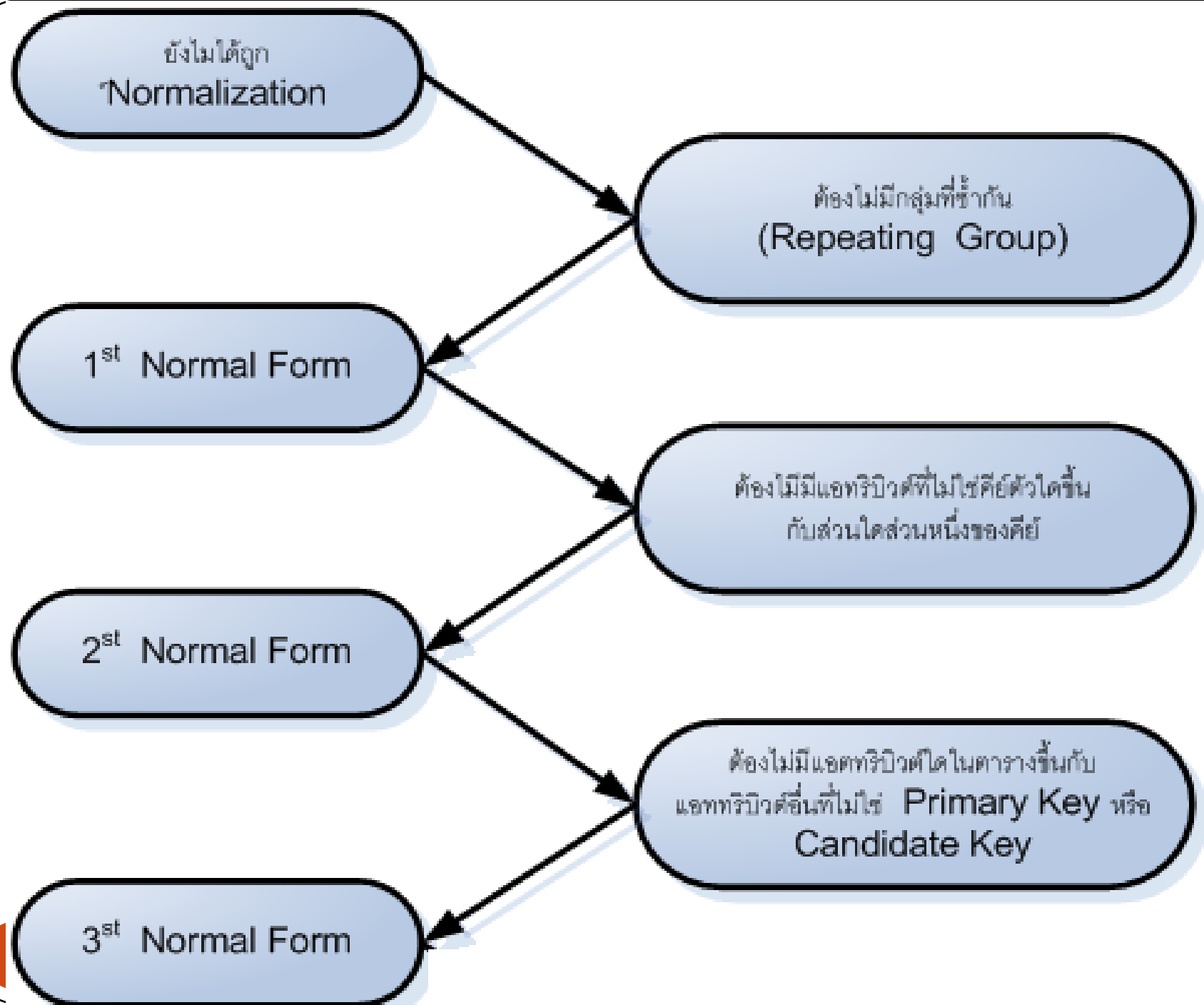
คำอธิบาย เลขประจำตัว เป็นคีย์หลัก (Primary Key) ของตาราง

เลขประจำตัว      → ชื่อสกุล, ที่อยู่, ตำแหน่ง

ตำแหน่ง          → รถประจำตำแหน่ง

## นอร์มัลไลเซชัน (Normalization)

- Normalization คือ กระบวนการปรับปรุงโครงสร้างข้อมูลของฐานข้อมูลที่มีความซ้ำซ้อนให้อยู่ในรูปแบบที่เป็นบรรทัดฐาน (Normal Form)
- การนอร์มัลไลเซชันมีได้ถึง 5 ระดับ ในระดับที่ 3 ก็จัดว่าเพียงพอสำหรับการออกแบบฐานข้อมูลในปัจจุบัน
  - 1NF - กำจัด repeating group  
(กำจัดกลุ่มของข้อมูลที่มีความซ้ำซ้อน)
  - 2NF - กำจัด partial dependency  
(กำจัดความสัมพันธ์ต่อกันบางส่วน)
  - 3NF - กำจัด transitive dependency  
(กำจัดความสัมพันธ์ต่อกันของแอตทริบิวต์ที่ไม่ใช่คีย์หลัก)



## First Normal Form (1NF)

- ทุก Attribute ในแต่ละ record จะเป็น single value ไม่มี ค่าของกลุ่ม ข้อมูลที่ซ้ำกัน (Repeating Group)
- ข้อมูลทุกแถว (Tuple) ต้องมีค่าไม่ซ้ำกัน

ตารางที่มีลักษณะข้อมูลเป็น Repeating group

รหัสนักศึกษา	ชื่อ	นามสกุล	รหัสวิชาที่ลงทะเบียน
001	สมชาย	สมใจนึก	<div>204-101</div> <div>204-204</div> <div>204-205</div>
002	ธีรชาย	บุญมาศ	<div>204-102</div> <div>204-204</div>

Repeating Group

ตารางที่มีลักษณะข้อมูลเป็น Repeating group

<u>รหัสนักศึกษา</u>	<u>ชื่อ</u>	<u>นามสกุล</u>	<u>รหัสวิชาที่ลงทะเบียน</u>
001	สมชาย	สมใจนึก	204-101
			204-204
			204-205
002	ธีรชาย	บุญมาศ	204-102
			204-204

เราสามารถทำให้อยู่ในรูป 1NF ได้ดังนี้

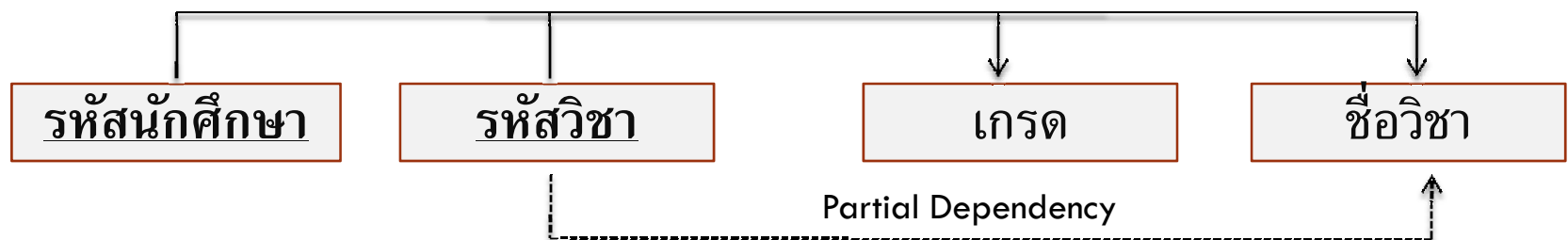
<u>รหัสนักศึกษา</u>	<u>ชื่อ</u>	<u>นามสกุล</u>	<u>รหัสวิชาที่ลงทะเบียน</u>
001	สมชาย	สมใจนึก	204-100
001	สมชาย	สมใจนึก	204-204
001	สมชาย	สมใจนึก	204-125
002	ธีรชาย	บุญมาศ	204-102
002	ธีรชาย	บุญมาศ	204-204

## Second Normal Form (2NF)

- 1. ต้องเป็น First Normal Form (1NF) มาก่อน
- 2. ต้องไม่มี Partial Dependency (การขึ้นต่อกันบางส่วน)
- สรุปก็คือ นอร์มัลไลเซชันระดับที่ 2 (Second normal form : 2NF) เป็นการขจัดแอตทริบิวต์ที่ ไม่ขึ้นกับทั้งส่วนของคีย์หลัก ออกไป เพื่อให้แอตทริบิวต์อื่นทั้งหมดขึ้นตรงกับส่วนที่เป็นคีย์หลักทั้งหมดเท่านั้น

## ตัวอย่างตารางที่ Partial Dependency (การขึ้นต่อกันบางส่วน)

รหัสนักศึกษา	รหัสวิชา	เกรด	ชื่อวิชา
534267001	F01	A	การเขียนโปรแกรม
534267001	F02	B	การออกแบบฐานข้อมูล
534267002	F01	D	การเขียนโปรแกรม
534267002	F02	A	การออกแบบฐานข้อมูล
534267003	F01	A	การเขียนโปรแกรม
534267003	F02	C	การออกแบบฐานข้อมูล



รหัสนักศึกษา, รหัสวิชา  
รหัสวิชา



เกรด, ชื่อวิชา  
ชื่อวิชา

## Second Normal Form (2NF)

### □ วิธีขจัดปัญหา

- 1) ต้องสร้างตารางเพิ่ม
- 2) นำคอลัมย์ที่มีปัญหาไปใส่ในตารางที่สร้างเพิ่ม
- 3) กำหนดคีย์หลักให้กับตารางที่สร้างใหม่
- 4) แอททริบิวต์ใดในตารางเดิม เมื่อนำไปใส่ในตารางใหม่ให้ตัดออกจากตารางเดิม ยกเว้น ส่วนของคีย์หลัก คงไว้ในตารางเดิม



## ตารางผลการเรียน

รหัสนักศึกษา	รหัสวิชา	เกรด	ชื่อวิชา
534267001	F01	A	การเขียนโปรแกรม
534267001	F02	B	การออกแบบฐานข้อมูล
534267002	F01	D	การเขียนโปรแกรม
534267002	F02	A	การออกแบบฐานข้อมูล
534267003	F01	A	การเขียนโปรแกรม
534267003	F02	C	การออกแบบฐานข้อมูล

ตารางนี้เมื่อทำให้อยู่ใน  
รูป 2 NF จะได้ 2  
ตารางดังนี้

## ตารางผลการเรียน

รหัสนักศึกษา	รหัสวิชา	เกรด
534267001	F01	A
534267001	F02	B
534267002	F01	D
534267002	F02	A
534267003	F01	A
534267003	F02	C

## ตารางวิชา

รหัสวิชา	ชื่อวิชา
F01	การเขียนโปรแกรม
F02	การออกแบบฐานข้อมูล

# ให้นักศึกษานอมนัลไลซ์ตารางนี้ให้อยู่ในรูปแบบ 2NF

## ตารางผลการอบรม

<u>รหัสผู้เข้า</u> <u>อบรม</u>	<u>รหัสครอส</u> <u>อบรม</u>	ชื่อผู้เข้าอบรม	ชื่อครอสอบรม	ผลการทดสอบ
0001	TR01	นายเอ ใจดี	การซ่อมไฟฟ้า	ผ่าน
0001	TR05	นายเอ ใจดี	การซ่อมตู้เย็น	ผ่าน
0002	TR03	นางบี ใจกล้า	การทำอาหาร	ไม่ผ่าน
0002	TR09	นางบี ใจกล้า	การเลี้ยงเด็ก	ผ่าน
0003	TR01	นายรวย มีเงิน	การซ่อมไฟฟ้า	ผ่าน
0003	TR05	นายรวย มีเงิน	การซ่อมตู้เย็น	ไม่ผ่าน

## Third Normal Form (3NF)


- 1. Relation นั้นจะต้องมีคุณสมบัติ 2NF
- 2. ต้องไม่มีความสัมพันธ์ระหว่าง Non-key Attribute หรือ  
ไม่มี Transitive Dependency

สรุป : แอททริบิวต์ที่ไม่ใช่คีย์หลัก ต้องไม่ขึ้นต่อกันเอง


## Third Normal Form (3NF)

### □ วิธีขจัดปัญหา

1. สร้างตารางเพิ่ม
2. นำแอททริบิวต์ที่มีปัญหามาใส่ในตารางใหม่
3. กำหนดคีย์หลัก
4. แอททริบิวต์ที่ย้ายจากตารางเดิมไปใส่ในตารางใหม่ให้ตัดออกจากตารางเดิม
5. นำคีย์หลักในข้อ 3 ไปใส่ในตารางเดิม



<u>รหัสพนักงาน</u>	ชื่อสกุล	รหัสแผนก	ชื่อแผนก	เงินเดือน
P001	นพเกศ แก้วใส	A001	บัญชี	25000
P002	วารุณี รวดเร็ว	F001	การเงิน	30000



- ❑ คีย์หลักของตารางนี้คือ รหัสพนักงาน
- ❑ จากตารางยังมีฟังก์ชันการขึ้นต่อกันแบบ Transitive Dependency อยู่ คือ
- ❑ รหัสแผนก ซึ่งไม่ใช่คีย์หลักของตาราง แต่สามารถระบุค่า ชื่อแผนก ได้ คือ ถ้ารู้รหัสแผนก ก็จะรู้ชื่อแผนก

จากตารางข้างบน ทำให้อยู่ในรูป 3 NF จะได้ 2 ตารางข้างล่างนี้

<u>รหัสพนักงาน</u>	ชื่อสกุล	เงินเดือน	รหัสแผนก
P001	นพเกศ แก้วใส	25000	A001
P002	วารุณี รวดเร็ว	30000	F001

<u>รหัสแผนก</u>	ชื่อแผนก
A001	บัญชี
F001	การเงิน

## สรุป Normalization

- 1NF ทุกแอตทริบิวต์ในแต่ละแถวมีค่าของข้อมูลเพียงค่าเดียว
- 2NF รีเลชันนั้นต้องไม่มีความสัมพันธ์ระหว่างแอตทริบิวต์แบบบางส่วน (แอตทริบิวต์ทุกตัวต้องขึ้นกับคีย์หลักทุกตัว ไม่ขึ้นอยู่กับตัวใดตัวหนึ่ง)
- 3NF ทุกแอตทริบิวต์ที่ไม่ใช่คีย์หลักไม่มีคุณสมบัติในการกำหนดค่าของแอตทริบิวต์อื่น

## แบบฝึกหัด

ให้นักศึกษาแปลงตารางต่อไปนี้ให้อยู่ในรูป NF1-NF3 โดยละเอียด

### การสั่งซื้อสินค้า

เลขที่ใบสั่งซื้อ	วันที่ซื้อ	รหัสลูกค้า	ชื่อผู้สั่ง	รหัสสินค้า	ชื่อสินค้า	จำนวนที่ซื้อ	ราคาต่อหน่วย
OR001	02/09/2552	C001	เดวิด	AB12	ตู้เย็น	4	4000
				CD01	พัดลม	3	2000
				PC09	แอร์	4	6000
OR002	02/09/2552	C005	ไมเคิล	TP01	เตาอบ	3	3000
				CD01	พัดลม	2	2000