

File-based System

ระบบไฟล์มีข้อดีในด้านราคาน้ำหนักเบา จ่ายไฟฟ้า → Database System

Problem of File Based System

• ช่องทางการเข้าถึงสอง Separation and Isolation data

เพิ่มพื้นที่บันทึกต้องหัวใจยังคงเวลา

• Duplication of Data นำเข้าข้อมูลเดียวกัน

→ Data inconsistency ข้อมูลไม่สอดคล้องกัน

ex. บันทึก GPA นำเข้ามาสองครั้ง รหัส GPA
ก็จะมีสองตัว

Data anomalies ข้อผิดพลาด

- Modification anomalies : แก้ไขข้อมูลเดิมๆ
- Insertion
- Deletion

• Incompatible file formats ไฟล์บันทึกต้องไปต่อ

ต้องแปลงรูปแบบไฟล์ใหม่ ไม่สามารถใช้ร่วมกันได้

• Fixed Queries/proliferation of application programs

Programmer ต้องเขียนโปรแกรมตรวจสอบ

ข้อมูลที่ใช้งานบบ

Database Architecture

Abstract View ดูข้อมูล Store, Manage

= easy to use system and efficiency =

ex. Student Data Abstract View (User view)

- High Level > studentID, Name, Major
studentID, Kavikara-Prathum, ComSci

- Intermediate Level > structure

ID → char(3)

Firstname → varchar(20)

Lastname → varchar(20)

Major → varchar(20)

- Low Level > พื้นฐาน เช่น Bit, Byte, Block.

THREE LEVEL ARCHITECTURE ANSI

External Level (ม่อนต์)

Abstract View

Data from databases

studentID, Kavikara-Prathum, ComSci

Conceptual Level (โครงสร้าง)

conceptual schema show in Data Model

ex. เก็บไว้ในรูปแบบที่ต้องการ hardware

Internal Level (ภายใน)

เก็บไว้ในรูปแบบที่สะดวก

File organization เก็บใน system

⇒ Fast & efficient to access data

ex. MySQL (Physical Level)



logical

Data Independence

ไม่ต้องเปลี่ยน data level แล้ว ยังไงก็ไม่ต้องเปลี่ยน

• Logical

• Physical



Multi-User DBMS Architecture

• Tele processing ดำเนินการทางไกล



Client Server Architecture

• Client Server + Work Station

• LAN Local Area Network

- Large amount of network traffic

- High cost of server

- Complexity because concurrency control



INTRODUCTION TO DATA BASE SYSTEM

CHAPTER 1

Data Base

→ End user data (Raw facts) ; ข้อมูลที่บันทึกต้องการ

→ Meta Data (Data about data) ; ข้อมูลที่บันทึกต้องการ
Type Data → Non-numeric Data type : ตัวอักษร หรือ string char

ID, Name, Professor

→ Numeric Data type : ตัวเลข เช่น int, double, float

6.00

Why Store Data IN Database?

+ Efficiency

+ Convenience

+ Reliability

+ Security

Data Base Characteristics

- Persistent data store in Database

ไม่หาย失หายอย่าง ex. Disk

ยกเว้น Disk Failure

- Shared can Multiclient ผู้ใช้งาน

สามารถเข้าถึง user ต่อไปได้

ให้ความยืดหยุ่น

- Interrelated data relate to

ทำให้ใช้efficiency

Identity: student, Professor, Course

Relate: สาระเรียนที่สอนในห้องเรียน

เชื่อมโยงกันอย่างแน่นหนา

Advantage of Data System

+ Program Data Dependency

- Unintended dependency Meta Data dentro program

- Inconsistent Data Structure, Characteristics

ที่ต้องการ program

Program > Data Dependent, Inconsistent:

+ Minimal Data Redundancy

การซ้ำของข้อมูล Data

+ Improved Data Consistency

รักษาความถูกต้องของข้อมูลที่มีอยู่

Evolution OF Database

Gen 1 1960s File

File Structure

Gen 2 1970s Network Navigation

Network and hierarchies of related records

Gen 3 1980s Relation

Non-procedural languages

Gen 4 1990s Object

Multi-media, distributed processing

2000 and Beyond

2000 & Beyond

- relational database ฐานข้อมูลที่ทำงานลับซึ่งกัน

- Not Only SQL - MongoDB, Apache Cassandra, Hadoop

DBMS

Data Base Management System

Software for manage Data Base System

DBMS คืออะไร สำคัญneed อะไร ประโยชน์

→ Data base Definition คือ Data Base ต้อง Entity + Relationship

→ Data base Access Store, Update, Query

→ Transaction Processing ไม่สามารถ transaction

↓ ต้องบันทึกใน transaction

↓ ประสบความเสี่ยง transaction

↓ Transaction: ต้องรักษาความถูกต้องของข้อมูลที่มีอยู่

Store Data in DataBase

Goals

Retrieve Data from DataBase

- convenience - efficiency - consistency

include the ability to store and retrieve

- Data consistency across multiple data

- System availability - scalability - adaptability

File Based System

↓
ກາງຈະບໍາໄວ໌
ປີລັບປິດເປົ້າບຸນຫຼາຍໆຂອງສູລະເການ



Data Base System *

ຳ ACID Atomicity Consistency Isolation Durability

ເພື່ອຄວບຄຸມ Data Base ອັນ Transaction ມີ User

Atomicity: ຂອງບັນດາມີຄວາມຕິດພາດ Transaction ສີໂລດືບ → completed
→ Cancelled All

Consistency: Data ທີ່ເປັນຂອດ Data Base ທີ່ສູງຄູກເຫັນອຽນກົນເປັນໃນການເງື່ອນໄໝ
ທີ່ຖືກຕໍ່າງໆໄດ້

Isolation: ທັນສົກມີຄວາມຕິດພາດ 1 Transaction ຮະຫວັງໄນ້ການກາບ Transaction ທີ່ໆ

Durability: Transaction ສີໂລດືບ ຕົວດູກເຫັນທັກລົງ Data Base ກົບຜົວນ
ເປົ້າຮ່ວມມືນຂອງລົງໄວ້ເກີດປິດຢາຍເວັ້ນທີ່ ດັ່ງນີ້ແລ້ວໄຟ້ ex. ປັບກັບລົງ

ໃນ Hard disk

DataBase = Schema + Instance

ID	Name	Lastname	Address	Major	Schema
650710524	Kanisara	Prasitnui	Songkhla	CS	[]
650710077	Natthaphon	Ruengwarapit	Bangkok	CS	Instance

Data Model : Data + ຄວາມສ້າງໃໝ່ + ຄວາມຂອງຍຸ + ເງື່ອນໄໄລ & ດັບຕິປະກອບ

Relation Model: show tables & column name

Object-Based Model: ພິເສດວິຊາກາ Relation "OOP" ex. Entity-Relation Model or ER Model

Semi-Structured Model: XML (Extensible Markup Language) ດາວໂຫຼວດໂຫຼວດ ບໍາດັກ ພິເສດວິຊາ

SQL ສັບກັນທີ່ໂຄງສະຕິງ ໃຫ້ສົດຂອງອັນດີເກີດ

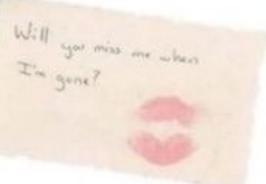
+ Data Definition Language "DDL": schema w; DataBase ms w; Table ex. create, delete, alter...

+ Data Manipulation Language "DML": form data in DB ex. insert, delete, update

+ Data Query Language "DQL": find data, query data ex. select, count, sum, min, max, average...

+ Data Control Language "DCL": control user right in DB ex. GRANT, REVOKE or transaction ex. commit, rollback

Ex. SQL select name, age, email from TB_member where age <



DATA BASE SYSTEM ARCHITECTURE

แบบจำลอง 3 Tier

• Presentation : ลักษณะเชื่อมต่อผู้ใช้ ติดต่อกับ User

UI: user Interface Text mode

Graphic User Interface

• Business Logic : main process user-data-data base

ต้องร่วมกันให้เข้ากันได้ในการทำงาน

soft ware by Programmer

DBMS ระบบจัดการฐานข้อมูล

• Database : ฐานข้อมูล, data ที่ต้องเก็บใน DB

เพื่อ Data Independence

ข้อมูลเชื่อมต่อที่เก็บไว้จะถูกแปลงให้ตรงกับชื่อในระบบ
กระบวนการนั้นๆ

1- Physical โครงสร้างทางกายภาพ schema

ไม่ระบุรายละเอียดของ hardware

ex. หน่วยความจำ Hard disk ของ Server

ไม่ระบุ database อยู่ใน Hard disk

2- Logical ผลลัพธ์ schema ที่ทางเรา

หันมาเก็บ schema ในรูป view

ex. หน้า เว็บ ตารางใน DB ไม่ระบุ hardware

soft ระบุมา หรือลักษณะต้องการกับ User

Architecture of Data Base

- หมายความ การประมวลผลของ Data Base System ที่process ต้องดำเนิน

- Centralize “ศูนย์รวม” ประมวลระบบฐานข้อมูลไว้ที่ Server ก็ได้เช่น

- การประมวลผลเบิกบาน Server, Client

- เป็นยุคสมัยดีปัจจุบันมีความหลากหลายมาก

1 Tier or Centralize

Presentation + Business Logic + Database

อยู่ใน Computer 1 เท่านั้น User ต้องต่อ Server

แล้วรับการประมวลผล

2 Tier or Client-Server

Presentation + Business Logic ; Client

Data Base : Server

ข้อดี = Data base ที่บันทึกไว้ใน client ให้สามารถเข้าถึงได้

Data base ใหญ่

ข้อเสีย = Client ต้องประมวลผลเองทั้งหมด

LAN ต้องเสียบสาย และคอมพิวเตอร์ต้องสูง

Server ต้องเป็นกลางที่client

ที่ใช้งานร่วมกันแบบ → ล่ม, รอดูงาน

3 Tier with Data Independence

Presentation - Client

Business logic - Application Server

“Middle Tier” ต้องต่อ software ประยุกต์ for ประมวลผล

Database - Database Server

ต้องต่อ DBMS data base management system

ข้อดี = ประมวลผลที่พิเศษ ของ Client ต้องต่อ กับ User ได้รวดเร็ว

. Middle Tier 芮ิ่ง, ลดขนาดไฟล์ต่อตัว

. ลดปัญหาความคงทนของข้อมูล Data Integrity

ข้อเสีย = . ฝึกความซับซ้อน complex ในการติดต่อตัวต่อตัว
มากกว่า 2-Tier

n Tier or Multitier

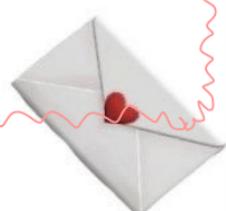
เพิ่มจำนวนตัว 3 tier ไม่ยังต้องมี tier มากกว่า

- ไม่ต้อง Business logic & Data Base ให้แยกอย่างเดียว

ให้ต้องหากำหนดของผลลัพธ์

- แยกการสืบค้นข้อมูลใน Database ด้วยคำสั่ง SQL

และการ Stored Procedure ให้คำสั่งที่เข้าไปอยู่



♥ Relational Data Model ♥

Relation → Table នៃពីរ

branchNo ①	street ②	City ③	Postcode ④
3003	143 ផ្លូវលេខ ៩	ភ្នំពេញ	10110

+ Tuple = row

+ Degree = column Attribute = 4

+ Cardinality = #rows tuple = 1

និងទាំង Table នឹង row/column នូវ Schema របស់យើង

Ex

name	major	salary
Phon	CS	50,000
Cin	CS	55,000
Mean	CS	53,000

Table មានចំណាំ

Table មានចំណាំ Schema → name, major, salary

If Degree = 3 , Cardinality = 3

Instance of 3 tuple
- Phon major CS salary 50000
- Cin major CS salary 55000
- Mean major CS salary 53000

Tuple = Attribute និងទាំងអស់របស់យើង

I. Domain: Attribute ត្រូវបានគិតឡើងជាប្រភព

ex. Attribute មិនត្រូវបានគិតឡើងជាប្រភព ដែលមិនមែនជាប្រភព

II. Atomic: Attribute ត្រូវបានគិតឡើងជាប្រភព ដូចជាការពារិនិយោគ

ex. phone-number & postnumber

III. Unorder: ការណែនាំ Tuple នៅក្នុងគ្មាន់នៃទូទៅ Data in Table
ត្រូវបានបញ្ជាក់ថា

IV. Null: នៅពេលបង្កើតឡើង Table នៅ

Data Type

Numeric: ចំនួនតាមរយៈ

Character: String: ស៊ូឡូ

Bit-String: Bit នៃ computer Binary

Boolean: TRUE / FALSE / UNKNOWN

Date: DATE / DAY / MONTH / YEAR

HOUR / MINUTE / SECOND

YY-MM-DD , HH-MM-SS

Ex: 2019,0003,...

Relation → relation(table) នៅពីរ

- Attribute នៃ relation នៅមិនចំនួន + នៅទាំង

- ត្រូវបានគិតឡើង

- គ្មាន់ cell នៃប្រអប់ atomic មួយគ្នា នៅទាំង

- Tuple នៃ Relation ត្រូវបានគិតឡើង

- សំណើនាំ attribute នៅក្នុងទូទៅនៃទូទៅ នៅទាំង

KEY : នូវចំណាំនៃទូទៅនៃទូទៅនៃទូទៅ

stdID	stdName
710077	Mr. Phon
710524	Ms. Cin
710572	Ms. Meen

* Super Key: Uniquely identifies

stdID	stdName	stdMajor
5907010	phon	ComputerEng
5907012	cin	ComputerEng
5907018	meen	ComputerEng

* Candidate Key: superkey នៃទូទៅ or attribute នៃទូទៅនៃទូទៅ

Tuple ត្រូវបានគិតឡើង

STUDENT

stdID	stdName	stdMajor	stdSSN
5907010	phon	ComputerEng	1234567890123
5907012	cin	ComputerEng	1234567890124
5907018	meen	ComputerEng	1234567890125

Attribute → stdID, stdName, stdMajor, stdSSN

Tuple ត្រូវបានគិតឡើង (stdID, stdName) → stdID, stdSSN

PK Primary key: នូវ key នៃទូទៅដែលគិតឡើងនៅ Relation (Table)

លាងនៃ Candidate key

stdID → Primary Key នៃ stdID

stdSSN → Primary Key នៃ stdSSN

PK Alternate key: candidate key នៃទូទៅនៃទូទៅ នៃ stdSSN

candidate key → stdID, stdSSN

Primary key → stdID ... Alternate key → stdSSN

Primary key → stdSSN ... Alternate key → stdID

PK Foreign Key: key នៃទូទៅនៃទូទៅ Attribute នៃ Table នៃទូទៅនៃទូទៅ

នូវ key នៃទូទៅនៃទូទៅ នៃទូទៅនៃទូទៅ

STUDENT

stdID	stdName	genderCode
077	phon	1
524	cin	2
572	meen	2

GENDER

genderCode	gender
1	Male
2	Female

Foreign Key នៃ STUDENT នៃ gender Code

Integrity Rules ; នូវការស្វែងរកចំណាំ នៃទូទៅនៃទូទៅ

PK Entity Integrity: Primary Key នៃ Table ត្រូវបានគិតឡើង (unique)

ឬត្រូវ NULL នៅពេលបង្កើតឡើង

Relationship Integrity: Entity Integrity នៃទូទៅនៃទូទៅ

នូវតាមលក្ខណៈនៃ Relation

major	minGrade	maxGrade
ComputerEng	50	100
Math	60	100
Chemistry	60	100

minor	maxGrade
ComputerSci	100
Math	100

Major key → majorCode & minGrade

Minor key → minorCode & maxGrade

PK Referential Integrity: Foreign key នៃ relation_2 ត្រូវបានគិតឡើង Primary key នៃ relation_2

ឬត្រូវ NULL



♥ Relational Data Model ♥ (mid)

Delete & Update Action for Referenced Rows

- delete / edit data in Relation_1, និងការលើកការពី

Relation_2 ដែលបានស្ថិតនៅ Value

ឬ foreign key

- ត្រូវជាលើក / edit because??

4. **Restrict:** និយាយនៃ value នៃទុក

Primary key នៃ table_1 គឺត្រូវ

នៅ table_2

PKID	attribute	valueCode
500010	university	1
500012	university	2
500018	university	3

នៅលើកការពី "value"

→ **Cascade:** បញ្ចូលការពី បានប្រើបាន Primary key នៃ table_1 នៅលើកការពី foreign key នៃ table_2

→ **Nullify:** និយាយ foreign key នៃ relation_2 ត្រូវ NULL នៅតុកការពី primary key នៃ relation_1 ត្រូវ

→ **Default:** del / edit និយាយ primary key នៃ table_1 នៅលើកការពី foreign key នៃ table_2

NOTE:

Relational Data Model និងការរួមចំណាំ

entityTable

Attribute "Column" Tuple "row"

- At Domain (គុណភាព) និងការប្រើបានក្នុងទឹក Data
- Atomic និងការពី attribute និងពីរ
- Unorder និងការប្រើបានក្នុងទឹក relation - order
- Null និងការប្រើបានក្នុង table

Data Type និងការរាយការណ៍ Data

Primary Key for attribute និងការរាយការណ៍, តម្លៃការងារ (row tuple) និងការ

- Super Key តើ column / ឱ្យ column សារុបតែងត្រូវ unique និង row (tuple)
- Candidate key តើរាយការណ៍ត្រូវត្រូវ
- Primary key និងការណ៍ Candidate Key តើត្រូវ null
- Foreign key តើរាយការណ៍ត្រូវ Primary key នៃទឹក, table

Relational Algebra

1. Selection "σ"

$\sigma_{\text{Salary} < 35000}$ (Teacher)

select salary from Teacher where salary < 35000;

2. Projection "Π" និង Π : column

$\Pi_{\text{name}, \text{price}}$ (Product)

select name, price from Product;

3. Union "∪"

$\Pi_{\text{id}, \text{name}}(\text{Depositor}) \cup \Pi_{\text{id}, \text{name}}(\text{Borrower})$

4. Set Difference "Π - Π"

$\Pi_{\text{name}}(\text{Depositor}) - \Pi_{\text{name}}(\text{Borrower})$

5. Intersection "Π ∩ Π"

$\Pi_{\text{name}}(\text{Depositor}) \cap \Pi_{\text{name}}(\text{Borrower})$

6. Cartesian Product "×" cross

Depositor × Borrower

Join ; $\Pi_{\text{d.name}} = \Pi_{\text{b.name}}$ (Depositor X Borrowers)

7. Natural Join "× oo" ឬនៅក្នុង relation

និយាយនៃ foreign key នៃ T_1 និង Primary Key

នៃ T_2

Product on Category

use $\Pi_{\text{p.ID}, \text{p.name}, \text{p.price}, \text{p.cat_id}, \text{cat.name}}$ ($\Pi_{\text{p.cat_id}} = \Pi_{\text{cat.ID}}$)



Entity-Relationship Model

ER DIAGRAM

Data Base Design

+ Conceptual (โครงสร้าง) → ER Diagram

+ Logical (โครงสร้าง)

+ Physical (รูปแบบ)

Logical คือโครงสร้างที่ใช้ใน DBMS
物理的構造: ความต้องการของผู้ใช้งาน

+ Entity set

• Strong (Regular Entity)
ไม่ต้องมี PK

• Weak Entity
ต้องมี PK ใน Entity ด้วย

• Associative/composite
คู่ๆ Many to Many

+ Attribute

• simple

• composite

• Single value

• Multi Value

• Derived ได้มาจากการคำนวณ Attribute ที่มีอยู่

• Key ห้ามต้องเป็น PK/FK
not null!

Redundant Attribute บันทึกซ้ำ

+ Relationship

NOTE:

Entity - Relationship Model

“ER MODEL”

แบบจำลองความสัมพันธ์ที่สำคัญ

ให้สัมผัสกับองค์ประกอบ ER-D

2 แบบที่มี

- Chen: Entity flow Chart

- Crow's foot: ผู้เชื่อมต่อ

Entity : ตัวเรื่องราว
Attribute : ข้อมูลของ entity
Relationship

* Entity Type

• Strong Entity

• Weak Entity

* Attribute Type

• Simple (single component) ไม่มีอย่างใดอย่างหนึ่ง

• Composite ถ้า Attribute สองตัว ex. name <first> <last>

• Single value ex. สีเสื้อ: แดง ขาว

• Multivalue ex. นามสกุล

• Derived หลักฐานของ Attribute ใหม่ ex. Age วันเกิดของ Birth

ER DIAGRAM

Peter Chen กลางชื่อ Chen → Entity ตัวเรื่องราว, Relationship คำว่า

= Entity (Strong/Regular)

= Attribute

1 1 = 1:1 and

= Weak Entity

= Key Attribute

1 N = 1:N (Total)
(ทั้งหมด)

= Associative Entity
Composite

= Multi Value Attribute

1 1 = 1:1 And

= Relationship

= component Attribute

1 N = 1:N (Partial)
บางส่วน

= Identifying Relationship

= Derived Attribute

Crow's Foot

Entity
Attribute
Attribute
.
.

Entity
key
:
:
:

Entity	
Type	
key	:
Attribute	:
Relationship	:
Component	:
Composite	:
Derived	:

= one

entity คือ noun

= Many

attribute →

= Zero or One

relationship คำว่า

= One and Only One

= Zero or More

= One or More

= zero or One

= zero or More

Will you miss me when
I'm gone?



Cardinality Ratio

จำนวนส่วนของความสัมพันธ์

① One-to-One 1:1 PK entity → non PK entity 2
Entity set ตัวเรื่องราว ที่ไม่ใช่ entity set

② One to Many 1:N

③ Many to One N:1

④ Many to Many M:N

Participation

ผลของการที่ต้องเขียน Entity set สองตัว

1. Total ต้องเขียนลงใน Entity

2. Partial ต้องเขียนลงใน Entity อย่างน้อย 1

→ Mandatory (มีอยู่ใน Entity ต้องเขียนลงใน Entity ด้วย)

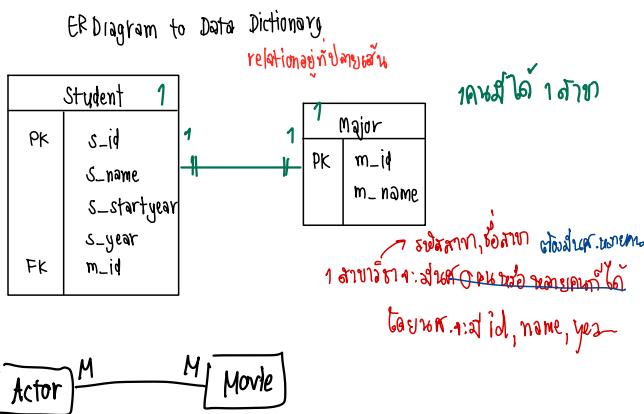
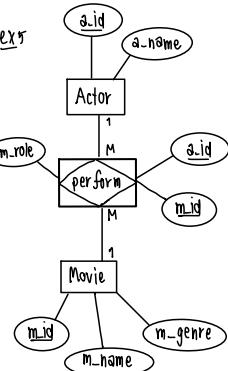
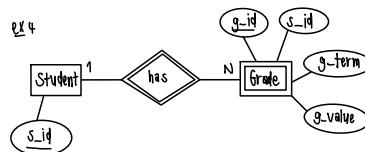
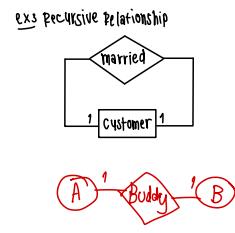
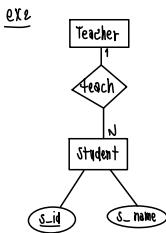
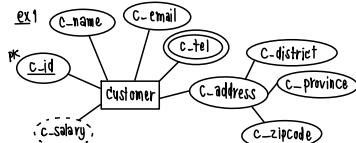
= One and Only One

= One and More

→ Optional (อาจจะต้องเขียนลงใน Entity บ้างบ้าง)

= zero or One

= zero or More

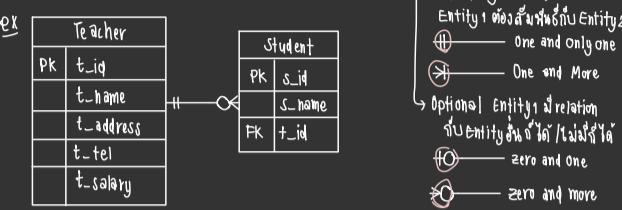


ER DIAGRAM (Crow's Foot)



→ = One
 → = Many
 +○ = Zero or One
 + = One and Only One
 ○ = Zero or More
 × = One or More

Crow's Foot
 Entity is noun
 Attribute →
 Relationship business
 1. Multiplicity of relationship
 Lone, many
 2. Minimum of defining Multiplicity



Relationship business

1. Multiplicity of relationship

Lone, many

2. Minimum of defining Multiplicity

Mandatory (요구)

Entity 1 의존 Entity 2

One and only one

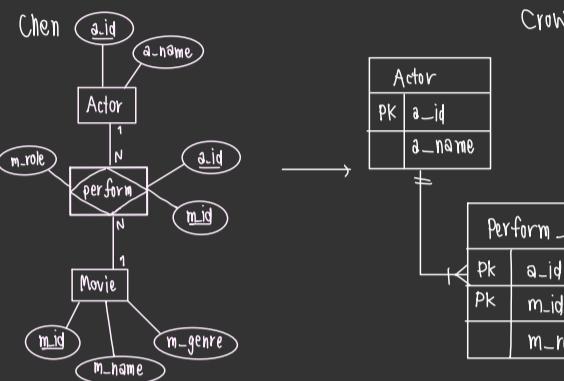
One and More

Optional Entity 1의 relation

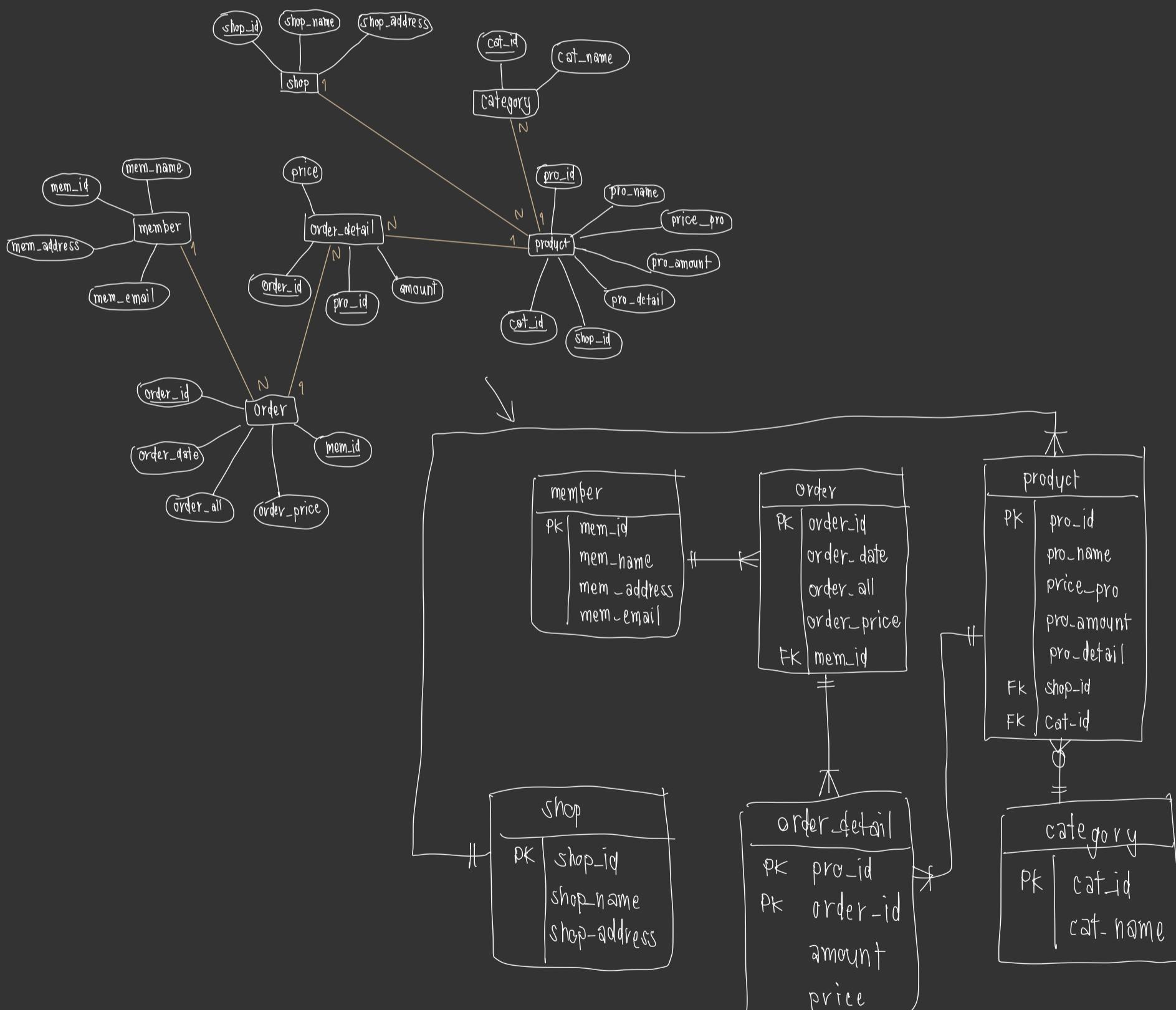
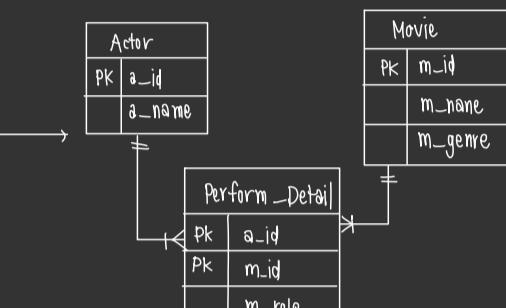
Entity 1 의존 Entity 2

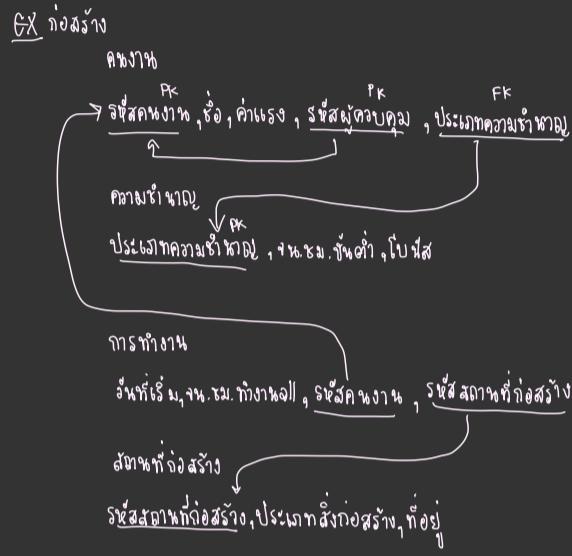
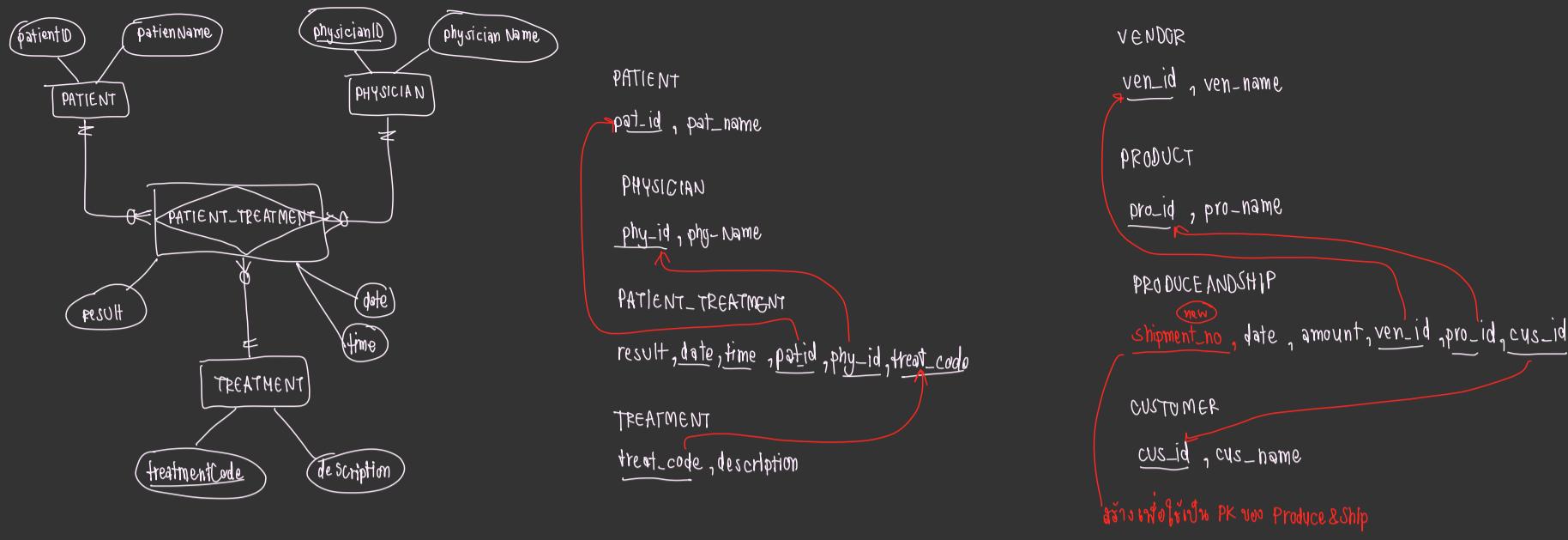
zero and one

zero and more

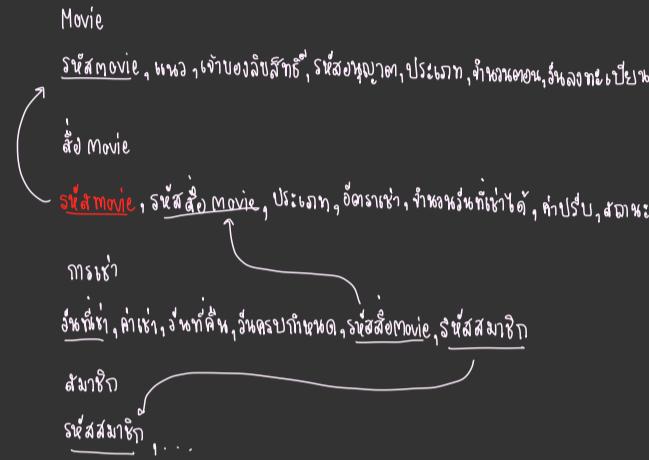


Crow

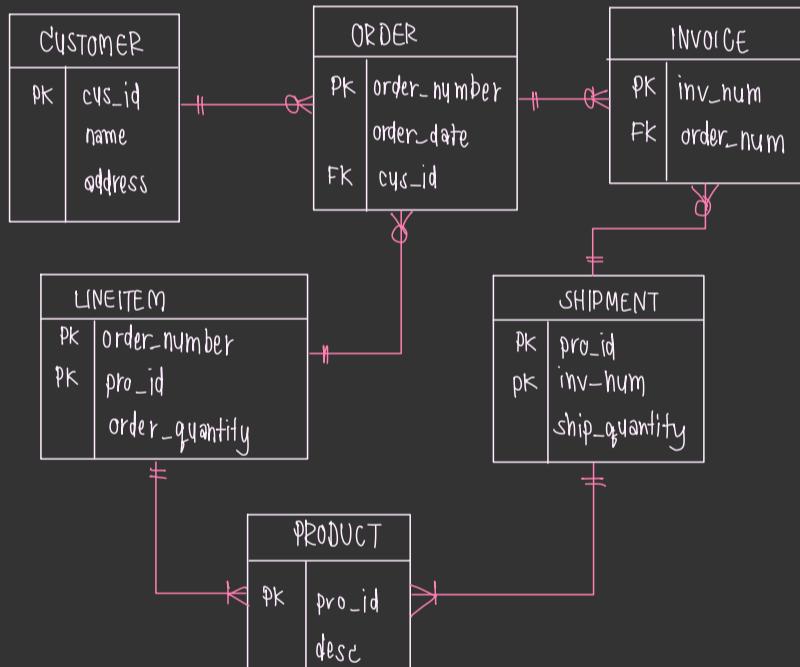
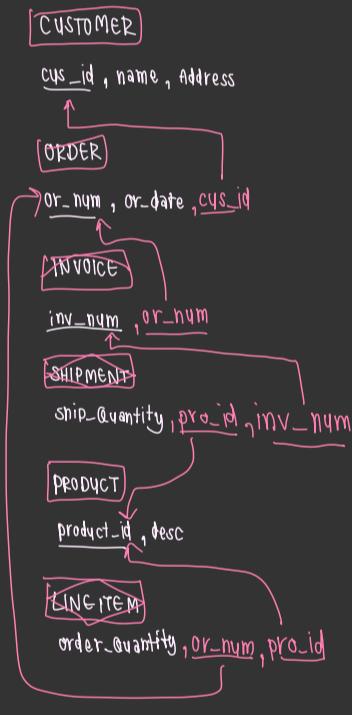
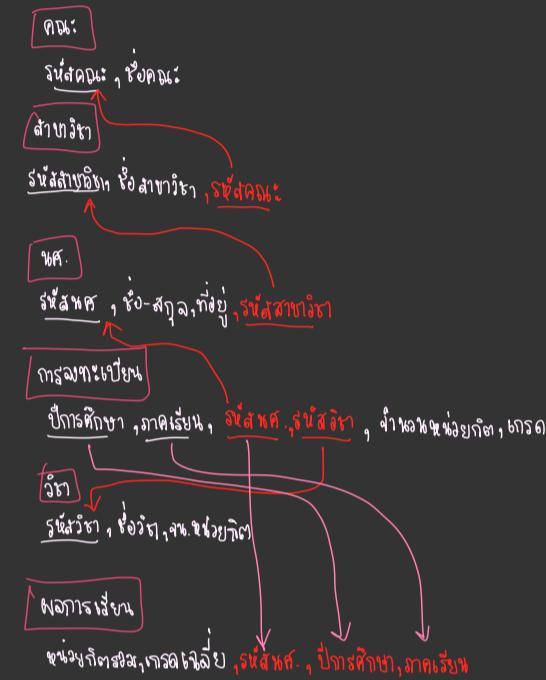


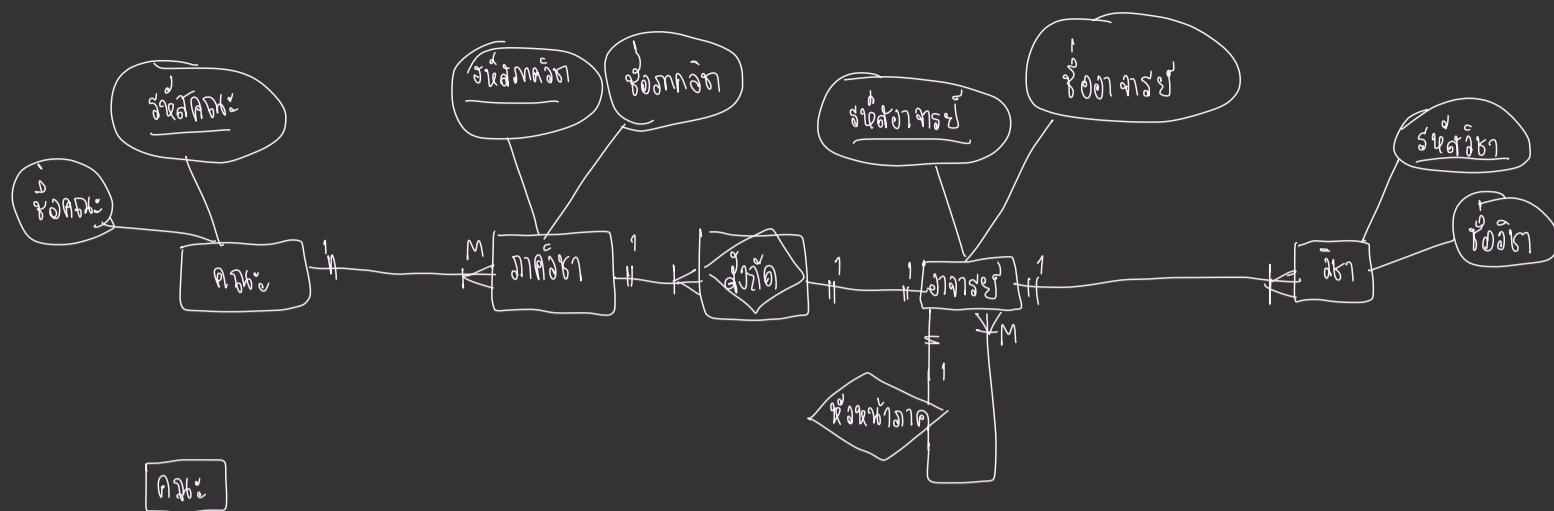


Ex Movie

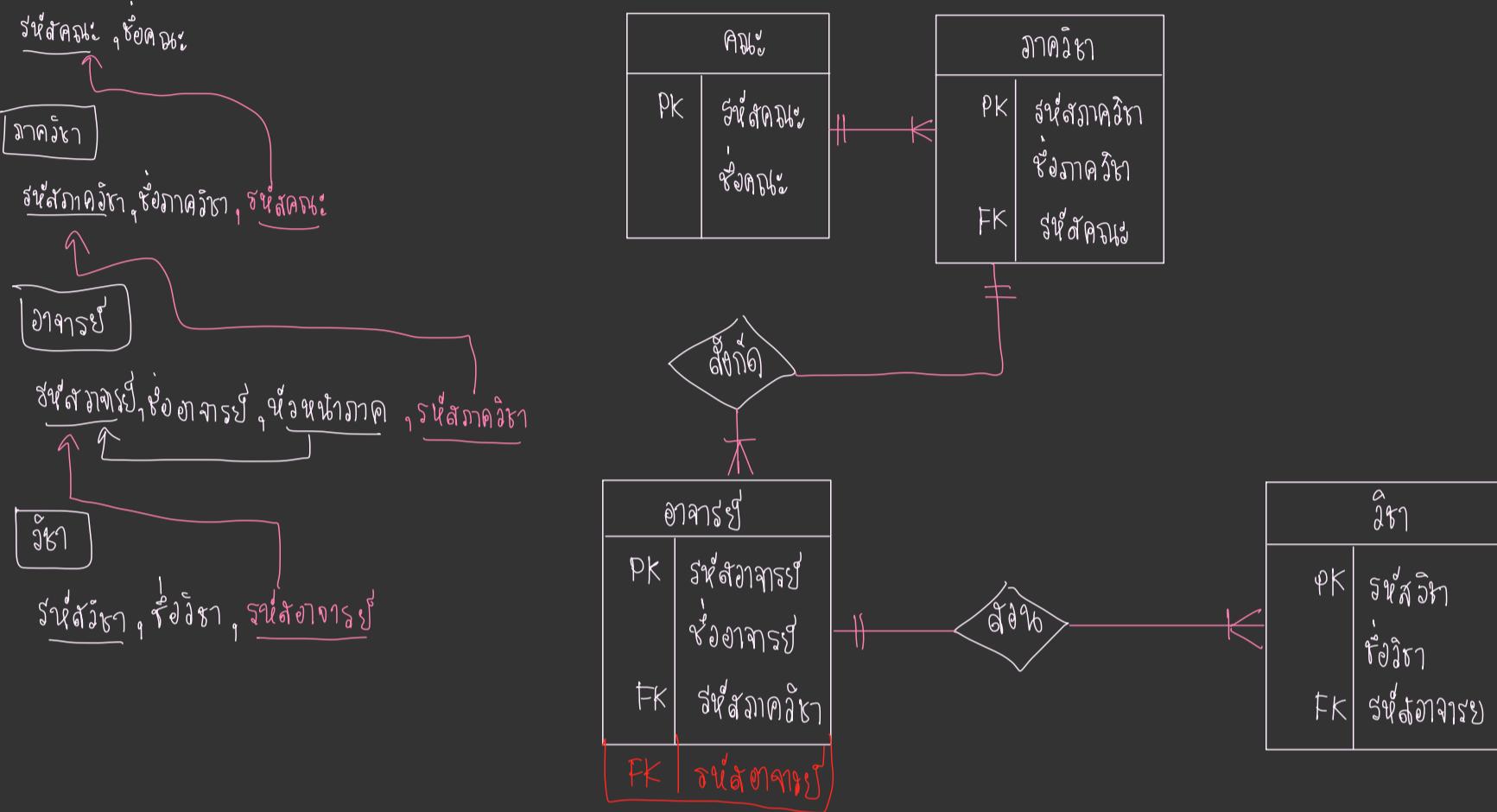


Ex การลงขายเพื่อนำเงิน



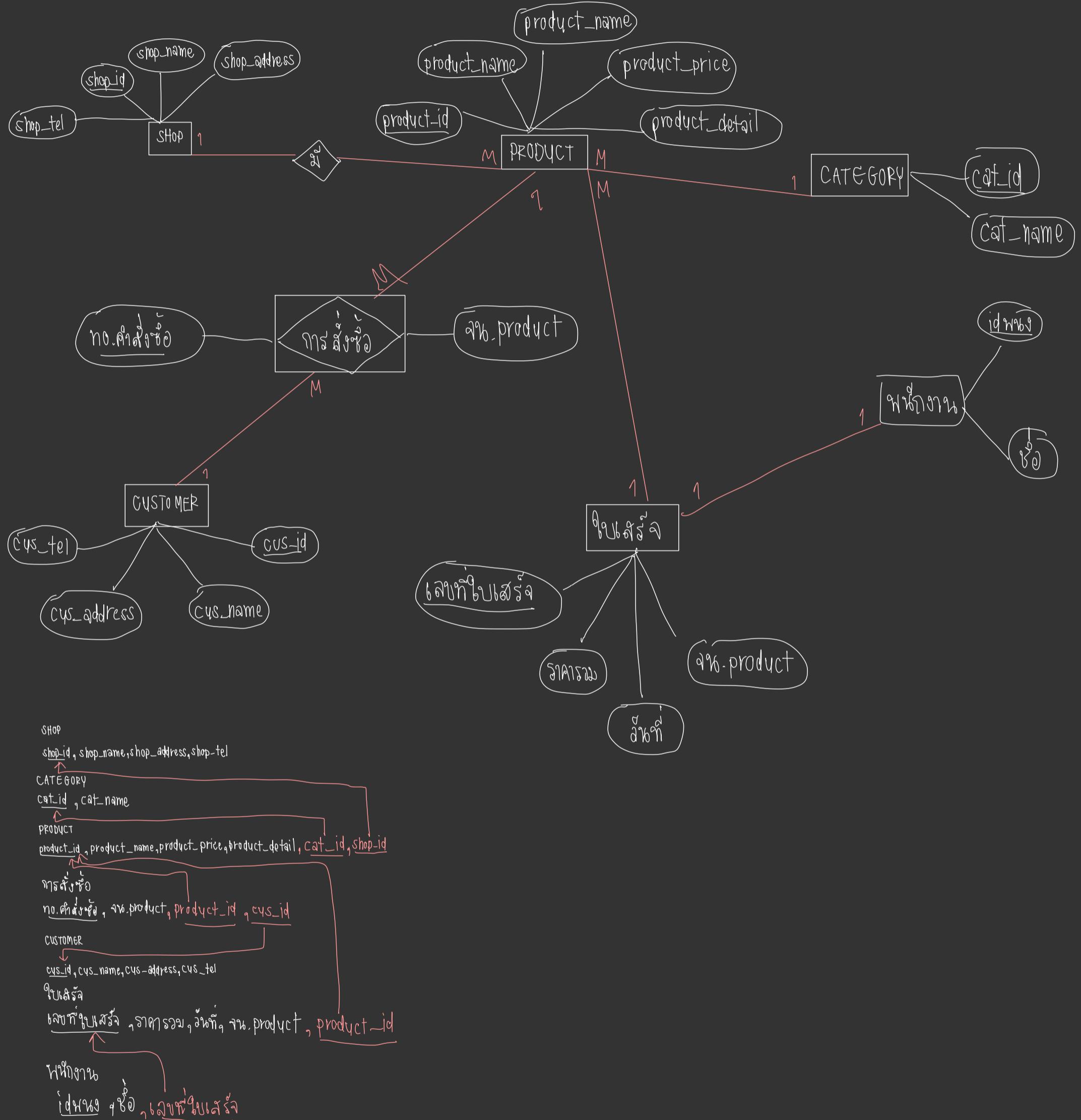


គម្រោង



- ให้เขียน ER-Diagram ของระบบการขายสินค้า
- ลูกค้าห้ามคนสามารถที่จะซื้อสินค้าได้หลายชิ้น
- ใบเสร็จรับเงิน 1 ใบมีสินค้าได้หลายชิ้น
- ใบใบเสร็จแต่ละใบจะมีพนักขายเพียงคนเดียวเท่านั้น
- สินค้าจะประเภทของตัวเอง

ร้านABC คอมพิวเตอร์ โดยที่



Database 7-91

