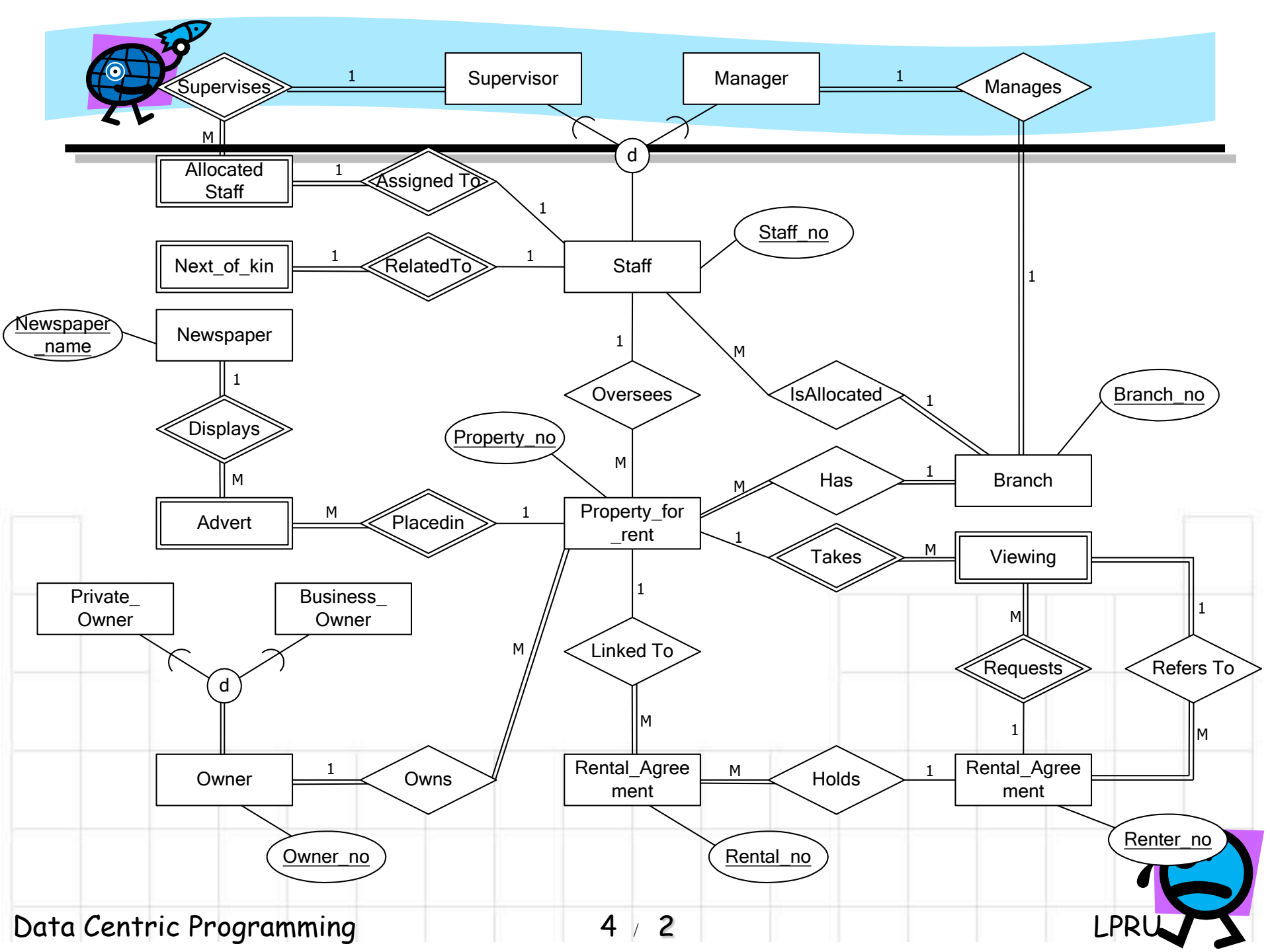
Entity-Relationship Modeling

- E-R Model เป็น high-level conceptual Data Model
- พัฒนาโดย Chen (1976) เพื่อช่วย DB design
- Conceptual Data model เป็น set ของ concept ที่ใช้อธิบาย structure DB และการ access DB
- จุดประสงค์ของการพัฒนา high-level DM
  - ช่วย support perception ของ user ต่อ data
  - ซ่อน technical aspect
  - สร้าง model ที่เป็นอิสระต่อ DBMS และ Hardware
- Concept ของ E-R model ประกอบด้วย
  - Entity types (object ใน real world)
  - Relationship Types
  - Attributes


Supervises
1
Supervisor
Manager
1
Manages
M
d
Allocated Staff
1
Assigned To
1
Staff_no
Next_of_kin
1
RelatedTo
1
Staff
1
Newspaper _name
Newspaper
1
Displays
M
1
Oversees
M
IsAllocated
1
Branch_no
Property_no
M
Has
1
Branch
Advert
M
Placedin
1
Property_for _rent
M
1
Takes
M
Viewing
Private_ Owner
Business_ Owner
d
M
1
Linked To
M
1
Requests
1
Refers To
M
Owner
1
Owns
Rental_Agree ment
M
Holds
1
Rental_Agree ment
Owner_no
Rental_no
Renter_no

Entity-Relationship Modeling

Entity Types

- เป็น object หรือ concept ซึ่งถูกกำหนดโดยองค์กรเป็น independent existence
- เป็น basic concept หนึ่งของ E-R Model
- เป็น represent set ของ object ซึ่งมีคุณสมบัติเหมือนกัน
- Exist อยู่ในระบบได้อย่างอิสระ ซึ่งอาจจะเป็น
  - real (physical) existence อาทิ พนักงาน,ลูกค้า,บ้านที่ดิน เป็นต้น
  - abstract existence (conceptual) อาทิ ประสบการณ์, ความต้องการ, แนวคิด เป็นต้น
- ไม่มี strict formal definition สำหรับใช้ในการ define entity ผู้ออกแบบอาจจะ identify ต่างกันได้
- Entity อาจจะแบ่งได้ 2 ประเภท คือ
  - Weak entity type
  - Strong entity type

Entity-Relationship Modeling

Entity Types

- Weak Entity Type เป็น entity ซึ่ง existence-dependent กับ entity อื่น ๆ เช่น next_of_Kin จะมีไม่ได้ถ้าไม่มี Staff (สามารถเรียกว่า Child, dependent, subordinate)

- Strong entity type เป็น entity ซึ่งไม่ existence-dependent กับ entity อื่น เช่น Staff, Branch, Renter (สามารถเรียกว่า parent, owner, dominant)

Diagrammatic Representation

Storage Entity - rectangle

Weak Entity - double lines rectangle

Entity-Relationship Modeling

Attributes

- เป็น property ของ entity type หรือ relationship type เช่น Branch อาจจะอธิบายได้ ด้วย
  - เลขที่สาขา -> Bno
  - ที่อยู่ -> Address
  - เบอร์โทรศัพท์ -> Tel_no
  - เบอร์โทรสาร -> Fax_no
- Attribute ของ entity จะมีค่าซึ่งใช้อธิบาย entity
- ซึ่งค่าของ attribute นี้จะใช้นำเสนอ main part ของข้อมูลใน Domain

Attribute Domain

- เป็น set ของ value ที่อาจจะถูกกำหนดไปยัง attribute ได้
- เป็น potential value ของ attribute เช่น
  - Tel_no ในเขต กทม. รหัสทางไกลต้องขึ้นต้นด้วย (02)
  - ทะเบียนรถใหม่ต้องนำหน้าด้วยอักษร กก- บอ
- attribute อาจจะ share domain กันได้
- domain จะต้องถูกกำหนดสำหรับแต่ละ attribute ใน E-R Model

Entity-Relationship Modeling

- **Attributes** : อาจจะแบ่งได้ 2 ประเภท
  - **Simple Attribute** (หรือ Atomic Attr.) เป็น attr. ที่ประกอบด้วย single component ซึ่งเป็น independent existence ไม่สามารถจะแบ่งแยกได้ เช่น sex, salary
  - **Composite Attribute** เป็น attr. ที่ประกอบด้วย multiple component ซึ่งแต่ละ component เป็น independent existence อาจจะแบ่งแยก attr. ให้เล็กลงได้ เช่น address, birthdate

    นอกจากนั้นยังสามารถดูตามค่าของ attr. ได้ ดังเช่น
  - **Single Valued Attribute** เป็น attr. ที่มี single value สำหรับ single entity เช่น Branch_address จะมี address เดียว เป็นต้น
  - **Multi-valued Attribute** เป็น attr. ที่มี multiple value สำหรับ single entity เช่น tel_no มีหลาย ๆ เบอร์สำหรับสาขาหนึ่ง ๆ
  - **Derived Attribute** เป็น attr. ที่นำเสนอค่าซึ่งสามารถจะ derive ได้จาก attr. อื่นที่สัมพันธ์กัน เช่น age, on of staff, profit

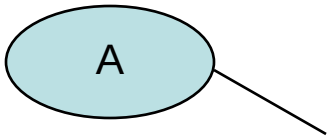
Entity-Relationship Modeling

Keys

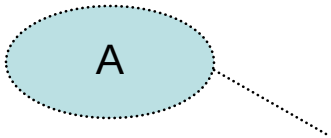
- **Key** = data item ที่ uniquely identify แต่ละ occurrence ของ entity
- **Candidate key** เป็น set ของ attr. ที่สามารถจะuniquely identify แต่ละ occurrence ของ entity type ได้ เช่น
  Branch มี candidate key คือ .....
  Staff มี candidate key คือ .....
- **Primary key** เป็น key หลักที่จะใช้ ซึ่งถูกเลือกมาจาก set ของ candidate key
- **Alternate key** เป็น candidate key ตัวอื่น ๆ ที่ไม่ได้ถูกเลือก
- **Composite key** เป็น candidate key ที่ประกอบขึ้นจาก attribute ตั้งแต่ 2 ตัวขึ้นไป
- หลักในการเลือก primary key
  - ขนาดของ attribute
  - จำนวนของ attribute ที่ต้องการ
  - ความ uniqueness ทั้งในปัจจุบันและอนาคต

Entity-Relationship Modeling

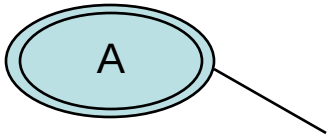
Diagrammatic Representation ของ Attribute



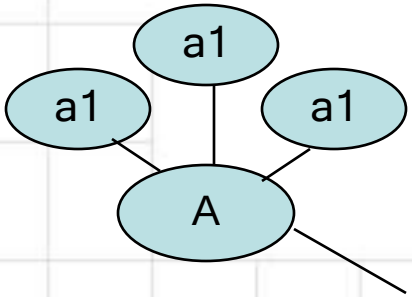
Attribute
แทนด้วย ellipse เชื่อมโยงกับ entity ด้วยเส้นต้นและบรรจุชื่อของ attribute name



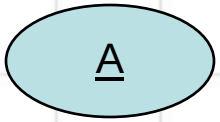
Derived Attribute
แทนด้วย dot ellipse เชื่อมโยงกับ entity ด้วย dot line



Multi- valued Attribute
แทนด้วย double lines ellipse



Composite Attribute
แทนด้วย ellipse โดยที่มี ellipse ที่แทนแต่ละ component แยกออกมาจากellipse ที่แทน composite Attr. นั้น ๆ



Primary key จะขีดเส้นใต้ attribute ที่ประกอบเป็น primary key ของ entity นั้น ๆ

Entity-Relationship Modeling

Relationship Types (R Types)

- **R types** เป็น association ที่มีความหมายระหว่าง entity types
- แต่ละ R type จะกำหนดชื่อที่ใช้บอก function ของ R type นั้น ๆ
- **Relationship** จะเป็นความสัมพันธ์ระหว่าง entity ซึ่งจะประกอบด้วย entity ตัวหนึ่งที่อยู่ในแต่ละ Entity type อาจเรียกว่า **Relationship occurrence** หรือ **Relationship instance**

- **Semantic Net**
  เป็น object-level diagram ซึ่งใช้สัญลักษณ์ดังนี้ ellipse ที่แทนแต่ละ component แยกออกมาจาก ellipse ที่แทน composite Attr. นั้น ๆ

  - แทน entity

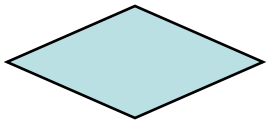
  ◈ แทน relationship

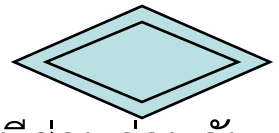
  ——— แทน association ระหว่าง instance

Entity-Relationship Modeling

Relationship Types (cont.)

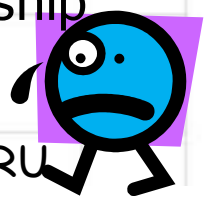
- สำหรับ **E-R Diagram** relationship จะแทนด้วย สี่เหลี่ยมขนมเปียกปูน (diamond) ซึ่ง diagrammatic representation

  Storage Relationship - diamond 

  Weak Relationship - double lines diamond 

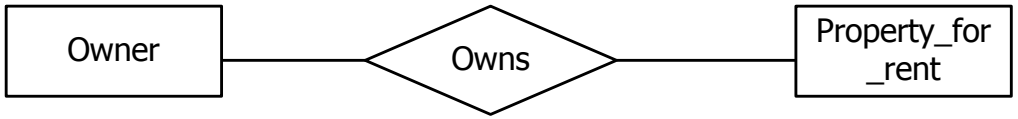
- **Degree** ของ **Relationship** เท่ากับจำนวน entity ที่มีส่วนร่วมกับ Relationship นั้น ๆ
  - **Binary Relationship** = R ที่มี degree 2
  - **Ternary Relationship** = R ที่มี degree 3
  - **Quaternary Relationship** = R ที่มี degree 4

- **Recursive Relationship** คือ relationship ที่ entity เดียวกันที่มีส่วนร่วมมากกว่า 1 ครั้งด้วย role ที่แตกต่างกัน บางครั้งเรียกว่า Unary Relationship

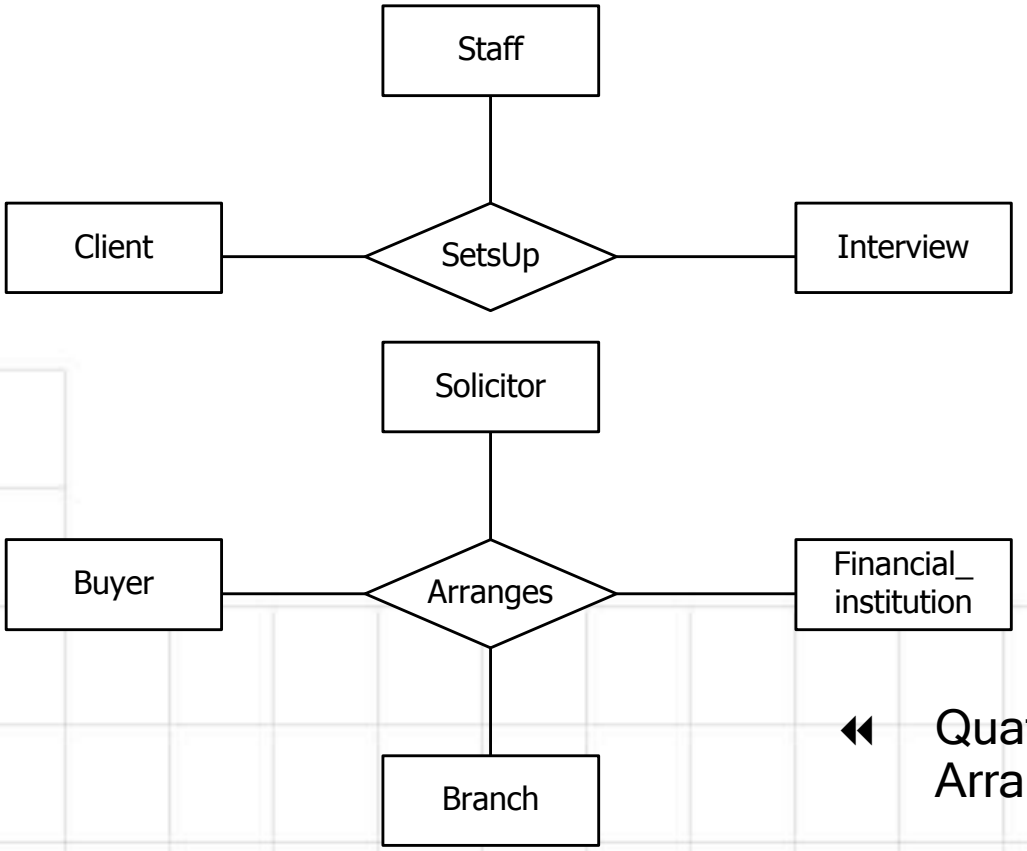


Entity-Relationship Modeling

Relationship Types (cont.)



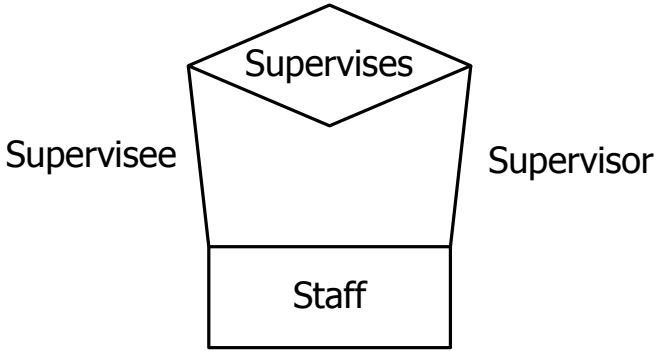
- Binary Relationship of owns



- Ternary Relationship of SetsUp
- Quaternary Relationship of Arranges

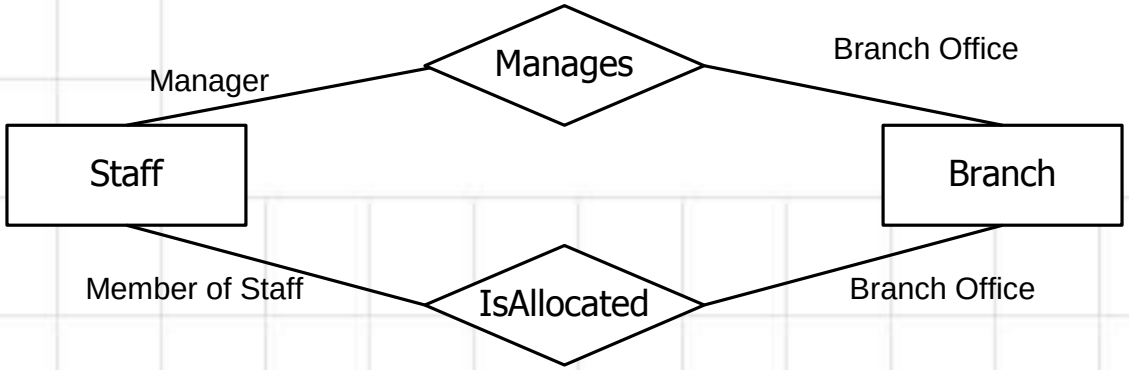
Entity-Relationship Modeling

Relationship Types (cont.)



- Example of Recursive Relationship of Supervises

ซึ่งบาง Entity อาจจะสัมพันธ์กันได้มากกว่า 1 role ดังนั้นในกรณีนี้ต้องใช้ role name กำกับด้วย



- Example of entities associated through two distinct relationships called Manages and IsAllocated with role name

Entity-Relationship Modeling

Structural Constraints

Constraint

- เงื่อนไขหรือข้อจำกัดต่าง ๆ ที่อาจจะกำหนดบน entities ที่ participate ใน relationship
- ข้อจำกัดต่าง ๆ บน relationship ควรจะสะท้อนให้เห็นใน real world
- มีอยู่ 2 ลักษณะใหญ่ ๆ คือ
  - Cardinality Constraints
  - Participation Constraints

◀◀ **Cardinality Constraints** เป็นข้อจำกัดเกี่ยวกับจำนวน relationships สำหรับแต่ละ Entity ที่ participate

Cardinality Ratio เป็นจำนวน relationship ที่เป็นไปได้ สำหรับแต่ละ Entity ที่ participate ซึ่งสำหรับ Binary Relationship จะได้ cardinality ratio ดังนี้

- One-to-One (1 : 1)
- One-to-Many (1 : M)
- Many-to-Many (N : M)

Entity-Relationship Modeling

- **Structural Constraints**
  - **Cardinality Constraints** (cont.)
    - Cardinality ratio ระหว่าง Entity เป็นฟังก์ชันของ policies ที่กำหนดขึ้นมาโดยองค์กร
    - rule ที่ใช้กำหนด cardinality เรียกว่า business rules
    - แต่บาง business rules ก็ไม่สามารถ represent ใน E-R Model ได้ เช่นกำหนดตามความต้องการที่พนักงานจะได้รับวันหยุดเพิ่ม สำหรับทุก ๆ ปีที่ทำงานกับองค์กรนั้นๆ

One-to-One Relationship (C. ratio = 1 : 1)

- single set ใน entity หนึ่งจะ associate กับ single set ในอีก entity หนึ่ง

One-to-Many Relationship (C. ratio = 1 : M)

- single set ใน entity หนึ่งจะ associate กับ หลาย ๆ set ใน entityหนึ่ง

Entity-Relationship Modeling

- **Structural Constraints**
  - **Cardinality Constraints** (cont.)

    Many-to-Many Relationship (C. ratio = N : M)
    - single set ใน entity A จะ associate กับ หลาย ๆ set ใน entity B (1:M)
    - ในทางกลับกัน single set ใน entity B จะ associate กับ หลาย ๆ set ใน entity A (N:1)

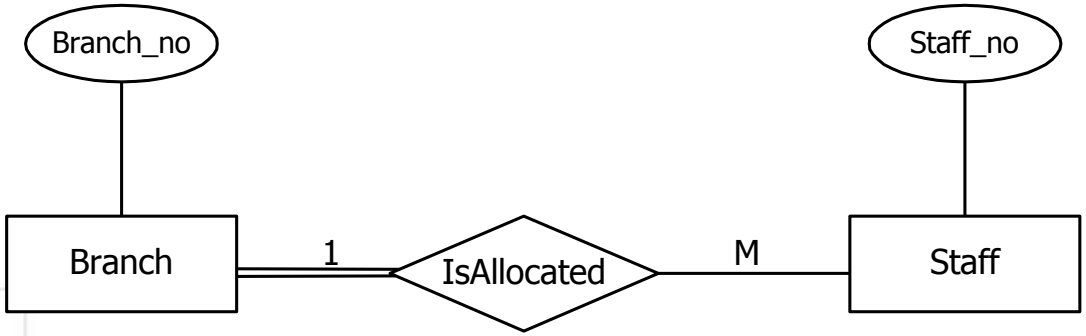
  - **Participation Constraints**
    - จะ determine the existence ของ entity ซึ่งขึ้นอยู่กับการที่มัน associate กับ entity ตัวอื่น ๆ ผ่านทาง relationship หรือไม่
    - มี participate constraints อยู่ 2 ประเภท ได้แก่

      Total : existence ของ entity บังคับว่าทุก entity ใน entity set จะมีการ associate กับ entity ใด entity หนึ่งด้วย particular relationship (=Mandatory)

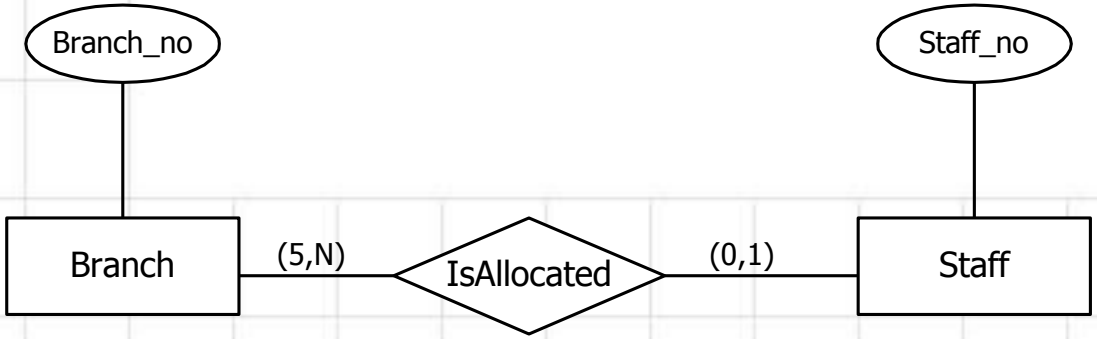
      Partial : สำหรับ participate ที่ไม่เป็น total (=Optional)

Structural Constraints

Participation Constraints (cont.)



- แสดง Branch เป็น Total Participate เพราะทุก Branch จะมีการ Allocate Staff เสมอ



- แสดง Staff เป็น Partial Participate เพราะ Staff ในบางครั้งอาจจะไม่ถูกกำหนดให้ไปที่ Branch ใด ๆ เลย
- Notation Min,Max

Entity-Relationship Modeling

- **Problems with E-R Models**
  - ปัญหาที่อาจจะเกิดขึ้นเมื่อกำลัง design conceptual data model คือ Connection Traps
  - Connection Traps เกิดจากการ interpret ที่ผิดพลาดของความหมายใน relationship แต่ละประเภท
  - Connection Traps มี 2 ประเภท คือ Fun, Chasm

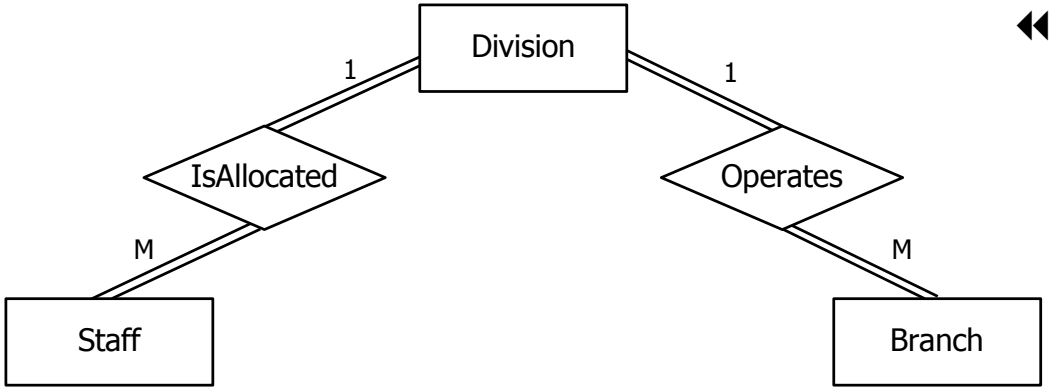
1. Fan Trap

- เป็น trap ที่เกิดขึ้น เมื่อ model ที่แสดง relationship นั้น ambiguous
- Fun trap จะเกิดขึ้นเมื่อ 1 : M relationship 2 ตัวหรือมากกว่า fan out ออกมาจาก Entity เดียวกัน

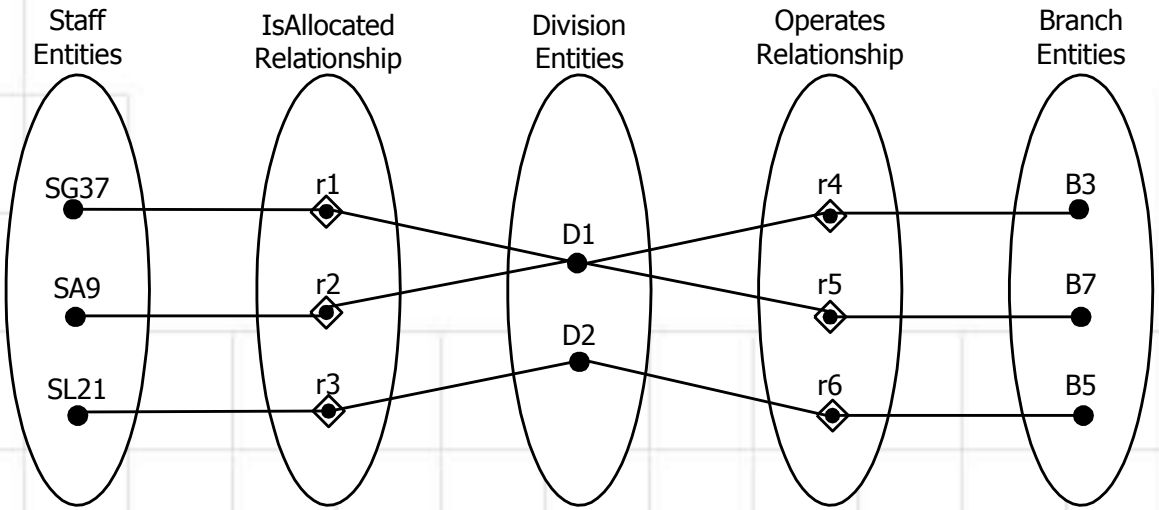
2. Chasm Trap

- เกิดขึ้นเมื่อ model ที่สร้างขึ้นมี relationship ระหว่าง entity type แต่ว่า pathway ไม่ exist ระหว่าง entity occurrence

Fan Trap : Example



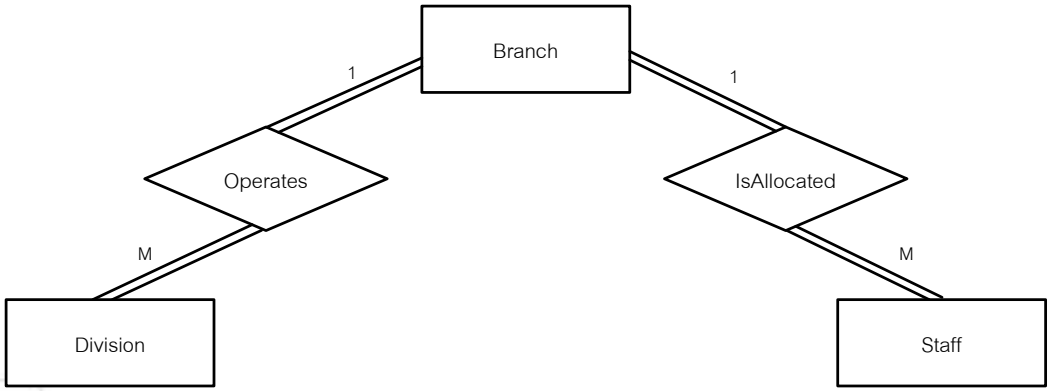
- แสดง 1: M relationship 2 ตัว fan out ออกจาก Division Entity ทั้งคู่

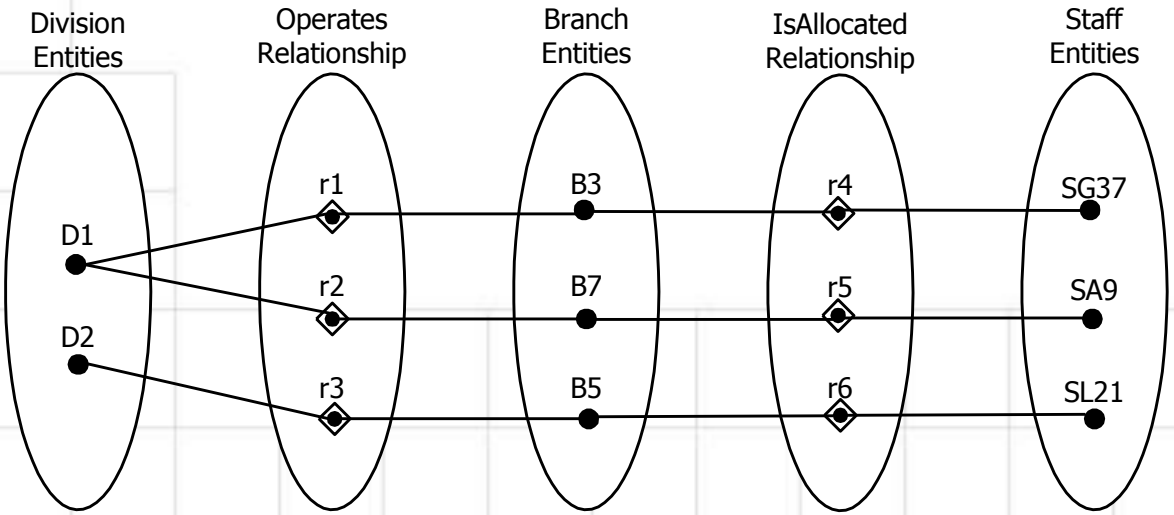


- เมื่อนำเสนอด้วย semantic net ที่ occurrence level จะพบ ambiguous เกิดขึ้น เช่น ไม่สามารถบอกได้ชัดเจนว่า staff คนใด ทำงานที่Branch ไหน

Entity-Relationship Modeling

- **Fan Trap :** เราสามารถแก้ปัญหา โดยการ restructure model ใหม่โดยอย่าให้ 1:M relationship fun out ออกจากการ entity เดียวกัน

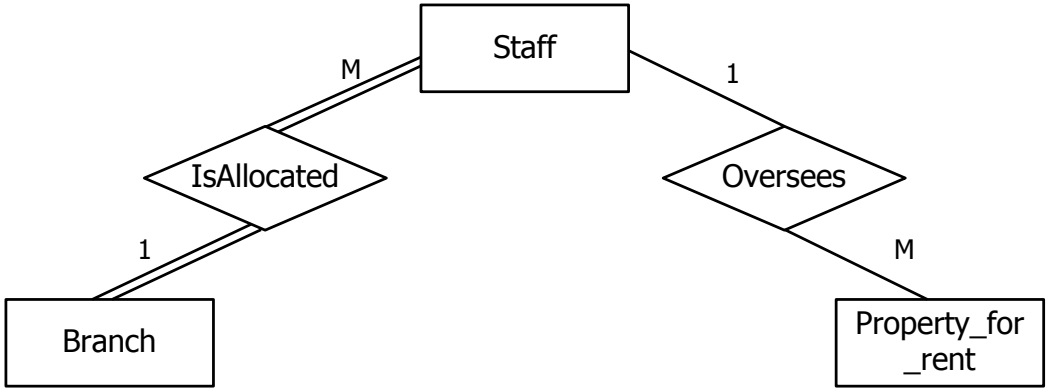




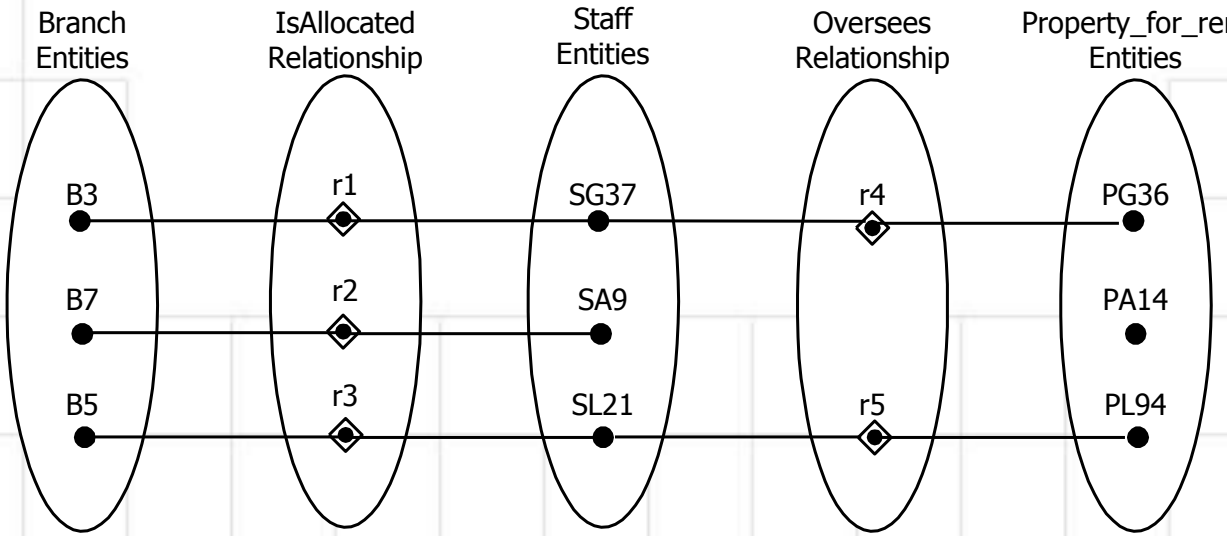
- ตัวอย่างการแก้ปัญหา โดยจะไม่เกิดปัญหา ambiguous อีกต่อไป

Entity-Relationship Modeling

Chasm Trap : Example



- แสดง E-R diagram และ semantic net ซึ่งไม่สามารถตอบได้ว่า properties ใด available ที่แต่ละ branch?



Entity-Relationship Modeling

- **Chasm Trap :** เพื่อขจัด chasm trap จะต้องเพิ่ม missing relationship ดังนี้

แสดง E-R diagram
และ semantic net ซึ่ง
เพิ่ม relationship Has
เพื่อขจัด chasm trap