อ.ดร.สุรัตน์ โคอินทรางกูร

Entity-Relationship Diagram

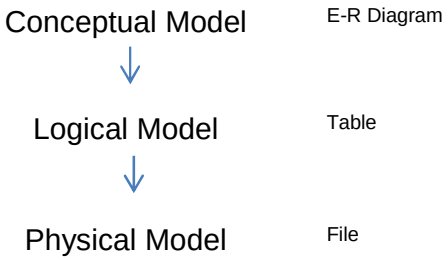
Entity Relationship Diagram หรือ E-R Diagram เป็น diagram เพื่อใช้ model ข้อมูลเพื่อให้ทราบว่าระบบนั้นๆ มีข้อมูลอะไรอยู่บ้าง diagram ดังกล่าวจะใช้ในการออกแบบฐานข้อมูล สิ่งที่เราสนใจจะเก็บจะเรียกว่า Entity และระหว่าง Entity จะมีความสัมพันธ์กัน เพราะฉนั้นผู้ออกแบบจะต้องหา Entity และความสัมพันธ์ระหว่าง Entity ให้เจอ

การออกแบบฐานข้อมูลจะเริ่มจากระดับแนวคิด (Conceptual Model) หลังจากนั้นจะแปลงจากระดับแนวคิดให้เป็นระดับตรรก (Logical Model) ระดับตรรกคือระดับที่ DBMS ใช้เพื่อให้ผู้ใช้เห็นว่าข้อมูลถูกจัดเก็บอย่างไรเช่น Oracle ให้ผู้ใช้เห็นว่าข้อมูลถูกจัดเก็บอยู่ในตาราง เรียกว่า Relational Database ซึ่งเป็นระดับตรรก (Logical Model)

E-R Diagram เป็นการออกแบบในระดับ Conceptual Model (ระดับแนวคิด) นั่นหมายถึงเมื่อเราวาด E-R Diagram เสร็จเราจะแปลง E-R ให้เป็น table หาก DBMS ที่เราใช้เป็นประเภท Relational Database เพราะฉนั้นอาจสรุปสาระสำคัญได้ว่า

1. การวาด E-R เป็นอิสระจาก DBMS หมายถึงเราใช้ E-R ในการ model ข้อมูล ส่วน DBMS จะใช้อะไรก็ได้
2. Entity ไม่ใช่ table และ table ไม่ใช่ Entity เพราะ Entity เป็นการมองในระดับ conceptual ส่วน table เป็นการมองในระดับ logical แม้ว่าสุดท้ายแล้ว Entity จะถูกแปลงเป็น table ก็ตาม
3. E-R จะเหมือนแผนที่ ทำให้เรารู้ว่าข้อมูลในองค์กรเรามีอะไรบ้าง และถูกจัดเก็บอยู่ที่ไหน

แผนภาพแสดงลำดับการออกแบบฐานข้อมูล



มาตรฐานของ E-R Diagram

E-R Diagram ได้ถูกนำเสนอครั้งแรกโดย Peter Chen เมื่อปี 1976 หลักจากนั้นมีการเพิ่มความซับซ้อน เข้ามาใน model เรียก Extended E-R model อย่างไรก็ดี E-R ก็ถูกนำเสนอมาตรฐานอื่นๆ ตามมาอีกหลายมาตรฐานเช่น Crow's foot, UML, IDEF1X ในเอกสารฉบับนี้จะยึดตามมาตรฐาน Crow's foot

Entity

Entity คือสิ่งที่เราสนใจจะเก็บและเป็นสิ่งที่แยกจากสิ่งอื่น เช่น พนักงาน ฝ่าย แทนด้วยสัญญลักษณ์สี่เหลี่ยมผืนผ้า

Attribute

Attribute คือสิ่งที่อธิบาย Entity เช่น Attribute ของ Entity พนักงาน คือ รหัสพนักงาน ชื่อ เพศ อายุ attribute ที่ทำหน้าที่บ่งชี้แต่ละสมาชิก (occurrence) เรียก attribute นั้นว่า key เช่นรหัสพนักงาน

Relationship

Relationship คือความสัมพันธ์ระหว่าง Entities แทนโดยการโยงเส้นตรง จำนวน entities ที่สัมพันธ์จะเรียกว่า **degree**

- Binary Relationship เป็นความสัมพันธ์ที่เกิดขึ้นระหว่างสอง entities และเป็นความสัมพันธ์ที่พบมากที่สุด เช่น ระหว่าง EMP กับ DEPT ซึ่งเราเรียกว่า Binary Relationships
- Unary Relationship (หรืออาจเรียกว่า Recursive Relationship) เป็นความที่เกิดขึ้นภายใน entity เดียวกัน เช่น Emp เราต้องการเก็บว่าพนักงานแต่ละคนใครเป็นหัวหน้า ใน entity EMP จะมี attribute หัวหน้าเพิ่มเข้ามาเพื่อเก็บว่าใครเป็นหัวหน้า
- Ternary Relationship เป็นความสัมพันธ์ที่เกิดขึ้นระหว่างสาม entities เช่น **Supplier** ส่ง **Part** ให้กับ **Project** ไหน อย่างไรก็ดีหากเราใช้ software ช่วยออกแบบ software ทุกตัวจะยอมรับแค่ Binary Relationship นั่นหมายถึงหนึ่งความสัมพันธ์ Ternary Relationship: Supplier – Part – Project เราต้องแปลงเป็นสาม Binary Relationship: Supplier – Part, Part – Project, Project - Supplier

Cardinality

คือประเภทความสัมพันธ์ระหว่าง Entity ซึ่งอาจแบ่งออกเป็น Maximum Cardinality และ Minimum Cardinality

Maximum Cardinality คือระดับความสัมพันธ์สูงสุด ที่ระหว่าง entity มีต่อกันจะมีอยู่สามระดับด้วยกันคือ

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

1 : 1	พนักงานหนึ่งคน มีห้องทำงานได้หนึ่งห้อง ห้องทำงานหนึ่งห้อง มีพนักงานได้ 1 คน	
1 : M	พนักงานหนึ่งคน อยู่ได้หนึ่งฝ่าย ฝ่ายหนึ่งฝ่าย มีพนักงานได้หลาย คน	
M : N	พนักงานหนึ่งคน เป็นกรรมการได้หลายคณะ กรรมการแต่ละคณะ มีพนักงานได้หลายคน	

Minimum Cardinality คือระดับความสัมพันธ์น้อยสุด ซึ่งแบ่งออกเป็น **มีก็ได้ไม่มีก็ได้** (Optional) แทนด้วยเครื่องหมายวงกลม และ **ต้องมี** (Mandatory) แทนด้วยเครื่องหมายขีด

	พนักงานทุกคนต้องมีห้องให้ทำงาน ห้องทำงาน อาจมีหรือไม่มีพนักงานอยู่	
	พนักงานทุกคนต้องมีฝ่ายอยู่ ทุกฝ่ายต้องมีพนักงานทำงาน	
	พนักงานอาจเป็นหรือไม่เป็นกรรมการได้ กรรมการทุกคณะ ต้องมีพนักงาน	

Identifying and Non Identifying Relationship

Identifying Relationship

ความสัมพันธ์ระหว่าง entity EMP และ EMP_TEL ให้สังเกตุว่า key ของ EMP_TEL (empid, tel) ส่วนหนึ่งจะนำมาจาก key ของ entity EMP (empid) เราจะเรียกความสัมพันธ์แบบนี้ว่า Identifying relationship และจะแทนด้วยเส้นหนาทึบ Entity ที่ key หรือส่วนหนึ่งของ key ถูกนำมาจาก entity อื่นอาจเรียกว่า ID-Dependent Entity หรือ Weak Entity

Non Identifying Relationship

ความสัมพันธ์ระหว่าง entity EMP และ DEPT ให้สังเกตุว่า key ของ DEPT (deptid) เป็นอิสระ มิได้นำมาจาก key ของ entity EMP (empid) หรือในทางกลับกัน key ของ EMP ก็มิได้นำมาจาก key ของ DEPT เราจะเรียกความสัมพันธ์แบบนี้ว่า Nonidentifying relationship และจะแทนด้วยเส้นปะ Entities EMP และ DEPT เรียกว่า Regular Entity

table ที่ foreign key อยู่เรียกว่า Referencing table หรือ table รอง ส่วน table ที่ถูก foreign key อ้างอิงหรือ table ที่ primary key อยู่เรียก table หลัก หรือ Referenced table

Minimum Cardinality

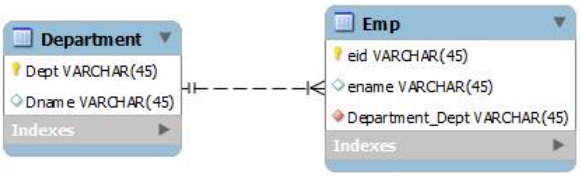
Minimum Cardinality จะมีผลต่อความถูกต้องของ table คือจะยอมให้ column ที่เป็น **foreign key** รับค่าว่างได้หรือไม่โดยหาก Minimum Cardinality เป็น Mandatory แล้ว Foreign key ห้ามเป็นค่าว่าง แต่หากเป็น Optional แล้ว Foreign key เป็นค่าว่างได้ ดังตัวอย่างต่อไปนี้

พนักงานทุกคนต้องมีห้องให้ทำงาน ห้องทำงาน อาจมีหรือไม่มีพนักงานอยู่ ความสัมพันธ์ 1:1 เราอาจกำหนดให้ foreign key อยู่ ที่ table ใดก็ได้ (ได้ table เดียว) หากเรากำหนดให้ foreign key อยู่ที่ EMP นั่นคือ attribute Officeno ใน table EMP ห้ามเป็นค่า Null แต่หาก เรากำหนดให้ foreign key อยู่ที่ Office นั่นหมายถึง attribute EMPID จะเป็นค่า Null หรือไม่ก็ได้เนื่องจากห้องทำงาน อาจมีหรือไม่มีพนักงานอยู่	
พนักงานทุกคนต้องมีฝ่ายอยู่ ทุกฝ่ายต้องมีพนักงานทำงาน กรณีดังกล่าว foreign key DEPTID จะอยู่ที่ EMP ห้ามรับค่าว่าง หากเราใช้ SQL Workbench จะกำหนดให้ DEPTID ที่ EMP ห้ามรับค่า Null โดยอัตโนมัติ	
พนักงานอาจเป็นหรือไม่เป็นกรรมการได้ กรรมการทุกคณะ ต้องมีพนักงาน กรณีความสัมพันธ์แบบ Many to Many จะเกิด Entity ตรงกลางขึ้นมาเรียก Association Entity และ Primary key ของ Association Entity จะเป็น Foreign key ด้วยเสมอ เพราะฉนั้น Foreign key จะไม่รับค่า Null เนื่องจาก Primary key ห้ามเป็นค่า Null ตามกฎ Entity Integrity	

สรุปประเด็นสำคัญ: Minimum Cardinality จะมีผลกับ Foreign key เท่านั้น ว่ารับค่าว่างได้หรือไม่ และจะมีผลต่อ Entities ที่มีความสัมพันธ์ระหว่างกันเป็น 1:1, 1:M ส่วน M:N จะไม่มีผล

Foreign Key Rules

หลังจากที่เรากำหนด Foreign key เสร็จแล้ว เราจะต้องมาพิจารณาว่า หากมีการเปลี่ยนค่า primary key หรือลบ row ขึ้นกับ Entity ที่ Foreign key อ้างอิงอยู่แล้ว ค่า Foreign key หรือ row ที่ Foreign key อยู่จะเป็นอย่างไร



กรณี Update

DEPARTMENT

dept	dname
AC	Acconting Department
IS	Information Department
MK	Marketing Department

EMP

id	fname	lname	sal	dept
101	Tom	White	10000	AC
102	Mary	Brown	12000	IS
103	Mike	Green	14000	AC

Primary key: DEPT ที่ entity: DEPARTMENT ถูกอ้างอิงโดย Foreign key: DEPT ที่ entity: EMP โดยปกติแล้ว Foreign key rule จะมีให้เลือก 2 อย่างคือ ***Restrict*** และ ***Cascade***

Restrict หากเราต้องการเปลี่ยน Primary key ของ DEPT เช่นจาก AC เป็น ACC ระบบจะเช็คว่ามี foreign key ที่เป็นค่า AC อยู่หรือไม่ที่ EMP ถ้าไม่มีจะอนุญาตให้เปลี่ยนจาก AC เป็น ACC แต่ถ้ามีจะไม่อนุญาตให้เปลี่ยน จากตาราง EMP มี foreign key ซึ่งเป็นค่า AC อยู่สอง rows เพราะฉนั้นหากเราออกแบบให้ foreign key rule ใช้ Restrict แล้ว ระบบจะไม่อนุญาตให้ผู้ใช้เปลี่ยนจาก AC เป็น ACC ได้ ข้อสังเกตุ เฉพาะ MK เปลี่ยนได้ เนื่องจากไม่มี MK เป็น foreign key เลย

Cascade หากเราต้องการเปลี่ยน Primary key ของ DEPT เช่นจาก AC เป็น ACC ระบบจะวิ่งไปเปลียน foreign key ทุก rows ที่เป็นค่า AC ให้เป็น ACC จะได้ผลลัพธ์ดังตารางด้านล่าง

DEPARTMENT

dept	dname
ACC	Acconting Department
IS	Information Department
MK	Marketing Department

EMP

id	fname	lname	sal	dept
101	Tom	White	10000	ACC
102	Mary	Brown	12000	IS
103	Mike	Green	14000	ACC

กรณี Delete

DEPARTMENT

dept	dname
AC	Acconting Department
IS	Information Department
MK	Marketing Department

EMP

id	fname	lname	sal	dept
101	Tom	White	10000	AC
102	Mary	Brown	12000	IS
103	Mike	Green	14000	AC

กรณี Delete จะเป็นเช่นเดียวกับกรณี Update ซึ่ง Foreign key rule สำหรับกรณี Delete จะมีให้เลือก 2 อย่างคือ ***Restrict*** และ ***Cascade***

Restrict หากเราต้องการลบ row ของ DEPT เช่นลบ row AC ระบบจะเช็คว่ามี foreign key ที่ EMP อยู่หรือไม่ซึ่งมีอยู่สอง rows ระบบจะไม่อนุญาตไม่ลบ ข้อสังเกตุ เฉพาะ MK ที่ลบได้ เนื่องจากไม่มี MK เป็น foreign key เลย

Cascade หากเราต้องการลบ row AC ที่ DEPARTMENT ระบบจะวิ่งไปลบทุก rows ที่ค่า foreign key เป็นค่า AC จะได้ผลลัพธ์ดังตารางด้านล่าง

DEPARTMENT

dept	dname
IS	Information Department
MK	Marketing Department

EMP

id	fname	lname	sal	dept
102	Mary	Brown	12000	IS

สรุป เมื่อมีการกำหนด foreign key เราจะต้องมีการกำหนด foreign key rule ซึ่งจะต้องแยกพิจารณาเป็นกรณี **Update** และ **Delete** ซึ่งแต่ละกรณีจะต้องเลือกใช้ **Restrict** หรือ **Cascade**

Weak Entities

Weak Entities คือ Entities ที่อยู่ไม่ได้หาก Parent Entities ไม่อยู่ ซึ่งจริงๆ แล้ว weak entities ก็คือ entities ที่มีความสัมพันธ์แบบ Identifying Relationship นั่นเอง จากตารางด้านล่างจะสรุปได้ดังนี้ สำหรับโปรแกรม ER-Workbench ไม่มีสัญลักษณ์พิเศษอะไรสำหรับ Weak Entities

Parent Entity: EMP

Weak Entities: EMP_TEL, EMP_EDUCATION, EMP_EMAIL

EMP

id	fname	lname	sal	dept
101	Tom	White	10000	AC
102	Mary	Brown	12000	IS

EMP_TEL

id	tel
101	812345678
101	812223333
102	816668899

EMP_EDUCATION

id	Degree	From	grade	Year
101	B	TU	3.25	2550
101	M	CU	3.75	2552
102	B	CU	3.23	2550

EMP_EMAIL

id	email
101	tom@hotmail.com
101	tom@gmail.com

Parent Entities: EMP, COMMITTEE

Weak Entity: EMP_COMMITTEE

COMMITTEE

cid	cname	start	end
C01	Quality Group	6/15/2012	10/1/2013
C02	KM Group	6/20/2012	6/20/2013
C03	QA Group	7/10/2012	12/31/2012

EMP_COMMITTEE

id	cid
101	C01
101	C03
102	C01
102	C02
102	C03

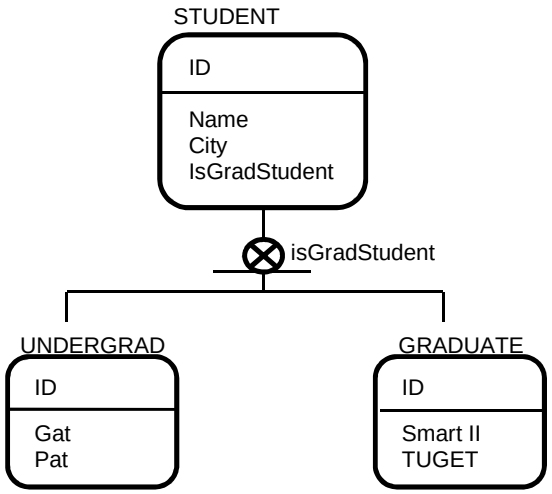
Subtype Entities

Extended E-R ได้แนะนำ Subtype Entities ซึ่ง Subtype Entities เป็น Entities ย่อยของ Supertype Entites ดังตัวอย่าง นักศึกษาปริญญาตรี (Undergrad) และนักศึกษาปริญญาโท (Graduate) มีสิ่งที่ต้องการเก็บเหมือนกันคือ Id, Name, City ซึ่งจะเก็บไว้ที่ Entity: STUDENT ส่วนที่ต่างกันเช่นนักศึกษาปริญญาตรีเราสนใจคะแนน Gat, Pat ส่วนปริญญาโทเราสนในคะแนน Smart II และ TU GET จะเก็บไว้ที่ Entities: UNDERGRAD และ GRADUATE ตามลำดับ

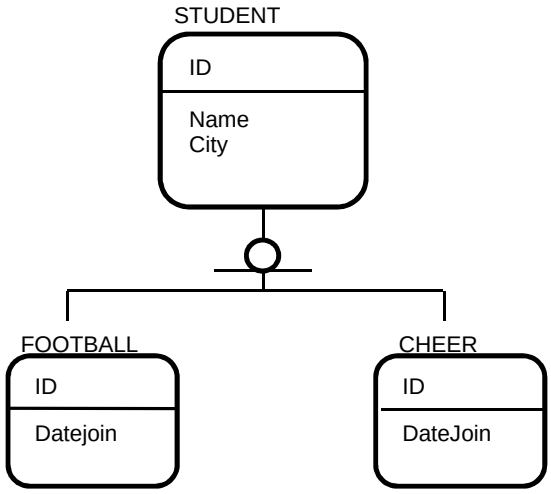
จุดประสงค์ในการออกแบบเป็น Supertype และ Subtype Entities คือจะทำให้ Entities ส่วนใหญ่ไม่เป็นค่า NULL เช่น หากเรารวม UNDERGRAD และ GRADUATE อยู่ใน Entities เดียวกันแล้ว นักศึกษาปริญญาตรีค่า attributes: Smart II และ TU GET ของทุกคนจะเป็นค่า NULL

Attribute: IsGradStudent เป็นตัวชี้ให้รู้ว่านักศึกษาผู้นี้เป็นปริญญาตรี หรือโท ซึ่งเรียกว่า Discriminator และจะเป็นได้เพียงตรีหรือโทอย่างใดอย่างหนึ่งเท่านั้นเรียก Exclusive Subtypes with Discriminator

ส่วนรูปด้านขวาเรียก Inclusive Subtypes คือนักศึกษาหนึ่งคนจะอยู่ชมรมไหนก็ได้และอยู่ได้มากกว่าหนึ่งชมรม



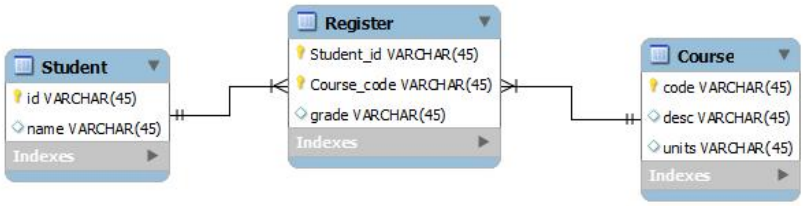
Exclusive Subtypes with Discriminator



Inclusive Subtypes

Association Relationship

หากความสัมพันธ์ระหว่าง Entity เป็นแบบ M:N แล้ว เราจะเรียกความสัมพันธ์นั้นว่า **Association Relationship** ตัวอย่าง Entities: STUDENT และ COURSE เป็นแบบ M:N กล่าวคือ นักศึกษาลงได้หลายวิชา หนึ่งวิชาลงได้หลายคน กรณีความสัมพันธ์แบบ M:N เราจะต้องสร้าง Entity ใหม่ขึ้นมาเพื่อแปลงความสัมพันธ์จาก M:N เป็น 1:M และ M:1 (MySQL Workbench จะสร้างให้โดยอัตโนมัติหากเรากำหนดความสัมพันธ์ระหว่าง Entities เป็นแบบ M:N) Entity ใหม่ที่สร้างขึ้นมาตามตัวอย่างด้านล่างคือ Register เราจะเรียก Entity Register ว่า **Association Entity**



ให้สังเกตุว่าความสัมพันธ์ระหว่าง Student-Register และ Course-Register จะเป็นแบบ Identifying Relationship นั่นคือ primary key ของ Student และ Course จะเป็นส่วนหนึ่งของ Primary key ของ Register และอาจกล่าวได้อีกอย่างคือ Register เป็น Weak Entity

ปัญหา Fan Traps และ Chasm Traps

ในการออกแบบ E-R มักจะเจอปัญหาการออกแบบผิดสองลักษณะด้วยกันคือ

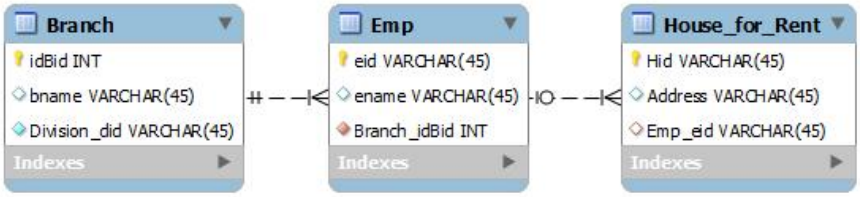
Fan Traps

Fan Trap จะเกิดขึ้นเมื่อมีความสัมพันธ์ 1:M ออกจาก Entity เดียวกันมากกว่าหนึ่งความสัมพันธ์

จากรูป ER ข้างต้นเราสามารถตอบได้ว่า Emp (พนักงาน) อยู่ Division (ฝ่าย) ไหน Division นั้นมี Branch (สาขา) ไหนบ้าง แต่ไม่สามารถตอบว่า Emp อยู่สาขาไหน ปัญหาดังกล่าวเรียกว่า Fan Traps เรามสามารถแก้ปัญหา Fan Traps ด้วย ER ด้านล่าง

Chasm Traps

Chasm Trap จะเกิดขึ้นเมื่อ Minimum Cardinality เป็น optional จากรูปบริษัทเปิดบ้านให้เช่า House_for_Rent บางหลังอาจจะไม่มีพนักงานดูแล House_for_Rent ที่ไม่มีพนักงานดูแลจะไม่รู้เลยว่าอยู่ Branch ไหน



เราสามารถแก้ปัญหา Chasm Traps ได้โดยการสร้างความสัมพันธ์ระหว่าง Branch กับ House_for_Rent ขึ้นมา ก็จะทำให้ทราบว่า House_for_Rent แต่ละหลังดูแลโดย Branch ใด