

ดร.โปรดปราน บุณยพุกสนะ

# พื้นฐานการอวอกแบบ ชาวบ้านมูลเชิงสัมพันธ์



โปรดปราน บุณยพุก gon

พื้นฐานการออกแบบฐานข้อมูลเชิงสัมพันธ์ / โปรดปราน บุณยพุก gon

1. ระบบการจัดการฐานข้อมูล
2. ฐานข้อมูลเชิงสัมพันธ์

พิมพ์ครั้งที่ 1 จำนวน 100 เล่ม มกราคม 2560

ลงวันลิขสิทธิ์ตาม พ.ร.บ. ลิขสิทธิ์ พ.ศ. 2537/2540

โดย โปรดปราน บุณยพุก gon

การผลิตและการออกเลียนตราเล่มนี้ไม่ว่ารูปแบบใดทั้งสิ้น<sup>1</sup>  
ต้องได้รับอนุญาตเป็นลายลักษณ์อักษรจากเจ้าของลิขสิทธิ์

จัดพิมพ์โดย

โปรดปราน บุณยพุก gon

ภาควิชาศิวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

จุฬาลงกรณ์มหาวิทยาลัย

พญาไท กรุงเทพฯ 10330

ออกแบบปก : กวิน เมศวร์คิริตรากุล

ออกแบบรูปเล่ม : โปรดปราน บุณยพุก gon

ภาพประกอบ : ชลธร ขัณฑ์จรเกียรติ และ กวิน เมศวร์คิริตรากุล

พิมพ์ที่ บริษัท นีโอ ดิจิตอล จำกัด เลขที่ 666 ช.สาธุประดิษฐ์ 58 แยก 22 (ประสานใจ)  
แขวงบางโพงพาง เขตยวานนาวา กรุงเทพฯ 10120 โทรศัพท์ 0-2683-8388

## คำนำ

การออกแบบฐานข้อมูลเป็นทักษะที่สำคัญสำหรับวิศวกรคอมพิวเตอร์ เพราะองค์กรมีว่า ขนาดเล็กหรือใหญ่ล้วนจำเป็นต้องใช้ฐานข้อมูลในการดำเนินธุรกิจหรือกิจกรรมใด ๆ ที่มีข้อมูลจำนวนมาก หากออกแบบได้ไม่เหมาะสมจะส่งผลให้การทำงานของระบบฐานข้อมูลนั้นล้มเหลวได้ง่าย

ตำราพื้นฐานการออกแบบฐานข้อมูลเชิงสัมพันธ์นี้ เป็นการรวบรวมความรู้พื้นฐาน แนวคิด เครื่องมือ วิธีการ และตัวอย่างการออกแบบฐานข้อมูลเชิงสัมพันธ์ เพื่อเป็นตำราประกอบ รายวิชา 2110422 การออกแบบระบบการจัดการฐานข้อมูล (Database Management Systems Design) สำหรับนิสิตปริญญาตรี ของภาควิชาวิศวกรรมคอมพิวเตอร์ คณะ วิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย เนื้อหาครอบคลุมคำอธิบายรายวิชา แต่อាជมี ศัพท์เบ็ดลี่ย์แปลงไปบ้างเพื่อให้เหมาะสมและสอดคล้องกับความเปลี่ยนแปลงของ เทคโนโลยีฐานข้อมูลที่มีการพัฒนาอย่างรวดเร็ว

ตำนานี้เน้นภาคทฤษฎีและมีตัวอย่างให้พ่อฝึกฝนปฏิบัติได้ ผู้เขียนเชื่อว่าได้ถ่ายทอดความรู้ เพียงพอต่อการศึกษาระดับปริญญาบัณฑิตด้านวิศวกรรมคอมพิวเตอร์ ที่ต้องเข้าใจการ ทำงานและการคำนวณภายในฐานข้อมูล จึงจะออกแบบฐานข้อมูลได้อย่างมีประสิทธิภาพ และเชื่อว่าจะมีพื้นฐานที่เข้มแข็งพอที่จะพัฒนาองค์ความรู้ใหม่ต่อไปในระดับบัณฑิตศึกษา ได้ ความรู้ในตำนานี้เป็นพื้นฐานสำหรับวิศวกรคอมพิวเตอร์ที่จะประกอบวิชาชีพในองค์กร ที่ใช้ฐานข้อมูลโดยเฉพาะฐานข้อมูลเชิงสัมพันธ์

สุดท้ายนี้ ผู้เขียนขอขอบคุณครอบครัวที่ให้เวลาและกำลังใจ ขอบคุณลูกศิษย์ทั้งหลาย โดยเฉพาะ กฤตย์ กังวาลพงศ์พันธุ์ ชลธร ขวัญจรเกียรติ กวิน เมศร์ศิริตรรกฎ สัญชัย จักรีรังกร เมสินี นาคมณี วิสิฐ์ วงศ์ชัยอนุกูล ธัญนุช ฉันทานุรักษ์ และสุภัชชา เมหรัตน ชร ที่เป็นแรงสำคัญผลักดันให้ตั้งใจเขียนนี้สู่เรื่องลึกลับ

โปรดปราน บุณยพุก gon  
มกราคม 2560



# สารบัญ

สารบัญ .....	i
สารบัญภาพ.....	xii
บทที่ 1 ฐานข้อมูลเบื้องต้น .....	1
1.1 ฐานข้อมูล ระบบการจัดการฐานข้อมูล และระบบ ฐานข้อมูล .....	3
1.2 ความสัมพันธ์ของระบบการจัดการฐานข้อมูลและระบบ ฐานข้อมูล .....	7
1.3 คุณสมบัติของฐานข้อมูล .....	10
1.4 คุณสมบัติของระบบการจัดการฐานข้อมูล .....	10
1.5 วิัฒนาการของระบบการจัดการฐานข้อมูล.....	13
1.5.1 ช่วงที่ 1 : ยุคการจัดการข้อมูลแบบไฟล์ .....	14
1.5.2 ช่วงที่ 2 : ยุคการจัดการข้อมูลแบบการเข้าถึงไฟล์ .....	15
1.5.3 ช่วงที่ 3 : ยุคระบบฐานข้อมูลในช่วงแรกเริ่ม... .....	16

1.5.4	ช่วงที่ 4 : ระบบฐานข้อมูลเชิงสัมพันธ์ .....	17
1.5.5	ช่วงที่ 5 : ყุคปัจจุบัน .....	19
1.6	บทสรุปและเรื่องหวานคิด .....	19
	แบบฝึกหัดท้ายบท.....	20
บทที่ 2	การพัฒนาระบบฐานข้อมูล.....	21
2.1	การพัฒนาระบบฐานข้อมูล.....	24
2.2	สถาปัตยกรรมระบบฐานข้อมูล .....	25
2.2.1	สถาปัตยกรรมแบบทรีศิริมา .....	26
2.2.2	สถาปัตยกรรมแบบรวมศูนย์ .....	29
2.2.3	สถาปัตยกรรมแบบลูกข่ายแม่ข่าย.....	29
2.2.4	สถาปัตยกรรมแบบคลาวด์ .....	32
2.3	ภาษาฐานข้อมูล .....	33
2.3.1	ภาษาดีตีแอล.....	33
2.3.2	ภาษาดีเอ็มแอล .....	33
2.4	ประเภทของระบบการจัดการฐานข้อมูล .....	34
2.5	ภาพรวมของระบบการจัดการฐานข้อมูล .....	38
2.6	บทสรุปและเรื่องหวานคิด .....	39
	แบบฝึกหัดท้ายบท.....	41
บทที่ 3	การจำลองข้อมูล .....	43

3.1	การจำลองข้อมูล.....	44
3.2	แบบจำลองข้อมูล.....	45
3.2.1	แบบจำลองข้อมูลเชิงแนวคิด.....	45
3.2.2	แบบจำลองข้อมูลเชิงภาษาพ .....	45
3.2.3	แบบจำลองข้อมูลเชิงปฏิบัติการ.....	45
3.3	แบบจำลองอีอาร์ .....	46
3.3.1	เอนทิตี้ .....	49
3.3.2	แอ็ตทริบิวต์.....	52
3.3.3	รีเลชันชิป .....	57
3.3.4	คาร์ตินิลิตี้.....	61
3.3.5	เงื่อนไขแบ่งคับเข้าร่วม .....	63
3.3.6	สรุปสัญกรณ์อีอาร์.....	65
3.3.7	สัญกรณ์เพิ่มเติม.....	66
3.4	แบบจำลองอีอาร์.....	69
3.4.1	ชับคลาส/ชูเปอร์คลาส .....	70
3.4.2	การรับทดสอบ .....	72
3.4.3	สเปเชียลไลเซชันและเจเนอเรชัน.....	73
3.4.4	การกำหนดสมाचิกชับคลาส.....	74
3.4.5	ความมีส่วนร่วมของชับคลาส .....	76

3.4.6	ความสมบูรณ์ของความสัมพันธ์ .....	77
3.4.7	กฎการแทรกรและ การลบข้อมูลในอีอีอาร์ .....	78
3.4.8	จำนวนชั้นของスペเชียลไลเซชัน .....	79
3.4.9	แคทากอรี .....	81
3.5	เครื่องมือสร้างแบบจำลองข้อมูล .....	83
3.6	บทสรุปและเรื่องชวนคิด .....	85
	แบบฝึกหัดท้ายบท .....	86
<b>บทที่ 4</b>	<b>ฐานข้อมูลเชิงสัมพันธ์ .....</b>	<b>87</b>
4.1	รีเลชัน .....	89
4.1.1	ทูเพิล .....	91
4.1.2	แอตทริบิวต์ .....	92
4.1.3	โดเมน .....	92
4.2	สัญกรณ์คณิตศาสตร์ .....	93
4.3	เงื่อนไขบังคับของฐานข้อมูลเชิงสัมพันธ์ .....	96
4.3.1	เงื่อนไขโดเมน .....	97
4.3.2	เงื่อนไขคีร์ .....	97
4.3.3	เงื่อนไขบูรณภาพโอนทิศ .....	99
4.3.4	เงื่อนไขบูรณภาพอ้างอิง .....	100
4.3.5	เงื่อนไขบังคับอื่น ๆ .....	101

4.4	การปรับปรุงข้อมูลในรีเลชัน .....	102
4.4.1	การแทรก.....	102
4.4.2	การลบ .....	103
4.4.3	การแก้ไข.....	103
4.5	บทสรุปและเรื่องหวานคิด .....	104
	แบบฝึกหัดท้ายบท.....	105
บทที่ 5	แบบจำลองกับฐานข้อมูลเชิงสัมพันธ์ .....	107
5.1	ขั้นตอนการแปลงอีอาร์และอีอาร์เป็นรีเลชัน .....	108
5.1.1	ขั้นตอนที่ 1 : แปลงเอนทิตี้ .....	110
5.1.2	ขั้นตอนที่ 2 : แปลงวีคเอนทิตี้.....	111
5.1.3	ขั้นตอนที่ 3 : แปลงรีเลชันชิบไฟป์ 1:1 .....	112
5.1.4	ขั้นตอนที่ 4 : แปลงรีเลชันชิบไฟป์ 1:N.....	114
5.1.5	ขั้นตอนที่ 5 : แปลงรีเลชันชิบไฟป์ M:N.....	115
5.1.6	ขั้นตอนที่ 6 : แปลงแอตทริบิวต์หลายค่า.....	116
5.1.7	ขั้นตอนที่ 7 : แปลงรีเลชันชิบแบบเอนนาเรี... 117	
5.1.8	ขั้นตอนที่ 8 : แปลงซับคลาสๆเปอร์คลาส.....	118
5.1.9	ขั้นตอนที่ 9 : แปลงยูเนียน .....	123
5.2	บทสรุปและเรื่องหวานคิด .....	124
	แบบฝึกหัดท้ายบท.....	126

บทที่ 6 การนอร์ม่าไลซ์ .....	129
6.1 ปัญหาจากความเข้าช้อน .....	130
6.1.1 ความผิดพลาดจากการแก้ไข .....	130
6.1.2 ความผิดพลาดจากการแทรก .....	131
6.1.3 ความผิดพลาดจากการลบ .....	131
6.2 พังก์ชันลัดดีเพนเดนซี .....	132
6.3 การหาพังก์ชันลัดดีเพนเดนซีจากข้อมูล .....	134
6.4 การหาพังก์ชันลัดดีเพนเดนซีทั้งหมด .....	136
6.4.1 เอฟโคลเชอร์ .....	136
6.4.2 เอกซ์โคลเชอร์ .....	140
6.5 การหาคีย์จาก $F^+$ และ $X^+$ .....	141
6.6 การนอร์ม่าไลซ์ .....	142
6.7 ขั้นตอนการนอร์ม่าไลซ์ .....	143
6.7.1 นอร์มัลฟอร์มที่ 1 .....	143
6.7.2 นอร์มัลฟอร์มที่ 2 .....	147
6.7.3 นอร์มัลฟอร์มที่ 3 .....	150
6.7.4 นอร์มัลฟอร์มบอยซ์คอดต์ .....	154
6.8 บทสรุปและเรื่องawanคิด .....	157
แบบฝึกหัดท้ายบท .....	158

บทที่ 7 พีชคณิตเชิงสัมพันธ์และแคลคูลัสเชิงสัมพันธ์ .....	159
7.1 หลักการของพีชคณิตเชิงสัมพันธ์ .....	160
7.2 โอลเปอเรชันเชิงสัมพันธ์.....	161
7.2.1 SELECT .....	162
7.2.2 PROJECT .....	165
7.2.3 RENAME .....	166
7.2.4 UNION .....	168
7.2.5 INTERSECTION .....	170
7.2.6 MINUS.....	170
7.2.7 CARTESIAN.....	173
7.2.8 DIVISION.....	177
7.2.9 JOIN .....	179
7.2.10 OUTER JOIN .....	183
7.2.11 แออกกรีเกตฟังก์ชันและการกรุป .....	186
7.3 เขตสมบูรณ์ของโอลเปอเรชันเชิงสัมพันธ์ .....	188
7.4 แคลคูลัสเชิงสัมพันธ์.....	189
7.4.1 แคลคูลัสเชิงสัมพันธ์อิงทุเพิล.....	190
7.4.2 แคลคูลัสเชิงสัมพันธ์อิงโดเมน .....	191

7.5	เปรียบเทียบพีชคณิตเชิงสัมพันธ์กับแคลคูลัสเชิงสัมพันธ์ .....	193
7.6	บทสรุปและเรื่องชวนคิด .....	194
	แบบฝึกหัดท้ายบท.....	195
บทที่ 8	ภาษาເອສຄົວແອລ.....	199
8.1	ຮັບການພາສາເອສຄົວແອລ .....	200
8.2	คำສົ່ງທີ່ເຂັ້ມງວຍ .....	204
8.2.1	ສ້າງເຈົ້າເລັ້ນ .....	205
8.2.2	ລບປີເລັ້ນ .....	205
8.2.3	ແກ້ໄຂເຈົ້າເລັ້ນ.....	206
8.2.4	ເພີ່ມຫຼຸເພີລ .....	206
8.2.5	ລບຖຸເພີລ.....	206
8.2.6	ອັບເດຕຖຸເພີລ.....	207
8.2.7	ກຳນົດຄື່ນໜັກແລະຄື່ນຄູ່ແໜ່ງ .....	207
8.2.8	ກຳນົດພອຣັນຄື່ນ.....	208
8.2.9	ກຳນົດທາງເລືອກເມື່ອລົບຫຼືອັບເດຕຂໍ້ມູນລົ້າງອີງ ຫັ້ມຕາຮາງ.....	209
8.2.10	ສ້າງວິວ .....	209
8.3	บทสรุปและเรื่องชวนคิด .....	210

แบบฝึกหัดท้ายบท.....	211
<b>บทที่ 9 การจัดเก็บและการทำดัชนี.....</b>	<b>215</b>
9.1 หน่วยเก็บข้อมูลภายนอก.....	218
9.1.1 ประเภทของหน่วยเก็บข้อมูลภายนอก .....	218
9.2 ไฟล์และวิธีการเข้าถึง.....	219
9.3 การจัดโครงสร้างไฟล์.....	220
9.4 ดัชนี.....	221
9.5 คีย์ค้นหาและรายการข้อมูล .....	222
9.6 วิธีการสร้างรายการข้อมูล .....	223
9.6.1 วิธีที่ 1 โครงสร้างไฟล์แบบดัชนี.....	223
9.6.2 วิธีที่ 2 คู่ <k,rid>.....	224
9.6.3 วิธีที่ 3 <k,rid-list>.....	224
9.7 ประเภทของดัชนี .....	224
9.8 โครงสร้างของดัชนี.....	229
9.8.1 โครงสร้างดัชนีแบบแทช .....	229
9.8.2 โครงสร้างดัชนีแบบตันไม้ .....	233
9.9 การปรับความสมดุลของต้นไม้บีเพลส .....	237
9.10 การคำนวณเวลา .....	241
9.11 การะของฐานข้อมูล.....	251

9.11.1	ภาระการสอบคาม .....	252
9.11.2	ภาระการอัปเดต .....	252
9.12	ข้อแนะนำในการเลือกตั้งชี.....	253
9.13	อินเติ๊กซ์โอนลี .....	257
9.14	บทสรุปและเรื่องชวนคิด .....	258
	แบบฝึกหัดท้ายบท.....	259
	บทส่งท้าย.....	261
	บรรณานุกรม.....	263
	ตั้งชี.....	270

# สารบัญภาพ

รูปที่ 1 ความสัมพันธ์ระหว่าง ฐานข้อมูล ระบบการจัดการฐานข้อมูล และระบบฐานข้อมูล .....	5
รูปที่ 2 ตารางฐานข้อมูลมหาวิทยาลัย.....	7
รูปที่ 3 ระบบการจัดการฐานข้อมูลและระบบฐานข้อมูล.....	8
รูปที่ 4 การจัดการข้อมูลแบบไฟล์ .....	14
รูปที่ 5 การจัดการข้อมูลแบบการเข้าถึงไฟล์ .....	16
รูปที่ 6 ระบบฐานข้อมูลในช่วงแรกเริ่ม.....	17
รูปที่ 7 ระบบฐานข้อมูลเชิงสัมพันธ์.....	18
รูปที่ 8 ความสัมพันธ์ระหว่างการพัฒนาแอปพลิเคชัน กับการพัฒนาระบบฐานข้อมูล .....	23
รูปที่ 9 ขั้นตอนการพัฒนาระบบฐานข้อมูล .....	24
รูปที่ 10 สถาปัตยกรรมแบบทรีสตีม่า.....	27
รูปที่ 11 สถาปัตยกรรมแบบบลูกูป่ายแม่ป่ายสองชั้น .....	30
รูปที่ 12 สถาปัตยกรรมแบบบลูกูป่ายแม่ป่ายสามชั้น .....	31
รูปที่ 13 สถาปัตยกรรมแบบคลาวด์ .....	32
รูปที่ 14 ฐานข้อมูลเชิงลำดับชั้น .....	35
รูปที่ 15 ฐานข้อมูลเชิงเครือข่าย.....	35
รูปที่ 16 ฐานข้อมูลเชิงสัมพันธ์.....	36
รูปที่ 17 การทำงานภายใต้ระบบการจัดการฐานข้อมูล .....	39
รูปที่ 18 แผนภาพอีอาร์แสดงระบบฐานข้อมูลของบริษัทดีบี.....	49
รูปที่ 19 วีคเอนกิตตี้ไฟปี .....	51

รูปที่ 20 เอนทิเตและแอตทริบิวต์จากแผนภาพอีอาร์บริษัทดีบี.....	54
รูปที่ 21 แอตทริบิวต์ร่วม .....	55
รูปที่ 22 รีเลชันชิปที่สอดคล้องกับบทบาท .....	57
รูปที่ 23 รีเลชันชิปแบบเวียนเกิด .....	59
รูปที่ 24 รีเลชันชิปแบบเทอนารี.....	60
รูปที่ 25 แอตทริบิวต์ของรีเลชันชิป.....	60
รูปที่ 26 แผนภาพอีอาร์แสดงระบบฐานข้อมูลของบริษัทดีบี (ช้า).....	62
รูปที่ 27 เนื่องไฟบังคับเข้าร่วมแบบบังคับและแบบไม่บังคับ .....	64
รูปที่ 28 สรุปสัญกรณ์อีอาร์.....	65
รูปที่ 29 เกรียงพิธีบัญญัติแบบต่าง ๆ .....	67
รูปที่ 30 สัญกรณ์ย่อเม็งแอล .....	68
รูปที่ 31 แผนภาพอีอาร์.....	70
รูปที่ 32 ชูเปอร์คลาสชั้บคลาส.....	71
รูปที่ 33 ชั้บคลาสที่ใช้เงื่อนไขกำหนด .....	75
รูปที่ 34 สเปเชียลไลเซชันแบบแลตทิซ .....	79
รูปที่ 35 สเปเชียลไลเซชันแบบคำตั้งชั้นและแบบแลตทิซ.....	80
รูปที่ 36 ชั้บคลาสประเภทแคทגורีและการรูปเนียน .....	82
รูปที่ 37 หน้าจอของ PowerDesigner .....	83
รูปที่ 38 ดร. เอดการ์ เอฟ คออดด์ .....	88
รูปที่ 39 องค์ประกอบของรีเลชัน .....	89
รูปที่ 40 รีเลชัน STUDENT.....	90
รูปที่ 41 ศัพท์ที่ใช้ในฐานข้อมูลเชิงสัมพันธ์.....	96
รูปที่ 42 รีเลชัน CAR.....	98
รูปที่ 43 รีเลชันบริษัทดีบีแปลงจากอีอาร์.....	101

รูปที่ 44 แผนภาพอีอาร์แสดงระบบฐานข้อมูลของบริษัทดีปี.....	109
รูปที่ 45 รีเลชันแสดงระบบฐานข้อมูลของบริษัทดีปี .....	109
รูปที่ 46 แปลงเอนทิตี้ไทย.....	110
รูปที่ 47 แปลงวีคเอนทิตี้ไทย .....	112
รูปที่ 48 แปลงรีเลชันชิบไทย 1:1.....	113
รูปที่ 49 แปลงรีเลชันชิบไทย 1:N .....	115
รูปที่ 50 แปลงรีเลชันชิบไทย m:n.....	116
รูปที่ 51 แผนภาพอีอาร์แสดงรีเลชันชิบแบบเทอนารี.....	117
รูปที่ 52 ผลการแปลงรีเลชันชิบแบบเทอนารี.....	118
รูปที่ 53 แผนภาพอีอาร์แสดงชูเปอร์คลาสซับคลาส EMPLOYEE.....	119
รูปที่ 54 ตัวอย่างการแปลงด้วยวิธีวิธีแยกรีเลชันชูเปอร์คลาสและซับคลาส .....	119
รูปที่ 55 แผนภาพอีอาร์แสดงชูเปอร์คลาสซับคลาส VEHICLE .....	120
รูปที่ 56 ตัวอย่างการแปลงด้วยวิธีแยกรีเลชันใช้ซับคลาส .....	120
รูปที่ 57 แผนภาพอีอาร์แสดงชูเปอร์คลาสซับคลาสแบบแอ็ตทริบิวต์กำหนด .....	121
รูปที่ 58 ตัวอย่างการแปลงด้วยวิธีรวมรีเลชันพร้อมแอ็ตทริบิวต์ระบุตัวเดียว .....	121
รูปที่ 59 แผนภาพอีอาร์แสดงชูเปอร์คลาสซับคลาสแบบ overlap.....	122
รูปที่ 60 วิธีรวมรีเลชันพร้อมแอ็ตทริบิวต์ระบุหลายตัว.....	122
รูปที่ 61 แผนภาพอีอาร์แสดงแคทหอกอรี.....	123
รูปที่ 62 แสดงการแปลงแบบยืนยัน.....	124
รูปที่ 63 รีเลชัน Employee .....	134
รูปที่ 64 Hourly_Emps=(SNLRWH).....	135

รูปที่ 65 ย่อรีเลชัน Hourly_Emps .....	136
รูปที่ 66 รีเลชันที่มีแอตทริบิวต์หลายค่า.....	144
รูปที่ 67 DEPT_LOCATION 1NF.....	144
รูปที่ 68 รีเลชันในรูปแบบ 1NF ด้วยวิธีขยายคีรี.....	145
รูปที่ 69 สมมติของบริษัทดีปี .....	146
รูปที่ 70 สถานะของข้อมูลในฐานข้อมูลบริษัทดีปี .....	147
รูปที่ 71 รีเลชัน EMP_PROJ.....	148
รูปที่ 72 FD ของ EMP_PROJ .....	149
รูปที่ 73 EMP_PROJ ย่อให้อยู่ในรูปแบบ 2NF.....	150
รูปที่ 74 รีเลชัน EMP_DEPT .....	150
รูปที่ 75 FD ของ EMP_DEPT .....	151
รูปที่ 76 EMP_PROJ ย่อให้อยู่ในรูปแบบ 3NF.....	152
รูปที่ 77 รีเลชัน LOTS กับ 4 FD .....	152
รูปที่ 78 รีเลชัน LOTS1 และ LOTS2 ในรูปแบบ 2NF.....	153
รูปที่ 79 ย่อรีเลชัน LOTS1 เป็น LOTS1A และ LOTS1B ในรูปแบบ 3NF .....	154
รูปที่ 80 รีเลชัน LOTS ย่อให้อยู่ในรูปแบบ 3NF .....	154
รูปที่ 81 รีเลชัน LOTS1A ที่ไม่ผ่านบีชีอีนเอฟ .....	155
รูปที่ 82 รีเลชัน LOTS1AX และ LOTS1AY ในรูปแบบ BCNF .....	155
รูปที่ 83 รีเลชัน TEACH.....	156
รูปที่ 84 รีเลชันบริษัทดีปี .....	163
รูปที่ 85 ผลลัพธ์การ SELECT .....	164
รูปที่ 86 ผลลัพธ์การ PROJECT LNAME, FNAME, SALARY .....	166
รูปที่ 87 ผลลัพธ์การ PROJECT SEX, SALARY.....	166

รูปที่ 88 โโคเปอเรชันยูนียัน .....	169
รูปที่ 89 การ UNION INTERSECTION และ MINUS.....	171
รูปที่ 90 ผลการ SELECT และ PROJECT .....	175
รูปที่ 91 ผลการ CARTESIAN .....	176
รูปที่ 92 CARTESIAN ตามด้วย SELECT .....	176
รูปที่ 93 ขั้นตอนและผลลัพธ์ของ DIVISION ssns .....	178
รูปที่ 94 ขั้นตอนและผลลัพธ์ของ DIVISION R/S.....	178
รูปที่ 95 รีเลชันบริษัทดีปี .....	180
รูปที่ 96 ผลลัพธ์ของ EQUI JOIN .....	181
รูปที่ 97 ขั้นตอนและผลลัพธ์ของ NATURAL JOIN .....	182
รูปที่ 98 ขั้นตอนและผลลัพธ์การ LEFT OUTER JOIN .....	184
รูปที่ 99 ขั้นตอนและผลลัพธ์การ RIGHT OUTER JOIN.....	185
รูปที่ 100 ขั้นตอนและผลลัพธ์การ FULL OUTER JOIN.....	186
รูปที่ 101 ผลลัพธ์การแยกก្នុង .....	188
รูปที่ 102 ผลลัพธ์การกรุ๊ป .....	188
รูปที่ 103 รีเลชัน STUDENTS .....	202
รูปที่ 104 ผลลัพธ์การ SELECT * .....	203
รูปที่ 105 รีเลชัน ENROLL .....	204
รูปที่ 106 ผลลัพธ์นักเรียน gpa = 4.0 .....	204
รูปที่ 107 โครงสร้างของระบบการจัดการฐานข้อมูล .....	217
รูปที่ 108 รีเลชัน Employees .....	223
รูปที่ 109 ดัชนีคัลสเตอร์กับดัชนีไม่คัลสเตอร์.....	225
รูปที่ 110 ดัชนีหลัก.....	226
รูปที่ 111 ดัชนีคัลสเตอร์.....	227

รูปที่ 112	ดัชนีคลัสเตอร์.....	228
รูปที่ 113	พังก์ชันแซช.....	231
รูปที่ 114	ดัชนีหลัก.....	232
รูปที่ 115	โครงสร้างแบบต้นไม้บีเพลส .....	233
รูปที่ 116	รายการดัชนี.....	234
รูปที่ 117	การใช้ต้นไม้บีเพลส ดัชนีด้วย age .....	235
รูปที่ 118	ต้นไม้บีเพลส ให้ age เป็น k .....	236
รูปที่ 119	แทรกข้อมูล $\text{age}^* = 70^*$ .....	238
รูปที่ 120	แทรกข้อมูล $\text{age}^* = 95^*$ .....	239
รูปที่ 121	เพิ่มชั้นของต้นไม้.....	239
รูปที่ 122	โหนดเพิ่มชั้น.....	240
รูปที่ 123	ต้นไม้บีเพลสสมดุล .....	240
รูปที่ 124	ดัชนีแบบคีร์คันหาร่วม.....	256

## สารบัญตาราง

ตารางที่ 1 เครื่องมือแบบจำลองฐานข้อมูล .....	84
ตารางที่ 2 ศัพท์ที่ใช้ในฐานข้อมูลเชิงสัมพันธ์ .....	95
ตารางที่ 3 ศัพท์ที่ใช้ในแบบจำลองเชิงสัมพันธ์และพีซคณิตเชิงสัมพันธ์กับศัพท์ที่ใช้ในเอกสารวิเคราะห์เวลา .....	202
ตารางที่ 4 ผลการวิเคราะห์เวลา .....	244



## บทที่ 1

### ฐานข้อมูลเบื้องต้น

#### วัตถุประสงค์

- เพื่อให้ทราบนิยามของฐานข้อมูล
- เพื่อให้ทราบคุณสมบัติของฐานข้อมูล
- เพื่อให้เข้าใจความแตกต่างระหว่างฐานข้อมูล ระบบการจัดการฐานข้อมูล และระบบฐานข้อมูล
- เพื่อให้ทราบประวัติความเป็นมาของฐานข้อมูล

ข้อมูลเป็นสิ่งที่อยู่รอบตัวเรา มีอยู่มากมาย และต้องจัดเก็บอย่างมีระเบียบ เพื่อให้นำไปใช้งานได้อย่างมีประสิทธิภาพ ทำได้ด้วยการนำข้อมูลที่สนใจมา ออกแบบแล้วจัดเก็บเป็นฐานข้อมูลตามเรื่องนั้น ๆ อย่างเหมาะสม ขอ ยกตัวอย่างระบบฐานข้อมูลทะเบียนนิสิตจุฬาฯ หรือที่นิสิตเรียกอย่างติดปาก ว่า เร็กจุฬาฯ (Reg Chula) เป็นสิ่งที่ใกล้ตัวและสำคัญต่อนิสิตจุฬาฯ ทุกคน เพราะเร็กจุฬาฯ เป็นฐานข้อมูลที่เก็บข้อมูลของนิสิตໄ่าวเป็นจำนวนมากมาก ทั้งชื่อ นามสกุล วิชาที่ลงทะเบียน ชื่ออาจารย์ที่ปรึกษา ชื่อผู้ปกครอง สถานที่ติดต่อ โรงเรียนมัธยมฯ ที่จบการศึกษามา หรือผลการเรียนสมัยมัธยมฯ แล้วผู้เรียน คิดว่าเร็กจุฬาฯ เป็นฐานข้อมูลที่ใหญ่ที่สุดในมหาวิทยาลัยหรือไม่ ปัจจุบัน จุฬาฯ มีนิสิตประมาณ 40,000 คน และมีนิสิตจบการศึกษาเกือบ 10,000 คนต่อปี ยิ่งไปกว่านั้นจุฬาฯ ยังถูกก่อตั้งมานานกว่าร้อยปีแล้ว ดังนั้น ฐานข้อมูลของเร็กจุฬาฯ จึงมีขนาดใหญ่มาก แต่ว่าขนาดฐานข้อมูลของเร็ก จุฬาฯ ที่ว่าใหญ่นั้น เมื่อนำไปเทียบกับระบบฐานข้อมูลประชากร ระบบ ฐานข้อมูลของธนาคาร ระบบฐานข้อมูลบริษัทโทรคมนาคมแล้ว อาจพบว่า ข้อมูลในเร็กจุฬาฯ อาจดูมีขนาดเล็กกว่ามาก เช่น ข้อมูลของบริษัท โทรคมนาคมนั้นจะต้องมีการจัดเก็บล็อก (log) การโทรศัพท์ทั้งหมด เมื่อ ข้อมูลที่เก็บมีจำนวนมากขึ้น ฐานข้อมูลก็จะมีขนาดใหญ่และซับซ้อน ตามกันไป

ในบทนี้ ผู้เรียนจะได้ทราบนิยามของฐานข้อมูล คุณสมบัติของฐานข้อมูล ความแตกต่างระหว่างฐานข้อมูล ระบบฐานข้อมูล ระบบการจัดการ ฐานข้อมูล และประวัติความเป็นมาของฐานข้อมูล ตำราเล่มนี้อิงมาจาก หนังสือ Fundamentals of Database Systems ของศาสตราจารย์ เอกมารคีและศาสตราจารย์นava เ [1-3]

## 1.1 ฐานข้อมูล ระบบการจัดการฐานข้อมูล และระบบฐานข้อมูล

วิชาเกี่ยวกับการออกแบบฐานข้อมูลมีมาช้านาน เพราะเป็นศาสตร์สำคัญในวงการวิศวกรรมคอมพิวเตอร์ วิทยาศาสตร์คอมพิวเตอร์ และเทคโนโลยีสารสนเทศ และยังเกี่ยวข้องกับหลักสูตรทางธุรกิจ เพราะธุรกิจส่วนใหญ่มีความจำเป็นต้องใช้ฐานข้อมูล ดังนั้นจึงมีผู้นิยามคำว่าฐานข้อมูลไว้มากมาย นิยามที่ชัดเจนมาจากปรมารายผู้เขียนหนังสือวิชาระบบการจัดการฐานข้อมูลที่ใช้เรียนกันอย่างกว้างขวาง [1-3] กล่าวไว้ว่า “ฐานข้อมูล (database) คือ “กลุ่ม (collection) ข้อมูลที่เป็นตัวแทนสถานการณ์จริงในมุมที่สนใจ ข้อมูลเหล่านั้นต้องมีความสอดคล้องกัน (coherent) และมีความหมายในตัวเอง (inherent meaning) นอกจากนี้ฐานข้อมูลจะต้องได้รับการออกแบบและสร้างเพื่อใช้บันทึกข้อมูลตามวัตถุประสงค์นั้น” ฐานข้อมูลอาจมีขนาดเท่าใดก็ได้และอาจมีความซับซ้อนไม่เท่ากัน ส่วนข้อมูล (data) คือข้อเท็จจริงที่สามารถบันทึกได้และมีความหมายโดยปริยาย

จากกล่าวโดยสรุปได้ว่า ฐานข้อมูล คือกลุ่มของข้อมูลที่เกี่ยวข้องกัน เช่น ฐานข้อมูลร้านอาหาร อาจบันทึกข้อมูลตัวร้าน เมนูอาหาร จำนวนโต๊ะและที่นั่ง ราคาอาหาร พนักงานในร้าน แต่ไม่รวมข้อมูลบัญชีธนาคารของเจ้าของร้าน หรือข้อมูลการท่องเที่ยวของเจ้าของร้าน เป็นต้น กล่าวคือ ข้อมูลในฐานข้อมูลหนึ่ง ๆ ถ้าไม่มีความสัมพันธ์ซึ่งกันและกันก็ไม่ควรอยู่ในฐานข้อมูลเดียวกัน

ในฐานข้อมูลนึง ๆ อาจบันทึกข้อมูลได้หลากหลายชนิด นอกเหนือจากข้อมูลที่เป็นข้อความ (text) หรือตัวเลข (numeric) ที่เห็นกันทั่วไปแล้ว ยังมีข้อมูลชนิดอื่น เช่น ข้อมูลมัลติมีเดีย (multimedia) อันหมายถึงรูปภาพคลิปวิดีโอ (video clip) เสียง (voice) ข้อมูลทางภูมิศาสตร์หรือเชิงพื้นที่ (spatial data) เช่น ข้อมูลภูมิอากาศ หรือภาพถ่ายดาวเทียม และอื่น ๆ อย่างไรก็ตาม ฐานข้อมูลแต่ละฐานควรเก็บข้อมูลที่เกี่ยวข้องกันอย่างชัดเจน

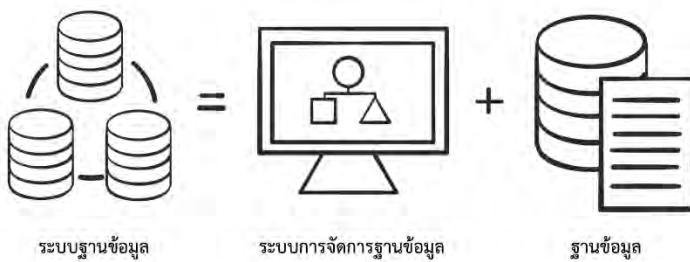
ส่วนการประมวลผลข้อมูลในฐานข้อมูลนั้นจะอาศัย ระบบการจัดการฐานข้อมูล (database management system: DBMS) [1] ซึ่งเป็นซอฟต์แวร์ที่มีความสามารถในการสร้างฐานข้อมูล จัดการฐานข้อมูล บำรุงรักษาข้อมูลและฐานข้อมูลในคอมพิวเตอร์ การจัดการฐานข้อมูลนั้นหมายรวมทั้งการบันทึก การสืบค้น การคำนวณ การปรับปรุงข้อมูล การจัดการผู้ใช้งาน การดูแลความปลอดภัยและอื่น ๆ ส่วนการบำรุงรักษานั้น มีตัวอย่างคือ การสำรองข้อมูล และการกู้คืน เป็นต้น

ซอฟต์แวร์ประเภทระบบการจัดการฐานข้อมูลนั้นมีผู้พัฒนาอยู่หลายบริษัท เช่น ออร่าเคิล (Oracle) หรือมาיהເອສົກີແວລ (MySQL) ที่ใช้กันอย่างแพร่หลายในองค์กรต่าง ๆ [1] อย่างไรก็ตาม หน่วยงานต่าง ๆ จะต้องพิจารณาเลือกใช้ผลิตภัณฑ์ที่เหมาะสมกับงานของตน โดยในเบื้องต้น สามารถเลือกได้จากลักษณะของข้อมูลที่แตกต่างกัน ซึ่งจะต้องมีการออกแบบและจัดเก็บต่างกันไป ดังนั้นจำเป็นต้องมีเครื่องมือการจัดเก็บที่เฉพาะเจาะจงด้วย เช่น หากมีข้อมูลแนวที่เป็นหลัก ก็ควรพิจารณาระบบการจัดการฐานข้อมูลประเภทระบบการจัดการฐานข้อมูลเชิงพื้นที่เป็นต้น

นอกจากจะเลือกผลิตภัณฑ์ระบบการจัดการฐานข้อมูลตามลักษณะข้อมูลแล้ว ผู้ใช้ยังควรพิจารณาความรวดเร็วในการตอบสนองของระบบด้วย เช่น ระบบที่ต้องการการทำงานแบบเวลาจริง (real-time) จะเป็นต้องใช้ระบบการจัดการฐานข้อมูลที่ออกแบบมาโดยเฉพาะ เช่น งานควบคุมเครื่องจักรหรืออ่านพาหนะที่ต้องกระทำการ (execute) แบบทันที

ส่วนคำว่า ระบบฐานข้อมูล (database system) [1] นั้น หมายถึงซอฟต์แวร์ระบบการจัดการฐานข้อมูล (DBMS) พร้อมกับฐานข้อมูล ซึ่งรวมถึงแอปพลิเคชันด้วย ดังนั้นเริกฯ ฯ ที่กล่าวมาก่อนหน้านี้ ควรเรียกให้ถูกต้องว่า ระบบฐานข้อมูลที่เปลี่ยนนิสิตฯ ฯ

จะเห็นได้ว่าระบบฐานข้อมูลนี้ จะประกอบด้วยระบบการจัดการฐานข้อมูล หนึ่งระบบกับฐานข้อมูลหลายฐาน รูปที่ 1 แสดงความสัมพันธ์ระหว่างฐานข้อมูล ระบบการจัดการฐานข้อมูล และระบบฐานข้อมูล



ระบบฐานข้อมูล

ระบบการจัดการฐานข้อมูล

ฐานข้อมูล

รูปที่ 1 ความสัมพันธ์ระหว่าง ฐานข้อมูล ระบบการจัดการฐานข้อมูล และระบบฐานข้อมูล

## ตัวอย่าง

สมมุติเราต้องการสร้างระบบฐานข้อมูลมหาวิทยาลัย (university database system) จะต้องเริ่มว่าระบบของเรามีความต้องการเก็บข้อมูลส่วนใดของมหาวิทยาลัย ในตัวอย่างนี้ ขอกำหนดว่าต้องการเก็บข้อมูลเกี่ยวกับวิชาและตอนเรียนที่นิสิตลงทะเบียนเรียน ก็จะได้ฐานข้อมูล 3 ฐาน ที่รองรับขอบเขตงานที่เราสนใจ ได้แก่ ฐานข้อมูลนิสิต ฐานข้อมูลวิชา และฐานข้อมูลตอนเรียน วิธีเขียนชื่อฐานข้อมูลนั้น นิยมเขียนด้วยตัวพิมพ์ใหญ่ ดังนั้น ขอตั้งชื่อฐานข้อมูลทั้ง 3 ฐาน ดังนี้

STUDENT

COURSE

SECTION

ระบบการจัดการฐานข้อมูลมีเครื่องมือแสดงโครงสร้างของข้อมูลแต่ละฐานให้ผู้ใช้งานเข้าใจได้ง่าย วิธีหนึ่งคือการแสดงข้อมูลในรูปแบบตาราง ดังตัวอย่าง ในรูปที่ 2

สำหรับระบบฐานข้อมูลขนาดใหญ่ มักมีผู้ใช้งานหลายคนพร้อมกัน สิ่งที่ต้องคำนึงถึงในการออกแบบคือการเข้าถึง (access) ที่เหมาะสม ตลอดจนการรักษาความปลอดภัยของข้อมูล และความถูกต้องของข้อมูล ในแง่ของการเข้าถึงข้อมูลนั้น ผู้ออกแบบระบบฐานข้อมูลจำต้องคำนึงถึงการออกแบบให้ระบบเสถียร แม้มีผู้ใช้งานที่เข้าถึงข้อมูลหลายคนพร้อมกัน และยังต้องคำนึงถึงความปลอดภัยให้ผู้ใช้งานเข้าถึงข้อมูลเฉพาะที่ได้รับสิทธิ์ด้วย นอกเหนือจากนั้น ในแง่ของความปลอดภัยของข้อมูลยังต้องจัดการเรื่องการสำรอง (backup) ข้อมูลเป็นระยะ เพื่อให้การคืนสภาพ หรือการนำกลับมาใช้

(recovery) สามารถทำได้สะดวกหากระบบเกิดความชำรุดข้อง (disaster) อีกด้วย

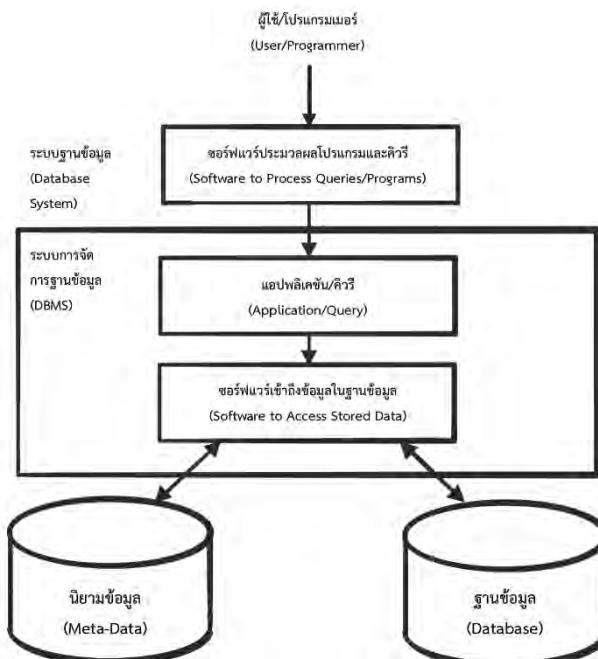
STUDENT	Name	StudentID	Section	Major	
	Kawin	5970105021	2110101	Computer Engineering	
	Chonlatorn	5970136421	2110101	Computer Engineering	
COURSE	CourseName	CourseNumber	Credits	Department	
	Discrete Structures and Computability	2110202	4	Computer Engineering	
	Introduction to Data Structures	2110211	3	Computer Engineering	
	Programming Methodology I	2110215	3	Computer Engineering	
SECTION	SectionIdentifier	CourseNumber	Semester	Year	Instructor
	85	2110202	First	2018	Proadpran
	92	2110211	First	2018	Atiwong
	102	2110215	Second	2018	Athasit

รูปที่ 2 ตารางฐานข้อมูลคอมพิวเตอร์ราย

## 1.2 ความสัมพันธ์ของระบบการจัดการฐานข้อมูลและระบบฐานข้อมูล

แนวคิดของความสัมพันธ์ของระบบการจัดการฐานข้อมูลและระบบฐานข้อมูลแสดงในรูปที่ 3 ซึ่งปรับมาจาก [2] แสดงความสัมพันธ์ของระบบการจัดการฐานข้อมูลและระบบฐานข้อมูลระดับแนวคิด เมื่อพิจารณารูปจากบนลงล่าง จะเริ่มจากผู้ใช้งานระบบฐานข้อมูล ซึ่งหมายรวมถึงผู้ใช้งานข้อมูล และโปรแกรมเมอร์ (programmer) โดยที่ผู้ใช้ข้อมูลมักใช้โปรแกรมสำหรับรูปหรือแอปพลิเคชันในการเรียกดูข้อมูล ส่วนโปรแกรมเมอร์มักเรียกใช้ข้อมูล

ผ่านภาษาคิวเรียร์หรือภาษาสอบถาม (query language) ในการสืบค้นข้อมูล เพื่อให้ได้ข้อคำถาม (query) จากจุดนี้จะเป็นหน้าที่ของระบบการจัดการฐานข้อมูล โดยคร่าวๆ แล้วระบบการจัดการฐานข้อมูลมีหน้าที่นำข้อคำถามนั้นๆ ไปประมวลผล แล้วส่งคำสั่งเพื่อทำให้รู้ว่าจะเข้าถึงข้อมูล หรือเรียกข้อมูลจากฐานข้อมูลผ่านนิยามข้อมูล (meta-data) ที่จัดเก็บไว้แยกกันอย่างไร ทั้งนี้ขึ้นให้สังเกตว่าขั้นตอนการดึงข้อมูลจากหน่วยเก็บฐานข้อมูล (stored database) นั้นไม่จำเป็นว่าจะต้องเป็นงานของระบบการจัดการฐานข้อมูล



รูปที่ 3 ระบบการจัดการฐานข้อมูลและระบบฐานข้อมูล

จะเห็นได้ว่าหน้าที่ของระบบการจัดการฐานข้อมูลคือส่วนกลางของรูปที่ 3 ส่วนรายละเอียดของความสามารถของระบบการจัดการฐานข้อมูลมีหลายอย่าง ขอกล่าวถึงความสามารถที่สำคัญ ได้แก่

1. สามารถกำหนดคุณลักษณะของฐานข้อมูล ตั้งแต่ชนิดของข้อมูล (data type) โครงสร้างของข้อมูล (structure) และเงื่อนไขบางคับ (constraint) ต่าง ๆ
2. สามารถสร้างฐานข้อมูลและบรรจุข้อมูลลงฐานข้อมูลในหน่วยเก็บรอง (secondary storage)
3. สามารถจัดดำเนินการ (manipulate) ฐานข้อมูล เช่น การแทรกข้อมูล (insertion) การลบ (deletion) การแก้ไข (modification) การสอบถาม และการจัดทำรายงาน (report)
4. สามารถประมวลผลพร้อมกันและใช้ข้อมูลร่วมกัน (concurrent processing and sharing) ให้สามารถมีผู้ใช้งานหลายคนพร้อมกันได้ มีโปรแกรมหลายโปรแกรมทำงานพร้อมกันได้ โดยที่ข้อมูลยังคงความถูกต้อง
5. สามารถจัดการป้องกันและตรวจสอบความปลอดภัย ผ่านทางการตรวจสอบการเข้าถึงที่ไม่ได้รับการอนุญาต (unauthorized access) และจัดการด้านความมั่นคงของข้อมูลเมื่อซอฟต์แวร์หรือฮาร์ดแวร์มีปัญหา
6. สามารถทำการปรับปรุงและบำรุงรักษาฐานข้อมูล (maintenance) และปรับเปลี่ยนระบบไปตามความต้องการที่เปลี่ยนไป

### 1.3 คุณสมบัติของฐานข้อมูล

ข้อมูลที่เก็บไว้ในฐานข้อมูลเดียวกันควร้มีความเกี่ยวข้องกันและมีความหมายในตัวเอง ฐานข้อมูลแต่ละฐานจะต้องถูกออกแบบแล้วสร้างขึ้นตามแบบเพื่อใช้บรรจุข้อมูลสำหรับวัตถุประสงค์อันใดอันหนึ่งที่แน่ชัด ฐานข้อมูลสามารถเป็นตัวแทนเหตุการณ์หรือเป็นแบบจำลองของปัญหาจริงหรือส่วนหนึ่งของปัญหาจริงที่ข้อมูลในฐานข้อมูลนั้นเกี่ยวข้อง มีขอบเขตแน่นชัด เช่น ฐานข้อมูลการลงทะเบียนของนิสิตในมหาวิทยาลัย จะมีขอบเขตคือ ข้อมูลเกี่ยวกับการลงทะเบียนเรียนของนิสิต ชื่อนามสกุlnิสิต ปีที่นิสิตเข้าเรียน สาขาวิชาที่นิสิตเรียน วิชาที่ต้องการลงทะเบียน ปีการศึกษาที่ลงทะเบียน จำนวนนิสิตของแต่ละวิชา ห้องเรียน ตำราเรียน อาจารย์ผู้สอน อาจารย์ที่ปรึกษา และเกรด เป็นต้น ขอบเขตของฐานข้อมูลนี้ไม่มีความซ้ำซ้อนอื่นใดที่ไม่เกี่ยวข้อง เช่น ข้อมูลเงินเดือนของอาจารย์ ข้อมูลตำแหน่งเจ้าหน้าที่ ข้อมูลห้องสือในห้องสมุด เป็นต้น

ฐานข้อมูลที่ดีจะต้องสามารถจัดเก็บและจัดการข้อมูลให้มีข้อมูลที่ถูกต้อง ทันกาลอยู่เสมอ ต้องไม่มีความซ้ำซ้อนกันเอง มีความซ้ำซ้อน (redundancy) น้อยที่สุด และหากจำเป็นต้องซ้ำซ้อนก็ต้องมีการจัดการให้ข้อมูลถูกต้องทั่วถึง กันทั่วระบบ สามารถเรียกข้อมูลมาใช้งานได้ในเวลาอันเหมาะสมกับงานนั้น ๆ

### 1.4 คุณสมบัติของระบบการจัดการฐานข้อมูล

คุณสมบัติของระบบการจัดการฐานข้อมูลมีหลายประการ จาก [1] ได้ระบุที่สำคัญ ได้แก่

1. ผู้ใช้งานสามารถเข้าใจข้อมูลได้ง่าย เพราะระบบการจัดการฐานข้อมูลจะทำสารบัญหรือแค็ตตาล็อก (catalog) ข้อมูลในระบบโดยเก็บในรูปแบบนิยามข้อมูล ทำให้ผู้ใช้งานสามารถเข้าใจความหมายของทุกข้อมูลได้เหมือน ๆ กัน โดยคำอธิบายนั้นประกอบด้วยโครงสร้างของข้อมูล ชนิดรูปแบบของข้อมูล พร้อมทั้งข้อบังคับต่าง ๆ ถ้ามี โดยระบบการจัดการฐานข้อมูลจะแยกเก็บนิยามข้อมูลไว้แยกจากหน่วยเก็บข้อมูลและแยกจากแอปพลิเคชัน จึงเป็นคุณสมบัติสำคัญที่ทำให้เราสามารถนำข้อมูลชุดเดิมไปใช้ในแอปพลิเคชันต่างกันได้ เช่น เรายสามารถใช้ฐานข้อมูลนิสิตในระบบฐานข้อมูลศิษย์เก่าได้ แม้ว่าระบบฐานข้อมูลศิษย์เก่าจะแยกจากระบบฐานข้อมูลมหาวิทยาลัย เป็นต้น
2. มีความอิสระระหว่างแอปพลิเคชันกับข้อมูล หมายความว่าระบบฐานข้อมูลจะอนุญาตให้เปลี่ยนแปลงโครงสร้างของหน่วยเก็บข้อมูล และการจัดการต่าง ๆ โดยไม่จำเป็นต้องเปลี่ยนแอปพลิเคชัน เนื่องจากโครงสร้างของไฟล์ข้อมูลนั้นถูกแยกออกจากแอปพลิเคชัน ซึ่งไม่เหมือนกระบวนการประมวลผลไฟล์ (file processing) แบบดั้งเดิมที่โครงสร้างของไฟล์ข้อมูลจะถูกผังติดกับแอปพลิเคชัน จึงต้องทำการเปลี่ยนแปลงโปรแกรมทั้งหมดเมื่อต้องการเปลี่ยนโครงสร้างไฟล์
3. มีความอิสระระหว่างโปรแกรมกับการจัดการ กรณีจะเกิดเมื่อใช้ฐานข้อมูลเชิงวัตถุ (object-oriented database) ซึ่งผู้ใช้สามารถระบุการจัดการหรือโอเพอเรชัน (operation) ให้กับข้อมูล โดยที่โอเพอเรชันประกอบด้วยอินเทอร์เฟซ (interface) และ อิมพลิเมนเทชัน (implementation) คุณสมบัติข้อนี้คือเมื่อมีการเปลี่ยนแปลงส่วนอิมพลิเมนเทชันจะไม่กระทบต่ออินเทอร์เฟซ เราเรียกคุณสมบัตินี้ว่า โปรแกรม

กับการจัดการนั้นแยกกันได้หรือมีอิสระต่อกัน (program-operation independence) นั่นเอง

4. มีการใช้แบบจำลองข้อมูล (data model) ในระบบการจัดการฐานข้อมูล ทำให้ผู้ใช้สามารถกำหนดสาระสำคัญของข้อมูล (data abstraction) ได้ง่ายทั้งที่ข้อมูลอาจมีความสัมพันธ์ที่ซับซ้อน ซึ่งมีผลทำให้ผู้ใช้งานสามารถทำงานผ่านโมเดลระดับแนวคิด (conceptual model) โดยชื่นรายละเอียดของหน่วยเก็บข้อมูลที่ซับซ้อนไว้ จากนั้นระบบการจัดการฐานข้อมูลจะทำหน้าที่ในการดึงข้อมูลได้อย่างถูกต้องโดยใช้รายละเอียดจากแค็ตตาล็อกและนิยามข้อมูลเป็นพอ
5. มีการรองรับมุมมองสำหรับผู้ใช้ต่างกันได้ กล่าวคือ ผู้ใช้แต่ละคนหรือแต่ละกลุ่มสามารถเรียกใช้ข้อมูลด้วยมุมมอง (view) ที่แตกต่างกัน แต่ละมุมมองจะทำให้ผู้ใช้เข้าถึงส่วนของข้อมูลที่ได้รับอนุญาตได้อย่างสะดวกและสร้างความปลอดภัยต่อฐานข้อมูลด้วย
6. มีการจัดการการใช้ข้อมูลร่วมกัน (sharing) และประมวลผลรายการเมื่อมีผู้ใช้หลายคนพร้อมกัน (multiuser transaction processing) เพราะในระบบฐานข้อมูลใด ๆ มักมีผู้ใช้หลายคนพร้อมกันซึ่งอาจมีสิทธิ์ในการแก้ไขข้อมูลได้ ดังนั้นระบบการจัดการฐานข้อมูลจึงจำเป็นต้องรองรับการทำงานแบบพร้อมกันได้อย่างมีประสิทธิภาพ เช่น การจัดลำดับการทำการ (transaction) อย่างถูกต้องจนครบสมบูรณ์ ซึ่งนับว่าเป็นส่วนสำคัญมากสำหรับแอปพลิเคชันประเภทออนไลน์ ทั้งยังเป็นการป้องกันการเข้าถึงข้อมูลโดยไม่ได้รับอนุญาตอีกด้วย
7. สามารถควบคุมความซ้ำซ้อน ในการจัดเก็บข้อมูล ซึ่งความซ้ำซ้อนของข้อมูลนั้นเป็นสาเหตุของปัญหามากมาย เช่น สิ้นเปลืองพื้นที่ในการเก็บข้อมูล หรือข้อมูลที่เก็บไว้ไม่ตรงกัน

8. มีการจัดการการสอบถาม ได้อย่างมีประสิทธิภาพ เช่น สามารถกำหนดตัวนี้ (index) เพื่อให้เข้าถึงข้อมูลที่ใช้บ่อยได้รวดเร็วขึ้น
9. มีความสามารถในการสำรองและกู้คืนหากระบบฐานข้อมูลมีปัญหาหรือล่ม (disaster)
10. สามารถระบุเงื่อนไขบังคับต่าง ๆ ได้หลายรูปแบบและหลายลำดับที่ช่วยลดความผิดพลาดของข้อมูล เช่น บังคับให้ข้อมูลต้องไม่ซ้ำด้วยข้อบังคับของคีย์ (key constraint) ข้อมูลต้องมีข้อบังคับความเป็นบูรณาภพ (integrity constraint) เป็นต้น
11. มีศักยภาพในการควบคุมมาตรฐาน โดยเฉพาะการทำแอปพลิเคชันฐานข้อมูลในองค์กรขนาดใหญ่ที่มีโปรแกรมเมอร์จำนวนมากที่สร้างหรือใช้ฐานข้อมูลเดียวกัน เพราะทำให้ทุกคนได้เข้าใจข้อมูลเหมือน ๆ กันได้ด้วยการใช้พจนานุกรมข้อมูล (data dictionary)

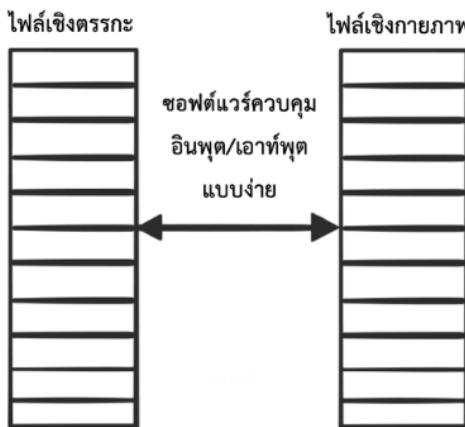
## 1.5 วิัฒนาการของระบบการจัดการฐานข้อมูล

แนวคิดของระบบการจัดการฐานข้อมูลมีมาแต่โบราณ สามารถแบ่งได้เป็น 5 ช่วง โดยช่วงที่ 1-4 ถูกอิงจาก [4, 5] และช่วงที่ 5 คือ ยุคปัจจุบัน มีรายละเอียดดังนี้

- ช่วงที่ 1 : ยุคการจัดการข้อมูลแบบไฟล์
- ช่วงที่ 2 : ยุคการจัดการข้อมูลแบบการเข้าถึงไฟล์
- ช่วงที่ 3 : ยุคระบบฐานข้อมูลในช่วงแรกเริ่ม
- ช่วงที่ 4 : ยุคระบบฐานข้อมูลเชิงสัมพันธ์
- ช่วงที่ 5 : ยุคปัจจุบัน

### 1.5.1 ช่วงที่ 1 : ยุคการจัดการข้อมูลแบบไฟล์

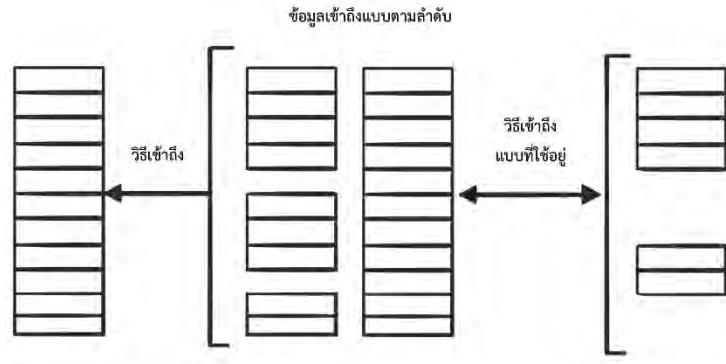
คือการจัดระบบไฟล์แบบลำดับอนุกรม (serial) หมายถึง โครงสร้างข้อมูลทางกายภาพ (physical data structure) มีลำดับคู่กับโครงสร้างข้อมูลเชิงตรรกะ (logical data structure) เสมอ และใช้กระบวนการประมวลผลแบบกลุ่ม (batch-processing) ทำให้ไม่สามารถเข้าถึงข้อมูลแบบเวลาจริงได้ จะเห็นได้ว่าการจัดเก็บไฟล์จะเกิดการซ้ำซ้อนโดยปริยาย โปรแกรมเมอร์จำเป็นต้องเขียนโปรแกรมสำหรับการใช้ข้อมูลชนนี้ ๆ โดยเฉพาะ เมื่อมีการเปลี่ยนแปลงโครงสร้างข้อมูลหรือเปลี่ยนแปลงโปรแกรมใด ๆ โปรแกรมเมอร์จะต้องเขียนโปรแกรมใหม่และจัดเรียงข้อมูลใหม่เสมอ เกิดการซ้ำซ้อนระหว่างไฟล์ข้อมูลอย่างมาก รูปที่ 4 แสดงความสัมพันธ์ของข้อมูลแบบไฟล์ หรือแบบลำดับอนุกรมนี้



รูปที่ 4 การจัดการข้อมูลแบบไฟล์

### 1.5.2 ช่วงที่ 2 : ยุคการจัดการข้อมูลแบบการเข้าถึงไฟล์

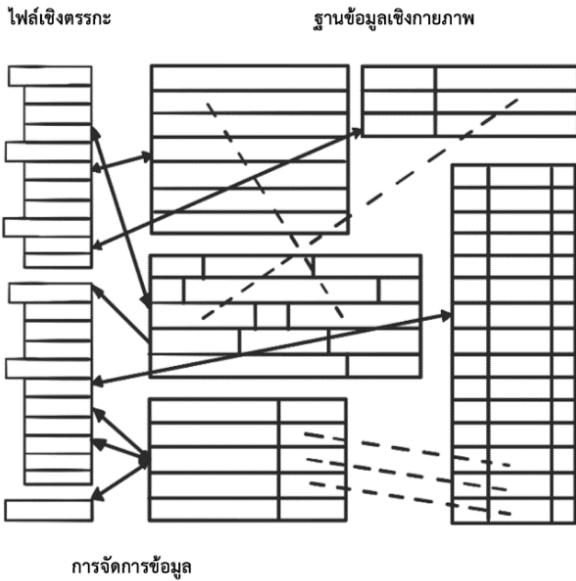
ยุคการจัดการข้อมูลแบบการเข้าถึงไฟล์ (File Access Method) ดังแสดงในรูปที่ 5 เกิดขึ้นในช่วงปลาย ค.ศ. 1960 ได้มีการเสนอการเข้าถึงข้อมูลแบบตามลำดับ (Serial access) หรือการเข้าถึงข้อมูลแบบสุ่ม (Random access) ซึ่งทำได้ด้วยการแยกไฟล์เชิงตรรกะ (Logical file) และไฟล์ข้อมูลจริง (Physical file) ออกจากกัน ส่วนการประมวลผลนั้น มีทั้งการประมวลผลแบบเดิม คือแบบกลุ่ม (Batch) และมีแบบใหม่เกิดขึ้น คือแบบอินไลน์ (Inline) ซึ่งนับเป็นการประมวลผลแบบเวลาจริง ด้วยวิธีนี้ ทำให้โปรแกรมเมอร์สามารถเปลี่ยนแปลงหน่วยเก็บข้อมูลได้โดยไม่จำเป็นต้องแก้ไขแอปพลิเคชัน การจัดโครงสร้างข้อมูลอาจอยู่ในรูปแบบลำดับอนุกรม ลำดับเชิงตัวเลข (Index sequential) หรือเข้าถึงข้อมูลตรง ๆ ได้ (Simple direct access) ในยุคนี้ ยังไม่มีการใช้คีย์หลายตัว (Multiple-key) ในการสืบค้นข้อมูล แม้จะเริ่มมีการตรวจสอบความปลอดภัยของข้อมูล แต่นับว่ายังไม่ปลอดภัยนัก ยังคงพบความซ้ำซ้อนของข้อมูลจำนวนมากอยู่



รูปที่ 5 การจัดการข้อมูลแบบการเข้าถึงไฟล์

### 1.5.3 ช่วงที่ 3 : ยุคระบบฐานข้อมูลในช่วงแรกเริ่ม

ตั้งแต่ช่วงต้น ค.ศ. 1970 ได้มีการสร้างไฟล์เชิงตรรกะมากกว่าหนึ่งไฟล์ ไฟล์เชิงตรรกะนี้จะมีความแตกต่างกัน เพื่อให้เข้าถึงข้อมูลเชิงกายภาพ (physical data) หรือเรียกว่าข้อมูลจริงๆ ดูเดียวกันได้ ดังแสดงในรูปที่ 6 นับเป็นการรองรับให้แอปพลิเคชันต่างกันสามารถเข้าถึงข้อมูลชุดเดียว กัน ส่งผลให้ลดความซ้ำซ้อนของข้อมูลและทำให้เกิดบูรณาพของข้อมูล (data integrity) และการแก้ไขโปรแกรมสามารถทำได้โดยไม่ต้องปรับแก้ข้อมูลตาม วิธีการที่เกิดขึ้นในยุคนี้ทำให้สามารถเข้าถึงข้อมูลได้ทั้งระดับเขตข้อมูลหรือฟิลด์ (field) หรือในระดับกลุ่ม (group) ได้ และยังรองรับการสืบค้นข้อมูลด้วยมัลติเพลคิร์ หรือคิร์หลายตัวได้ด้วย ทำให้มีการจัดระบบข้อมูลที่ซับซ้อนโดยไม่ทำให้โปรแกรมแอปพลิเคชันซับซ้อนได้



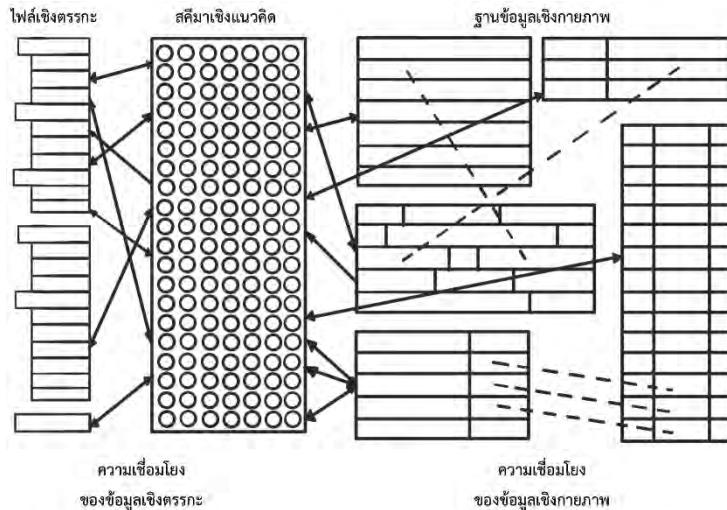
รูปที่ 6 ระบบฐานข้อมูลในช่วงแรกเริ่ม

#### 1.5.4 ช่วงที่ 4 : ระบบฐานข้อมูลเชิงสัมพันธ์

ในยุคนี้ ระบบการจัดการฐานข้อมูลมีการแยกข้อมูลเชิงตรรกะ และข้อมูลจริง ออกจากกันอย่างอิสระ ดังแสดงในรูปที่ 7 ทำให้สามารถพัฒนาระบบฐานข้อมูลได้ง่ายขึ้นมาก อีกทั้งยังเพิ่มความสามารถในการติดตาม (monitor) การบำรุงรักษา การจัดการความปลอดภัย และความสามารถอื่น ๆ ดังได้กล่าวไว้ก่อนหน้านี้ รูปแบบของระบบฐานข้อมูลแบบนี้เรียกว่า ระบบฐานข้อมูลเชิงสัมพันธ์ (relational database system) ซึ่งให้ความสำคัญกับความสัมพันธ์ของข้อมูลในระบบเป็นหลัก เกิดจากบริษัท

ไอปีเอ็มในช่วงต้น ค.ศ. 1970 และตามมาด้วยระบบการจัดการฐานข้อมูลเชิงสัมพันธ์ (relational DBMS) ตั้งแต่ช่วงทศวรรษที่ 1980

จากนั้น มีผู้เสนอระบบการจัดการฐานข้อมูลเชิงวัตถุขึ้นในช่วงปลายทศวรรษที่ 1990 เพื่อตอบสนองการเขียนโปรแกรมด้วยภาษาเชิงวัตถุ (object-oriented language) แต่การใช้งานไม่เป็นที่นิยมเท่ากับระบบการจัดการฐานข้อมูลเชิงสัมพันธ์ที่เป็นที่นิยม ได้แก่ ออราเคิล มาיהเอสคิวแอล และ เอสคิวแอลเซิร์ฟเวอร์ เป็นต้น นอกจากนั้น ยังมีผลิตภัณฑ์ที่ออกแบบเพื่อรับข้อมูลที่นокเนื้อจากตัวอักษรและตัวเลข เพื่อให้รองรับข้อมูลภาพนิ่ง ภาพเคลื่อนไหว ข้อมูลเสียง ข้อมูลเชิงพื้นที่ และอื่น ๆ



รูปที่ 7 ระบบฐานข้อมูลเชิงสัมพันธ์

### 1.5.5 ช่วงที่ 5 : ยุคปัจจุบัน

เมื่อ ค.ศ. 1998 ได้มีผู้เสนอแนวคิดฐานข้อมูลแบบโนอีสคิวแอล (NoSQL หรือ Not-Only-SQL) ที่ทำให้เกิดการเปลี่ยนแปลงครั้งใหญ่ในการฐานข้อมูล ตัวอย่างระบบการจัดการฐานข้อมูลแบบโนอีสคิวแอล ได้แก่ mongoDB (MongoDB) Redis (Redis) คัสชานดรา (Cassandra) เป็นต้น ซึ่งแต่ละผลิตภัณฑ์ยังมีรายละเอียดต่างกันอีกด้วย โดยรวมแล้วแนวคิดแบบโนอีสคิวแอลนี้จะเหมาะสมกับข้อมูลที่ไม่มีโครงสร้างแน่นชัด และเหมาะสมกับข้อมูลขนาดใหญ่ในยุคบิ๊กดาต้า (big data) เช่น ข้อมูลในสังคมออนไลน์ เป็นต้น อย่างไรก็ตี ความนิยมในระบบฐานข้อมูลเชิงสัมพันธ์ยังสูงกว่าฐานข้อมูลแบบโนอีสคิวแอลอยู่มาก

## 1.6 บทสรุปและเรื่องชวนคิด

ถึงแม้การใช้ระบบการจัดการฐานข้อมูลจะมีข้อดีมากมายและเป็นที่นิยมแพร่หลาย แต่การเลือกใช้ระบบการจัดการฐานข้อมูลก็ต้องพิจารณาอย่างถี่ถ้วนก่อนจะใช้ เนื่องจากการใช้ระบบการจัดการฐานข้อมูลมีค่าใช้จ่ายสูง เพราะมักต้องการการลงทุนทางยาgard แวร์และซอฟต์แวร์ค่อนข้างสูง ดังนั้น สำหรับกรณีที่ฐานข้อมูลและแอปพลิเคชันไม่ซับซ้อนมากนัก หรือมีการเปลี่ยนแปลงน้อย หรือมีผู้ใช้แค่คนเดียว ก็ไม่มีความจำเป็นได ๆ ที่ต้องใช้ระบบการจัดการฐานข้อมูล

สำหรับองค์กรขนาดใหญ่ คงหลีกเลี่ยงการใช้ฐานข้อมูลไม่ได้ แต่ด้วยรูปแบบของข้อมูลสมัยใหม่ที่ไม่มีโครงสร้างที่แน่นอน ผู้เรียนคิดว่าฐานข้อมูลเชิงสัมพันธ์ที่ได้รับความนิยมมาาวา 50 ปี จะถึงวันหมดอายุหรือยัง

## แบบฝึกหัดท้ายบท

1. จงยกตัวอย่างโครงการที่พับเห็นและควรมีฐานข้อมูลในการทำงานนั้น ๆ มา 3 โครงการ พร้อมยกตัวอย่างฐานข้อมูลที่ฐานข้อมูลนั้นควรมี
2. จงอธิบายความแตกต่างระหว่างระบบฐานข้อมูลและระบบการจัดการฐานข้อมูล
3. จงเปรียบเทียบความแตกต่างของฐานข้อมูลยุคการจัดการข้อมูลแบบไฟล์ กับยุคปัจจุบัน
4. เมื่อได้ถึงควรเลือกใช้ระบบการจัดการฐานข้อมูลแบบโนเอสคิวแอล
5. จงยกตัวอย่างสิ่งที่ควรคำนึงในการออกแบบฐานข้อมูล
6. หากมีโปรแกรมเมอร์ทำงานติดต่อกับฐานข้อมูลเดียวกันเป็นจำนวนมาก สิ่งใดช่วยให้ทุกคนสามารถทำงานร่วมกันได้โดยเข้าใจฐานข้อมูลตรงกัน
7. จงบอกคุณสมบัติของข้อมูลภายในฐานข้อมูลเดียวกัน
8. “ระบบการจัดการฐานข้อมูลหนึ่ง ๆ สามารถเก็บข้อมูลได้หลายชนิด ดังนั้นจึงสามารถใช้กับงานที่มีข้อมูลลักษณะใดก็ได้” จากข้อความข้างต้น ผู้เรียนเห็นด้วยหรือไม่ จงอภิปราย
9. ภาษาสอบถาม มีหน้าที่อะไรในระบบฐานข้อมูล
10. สิ่งใดทำให้หลายแอปพลิเคชันสามารถประมวลผลและใช้ข้อมูลร่วมกันได้

## บทที่ 2

### การพัฒนาระบบฐานข้อมูล

#### วัตถุประสงค์

- เพื่อให้เข้าใจขั้นตอนการพัฒนาระบบฐานข้อมูล
- เพื่อให้รู้จักสถาปัตยกรรมข้อมูลแบบต่าง ๆ
- เพื่อให้รู้จักภาษาฐานข้อมูล
- เพื่อให้รู้จักภาษาที่ใช้ในการจัดการฐานข้อมูล
- เพื่อให้รู้จักรอบการจัดการฐานข้อมูลประเภทต่าง ๆ

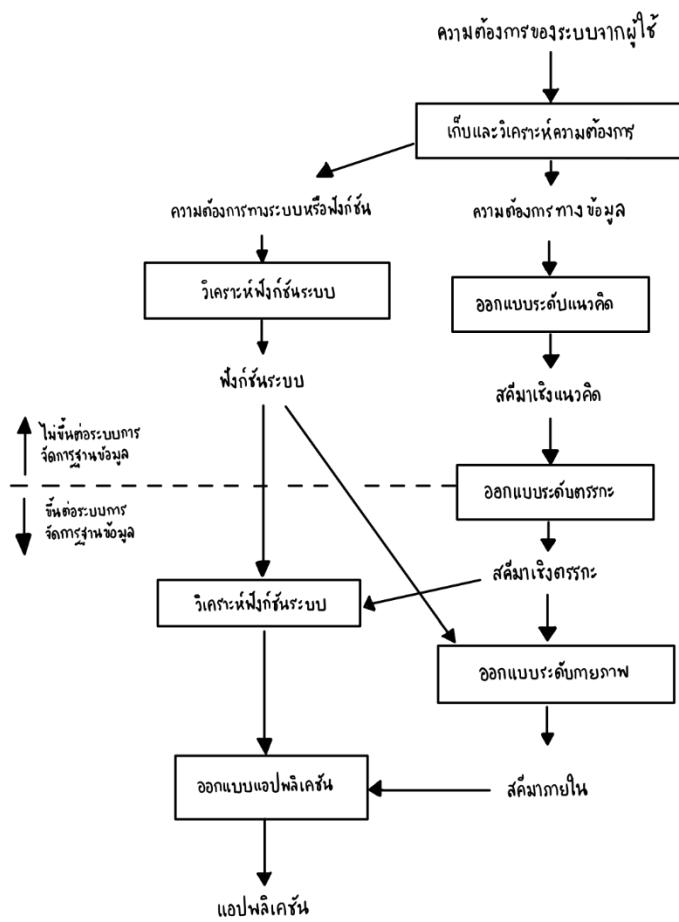
การพัฒนาระบบฐานข้อมูลมักเกิดควบคู่กับการพัฒนาระบบหรือแอปพลิเคชันที่ใช้ข้อมูลนั้น ๆ เพื่อตอบโจทย์จากเจ้าของงาน เนื่องจากแอปพลิเคชันส่วนใหญ่โดยเฉพาะในธุรกิจจำเป็นที่จะต้องใช้ฐานข้อมูลเป็นส่วนหนึ่งในการทำงานเสมอ

กระบวนการทางวิศวกรรมซอฟต์แวร์ [6-8] ระบุว่ากระบวนการสร้างซอฟต์แวร์หรือแอปพลิเคชัน (ต่อไปนี้จะเรียกว่า แอปพลิเคชัน) มักเริ่มจาก การเก็บข้อมูลความต้องการทางระบบจากผู้ใช้ และนำไปวิเคราะห์ความต้องการ ออกแบบ เอกซ์เพรสเซนต์ ทดสอบ และวิจัยจนกระบวนการ

ขอนเน้นที่ขั้นตอนการวิเคราะห์ความต้องการ ซึ่งจะได้ผลการวิเคราะห์เป็น 2 ประเด็นหลัก ได้แก่ ความต้องการทางระบบหรือฟังก์ชัน และความต้องการทางข้อมูล

สำหรับแอปพลิเคชันที่ต้องใช้ฐานข้อมูล ก็จะต้องดำเนินการออกแบบฐานข้อมูลควบคู่กันไป โดยพิจารณาว่าแอปพลิเคชันนั้นจำเป็นต้องใช้ข้อมูลอะไรในขั้นตอนใดบ้าง

รูปที่ 8 แสดงความสัมพันธ์ระหว่างการพัฒนาแอปพลิเคชัน กับการพัฒนาระบบฐานข้อมูล ซึ่งเป็นสองกระบวนการที่มักต้องทำควบคู่กัน

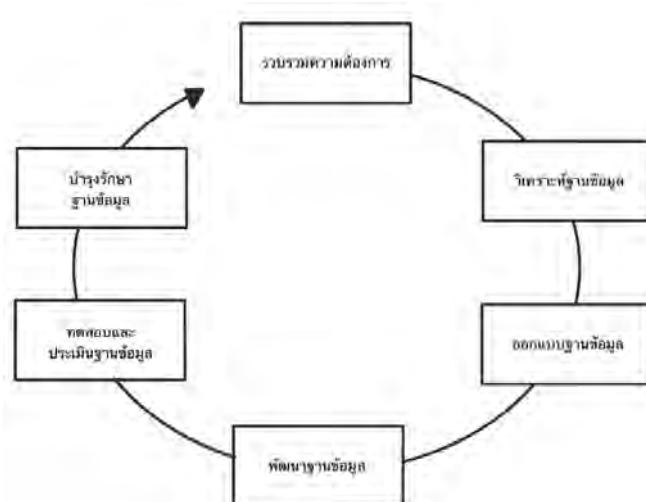


รูปที่ 8 ความสัมพันธ์ระหว่างการพัฒนาแอปพลิเคชัน กับการพัฒนาระบบฐานข้อมูล

## 2.1 การพัฒนาระบบฐานข้อมูล

ขั้นตอนการพัฒนาระบบฐานข้อมูล [1, 9-14] แสดงในรูปที่ 9 ประกอบด้วย 6 ขั้นตอนคือ

1. รวบรวมความต้องการ (requirement gathering)
2. วิเคราะห์ฐานข้อมูล (database analysis)
3. ออกแบบฐานข้อมูล (database design)
4. พัฒนาฐานข้อมูล (database implementation)
5. ทดสอบและประเมินฐานข้อมูล (database testing and evaluation)
6. บำรุงรักษาฐานข้อมูล (database maintenance)



รูปที่ 9 ขั้นตอนการพัฒนาระบบฐานข้อมูล

ขันตอนทั้งทั้งนี้ สามารถนำมาใช้ในการออกแบบระบบฐานข้อมูลได้ สำหรับ ตำรานี้ เราจะเน้นขันตอนที่ 3 คือการออกแบบฐานข้อมูล ซึ่งจะกล่าวโดย ละเอียดต่อไป

อย่างไรก็ตี การออกแบบฐานข้อมูลยังต้องคำนึงถึงระบบคอมพิวเตอร์ที่จะ ใช้งานฐานข้อมูลนั้นเป็นหลักด้วย เพราะสถาปัตยกรรมของระบบ คอมพิวเตอร์ที่ต่างยุคกันนั้นจะส่งผลถึงการทำงานร่วมกันระหว่างระบบ ทั้งหมดกับระบบฐานข้อมูล เราจึงต้องรู้จักสถาปัตยกรรมระบบฐานข้อมูล เพื่อให้เข้าใจว่าระบบทั้งหมดมีการติดต่อหรือทำงานร่วมกันกับระบบ ฐานข้อมูลอย่างไร

## 2.2 สถาปัตยกรรมระบบฐานข้อมูล

ยุคสมัยของคอมพิวเตอร์มีการเปลี่ยนแปลงอย่างรวดเร็ว สามารถเห็น ความแตกต่างได้อย่างชัดเจน ทั้งในเรื่องขนาดและความสามารถในการ ประมวลผล เริ่มจากคอมพิวเตอร์ระบบเมนเฟรม (mainframe) ที่มี ขนาดใหญ่มากจนถึงโทรศัพท์แบบสมาร์ตโฟนที่ใช้กันอย่างแพร่หลายทุกวันนี้

ในยุคแรก ๆ ที่เริ่มใช้แนวคิดแบบฐานข้อมูลในการทำแอปพลิเคชัน ไฮร์ดแวร์ ทางคอมพิวเตอร์ที่ใช้ในองค์กรสมัยนั้นมีขนาดใหญ่ คือระบบเมนเฟรมที่ทำการ ประมวลผลแบบศูนย์รวม จากนั้น เมื่อมีความเปลี่ยนแปลงทางระบบ ไฮร์ดแวร์พร้อมทั้งมีระบบปฏิบัติการหลากหลายขึ้นเพื่อรองรับขนาดและ รูปแบบของธุกรรมที่ต่างกัน ทำให้การออกแบบนิດโมดูลา (modular design) เกิดขึ้น นั่นคือการแบ่งการประมวลผลเป็นส่วน ๆ แทนการ ประมวลผลแบบรวมศูนย์ในระบบเมนเฟรม ต่อมา ก็เกิดสถาปัตยกรรมแบบ

ลูกที่อยู่ในเครือข่าย (client/server) ที่เป็นที่นิยมอย่างแพร่หลาย พร้อมกับคอมพิวเตอร์ส่วนบุคคล (personal computer) มีความสามารถในการประมวลผลสูงขึ้นเรื่อยๆ จึงมีการเชื่อมต่อระหว่างคอมพิวเตอร์ส่วนบุคคลไปยังเครื่องแม่ข่าย (server machine) เพื่อจัดให้การประมวลผลเร็วขึ้น ทั้งนี้องค์กรทั้งหลายจะต้องประเมินพื้นที่สำหรับการจัดเก็บข้อมูลให้เหมาะสมกับปริมาณงานอย่างมีประสิทธิภาพ

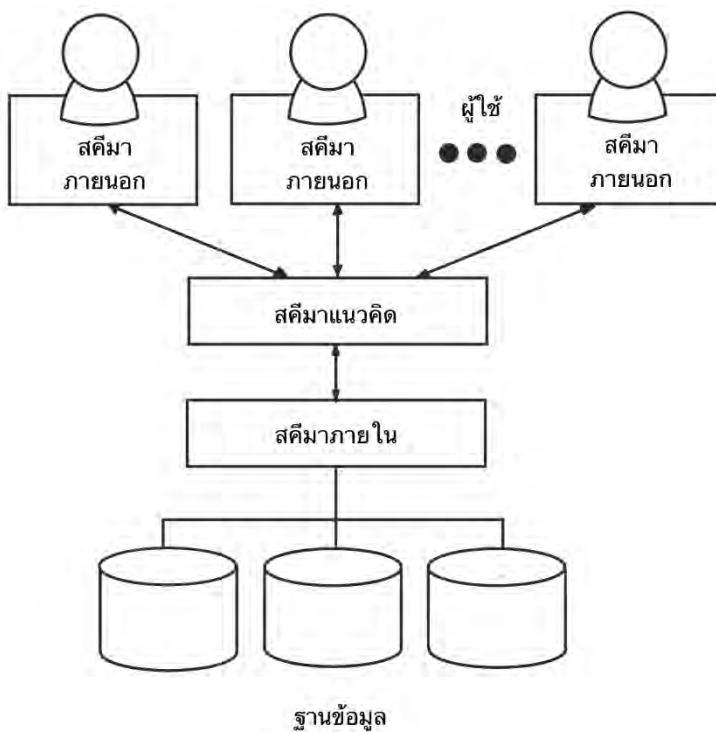
ในปัจจุบัน องค์กรขนาดใหญ่มีทางเลือกที่จะไม่เป็นเจ้าของแม่ข่าย (server) โดยอาจเลือกใช้บริการคลาวด์ (cloud service) [15-17] ซึ่งมีความยืดหยุ่นสูง โดยเฉพาะเรื่องพื้นที่การจัดเก็บข้อมูล ตลอดจนการให้บริการในการดูแลข้อมูล ทำให้ทางเลือกนี้เป็นที่นิยมอย่างมาก

สถาปัตยกรรมระบบฐานข้อมูล คือรูปแบบโครงสร้างและองค์ประกอบระดับต่างๆ ที่ประกอบเป็นระบบฐานข้อมูล มีหลายแบบ ในตำนานี้จะอธิบายสถาปัตยกรรม 4 แบบ ได้แก่ 1) สถาปัตยกรรมแบบทรีสคีมา 2) สถาปัตยกรรมแบบรวมศูนย์ 3) สถาปัตยกรรมแบบลูกข่ายแม่ข่าย 4) สถาปัตยกรรมแบบคลาวด์

### 2.2.1 สถาปัตยกรรมแบบทรีสคีมา

สถาปัตยกรรมแบบทรีสคีมา (three-schema architecture) ดังแสดงในรูปที่ 10 ซึ่งนำมาจาก [2] ประกอบด้วยสคีมา (schema) 3 ระดับ ซึ่งคำว่าสคีมาในภาษาไทยแปลว่า “เค้าร่าง” อธิบายได้ว่าเป็นวิธีเสนอหรือแสดงความคิดบางอย่าง

สถาปัตยกรรมแบบทรีสคีเม่า ออกแบบมาให้มีความสามารถในการรองรับการจัดการให้ข้อมูลมีความอิสระ เช่น สามารถแก้ไขโปรแกรมโดยไม่ต้องปรับแก้ฐานข้อมูล และสามารถแก้ไขโปรแกรมโดยไม่กระทบกับการดำเนินการ เป็นต้น นอกจากนี้สถาปัตยกรรมแบบทรีสคีเมามีการรองรับมุมมองหรือวิวของข้อมูลได้จำนวนมาก เพื่อให้ผู้ใช้แต่ละประเภทหรือแต่ละคนสามารถเรียกใช้หรือแก้ไขข้อมูลได้ตามความเหมาะสมสมอิกด้วย



รูปที่ 10 สถาปัตยกรรมแบบทรีสคีเม่า

สถาปัตยกรรมแบบทรีสคีมาແຍກเป็น 3 ระดับ ได้แก่ ระดับภายนอก (external level) ระดับแนวคิด (conceptual level) และระดับภายในใน (internal level) ในการทำงานของระบบการจัดการฐานข้อมูลนั้น จะเริ่มด้วยการแปลงความต้องการ (request) ที่ระบุไว้ที่สคีมาภายนอก ไปเป็นสคีมาแนวคิด และแปลงเป็นสคีมาภายในเพื่อดำเนินการกับฐานข้อมูลที่ถูกเก็บไว้ต่อไป กระบวนการดังกล่าวนี้เรียกว่า การแมป (mapping) หรือการแปลง โดยมีรายละเอียดดังนี้

สคีมาภายนอก – ใช้บรรยายโครงสร้างหน่วยเก็บข้อมูลทางกายภาพของข้อมูลทั้งหมดโดยใช้แบบจำลองเชิงกายภาพ มีการบรรยายรายละเอียดเกี่ยวกับหน่วยเก็บข้อมูลและเส้นทางการเข้าถึงฐานข้อมูลโดยสมบูรณ์

สคีมาแนวคิด – ใช้บรรยายโครงสร้างของฐานข้อมูลทั้งหมดในระดับแนวคิดไม่มีข้อมูลเชิงกายภาพปรากฏในสคีมานี้ เน้นการบรรยายกลุ่มข้อมูล ความสัมพันธ์ระหว่างกลุ่มข้อมูล การดำเนินการที่ข้อมูลนั้นทำได้ และข้อบังคับต่าง ๆ โดยใช้แบบจำลองข้อมูลเชิงแนวคิด (Conceptual data model) หรือแบบจำลองข้อมูลเชิงปฏิบัติการ (Implementation data model) หรือ Representation data model)

สคีมาภายนอก – ใช้บรรยายมุมมองหรือวิว (view) ที่หลากหลายของผู้ใช้ โดยที่ร่วมแล้วมักใช้แบบจำลองข้อมูลเชิงปฏิบัติการ

## 2.2.2 สถาปัตยกรรมแบบรวมศูนย์

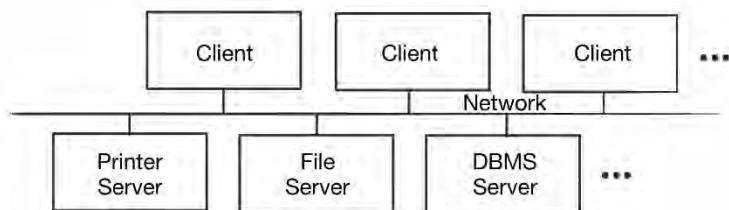
สถาปัตยกรรมสำหรับระบบการจัดการฐานข้อมูลนั้น ได้รับอิทธิพลจากสถาปัตยกรรมระบบคอมพิวเตอร์ทั่วไป ซึ่งในช่วงแรกนั้นมีการใช้คอมพิวเตอร์เมนเฟรมในกระบวนการฟังก์ชันทั้งหมดของระบบ โดยรวมทุกอย่างไว้ในระบบเดียว ไม่ว่าจะเป็นระบบการจัดการฐานข้อมูลแอปพลิเคชัน หรือ คอมไฟเลอร์ เรียกว่า สถาปัตยกรรมแบบรวมศูนย์ (centralized architecture)

## 2.2.3 สถาปัตยกรรมแบบลูกข่ายแม่ข่าย

สถาปัตยกรรมแบบลูกข่ายแม่ข่าย (client/server architecture) ใช้แนวคิดในการนำบริการกลางวางไว้ที่แม่ข่าย เช่น แม่ข่ายจัดการไฟล์ (file server) แม่ข่ายจัดการเครื่องพิมพ์ (printer server) แม่ข่ายเว็บ (web server) และแม่ข่ายอีเมล (email server) เป็นต้น ส่วนเครื่องลูกข่าย (client) จะประกอบด้วยส่วนติดต่อกับผู้ใช้ที่เหมาะสมในการเข้าถึงและใช้ทรัพยากรของเครื่องแม่ข่าย ซึ่งเครื่องลูกข่ายอาจเป็นเครื่องที่ไม่มีดิสก์ หรือเป็นคอมพิวเตอร์ส่วนบุคคลหรือเป็นเวิร์กสเตชัน (workstation) ที่ติดตั้งเฉพาะซอฟต์แวร์ลูกข่าย โดยเครื่องลูกข่ายจะถูกเชื่อมต่อสู่เครื่องแม่ข่ายผ่านทางระบบเครือข่าย (network) เช่น ระบบเครือข่ายท้องถิ่น (Local Area Network : LAN) หรือเครือข่ายไร้สาย (wireless network) เป็นต้น ซึ่งสถาปัตยกรรมแบบลูกข่ายแม่ข่ายนี้แบ่งได้ 2 ประเภท ได้แก่ 1) สถาปัตยกรรมแบบลูกข่ายแม่ข่ายสองชั้น 2) สถาปัตยกรรมแบบลูกข่ายแม่ข่ายสามชั้น

### 2.2.3.1 สถาปัตยกรรมแบบลูกข่ายแม่ลูกสองชั้น

รูปที่ 11 แสดงให้เห็นการแบ่งชั้นเป็น 2 ชั้น (two-tier client/server architecture) คือชั้nlูกข่าย และชั้นแม่ข่าย ที่ทำงานร่วมกันผ่านระบบเครือข่าย โดยที่โปรแกรมติดต่อผู้ใช้ (user interface program) และแอปพลิเคชันต่าง ๆ จะทำงานอยู่บนฝั่งของเครื่องลูกข่าย และจะเข้าถึงฐานข้อมูลโดยใช้อีดีบีซี (Open Database Connectivity : ODBC) ซึ่งมีส่วนติดต่อเอฟไอโอ (Application Program Interface : API) ที่อนุญาตให้โปรแกรมฝั่งลูกข่ายสามารถเรียกฐานข้อมูลได้ ส่วนโปรแกรมลูกข่าย (client program) แต่ละโปรแกรมอาจเขียนต่อ กับระบบฐานข้อมูลได้หลายระบบ นอกจากนี้ ลูกข่ายสามารถใช้พังก์ชันต่าง ๆ จากแม่ข่ายได้ เช่น พังก์ชันพจนานุกรมข้อมูล (data dictionary function) หรือการคืนสภาพ (recovery)

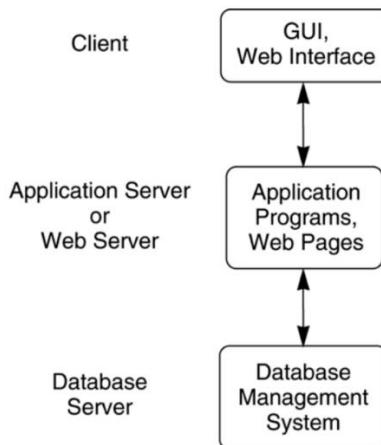


รูปที่ 11 สถาปัตยกรรมแบบลูกข่ายแม่ลูกสองชั้น

### 2.2.3.2 สถาปัตยกรรมแบบลูกข่ายแม่ข่ายสามชั้น

รูปที่ 12 แสดงสถาปัตยกรรมแบบลูกข่ายแม่ข่ายสามชั้น (three-tier client/server architecture) มักใช้ในเว็บแอปพลิเคชัน (web application) โดยมีการเพิ่มชั้นกลาง (intermediate layer) ระหว่างลูกข่ายและแม่ข่ายฐานข้อมูล (database server) ซึ่งเรียกชั้นกลางนี้ว่าแม่ข่ายแอปพลิเคชัน (application server) หรือแม่ข่ายเว็บ (web server) โดยจะเป็นส่วนที่ทำหน้าที่เก็บซอฟต์แวร์การเชื่อมต่อเว็บ (web connectivity software) รวมถึงกฎและข้อบังคับต่าง ๆ ของแอปพลิเคชันในการเข้าถึงฐานข้อมูลในแม่ข่ายฐานข้อมูล

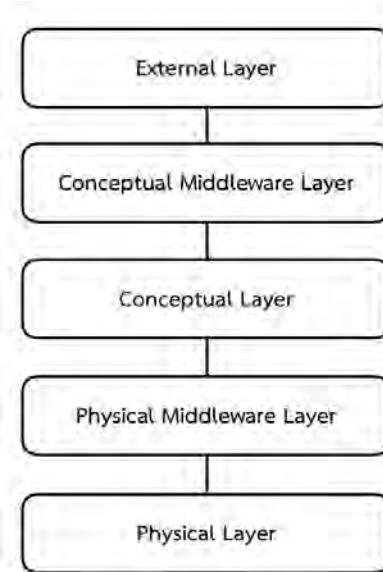
นอกจากนี้ยังมีลักษณะพิเศษเพิ่มเติมเกี่ยวกับความปลอดภัย โดยจะนำข้อมูลมาเข้ารหัส (encrypt) ที่เครื่องแม่ข่ายก่อนจะทำการส่งต่อออกไป向外รหัส (decrypt) ที่เครื่องลูกข่ายด้วย



รูปที่ 12 สถาปัตยกรรมแบบลูกข่ายแม่ข่ายสามชั้น

#### 2.2.4 สถาปัตยกรรมแบบคลาวด์

สถาปัตยกรรมแบบคลาวด์ (cloud architecture) หรือระบบการจัดการฐานข้อมูลแบบคลาวด์ เป็นแนวปฏิบัติที่ได้รับความนิยมอย่างรวดเร็ว ระบบการจัดการฐานข้อมูลแบบคลาวด์นี้ทำงานบนแพลตฟอร์มการประมวลผลคลาวด์ (cloud computing platform) เช่น บริการเว็บของอเมซอน (Amazon Web Service : AWS) ที่ให้บริการทั้งการประมวลผล พื้นที่จัดเก็บข้อมูล การรักษาความปลอดภัย การสำรองข้อมูลและอื่น ๆ อีกมากมายที่ทำให้ลดความยุ่งยากในการติดตั้งและดูแลระบบ มีทั้งแบบบริการฟรีและแบบเก็บเงิน รูปที่ 13 แสดงสถาปัตยกรรมระบบการจัดการฐานข้อมูลแบบคลาวด์ [18]



รูปที่ 13 สถาปัตยกรรมแบบคลาวด์

## 2.3 ภาษาฐานข้อมูล

การจัดการข้อมูลภายในฐานข้อมูลนั้น เราจำเป็นต้องใช้ภาษาคอมพิวเตอร์ที่ออกแบบเฉพาะเจาะจงให้เหมาะสมกับการจัดการข้อมูล ภาษาฐานข้อมูลนี้แบ่งได้สองกลุ่ม [19, 20] คือ 1) ภาษาดีดีแอล และ 2) ภาษาดีเอ็มแอล

### 2.3.1 ภาษาดีดีแอล

ภาษาดีดีแอล (data definition language : DDL) เป็นภาษาที่ผู้ดูแลฐานข้อมูลและผู้ออกแบบฐานข้อมูลใช้ในการระบุสคีมาเชิงแนวคิดของฐานข้อมูล ซึ่งระบบการจัดการฐานข้อมูลส่วนใหญ่ก็มักใช้ภาษาดีดีแอลในการกำหนดสคีมาภายในและสคีมาภายนอกด้วยเช่นกัน นอกจากนี้ระบบการจัดการฐานข้อมูลบางตัว ยังมีการเพิ่มภาษาเอสตีแอล (storage definition language : SDL) และภาษาวีดีแอล (view definition language : VDL) ในการกำหนดสคีมาภายในและภายนอกโดยเฉพาะอีกด้วย แต่ก็ยังสามารถจัดอยู่ในดีดีแอลได้

### 2.3.2 ภาษาดีเอ็มแอล

ภาษาดีเอ็มแอล (data manipulation language : DML) ใช้ในการระบุการสืบค้นฐานข้อมูลต่าง ๆ เช่น การลบ การแทรก เป็นต้น โดยสามารถนำคำสั่งดีเอ็มแอลไปฝังติดไว้กับโปรแกรมที่เขียนด้วยภาษาใด ๆ เช่น ภาษา Java หรือภาษา C++ นอกจากนี้ยังมีภาษาดีเอ็มแอลที่สามารถนำมาใช้สืบค้นข้อมูลโดยตรงคือ ภาษาสอบถาม ที่ใช้กันแพร่หลายในปัจจุบัน ภาษาดีเอ็มแอลสามารถแบ่งย่อยได้สองประเภทคือ 1) ดีเอ็มแอลระดับสูง เช่น ภาษาอาลกอริทึม

แหล่งที่ทำงานแบบไม่ต้องระบุขั้นตอน (non-procedural) และ 2) ดีเอ็มแอล ระดับต่ำ ที่ทำงานทีละคำสั่ง (record-at-a-time)

## 2.4 ประเภทของระบบการจัดการฐานข้อมูล

ประเภทของระบบการจัดการฐานข้อมูลสามารถพิจารณาได้จากหลาย กฎเกณฑ์

หากพิจารณาจากจำนวนผู้ใช้ จะแบ่งได้สองประเภทคือ 1) ผู้ใช้เดียว ซึ่งมักใช้ กับคอมพิวเตอร์ส่วนบุคคล และ 2) ผู้ใช้หลายคน ซึ่งระบบการจัดการ ฐานข้อมูลส่วนใหญ่จดอยู่ในประเภทนี้

และหากพิจารณาจากลักษณะของไซต์ (site) หรือสถานที่ตั้ง [2] แบ่งได้สอง ประเภทคือ

- 1) แบบรวมศูนย์ ใช้คอมพิวเตอร์หนึ่งเครื่องกับหนึ่งฐานข้อมูล
- 2) แบบกระจาย (distributed) ใช้คอมพิวเตอร์หลายเครื่องและหลาย ฐานข้อมูล สำหรับระบบฐานข้อมูลแบบกระจาย (distributed database system) นั้นก็คือระบบฐานข้อมูลที่ใช้สถาปัตยกรรม แบบลูกข่ายแม่ข่ายนั่นเอง ทั้งนี้ยังสามารถแบ่งย่อยได้เป็นแบบเอก พันธ์ (homogeneous) แบบวิริพันธ์ (heterogeneous) และ แบบสหพันธ์ (federated หรือ multidatabase) ได้อีกด้วย

เมื่อพิจารณาจากโครงสร้างของฐานข้อมูล [21] จะแบ่งประเภทได้ 4 ประเภท ได้แก่

- 1) ฐานข้อมูลเชิงลำดับชั้น (hierarchical database) ดังแสดงตัวอย่างในรูปที่ 14
- 2) ฐานข้อมูลเชิงเครือข่าย (network database) และแสดงตัวอย่างในรูปที่ 15
- 3) ฐานข้อมูลเชิงสัมพันธ์ (relational database) และแสดงตัวอย่างในรูปที่ 16
- 4) ฐานข้อมูลเชิงวัตถุ (object-oriented database)

P1	Nut	Red	12	London
S2	Jones	10	Paris	400
S1	Smith	20	London	200

P2	Bolt	Green	17	Paris
S3	Blake	30	Paris	200
S2	Jones	10	Paris	400

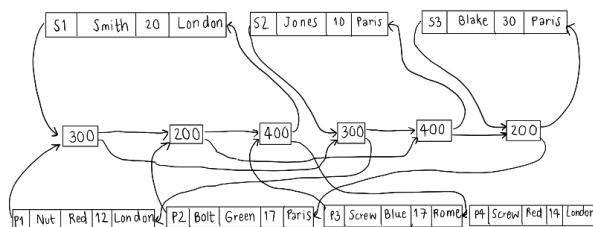
  

S1	Smith	20	London	200
----	-------	----	--------	-----

P3	Screw	Blue	17	Rome
S1	Smith	20	London	200

P4	Screw	Red	14	London
----	-------	-----	----	--------

รูปที่ 14 ฐานข้อมูลเชิงลำดับชั้น



รูปที่ 15 ฐานข้อมูลเชิงเครือข่าย

S	S=	SNAME	STATUS	CITY	SP	S=	P=	QTY
S1		Smith	20	London	S1	P1	300	
S2		Jones	10	Paris	S1	P2	200	
S3		Blake	30	Paris	S1	P3	400	
					S2	P1	300	
					S2	P2	400	
					S3	P2	200	

P	P=	PNAME	COLOR	WEIGHT	CITY
P1		Nut	Red	12	London
P2		Bolt	Green	17	Paris
P3		Screw	Blue	17	Rome
P4		Screw	Red	14	London

### รูปที่ 16 ฐานข้อมูลเชิงสัมพันธ์

อย่างไรก็ดี ได้มีระบบการจัดการฐานข้อมูลในยุคหลังเกิดขึ้นมาอีกหลายประเภท และก็หายหรือล้าสมัยไปอีกมาก ดังนั้น เราอาจแบ่งประเภทแบบใหม่ให้สอดคล้องกับยุคปัจจุบันได้ดังนี้ [22]

- ระบบการจัดการฐานข้อมูลเชิงสัมพันธ์ (relational DBMS) ที่พึ่งพาภาษา(es)คิวแอลเป็นหลักในการจัดการข้อมูล
- ระบบการจัดการฐานข้อมูลโน(es)คิวแอล (NoSQL) ย่อมาจาก Not-Only-SQL หมายถึงไม่จำเป็นต้องพึ่งพาภาษา(es)คิวแอลเท่านั้น โดยที่โน(es)คิวแอลเป็นแนวทางในการจัดการข้อมูลที่เหมาะสมกับข้อมูลขนาดใหญ่ ที่อยู่อย่าง กระจายตัว หลากหลายรูปแบบ ไม่มีโครงสร้างแน่นชัด เช่น ข้อมูลบนอินเทอร์เน็ต และยังประมวลผลได้รวดเร็วระดับเวลาจริง โน(es)คิวแอลแบ่งเป็นประเภทอยู่ได้คือ

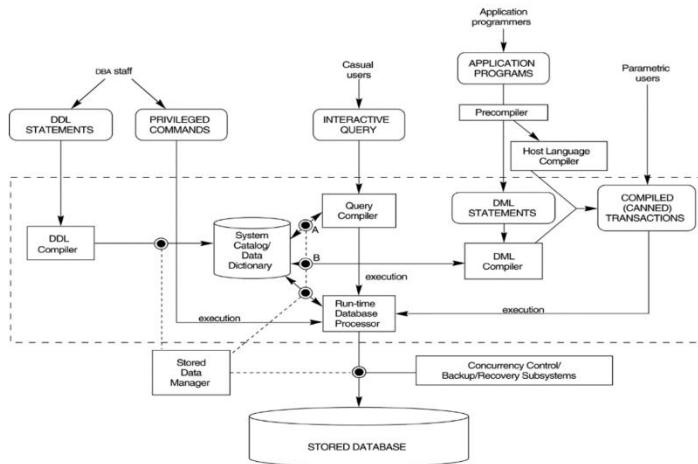
- 2.1. ฐานข้อมูลแบบเอกสาร (Document stores) เป็นฐานข้อมูลในเอกสารและ ที่ออกแบบมาให้เหมาะสมกับการบันทึกข้อมูลรูปแบบใดๆ เป็นชุดข้อความขนาดใหญ่ เช่น JSON structure ในลักษณะเอกสาร ตัวอย่างของ NoSQL database ผลิตภัณฑ์ที่เป็นที่รู้จักได้แก่ MongoDB และ CouchDB
- 2.2. ฐานข้อมูลค่าคีย์ (Key-value stores) เป็นฐานข้อมูลที่สามารถจัดเก็บข้อมูลรูปแบบใดๆ ด้วยการระบุค่าคีย์ของข้อมูลนั้นๆ ตัวอย่างผลิตภัณฑ์ ได้แก่ Redis, Amazon S3 (Dynamo) และ Riak
- 2.3. ฐานข้อมูลคอลัมน์กว้าง (Wide-column stores) เป็นฐานข้อมูลที่ออกแบบให้จัดเก็บข้อมูลแบบคอลัมน์แทนการเก็บแบบบรรทัด ผลิตภัณฑ์ที่เป็นที่นิยมคือ Cassandra
- 2.4. ฐานข้อมูลแบบกราฟ (Graph database) เป็นฐานข้อมูลที่ออกแบบมาให้ผู้ใช้สามารถระบุความสัมพันธ์ของข้อมูลแต่ละชุด ในรูปแบบเครือข่ายหรือกราฟ ตัวอย่างของผลิตภัณฑ์ ได้แก่ Neo4j

ทั้งนี้เรายังพบว่าฐานข้อมูลประเภทโน้ตอสคิวแอลจะเกิดแบบใหม่ๆ ได้ค่อนข้างรวดเร็ว แต่อาจยังไม่เสถียรหรือแพร์ทลายมากนัก จึงยังไม่นำมากล่าวในที่นี้

## 2.5 ภาพรวมของระบบการจัดการฐานข้อมูล

การทำงานภายในระบบการจัดการฐานข้อมูลมีรายละเอียดดังแสดงในรูปที่ 17 มาจาก [2] ให้สังเกตว่าส่วนบนของรูปคือผู้ใช้ระบบฐานข้อมูลทั้งหลาย ไม่ว่าจะเป็นโปรแกรมเมอร์ ผู้ดูแลฐานข้อมูล หรือผู้ใช้ทั่วไป เมื่อผู้ใช้ระดับต่าง ๆ ส่งคำสั่งเพื่อเรียกใช้ข้อมูลแล้ว ระบบการจัดการฐานข้อมูลจะส่งคำสั่งจากหน่วยอยู่หนึ่งไปยังหน่วยอยู่อีกตัวไปตามลำดับในรูป

โดยยกตัวอย่างผู้ใช้ที่เป็นผู้ดูแลฐานข้อมูล (Database Administrator staff : DBA staff) ที่มักเขียนคำสั่งด้วยภาษาดีดีแอลเพื่อจัดการข้อมูล คำสั่นนี้จะถูกส่งไปยังดีดีแอลคอมโพเดอร์ให้แปลงคำสั่นนั้นให้ระบบการจัดการฐานข้อมูลเข้าใจ แล้วส่งต่อไปยังแค็ตตาล็อกหรือพจนานุกรมข้อมูล เมื่อระบบการจัดการฐานข้อมูลเข้าใจแล้วว่าจะต้องไปดึงข้อมูลอะไรบ้าง ก็จะส่งต่อไปยังหน่วยประมวลผลฐานข้อมูลรันไทม์ (run-time database processor) ที่จะทำงานร่วมกับหน่วยจัดการการเก็บข้อมูล (stored data manager) และ หน่วยจัดการภาวะพร้อมกัน (concurrency control) ก่อนจะดึงข้อมูลจากหน่วยความจำแล้วแสดงคำตอบ



รูปที่ 17 การทำงานภายในระบบการจัดการฐานข้อมูล

ฟังก์ชันของระบบการจัดการฐานข้อมูลยังมีอีกมากmany เช่น การแปลงข้อมูล (data conversion tool) การสำรองข้อมูล การปรับโครงสร้างฐานข้อมูล (reorganizing) การสร้างรายงาน (report generation) การติดตามประสิทธิภาพการทำงาน (performance monitoring) การเรียงข้อมูล (sorting) การดูแลตรวจสอบผู้ใช้ (user monitoring) หรือการบีบอัดข้อมูล (data compression) เป็นต้น

## 2.6 บทสรุปและเรื่องชวนคิด

การพัฒนาระบบฐานข้อมูลมีการเปลี่ยนแปลงไปมาก ผ่านมาหลายยุคหลายสมัย มีความซับซ้อนท้าทาย ทำให้สถาปัตยกรรมระบบคอมพิวเตอร์เปลี่ยน ตามกันแทบไม่ทัน จนในปัจจุบันองค์กรขนาดใหญ่หลายแห่งก็ใช้บริการ

คลาวด์เพื่อเก็บและจัดการระบบการจัดการฐานข้อมูลของตนเอง เพราจะอาจ  
ประยุกต์ทรัพยากรได้หลายด้าน ผู้เรียนคิดว่าปัจจัยอะไรบ้างที่ส่งผลในการ  
ตัดสินใจให้องค์กรเลือกใช้บริการระบบการจัดการฐานข้อมูลแบบคลาวด์  
และแนวโน้มการเปลี่ยนแปลงนี้จะส่งผลกระทบต่องานในวงการคอมพิวเตอร์อย่างไร  
บ้าง

## แบบฝึกหัดท้ายบท

1. จงอธิบายข้อแตกต่างระหว่างภาษาดีดีแอลและภาษาดีเอ็มแอล
2. ขั้นตอนการวิเคราะห์ความต้องการจากผู้ใช้ ประกอบด้วยสิ่งที่ต้องพิจารณาหลักกี่ประเด็น อะไรบ้าง เพราะเหตุใดจึงต้องเน้นที่ประเด็นหลักเหล่านี้
3. กระบวนการออกแบบและพัฒนาฐานข้อมูล ควรทำไปพร้อมซึ่งได้ในกระบวนการทางวิศวกรรมซอฟต์แวร์
4. การออกแบบและพัฒนาฐานข้อมูลควบคู่ไปกับการออกแบบและพัฒนาแอปพลิเคชันการพัฒนาระบบฐานข้อมูล มีกี่ขั้นตอน อะไรบ้าง
5. ระบบปฏิบัติการที่ออกแบบด้วยการออกแบบชนิดโมดูล่า มีลักษณะการประมวลผลเป็นอย่างไร
6. เหตุใดองค์กรบางแห่งจะไม่เลือกเป็นเจ้าของแม่ข่าย แต่ใช้บริการบนคลาวด์แทน
7. จงอธิบายความแตกต่างระหว่างสถาปัตยกรรมฐานข้อมูลแบบลูกข่าย แม่ข่ายสองชั้น กับสถาปัตยกรรมฐานข้อมูลแบบลูกข่ายแม่ข่ายสามชั้น
8. จงยกตัวอย่างเกณฑ์ในการแบ่งประเภทของระบบการจัดการฐานข้อมูล มา 3 ตัวอย่าง พิรőมบอกว่าถ้าแบ่งด้วยเกณฑ์นั้น จะแบ่งระบบการจัดการฐานข้อมูลได้กี่ประเภท อะไรบ้าง
9. ฐานข้อมูลแบบโนอสคิวแอลเหมาะกับข้อมูลแบบใด
10. จงยกตัวอย่างฟังก์ชันของระบบการจัดการฐานข้อมูลมาพอสักเชป



## บทที่ 3

### การจำลองข้อมูล

#### วัตถุประสงค์

- เพื่อให้เข้าใจวัตถุประสงค์และความสำคัญของการจำลองข้อมูล
- เพื่อให้รู้จักแบบจำลองข้อมูลอีอาร์และอีอาร์
- เพื่อให้สามารถระบุเงนทิตี รีเลชันชิป และแอตทริบิวต์
- เพื่อให้สามารถออกแบบฐานข้อมูลด้วยแผนภาพอีอาร์และอีอาร์

การทำางานของระบบฐานข้อมูลในอาร์ดแวร์ต่างระบบต่างยุคหนึ่ง จำเป็นต้องใช้แบบจำลองข้อมูล (data model) ที่เหมาะสมกับอุปกรณ์และเครื่องมือที่ต่างกันไป แบบจำลองข้อมูลคือวิธีอธิบายโครงสร้างของฐานข้อมูล ซึ่งมีหลายแบบ สำหรับในตำนานี้ เราจะพูดถึงฐานข้อมูลเชิงสัมพันธ์ ดังนั้น แบบจำลองข้อมูลที่เราจะเน้น คือแบบจำลองข้อมูลเชิงสัมพันธ์ ซึ่งวิธีอธิบายโครงสร้างของฐานข้อมูลเชิงสัมพันธ์จะพูดถึง ชนิดของข้อมูล ความสัมพันธ์ของข้อมูล และเงื่อนไขบังคับต่าง ๆ ผลลัพธ์จากการทำแบบจำลองข้อมูลนี้ จะนำไปใช้ต่อในขั้นตอนลัดไป คือขั้นตอนการดำเนินการบนแบบจำลองฐานข้อมูล (data model operations) การดำเนินการ อาจแบ่งได้เป็น 2 ระดับคือ 1) การดำเนินการระดับพื้นฐาน (basic operation) ที่ระบบการจัดการฐานข้อมูลสามารถรับได้ เช่น การเพิ่ม การลบ การแก้ไข หรือการสืบค้นข้อมูล เป็นต้น และ 2) การดำเนินการตามที่ผู้ใช้ระบุ (user-defined operation) หมายถึง การดำเนินการที่โปรแกรมเมอร์เขียนตามความต้องการในการใช้ข้อมูล เช่น COMPUTE\_GPA for STUDENT เป็นต้น

### 3.1 การจำลองข้อมูล

ในการออกแบบฐานข้อมูลนั้น เราจำเป็นที่จะต้องมีความเข้าใจและสามารถออกแบบฐานข้อมูลโดยใช้แบบจำลองข้อมูลเป็นเครื่องมือแสดงความคิดนั้น ๆ ใช้เป็นการสื่อสารระหว่างโปรแกรมเมอร์ด้วยกันหรือสื่อสารกับผู้ใช้ระบบ ซึ่งจะทำให้เราได้เห็นถึงพื้นฐานในการออกแบบโครงสร้างข้อมูลและเข้าใจถึงแบบจำลองของข้อมูลในแนวคิดระดับสูง

## 3.2 แบบจำลองข้อมูล

แบบจำลองข้อมูลมีหลากหลายแบบ มีความเหมาะสมในการใช้งานต่างกันไป ในที่นี้ ขอยกตัวอย่างแบบจำลองที่มักใช้กันอย่างแพร่หลาย 3 แบบ คือ

### 3.2.1 แบบจำลองข้อมูลเชิงแนวคิด

แบบจำลองข้อมูลเชิงแนวคิด (conceptual data model) เป็นแบบจำลองระดับสูง (high-level) ที่แสดงความหมาย (semantic) ของระบบฐานข้อมูลนั้น ๆ โดยแสดงแนวคิดของข้อมูลให้ใกล้เคียงกับความเข้าใจของผู้ใช้ทั่วไป ใช้ในการสื่อสารระหว่างโปรแกรมเมอร์กับผู้ใช้งานระบบ

เนื่องจากตำแหน่งนี้จะครอบคลุมการออกแบบระบบการจัดการฐานข้อมูล จึงเน้นการออกแบบเชิงแนวคิด

### 3.2.2 แบบจำลองข้อมูลเชิงกายภาพ

แบบจำลองข้อมูลเชิงกายภาพ (physical data model) เป็นแบบจำลองระดับต่ำ (low-level) หรือระดับภายใน (internal) เพื่อระบุรายละเอียดใน การจัดเก็บข้อมูลลงในกายภาพของฐานข้อมูลนั้น ๆ เช่น ตำแหน่งของข้อมูล ในหน่วยความจำ ขนาดและรูปแบบของข้อมูล เป็นต้น

### 3.2.3 แบบจำลองข้อมูลเชิงปฏิบัติการ

แบบจำลองข้อมูลเชิงปฏิบัติการ (implementation data model หรือ representational data model) แบบจำลองนี้ เป็นแบบจำลองที่อยู่

กิ่งกลางระหว่าง 2 แบบแรก มีการแสดงรวมรายละเอียดทั้ง 2 แบบแรกไว้ด้วยกัน อย่างไรก็ตาม แบบจำลองนี้ไม่ได้รับความนิยมเท่า 2 แบบแรก

### 3.3 แบบจำลองอีอาร์

ยุคของฐานข้อมูลเชิงสัมพันธ์ที่เริ่มมาตั้งแต่ศวรรษ 1970 มีพื้นฐานอยู่บนแบบจำลองเชิงสัมพันธ์ แบบจำลองนี้มีชื่อเรียกว่า แบบจำลองเอนทิตี รีเลชันชิป (Entity-Relationship model) [1, 10, 12-15, 23-25] หรือนิยมเรียกย่อได้ว่า แบบจำลองอีอาร์ (ER model) แบบจำลองอีอาร์นี้ออกแบบมาเพื่อให้ผู้ใช้อธิบายภาพรวมหรือแนวคิดของฐานข้อมูลนั้น โดยเน้นความสัมพันธ์ระหว่างข้อมูลในฐานข้อมูลนั้น ๆ องค์ประกอบสำคัญของรูปแบบอีอาร์ 3 ประการ คือ เอนทิตี รีเลชันชิป และแอตทริบิวต์ ตำราเล่มนี้เลือกใช้ทับศัพท์ภาษาอังกฤษ เพราะเขื่อว่าจะทำให้ลดความสับสนได้

1. เอนทิตี (entity) ศัพท์บัญญัติราชบัณฑิตยสถาน ให้ใช้ว่า เอนทิตี
2. แอตทริบิวต์ (attribute) ศัพท์บัญญัติราชบัณฑิตยสถานคือ ลักษณะประจำ แต่เนื่องจากเป็นศัพท์ที่ใช้แสดงสัญลักษณ์เฉพาะทาง ตำรานี้จึงขอใช้ทับศัพท์ว่า แอตทริบิวต์
3. รีเลชันชิป (relationship) ทั่วไปหมายถึงความสัมพันธ์ แต่เนื่องจากเป็นศัพท์ที่ใช้แสดงสัญลักษณ์เฉพาะทาง จึงขอทับศัพท์ว่า รีเลชันชิป

เราสามารถเขียนแบบจำลองความสัมพันธ์ระหว่างเอนทิตีกับรีเลชันชิป รวมทั้งแอตทริบิวต์ด้วยแผนภาพ และแผนภาพที่ถูกออกแบบมาสำหรับแบบจำลองนี้คือ แผนภาพความสัมพันธ์ระหว่างเอนทิตีกับรีเลชันชิป (Entity-Relationship diagram หรือ ER diagram) ต่อไปนี้จะเรียกว่า แผนภาพ

อีอาร์ที่แสดงให้เห็นถึงเอนทิตี้และรีเลชันชิปของข้อมูลได้อย่างง่ายดาย ไม่ซับซ้อน

เพื่อให้เกิดความเข้าใจที่ชัดเจน ขอใช้แผนภาพอีอาร์เพื่อแสดงแบบจำลองฐานข้อมูลจากตัวอย่างระบบฐานข้อมูลของบริษัทแห่งหนึ่ง ให้ชื่อว่าบริษัทดีบี และใช้รูปแบบการเขียนที่นิยมในวงการฐานข้อมูล คือการใช้อักษรภาษาอังกฤษตัวใหญ่แสดงชื่อของข้อมูลโดยเฉพาะเอนทิตี้ ชื่่งโจทย์และผลลัพธ์จากนี้นำมายัง [2]

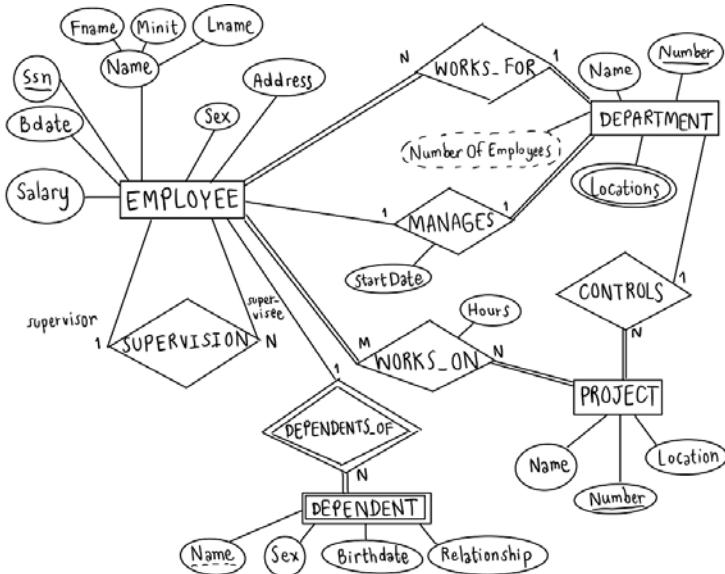
โจทย์กำหนดให้ดังนี้

- บริษัทดีบีมีการจัดระบบโดยเริ่มจากการแบ่งเป็นแต่ละแผนก ซึ่งเราจะเรียกว่า DEPARTMENT ถือว่าเป็นหน่วยย่อยของบริษัท โดยให้ DEPARTMENT มีชื่อและรหัสที่แตกต่างกัน
- แต่ละ DEPARTMENT มีพนักงานหลายคน ซึ่งเราจะเรียกว่า EMPLOYEE ทั้งนี้ EMPLOYEE แต่ละคนสามารถทำงานได้ใน DEPARTMENT เดียวเท่านั้น
- แต่ละ DEPARTMENT มีผู้จัดการ ซึ่งเราจะเรียกว่า MANAGER แต่ละ DEPARTMENT มีผู้จัดการหนึ่งคนเท่านั้น และเราต้องการเก็บข้อมูลวันที่เริ่มต้นเป็นผู้จัดการ DEPARTMENT นั้นไว้
- แต่ละ DEPARTMENT อาจมีสถานที่ตั้งได้หลายที่ ซึ่งเราจะเรียกสถานที่ว่า LOCATION
- นอกจากราชานั้น EMPLOYEE แต่ละคนจะทำงานในโครงการ ซึ่งเราจะเรียกว่า PROJECT แต่ละคนทำได้คนละหลาย PROJECT ซึ่งแต่ละ

PROJECT จะมีชื่อ เลขที่ และ LOCATION โดยแต่ละ PROJECT นั้นตั้งอยู่ใน LOCATION เดียวเท่านั้น

- บริษัทต้องการเก็บว่า EMPLOYEE แต่ละคนทำงานให้แต่ละ PROJECT กี่ชั่วโมงด้วย
- แต่ละ DEPARTMENT มีหน้าที่ควบคุม PROJECT ซึ่งมีได้หลาย PROJECT ต่อ DEPARTMENT
- ในส่วนของ EMPLOYEE บริษัทต้องการจัดเก็บข้อมูลของ EMPLOYEE คือ ชื่อพนักงาน รหัสประจำตัว ที่อยู่ เงินเดือน เพศ และวันเกิด
- บริษัทมีการกำหนดผู้คุมงาน ซึ่งเราจะเรียกว่า SUPERVISOR และ ต้องการบันทึกว่า EMPLOYEE คนใดเป็น SUPERVISOR ของคนใด อีกด้วย
- นอกจากนี้ยังมีการเก็บข้อมูลผู้อยู่ในอุปการะของ EMPLOYEE ซึ่ง เราจะเรียกว่า DEPENDENT เช่น สามี ภรรยา บุตร หรือ คนในครอบครัว ซึ่ง EMPLOYEE แต่ละคน อาจมี DEPENDENT ได้หลาย คน ข้อมูล DEPENDENT ที่สนใจคือ ชื่อ เพศ วันเกิด และ ความสัมพันธ์กับ EMPLOYEE

จากข้อมูลที่โจทย์กำหนด เราสามารถเขียนแผนภาพอีกรูปได้ดังแสดงในรูปที่ 18



รูปที่ 18 แผนภาพอีอาร์แสดงระบบฐานข้อมูลของบริษัทดีบี

### 3.3.1 เอนทิตี้

เอนทิตี้ (entity) แสดงในภาพด้วยกล่องสีเหลืองผึ้ง พื้นที่ภายในคือ สิ่งใด ๆ ที่เราให้ความสำคัญ และเป็นส่วนที่เราต้องการนำเสนอในฐานข้อมูล โดยอาจจะเป็นได้ทั้ง คน สัตว์ สิ่งของ หรือสรรพสิ่งใด ๆ ถ้ายกตัวอย่างตามแผนภาพอีอาร์ในรูปที่ 18 จะเห็นตัวอย่างของเอนทิตี้ เช่น EMPLOYEE คือพนักงานแต่ละคน เช่น

สมชาย รักการเรียน, สมหญิง รักการทำงาน, สมกมล รักการบ้าน

เมื่อกำเนิดคลาสเอนทิตี้ที่มีลักษณะร่วมกันมากกัน เราจะเรียกว่า เอ็นทิตี้ไทย (entity type) แปลเป็นไทยว่าแบบเอนทิตี้ แต่เราขอทับศัพท์ว่า เ昂ทิตี้ไทย เพื่อให้มีต้องสับระหว่างไทยอังกฤษในศัพท์เดียวกันที่น่าจะทำให้เข้าใจยาก

ดังนั้น DEPARTMENT ที่ 1 หรือ DEPARTMENT ที่ 2 เรียกว่าเป็น เ昂ทิตี้ และเมื่อทุก DEPARTMENT มาร่วมกัน ให้เรียกว่าเป็น เ昂ทิตี้ไทย DEPARTMENT

ในทำงดีယกัน PROJECT แต่ละ PROJECT ก็เป็นเอนทิตี้ เมื่ออา เ昂ทิตี้ PROJECT มาร่วมกันทั้งหมด ก็เรียกว่า เ昂ทิตี้ไทย PROJECT

สรุปจากตัวอย่างนี้ก็จะมี เ昂ทิตี้ไทย EMPLOYEE, DEPARTMENT, PROJECT และ DEPENDENT

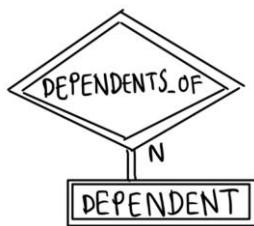
สังเกตจากแผนภาพ จะเห็นว่า เราแทนเอนทิตี้ไทยด้วยกล่องสี่เหลี่ยมผืนผ้า และเขียนชื่อเอนทิตี้ไทยไว้ในกล่องด้วยตัวพิมพ์ใหญ่

ตามที่ได้กล่าวไปแล้วข้างต้นว่า เ昂ทิตี้ที่มีแอ็ตทริบิวต์พื้นฐานเหมือนกัน จะถูกจัดกลุ่มหรือจัดประเภทเป็นเอนทิตี้ไทยเดียวกัน เช่น เ昂ทิตี้ไทย EMPLOYEE เ昂ทิตี้ไทย PROJECT ส่วนเอนทิตี้แต่ละเอนทิตี้เมื่อมาร่วมกัน จะเรียกว่า กลุ่มเอนทิตี้ (entity set) หรือ เอกซ์เทนชัน (extension)

จากแผนภาพอีกครั้งที่แสดงไว้ก่อนหน้านี้ จะพบเอนทิตี้ไทย 2 ประเภท ได้แก่

1. เ昂ทิตี้ไทยปกติ (regular entity type) ให้เรียกว่า เ昂ทิตี้ไทย
2. วีคเอนทิตี้ไทย (weak entity type)

เอนทิตี้ไทยปักติที่ปรากฏในแผนภาพอีอาร์มี 3 ตัว คือ EMPLOYEE, DEPARTMENT และ PROJECT ส่วน DEPENDENT นั้นเป็นวีคเอนทิตี้ไทย แสดงด้วยกล่องสี่เหลี่ยมผืนผ้าเส้นคู่ หมายความว่าเป็นเอนทิตี้ที่ไม่มีแอ็ตทริบิวต์ใดเป็นคีย์ (Key attribute) ของตนเองที่จะทำให้ข้อมูลในฐานข้อมูลนั้นไม่ซ้ำ แต่อาจมีคีย์บางส่วน (partial key) ได้ ให้แสดงคีย์บางส่วนด้วยเส้นใต้แบบประดับงั้น ดังนั้นวีคเอนทิตี้ไทยจำเป็นต้องใช้คีย์จากเอนทิตี้ไทยที่มีแอ็ตทริบิวต์เชื่อมถึงกัน เรียกเอนทิตี้ไทยดังกล่าวว่า “เอนทิตี้ไทยเจ้าของ (owner entity type) หรือเอนทิตี้ไทยระบุ (identifying entity type) และเรียกความสัมพันธ์ที่เข้มต่อระหว่างวีคเอนทิตี้ไทยกับชนิดระบุว่า รีเลชันชิป “ไทยระบุ” (identifying relationship type) โดยวีคเอนทิตี้ไทยนี้จะใช้คีย์จากเอนทิตี้ไทยระบุมารวมกับคีย์บางส่วน (partial key) ของตัวเอง เพื่อใช้เป็นคีย์ที่ทำให้ข้อมูลทุกบรรทัดในฐานข้อมูลนั้นไม่ซ้ำกัน จากรูปที่ 19 จะเห็นว่า DEPENDENT ต้องนำคีย์มาจาก EMPLOYEE คือ ID มาเป็นคีย์ร่วม เราเรียกว่า EMPLOYEE ว่าเอนทิตี้ชนิดระบุ เรียก DEPENDENTS\_OF ว่า รีเลชันชิป “ไทยระบุ”



รูปที่ 19 วีคเอนทิตี้ไทย

### 3.3.2 แอตทริบิวต์

ในแผนภาพอีอาร์ เราใช้วริพร้อมชื่อภายในวงรีเพื่อแสดงแอตทริบิวต์ เองที่ติดเนื่องที่จะมีแอตทริบิวต์ จากตัวอย่าง จะเห็นได้ว่า สมชาย รักการเรียน มีเลขบัตรประจำตัว เพศ ที่อยู่ วันเกิด และเงินเดือน ที่มีค่าดังนี้ เลขบัตรประจำตัว = 1234567891012, เพศ = ชาย, ที่อยู่ = 10 ถนน พระรามสี่ บางรัก กทม 10500, วันเกิด = 10-5-1995 และเงินเดือน = 25,000

สมหญิง รักการงาน, เลขบัตรประจำตัว = 1234567891015, เพศ = หญิง, ที่อยู่ = 15 ถนนพระรามสี่ บางรัก กทม 10500, วันเกิด = 10-6-1992 และ เงินเดือน = 35,000

สมกมล รักการบ้าน, เลขบัตรประจำตัว = 3234567891012, เพศ = หญิง, ที่อยู่ = 20 ถนนพระรามสี่ บางรัก กทม 10500, วันเกิด = 2-3-1985 และ เงินเดือน = 250,000

ทั้งชื่อ นามสกุล เลขบัตรประจำตัว เพศ ที่อยู่ วันเกิด และเงินเดือน คือ แอตทริบิวต์ หรือในแผนภาพที่แสดงในรูปที่ 18 ให้ชื่อแอตทริบิวต์เหล่านี้ว่า Name, Lastname, ID, Address, Birthdate, Salary

แอตทริบิวต์จะมีค่า (value) และมีชุดของข้อมูล (value set) ที่เหมาะสม เช่น จำนวนเต็ม (integer) ข้อความ (string) ตัวเลข (numeric) เป็นต้น

สังเกตจากแผนภาพในรูปที่ 18 จะเห็นว่า เราแทนแอตทริบิวต์ด้วยวงรี ส่วนชื่อแอตทริบิวต์จะขึ้นต้นด้วยตัวพิมพ์ใหญ่แล้วตามด้วยตัวพิมพ์เล็กหรืออาจแทรกตัวพิมพ์ใหญ่ในคำหลังเพื่อให้อ่านได้ง่าย เช่น StreetAddress เป็นต้น

ทั้งนี้ ทุก ๆ เอนทิตีความมีแอตทริบิวต์ที่เป็นคีย์ เรียกว่า คีย์แอตทริบิวต์ (key attribute) ที่ทำให้ข้อมูลแต่ละบรรทัดในฐานข้อมูลนั้นไม่ซ้ำกัน เช่น เอนทิตี EMPLOYEE มี EmployeeID เป็นคีย์ของเอนทิตินั้นเอง ให้ขีดเส้นใต้ชื่อแอตทริบิวต์เพื่อแสดงความเป็นคีย์

อย่างไรก็ตี มีความเป็นไปได้ที่เอนทิตีหนึ่ง ๆ อาจมีคีย์มากกว่าหนึ่งคีย์ เช่น เอนทิตีไทย CAR อาจมีคีย์คือ Vin หมายถึงเลขตัวถังรถ (Vehicle Identification Number), หรือ LicensePlate หมายถึง ทะเบียนรถ อันประกอบด้วยเลขทะเบียนกับจังหวัด (RegistrationNumber, Province) เป็นต้น

นอกจากนั้น คีย์แอตทริบิวต์ ไม่จำเป็นต้องเป็นแอตทริบิวต์เดียว หากแอตทริบิวต์เพียงตัวเดียวไม่ทำให้ข้อมูลแต่ละบรรทัดไม่ซ้ำ จึงจะต้องกำหนดแอตทริบิวต์หลายตัวมาเป็นคีย์ของเอนทิตินั้นให้ได้ เช่น LicensePlate ที่ยกตัวอย่างข้างต้น เป็นคีย์ที่ประกอบด้วยแอตทริบิวต์ 2 แอตทริบิวต์ คือ RegistrationNumber กับ Province เป็นต้น

ดังนั้น จากแผนภาพอีกครั้งในรูปที่ 18 เราสามารถแยกเอนทิตี และแอตทริบิวต์ได้ดังแสดงในรูปที่ 20 คือมีเอนทิตี้ไทย 4 ตัว ได้แก่ DEPARTMENT, PROJECT, EMPLOYEE และ DEPENDENT หากเขียนแทนด้วยรูปแบบนี้จะไม่สามารถแยกได้ว่าเป็นเอนทิตี้ไทยหรือวีเอนทิตี้ไทย

แต่ละเอนทิตี้ไปทั้งชนิดธรรมด้าและชนิดวีค จะประกอบด้วยแอตทริบิวต์ต่าง ๆ จากรูปนี้จะเห็นว่าความสามารถใช้เครื่องหมายวงเล็บเป็นคู่ {} เพื่อแสดงว่าเป็นแอตทริบิวต์หลายค่า และใช้เครื่องหมายวงเล็บ () แสดงแอตทริบิวต์ร่วมได้อีกด้วย

**DEPARTMENT**

Name, Number, {Locations}, Manager, ManagerStartDate

**PROJECT**

Name, Number, Location, ControllingDepartment

**EMPLOYEE**

Name (FName, MInit, LName), SSN, Sex, Address, Salary, BirthDate, Department, Supervisor, {WorksOn (Project, Hours)}

**DEPENDENT**

Employee, DependentName, Sex, BirthDate, Relationship

รูปที่ 20 เอนทิตี้และแอตทริบิวต์จากแผนภาพอธิบายดังนี้

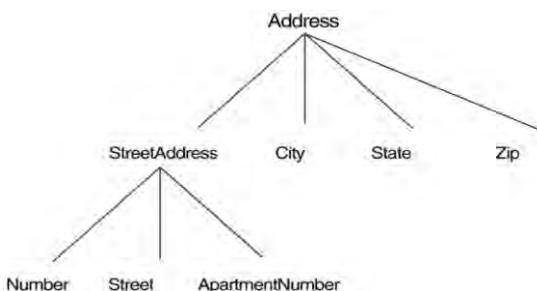
แอตทริบิวต์มีหลายประเภท ขึ้นอยู่กับมุ่งมองในการพิจารณาแยกประเภท เราจะสามารถแบ่งออกได้หลายแบบ

แบ่งตามลักษณะการแยกย่อย สามารถจำแนกได้เป็นสองประเภท คือ

1. แอตทริบิวต์อย่างง่าย (simple attribute)
2. แอตทริบิวต์ร่วม (composite attribute)

โดยแอตทริบิวต์อย่างง่ายจะไม่สามารถจะแบ่งเป็นส่วนประกอบย่อย ๆ ของแอตทริบิวต์ลงมาได้อีก เช่น เมื่อพิจารณาเอนทิตี้ EMPLOYEE เราจะพบว่า แอตทริบิวต์ ID และ SEX เป็นแอตทริบิวต์อย่างง่าย คือ ไม่สามารถแยกย่อยลงมาได้อีกในขณะที่แอตทริบิวต์ร่วมเป็นแอตทริบิวต์ที่สามารถแยกย่อยลงมาได้อีก เช่น ADDRESS จะสามารถแยกส่วนย่อยได้เป็น (StreetAddress,

City, State, Zip) เป็นต้น ให้แสดงแอ็ตทริบิวต์ร่วมด้วยการเขียนชุดของ แอ็ตทริบิวต์ร่วมแต่ละตัวด้วยวงรีพร้อมชื่อเมืองเดิม แต่ให้วงรีเหล่านี้แตก เส้นออกจากแอ็ตทริบิวต์ต้นของชุดนั้น ในขณะเดียวกันลักษณะของ แอ็ตทริบิวต์ร่วมนั้นสามารถที่จะประกอบช้อนกันได้มากกว่านี้ขึ้นด้วย เช่นเดียวกัน เพราะในกรณีของ StreetAddress ยังสามารถแยกย่อยออกได้ เป็น (Number, Street, ApartmentNumber) ดังตัวอย่างในรูปที่ 21



รูปที่ 21 แอ็ตทริบิวต์ร่วม

แบ่งตามค่า สามารถจำแนกได้เป็นสองประเภท คือ

1. แอ็ตทริบิวต์ค่าเดียว (single-valued attribute)
2. แอ็ตทริบิวต์หลายค่า (multi-valued attribute)

โดยแอ็ตทริบิวต์ค่าเดียวจะจัดเก็บค่าข้อมูลเพียงค่าเดียว เช่น AGE เก็บค่า อายุเพียงค่าเดียว หรือ SALARY เก็บค่าเงินเดือนเพียงค่าเดียว ในขณะที่ LOCATION เป็นแอ็ตทริบิวต์หลายค่า จึงเก็บค่าสถานที่ตั้งได้หลายค่า เรา แสดงแอ็ตทริบิวต์หลายค่าด้วยวงรีเส้นคู่

ตัวอย่างเพิ่มเติม เช่น หากเราต้องการออกแบบให้เก็บข้อมูลเลขหมายโทรศัพท์ของคนคนหนึ่งได้มากกว่าหนึ่งเลขหมาย ก็สามารถกำหนดใช้แอตทริบิวต์แบบหลายค่าได้

แบ่งตามการจัดเก็บ จะได้แอตทริบิวต์สองประเภทคือ

1. แอตทริบิวต์ค่าบันทึก (stored attribute)
2. แอตทริบิวต์ค่าคำนวณ (derived attribute)

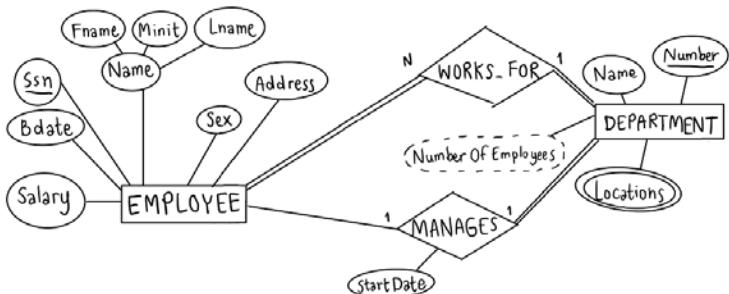
โดยแอตทริบิวต์ค่าบันทึกจะจัดเก็บค่าที่ผู้ใช้งานหรือโปรแกรมเมอร์ระบุหรือพิมพ์เข้ามาเก็บในฐานข้อมูล เช่น BIRTHDATE, NAME, LNAME จำเป็นต้องมีการระบุค่าจากผู้ใช้ ส่วนแอตทริบิวต์ค่าคำนวณ จะไม่มาจากผู้ใช้ แต่จะเกิดจากการคำนวณค่ามาจากแอตทริบิวต์อื่นแล้วมาเก็บไว้ เช่น AGE คำนวณจาก BIRTHDATE ได้เสมอ และทำให้ลดความผิดพลาดได้ดี ให้แสดงแอตทริบิวต์ค่าคำนวณด้วยวงรีส์เส้นประ

สังเกตว่าแต่ละแอตทริบิวต์ก็จะมีค่าเก็บไว้ ไม่ว่าจะมาจากการคำนวณหรือผู้ใช้ใส่เข้ามา ก็ตี แต่ก็มีกรณีที่แอตทริบิวต์หนึ่ง ๆ อาจไม่มีค่าที่ควรจัดเก็บหมายถึง อาจไม่มีข้อมูลที่เหมาะสม หรืออาจไม่รู้ข้อมูลนั้นก็ได้ ในกรณีนี้ฐานข้อมูลจะเก็บค่าพิเศษเรียกว่า ค่าว่าง (null value) ค่าว่างนี้สำคัญมากในการจัดเก็บข้อมูลลงในฐานข้อมูล ในกรณีแอตทริบิวต์ใดไม่มีค่าที่สามารถจัดเก็บได้ ให้กำหนดเป็น ค่าว่าง หรือ นัล (null) เพื่อแสดงความว่างนั้น เช่น จากรูปที่ 21 มีแอตทริบิวต์ StreetAddress ซึ่งแตกเป็น Number, Street และ ApartmentNumber ที่ออกแบบไว้เก็บหมายเลขอห้องของอพาร์ตเมนต์ ที่อยู่ แต่เนื่องจากไม่ใช่ทุกคนจะมีที่อยู่อาศัยแบบอพาร์ตเมนต์ จึงอาจไม่มี

ข้อมูล ApartmentNumber เช่นกรณีอย่างนี้ ก็จะเก็บค่า นัล ไว้ใน แอตทริบิวต์ดังกล่าวแทน ห้ามเก็บค่า 0 หรือเคาระรุค เพราะฐานข้อมูลจะ เข้าใจว่าเก็บค่าศูนย์หรือวรรคแล้วส่งผลให้ข้อมูลผิดพลาดได้

### 3.3.3 รีเลชันชิป

รีเลชันชิป ทำหน้าที่เชื่อมโยงระหว่างเอนทิตี้ไปตั้งแต่ 2 เอ็นทิตี้ไปขึ้นไป เอียนแสดงด้วยสี่เหลี่ยมข้าวหลามตัดพร้อมระบุชื่อข้างใน แต่ละรีเลชันชิปจะ มีชื่อสื่อถึงวัตถุประสงค์ของรีเลชันชิปนั้น ๆ เช่น เราอาจสร้าง รีเลชันชิประหว่าง EMPLOYEE กับ DEPARTMENT ให้มีความสัมพันธ์กัน เมื่อ มี EMPLOYEE เป็นพนักงานอยู่ใน DEPARTMENT เป็นต้น ในกรณีนี้ รีเลชันชิประหว่าง EMPLOYEE กับ DEPARTMENT ไม่จำเป็นต้องมีรีเลชัน ชิปแบบเดียบ เราสามารถแสดงรีเลชันชิปที่สอดคล้องกับบทบาทรีเลชันชิป เหล่านี้ได้ อยiktัวอย่างประกอบดังแสดงในรูปที่ 22



รูปที่ 22 รีเลชันชิปที่สอดคล้องกับบทบาท

สมมุติว่าเราต้องการแสดงรีเลชันชิประหว่าง EMPLOYEE ว่าใครทำงานใน DEPARTMENT ได้ ให้สร้างรีเลชันชิปซึ่งว่า WORKS\_FOR เชื่อมระหว่าง EMPLOYEE และ DEPARTMENT

ที่นี่เราอาจกำหนดรีเลชันชิประหว่างเอนทิตี้ไปปัจจุบันเดิม ที่ต่างบทบาทความสัมพันธ์ก็ได้ เช่น ต้องการแสดงว่า EMPLOYEE สมหญิง เป็นผู้จัดการ DEPARTMENT การเงิน ก็ให้สร้างรีเลชันชิประหว่าง EMPLOYEE กับ DEPARTMENT ซึ่งว่า MANAGES เป็นต้น

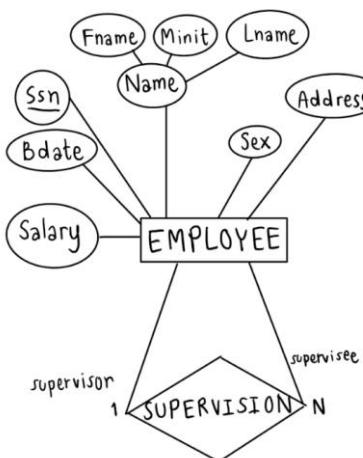
เมื่อนำมาข้อมูลรีเลชันชิปประเภทเดียวกันมาร่วมไว้ด้วยกัน จะเรียกว่า รีเลชันชิปไปป (relationship type) ซึ่งเป็นไปในทำนองเดียวกันกับ เอนทิตี้ไปป จากตัวอย่างข้างบน เราต้องเรียก WORKS\_ON และ MANAGES ว่าเป็นรีเลชันชิปไปป

จากแผนภาพอิอาร์ก่อนหน้านี้ สรุปได้ว่ามีรีเลชันชิปไปปทั้งหมด 6 แบบ ความสัมพันธ์ คือ WORKS\_FOR, MANAGES, CONTROLS, WORKS\_ON, SUPERVISION และ DEPENDENTS\_OF

สังเกตว่าการเขียนชื่อแบบความสัมพันธ์จะใช้ตัวพิมพ์ใหญ่ทั้งหมด เหมือนกับ การเขียนชื่อเอนทิตี้ไปป

ส่วนรีเลชันชิปไปป DEPENDENTS\_OF นั้น เขียนด้วยสัญลักษณ์ สี่เหลี่ยม ข้าวหลามตัดสีน้ำเงิน เพื่อแสดงความเชื่อมโยงกับ วีคเอนทิตี้ไปป เราจึงเรียกรีเลชันชิปนี้ว่า วีครีเลชันชิป (weak relationship)

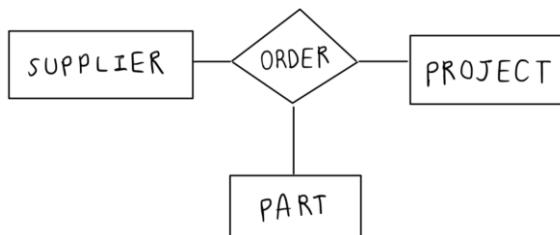
ยังมีความสัมพันธ์แบบพิเศษ คือแบบที่ เชื่อมความสัมพันธ์วนกลับไปยัง  
เอนทิตี้ใหม่ เรียกเหตุการณ์นี้ว่า เกิดความสัมพันธ์แบบเวียนเกิด  
(recursive relationship) จากตัวอย่างในรูปที่ 23 จะเห็นว่า relation ชื่อ  
SUPERVISION ที่มีเอนทิตี้ใหม่ทั้งสองข้างเป็นเอนทิตี้ใหม่  
เดียวกัน คือ EMPLOYEE นั่นเอง



รูปที่ 23 รีเลชันชิปแบบเวียนเกิด

รีเลชันชิปใหม่จะมีระดับหรือดีกรีความสัมพันธ์ (degree of relationship type) กำหนดจากจำนวนเอนทิตี้ใหม่ที่อยู่รอบข้าง หากรีเลชันชิปใหม่นั้น เชื่อมเอนทิตี้ใหม่ 2 เอนทิตี้ใหม่ ก็จะกล่าวได้ว่ามีดีกรีรีเลชันชิปใหม่เป็น 2 หรือเรียกว่าแบบใบหนารี (binary) ซึ่งจากแผนภาพอีกครั้ง จะเห็นว่าทั้ง MANAGES และ WORKS\_ON ต่างก็มีดีกรีเท่ากับ 2 หรือเรียกว่ารีเลชันชิปแบบใบหนารี

หากเริ่มเขียนชิปไทยมีการเชื่อมโยง 3 เอนทิตี้ไปเป็นข้อความเดียว ก็จะมีดีกรีเท่ากับ 3 หรือเรียกว่าเทอนารี (ternary) แสดงในรูปที่ 24 และมากกว่าสาม ก็จะเรียกว่า เออนนารี (n-ary)



รูปที่ 24 รีเลชันชิปแบบเทอนารี

ที่ผ่านมาเราได้พูดถึงแต่แอ็ตทริบิวต์ปกติทั่วไปคือแอ็ตทริบิวต์ของเอนทิตี้แต่ยังมีแอ็ตทริบิวต์ที่เกิดจากรีเลชันชิปได้อีกด้วย สังเกตจากแผนภาพวีอาร์ จะเห็นแอ็ตทริบิวต์ Hours ดังแสดงในรูปที่ 25 ที่เกิดจากรีเลชันชิประหว่าง EMPLOYEE กับ PROJECT เพราะบันทึกเวลาการทำงานของ EMPLOYEE ใน PROJECT นั้น ๆ ข้อมูลนี้จะไม่เกิดขึ้น หากไม่มีรีเลชันชิประหว่างเอนทิตี้ทั้งสอง



รูปที่ 25 แอ็ตทริบิวต์ของรีเลชันชิป

ยังมีเรื่องที่สำคัญที่ยังไม่ได้กล่าวถึงอีกเรื่อง คือความสามารถระบุเงื่อนไขบังคับให้กับรีเลชันชิปໄທป์ใด ๆ ทั้งนี้เพื่อชี้ว่ารีเลชันชิปนั้น ๆ สามารถเกิดขึ้นได้กี่ครั้ง เงื่อนไขบังคับที่สำคัญมี 2 อย่างคือ

- cardinality
- เงื่อนไขบังคับเข้าร่วม

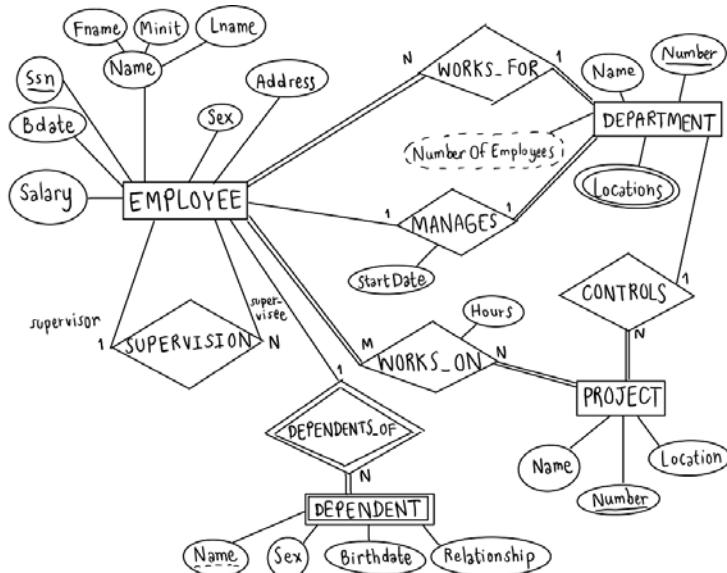
### 3.3.4 cardinality

cardinality (cardinality) คือสัดส่วน (ratio) ของรีเลชันชิป ซึ่งเป็นตัวบ่งชี้ว่าจำนวนรีเลชันชิปมากที่สุดที่แต่ละเอนิทีสามารถเข้าร่วมในรีเลชันชิปนั้นได้มีค่าเท่าใด ซึ่งมีความเป็นไปได้ 3 แบบ คือ

- 1:1 แสดงรีเลชันชิปแบบหนึ่งต่อหนึ่ง นิยมอ่านว่า วันทูวัน (one-to-one)
- 1:N หรือ N:1 แสดงรีเลชันชิปแบบหนึ่งต่อหลายหรือหลายต่อหนึ่ง นิยมอ่านว่า วันทูเมเน (one-to-many) หรือ เมนีทูวัน (many-to-one)
- M:N แสดงรีเลชันชิปแบบหลายต่อหลาย นิยมอ่านว่า เมนีทูเมเน (many-to-many)

ขอยกตัวอย่างประกอบเพื่อให้เข้าใจได้ง่ายขึ้น ให้ดูแผนภาพอีกรูปเดิมที่นำมาแสดงอีกครั้งในรูปที่ 26 จะเห็นรีเลชันชิปໄທป์ MANAGES ที่เชื่อมระหว่าง EMPLOYEE กับ DEPARTMENT ในกรณีนี้ จะเห็นเลข 1 กำกับอยู่บนเส้นที่เชื่อมระหว่างรีเลชันชิปໄທป์นั้นไปยัง EMPLOYEE และ DEPARTMENT ซึ่ง

หมายความว่า EMPLOYEE 1 คน จะ MANAGE ได้อย่างมาก 1 DEPARTMENT และ แต่ละ DEPARTMENT มี EMPLOYEE ที่เป็นผู้จัดการ หรือ MANAGE ได้อย่างมากคนเดียวเท่านั้น



รูปที่ 26 แผนภาพอธิบายแสดงระบบฐานข้อมูลของบริษัทดีบี (ช้า)

ถัดไปให้ดูรีเลชันชิปไปที่ WORKS\_FOR ที่มีเลข 1 กำกับบนสันรีเลชันชิปที่ซึ่งไปยัง DEPARTMENT และ N ซึ่งไปยัง EMPLOYEE ในกรณีนี้เป็นความสัมพันธ์แบบ 1:N ซึ่งหมายความว่า EMPLOYEE 1 คน อาจทำงานได้อย่างมาก 1 DEPARTMENT ทว่าแต่ละ DEPARTMENT อาจมี EMPLOYEE ได้หลายคน คือกี่คนก็ได้

จากนั้นให้ดูรีเลชันชิปไปที่ WORKS\_ON จะเห็นอักษร M และ N กำกับบนเส้นเชื่อมระหว่าง WORKS\_ON ที่ซึ่งเป็น EMPLOYEE และ PROJECT นั่นหมายความว่า EMPLOYEE คนหนึ่งอาจทำงานได้หลาย PROJECT และ PROJECT หนึ่ง ๆ จะมี EMPLOYEE ได้หลายคน สังเกตว่าการกำกับแบบ M:N ให้ใส่อักษร M ไว้ข้างหนึ่ง และ N ไว้อีกข้างหนึ่ง อาจเป็นข้างใดก็ได้

### 3.3.5 เงื่อนไขบังคับเข้าร่วม

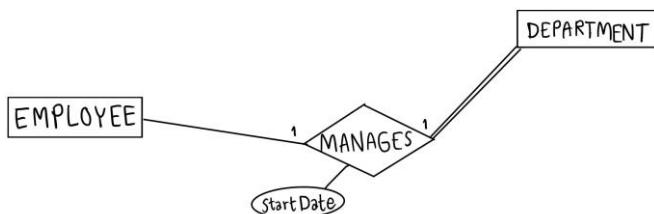
เงื่อนไขบังคับเข้าร่วม (participation constraint) เป็นการกำหนดค่าต่ำสุดของความสัมพันธ์นั้น หรืออาจมองว่าคือจำนวนรีเลชันชิปน้อยสุดที่ต้องมีก็ได้ ค่าที่ว่านี้มีความเป็นไปได้แค่ 2 แบบคือ

- การเข้าร่วมแบบไม่บังคับ
- การเข้าร่วมแบบบังคับ

การเข้าร่วมแบบไม่บังคับ (partial participation) ภาษาอังกฤษใช้คำว่า partial ที่แปลตรง ๆ ได้ว่าบางส่วน แต่ในตรานี้ขอใช้คำว่าไม่บังคับ เพราะโดยความหมายนั้น ต้องการสื่อว่ารีเลชันชิปนั้นอาจเกิดหรือไม่เกิดก็ได้ ซึ่งตรงข้ามกับ การเข้าร่วมแบบบังคับ (total participation) ที่หมายความว่า ต้องมีรีเลชันชิปนั้นเกิดขึ้นเสมอ อาจมองอีกนัยหนึ่งว่าเป็นค่าต่ำสุดก็ได้ ซึ่งก็คืออย่างน้อยต้องมี 1 รีเลชันชิปเกิดขึ้นนั่นเอง

การเข้าร่วมแบบไม่บังคับ ให้แสดงด้วยเส้นเชื่อมแบบเส้นเดี่ยว ส่วนการเข้าร่วมแบบบังคับ ให้แสดงด้วยเส้นคู่

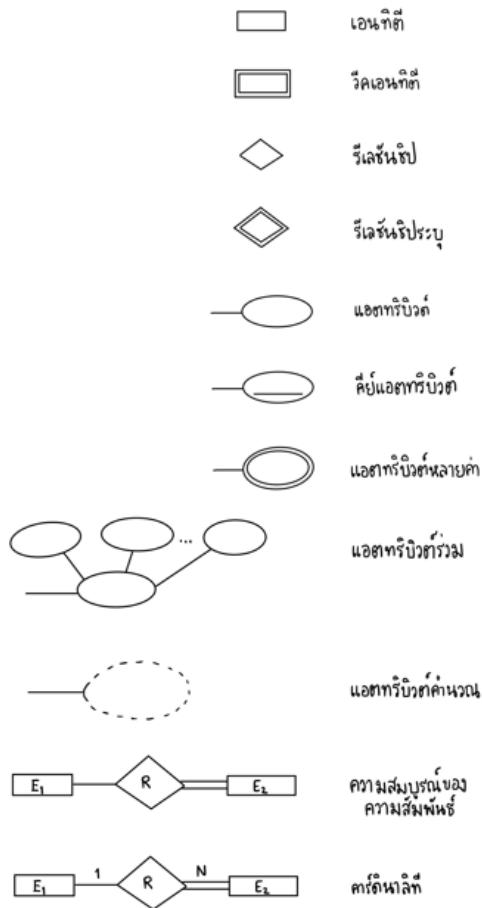
เพื่อให้เห็นภาพชัดเจน ขยายตัวอย่างรีเลชันชิปไป MANAGES แสดงในรูปที่ 27 ซึ่งเข้มระหว่าง EMPLOYEE กับ DEPARTMENT ซึ่งจะเห็นได้ว่ามีเส้นเดี่ยวเชื่อมระหว่างเอนทิตี้ไป EMPLOYEE กับ DEPARTMENT รีเลชันชิปไป MANAGES นั้นหมายความว่าอาจไม่มี EMPLOYEE ที่มีบทบาทผู้จัดการเลยก็ได้ แต่ในขณะเดียวกัน เราจะเห็นเส้นคู่กำกับระหว่าง MANAGES กับ DEPARTMENT นั้นหมายความว่า DEPARTMENT ต้องมีผู้จัดการอย่างน้อย 1 คน



รูปที่ 27 เส้นไขบงคับเข้าร่วมแบบบังคับและแบบไม่บังคับ

### 3.3.6 สรุปสัญกรณ์อีอาร์

สัญกรณ์ (notation) ที่ใช้ในตำรานี้ มีดังแสดงในรูปที่ 28

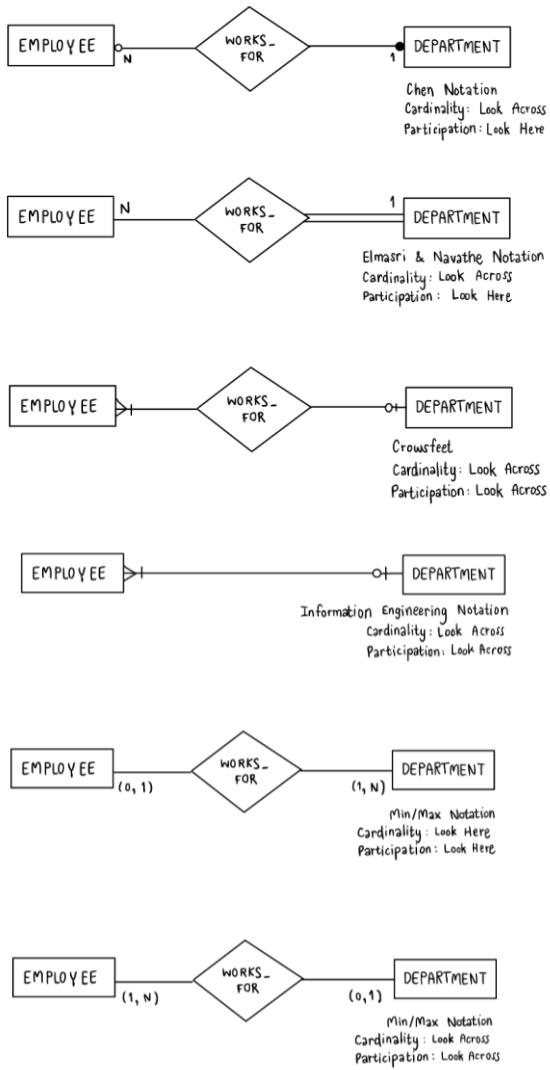


รูปที่ 28 สรุปสัญกรณ์อีอาร์

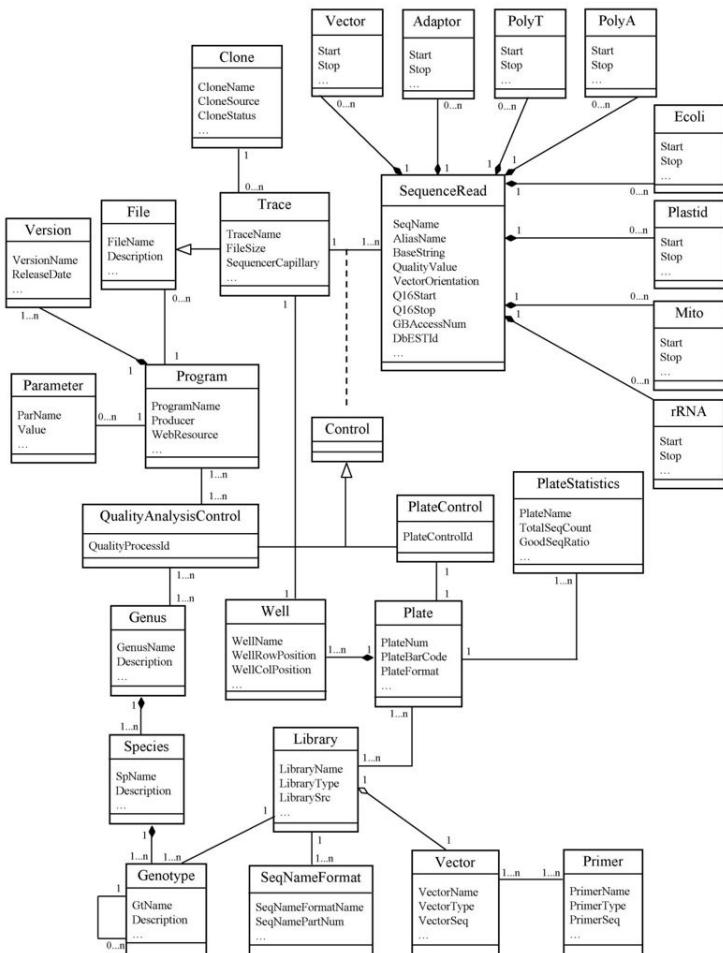
### 3.3.7 สัญกรณ์เพิ่มเติม

ตำราว่าด้วยเรื่องแผนภาพอีอาร์นั้น มีการนำเสนองานใช้สัญกรณ์ (notation) ต่างกันอยู่บ้าง ที่ใช้ในตำนานจะเป็นวิธีของศาสตราจารย์เอลมาศรีและศาสตราจารย์นาวาธ [1] อย่างไรก็ได้ ยังมีสัญกรณ์อีกหลายชุด ต่างก็สร้างมาเพื่อใช้แสดงแผนภาพระดับแนวคิดในการออกแบบฐานข้อมูลทั้งสิ้น แต่ก็อาจมีรายละเอียดต่างกันบ้าง

รูปที่ 29 จาก [26] เปรียบเทียบสัญกรณ์แบบต่าง ๆ และรูปที่ 30 แสดงตัวอย่างสัญกรณ์ยูเอ็มแอล (Unified Modeling Language : UML) [27] ซึ่งถ้านำไปเทียบกับแผนภาพอีอาร์ด้วยสัญกรณ์ชุดก่อนก็จะได้ความหมายคล้ายคลึงกัน



รูปที่ 29 เปรียบเทียบสัญกรณ์แบบต่าง ๆ



รูปที่ 30 สัญกรณ์ย่อเอ็มแอล

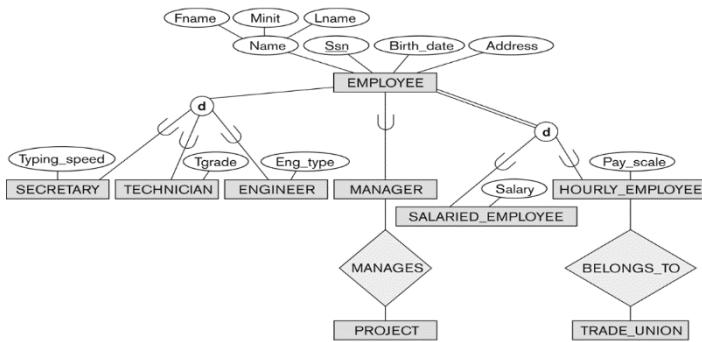
### 3.4 แบบจำลองอีอีอาร์

ที่ผ่านมาได้กล่าวถึงแบบจำลองอีอาร์เพื่อแสดงระบบฐานข้อมูลแบบที่นิยมใช้ตั้งแต่ยุคที่เริ่มใช้ฐานข้อมูลกันอย่างแพร่หลาย อย่างไรก็ต้องมีในยุคต่อ ๆ มาที่เทคโนโลยีมีการเปลี่ยนแปลงไป ก็มีการนำเสนอแบบจำลองใหม่ ๆ ที่รองรับแนวคิดที่ทันสมัยขึ้นเรื่อย ๆ ที่สำคัญคือยุคที่เปลี่ยนแปลงมานิยมการเขียนโปรแกรมด้วยภาษาเชิงวัตถุ (Object-Oriented language) ได้มีผู้ปรับปรุงแบบจำลองอีอาร์แบบขยายหรือแบบปรับปรุง (Extended หรือ Enhanced ER model) [28, 29] หรือเรียกว่า แบบจำลองอีอีอาร์ (EER model)

เพื่อรองรับแนวคิดเชิงวัตถุ แบบจำลองอีอีอาร์จึงต้องพนักความสามารถเชิงวัตถุให้เข้ากับฐานข้อมูล แนวคิดดังกล่าว ได้แก่

1. ชั้บคลาส/ชูเปอร์คลาส (subclass/superclass)
2. สเปเชียลไลเซชัน/เจเนอเรชัน (specialization/generalization)
3. การรับทอด (inheritance)

การแสดงแบบจำลองอีอีอาร์จะใช้แผนภาพเรียกว่าแผนภาพอีอีอาร์ มีลักษณะคล้ายแผนภาพอีอาร์ และมีสัญกรณ์เพิ่มเติมเพื่อรองรับแนวคิด 3 ข้อที่ได้กล่าวไป รูปที่ 31 แสดงตัวอย่างแผนภาพอีอีอาร์ อ้างอิงมาจาก [2]

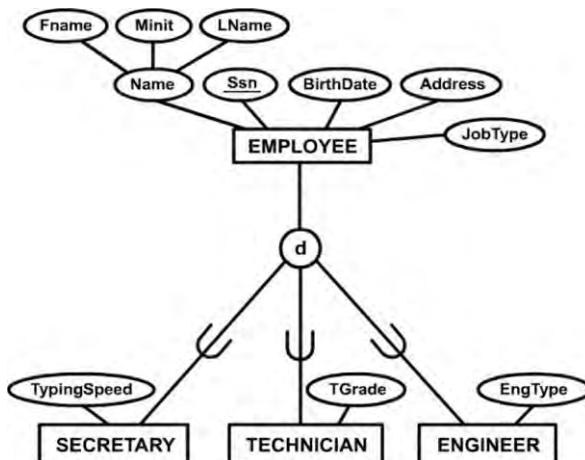


รูปที่ 31 แผนภาพอีอาร์

### 3.4.1 ชั้บคลาส/ชูเปอร์คลาส

ชั้บคลาส/ชูเปอร์คลาส (subclass/superclass) คือการประยุกต์แนวคิดเชิงวัตถุมาใช้กับฐานข้อมูล ในลักษณะคลาส โดยให้มองว่าเอนทิตี้แต่ละเอนทิตี้คือคลาส ดังนั้น เมื่อนำแนวคิดว่าคลาสสามารถแตกย่อยได้ ก็เรียก คลาสหลัก ว่า ชูเปอร์คลาส และเรียก คลาสย่อย ว่า ชั้บคลาส ตำรานี้จะเรียกว่า ชูเปอร์คลาสและชั้บคลาส ซึ่งก็คือเอนทิตี้ที่เป็นนั้นเอง แต่เป็นเอนทิตี้ใหม่ที่แยกย่อยได้ ชูเปอร์คลาสก็คือเอนทิตี้หลัก ส่วนชั้บคลาสคือเอนทิตี้ย่อยมาจากการแยกย่อย ให้เข้าใจว่าคือเอนทิตี้ใหม่ที่มีคุณสมบัติเป็นชูเปอร์คลาสหรือชั้บคลาสตามหน้าที่ที่กำหนด สำหรับเอนทิตี้ที่ไม่ได้มีการแตกย่อย ก็ให้อ่านว่า เอนทิตี้ใหม่ก็ได้ การแตกย่อยนี้ทำให้เกิดความสัมพันธ์แบบชูเปอร์คลาสชั้บคลาส

บางตำรา [13] ก็เรียกความสัมพันธ์ลักษณะซึ่งคลาสชูเบอร์คลาสนี้ว่า ความสัมพันธ์อีสเอหรืออีสแอน (IS-A or IS-AN relationship) เช่น จากรูปที่ 32 อ่านได้ว่า SECRETARY IS-AN EMPLOYEE, TECHNICIAN IS-AN EMPLOYEE เป็นต้น



รูปที่ 32 ชูเบอร์คลาสซับคลาส

รูปที่ 32 แสดงชูเบอร์คลาส EMPLOYEE ที่แตกซับคลาสได้ 3 ซับคลาส คือ SECRETARY, TECHNICIAN และ ENGINEER นอกจากนั้น แต่ละซับคลาสยัง มีแอตทริบิวต์เพิ่ม เรียกว่า แอตทริบิวต์เฉพาะ (specific attribute) คือ ซับคลาส SECRETARY มีแอตทริบิวต์เฉพาะคือ TypingSpeed, TECHNICIAN มีแอตทริบิวต์เฉพาะคือ TGrade และ ENGINEER มี แอตทริบิวต์เฉพาะคือ EngType

สังเกตความแตกต่างระหว่างแบบจำลองอีอาร์และแบบจำลองอีอาร์ จะเห็นได้ว่าแบบจำลองอีอาร์เดิมนั้นไม่สามารถรองรับเงื่อนไขต่อไปนี้ได้ หากใช้แบบเดิมก็จะเป็นต้องนำแอตทริบิวต์ TypingSpeed, TGrade และ EngType ไปเพิ่มอยู่ในเอนทิตี้ไทป์ EMPLOYEE

ทั้งนี้เอนทิตี้ใด ๆ ไม่สามารถอยู่ในฐานข้อมูลโดยเป็นเพียงสมาชิกของชั้บคลาสได้ จะต้องเป็นสมาชิกของชั้นเบอร์คลาสด้วย ในขณะที่สมาชิกของชั้นเบอร์คลาสอาจเป็นสมาชิกของคลาสอยู่ก็ได้ เช่น วิศวกรที่เป็นลูกจ้างประเภทเงินเดือน อาจอยู่ในสองคลาสอยู่คือ ENGINEER และ SALARIED\_EMPLOYEE แต่ทั้งนี้ไม่จำเป็นว่าเอนทิตี้ทุกตัวที่อยู่ในชั้นเบอร์คลาสจะต้องเป็นสมาชิกของคลาสอยู่เสมอไป

### 3.4.2 การรับทอด

การรับทอด (inheritance) ในแนวคิดเชิงวัตถุ คือ การคงความสามารถบางอย่างของวัตถุที่สัมพันธ์กัน เมื่อนำมาประยุกต์กับฐานข้อมูล ก็คือ การกำหนดให้เอนทิตี้ที่เกี่ยวข้องกันจะต้องรับทอดคุณสมบัติซึ่งกันและกัน ด้วย การรับทอดนี้จะเกิดขึ้นเมื่อมีความสัมพันธ์แบบชั้บคลาสชั้นเบอร์คลาส ทำให้ชั้บคลาสรับทอดคุณสมบัติของชั้นเบอร์คลาสไว้นั่นเอง

ความสำคัญคือ ชั้บคลาสทั้งหมดจะต้องคงคุณสมบัติจากชั้นเบอร์คลาสด้วยเสมอ แต่อาจมีแอตทริบิวต์พิเศษเพิ่มมาในแต่ละชั้บคลาสได้

ด้านซ้ายของรูปที่ 31 แสดงชั้นเบอร์คลาส EMPLOYEE และชั้บคลาส SECRETARY, TECHNICIAN และ ENGINEER ทั้ง 3 ชั้บคลาสนี้จะคง

แอตทริบิวต์ทั้งหมดของชูเปอร์คลาส EMPLOYEE ได้แก่ Name, SSN, Birthdate และ Address นอกจากนั้นแต่ละชั้บคลาสยังมีแอตทริบิวต์เพิ่มคือชั้บคลาส SECRETARY เพิ่มแอตทริบิวต์ TypingSpeed, TECHNICIAN เพิ่มแอตทริบิวต์ TGrade และ ENGINEER เพิ่มแอตทริบิวต์ EngType

นั่นหมายความว่าจากการรับ托จะทำให้ชั้บคลาส SECRETARY มีแอตทริบิวต์คือ Name, SSN, Birthdate, Address, TypingSpeed ส่วนชั้บคลาส TECHNICIAN และแอตทริบิวต์คือ Name, SSN, Birthdate, Address, TGrade และ ENGINEER แอตทริบิวต์คือ Name, SSN, Birthdate, Address, EngType

### 3.4.3 สเปเชียลไอลเซชันและเจเนอรัลไอลเซชัน

สเปเชียลไอลเซชัน (specialization) และ เจเนอรัลไอลเซชัน (generalization) เป็นกระบวนการในการกำหนดสมาชิกของชั้บคลาสและชูเปอร์คลาสทั้งคู่ เพียงแต่ดำเนินการไม่เหมือนกัน กล่าวคือ กระบวนการสเปเชียลไอลเซชันคือการแตกย่อยจากบนลงล่าง (top-down) คือกำหนดชูเปอร์คลาสก่อน แล้วแยกย่อยให้ได้ชั้บคลาส ส่วนเจเนอรัลไอลเซชันนั้นเป็นการรวมชั้บคลาสหลายชั้บคลาสให้ได้ชูเปอร์คลาส หรือมองว่าเป็นการจัดกลุ่มจากล่างขึ้นบน (bottom-up) นั่นเอง

ยกตัวอย่างจากรูปที่ 31 ที่มีชูเปอร์คลาส EMPLOYEE ซึ่งมีความสัมพันธ์กับชั้บคลาส SALARIED\_EMPLOYEE กับ HOURLY\_EMPLOYEE หากกระบวนการกำหนดชูเปอร์คลาสชั้บคลาสนี้ เริ่มคิดจาก EMPLOYEE และแยกเป็น SALARIED\_EMPLOYEE กับ HOURLY\_EMPLOYEE ก็ให้เรียกว่า

เป็นกระบวนการสเปเชียลไลเซชัน ในทางกลับกัน หากกระบวนการนี้เริ่มจากกำหนด SALARIED\_EMPLOYEE กับ HOURLY\_EMPLOYEE ก่อน แล้ว นำมารวมเป็น EMPLOYEE ก็ให้เรียกว่าเป็นกระบวนการเจเนอรัลไลเซชัน

การเขียนแผนภาพอีอาร์ ให้เขียนสัญลักษณ์แสดงความสัมพันธ์แบบชูเบอร์ คลาสซับคลาสด้วยตacheo ลักษณะคล้ายตัว U กำกับบนเส้นความสัมพันธ์นั้น

เมื่อมีมากกว่าหนึ่งชับคลาส ให้เขียนสัญลักษณ์วงกลมเชื่อมชับคลาสทั้งหมด เข้าด้วยกัน และเขียนตacheo บนเส้นความสัมพันธ์จากชับคลาสมาอยังวงกลมนั้น ดังตัวอย่างก่อนหน้า

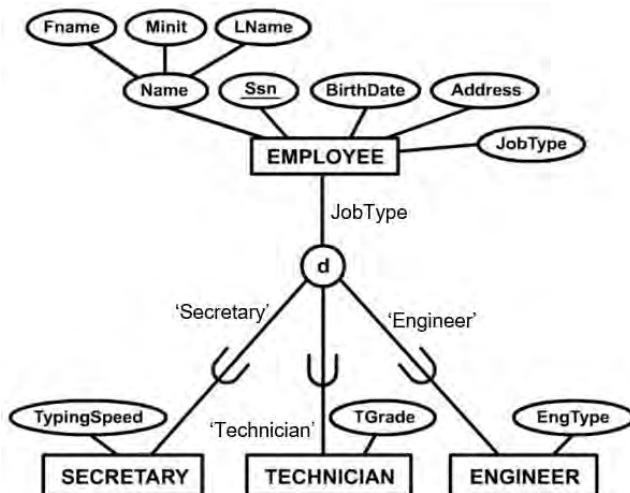
ขอให้สังเกตด้วยว่า ชับคลาสก์สามารถมีรีเลชันชิปกับเอนทิตี้อื่นได้ จากรูป จะพบตัวอย่าง ชับคลาส HOURLY\_EMPLOYEE มีรีเลชันชิปไปที่ BEONGS\_TO เชื่อมกับเอนทิตี้ที่ไป TRADE\_UNION และ ชับคลาส MANAGER มีรีเลชันชิปไปที่ MANAGES เชื่อมกับเอนทิตี้ที่ไป PROJECT ด้วย

### 3.4.4 การกำหนดสมาชิกชับคลาส

การกำหนดหรือระบุสมาชิกชับคลาสสามารถทำได้ 2 วิธี คือ 1) ใช้เงื่อนไขกำหนด หรือ 2) ผู้ใช้กำหนด

ชับคลาสที่ใช้เงื่อนไขกำหนด (predicate-defined subclass หรือ condition-defined subclass) คือชับคลาสที่มีค่าของแอตทริบิวต์ในชูเบอร์ คลาสเป็นตัวระบุว่าเป็นชับคลาสได้ เราเรียกแอตทริบิวต์นี้ว่า แอตทริบิวต์แบบเพรดิเคตกำหนด (defining predicate) วิธีแสดงชับคลาสที่ใช้เงื่อนไขกำหนดนี้ทำได้โดยการเขียนชื่อแอตทริบิวต์ที่เป็นตัวกำหนดลงบนเส้น

ระหว่างชูเปอร์คลาสกับวงกลม และเขียนค่าที่ใช้ในการกำหนดชั้บคลาสบนเส้นระหว่างวงกลมกับชั้บคลาส รูปที่ 33 แสดงตัวอย่างกรณีดังกล่าว จะเห็นว่า แอ็ตทริบิวต์ Job\_Type เป็นแอ็ตทริบิวต์แบบเพรดิเคตกำหนด เมื่อค่าของ Job\_Type เป็น ‘Secretary’ ให้กำหนดชั้บคลาสเป็น SECRETARY เมื่อค่าของ Job\_Type เป็น ‘Technician’ ให้กำหนดชั้บคลาสเป็น TECHNICIAN และเมื่อค่าของ Job\_Type เป็น ‘Engineer’ ให้กำหนดชั้บคลาสเป็น ENGINEER



รูปที่ 33 ชั้บคลาสที่ใช้เงื่อนไขกำหนด

จะเห็นว่ากระบวนการดังกล่าวคือกระบวนการสเปเชียลไลเซชัน และเนื่องจากเป็นการสเปเชียลไลเซชันด้วยแอ็ตทริบิวต์ จึงเรียกกระบวนการนี้ได้ว่ากระบวนการสเปเชียลไลเซชันแบบแอ็ตทริบิวต์กำหนด (attribute defined-specialization)

ในกรณีที่ไม่มีแอ็ตทริบิวต์แบบเพรดิคเตเป็นตัวกำหนดชั้นคลาส เราจะต้องให้ผู้ใช้เป็นผู้กำหนดว่าจะเป็นชั้นคลาสใด เราเรียกชั้นคลาสกรณีนี้ว่าเป็นชั้นคลาสที่ผู้ใช้กำหนด (user-defined subclass)

### 3.4.5 ความมีส่วนร่วมของชั้นคลาส

ความมีส่วนร่วมของชั้นคลาส (disjointness constraint) คือ การระบุประเภทความสัมพันธ์ระหว่างชั้นคลาสชั้นคลาสมีเพียง 2 แบบคือแบบดิสจอยท์ (disjoint) และแบบ โอลเออร์แลป (overlap)

แบบดิสจอยท์นั้นหมายถึงชั้นคลาสหนึ่ง ๆ จะสามารถแยกย่อยได้เป็นชั้นคลาสได้ชั้นคลาสหนึ่งเท่านั้น ตัวอย่างจากรูปจะเห็นว่าชั้นคลาส EMPLOYEE จะเป็นได้เพียง SECRETARY หรือ ENGINEER หรือ TECHNICIAN เท่านั้น ไม่สามารถเป็นสมาชิกได้มากกว่าหนึ่งชั้นคลาส

ส่วนแบบโอลเออร์แลปนั้นหมายถึงชั้นคลาสหนึ่ง ๆ จะเป็นสมาชิกได้มากกว่าหนึ่งชั้นคลาส เช่น EMPLOYEE คนหนึ่งอาจเป็นพนักงานประเภทรับเงินเดือน คือ SALARIED\_EMPLOYEE และ เป็นพนักงานประเภทรับค่าจ้างรายชั่วโมง คือ HOURLY\_EMPLOYEE ก็ได้

ในแผนภาพอีอาร์ ให้เขียน d ในวงกลมเพื่อแสดงความมีส่วนร่วมแบบดิสจอยท์ หรือเขียน o ในวงกลม เพื่อแสดงความมีส่วนร่วมแบบโอลเออร์แลป ดังแสดงในรูปที่ 31

### 3.4.6 ความสมบูรณ์ของความสัมพันธ์

ความสมบูรณ์ของความสัมพันธ์ (completeness constraint) คือการระบุว่าเอนทิตี้ที่เป็นชุดเปอร์คลาสนั้นมีความสัมพันธ์กับชั้บคลาสแบบสมบูรณ์ หรือไม่ หรืออธิบายง่าย ๆ คือ ชุดเปอร์คลาสนั้นจะต้องแยกเป็นชั้บคลาส หรือไม่ ค่าที่เป็นไปได้มีเพียง 2 แบบ คือแบบบังคับ (total) และแบบไม่บังคับ (partial) กรณีบังคับหมายความว่า เอนทิตี้ในชุดเปอร์คลาสนั้น ๆ จะต้องเป็นสมาชิกของชั้บคลาส ถ้าแบบไม่บังคับ หมายความว่าเอนทิตี้ในชุดเปอร์คลาสนั้น ไม่จำเป็นต้องแตกรายอยู่เป็นชั้บคลาส

ในแผนภาพอีอาร์ ให้เขียนเส้นคู่จากชุดเปอร์คลาสสามอย่างวงกลม เพื่อแสดงความสัมพันธ์แบบบังคับ หรือเขียนเส้นเดียว เพื่อแสดงความสัมพันธ์แบบไม่บังคับ

จะเห็นว่ากระบวนการดังกล่าวคือกระบวนการสเปเชียลไลเซชัน เพราะต้องเริ่มพิจารณาจากชุดเปอร์คลาสก่อน และในกรณีที่กำหนดความสมบูรณ์ของความสัมพันธ์เป็นแบบบังคับนั้น ให้เรียกกระบวนการนี้ได้ว่า กระบวนการสเปเชียลไลเซชันแบบบังคับ (total specialization) หากเป็นแบบไม่บังคับ ให้เรียกว่า กระบวนการสเปเชียลไลเซชันแบบไม่บังคับ (partial specialization)

ตั้งนั้น เมื่อคำนึงถึงข้อกำหนดทั้งหมดที่ผ่านมา เราจะมีเจเนอร์ล์ไลเซชัน หรือสเปเชียลไลเซชัน 4 ประเภท คือ

1. ดิสจอยท์-บังคับ (disjoint-total)
2. ดิสจอยท์-ไม่บังคับ (disjoint-partial)
3. โอเวอร์แลป-บังคับ (overlapping-total)
4. โอเวอร์แลป-ไม่บังคับ (overlapping-partial)

### 3.4.7 กฎการแทรกและการลบข้อมูลในอีอาร์

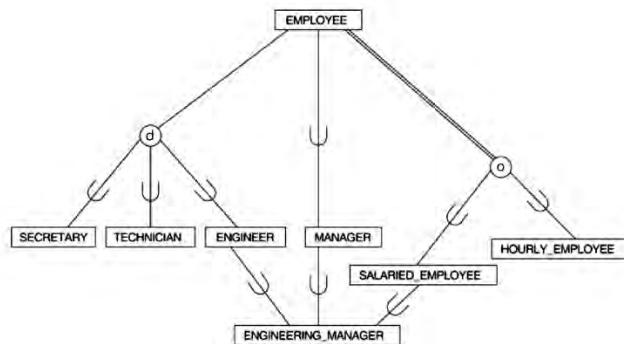
การปรับแก้ข้อมูลในฐานข้อมูลนั้น เมื่อใช้แบบจำลองอีอาร์ จำเป็นต้องคำนึงถึงคุณสมบัติการรับทดลอง เราระบุผลกระบวนการในการรับทดลอง เมื่อได้ก็ตามที่มีการปรับปรุงข้อมูลดังนี้

1. การลบเอนทิตีออกจากชูเปอร์คลาส จะส่งผลให้ต้องลบเอนทิตีในชั้บคลาสทั้งหมดเสมอ
2. การแทรกเอนทิตีที่ชั้บคลาส จะส่งผลให้ต้องแทรกเอนทิตีที่ชูเปอร์คลาสทั้งหมดเสมอ
3. การแทรกเอนทิตีที่ชูเปอร์คลาส หากใช้กระบวนการการสเปเชียลไลเซชันแบบแอ็ตทริบิวต์กำหนด จะส่งผลให้เกิดการแทรกเอนทิตีที่ชั้บคลาสบางชั้บคลาส
4. การแทรกเอนทิตีที่ชูเปอร์คลาส หากใช้กระบวนการการสเปเชียลไลเซชันแบบบังคับ จะส่งผลให้เกิดการแทรกเอนทิตีที่ชั้บคลาสอย่างน้อยหนึ่งชั้บคลาส (หากความมีส่วนร่วมของชั้บคลาส เป็นแบบดิสจอยท์จะมีการแทรกเอนทิตีเพียงหนึ่งชั้บคลาสเท่านั้น)

### 3.4.8 จำนวนชั้นของสเปเชียลไลเซชัน

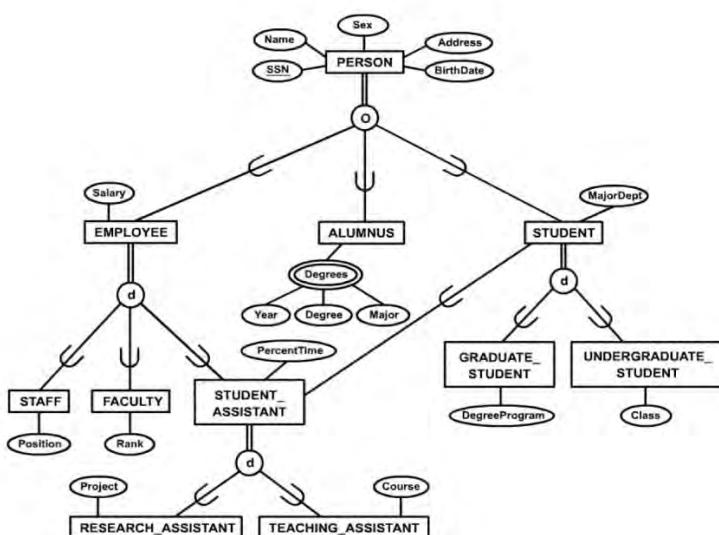
กระบวนการสเปเชียลไลเซชันหรือเจเนอรัลไลเซชันมักไม่เกิดเพียงชั้นเดียว หมายความว่า เมื่อซูเปอร์คลาสได้ถูกแตกย่อยเป็นชั้บคลาสแล้ว ยังอาจถูกแตกย่อยไปเรื่อย ๆ ได้ ถ้ามองกลับกันคือเจเนอรัลไลเซชันหลายทอดนั้นเอง

เมื่อมีการแตกย่อยแล้ว ให้พิจารณาที่ชั้บคลาส หากชั้บคลาสนั้นมีซูเปอร์คลาสเพียงชั้นเดียว ก็จะรับทอดคุณสมบัติจากซูเปอร์คลาสนั้นและซูเปอร์คลาสก่อนหน้านั้นขึ้นไปทุกชั้นจนหมดซูเปอร์คลาสถึงต้นต่อหรือราก (root) คุณสมบัติที่รับทอดในฐานข้อมูลก็คือแอตทริบิวต์ทั้งหมดจากทุกโอนที่ติดไว้ปัจจุบันนี้ และเรียกโครงสร้างความสัมพันธ์นี้ว่า สเปเชียลไลเซชันแบบกำดับชั้น (specialization hierarchies) ซึ่งมีลักษณะโครงสร้างแบบต้นไม้ ตัวอย่างในรูปที่ 34 จะพบว่า ENGINEER เป็นชั้บคลาสของ EMPLOYEE และยังมีชั้บคลาสต่อลงไปอีกชั้นคือ ENGINEERING\_MANAGER



รูปที่ 34 สเปเชียลไลเซชันแบบผลติช

อีกรูปคือเมื่อพิจารณซับคลาสแล้วพบว่ามีชูเปอร์คลาสมากกว่าหนึ่ง ซับคลาสนั้นจะรับทอดคุณสมบัติจากชูเปอร์คลาสทุกชูเปอร์คลาสที่เกี่ยวข้อง คือแอ็ตทริบิวต์ทั้งหมดจากทุกเอนทิตี้ที่เป็นรากและรีเลชันชิปทั้งหมด เราเรียก กรณีนี้ว่า การรับทอดแบบหลาย (multiple inheritance) และเรียก ซับคลาสลักษณะนี้ว่า ซับคลาสร่วม (Shared subclass) และให้เรียก โครงสร้างความสัมพันธ์นี้ว่า สเปเชียลไลเซชันแบบแلاتทิซ (Specialization Lattice) ซึ่งมีโครงสร้างแบบกราฟระบุทิศทาง (Directed acyclic graph) แสดงด้วยอย่างซับคลาสร่วมคือ STUDENT\_ASSISTANT ที่มีชูเปอร์คลาสหลัก 2 ชูเปอร์คลาส ได้แก่ EMPLOYEE และ STUDENT ดังรูปที่ 35 แสดงอี่อาร์ แบบซับช้อน มีทั้งสเปเชียลไลเซชันแบบลำดับชั้นและแบบแلاتทิซ



รูปที่ 35 สเปเชียลไลเซชันแบบลำดับชั้นและแบบแلاتทิซ

### 3.4.9 แคทेगอรี

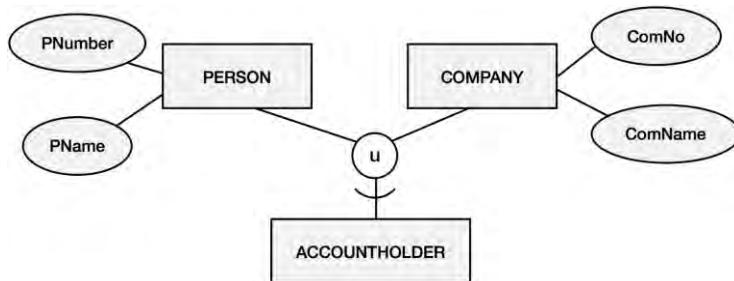
แคทेगอรี (Category) เป็นประเภทหนึ่งของชั้บคลาส เกิดเมื่อชั้บคลาสมีชูเปอร์คลาสหลายชูเปอร์คลาส แต่ชูเปอร์คลาสเหล่านั้นเป็นเอนทิตี้ที่ต่างกัน ดังนั้นการรับทอดจะไม่ใช่การรับทอดแบบหลาย แต่เป็นการรับทอดแบบเลือก (selective inheritance) ในฐานข้อมูลก็คือรับทอดแอ็ตทริบิวต์และรีเลชันชิปไปทั้งหมดจากเฉพาะชูเปอร์คลาสที่มีความสัมพันธ์เชื่อมโยงกับชั้บคลาสนั้น ๆ เท่านั้น

ยังความแตกต่างระหว่างแคทेगอรีกับชั้บคลาสร่วม ที่สำคัญคือกรณีชั้บคลาสร่วมนั้น เอนทิตี้ที่อยู่ที่ชั้บคลาสคือเอนทิตี้ที่ได้มาจากการอินเตอร์เซกชัน (intersection) กันระหว่างชูเปอร์คลาสของชั้บคลาสร่วมนั้น

ส่วนแคทेगอรีนั้น ถ้านำเอนทิตี้จากชูเปอร์คลาスマินเตอร์เซกชันกันจะได้ เชตว่าง เพราะชูเปอร์คลาสเหล่านั้นไม่มีความเกี่ยวข้องกัน ไม่มีเอนทิตี้ร่วมกัน การรับทอดจึงต้องรับทอดจากชูเปอร์คลาสที่ชั้บคลาสนั้นเป็นสมาชิก

ดังนั้นการรับทอดคุณสมบัติของชั้บคลาสประเภทแคทेगอรีจึงจำเป็นต้องใช้ วิริยเนียน (union) เพื่อรวมแอ็ตทริบิวต์และรีเลชันชิปไปทั้งหมดของชูเปอร์คลาส เมื่อเขียนแผนภาพอีอาร์ ให้แสดงการยูเนียนด้วยตัว U ในวงกลม

จากรูปที่ 36 พบว่าชั้บคลาส ACCOUNTHOLDER เป็นประเภทแคทดอกอธิ มีชูเปอร์คลาส PERSON และ COMPANY ที่เป็นเอ็นทิตี้ไทป์ที่ไม่เกี่ยวกัน ไม่มีแอตทริบิวต์ร่วมกัน ดังนั้นการรับ托ดจึงต้องยูเนียนคุณสมบัติของชูเปอร์คลาสทั้งสองเข้ามาด้วยกัน



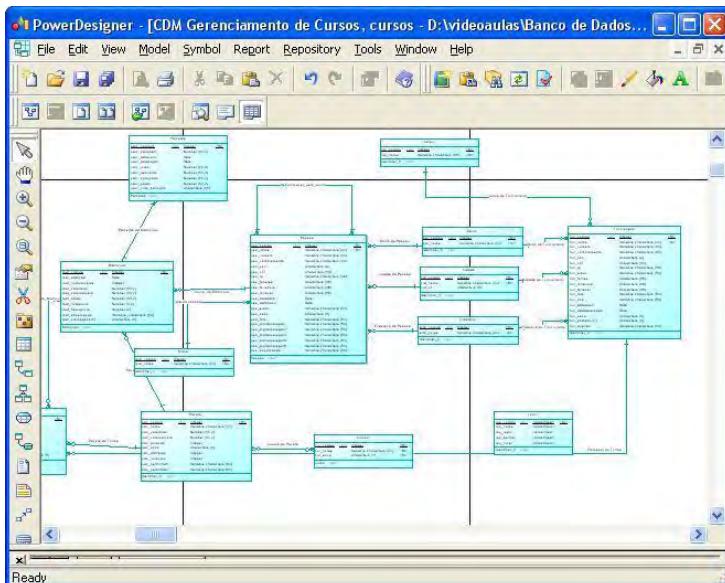
รูปที่ 36 ชั้บคลาสประเภทแคทดอกอธิและ การยูเนียน

อย่างไรก็ได้ ความสัมพันธ์ระหว่างแคทดอกอธิกับชูเปอร์คลาสนั้นเป็นได้ทั้งแบบบังคับหรือไม่บังคับ หากเป็นแบบไม่บังคับ ก็จะจำลองได้ด้วยการยูเนียนเท่านั้น

หากเป็นแบบบังคับ นั่นหมายความว่าชั้บคลาสจะต้องรับ托ดคุณสมบัติทั้งหมดของทุกชูเปอร์คลาส ซึ่งอาจจำลองด้วยกระบวนการสเปเชียลไลเซชัน ธรรมดาก็ได้ จากตัวอย่าง ก็จะเปลี่ยนเป็นให้ ACCOUNTHOLDER เป็นชูเปอร์คลาสที่มีชั้บคลาสคือ PERSON และ COMPANY

### 3.5 เครื่องมือสร้างแบบจำลองข้อมูล

มีเครื่องมือสร้างแบบจำลองข้อมูล (modeling tool) ที่ได้รับความนิยม  
เป็นจำนวนมากที่รองรับแบบจำลองระดับแนวคิด (conceptual modeling)  
และสามารถแปลงเป็นที่ติดต่อเข้ากันได้ เช่น UML หรือ BPMN ฯลฯ  
และสามารถใช้ในการออกแบบระบบ ตัวอย่างเช่น สถาปัตยกรรม  
ระบบสารสนเทศ (Information System Architecture) ที่ใช้ UML ในการ  
แสดงโครงสร้างของระบบ หรือ สถาปัตยกรรมการจัดการฐานข้อมูล (Database Architecture) ที่ใช้ BPMN ในการ  
แสดงวิธีการดำเนินการของฐานข้อมูล ฯลฯ



รูปที่ 37 หน้าจอของ PowerDesigner

ตารางที่ 1 แสดงตัวอย่างเครื่องมือที่เป็นที่นิยมในปัจจุบัน [31] พร้อมแพลตฟอร์มและระบบปฏิบัติการ (operating system) ที่รองรับ มีทั้งที่ให้บริการฟรี จนถึงเครื่องมือราคาแพงที่มีค่าความสามารถสูงตามไปด้วย

ตารางที่ 1 เครื่องมือแบบจำลองฐานข้อมูล

ชื่อเครื่องมือ	บริษัท	แพลตฟอร์มที่รองรับ	OS ที่รองรับ
Oracle SQL Developer Data Modeler	Oracle	Oracle, MS SQL Server, IBM DB2	Cross-platform
Navicat Data Modeler	PremiumSoft	MySQL, MS SQL Server, PostgreSQL, Oracle, SQLite	Windows, macOS, Linux
Database Deployment Manager	The Unauthorized Frog project	CUBRID, MySQL, SQLite	Windows, Linux
Software Ideas Modeler	Dusan Rodina	MS SQL Server, MySQL	Windows
Open ModelSphere	Grandite	MS SQL Server, MySQL, PostgreSQL, Oracle, DB2	Windows, macOS, Linux

### 3.6 บทสรุปและเรื่องชวนคิด

บทนี้เราได้เรียนรู้เรื่องการจำลองข้อมูลโดยเฉพาะสำหรับฐานข้อมูลเชิงสัมพันธ์แบบจำลองและแผนภาพที่เป็นเครื่องมือสำคัญ คือ แผนภาพอีอาร์และอีอาร์ อย่างไรก็ดี ด้วยเทคโนโลยีที่ก้าวหน้าอย่างรวดเร็ว ก็มีผู้เสนอแบบจำลองและแผนภาพเพิ่มมาอีกมากmanyดังที่กล่าวเป็นตัวอย่างก่อนหน้านี้ ที่กำหนดให้สัญลักษณ์ที่ต่างกันเพื่อสื่อความหมายคล้ายกัน ด้วยความรู้พื้นฐานที่ได้จากการเรียนบทนี้ หวังว่าผู้เรียนจะสามารถประยุกต์ความรู้นี้เพื่ออ่านแผนภาพอื่นที่ใช้ออกแบบระบบฐานข้อมูลได้ และอย่างให้คิดพัฒนาแบบที่เป็นมาตรฐานให้ดียิ่งขึ้นไป หรือคิดว่าในอนาคตเรายังจำเป็นต้องใช้แผนภาพเพื่อออกรูปแบบฐานข้อมูลอีกรึไม่ ถ้าเครื่องมือการออกแบบแบบมีประสิทธิภาพสูงกว่านี้ จะเป็นไปได้ไหมว่าไม่จำเป็นต้องมีคนออกแบบระบบฐานข้อมูลอีกต่อไป

## แบบฝึกหัดท้ายบท

1. จงอธิบายความแตกต่างระหว่างเอนทิตี้และเอนทิตี้ไฟป์
2. จงอธิบายความแตกต่างระหว่างเส้นแสดงความสัมพันธ์เชื่อมระหว่างเอนทิตี้ที่เป็นเส้นคู่กับเส้นเดี่ยว
3. จงอธิบายความสัมพันธ์แบบเวียนเกิด
4. จงอธิบายความแตกต่างระหว่างแบบจำลองอีอาร์กับแบบจำลอง อีอีอาร์
5. แบบจำลองอีอีอาร์มีการเพิ่มความสามารถได้บ้างขึ้นมา เพื่อที่จะรองรับแนวคิดเชิงวัตถุให้เข้ากับฐานข้อมูล
6. จงอธิบายรีเลชันของเจเนอเรลไลเซชัน และ สเปเชียลไลเซชัน ในการทำงานแบบล่างขึ้นบน (bottom-up) และบนลงล่าง (top-down)
7. แบบจำลองข้อมูลหมายถึงอะไร จงยกตัวอย่างแบบจำลองที่มักใช้กันอย่างแพร่หลายมากอย่างน้อย 3 แบบ
8. จงบอกประโยชน์ของแบบจำลองข้อมูล
9. จงอธิบายความแตกต่างระหว่างแบบจำลองข้อมูลเชิงแนวคิดกับแบบจำลองข้อมูลเชิงกายภาพ
10. แบบจำลองอีอาร์ใช้อธิบายสิ่งใดของฐานข้อมูล และมีองค์ประกอบสำคัญอะไรบ้าง

## บทที่ 4

### ฐานข้อมูลเชิงสัมพันธ์

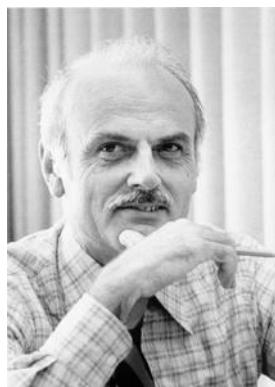
#### วัตถุประสงค์

- เพื่อให้ทราบแนวคิดของรูปแบบฐานข้อมูลเชิงสัมพันธ์
- เพื่อให้รู้จักองค์ประกอบของรีเลชัน
- เพื่อให้เข้าใจสัญกรณ์คอมพิวเตอร์ที่ใช้ในทฤษฎีรีเลชัน
- เพื่อให้ทราบเงื่อนไขบังคับของฐานข้อมูลเชิงสัมพันธ์
- สามารถสร้างข้อกำหนดการใช้งานในการแทรกข้อมูล การลบ  
ข้อมูลและการเปลี่ยนแปลงข้อมูล
- เพื่อให้ทราบสิ่งที่ต้องพิจารณาเมื่อทำการปรับปรุงข้อมูลในรีเลชัน

แนวคิดฐานข้อมูลเชิงสัมพันธ์ (relational database) [32-40] นี้ลูกคิดค้นโดย ดร. เอด加ร์ เอฟ คอตต์ (Dr. Edgar F. Codd) [41] ชาวอังกฤษ เมื่อ ค.ศ. 1970 ขณะที่ทำงานให้หน่วยวิจัยหน่วยหนึ่งของบริษัทไอบีเอ็ม นวัตกรรมครั้งนี้เป็นก้าวกระโดดในวงการคอมพิวเตอร์ที่สำคัญยิ่ง เป็นผลให้ ดร. คอตต์ได้รับรางวัลทั่วโลก (Turing award) [42] เมื่อ ค.ศ. 1981 และยังได้รับการพัฒนามาเรื่อยๆ หลักการของฐานข้อมูลเชิงสัมพันธ์อาศัยหลักการจากคณิตศาสตร์เรื่องเซต (mathematical concept of set) เป็นพื้นฐานและนำมายัง ทฤษฎีเรียลเซ็ต (theory of relation)

หัวใจของฐานข้อมูลเชิงสัมพันธ์ คือ การมองข้อมูลในรูปแบบของตาราง หรือเรียกอย่างเป็นทางการในศาสตร์ด้านฐานข้อมูลว่า รีเลชัน ซึ่งสื่อถึงความสัมพันธ์ของข้อมูลทั้งหลาย

ตำราบทนี้จึงเป็นบทที่สำคัญมาก ผู้เรียนจะต้องเข้าใจรีเลชันและข้อบังคับกฎหมายต่างๆ อย่างแม่นยำ เพื่อนำไปประยุกต์ใช้ได้ต่อไป



รูปที่ 38 ดร. เอด加ร์ เอฟ คอตต์

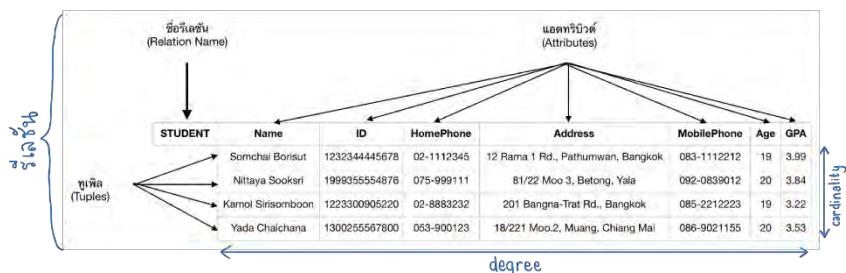
## 4.1 รีเลชัน

หัวใจของฐานข้อมูลเชิงสัมพันธ์คือการรวมฐานข้อมูลในรูปแบบของ ตาราง (table) สองมิติของข้อมูลจำนวนมากที่มีความสัมพันธ์กัน เราเรียกตารางเหล่านี้ว่า รีเลชัน (relation)

รีเลชันมีลักษณะเป็นตารางที่ประกอบด้วย แถว (row) ของข้อมูลมาเรียงต่อ กันเป็นรีเลชัน หรืออาจมองว่าประกอบด้วย คอลัมน์ (column) ของข้อมูลที่ นำมาเรียงต่อกันก็จะได้รีเลชันเข่นกัน

บางคำเรียกแถวข้อมูลว่า เรคคอร์ด (record) และเรียกคอมลัมน์ข้อมูลว่า ฟิลด์ (field)

ข้อมูลในแต่ละแถวต้องไม่ซ้ำกัน อาจกำหนดรหัสให้ข้อมูลในคอลัมน์ใด คอลัมน์หนึ่งแตกต่างกันทุกแถว รหัสที่ว่าไน้มักถูกใช้ในการอ้างถึงข้อมูลใน รีเลชัน โดยคอลัมน์ที่ถูกอ้างถึงนั้นจะเป็นคอลัมน์ที่สำคัญ เราเรียกชื่อคอลัมน์ เหล่านี้ว่า แอตทริบิวต์ (attribute) ดังแสดงในรูปที่ 39



รูปที่ 39 องค์ประกอบของรีเลชัน

นิยามทางคณิตศาสตร์ของรีเลชัน มีรูปแบบดังนี้

$$R(A_1, A_2, \dots, A_n)$$

R คือ ชื่อรีเลชัน

$A_i$  คือ ชื่อแอตทริบิวต์ตัวที่ i โดยที่ i มีค่าตั้งแต่ 1 ถึง n

แต่ละแอตทริบิวต์จะมีโดเมน (domain) กำกับ เช่นว่า  $\text{Dom}(A_i)$

ตัวอย่าง

STUDENT(Name, ID, HomePhone, Address, MobilePhone,  
Age, GPA)

หมายถึง รีเลชันชื่อ STUDENT ประกอบด้วย 7 แอตทริบิวต์ ได้แก่ Name, ID, HomePhone, Address, MobilePhone, Age และ GPA โดยทุกแอตทริบิวต์มีโดเมนกำกับ เช่น กำหนดให้แอตทริบิวต์ ID มีโดเมนเป็นข้อความ (text) ความยาว 13 ตัวอักษร (digit) เป็นต้น ตัวอย่างข้อมูลแสดงในรูปที่ 40 รีเลชัน STUDENT

STUDENT	Name	ID	HomePhone	Address	MobilePhone	Age	GPA
Somchai Borisut	123234445678	02-1112345	12 Rama 1 Rd., Pathumwan, Bangkok	083-1112212	19	3.99	
Nittaya Sookpri	1999355554876	075-999111	81/22 Moo 3, Betong, Yala	092-0839012	20	3.84	
Kamol Sirisomborn	1223300905220	02-8883232	201 Bangna-Trat Rd., Bangkok	085-2212223	19	3.22	
Yada Chaichana	1300255567800	053-900123	18/221 Moo 2, Muang, Chiang Mai	086-9021155	20	3.53	

รูปที่ 40 รีเลชัน STUDENT

นอกจากคำว่า “รีเลชัน” แล้ว จะเห็นได้ว่าสัพท์ที่เกี่ยวข้องมี ทูเพิล แอ็ตทริบิวต์ และโดเมน จึงขออธิบายละเอียดดังนี้

#### 4.1.1 ทูเพิล

เราเรียกแຄวของข้อมูลว่า ทูเพิล (tuple) ทูเพิลในรีเลชันเดียวกันต้องมี จำนวนคอลัมน์เท่ากัน และแต่ละคอลัมน์ต้องมีลักษณะข้อมูลเดียวกัน เช่น

< “Somchai Borisud”, “1232344445678”, “021112345”, “12 Rama 1 Rd., Pathumwan, Bangkok”, “0831112212”, 19, 3.99 >

< “Nittaya Sooksri”, “1999355554876”, “075999111”, “81/22 Moo 3, Betong, Yala”, “0920839012”, 20, 3.84 >

< “Kamol Sirisomboon”, “1223300905220”, “028883232”, “201 Bangna-Trat Rd., Bangkok”, “0852212223”, 19, 3.22 >

< “Yada Chaichana”, “1300255567800”, “053900123”, “18/221 Moo 2, Muang, Chiang Mai”, “0869021155”, 20, 3.53 >

ทั้ง 4 ทูเพิลนี้มี 7 คอลัมน์ ที่ลักษณะข้อมูลที่เรียงต่อกันนั้นเหมือนกันทั้ง 4 ทูเพิล ซึ่งสอดคล้องกับรีเลชันตัวอย่าง STUDENT(Name, ID, HomePhone, Address, MobilePhone, Age, GPA) นั้นเอง

#### 4.1.2 แอตทริบิวต์

แอตทริบิวต์ (attribute) ก็คือคอลัมน์ในตาราง จากตัวอย่างก่อนหน้า STUDENT(Name, ID, HomePhone, Address, MobilePhone, Age, GPA) นั้น มีอยู่ 7 แอตทริบิวต์ คือ Name, ID, HomePhone, Address, MobilePhone, Age, และ GPA

#### 4.1.3 โดเมน

โดเมน (domain) คือ ลักษณะข้อมูล (data type) ที่กำหนดให้แต่ละ แอตทริบิวต์ โดเมนดังกล่าวอาจเป็นได้ทั้ง integer, real, character, boolean, date, time, time-stamp หรืออื่นใดที่ใช้ในคอมพิวเตอร์ เช่น กำหนดให้ Name มีโดเมน 50 character อกแบบมาเพื่อเก็บชื่อของ Student

การกำหนดโดเมนนั้น ให้กำหนดตามความหมายและวัตถุประสงค์ที่เราจะใช้ แอตทริบิวต์นั้น ๆ เช่น เลขประจำตัวประชาชน ประกอบด้วยตัวเลข 13 หลัก ถ้าไม่พิจารณาให้ศักดิ์จากกำหนดโดเมนเป็น เลขจำนวนเต็ม (integer) แต่ด้วย ว่าเราไม่ได้จะนำเลขนี้ไปคำนวณและยอมให้หลักแรก ๆ เป็นศูนย์ได้ ดังนั้น วิธีที่ถูกต้องควรจะเลือกใช้โดเมนอักขระ (character) เป็นต้น

นอกเหนือจากนั้น เรา yang สามารถกำหนดรูปแบบหรือหน้าตาการปรากฏของ แอตทริบิวต์ได้ เช่น หมายเลขโทรศัพท์มือถือในประเทศไทย มีตัวเลข 10 ตัว ควรใช้ character ความยาว 10 และกำหนดให้แสดงรูปแบบเป็น xxx-xxx-xxxx ได้อีกด้วย แอตทริบิวต์ที่มักใช้รูปแบบกำกับอย่างนี้คือ date ที่

กำหนดให้แสดง xx-xx-xxxx ได้ ทั้งนี้ เมื่อกำหนดโดเมนและหน้าตาแล้ว ก็จะใช้โดเมนนี้ได้มากกว่าหนึ่งแอ็ตทริบิวต์ เช่น ให้ใช้โดเมน date กับ BirthDate, EntryDate, SubscriptionDate เพื่อให้ข้อมูลมีความสอดคล้องกันทั้งฐานข้อมูล

## 4.2 สัญกรณ์คณิตศาสตร์

เราสามารถเขียนสัญกรณ์คณิตศาสตร์เพื่ออธิบายรีเลชันได้ดังนี้

$$R(A_1, A_2, \dots, A_n)$$

R คือ スキมา (schema) ของรีเลชัน r มีตัวแปรที่เก็บ n ประกอบด้วยแอ็ตทริบิวต์  $A_1$  ถึง  $A_n$

r คือ สมาชิกของ R เมื่อเขียน  $r(R)$  หมายความว่า รีเลชัน r หนึ่งที่ใช้スキมา R

ดังนั้น r ประกอบด้วยเซตของ m ทุกเพลิ เขียนได้ว่า

$$r = \{t_1, t_2, \dots, t_n\}$$

แต่ละทุกเพลิ t ประกอบด้วย value v ที่เรียงกัน (ordered list)

$$t = \langle v_1, v_2, \dots, v_n \rangle$$

แต่ละ value  $v_i$  ต้องมีโดเมนตามที่กำหนดให้แอ็ตทริบิวต์  $A_i$  เขียนว่า  $\text{dom}(A_i)$  หรือค่าว่าง หรือ null

เราจะอ้างถึงค่าข้อมูลในทุกเพลิดด้วยการเขียน

$$t[A_i] = V_i \text{ หรือ } t.A_i = V_i$$

สรุปได้ว่า

- $R$  คือ ศักมีของรีเลชัน เรียกได้อีกอย่างว่า อินเทนชัน (intension) ของรีเลชัน
- $r(R)$  คือ สมาชิกของ  $R$  เรียกได้อีกอย่างว่า เอกซ์เทนชัน (extension) ของรีเลชัน

รีเลชันหนึ่ง ๆ เกิดจากผลการcarที่เขียนของเซต แต่ละเซตคือค่าจากโดเมน ซึ่งโดเมนจะถูกกำหนดจากผู้ออกแบบรีเลชันนั้น

ตัวอย่าง

Let  $S_1 = \{0,1\}$

Let  $S_2 = \{a,b,c\}$

Let  $R = S_1 \times S_2$

Then:

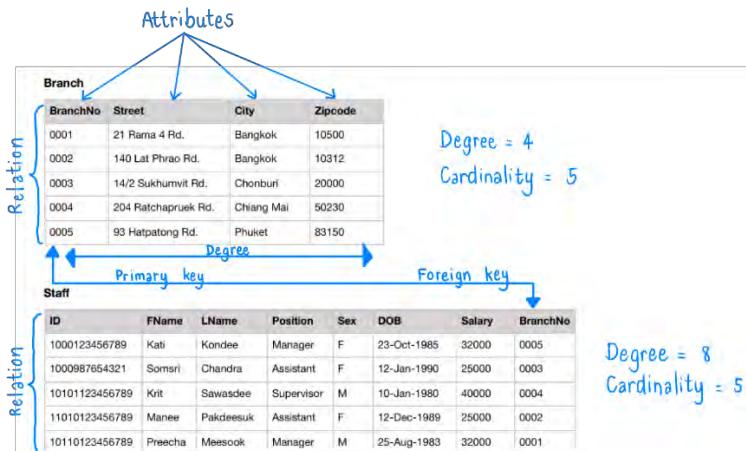
$$r(R) = \{<0,a>, <0,b>, <1,c>\}$$

เป็นเอกซ์เทนชันหรือสถานะ (state) หนึ่งที่เป็นไปได้ที่จะยืดหุ้น ประกอบด้วย 3 ทูเพิล

ตารางที่ 2 และรูปที่ 41 สรุปคัพท์ที่ใช้ในฐานข้อมูลเชิงสัมพันธ์

ตารางที่ 2 คัพท์ที่ใช้ในฐานข้อมูลเชิงสัมพันธ์

คัพท์แบบจำลองเชิงสัมพันธ์	คัพท์อื่นที่มีความหมายเดียวกัน
รีเลชัน (relation)	ตาราง (table)
ทูเพิล (tuple)	แถว (row) เรกคอร์ด (record)
แอตทริบิวต์ (attribute)	คอลัมน์ (column) หรือ พิลต์ (field)
โดเมน (domain)	ลักษณะข้อมูล (data type)
อินเทนชัน (intension) หรือ สคีมา (schema)	โครงสร้างตาราง (table definition)
เอกซ์เทนชัน (extension)	ตารางที่มีข้อมูล (populated table)
ดีกรี (degree)	จำนวนคอลัมน์หรือจำนวนพิลต์ (number of column/field)
cardinality (cardinality)	จำนวนแถวหรือจำนวนเรกคอร์ด (number of rows/record)



รูปที่ 41 ศัพท์ที่ใช้ในฐานข้อมูลเชิงรั้งพื้นที่

### 4.3 เงื่อนไขบังคับของฐานข้อมูลเชิงรั้งพื้นที่

ฐานข้อมูลเชิงรั้งพื้นที่มีเงื่อนไขบังคับ (constraint) ที่ทุกเริ่มหาดเดียวต้องปฏิบัติตาม ดังต่อไปนี้

1. เงื่อนไขโดเมน (domain constraint)
2. เงื่อนไขคีย์ (key constraint)
3. เงื่อนไขburณภาพเดอนทีตี (entity integrity constraint)
4. เงื่อนไขburณภาพอ้างอิง (referential integrity constraint)
5. เงื่อนไขอื่น ๆ

### 4.3.1 เงื่อนไขโดเมน

เงื่อนไขโดเมนระบุว่าแต่ละทุกเพิลจะประกอบด้วยค่าข้อมูลของแต่ละแอตทริบิวต์ต้องเป็นค่าของatomictype (atomic) คือไม่สามารถย่ออย่างได้อีก และต้องอยู่ในรูปแบบที่โดเมนนั้น ๆ กำหนด (valid)

### 4.3.2 เงื่อนไขคีย์

เงื่อนไขคีย์ กล่าวว่าต้องไม่มีทุกเพิลใดในรีเลชันที่มีค่าทั้งหมดซ้ำกัน ตั้งนั้นรีเลชันจึงควรจะมีคีย์ (key) ซึ่งคือแอตทริบิวต์ชุดหนึ่งซึ่งเป็นชับเชตของแอตทริบิวต์ทั้งหมดที่ทำให้ทุกเพิลใด ๆ ไม่ซ้ำกัน เราเรียกแอตทริบิวต์ชุดนี้ว่าชูเปอร์คีย์

#### 4.3.2.1 ชูเปอร์คีย์

ชูเปอร์คีย์ (superkey) ย่อว่า SK มีคุณสมบัติทำให้ทุกเพิลในรีเลชันไม่ซ้ำกัน

$$t_1.SK \neq t_2.SK$$

หมายความว่าชูเปอร์คีย์ของทุกเพิลที่หนึ่งต้องไม่เท่ากับชูเปอร์คีย์ของทุกเพิลที่สอง

แต่ละรีเลชันมีชูเปอร์คีย์เสมอ ค่าบริยาย (default) ชูเปอร์คีย์ คือ เชตของแอตทริบิวต์ทุกตัวของรีเลชัน

หากชูเปอร์คีย์นั้นประกอบด้วยเซตของแอตทริบิวต์จำนวนน้อยที่สุดที่ทำให้ทุกทุกเพิลไม่ซ้ำ ก็จะเรียกเซตของแอตทริบิวต์นั้นว่า คีย์

#### 4.3.2.2 คีย์

คีย์ (Key) คือ ชุดเปอร์คีย์ที่เล็กที่สุดที่ยังทำให้รีเลชันได้ ๆ ไม่มีทูเพิลซ้ำ

ตัวอย่าง

Car (LicenseNumber, EngineSerialNumber, Make, Model, Year)

CAR	<u>LicenseNumber</u>	<u>EngineSerialNumber</u>	Make	Model	Year
ก 290 กทม.	A69352	Ford	Mustang	2016	
กข 2123 ราชบุรี	B43696	Toyota	Camry	2015	
ศ 1012 ล่าพุน	X83554	Honda	Civic	2016	
ศง 5199 สุรินทร์	C43742	Nissan	Almera	2017	
ข 9999 นครราชสีมา	Y82935	BMW	i8	2018	
พจ 4163 ฉะเชิงเทรา	U028365	Toyota	Altis	2018	

รูปที่ 42 รีเลชัน CAR

รีเลชัน CAR ดังแสดงในรูปที่ 42 มีคีย์ที่เป็นไปได้ 2 ตัวคือ LicenseNumber หมายถึงทะเบียนรถซึ่งประกอบด้วยจังหวัดกับเลขทะเบียน ซึ่งไม่มีรีลัคนไหนที่ทะเบียนรถซ้ำกัน อีกคีย์ที่เป็นไปได้คือ EngineSerialNo คือเลขตัวถังรถยนต์ ซึ่งไม่มีโอกาสซ้ำกัน กับกรณีนี้ ให้เลือกเซตไดเซตหนึ่งเป็นคีย์ก็ได้

จะเห็นว่า หากชุดเปอร์คีย์มีแอตทริบิวต์เดียวเป็นสมาชิก ชุดเปอร์คีย์นั้นก็จะเป็นคีย์แน่นอน แต่ถ้าชุดเปอร์คีย์มีแอตทริบิวต์มากกว่า 1 ตัว จะต้องตรวจสอบว่า สามารถลดแอตทริบิวต์ได้ออกแล้วทำให้ทูเพิลไม่ซ้ำหรือไม่

นอกจากนั้นถ้าในรีเลชันนั้นมีคีย์คู่เพิ่ง (candidate key) ต้องเลือกคีย์ที่เหมาะสม 1 ตัว ให้เป็นคีย์หลัก (primary key) และทำการจัดลำดับให้ต่อไปนี้  
แอตทริบิวต์ที่เป็นคีย์หลัก

ในกรณีที่มีคีย์ที่เป็นไปได้มากกว่า 1 คีย์ เช่น จากตัวอย่างคือ {LicenseNumber} หรือ {SerialNo} เราเรียกคีย์ทั้งสองนี้ว่า แคนดิเดตคีย์ (candidate key) หรือคีย์คู่แข่ง ในทางปฏิบัติให้เลือกชุดใดชุดหนึ่งก็ได้มา เป็นคีย์ และเรียกตัวที่เลือกว่า ไพรามารีคีย์ (primary key) หรือ คีย์หลัก ย่อว่า PK

### 4.3.3 เงื่อนไขบูรณาภาพเอนทิตี้

เงื่อนไขบูรณาภาพเอนทิตี้ (entity integrity constraint) กล่าวว่า การที่เอนทิตี้จะมีความบูรณาภาพได้นั้น คีย์ทุกตัวต้องไม่ใช่ค่าว่าง (null)

$$S = \{R_1, R_2, R_3, \dots, R_n\}$$

ให้ S เป็นชื่อร้านข้อมูล R เป็นสคีมาของรีเลชันทั้งหมดที่อยู่ใน S

แอตทริบิวต์ที่เป็นคีย์หลัก (primary key : PK) ทุกๆเพลในแต่ละรีเลชันของ ฐานข้อมูล S ต้องไม่ใช่ค่าว่าง

$$t.PK \neq \text{null}$$

#### 4.3.4 เงื่อนไขบูรณาภาพอ้างอิง

เราจะพิจารณาเงื่อนไขบูรณาภาพอ้างอิง (referential integrity) เฉพาะเมื่อมีรีเลชัน 2 รีเลชัน ที่มีความสัมพันธ์กันและต้องอ้างอิงถึงกัน เราเรียกรีเลชัน 2 รีเลชันนี้ว่า รีเลชันผู้อ้าง (referencing relation) กับรีเลชันรับอ้าง (referenced relation)

การอ้างอิงระหว่างกันต้องใช้คีย์ชนิดหนึ่ง เรียกว่า พอเรนคีย์ (foreign key) เ啻นย่อว่า FK ทั้งนี้ ถ้าแปลเป็นไทย อาจแปลตรงตัวได้ว่าคีย์ต่างชาติหรือคีย์แปลกลปлом ซึ่งไม่ตรงกับความหมายของพอเรนคีย์ในฐานข้อมูล คำนี้จึงขอเลือกทับศัพท์ว่า พอเรนคีย์ เ啻นย่อว่า FK

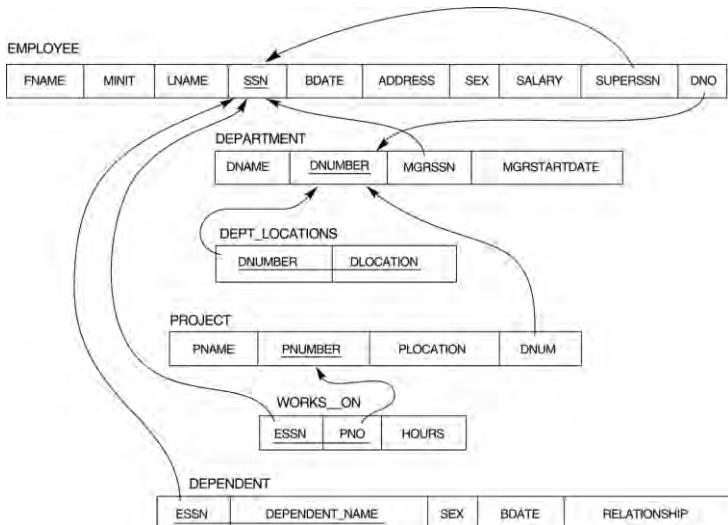
เงื่อนไขบูรณาภาพอ้างอิง กำหนดว่า ค่าจาก FK ของรีเลชันผู้อ้าง มีความเป็นไปได้ 2 อย่างคือ 1) ต้องมีค่าเท่ากับ PK ของรีเลชันรับอ้างซึ่งต้องมีค่านั้นอยู่แล้ว หรือ 2) เป็นค่าว่าง ในกรณีที่เป็นค่าว่าง FK ต้องไม่ใช่ส่วนหนึ่งของคีย์หลักของรีเลชันนั้น นั่นคือ

$$t_1.FK = t_2.PK \text{ หรือ}$$

$$t_1.FK = \text{null}$$

พิจารณารูปที่ 43 เส้นลูกศรแสดงความสัมพันธ์ระหว่างรีเลชันผู้อ้างไปยังรีเลชันรับอ้าง ดังนั้น แอตทริบิวต์ที่รีเลชันผู้อ้าง เรียกว่า พอเรนคีย์ FK ตัวอย่างจากรูปนี้มีจะเห็นว่ารีเลชัน PROJECT เมื่อเป็นรีเลชันผู้อ้าง จะมี DNUM เป็น FK อ้างอิงไปยัง DEPARTMENT ซึ่งมีหน้าที่เป็นรีเลชันรับอ้าง

ในกรณีนี้ค่าใน DNUM จะต้องมีปรากฏใน DNUMBER ที่ DEPARTMENT หรือเป็นค่าว่าง ซึ่งทำได้ เพราะ DNUM ไม่ใช่ PK ของ PROJECT



รูปที่ 43 รีเลชันบริษัทดีบีแอลจากอิอาร์

### 4.3.5 เเงื่อนไขบังคับอื่น ๆ

ยังมีเงื่อนไขบังคับที่สำคัญ เรียกว่า เงื่อนไขพังก์ชันนัลติเพนเดนซี (Functional dependency) หมายความว่า แอ็ตทริบิวต์เซตหนึ่งขึ้นต่อ แอ็ตทริบิวต์อีกเซตหนึ่ง หรืออธิบายง่าย ๆ ว่า เมื่อรู้แอ็ตทริบิวต์หนึ่งจะรู้อีก แอ็ตทริบิวต์หนึ่งแน่ ๆ เอียนได้ว่า

$$X \rightarrow Y$$

อ่านว่า X กำหนดฟังก์ชัน Y (X functionally determines Y) หรือ Y ขึ้นต่อ X (ฟังก์ชัน Y depends on X) อ่านอย่างย่อว่า X determines Y หรือ Y depends on X

รายละเอียดของเงื่อนไขจะอยู่ในบทที่ 6 ของตำรานี้

#### 4.4 การปรับปรุงข้อมูลในรีเลชัน

โดยธรรมชาติแล้วข้อมูลในฐานข้อมูลที่มีการใช้งานบ่อยมีการเปลี่ยนแปลงเสมอ เรามองว่าข้อมูลต่าง ๆ ได้รับการปรับปรุง ซึ่งการปรับปรุงมีได้ 3 กรณี คือ

- 1) การแทรก (insert) คือ การเพิ่มทูเพิลใหม่ลงในรีเลชัน
- 2) การลบ (delete) คือ การลบทูเพิลออกจากรีเลชัน
- 3) การแก้ไข (update หรือ modify) คือ การเปลี่ยนแปลงข้อมูลของทูเพิลในรีเลชัน

สิ่งที่ต้องคำนึงถึงเสมอคือ เมื่อใดที่ปรับปรุงข้อมูลในฐานข้อมูล จะต้องไม่ผิดเงื่อนไขบังคับใด ๆ ทั้งสิ้น มิฉะนั้นข้อมูลในฐานข้อมูลอาจผิดพลาดได้

##### 4.4.1 การแทรก

การแทรก (insert operation) คือการเพิ่มทูเพิลใหม่เข้าไปในรีเลชัน การแทรกข้อมูลมีความเสี่ยงต่อความผิดพลาดของข้อมูลที่ต้องระวัง คือ

- อาจกระทบเงื่อนไขโดเมน นั่นคือ ถ้าค่าใดในทุกเพลิที่จะเพิ่มเข้าไป ใหม่ไม่เป็นอะตอมมิก หรือไม่ตรงตามโดเมนของรีเลชันนั้น
- อาจกระทบเงื่อนไขคีย์ ถ้าค่าข้อมูลของคีย์ในทุกเพลิใหม่มีอยู่แล้วใน ทุกเพลิดเดิม
- อาจกระทบเงื่อนไขบูรณภาพเอนทิตี้ ถ้าค่าในคีย์หลักของทุกเพลิใหม่ เป็นค่าว่าง
- อาจกระทบเงื่อนไขบูรณภาพอ้างอิง ถ้าค่าฟอร์เรนคีย์ไม่ตรงกับ ค่าคีย์หลักของรีเลชันที่อ้างถึง

#### 4.4.2 การลบ

การลบ (delete operation) การลบข้อมูลมีความเสี่ยงต่อความผิดพลาด ของข้อมูลที่ต้องระวัง เฉพาะเงื่อนไขบูรณภาพอ้างอิงเท่านั้น นั่นคือ หาก ทุกเพลิถูกลบแล้วยังมีการอ้างอิงฟอร์เรนคีย์จากทุกเพลิอื่นในฐานข้อมูล จะไม่ สามารถลบข้อมูลได้ เหตุนี้สามารถแก้ไขได้ 4 วิธีคือ 1) ยกเลิกการลบ 2) ลบ ทุกเพลิมาอ้างทุกเพลิที่จะลบให้ครบ เรียกว่าแคนสเคด (cascade) 3) แก้ค่า แอตทริบิวต์ให้เป็นค่าว่างหรือเปลี่ยนเป็นค่าอื่นที่ไม่กระทบ หรือ 4) เขียน โปรแกรมจัดการการลบนั้นโดยเฉพาะ

#### 4.4.3 การแก้ไข

การแก้ไข (update operation) คือการปรับข้อมูลตามความต้องการ แต่การ แก้ไขข้อมูลมีความเสี่ยงต่อความผิดพลาดของข้อมูลที่ต้องระวังเหมือนกับ แทรก คือทั้ง 4 กรณี ได้แก่ เงื่อนไขโดเมน เงื่อนไขคีย์ เงื่อนไขบูรณภาพ เ昂ทิตี้ และเงื่อนไขบูรณภาพอ้างอิง

## 4.5 บทสรุปและเรื่องชวนคิด

บทนี้เราได้เรียนรู้แนวคิดและหลักการของฐานข้อมูลเชิงสัมพันธ์ ได้เข้าใจศัพท์ที่ใช้ในการฐานข้อมูลให้ในกราฟแบบฐานข้อมูลเชิงสัมพันธ์ ตลอดจนเงื่อนไขที่ต้องปฏิบัติเพื่อลดความผิดพลาดของข้อมูลให้ได้มากที่สุด ถึงแม้ในการทำงานอาจไม่ได้มีโอกาสได้คิดคำนวณเงื่อนไขต่าง ๆ ที่ระบุในบทนี้ แต่ วิศวกรคอมพิวเตอร์ทุกคนต้องเข้าใจที่มา เพื่อให้นำไปประยุกต์ใช้ได้อย่างถูกต้อง และทำให้มีโอกาสหายใจใหม่ ๆ ในอนาคตได้ สำหรับบทนี้ ผู้เรียนคงเข้าใจแล้วว่าหลักการของรีเลชันน์อาศัยคณิตศาสตร์เรื่องเซต แล้วฐานข้อมูลใหม่ ๆ ตระกูลโนเนสคิวแอล จะใช้หลักการคณิตศาสตร์ได้เป็นพื้นฐานได้บ้าง

## แบบฝึกหัดท้ายบท

1. หลักการของฐานข้อมูลเชิงสัมพันธ์อาศัยหลักการจากการคณิตศาสตร์เรื่องได
2. สิ่งใดถือเป็นสิ่งที่สำคัญที่สุดหรือเป็นหัวใจของฐานข้อมูลเชิงสัมพันธ์
3. การกำหนดโดเมนให้แต่ละแอ็ตทริบิวต์ ควรพิจารณาจากสิ่งใด
4. จงอธิบายความแตกต่างระหว่างชูเบอร์คีย์และคีย์
5. จงบอกเงื่อนไขที่ทำให้ฟอร์มารคีีย์สามารถเป็นค่าว่างได้
6. “รีเลชันใด ๆ สามารถมีคีย์ได้เพียงตัวเดียวเท่านั้น” เป็นคำกล่าวที่ถูกต้องหรือไม่ จงอธิบาย
7. ฟอร์มารคី เป็นคำที่ใช้เรียกแอ็ตทริบิวต์ในรีเลชันผู้อ้าง หรือรีเลชันผู้รับ อ้าง
8. การลบข้อมูลออกจากฐานข้อมูลนั้น หากทุกเพลิที่จะลบติดเงื่อนไข บูรณาภาพอ้างอิง กล่างคือ มีการอ้างอิงฟอร์มารคីจากทุกเพลิอื่นในฐานข้อมูล many-to-one ดังกล่าว สามารถแก้ไขได้กี่วิธี อย่างไรบ้าง
9. รีเลชัน Car\_Rent ประกอบด้วยแอ็ตทริบิวต์ Gear, Seat และ Fuel จะ หาจำนวนทุกเพลิสูงสุดของรีเลชัน Car\_Rent ถ้า โดเมน(Gear) = {Manual, Auto} โดเมน (Seat) = {5, 7, 12} โดเมน (Fuel) = {Diesel, Benzine, LPG, NGV}
10. การปรับปรุงข้อมูลในรีเลชัน ไม่ว่าจะเป็นการแทรก การลบ การแก้ไข จะต้องพิจารณาถึงสิ่งใดเป็นสำคัญ

11. จงพิจารณาเรื่องลักษันนี้ แล้วทำให้อยู่ในรูปแบบการเขียนโดยของรีเลชัน

Student(SSN, Name, Major, Bdate)

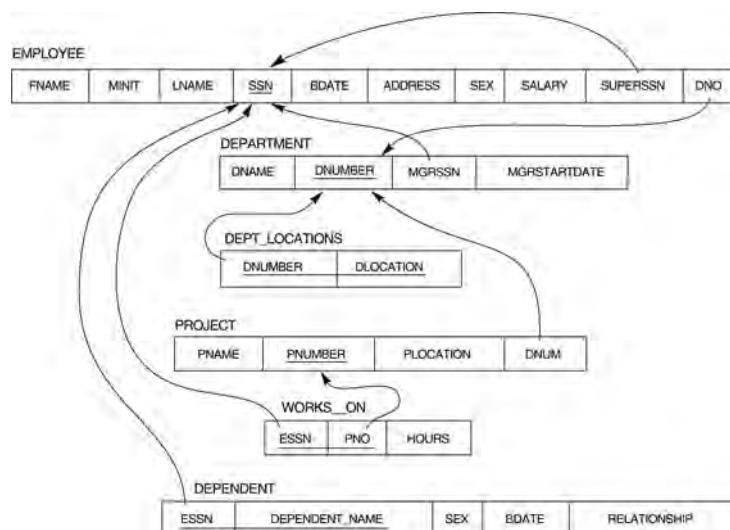
Course(Course#, Cname, Dept)

Enroll(SSN, Course#, Quarter, Grade)

Book\_Adoption(Course#, Quarter, Book\_ISBN)

Text(Book\_ISBN, Book\_title, Publisher, Author)

12. จงพิจารณาฐานข้อมูล Company และตรีบัวต์ได้ในรีเลชัน  
WORKS\_ON ไม่สามารถมีค่าเป็น NULL ได้ เพราะเหตุใด



## บทที่ 5

### แบบจำลองกับฐานข้อมูลเชิงสัมพันธ์

#### วัตถุประสงค์

- เพื่อให้เข้าใจกระบวนการออกแบบฐานข้อมูลจากระดับแนวคิดเป็นระดับตรรกะ
- เพื่อให้สามารถแปลงแผนภาพอีอาร์เป็นรีเลชันได้
- เพื่อให้สามารถแปลงแผนภาพอีอาร์เป็นรีเลชันได้

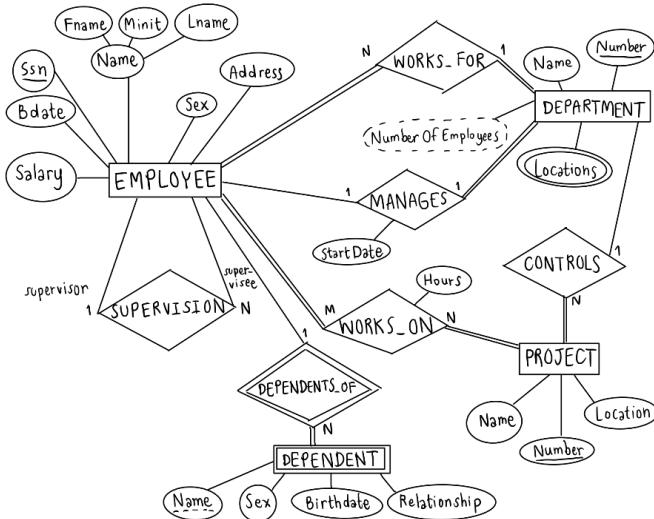
ที่ผ่านมาเราได้เรียนรู้ของการออกแบบฐานข้อมูลด้วยแบบจำลองอีอาร์และใช้แผนภาพอีอาร์เพื่อสื่อสารกับผู้ใช้ ส่วนนี้คือแบบจำลองระดับแนวคิด (conceptual level)

จากนั้นเราได้เรียนรู้เรื่องรีเลชัน คือวิธีการทำงานของฐานข้อมูลเชิงสัมพันธ์ ซึ่งเป็นแบบจำลองระดับตรรกะ (logical level)

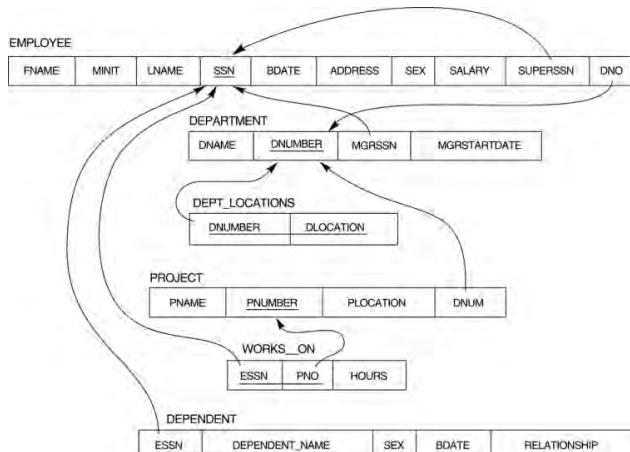
ในบทนี้ เราจะแปลงแบบจากอีอาร์หรืออีอาร์ให้อยู่ในรูปแบบของรีเลชันที่ฐานข้อมูลและคอมพิวเตอร์นำไปประมวลผลได้ [3, 13]

## 5.1 ขั้นตอนการแปลงอีอาร์และอีอาร์เป็นรีเลชัน

ขั้นตอนการแปลงอีอาร์มี 7 ขั้นตอน และการแปลงอีอาร์มีเพิ่มอีก 2 ขั้นตอน รูปที่ 45 แสดงรีเลชันผลลัพธ์ที่ได้จากการแปลงอีอาร์ระบบฐานข้อมูลของบริษัทดีบีจากรูปที่ 44 ตัวอย่างและขั้นตอนในบทนี้อ้างอิงมาจาก [2] การแปลงมีรายละเอียดดังต่อไปนี้



รูปที่ 44 แผนภาพอีอาร์แสดงระบบฐานข้อมูลของบริษัทดีบี



รูปที่ 45 รีเลชันแสดงระบบฐานข้อมูลของบริษัทดีบี

### 5.1.1 ขั้นตอนที่ 1 : แปลงเอนทิตี

แปลงเอนทิตีไทยจากอีกอาร์ชิวิลให้เป็นรีเลชัน ในขั้นนี้ เราสนใจเฉพาะเอนทิตีไทย ปกติ จะยังไม่รวมถึงเอนทิตีแบบบวิค ขอเรียกเอนทิตีไทยปกตินี้ว่าเอนทิตีไทย

การแปลงขั้นนี้ ให้สร้างรีเลชันใหม่ให้กับเอนทิตีไทยทุกตัว หนึ่งรีเลชันต่อหนึ่งเอนทิตีไทย จำนวนให้นำเอtotทริบิวต์ที่อยู่ในแต่ละเอนทิตีมาเป็นเอtotทริบิวต์ของรีเลชันใหม่นั้น แล้วเลือกเอtotทริบิวต์ที่เป็นคีย์ของเอนทิตี ให้เป็นคีย์หลัก (primary key) ของรีเลชันนั้น

ดังนั้น จากขั้นตอนที่ 1 เราจะได้รีเลชัน EMPLOYEE, DEPARTMENT และ PROJECT โดยมี SSN, DNumber และ PName เป็นคีย์หลักของรีเลชัน ตามลำดับ ดังแสดงในรูปที่ 46

EMPLOYEE

<u>SSN</u>	Fname	Minit	Lname	Bdate	Sex	Address	Salary
------------	-------	-------	-------	-------	-----	---------	--------

DEPARTMENT

Dname	<u>Dnumber</u>	DLocations
-------	----------------	------------

PROJECT

Pname	<u>Pnumber</u>	PLocation
-------	----------------	-----------

รูปที่ 46 แปลงเอนทิตีไทย

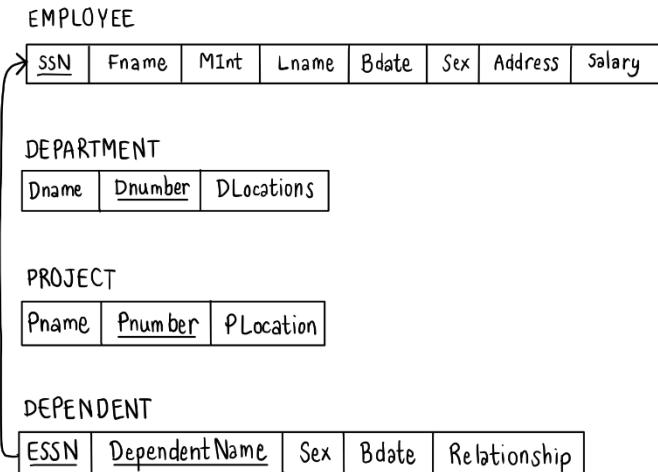
### 5.1.2 ขั้นตอนที่ 2 : แปลงวีคเอนทิตี้

แปลงวีคเอนทิตี้ให้เป็นรีเลชัน โดยการสร้างรีเลชันใหม่ แล้วนำแอ็ตทริบิวต์ทั้งหมดที่อยู่ในวีคเอนทิตี้ให้มาเป็นแอ็ตทริบิวต์ของรีเลชันใหม่ให้ครบ

จากนั้นให้สร้างความสัมพันธ์ระหว่างรีเลชันใหม่กับรีเลชันที่เป็นเจ้าของ วีคเอนทิตี้ กำหนดบทบาทของรีเลชันใหม่ว่าเป็น รีเลชันอ้างถึง และรีเลชันเจ้าของคือ รีเลชันรับอ้าง

จากนั้น ให้นำ PK จากรีเลชันรับอ้างมาเพิ่มเป็น PK ร่วมกับคีย์เดิมที่เป็นคีย์บางส่วน (partial key) และกำหนดให้ PK นั้นเป็นฟอร์เอนคีย์ (foreign key : FK) จากรีเลชันอ้างถึง ซึ่งไปยังคีย์หลัก (primary key : PK) ของรีเลชันรับอ้าง

จากตัวอย่าง ให้สร้างรีเลชันชื่อ DEPENDENT นำ SSN จาก EMPLOYEE มาเป็นคีย์ร่วมกับ DEPENDENT\_NAME เปลี่ยนชื่อ SSN เป็น ESSN เพื่อไม่ให้ซ้ำ ดังนั้น PK ของรีเลชัน DEPENDENT คือ {ESSN, DEPENDENT\_NAME} และกำหนดให้ ESSN เป็น FK ซึ่งไปยัง SSN ในรีเลชัน EMPLOYEE ดังแสดงในรูปที่ 47



### รูปที่ 47 แปลงวีคเอนทิตี้ไป

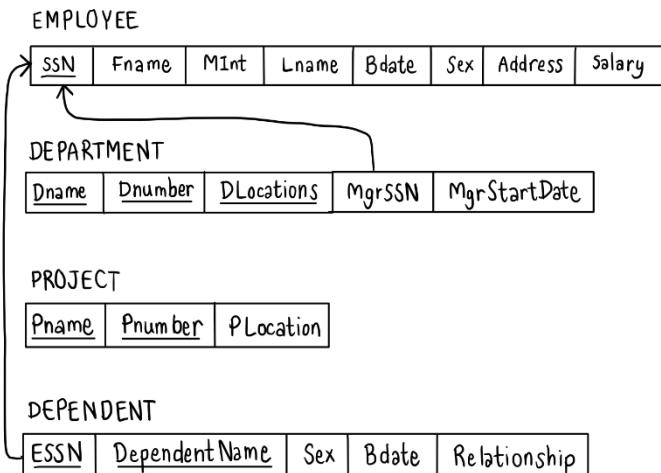
### 5.1.3 ขั้นตอนที่ 3 : แปลงรีเลชันชิปไปทป 1:1

รีเลชันชิปใน ER แบบ 1:1 สามารถแปลงเป็นรีเลชันได้ 3 วิธี

## วิธีที่ 1 ใช้ FK

เหมาจะกับรีเลชันซิปเป็นแบบใบหน้าวี ที่มีเงื่อนไขบังคับเข้าร่วมแบบบังคับ (total participation) ข้างหนึ่ง และแบบไม่บังคับ (partial participation) อีกข้างหนึ่ง ให้กำหนดรีเลชันผู้ที่บังคับเป็นหลัก แล้วนำ PK จากรีเลชันอีกผู้หนึ่งมาเพิ่มเป็นแอตทริบิวต์ใหม่เข้าไปในรีเลชัน จากนั้นให้กำหนดแอตทริบิวตนี้เป็น FK ซึ่งลับไปยังรีเลชันอีกผู้ ถ้ารีเลชันซิปนั้นมีแอตทริบิวต์ของตัวเอง ให้ร่วมแอตทริบิวตนั้นมาในรีเลชันผู้หลักด้วย

จากตัวอย่าง พบว่า รีเลชันชิป MANAGES เป็นแบบ 1:1 และมีเงื่อนไขบังคับเข้าร่วมแบบบังคับกับ DEPARTMENT และแบบไม่บังคับกับ EMPLOYEE ดังนั้น เราจึงยึด DEPARTMENT เป็นหลัก แล้วนำ PK ของ EMPLOYEE คือ SSN มาใส่เพิ่มใน DEPARTMENT เปลี่ยนชื่อจาก SSN เป็น MgrSSN เพื่อให้ไม่ซ้ำ แล้วย้าย FK MgrSSN จาก DEPARTMENT ไปยัง SSN ใน EMPLOYEE สุดท้าย ให้รวมแอ็ตทริบิวต์ MgrStartDate ไว้ในรีเลชัน DEPARTMENT ด้วยดังแสดงในรูปที่ 48



รูปที่ 48 แปลงรีเลชันชิปเป็น 1:1

## วิธีที่ 2 ผนวกรีเลชัน

วิธีนี้หมายความว่ากับรีเลชันชิปที่เงื่อนไขบังคับเข้าร่วมเป็นแบบบังคับ (total participation) ทั้งสองข้างของรีเลชันชิป กรณีที่หน้าแอ็ตทริบิวต์จากรีเลชัน

ทั้งสองมาร่วมกันทั้งหมด และหากว่ารีเลชันซึ่งเป็นนี้แอ็ตทริบิวต์ให้รวมไปด้วยกำหนดคีย์ของรีเลชันใหม่นี้ คือคีย์จากรีเลชันทั้งสองที่ผนวกกัน จะเกิดคีย์ร่วม

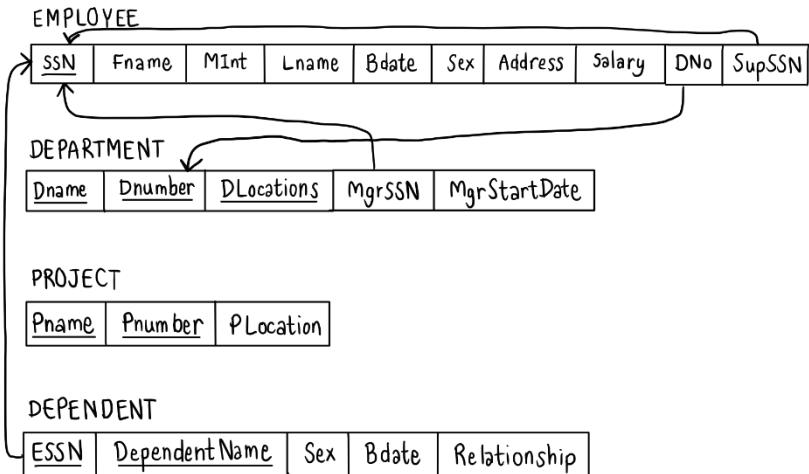
### วิธีที่ 3 ครอบเรเฟอเรนซ์

วิธีครอบเรเฟอเรนซ์ (cross reference) นี้ ให้สร้างรีเลชันใหม่ที่แอ็ตทริบิวต์ประกอบด้วยคีย์จากรีเลชันทั้งสอง และแอ็ตทริบิวต์ของรีเลชันซึ่งเป็นนั้น ถ้ามีจากนั้น ให้สร้าง FK จากคีย์เหล่านั้น โยงกลับไปที่รีเลชันเดิม

#### 5.1.4 ขั้นตอนที่ 4 : แปลงรีเลชันชิบໄท์ 1:N

ให้ยืดรีเลชันด้าน N เป็นหลัก แล้วนำ PK จากด้าน 1 มาเพิ่มเป็นแอ็ตทริบิวต์ในข้าง N และสร้าง FK โยงกลับไปด้าน 1 หากมีแอ็ตทริบิวต์ที่รีเลชันซึ่งเป็นนี้ให้นำมารวมที่รีเลชันหลักนี้ด้วย

จากตัวอย่าง รีเลชัน EMPLOYEE เป็นด้าน N และ รีเลชัน DEPARTMENT เป็นด้าน 1 จึงนำ Dnumber ซึ่งเป็น PK ใน DEPARTMENT มาเพิ่มในรีเลชัน EMPLOYEE เปลี่ยนชื่อ Dnumber เป็น DNo และโยง DNo เป็น FK กลับไปยัง Dnumber ใน DEPARTMENT ดังแสดงในรูปที่ 49

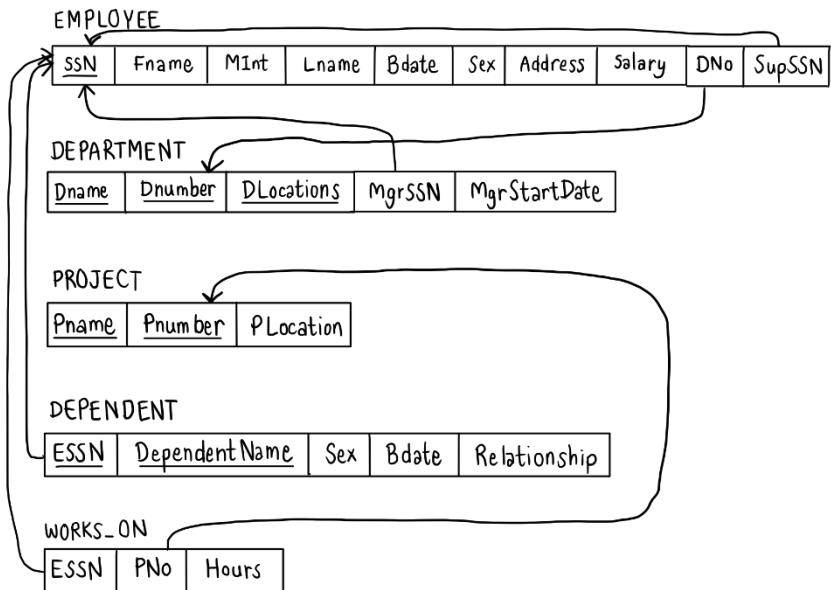


รูปที่ 49 แปลงรีเลชันชิบไปเป็น 1:N

### 5.1.5 ขั้นตอนที่ 5 : แปลงรีเลชันชิบเป็น M:N

ให้ใช้วิธีกรอสเรเพอเรนซ์ที่แสดงในขั้นตอนที่ 3

จากตัวอย่าง เราแปลงรีเลชันชิบ WORKS\_ON ที่เป็นแบบ M:N โดยสร้างรีเลชันใหม่ ชื่อ WORKS\_ON ที่ประกอบด้วย SSN จาก EMPLOYEE เปลี่ยนชื่อเป็น ESSN และ PNumber จาก PROJECT เปลี่ยนชื่อเป็น PNo มาเป็นคีย์ร่วม โดย FK ทั้งสองกลับไปที่รีเลชันหลัก แล้วรวม Hours มาด้วย ดังแสดงในรูปที่ 50



รูปที่ 50 แปลงรีเลชันชิบໄทปี m:n

### 5.1.6 ขั้นตอนที่ 6 : แปลงแอ็ตทริบิวต์หลายค่า

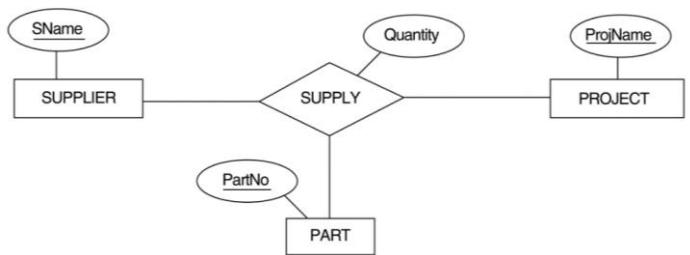
เมื่อพับแอ็ตทริบิวต์หลายค่า (multivalued attribute) ให้สร้างรีเลชันใหม่ ประกอบด้วยแอ็ตทริบิวต์หลายค่านั้นพร้อมกับคีย์ของรีเลชันที่พับ แอ็ตทริบิวต์หลายค่า แล้วกำหนดคีย์จากรีเลชันเดิมร่วมกับแอ็ตทริบิวต์หลายค่ามาเป็นคีย์ของรีเลชันใหม่นี้ แล้วโยง FK กลับไปยังรีเลชันหลักด้วย

จากตัวอย่าง แอ็ตทริบิวต์ Location เป็นแอ็ตทริบิวต์แบบหลายค่า ปรากฏอยู่ที่ DEPARTMENT ดังนั้น ให้สร้างรีเลชันใหม่ ชื่อ DEPT\_LOCATIONS

ประกอบด้วย แอ็ตทริบิวต์ Location กับ Dnumber กำหนดให้ทั้งสอง แอ็ตทริบิวต์เป็นคีย์ร่วม แล้วโดย FK กลับไป

### 5.1.7 ขั้นตอนที่ 7 : แปลงรีเลชันชิปแบบเอนนารี

ในกรณีที่รีเลชันชิปมีเอนทิตีมากกว่า 2 เออนทิตีขึ้นไปมาสัมพันธ์กัน ให้ใช้วิธี ครอบเร่อเรนซ์ที่แสดงในขั้นตอนที่ 3 คือให้สร้างรีเลชันใหม่ และนำคีย์จาก ทุกรีเลชันที่เกี่ยวข้องมากำหนดเป็นคีย์ร่วม พร้อมทั้งโยง FK กลับไปยัง รีเลชันหลักด้วย ถ้าพบแอ็ตทริบิวต์ของรีเลชันชิปก็ให้นำไปรวมด้วย จาก ตัวอย่าง พบท่อนารีหรือเอนนารีรีเลชันชิป ดังแสดงในรูปที่ 51 คือมี เอนทิตี SUPPLIER, PART และ PROJECT ดังนั้นให้สร้างรีเลชันใหม่ชื่อ SUPPLY ประกอบด้วย SName, ProjName และ PartNo แล้วโยง FK กลับไป พร้อม ทั้งรวมแอ็ตทริบิวต์ของรีเลชันชิปนั้นเข้าไปด้วย ซึ่งก็คือรวมแอ็ตทริบิวต์ Quantity เข้าไปด้วย ดังแสดงในรูปที่ 52



รูปที่ 51 แผนภาพอีอาร์แสดงรีเลชันชิปแบบเอนนารี

<u>SName</u>	...		
<u>ProjName</u>	...		
<u>PartNo</u>	...		
<u>SName</u>	<u>ProjName</u>	<u>PartNo</u>	<u>Quantity</u>

รูปที่ 52 ผลการแปลงรีเลชันชิปแบบเทอนารี

### 5.1.8 ขั้นตอนที่ 8 : แปลงซับคลาสชูเปอร์คลาส

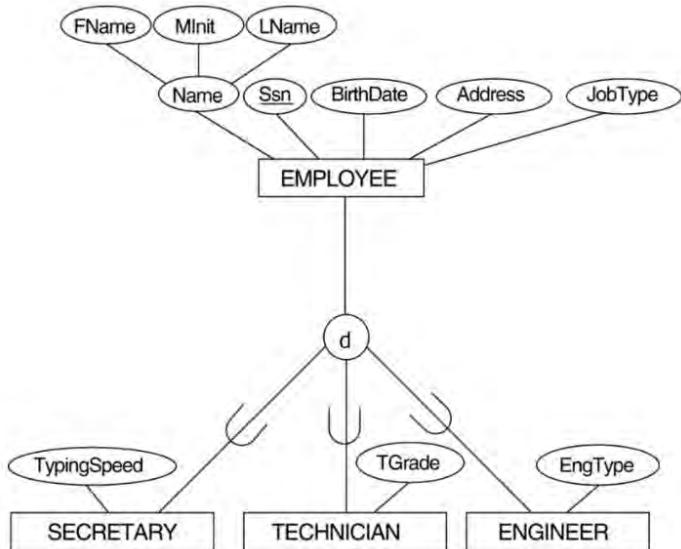
หากแผนภาพที่ใช้เป็นแผนภาพแบบอีอาร์ที่แสดงความสัมพันธ์ของข้อมูลแบบสเปเชียลไลเซชันหรือเจเนอเรลไลเซชัน จะมีวิธีให้เลือก 4 วิธี คือ

#### วิธีที่ 1 แยกรีเลชันชูเปอร์คลาสและซับคลาส

วิธีแยกรีเลชันชูเปอร์คลาสและซับคลาส (multiple relation – superclass and subclass) ให้สร้างรีเลชันใหม่สำหรับทุกชูเปอร์คลาสและซับคลาส สำหรับรีเลชันที่มาจากการชูเปอร์คลาส ให้รวมแอ็ตทริบิวต์ทั้งหมด แล้วกำหนด PK จากคีย์เดิม ส่วนรีเลชันที่มาจากการซับคลาส จะประกอบด้วยแอ็ตทริบิวต์ของตัวเองและคีย์ของชูเปอร์คลาส ดังแสดงการแปลงจากอีอาร์ในรูปที่ 53 เป็นรีเลชันในรูปที่ 54

ให้รวมแอ็ตทริบิวต์จากทั้งชูเปอร์คลาสและซับคลาスマ่ำว่าในรีเลชันเดียว และใช้คีย์จากการชูเปอร์คลาสเป็นคีย์หลัก

วิธีนี้ใช้ได้กับสเปเชียลไลเซชันทุกแบบไม่ว่าจะเป็น Total, Partial, Disjoint หรือ Overlap



รูปที่ 53 แผนภาพอธิบายแสดงชุมเปอร์คลาสซับคลาส EMPLOYEE

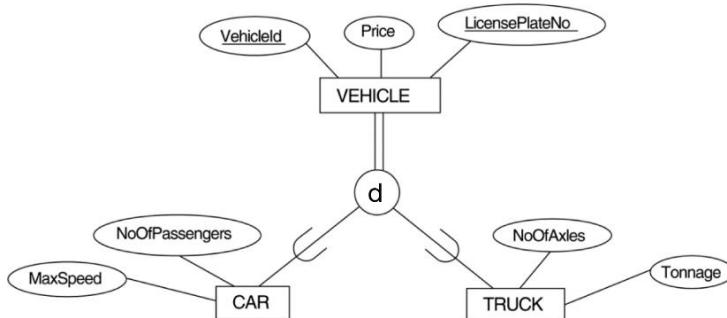
EMPLOYEE						
<u>SSN</u>	FName	MInit	LName	BirthDate	Address	JobType

SECRETARY		TECHNICIAN		ENGINEER	
<u>SSN</u>	TypingSpeed	<u>SSN</u>	TGrade	<u>SSN</u>	EngType

รูปที่ 54 ตัวอย่างการแปลงด้วยวิธีวิธีแยกหรือเลี้ยงชุมเปอร์คลาสและชับคลาส

## วิธีที่ 2 แยกรีเลชันให้ซับคลาส

วิธีแยกรีเลชันให้ซับคลาส (multiple relation – subclass only) ให้สร้างรีเลชันใหม่จากซับคลาสทุกตัว แล้วนำมารอตทริบิวต์ทั้งหมดจากซูเปอร์คลาส รวม พร้อมทั้งระบุคีย์ที่มาจากการซูเปอร์คลาสนั้น ๆ วิธีนี้ใช้ได้กับスペเชียลไลเซชันแบบ total รูปที่ 55 และรูปที่ 56 แสดงการแปลงดังกล่าว



รูปที่ 55 แผนภาพอวิอาร์แสดงซูเปอร์คลาสซับคลาส VEHICLE

CAR

VehicleId	LicensePlateNo	Price	MaxSpeed	NoOfPassengers
-----------	----------------	-------	----------	----------------

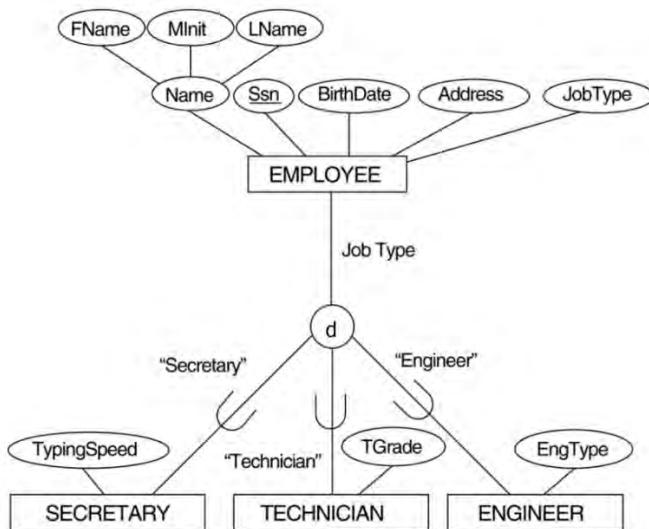
TRUCK

VehicleId	LicensePlateNo	Price	NoOfAxles	Tonnage
-----------	----------------	-------	-----------	---------

รูปที่ 56 ตัวอย่างการแปลงด้วยวิธีแยกรีเลชันให้ซับคลาส

### วิธีที่ 3 รวมรีเลชันพร้อมแอตทริบิวต์ระบุตัวเดียว

วิธีรวมรีเลชันพร้อมแอตทริบิวต์ระบุตัวเดียว (single relation – one type attribute) ให้สร้างรีเลชันใหม่ รวมแอตทริบิวต์จากชูเปอร์คลาสและชั้บคลาสทั้งหมดไว้ ตลอดจนแอตทริบิวต์ที่เป็นเพรดิคเตตกำหนด (predicate-defined attribute) บางตัวราเรียกแอตทริบิวต์ประเภทนี้ว่า ไทป์ (type) วิธีนี้หมายกับสเปเชียลไลเซชันแบบแอตทริบิวต์กำหนด (attribute defined-specialization) รูปที่ 58 แสดงผลการแปลงจากรูปที่ 57



รูปที่ 57 แผนภาพอีอาร์แสดงชูเปอร์คลาสชั้บคลาสแบบแอตทริบิวต์กำหนด

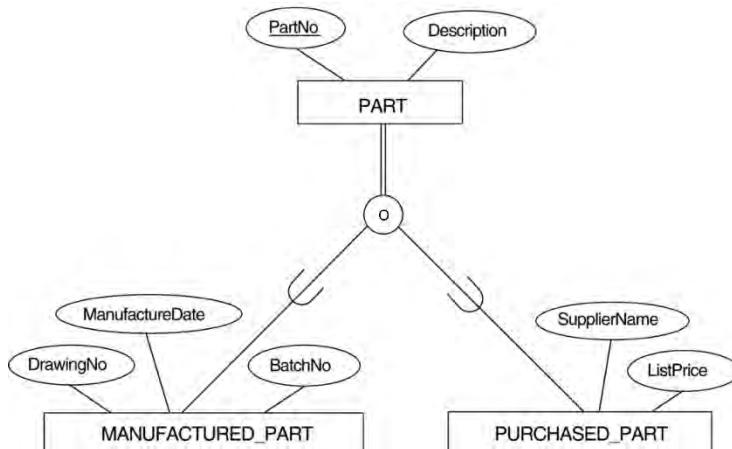
#### EMPLOYEE

SSN	FName	MInit	LName	BirthDate	Address	JobType	TypingSpeed	TGrade	EngType
-----	-------	-------	-------	-----------	---------	---------	-------------	--------	---------

รูปที่ 58 ตัวอย่างการแปลงด้วยวิธีรวมรีเลชันพร้อมแอตทริบิวต์ระบุตัวเดียว

#### วิธีที่ 4 รวมรีเลชันพร้อมแอ็ตทริบิวต์ระบุหลายตัว

วิธีรวมรีเลชันพร้อมแอ็ตทริบิวต์ระบุหลายตัว (Single Relation – Multiple Type Attribute) ให้รวมทุกแอ็ตทริบิวต์เป็นรีเลชันเดียว และเพิ่มแอ็ตทริบิวต์เท่ากับจำนวนชั้บคลาส กำหนดโดยเมนแอ็ตทริบิวต์ที่เพิ่มมานี้ให้เป็นบูลีนทั้งหมด เช่น ถ้ามี 3 ชั้บคลาสที่เพิ่มแอ็ตทริบิวต์ชนิดบูลีน 3 ตัว บูลีนใหม่นี้ทำหน้าที่ระบุว่าทูเพลินน์ ๆ เป็นสมาชิกของชั้บคลาสใด รูปที่ 60 แสดงการแปลงจากรูปที่ 59 วิธีนี้ใช้ได้กับสเปเชียลไลเซชันทุกแบบไม่ว่าจะเป็น total, partial, disjoint หรือ overlap



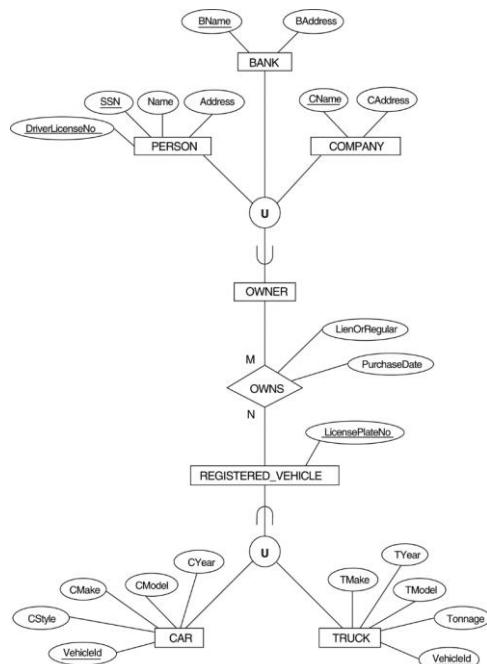
รูปที่ 59 แผนภาพอีอาร์แสดงงูเร่อร์คลาสชั้บคลาสแบบ overlap

PART								
PartNo	Description	MFlag	DrawingNo	ManufactureDate	BatchNo	PFlag	SupplierName	ListPrice

รูปที่ 60 วิธีรวมรีเลชันพร้อมแอ็ตทริบิวต์ระบุหลายตัว

### 5.1.9 ขั้นตอนที่ 9 : แปลงยูเนียน

เนื่องจากความสัมพันธ์แบบบัญเนียน มีชั้บคลาสประเภทแคทดกรีซึ่งมีชูเปอร์คลาสหลายชูเปอร์คลาสที่เป็นอนติติที่ต่างกัน ดังนั้นการรับทอดเป็นแบบเลือก (selective inheritance) ดังนั้น การแปลงอีอาร์แบบบัญเนียน จำเป็นต้องสร้างคีย์ใหม่ เรียกว่า คีย์เซอร์โรเกต (surrogate key) ให้กับแคทดกรีนน ๆ จากนั้นให้สร้างรีเลชันใหม่ให้กับชูเปอร์คลาสทั้งหมด และนำคีย์เซอร์โรเกตไปใช้เป็นคีย์หลักสำหรับทุกรีเลชันที่เกิดขึ้นใหม่ รูปที่ 62 แสดง การแปลงจากรูปที่ 61



รูปที่ 61 แผนภาพอีอาร์แสดงแคทดกรี

**PERSON**

SSN	DriverLicenseNo	Name	Address	OwnerId
-----	-----------------	------	---------	---------

**BANK**

BName	BAddress	OwnerId
-------	----------	---------

**COMPANY**

CName	CAddress	OwnerId
-------	----------	---------

**OWNER**

OwnerId
---------

**REGISTERED\_VEHICLE**

VehicleId	LicensePlateNumber
-----------	--------------------

**CAR**

VehicleId	CStyle	CMake	CModel	CYear
-----------	--------	-------	--------	-------

**TRUCK**

VehicleId	TMake	TModel	Tonnage	TYear
-----------	-------	--------	---------	-------

**OWNS**

OwnerId	VehicleId	PurchaseDate	LienOrRegular
---------	-----------	--------------	---------------

รูปที่ 62 แสดงการแปลงแบบบัญชียืน

## 5.2 บทสรุปและเรื่องชวนคิด

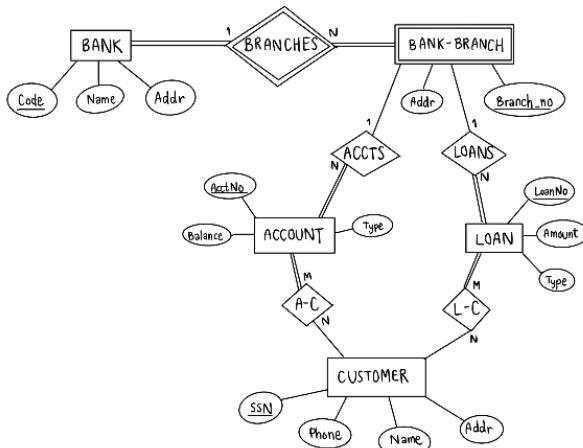
บทนี้เราได้เรียนรู้กระบวนการออกแบบฐานข้อมูลจากระดับแนวคิดเป็นระดับตรรกะ ตลอดจนขั้นตอนการแปลงแผนภาพอีอาร์และอีอาร์เป็นรีเลชันได้ จะเห็นได้ว่าคณิตศาสตร์ที่ใช้กับการทำงานของฐานข้อมูลเชิงสัมพันธ์นั้นไม่ซับซ้อนมากนัก จึงเป็นเหตุผลว่าเครื่องมือออกแบบฐานข้อมูลมักมีความสามารถที่จะแปลงแบบที่เราคาดเดาด้วยแผนภาพอีอาร์หรืออีอาร์ให้

อยู่ในรูปแบบบริเล็กซ์ได้ จากบทนี้ อย่างให้ผู้เรียนได้เห็นว่าการออกแบบที่ดี เป็นสิ่งสำคัญ เมื่อต้องการแก้ไขแบบ ควรแก้ที่แผนภาพก่อนเสมอ และอย่าง ให้ผู้เรียนศึกษาต่อว่ามีแบบจำลองและเครื่องมืออื่นที่รองรับฐานข้อมูลแบบ โนเนสคิวแอลหรือไม่

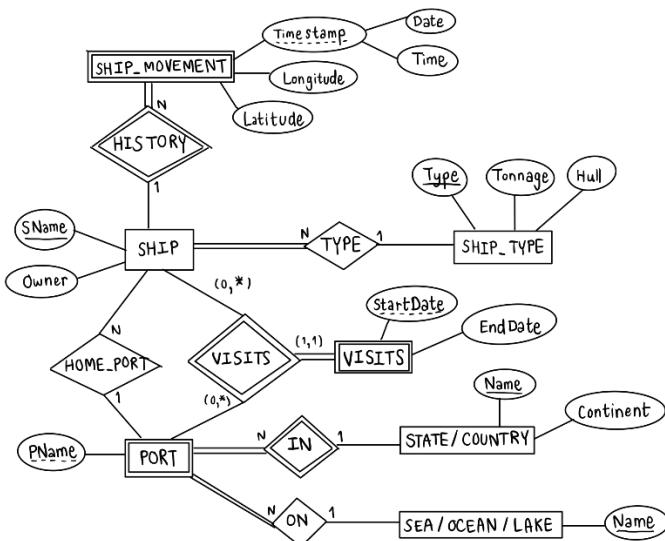
## แบบฝึกหัดท้ายบท

1. ขั้นตอนการแปลงอีอาร์และอีอาร์เป็นรีเลชัน มีกี่ขั้นตอน อะไรบ้าง
2. วิธีการแปลงซับคลาสซูเปอร์คลาสในขั้นตอนที่ 8 สำหรับแผนภาพแบบอีอาร์นั้นมีกี่วิธี อะไรบ้าง และแต่ละวิธีหมายความใดกับสเปเชียลไลเซชันแบบใดบ้าง
3. การแปลงรีเลชันชิปแบบใด ระหว่าง  $1:1$ ,  $1:N$  หรือ  $M:N$  จำเป็นต้องสร้างรีเลชันใหม่เพิ่มเสมอหรือไม่
4. การแปลงรีเลชันชิปแบบ  $1:N$  ผังใดที่จะต้องเพิ่มฟอร์เอนคีร์ ซึ่งเก็บคีย์หลักของอีกผังหนึ่ง
5. การแปลงรีเลชันแบบ  $1:1$  มีกี่วิธี อะไรบ้าง และแต่ละวิธีเมื่อแปลงแล้วจะเกิดรีเลชันสุดท้ายจำนวนกี่รีเลชัน
6. การเลือกวิธีการแปลงรีเลชันแบบ  $1:1$  แต่ละวิธี ควรใช้หลักเกณฑ์ใดในการพิจารณาเลือกวิธีการแปลง และพิจารณาอย่างไร
7. การแปลงรีเลชันชิปแบบonenar หากกำหนดให้มีค่าดีกรีความสัมพันธ์เป็น  $N$  แล้ว เมื่อแปลงเสร็จจะมีจำนวนรีเลชันเท่าใด

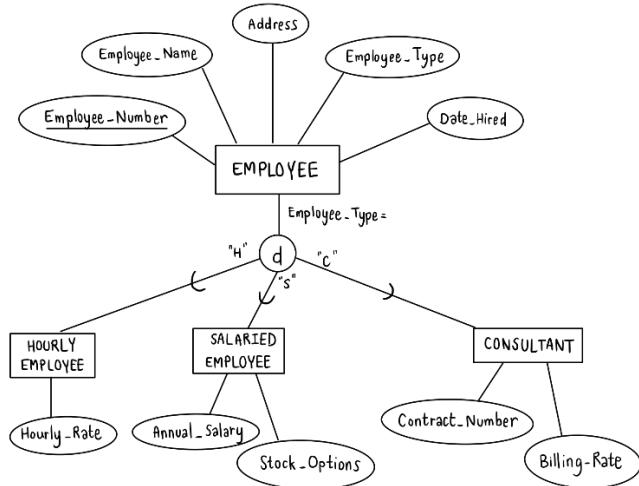
### 8. จงแปลงจากแผนภาพอธิการให้เป็นรีเลชัน



### 9. จงแปลงจากแผนภาพอธิการให้เป็นรีเลชัน



## 10. จงแปลงจากแผนภาพอีกอาร์ชิวีส์เป็นรีเลชัน



## บทที่ 6

### การนอร์ม่าໄලซ์

#### วัตถุประสงค์

- เพื่อให้ทราบถึงคุณสมบัติของฐานข้อมูลที่ดี
- เพื่อให้เข้าใจเรื่องฟังก์ชันนัลดีเพนเดนซี
- เพื่อให้เข้าใจรูปแบบนอร์มัลฟอร์มที่หนึ่ง ที่สอง ที่สาม และแบบบีชีเอ็นเอฟ
- เพื่อให้เข้าใจกระบวนการนอร์ม่าໄලซ์

ฐานข้อมูลที่ดีคือฐานข้อมูลที่มีความผิดพลาดน้อยที่สุด และมีความซ้ำซ้อน (redundancy) น้อยที่สุด เพราะเมื่อข้อมูลซ้ำซ้อนก็จะทำให้เปลืองพื้นที่การจัดเก็บ และอาจส่งผลให้ข้อมูลผิดพลาดหรือข้อมูลเพี้ยน (anomaly) ไปได้ เมื่อปรับปรุงเปลี่ยนแปลงข้อมูลซึ่งเกิดได้เมื่อมีการแทรก ลบ หรือแก้ไข ข้อมูลนั้นเอง

วิธีการลดความซ้ำซ้อนทำได้ด้วยการแยกรีเลชันออกเป็นรีเลชันย่อยซึ่งเรียกว่า กระบวนการย่ออย (decomposition) และปัจจัยที่ต้องคำนึงถึงในกระบวนการย่ออยนี้ ก็คือความขึ้นต่อ กันของข้อมูลในฐานข้อมูล เราเรียกปัจจัยนี้ว่า พังก์ชันนัลเดพenedenz

ส่วนการ นอร์มาไลซ์ คือกระบวนการย่ออยที่คำนึงถึงพังก์ชันนัลเดพenedenz ร่วมกับคีย์ ให้ตรงกับ นอร์มลฟอร์ม ระดับต่าง ๆ ซึ่งเราจะกล่าวถึงต่อไป [3, 10, 12, 13, 15, 25, 43]

## 6.1 ปัญหาจากความซ้ำซ้อน

เมื่อมีความซ้ำซ้อน (redundancy) ของข้อมูลในรีเลชัน จะมีโอกาสเกิดความผิดพลาดเมื่อมีการปรับปรุงข้อมูลได้มาก ความผิดพลาดจากการปรับปรุงข้อมูล แบ่งได้เป็น 3 กรณีคือ

### 6.1.1 ความผิดพลาดจากการแก้ไข

ความผิดพลาดจากการแก้ไข (update anomaly) อาจเกิดขึ้นได้เมื่อเราแก้ไขข้อมูลไม่ครบถ้วน

ตัวอย่าง

EMP\_PROJ(Emp#, Proj#, Ename, Pname, No\_hours)

ความผิดพลาดจากการแก้ไขอาจเกิดขึ้นได้ เช่น สมมุติว่าหากต้องการเปลี่ยนชื่อ Pname จาก “Human Resource” เป็น “Human Development” นั้น เราจะต้องไปแก้ไขข้อมูลดังกล่าวในทุกๆเพลิ หากแก้ไม่ครบ ข้อมูลก็จะผิดแน่นอน

#### 6.1.2 ความผิดพลาดจากการแทรก

ยกตัวอย่างความผิดพลาดจากการแทรก (insert anomaly) จากตัวอย่าง ก่อนหน้า นั่นคือ เราไม่สามารถแทรก Project ใหม่เข้าไปในรีเลชัน จนกว่าจะกำหนด Employee ให้มี Project เสียก่อน

#### 6.1.3 ความผิดพลาดจากการลบ

ความผิดพลาดจากการลบ (delete anomaly) จากตัวอย่างก่อนหน้า อาจเกิดขึ้นเมื่อต้องการลบ Project หนึ่งออกไปจะส่งผลให้มีการลบ Employee ที่ทำงานอยู่ใน Project นั้นทั้งหมด หรือ อีกทางหนึ่งถ้า Employee นั้นทำงานอยู่เพียง Project เดียว การลบ Employee นั้นจะส่งผลให้ Project นั้นหายไปจากรีเลชัน

ดังนั้น วิธีการที่ดีในการออกแบบฐานข้อมูลที่จะลดปัญหาความผิดพลาดนี้ได้คือการย่อรีเลชันนั้นเอง

## 6.2 พังก์ชันนัลตี้เพนเดนซี

พังก์ชันนัลตี้เพนเดนซี (functional dependency) คือ เงื่อนไขความขึ้นต่อ กันของแอ็ตทริบิวต์ในรีเลชัน ได้มาจากการตีความในขั้นตอนการรวบรวมและวิเคราะห์ความต้องการของระบบ เป็นความหมายของข้อมูลที่ตีความโดยผู้ออกแบบตามความตั้งใจของเจ้าของระบบ โดยเน้นที่ความสัมพันธ์ระหว่างแอ็ตทริบิวต์ในฐานข้อมูลนั้น ๆ

พังก์ชันนัลตี้เพนเดนซี อาจแปลเป็นไทยได้ว่าความขึ้นต่อ กันทางพังก์ชัน แต่เนื่องจากฟังแล้วเข้าใจยาก จึงขอใช้ทับศัพท์ว่า พังก์ชันนัลตี้เพนเดนซี ย่อว่า FD ต่อไปนี้จะเรียกว่า FD

ศัพท์คำว่า กำหนด (determine) และคำว่า ขึ้นกับ (depend) เป็นหัวใจสำคัญของ FD เมื่อเขียนสัญกรณ์คณิตศาสตร์แสดงพังก์ชันนัลตี้เพนเดนซี ให้ใช้เครื่องหมายลูกศร ดังนี้

$$X \rightarrow Y$$

อ่านว่า X กำหนดพังก์ชัน Y (X functionally determines Y) หรืออ่านว่า พังก์ชัน Y ขึ้นกับ X (Y depends on X) เมื่อ X และ Y คือเซตของแอ็ตทริบิวต์ X และ Y ตามลำดับ

ความหมายคือ เมื่อรู้ค่า X แล้วจะรู้ค่า Y หรือค่า Y ขึ้นกับค่า X โดย X และ Y อาจเป็นแอ็ตทริบิวต์หรือเซตของแอ็ตทริบิวต์ก็ได้

เราสามารถอธิบายพังก์ชันนัลตี้เพนเดนซีในเชิงคณิตศาสตร์ได้ดังนี้

$FD: X \rightarrow Y$  สำหรับทุกอินสแตนซ์ (instance)  $r$  ในรีเลชัน  $R$  เมื่อ

$$t_1 \in r, t_2 \in r, \pi_x(t_1) = \pi_x(t_2) \text{ implies } \pi_y(t_1) = \pi_y(t_2)$$

คือ หากทุกเพลิ  $t_1$  และ  $t_2$  เป็นสมาชิกของ  $r$  และ ค่า  $x$  ใน  $t_1$  เท่ากับ  $x$  ใน  $t_2$  แล้ว ค่า  $y$  ใน  $t_1$  ย่อมเท่ากับ  $y$  ใน  $t_2$  ด้วย โดย  $x$  และ  $y$  เป็นเซตของ แอ็ตทริบิวต์

จากตัวอย่างอินสแตนซ์ที่เป็นไปได้ เราสามารถตรวจสอบได้ว่า อินสแตนซ์นั้น สอดคล้องกับเงื่อนไขของ FD หรือไม่ แต่ไม่สามารถสรุปได้ว่า FD นั้นทรง (hold) สำหรับ  $r$  นั้น ๆ

เมื่อ  $K$  เป็นคีย์คู่แข่ง (candidate key) ของ  $R$  เราบอกได้ว่า  $K \rightarrow R$  แต่ไม่สามารถสรุปได้ว่า  $K$  เป็นคีย์คู่แข่งที่เล็กที่สุด (minimal)

$$A_1, A_2, \dots, A_n \rightarrow B$$

อ่านว่า  $A_1, A_2, \dots, A_n$  กำหนด  $B$

ถ้า

$$A_1, A_2, \dots, A_n \rightarrow B_1$$

$$A_1, A_2, \dots, A_n \rightarrow B_2$$

...

...

$$A_1, A_2, \dots, A_n \rightarrow B_m$$

ตั้งนั้น  $A_1, A_2, \dots, A_n \rightarrow B_1, B_2 \dots B_m$

### 6.3 การหาฟังก์ชันลักษณะเด่นซึ่งจากข้อมูล

จากข้อมูลในรีเลชัน Employee เราอาจกำหนด F ด้วยการสังเกตความ  
ขึ้นต่อ กันของข้อมูล ตัวอย่างและรูปในส่วนนี้นำมาจาก [2]

ตัวอย่าง

Employee				
emp_no	name	dept_no	dept_name	skills
1	Chonlathorn Kwankajornkiet	201	R&D	C
1	Chonlathorn Kwankajornkiet	201	R&D	Perl
1	Chonlathorn Kwankajornkiet	201	R&D	Java
2	Kallerk Treetipsounthorn	224	IT	Linux
2	Kallerk Treetipsounthorn	224	IT	Mac
3	Kawin Metsiritrakul	201	R&D	DB2
3	Kawin Metsiritrakul	201	R&D	Oracle
3	Kawin Metsiritrakul	201	R&D	Java

รูปที่ 63 รีเลชัน Employee

จากรีเลชัน Employee ในรูปที่ 63 จะเห็นได้ว่า emp\_no เดียว กันจะได้ name, dept\_no, และ dept\_name เดียว กัน เสมอ แต่ไม่ได้ skills เดียว กัน จึงเขียน F ได้ว่า

$$F = \{emp\_no \rightarrow name, dept\_no, dept\_name\}$$

ตัวอย่าง

ให้ Hourly\_Emps=(SSN, Name, Level, Rate, WageRate, Hours)

เขียนย่อว่า Hourly\_Emps=(SNLRWH)

S	N	L	R	W	H
123-22-3666	Chonlathorn	48	8	10	40
231-31-5368	Kailerk	22	8	10	30
131-24-3650	Kawin	35	5	7	30
434-26-3751	Sanchai	35	5	7	32
612-67-4134	Thanyanuch	35	8	10	40

รูปที่ 64 Hourly\_Emps=(SNLRWH)

จากข้อมูลในรูปที่ 64 จะเห็นได้ว่า S ไม่ซ้ำ จึงอาจอนุมานว่า S เป็นคีย์ นอกจานั้น จะเห็นว่า ค่า R เดียวกันจะได้ W เดียวกันเสมอ จึงเขียน F ได้ ดังนี้

$$F = \{S \rightarrow SNLRWH, R \rightarrow W\}$$

อย่างไรก็ตี จาก 2 ตัวอย่างข้างบนจะพบว่าข้อมูลมีความซ้ำซ้อนเกิดขึ้น ดังนั้นเราจึงควรย่อรีเลชันเพื่อลดความซ้ำซ้อนนั้น จากตัวอย่างนี้ เรา สามารถย่อรีเลชันได้ดังแสดงในรูปที่ 65

S	N	L	R	H
123-22-3666	Chonlathorn	48	8	40
231-31-5368	Kailerk	22	8	30
131-24-3650	Kawin	35	5	30
434-26-3751	Sanchai	35	5	32
612-67-4134	Thanyanuch	35	8	40

R	W
8	10
5	7

รูปที่ 65 ย่อรีเลชัน Hourly\_Emps

## 6.4 การหาฟังก์ชันนัลตี้เพนเดนซีทั้งหมด

เราจำเป็นที่จะต้องทราบฟังก์ชันนัลตี้เพนเดนซีทั้งหมดของรีเลชันเพื่อให้เข้าใจความขึ้นต่อ กันของข้อมูล และนำไปออกแบบฐานข้อมูลให้ละเอียดและสมบูรณ์ที่สุด การที่จะทราบฟังก์ชันนัลตี้เพนเดนซี F ทั้งหมดทำได้โดยการอนุमานจากกฎที่เรียกว่า สัจพจน์ของอาร์มสตรอง (Armstrong's axioms) [44] และจะได้  $F^+$  ( $F$ -closure) ซึ่งคือ F ทั้งหมดที่เป็นไปได้นอกจากนั้น เรายังสามารถใช้สัจพจน์ของอาร์มสตรองเพื่อหา  $X^+$  ( $X$ -closure) ซึ่งหมายถึงแอ็ตทริบิวต์ที่อนุมานได้จากแอ็ตทริบิวต์ตั้งต้นโดยคำนึงถึงดีเพนเดนซีของรีเลชันนั้น ๆ ได้อีกด้วย

### 6.4.1 เอฟโคลเชอร์

FD หนึ่ง ๆ แทนด้วย  $f$

เซตของ FD แทนด้วย F

การที่  $f$  หนึ่ง ๆ จะทรง (hold) ได้นั้น ก็ต่อเมื่อ  $F$  ทั้งหมดทรงด้วย เราจึงจำเป็นต้องหา  $F$  ทั้งหมดให้ได้ วิธีการหา  $F$  ทั้งหมดนี้ทำได้ด้วยการอนุมาน แล้วเรียกเซตของ  $F$  ทั้งหมดนี้ว่า  $F^+$  อ่านว่า เอฟโคลเชอร์ ( $F$ -closure)

เอฟโคลเชอร์ให้แทนด้วย  $F^+$  คือ  $F$  ทั้งหมดที่ได้มาจากการอนุมาน เมื่อเราทราบ  $F$  เราสามารถอนุมาน  $F$  ทั้งหมดได้จากกฎ 3 กฎ เรียกว่าสัจพจน์ของ าร์มสตรอง อันได้แก่

1. กฎการสะท้อน (reflexivity) ถ้า  $X \subseteq Y$  และ  $Y \rightarrow X$
2. กฎการแต่งเติม (augmentation) ถ้า  $X \rightarrow Y$  และ  $XZ \rightarrow YZ$   
สำหรับ  $Z$  ใด ๆ
3. กฎการถ่ายทอด (transitivity) ถ้า  $X \rightarrow Y$  และ  $Y \rightarrow Z$  และ  $X \rightarrow Z$

ทั้ง 3 กฎที่กล่าวมาเรียกได้ว่าเป็นกฎการอนุมานที่สมบูรณ์ในตัวแล้ว กฎอื่นใดที่ใช้ในการอนุมานล้วนมีรากฐานจาก 3 กฎนี้ทั้งสิ้น เช่น

- กฎการยูเนียน (union) ถ้า  $X \rightarrow Y$  และ  $X \rightarrow Z$  และ  $X \rightarrow YZ$
- กฎการย่อย (decomposition) ถ้า  $X \rightarrow YZ$  และ  $X \rightarrow Y$  และ  $X \rightarrow Z$

ตัวอย่าง

ให้  $R(A,B,C)$

$F = \{A \rightarrow B, B \rightarrow C\}$

เราสามารถอนุมาน  $F$  อื่น ๆ จากสัจพจน์ของอาร์มสตรองได้มากmany  
ขอยกตัวอย่างผลจากการอนุมาน ดังนี้

$A \rightarrow B$  (จาก  $f$  ที่ให้มา)

$B \rightarrow C$  (จาก  $f$  ที่ให้มา)

$A \rightarrow A$  (Reflexive)

$B \rightarrow B$  (Reflexive)

$C \rightarrow C$  (Reflexive)

$A \rightarrow C$  (Transitive)

$AC \rightarrow BC$  (Augmentation)

$BA \rightarrow CA$  (Augmentation)

ทั้งนี้ จำนวนของ  $f$  ใน  $F^+$  เป็นพังก์ชันเอกซ์โพเนนเชียลของจำนวน  
แอตทริบิวต์ ซึ่งจะมีขนาดใหญ่มาก ทำให้การคำนวณหา  $F^+$  เป็นเรื่องที่  
สิ้นเปลืองหากจะต้องหาทุก  $f$

ตัวอย่าง

ให้ Contracts(cid, sid, jid, did, pid, qty, value)

ขอย่อว่า Contracts(CSJDPQV)

และ C เป็นคิร์ตั้งนั้น

$C \rightarrow CSJDPQV$

ถ้าเราทราบว่าเมื่อแต่ละ Project จะซื้อ Part จะต้องสร้างสัญญาขึ้นมา เราจะได้  $f$  คือ  $JP \rightarrow C$

ถ้าเราระบุว่าแต่ละ Department สามารถซื้อ Part ได้อย่างมาก 1 จาก 1 Supplier จะได้  $f$  คือ  $SD \rightarrow P$

ดังนั้น  $F = \{C \rightarrow CSJDPQV, JP \rightarrow C, SD \rightarrow P\}$

เมื่อนำมาด้วยสักพจน์ของาร์มสตรองจะได้ว่า

$JP \rightarrow C, C \rightarrow CSJDPQV \text{ imply } JP \rightarrow CSJDPQV$

$SD \rightarrow P \text{ implies } SDJ \rightarrow JP$

$SDJ \rightarrow JP, JP \rightarrow CSJDPQV \text{ imply } SDJ \rightarrow CSJDPQV$

#### 6.4.2 เอกซ์คลูเซอร์

เราจะแทนแอ็ตทริบิวต์แต่ละตัวด้วย  $X$  และคำนวน เอกซ์คลูเซอร์ ( $X$ -closure หรือ  $X^+$ ) ด้วยแนวคิดเดียวกันกับ  $F^+$  นั่นคือการอนุมาน  $X$  ทั้งหมดด้วยสัจพจน์ของอาร์มสตรอง

ประযุชน์ของการคำนวน  $F^+$  และ  $X^+$  ในที่นี้ ก็เพื่อหาเขตของแอ็ตทริบิวต์  $X$  ที่เป็นคีย์ที่เหมาะสมของรีเลชันนั้น ๆ

ตัวอย่าง

$$R(A, B, C, D, E)$$

$$F = \{A \rightarrow D, D \rightarrow B, B \rightarrow C, E \rightarrow B\}$$

$$A^+ = \{A \text{ (reflexive)}, D, B \text{ (transitive)}, C \text{ (transitive)}\}$$

$$B^+ = \{B, C\}$$

$$C^+ = \{C\}$$

$$D^+ = \{D, B, C\}$$

$$E^+ = \{E, B, C\}$$

## 6.5 การหาคีร์จากราก $F^+$ และ $X^+$

เหตุผลที่จะคำนวณ  $F^+$  และ  $X^+$  ก็เพื่อจะหาคีร์ที่เหมาะสมกับรีเลชันนั้น นั่นหมายความว่า คีร์ที่ได้มานั้น จะกำหนดแอตทริบิวต์ทั้งหมดของรีเลชันได้ ซึ่งเราจะสามารถทราบได้ว่าคีร์นั้นจะกำหนดได้ทุกแอตทริบิวต์โดยการคำนวณด้วย  $F$  นั่นเอง

ดังนั้น ถ้าให้  $X$  เป็นคีร์ของรีเลชัน  $R$  จะเขียนได้ว่า  $X \rightarrow R$

นอกเหนือจากนั้น การที่จะเป็นคีร์ได้ เขตของแอตทริบิวต์  $X$  ต้องเล็กที่สุด ด้วย

ขั้นตอนการหาคีร์และคุ่警่ทั้งหมด ให้เริ่มจาก  $X$  ที่อยู่ช้ายของลูกศร เพราะแอตทริบิวต์ใด ๆ ที่อยู่ช้ายอยู่ในตัวกำหนดแอตทริบิวต์อื่นได้ ในขณะที่แอตทริบิวต์ที่อยู่ขาวของลูกศรย่อมเป็นแอตทริบิวต์ที่ขึ้นต่อตัวอื่น

ตัวอย่าง

ให้  $R(A, B, C)$

$$F = \{A \rightarrow B, B \rightarrow C\}$$

จะเห็นว่า  $A$  และ  $B$  อาจเป็นคีร์ได้ เพราะอยู่ช้ายของลูกศร แต่  $C$  ไม่มีโอกาสเป็นคีร์

จากนั้นให้คำนวณ  $A^+$  และ  $B^+$  จาก  $F$

$$A^+ = \{A, B, C\}$$

ดังนั้น A ต้องเป็นคิร์คูແبغแน่ ๆ เพราะสามารถกำหนดแอ็ตทริบิวต์ทุกตัวในรีเลชันได้ และถ้าไม่มีคิร์คูແبغอื่น ก็จะสรุปได้ว่า A เป็นคิร์คิลชันได้

ต่อไปให้คำนวน  $B^+$

$$B^+ = \{B, C\}$$

เห็นได้ว่า B ไม่ใช่คิร์คูແبغ เพราะไม่สามารถกำหนด A ได้

สรุปได้ว่า รีเลชันนี้ มีคิร์คิลชัน

## 6.6 การนอร์มาไอลซ์

การนอร์มาไอลซ์ (normalization) [3, 13, 43] คือการย่ออย่างไรเลชันเพื่อลดปัญหาความซ้ำซ้อนและลดโอกาสที่จะเกิดความผิดพลาดเมื่อมีการปรับแก้การย่ออยนี้จะย่อเพื่อให้อยู่ในรูปแบบต่าง ๆ ที่มีความเข้มงวดมากขึ้นตามลำดับ

รูปแบบที่ว่ามีเรียกว่า นอร์มัลฟอร์ม (normal form) ซึ่งตำราจะครอบคลุม 4 นอร์มัลฟอร์ม ได้แก่ นอร์มัลฟอร์มที่ 1, นอร์มัลฟอร์มที่ 2, นอร์มัลฟอร์มที่ 3, และนอร์มัลฟอร์มบีชี โดยการย่อเพื่อให้ได้นอร์มัลฟอร์มทั้งสี่นี้จะดำเนินถึง 2 ปัจจัยคือ 1) พงกชันนัดเดียวเดนซ์ และ 2) คิร์คิลชัน

ในการย่ออย่างไรเลชันนี้ เราจะต้องดำเนินถึงเป้าหมายสำคัญ 2 เป้าหมายคือ

1. “ไม่มีปัญหาเมื่อเชื่อมรีเลชันคืน” (lossless-join decomposition) กล่าวคือ เมื่อได้ที่ย่อรีเลชัน เราจะต้องมั่นใจว่า เมื่อเชื่อมรีเลชัน ย่ออยู่เหล่านั้นกลับคืนแล้วจะได้ข้อมูลชุดเดิม ข้อมูลไม่หายและ ข้อมูลไม่เกิน (non-additive) หรือไม่มีข้อมูลแฝลงปлом (spurious) หรืออีกแร คือมั่นใจว่าเรายังสามารถคืนคืนข้อมูลได ๆ ที่มิอยู่ในรีเลชันก่อนถูกย่อออกจากตารางที่ย่ออยแล้วได้เสมอ ปัญหานี้ มักไม่เกิดหากการย่ออยรีเลชันนั้นได้ยึดคีย์เป็นหลักสำคัญในการย่อ
2. พังก์ชันนัลตี้เพนเดนชียังคงสภาพ (dependency-preserving decomposition) กล่าวคือ เมื่อย่อรีเลชันแล้วยังคงเก็บ พังก์ชันนัลตี้เพนเดนชีเดิมไว้ได้ทั้งหมด

## 6.7 ขั้นตอนการนอร์มайлize

ตามที่ได้กล่าวไปแล้วว่าการนอร์มайлize คือกระบวนการย่ออยรีเลชันให้อยู่ใน รูปแบบต่าง ๆ ทั้ง 4 แบบ กระบวนการนอร์มайлize มีรายละเอียดดังนี้

### 6.7.1 นอร์มัลฟอร์มที่ 1

นอร์มัลฟอร์มที่ 1 (First Normal Form) มักเขียนย่อว่า 1NF ไม่อนุญาตให้ รีเลชันมีแอตทริบิวต์ร่วม หรือแอตทริบิวต์หลายค่า หรือความสัมพันธ์ซ้อนใน (nested) และแอตทริบิวต์ทั้งหมดต้องเป็นอะตอนมิก แยกต่อไปไม่ได้

รูปที่ 66 แสดงตัวอย่างแอตทริบิวต์หลายค่า คือ DLOCATIONS วิธีการแก้ไข มี 3 วิธีคือ

**DEPARTMENT**

DNAME	DNUMBER	DMGRSSN	DLOCATION

**DEPARTMENT**

DNAME	DNUMBER	DMGRSSN	DLOCATION
Research	5	333445555	(Bangkok, Phuket, Chiang Mai)
Administration	4	987654321	(Pathum Thani)
Headquarters	1	888665555	(Bangkok)

**DEPARTMENT**

DNAME	DNUMBER	DMGRSSN	DLOCATION
Research	5	333445555	Bangkok
Research	5	333445555	Phuket
Research	5	333445555	Chiang Mai
Administration	4	987654321	Pathum Thani
Headquarters	1	888665555	Bangkok

รูปที่ 66 รีเลชันที่ไม่แยกทรีบิวต์หลายค่า

- ให้สร้างรีเลชันใหม่ แล้วนำແອຕທຽບวิวัต์หลายค่า DLOCATION ไปใส่ไว้ พร้อมกับคีย์ของรีเลชันนั้น จะได้ผลดังแสดงใน

**DEPARTMENT**

DNAME	DNUMBER	DMGRSSN
Research	5	333445555
Administration	4	987654321
Headquarters	1	888665555

**DEPT\_LOCATION**

DNUMBER	DLOCATION
1	Bangkok
4	Phuket
5	Chiang Mai
5	Pathum Thani
5	Bangkok

รูปที่ 67 DEPT\_LOCATION 1NF

2. ขยายคีย์ให้ครอบคลุมถึงแอตทริบิวต์หลายค่านั้น ผลคือ

DEPARTMENT (DNAME, DNUMBER, DMGRSSN, DLOCATION)

DEPARTMENT			
DNAME	DNUMBER	DMGRSSN	DLOCATION
Research	5	333445555	Bangkok
Research	5	333445555	Phuket
Research	5	333445555	Chiang Mai
Administration	4	967654321	Pathum Thani
Headquarters	1	888665555	Bangkok

รูปที่ 68 รีเลชันในรูปแบบ 1NF ด้วยวิธีขยายคีย์

3. หากทราบจำนวนที่เป็นไปได้ของแอตทริบิวต์หลายค่า ให้เพิ่ม  
แอตทริบิวต์ตามจำนวนนั้น เช่น หากทราบว่ามี LOCATION 3 แห่ง  
ผลคือ

DEPARTMENT (DNAME, DNUMBER, DMGRSSN, DLOC1,  
DLOC2, DLOC3)

เมื่อปรับรีเลชันให้อยู่ในรูปแบบ 1NF แล้วจะมีสคีมาคร่าว ๆ ดังแสดงในรูปที่ 69 ส่วนรูปที่ 70 แสดงสถานะหนึ่งของข้อมูลในฐานข้อมูลบริษัทดีบี

**EMPLOYEE**

f.k.

ENAME	<u>SSN</u>	BDATE	ADDRESS	DNUMBER
-------	------------	-------	---------	---------

p.k.

**DEPARTMENT**

f.k.

DNAME	<u>DNUMBER</u>	DMGRSSN
-------	----------------	---------

p.k.

**DEPT\_LOCATIONS**

f.k.

DNUMBER	DLOCATION
p.k.	

**PROJECT**

f.k.

PNAME	<u>PNUMBER</u>	PLOCATION	DNUM
-------	----------------	-----------	------

p.k.

**WORKS\_ON**

f.k.

f.k.

SSN	<u>PNUMBER</u>	HOURS
p.k.		

รูปที่ 69 ศีล米ของบริษัทดีบี

**EMPLOYEE**

ENAME	SSN	BDATE	ADDRESS	DNUMBER
Smith,John B.	123456789	1965-01-09	731 Fondren,Houston,TX	5
Wong,Franklin T.	333445555	1955-12-08	638 Voss,Houston,TX	5
Zelaya,Alicia J.	999887777	1968-07-19	3321 Castle,Spring,TX	4
Wallace,Jennifer S.	987654321	1941-06-20	291 Berry,Bellaire,TX	4
Narayan,Rmesh K.	666884444	1962-09-15	975 Fire Oak,Humble,TX	5
English,Joyce A.	453453453	1972-07-31	5631 Rice,Houston,TX	5
Jabbar,Ahmad V.	987987987	1969-03-29	980 Dallas,Houston,TX	4
Borg,James E.	888665555	1937-11-10	450 Stone,Houston,TX	1

**DEPARTMENT**

DNAME	DNUMBER	DMGRSSN
Research	5	333445555
Administration	4	987654321
Headquarters	1	888665555

**DEPT\_LOCATIONS**

DNUMBER	DLOCATION
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

**WORKS\_ON**

SSN	PNUMBER	HOURS
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	null

**PROJECT**

PNAME	PNUMBER	PLOCATION	DNUM
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

รูปที่ 70 สถานะของข้อมูลในฐานข้อมูลบริษัทเดียวกัน

### 6.7.2 นอร์มลฟอร์มที่ 2

รีเลชันที่ผ่าน 1NF แต่ไม่ผ่านนอร์มลฟอร์มที่ 2 (2NF) จะมีปัญหาเรื่องความซ้ำซ้อนเกิดขึ้น เช่น เมื่อเชื่อมรีเลชัน EMPLOYEE กับ PROJECT ได้เป็น EMP\_PROJ จะเกิดความซ้ำซ้อนของข้อมูลในคอลัมน์ 2 ชุด ดังแสดงในรูปที่ 71 คือ คือข้อมูลในคอลัมน์ HOURS และ ENAME ซ้ำกันอยู่หลายทุเพล และข้อมูล PNAME PLOCATION ก็ซ้ำกันหลายทุเพล

EMP_PROJ		ข้าช้อน		ข้าช้อน	
SSN	PNUMBER	HOURS	ENAME	PNAME	PLOCATION
123456789	1	32.5	Smith,John B.	ProductX	Bellaire
123456789	2	7.5	Smith,John B.	ProductY	Sugarland
666884444	3	40.0	Narayan,Ramesh K.	ProductZ	Houston
453453453	1	20.0	English,Joyce A.	ProductX	Bellaire
453453453	2	20.0	English,Joyce A.	ProductY	Sugarland
333445555	2	10.0	Wong,Franklin T.	ProductY	Sugarland
333445555	3	10.0	Wong,Franklin T.	ProductZ	Houston
333445555	10	10.0	Wong,Franklin T.	Computerization	Stafford
333445555	20	10.0	Wong,Franklin T.	Reorganization	Houston
999887777	30	30.0	Zelaya,Alicia J.	Newbenefits	Stafford
999887777	10	10.0	Zelaya,Alicia J.	Computerization	Stafford
987987987	10	35.0	Jabbar,Ahmad V.	Computerization	Stafford
987987987	30	5.0	Jabbar,Ahmad V.	Newbenefits	Stafford
987654321	30	20.0	Wallace,Jennifer S.	Newbenefits	Stafford
987654321	20	15.0	Wallace,Jennifer S.	Reorganization	Houston
888665555	20	null	Borg,James E.	Reorganization	Houston

รูปที่ 71 รีเลชัน EMP\_PROJ

ดังนั้น เราจะย่อรีเลชันโดยคำนึงถึงคีย์ทั้งหมดของรีเลชัน เพราะถ้าหาก  
แอตทริบิวต์ขึ้นต่อคีย์แล้วจะทำให้ตารางลดความซ้ำซ้อนแน่นอน

นอร์มลัฟอร์มที่ 2 (Second Normal Form) หรือ 2NF ระบุว่า ต้องผ่าน  
นอร์มลัฟอร์มที่ 1 และทุกแอตทริบิวต์ที่ไม่ใช่ไพร์เม่แอตทริบิวต์จะต้องขึ้นต่อ  
ทุกคีย์ของรีเลชันแบบทั้งหมด

ไพร์เม่แอตทริบิวต์ (prime attribute) คือ แอตทริบิวต์ที่เป็นส่วนหนึ่งของคีย์  
รวมถึงคีย์คู่แข่ง (candidate key) ด้วย

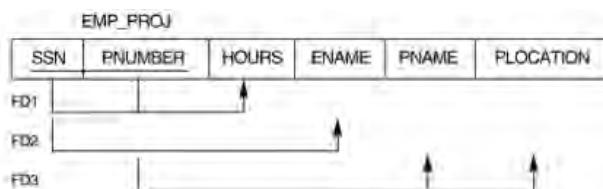
การขึ้นต่อ กันแบบทั้งหมด (full functional dependency) หมายถึง เมื่อให้  
 $FD Y \rightarrow Z$  และ จะไม่สามารถลดแอตทริบิวต์จากเซตของแอตทริบิวต์  $Y$   
แล้ว  $FD$  นั้นยังคงถูกต้องหรือทรง (hold) อยู่ เช่น

$$FD = \{\{SSN, PNUMBER\} \rightarrow HOURS\}$$

เป็นการขึ้นต่อ กันแบบทั้งหมด เนื่องจากไม่สามารถลด SSN หรือ PNUMBER ออกได้ หมายความว่าหากใช้เพียงตัวใดตัวหนึ่ง เป็นตัวกำหนด รีเลชันนี้ก็จะ ทรงไม่ได้ เพราะตัวใดตัวหนึ่งไม่สามารถระบุค่า HOURS ได้

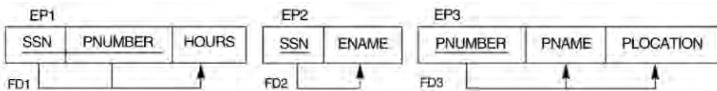
แต่ถ้า  $FD = \{\{SSN, PNUMBER\} \rightarrow ENAME\}$  นี้จะไม่ใช่แบบขึ้นต่อ กัน ทั้งหมด เพราะ ENAME ขึ้นต่อ SSN เท่านั้น เราจึงสามารถลด PNUMBER ออก แล้ว FD ยังคงถูกต้องได้

ตัวอย่าง 2NF แสดงในรูปที่ 72 จะเห็นว่า จาก FD1 นั้น HOURS ขึ้นต่อ  $\{SSN, PNUMBER\}$  จึงขึ้นต่อคีย์หลักทั้งหมด เพราะคีย์หลักคือ  $\{SSN, PNUMBER\}$  แต่ FD2 ไม่ผ่าน 2NF เพราะ ENAME ขึ้นต่อเพียง SSN ซึ่งเป็น เพียงส่วนหนึ่งของคีย์ (partial key) และ FD3 ก็ไม่ผ่าน 2NF ด้วยเหตุผลเดียวกัน คือ PNAME, PLOCATION ขึ้นต่อ PNUMBER ซึ่งเป็นส่วนหนึ่งของ คีย์หลักเข่นกัน



รูปที่ 72 FD ของ EMP\_PROJ

การแก้ไขคือให้ย่อไปเป็น 3 รีเลชัน โดยแยก partial key ออกไปทั้ง FD2 และ FD3 ดังแสดงในรูปที่ 73



รูปที่ 73 EMP\_PROJ ย่ออย่างไรให้อยู่ในรูปแบบ 2NF

### 6.7.3 นอร์มัลฟอร์มที่ 3

พิจารณารูปที่ 74 จะเห็นว่ารหัสประจำตัวของพนักงาน (EMPLOYEE) กับ DEPARTMENT ซึ่งจะเกิดความซ้ำซ้อนของข้อมูล คือข้อมูลในคอลัมน์ DNAME และ DMGRSSN ซ้ำกันหลายทุกเพิล

รหัสประจำตัวของพนักงาน (EMPLOYEE) กับ DEPARTMENT ผ่าน 2NF เพราะมีคีย์เดียว คือ SSN และไม่มีคีย์คู่แข่ง นั่นหมายความว่าแอตทริบิวต์ที่ไม่ใช่พรอมขึ้นต่อ SSN ทั้งหมด อย่างไร ก็ต้องมีคีย์เดียวที่ไม่ใช่ SSN ที่ไม่จำเป็น เรายังคงอยู่处在 2NF แต่ต้องแก้ไขโดยการนำคุณสมบัติที่ซ้ำซ้อนของ DNAME และ DMGRSSN ไปอยู่ในตารางเดียวกัน

EMP_DEPT							ช้าซ้อน	
ENAME	SSN	BDATE	ADDRESS	DNUMBER	DNAME	DMGRSSN		
Smith,John B.	123456789	1965-01-09	731 Fondren,Houston,TX	5	Research	33345555		
Wong,Franklin L.	333445555	1955-12-08	638 Voss,Houston,TX	5	Research	333445555		
Zeloya,Alicia J.	999887777	1968-07-19	3321 Castle,Spring,TX	4	Administration	987654321		
Walcox,Jennifer S.	987654321	1941-06-20	291 Berry,Bellaire,TX	4	Administration	987654321		
Narayan,Ramesh K.	666884444	1962-09-15	975 FireOak,Humble,TX	5	Research	33345555		
English,Joyce A.	453453453	1972-07-31	5631 Rice,Houston,TX	5	Research	33345555		
Jabbar,Ahmed V.	987987987	1969-03-29	980 Dallas,Houston,TX	4	Administration	987654321		
Borg,James E.	888665555	1937-11-10	450 Stone,Houston,TX	1	Headquarters	888665555		

รูปที่ 74 รหัสประจำตัว EMP\_DEPT

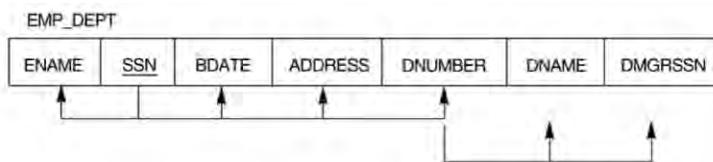
นอร์มัลฟอร์มที่ 3 (Third Normal Form) หรือ 3NF ระบุว่า รีเลชันใด ๆ จะอยู่ใน 3NF ได้นั้น จะต้องผ่าน 2NF และต้องไม่มีข้อต่อทุกคีย์แบบส่งผ่าน (transitive dependency) ซึ่งรวมถึงคีย์คู่แข่งด้วย

ความขึ้นต่อ กันแบบส่งผ่าน (Transitive dependency) จะเกิดขึ้นเมื่อ FD  
 $X \rightarrow Z$  มาจาก  $\{X \rightarrow Y \text{ และ } Y \rightarrow Z\}$

หรืออธิบาย 3NF ได้อีกแบบว่า 3NF กำหนดให้รีเลชันเป็น 3NF เมื่อ FD  
 $X \rightarrow A$  ทรงในรีเลชัน R และ

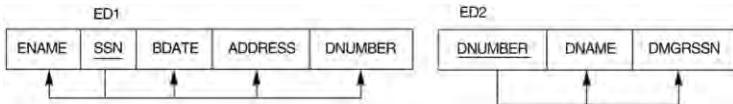
1. X เป็นคีย์ของ R หรือ
2. A เป็นไพร์เมตทริบิวต์

จากตัวอย่างจะเห็นว่า DNAME และ DMGRSSN ขึ้นต่อ DNUMBER และ DNUMBER ขึ้นต่อ SSN อีกที หรือมองกลับกันจะเห็นว่า SSN กำหนด DNUMBER และ DNUMBER กำหนด DNAME และ DMGRSSN ดังแสดงในรูปที่ 75 ซึ่งทำให้ไม่ผ่าน 3NF



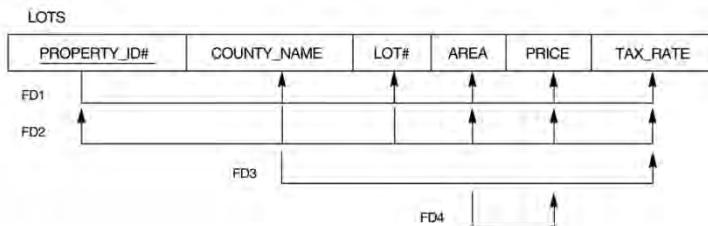
รูปที่ 75 FD ของ EMP\_DEPT

เมื่อกีดเหตุการณ์เช่นนี้ ให้ถอยตารางดังตัวอย่างในรูปที่ 76



รูปที่ 76 EMP\_PROJ ถอยให้อยู่ในรูปแบบ 3NF

ตัวอย่างการแปลงจาก 1NF ไป 2NF และ 3NF



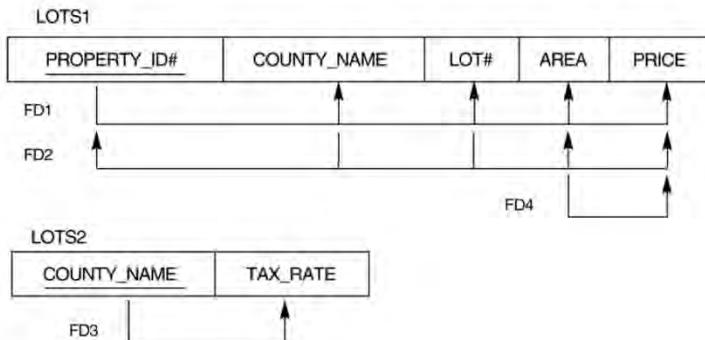
รูปที่ 77 รีเลชัน LOTS กับ 4 FD

จากรูปที่ 77 พบร่วม FD ที่ส่งผลให้รีเลชันตั้งต้นทืออยู่ในรูปแบบ 1NF แต่ไม่ผ่าน 2NF คือ FD3 เพราะเราสามารถตีความได้ว่ารีเลชันนี้มีคีย์คู่ เช่น 2 ชุด ได้แก่

- PROPERTY\_ID# จากขีดเส้นใต้ และที่ FD1 ระบุว่า PROPERTY\_ID# สามารถกำหนดแอ็ตทริบิวต์ในรีเลชันได้ทุกตัว
- {COUNTY\_NAME, LOT#} แม้จะไม่ได้ขีดเส้นใต้ระบุว่าเป็นคีย์หลัก แต่จาก FD2 ที่ระบุว่า {COUNTY\_NAME, LOT#} สามารถ

กำหนดแอ็ตทริบิวต์ในรีเลชันได้ทุกตัวนั้น ซึ่งให้เห็นว่า  
 $\{COUNTY\_NAME, LOT\# \}$  เป็นคีย์คู่แข่ง

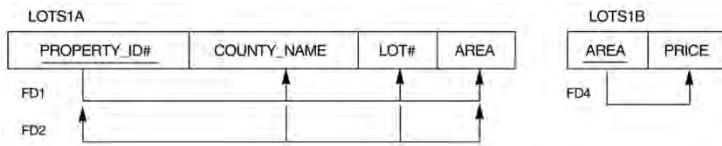
การแก้ไขกรณีนี้ ให้ย่อรีเลชัน โดยแยก FD3 ออกมาเป็น LOTS2  
 $(COUNTY\_NAME, TAX\_RATE)$  ดังแสดงใน รูปที่ 78



รูปที่ 78 รีเลชัน LOTS1 และ LOTS2 ในรูปแบบ 2NF

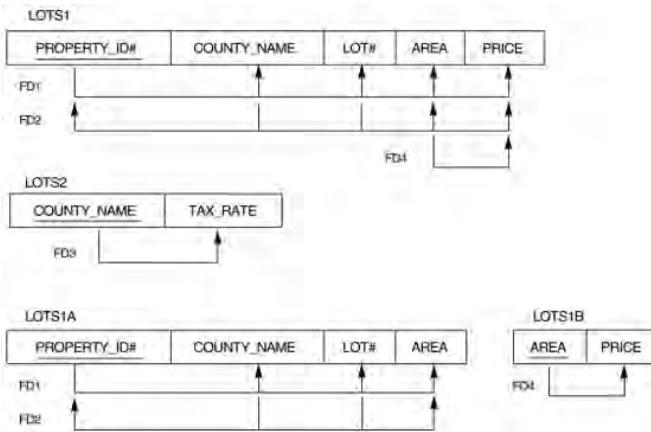
ส่วน FD ที่ส่งผลให้รีเลชันที่ย่ออยแล้วไม่ผ่าน 3NF คือ FD4 เพราะ PRICE ขึ้นต่อ AREA และ AREA ขึ้นต่อคีย์ คือ PROPERTY\_ID# อีกต่อหนึ่ง จึงเกิดการขึ้นตอกันแบบส่งผ่าน

การแก้ไขในกรณีนี้ ให้ย่อรีเลชัน โดยแยก FD4 ออกมาอีกรีเลชัน คือ LOTS1B(AREA, PRICE) ดังแสดงในรูปที่ 79



รูปที่ 79 ย่อรีเลชัน LOTS1 เป็น LOTS1A และ LOTS1B ในรูปแบบ 3NF

สรุปว่ารีเลชันทั้งหมดที่อยู่ในรูปแบบ 3NF แสดงในรูปที่ 80

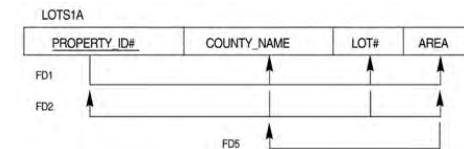


รูปที่ 80 รีเลชัน LOTS ย่อให้อยู่ในรูปแบบ 3NF

#### 6.7.4 นอร์มลฟอร์มบอยซ์คอดด์

นอร์มลฟอร์มบอยซ์คอดด์ (Boyce-Codd Normal Form) นิยมเรียก  
แบบอ้วว่า บีซีเอ็นเอฟ (BCNF) กำหนดว่ารีเลชันต้องอยู่ใน 3NF และเมื่อ FD  
 $X \rightarrow A$  ทรงในรีเลชัน R และ X ต้องเป็นชุดเปอร์คิร์ของ R เท่านั้น

เมื่อพิจารณารูปที่ 81 จะพบว่า FD5 ส่งผลให้ไม่ผ่านบีชีเอ็นเอฟ เพราะ AREA → COUNTY\_NAME และ AREA ไม่ใช่ชูเปอร์คีย์ของรีเลชันนี้



รูปที่ 81 รีเลชัน LOTS1A ที่ไม่ผ่านบีชีเอ็นเอฟ

วิธีการแก้คือให้ย่อรีเลชันโดยยึดแอตทริบิวต์ที่เป็นไฟร์มและไม่ใช่ชูเปอร์คีย์นั้นเป็นหลัก ให้มีอยู่ในรีเลชันอย่างทั้งสอง ดังแสดงในรูปที่ 82

LOTS1AX			LOTS1AY	
PROPERTY_ID#	AREA	LOT#	AREA	COUNTY_NAME

รูปที่ 82 รีเลชัน LOTS1AX และ LOTS1AY ในรูปแบบ BCNF

การแยกย่อยรีเลชันให้เป็นไปตามที่บีชีเอ็นเอฟกำหนดนั้น จำเป็นต้องยึด แอตทริบิวต์ที่เป็นไฟร์มและไม่ใช่ชูเปอร์คีย์เป็นหลัก นั่นเป็นเพราะการแยก เพียงแบบนี้เท่านั้นที่จะทำให้การเชื่อมรีเลชันคืนแล้วจะไม่เกิดปัญหาข้อมูล เกินหรือขาด อย่างไรก็ได้ เพื่อย่อรีเลชันในกรณีนี้แล้ว จะทำให้ไม่สามารถคง สภาพ FD ไว้ได้

ตัวอย่างรีเลชันในรูปที่ 83 กำหนด

FD1: {student, course} → instructor

FD2: instructor → course

เป็นรีเลชันที่ไม่ผ่านบีชีอีเน็นເອັບ ເພຣະຈາກ FD1 ອາຈນອກໄດ້ວ່າ {student, course} ເປັນຄີ່ຍ ຈຶ່ງເປັນໄພຮົມ ແຕ່ເມື່ອພບວ່າ instructor ຍັ້ນໄປກຳທັດໄພຮົມ ຈຶ່ງໃຫ້ໄໝ່ມີຜ່ານບື້ອື່ນເອັບ

STUDENT	COURSE	INSTRUCTOR
Kailerk	Database	Proadpran
Kawin	Database	Chotirat
Pranee	Operating Systems	Chonlatorn
Sanchai	Theory	Athasith
Somsak	Database	Wiwat
Supachart	Operating Systems	Chonlatorn
Thanyanuch	Database	Tarathip

ຮູບທີ 83 ຮືເລຂັນ TEACH

ເຮົາຈາຍຢ່ອຍຮືເລຂັນໄດ້ 3 ຄູ້ຄື່ອ

- {student, instructor} ແລະ {student, course}
- {course, instructor} ແລະ {course, student}
- {instructor, course} ແລະ {instructor, student}

ໃຫ້ສັງເກຕວ່າ ໄນວ່າຈະຢ່ອຍແບບຄູ່ໄດ້ກີ່ຈະສ່ວນໃຫ້ FD1 ນັ້ນໄໝຄົງສກາພແລ້ວ ຈຶ່ງ  
ບື້ອື່ນເອັບຍອນໃໝ່ເກີດໄດ້

อย่างไรก็ดี มีเพียงแค่ {instructor, course} และ {instructor, student} เท่านั้น ที่การเขียนตารางคืนแล้วจะไม่เกิดปัญหาข้อมูลเกิน

## 6.8 บทสรุปและเรื่องชวนคิด

หัวใจสำคัญของบทนี้คือการนำไลบรารีเหล่านี้เพื่อลดความซ้ำซ้อนของข้อมูล ซึ่งจะส่งผลให้ลดโอกาสเกิดความผิดพลาดเมื่อแก้ไขข้อมูลได้ แต่ก็ยังมีอีกหลายปัจจัยที่ต้องคำนึงถึง เช่น ปัญหาค่าว่าง (null value) ว่าควรย่อโยตาร่างโดยพิจารณาให้มีการเก็บค่าว่างอย่างเหมาะสมสมอีกด้วย เพราะค่าว่างอาจทำให้เพิ่มพื้นที่จัดเก็บและทำให้ลดประสิทธิภาพของฐานข้อมูลโดยไม่จำเป็น หากออกแบบได้ไม่ดี ในประเด็นนี้ ขอให้ผู้เรียนลองคิดต่อว่าหากจากค่าว่างแล้ว จะมีปัจจัยอะไรอีกบ้าง ที่อาจส่งผลให้ประสิทธิภาพของฐานข้อมูลลดลงได้

นอกเหนือจากนั้นเราได้เรียนเรื่องนอร์มัลฟอร์มและขั้นตอนการย่อโยตัวให้อยู่ในรูปแบบที่แต่ละนอร์มัลฟอร์มระบุ อย่างไรก็ดี ยังมีนอร์มัลฟอร์มอีกหลายรูปแบบที่ไม่ได้ครอบคลุมในตำนานี้ หากผู้เรียนมีโอกาส อยากรู้สังเกต และเรียนรู้ว่าองค์กรต่าง ๆ ได้กำหนดระดับนอร์มัลฟอร์มของฐานข้อมูลของเข้าที่ระดับใด และหากสนใจเรื่องนี้ ก็อาจศึกษานอนร์มัลฟอร์มรูปแบบอื่น ๆ ต่อไปด้วย

## แบบฝึกหัดท้ายบท

1. เอฟโคลเซอร์และเอกซ์โคลเซอร์คืออะไร และทำอย่างไรจึงจะหาส่องสีงันได้
2. เหตุใดเราจึงต้องคำนวณหาเอฟโคลเซอร์และเอกซ์โคลเซอร์
3. สัจพจน์ของอาร์มสตรองมีกฎพื้นฐานกี่ข้อ อะไรบ้าง
4. เพราะเหตุใดการคำนวณหาเอฟโคลเซอร์จึงเป็นเรื่องที่สิ้นเปลืองหากจะต้องหาทุก f
5. ขั้นตอนการหาค่าคงที่และค่าเบี่ยงเบนมาตรฐานใดๆ ก็ได้ในฟังก์ชันนัลตี้เพนเดนซี่ เพราะเหตุใด
6. ฐานข้อมูลที่ดีควรมีลักษณะอย่างไร
7. การน้อมร์ม่าໄලซ์คืออะไร และเหตุใดเราจึงควรทำการน้อมร์ม่าໄලซ์
8. เป้าหมายสำคัญที่ต้องคำนึงถึงเมื่อจะทำการน้อมร์ม่าໄලซ์รีเลชันมีกี่เป้าหมาย อะไรบ้าง จงอธิบาย
9. ไฟร์ม์แอตทริบิวต์คืออะไร
10. “การน้อมร์ม่าໄලซ์เป็นน้อมร์มัลฟอร์มทุกแบบ จะสามารถคงฟังก์ชันนัลตี้เพนเดนซ์ไว้ได้ครบเสมอ” จากข้อความข้างต้น ผู้เรียนเห็นด้วยหรือไม่ จงอภิปราย

## บทที่ 7

### พีชคณิตเชิงสัมพันธ์และแคลคูลัสเชิงสัมพันธ์

#### วัตถุประสงค์

- เพื่อให้เข้าใจการสอบตามฐานข้อมูลเชิงสัมพันธ์ด้วยพีชคณิตเชิงสัมพันธ์
- เพื่อศึกษาโอเปอเรเตอร์พีชคณิตเชิงสัมพันธ์
- เพื่อให้เข้าใจความแตกต่างระหว่างพีชคณิตเชิงสัมพันธ์กับแคลคูลัสเชิงสัมพันธ์

การสอบถาม (query) ฐานข้อมูลเชิงสัมพันธ์อาศัยภาษาทางคณิตศาสตร์ (mathematical query language) สองภาษาคือ พีชคณิตเชิงสัมพันธ์และ แคลคูลัสเชิงสัมพันธ์ [3, 13, 43]

พีชคณิตเชิงสัมพันธ์มีลักษณะการทำงานแบบปฏิบัติการ (operational) คือ แปลคำสั่งแล้วทำงานทีละคำสั่ง ทำงานทีละขั้นตอน เน้นการสอบถามข้อมูล ในรีเลชัน

ส่วนแคลคูลัสเชิงสัมพันธ์นั้นเป็นเชิงพรรรณา (declarative) ผู้เขียนคำสั่งโดยใช้แคลคูลัสเชิงสัมพันธ์มาระบุนิยามของผลลัพธ์ โดยไม่ต้องคำนึงถึงลำดับการทำงานของคำสั่งนั้น

กฎของคอตต์ (Codd's theorem) กล่าวว่า มินิพจน์พีชคณิตเชิงสัมพันธ์และ นิพจน์แคลคูลัสเชิงสัมพันธ์ ที่ให้ผลลัพธ์เดียวกันได้เสมอ (logically equivalent)

## 7.1 หลักการของพีชคณิตเชิงสัมพันธ์

พีชคณิตเชิงสัมพันธ์มีรากฐานมาจากทฤษฎีเซต (set theory) การดำเนินการ ได้ ๆ กับข้อมูลในฐานข้อมูล หากใช้พีชคณิตเชิงสัมพันธ์ (relational algebra) แล้ว ผลที่ได้จากการทำโดยเปลี่ยนพีชคณิต (algebra operation) หรือคำสั่งพีชคณิตนั้นจะทำให้เกิดรีเลชันใหม่ อาจเกิดรีเลชันเดียวยหรือหลาย รีเลชันก็ได้

เขตของໂປໂປເຣຊັບພຶ້ມືນິຕີເຊົາສັມພັນົດ ເຮັດວຽກວ່ານິພຈົນພຶ້ມືນິຕີເຊົາສັມພັນົດ (relational algebra expression) ສິ່ງຜລລັພົກົກົມືອງຮູ້ເລັ້ນທີ່ເປັນຜລຂອງ

การสอบถาม (query) ชุดนั้น ๆ หรืออาจพูดได้ว่า การสอบถามคือโอบอเรชันพีซีณิตที่เรียงต่อกันนั่นเอง

## 7.2 โอบอเรชันเชิงสัมพันธ์

โอบอเรชันเชิงสัมพันธ์ (relational operation) มีโอบอเรเตอร์ (operator) หลายตัว สามารถจัดแบ่งตามการใช้งานได้ดังนี้

โอบอเรชันเชิงสัมพันธ์จากเรียบชันเดียว (unary relational operations) ตัวโอบอเรเตอร์ ได้แก่ SELECT, PROJECT และ RENAME

โอบอเรชันเชิงสัมพันธ์จาก 2 รีเลชัน (binary relational operations) ตัวโอบอเรเตอร์ ได้แก่ JOIN และ DIVISION

โอบอเรชันเชิงสัมพันธ์จากหลายรีเลชัน ( $n$ -ary relation operations) จะนำเอาตัวโอบอเรชันหลายตัวมาใช้ร่วมกัน

การเขียนคำสั่งโอบอเรชันนั้น ให้ระวังการใช้อักษรตัวเล็กและตัวใหญ่ ในโอบอเรชันพีซีณิตเชิงสัมพันธ์เราจะใช้อักษรตัวเล็ก เช่น

$r - s$

หากเมื่อใดใช้อักษรตัวใหญ่ ให้เข้าใจว่าเป็นโอบอเรชันบนสคีมา เช่น

$R - S$

## 7.2.1 SELECT

คำสั่งหรือโอเปอเรชันที่ใช้มีรีเลชันเดียว ได้แก่ โอเปอเรเตอร์ SELECT ซึ่งมีหน้าที่เลือกชั้บเซตของทุกเพลจารีเลชันตามเงื่อนไขที่ต้องการ สามารถเขียนได้ในรูปแบบนี้

$$\sigma_p(r)$$

สัญลักษณ์  $\sigma$  อ่านว่า ซิกมา (sigma) เป็นสัญลักษณ์ที่ใช้แทนคำสั่ง SELECT

$p$  คือเงื่อนไขที่ต้องการ เขียนด้วยสมการแคลคูลัสที่เขียนต่อแต่ละเทอมด้วย  $\wedge$  (and),  $\vee$  (or),  $\neg$  (not)

แต่ละเทอมเขียนได้ดังนี้

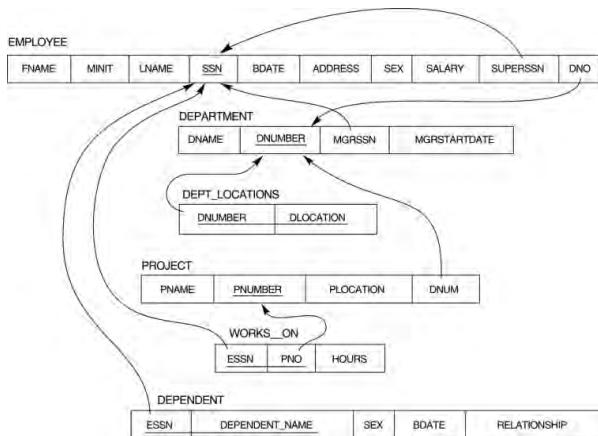
`<attribute> op <attribute> or <constant>`

โดย  $op$  อาจเป็น  $=, \neq, >, \geq, <, \leq$

$r$  เป็นชื่อของรีเลชันที่ต้องการดำเนินการ จะได้ผลลัพธ์

$$\sigma_p(r) = \{t \mid t \in r \text{ and } p(t)\}$$

## ตัวอย่าง



รูปที่ 84 รีเลชันบริษัทดีบี

จากรีเลชันบริษัทดีบีในรูปที่ 84 เราต้องการสอบถามหา EMPLOYEE ที่อยู่ DEPARTMENT No 4 และ EMPLOYEE ที่มีเงินเดือนมากกว่า 30,000

$\sigma_{DNO=4}(\text{employee})$

$\sigma_{\text{SALARY}>30000}(\text{employee})$

## ตัวอย่าง

$\sigma_{(DNO=4 \wedge SALARY>25000) \vee (DNO=5 \wedge SALARY>30000)}(\text{employee})$

หมายถึง ต้องการเลือกทุกเพลของรีเลชัน EMPLOYEE เฉพาะรายการที่ DNO = 5 และเงินเดือนมากกว่า 30,000 หรือ DNO = 4 และเงินเดือนมากกว่า 25,000

จะได้ผลลัพธ์ดังแสดงในรูปที่ 85

FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
Franklin	T	Wong	3334455555	1955-12-08	638 Voss,Houston,TX	M	40000	8888655555	5
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry,Bellaire,TX	F	43000	8888655555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 FireOak,Humble,TX	M	38000	3334455555	5

รูปที่ 85 ผลลัพธ์การ SELECT

คุณสมบัติของ SELECT มีดังนี้

โอเปอเรเตอร์ SELECT นั้น ผลที่ได้สคีมาเหมือนกับรีเลชันตั้งต้น คือ

$\sigma_p(r)$  จะได้สคีมาเดียวกันกับ  $r$

โอเปอเรเตอร์ SELECT มีคุณสมบัติการเรียงสับเปลี่ยน (commutative) หมายความว่าการเรียงต่างแบบไม่ส่งผลให้ผลลัพธ์ต่างไป

$$\sigma_{p1}(\sigma_{p2}(r)) = \sigma_{p2}(\sigma_{p1}(r))$$

## 7.2.2 PROJECT

ໂອເປົອເຮືອໂຕ PROJECT ເປັນໂອເປົອເຮັດໜີທີ່ໃຊ້ເມື່ອມີເລີກເລີຍໄວ້ ໃຫ້ເລືອກ  
ຂໍ້ມູນລາຍເນະຄອລົມນ໌ທີ່ຕ້ອງການຈາກທາງໜີ້ວິເລັ້ນນັ້ນ

$$\pi_{\langle \text{attribute list} \rangle}(r)$$

ສັນລັກຊົນ  $\pi$  ອ່ານວ່າ ພາຍ (pi) ເປັນສັນລັກຊົນທີ່ໃຊ້ແທນຄຳສັ່ງ PROJECT

attribute list ຄື່ອແອຕທຣີບົວຕ່າງໆ ຂີ່ອຄອລົມທີ່ຕ້ອງການເລືອກ

R ເປັນຊື່ຂອງວິເລັ້ນທີ່ຕ້ອງການດຳເນີນການ

ຜລລັພົງຈາກການ PROJECT ຄື່ອຖຸເພີ້ມທັງໝົດ ມາກພລກາຣ PROJECT ທຳໄໝວິ  
ຖຸເພີ້ມໜ້າ ກີ່ຈະລັດຖຸເພີ້ມທີ່ຈໍາກັນອອກ ເພຣະວິເລັ້ນຄື່ອເຊີຕອງຖຸເພີ້ມນັ້ນເອງ  
ດັ່ງນັ້ນຈຳນວນຖຸເພີ້ມທີ່ໄດ້ຈະນອຍກວ່າຫີ່ອເທົກກັບວິເລັ້ນຕັ້ງຕັ້ນເສມອ

ຕ້ວອຍ່າງ

$$\pi_{\text{LNAME,FNAME,SALARY}}(\text{employee})$$

ໜາຍເລື່ອງ ຕ້ອງການເລືອກຂໍ້ມູນທຸກຖຸເພີ້ມຈາກຄອລົມນ໌ຫີ່ອແອຕທຣີບົວຕ່າງໆ  
LNAME,FNAME ແລະ SALARY ຈາກວິເລັ້ນ EMPLOYEE

ຜລລັພົງແສດງໃນຮູບທີ່ 86

LNAME	FNAME	SALARY
Smith	John	30000
Wong	Franklin	40000
Zelaya	Alicia	25000
Wallace	Jennifer	43000
Narayan	Ramesh	38000
English	Joyce	25000
Jabbar	Ahmad	25000
Borg	James	55000

รูปที่ 86 ผลลัพธ์การ PROJECT LNAME, FNAME, SALARY

ตัวอย่าง

$$\pi_{\text{SEX}, \text{SALARY}}(\text{employee})$$

ผลลัพธ์แสดงในรูปที่ 87

SEX	SALARY
M	30000
M	40000
F	25000
F	43000
M	38000
M	25000
M	55000

รูปที่ 87 ผลลัพธ์การ PROJECT SEX, SALARY

### 7.2.3 RENAME

บางครั้งเราจำเป็นที่จะต้องนำผลลัพธ์จากโอลีเพอเรชันไปหาดำเนินการต่อ เราอาจสร้างรีเลชันระหว่างทางเพื่อกีบผลลัพธ์จากโอลีเพอเรชันหนึ่งไปใช้อ้างอิง

ต่อได้ โดยการให้ชื่อหรือเปลี่ยนชื่อรีเลขันนั้นด้วยโอบอเรเตอร์ RENAME นอกจากนั้น คำสั่ง RENAME นี้ยังสามารถใช้เปลี่ยนชื่อแอตทริบิวต์ได้ด้วยรูปแบบการเขียนมีดังนี้

$\rho (_{x(olddname1 \rightarrow newname1 \text{ or } position \rightarrow newname1, \dots), e})$

หรือ

$\rho (_{x(newname1, newname2, \dots, newnamen), e})$

สัญลักษณ์  $\rho$  อ่านว่า โรห์ (rho) ใช้แสดงคำสั่ง RENAME

e คือ expression ได ๆ

oldname คือ ชื่อเดิมของแอตทริบิวต์

newname คือ ชื่อใหม่ของแอตทริบิวต์

X คือ ชื่อใหม่ของรีเลขันผลลัพธ์

ในกรณีที่ไม่ได้ระบุ oldname นั้น newname จะแทนชื่อของแอตทริบิวต์ตามตำแหน่งที่เขียนในโอบอเรชัน RENAME

ตัวอย่าง

$\rho (_{\text{myRelation}(A \rightarrow E, 2 \rightarrow K), r - s})$

จากคำสั่งข้างบน จะมีการลบ r เลขัน r ด้วย s และวนำผลลัพธ์ไปเก็บไว้ในตารางหรือรีเลชันชื่อ myRelation และยังมีการเปลี่ยนชื่อแอตทริบิวต์ A เป็น E และเปลี่ยนชื่อแอตทริบิวต์ตำแหน่งที่ 2 เป็น K

อีกวิธีที่ใช้เปลี่ยนชื่อรีเลชัน คือการใช้อเปอเรเตอร์ ASSIGNMENT แทนด้วยลูกศรซ้าย  $\leftarrow$  เช่น

$$\text{newRelation} \leftarrow \sigma_{\text{DNO}=5}(\text{employee})$$
$$\text{result} \leftarrow \pi_{\text{LNAME}, \text{FNAME}, \text{SALARY}}(\text{newRelation})$$

หมายความว่า ให้ SELECT ทุกเพลที่ DNO = 5 จากรีเลชัน EMPLOYEE ไปเก็บไว้ในรีเลชันใหม่ ซึ่ว่า NEWRELATION จากนั้นก็อ้างอิงชื่อรีเลชันใหม่นั้นได้เลย

#### 7.2.4 UNION

โอเปอเรชันยูเนียน (UNION) เป็นการดำเนินการในโอเปอเรชันเชิงสัมพันธ์ จาก 2 รีเลชัน (binary relational operations) เพื่อเข้ามาร่าง 2 ตาราง ด้วยการนำทุกเพลจาก 2 ตารางมารวมกัน มีรูปแบบการเขียนคือ

$$r \cup s$$

สัญลักษณ์  $\cup$  อ่านว่า ยูเนียน (union)

R และ S คือชื่อรีเลชัน

## ผลลัพธ์จากการรูปเนียนคือ

$$r \cup s = \{t \mid t \in r \text{ or } t \in s\}$$

ทั้งนี้ การรูปเนียนบังคับว่า 2 ตารางที่นำมา.rูปเนียนกันต้องมีไฟล์เดียวกัน (type compatibility) นั่นหมายความว่าทั้งสองตารางมีจำนวนแอตทริบิวต์และโดเมน (domain) เดียวกัน และเรียงแอตทริบิวต์เหมือนกันด้วย แต่ไม่จำเป็นต้องมีชื่อแอตทริบิวต์เหมือนกัน ดังนั้นรีเลชันที่เป็นผลลัพธ์ของ  $r_1 \cup r_2$  นั้นจะมีชื่อแอตทริบิวต์ตามรีเลชันตัวแรก คือ  $r_1$

การรูปเนียนจะมีการสร้างรีเลชันใหม่ โดยจะนำทุกเพลิกของ R และทุกเพลิกของ S มารวมกัน ถ้ามีทุกเพลิกที่ซ้ำกันก็จะตัดทุกเพลิกนั้นออก

ตัวอย่าง

RESULT  $\leftarrow$  result1  $\cup$  result2

RESULT1	SSN
123456789	
333445555	
666884444	
453453453	

RESULT2	SSN
333445555	
8886665555	

RESULT	SSN
123456789	
333445555	
666884444	
453453453	
8886665555	

รูปที่ 88 โอเปอเรชันยูเนียน

จะเห็นได้ว่า ค่า 333445555 มีอยู่ทั้งในรีเลชัน RESULT1 และ RESULT2 เมื่อ UNION และจะตัดทุกเพลิกซ้ำนั้นออก

## 7.2.5 INTERSECTION

โอบอเรชันอินเตอร์เซกชัน เป็นการดำเนินการในโอบอเรชันเชิงสัมพันธ์จาก 2 รีเลชัน (binary relational operations) เพื่อหาส่วนร่วมของข้อมูลจากตาราง 2 ตาราง มีรูปแบบการเขียนคือ

$$r \cap s$$

สัญลักษณ์  $\cap$  อ่านว่า อินเตอร์เซก (intersect)

ได้ผลลัพธ์

$$r \cap s = \{t \mid t \in r \text{ and } t \in s\}$$

ผลลัพธ์จากโอบอเรชันอินเตอร์เซกชันจะได้รีเลชันใหม่ โดยจะนำทุกเพลที่อยู่ในรีเลชัน R และรีเลชัน S มาเก็บไว้ในตารางผลลัพธ์

การอินเตอร์เซก มีข้อบังคับเหมือนกับการยูเนียน คือ 2 ตารางที่นำมาอินเตอร์เซกกันต้องมีไปเปดယกัน นั่นหมายความว่าทั้งสองตารางมีจำนวนแอตทริบิวต์และโดเมนเดียวกัน เรียงเหมือนกัน แต่ไม่จำเป็นต้องมีชื่อแอตทริบิวต์เหมือนกัน

## 7.2.6 MINUS

โอบอเรชันไม่นัส หรือค่าต่างของเซต (set difference) คือการลบของรีเลชัน 2 รีเลชัน จึงเป็นโอบอเรเตอร์ที่ใช้ในโอบอเรชันเชิงสัมพันธ์จาก

2 รีเลชัน (binary relational operations) เพื่อหาส่วนต่างของตาราง  
2 ตาราง มีรูปแบบการเขียนคือ

$r - s$

สัญลักษณ์ – อ่านว่า ไม่นัส (minus)

ได้ผลลัพธ์

$$r - s = \{t \mid t \in r \text{ and } t \notin s\}$$

ผลลัพธ์จากการลบนี้จะได้รีเลชันใหม่ ที่โดยจะนำทุกเพิลผลต่างระหว่างตัวทั้ง R และตัวลบ S มาสร้างรีเลชันใหม่ ซึ่งถ้าทุกเพิลนั้นอยู่เฉพาะใน S อย่างเดียว หรือทุกเพิลนั้นอยู่ทั้งใน R และ S ให้ตัดทิ้ง

การลบนี้มีข้อบังคับเหมือนกับการยuniонและอินเตอร์เซก คือ สองตารางที่นำมาลบกันต้องมีไปเปิดไวกัน นั่นหมายความว่าทั้งสองตารางมีจำนวนแอตทริบิวต์และโดเมนเดียวกัน เรียงเหมือนกัน แต่ไม่จำเป็นต้องมีชื่อแอตทริบิวต์เหมือนกัน

(a)	<table border="1"> <thead> <tr> <th>STUDENT</th> <th>FN</th> <th>LN</th> </tr> </thead> <tbody> <tr> <td>Susan</td> <td>Yao</td> <td></td> </tr> <tr> <td>Ramesh</td> <td>Shah</td> <td></td> </tr> <tr> <td>Johnny</td> <td>Kohler</td> <td></td> </tr> <tr> <td>Barbara</td> <td>Jones</td> <td></td> </tr> <tr> <td>Amy</td> <td>Ford</td> <td></td> </tr> <tr> <td>Jimmy</td> <td>Wang</td> <td></td> </tr> <tr> <td>Ernest</td> <td>Gilbert</td> <td></td> </tr> </tbody> </table>	STUDENT	FN	LN	Susan	Yao		Ramesh	Shah		Johnny	Kohler		Barbara	Jones		Amy	Ford		Jimmy	Wang		Ernest	Gilbert		<table border="1"> <thead> <tr> <th>INSTRUCTOR</th> <th>FNAME</th> <th>LNAME</th> </tr> </thead> <tbody> <tr> <td>John</td> <td>Smith</td> <td></td> </tr> <tr> <td>Ricardo</td> <td>Browne</td> <td></td> </tr> <tr> <td>Susan</td> <td>Yao</td> <td></td> </tr> <tr> <td>Francis</td> <td>Johnson</td> <td></td> </tr> <tr> <td>Ramesh</td> <td>Shah</td> <td></td> </tr> </tbody> </table>	INSTRUCTOR	FNAME	LNAME	John	Smith		Ricardo	Browne		Susan	Yao		Francis	Johnson		Ramesh	Shah		(b)	<table border="1"> <thead> <tr> <th>FN</th> <th>LN</th> </tr> </thead> <tbody> <tr> <td>Susan</td> <td>Yao</td> </tr> <tr> <td>Ramesh</td> <td>Shah</td> </tr> <tr> <td>Johnny</td> <td>Kohler</td> </tr> <tr> <td>Barbara</td> <td>Jones</td> </tr> <tr> <td>Amy</td> <td>Ford</td> </tr> <tr> <td>Jimmy</td> <td>Wang</td> </tr> <tr> <td>Ernest</td> <td>Gilbert</td> </tr> <tr> <td>John</td> <td>Smith</td> </tr> <tr> <td>Ricardo</td> <td>Browne</td> </tr> <tr> <td>Francis</td> <td>Johnson</td> </tr> </tbody> </table>	FN	LN	Susan	Yao	Ramesh	Shah	Johnny	Kohler	Barbara	Jones	Amy	Ford	Jimmy	Wang	Ernest	Gilbert	John	Smith	Ricardo	Browne	Francis	Johnson
STUDENT	FN	LN																																																																		
Susan	Yao																																																																			
Ramesh	Shah																																																																			
Johnny	Kohler																																																																			
Barbara	Jones																																																																			
Amy	Ford																																																																			
Jimmy	Wang																																																																			
Ernest	Gilbert																																																																			
INSTRUCTOR	FNAME	LNAME																																																																		
John	Smith																																																																			
Ricardo	Browne																																																																			
Susan	Yao																																																																			
Francis	Johnson																																																																			
Ramesh	Shah																																																																			
FN	LN																																																																			
Susan	Yao																																																																			
Ramesh	Shah																																																																			
Johnny	Kohler																																																																			
Barbara	Jones																																																																			
Amy	Ford																																																																			
Jimmy	Wang																																																																			
Ernest	Gilbert																																																																			
John	Smith																																																																			
Ricardo	Browne																																																																			
Francis	Johnson																																																																			
(c)	<table border="1"> <thead> <tr> <th>FN</th> <th>LN</th> </tr> </thead> <tbody> <tr> <td>Susan</td> <td>Yao</td> </tr> <tr> <td>Ramesh</td> <td>Shah</td> </tr> </tbody> </table>	FN	LN	Susan	Yao	Ramesh	Shah	(d)	<table border="1"> <thead> <tr> <th>FN</th> <th>LN</th> </tr> </thead> <tbody> <tr> <td>Johnny</td> <td>Kohler</td> </tr> <tr> <td>Barbara</td> <td>Jones</td> </tr> <tr> <td>Amy</td> <td>Ford</td> </tr> <tr> <td>Jimmy</td> <td>Wang</td> </tr> <tr> <td>Ernest</td> <td>Gilbert</td> </tr> </tbody> </table>	FN	LN	Johnny	Kohler	Barbara	Jones	Amy	Ford	Jimmy	Wang	Ernest	Gilbert	(e)	<table border="1"> <thead> <tr> <th>FNAME</th> <th>LNAME</th> </tr> </thead> <tbody> <tr> <td>John</td> <td>Smith</td> </tr> <tr> <td>Ricardo</td> <td>Browne</td> </tr> <tr> <td>Francis</td> <td>Johnson</td> </tr> </tbody> </table>	FNAME	LNAME	John	Smith	Ricardo	Browne	Francis	Johnson																																					
FN	LN																																																																			
Susan	Yao																																																																			
Ramesh	Shah																																																																			
FN	LN																																																																			
Johnny	Kohler																																																																			
Barbara	Jones																																																																			
Amy	Ford																																																																			
Jimmy	Wang																																																																			
Ernest	Gilbert																																																																			
FNAME	LNAME																																																																			
John	Smith																																																																			
Ricardo	Browne																																																																			
Francis	Johnson																																																																			

รูปที่ 89 การ UNION INTERSECTION และ MINUS

### รูปที่ 89 แสดงการ UNION INTERSECTION และ MINUS

- (a) แสดงรีเลชัน STUDENT กับ รีเลชัน INSTRUCTOR
- (b) แสดงผลลัพธ์  $\text{student} \cup \text{instructor}$
- (c) แสดงผลลัพธ์  $\text{student} \cap \text{instructor}$
- (d) แสดงผลลัพธ์  $\text{student} - \text{instructor}$
- (e) แสดงผลลัพธ์  $\text{instructor} - \text{student}$

#### คุณสมบัติเพิ่มเติม

UNION กับ INTERSECTION มีคุณสมบัติ commutative หรือสมบัติ การสลับที่ คือสลับที่ได้ ผลลัพธ์ยังคงเดิม

$$r \cup s = s \cup r$$

$$r \cap s = s \cap r$$

UNION กับ INTERSECTION มีคุณสมบัติ associative หรือสมบัติการเปลี่ยนกลุ่ม คือสามารถทำให้อเปรเช่นกับรีเลชันพร้อมกันก็ได้ และจัดกลุ่มอย่างไรก็ได้ ผลลัพธ์จะเท่าเดิม

$$r \cup (s \cup t) = (r \cup s) \cup t$$

$$r \cap (s \cap t) = (r \cap s) \cap t$$

MINUS ไม่มีคุณสมบัติ commutative ดังนี้

$$r - s \neq s - r$$

ย้ำอีกครั้งว่าการ UNION, INTERSECT และ MINUS นั้น รีเลชันต้องมีให้ปีเดียวกัน คือมีจำนวนและทริบิวต์เท่ากัน แต่ละแอ็ตทริบิวต์มีโดเมนเดียวกัน และเรียงลำดับแอ็ตทริบิวต์เหมือนกัน แต่ซึ่งแอ็ตทริบิวต์ไม่จำเป็นต้องเหมือนกัน สามารถอธิบายด้วยคณิตศาสตร์ว่า

$$R1(A_1, A_2, \dots, A_n)$$

$$R2(B_1, B_2, \dots, B_n)$$

$$\text{dom}(A_i) = \text{dom}(B_i) \text{ for } i=1, 2, \dots, n$$

รีเลชันผลลัพธ์จะมีซึ่งแอ็ตทริบิวต์ตาม R1

### 7.2.7 CARTESIAN

ໂອເປົອເຮັນຄາຣ໌ທີ່ເຊື່ອ CARTESIAN ອີ່ ຄຣອສໂປຣດັກຕໍ່ (cross-product) ຂຶ້ກາຣເຊື່ອມື້ມຸລ 2 ຮີເລັບເຂົາດ້ວຍກັນໂດຍພິຈາລະນາຂໍ້ມູນທຸກຄູໃນຮີເລັບທັງສອງ ໂດຍທີ່ຮີເລັບທັງສອງໄມ່ຈຳເປັນຕ້ອງເປັນໄທປີເດືອກນໍາ ມາຍຄືງໄຟໄໝ ໄມຈຳເປັນຕ້ອງມີຈຳນວນແອຕທຽບຕໍ່ເທົ່ານັ້ນແລະໂດມັນເດືອກນໍາ ເຊີນຮູບແບບໄດ້ດັ່ງນີ້

$$r \times s$$

ผลลัพธ์ของ CARTESIAN จะเกิดรีเลชันใหม่ ประกอบด้วยแอตทริบิวต์จาก R และ S ทุกคู่ของทุกเพลิที่เป็นไปได้ (all possible combinations of rows)

ดังนั้นดีกรี (degree) หรือจำนวนแอตทริบิวต์ของ  $r \times s$  จะเท่ากับ จำนวนแอตทริบิวต์ R + จำนวนแอตทริบิวต์ S

และจำนวนทุกเพลิของ  $r \times s$  เชียนแทนว่า  $|r \times s| =$  จำนวนทุกเพลิใน R  $\times$  จำนวนทุกเพลิใน S

ตัวอย่าง

$$R(A_1, A_2, \dots, A_n)$$

$$S(B_1, B_2, \dots, B_m)$$

$$q \leftarrow r \times s$$

ได้ผลลัพธ์จากการ CARTESIAN ไปเก็บไว้ในรีเลชัน Q เรียกลำดับแอตทริบิวต์ ดังนี้

$$Q(A_1, A_2, \dots, A_n, B_1, B_2, \dots, B_m)$$

ซึ่งมีดีกรี (degree)  $= n + m$  แอตทริบิวต์ และมีจำนวนทุกเพลิ  $n \times m$

อย่างไรก็ได้ การ CARTESIAN เพียงอย่างเดียวจะสร้างรีเลชันที่มีข้อมูลเกินจากเดิม เป็นข้อมูลปลอม (spurious) ที่เกิดจากการคำนวณ แต่เราจำเป็นต้อง CARTESIAN เพื่อเชื่อมรีเลชันที่สกิมมต่างกัน ดังนั้นเราต้องใช้

CARTESIAN และตามด้วย SELECT เพื่อเลือกข้อมูลที่ถูกต้องมาเก็บไว้ ซึ่งเรียกว่า คำสั่ง JOIN

ตัวอย่าง

$$\text{female\_emps} \leftarrow \sigma_{\text{SEX} = 'F'}(\text{employee})$$

$$\text{empnames} \leftarrow \pi_{\text{FNAME}, \text{LNAME}, \text{SSN}} (\text{female\_emps})$$

รูปที่ 90 แสดงผลการ SELECT และ PROJECT เพื่อเตรียมข้อมูลไว้ดำเนินการต่อ

FEMALE_EMPS	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
Alicia	J	Zelaya	999887777	1968-07-19	3321 Castle, Spring, TX	F	25000	987654321	4	
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4	
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5	

EMPNAME	FNAME	LNAME	SSN
Alicia	Zelaya	999887777	
Jennifer	Wallace	987654321	
Joyce	English	453453453	

รูปที่ 90 ผลการ SELECT และ PROJECT

$$\text{emp_dependents} \leftarrow \text{empnames} \times \text{dependent}$$

ได้ผลลัพธ์คือรูปที่ 91

EMP_DEPENDENTS	FNAME	LNAME	SSN	ESSN	DEPENDENT_NAME	SEX	BDATE	• • •
Alicia	Zelaya	999887777	333445555		Alice	F	1986-04-05	• • •
Alicia	Zelaya	999887777	333445555		Theodore	M	1983-10-25	• • •
Alicia	Zelaya	999887777	333445555		Joy	F	1958-05-03	• • •
Alicia	Zelaya	999887777	333445555		Abner	M	1942-02-28	• • •
Alicia	Zelaya	999887777	123456789		Michael	M	1988-01-04	• • •
Alicia	Zelaya	999887777	123456789		Alice	F	1988-12-30	• • •
Alicia	Zelaya	999887777	123456789		Elizabeth	F	1967-05-05	• • •
Jennifer	Wallace	987654321	333445555		Alice	F	1986-04-05	• • •
Jennifer	Wallace	987654321	333445555		Theodore	M	1983-10-25	• • •
Jennifer	Wallace	987654321	333445555		Joy	F	1958-05-03	• • •
Jennifer	Wallace	987654321	987654321		Abner	M	1942-02-28	• • •
Jennifer	Wallace	987654321	123456789		Michael	M	1988-01-04	• • •
Jennifer	Wallace	987654321	123456789		Alice	F	1988-12-30	• • •
Jennifer	Wallace	987654321	123456789		Elizabeth	F	1967-05-05	• • •
Joyce	English	453453453	333445555		Alice	F	1986-04-05	• • •
Joyce	English	453453453	333445555		Theodore	M	1983-10-25	• • •
Joyce	English	453453453	333445555		Joy	F	1958-05-03	• • •
Joyce	English	453453453	987654321		Abner	M	1942-02-28	• • •
Joyce	English	453453453	123456789		Michael	M	1988-01-04	• • •
Joyce	English	453453453	123456789		Alice	F	1988-12-30	• • •
Joyce	English	453453453	123456789		Elizabeth	F	1967-05-05	• • •

รูปที่ 91 ผลการ CARTESIAN

จะเห็นว่า EMP\_DEPENDENTS ซึ่งเป็นผลจากการ CARTESIAN นั้นจะเกิดทุกเพลิลແປລກປລອມເກີນເລຍໄປมากມາຍ ໄນ່ມີຄວາມໝາຍ ນຳພາດຈາກການ CARTESIAN ມາໃຫ້ທັນທຶນໄດ້ ຈຳເປັນຕົ້ນ ຕັ້ງສະເໜີ

ເຊື່ອ ຈາກຕ້ວຍ່າງຂ້າງບນ ເຮົາຕົ້ນ ສະເໜີ SELECT ເພື່ອໃຫ້ໄດ້ມາຊື່ງທຸເພີລທີ່ຄູກຕົ້ນ ນັ້ນ ຄື້ອ ມີ SSN ແລະ ESSN ທີ່ຕຽກກັນ ດັ່ງນີ້

ACTUAL_DEPENDENTS	FNAME	LNAME	SSN	ESSN	DEPENDENT_NAME	SEX	BDATE	• • •
	Jennifer	Wallace	987654321	987654321	Abner	M	1942-02-28	• • •

RESULT	FNAME	LNAME	DEPENDENT_NAME
	Jennifer	Wallace	Abner

รูปที่ 92 CARTESIAN ຕາມດ້ວຍ SELECT

## 7.2.8 DIVISION

การหารรีเลชัน หรือ โอเปอเรชัน DIVISION อาจไม่ค่อยเห็นใช้บ่อยนัก แต่ก็มีประโยชน์อย่างมาก เช่น ต้องการหานักเรียนทุกคนที่ลงวิชา 5 วิชา เป็นต้น

รูปแบบ DIVISION เขียนได้ดังนี้

$$r / s \text{ หรือ } r \div s$$

$$\text{ให้ } R = (A_1, \dots, A_m, B_1, \dots, B_n)$$

$$S = (B_1, \dots, B_n)$$

$$r / s = \{ \langle x \rangle \mid \exists \langle x, y \rangle \in r \wedge \forall \langle y \rangle \in s \wedge x \in \pi_{R-S}(R) \}$$

ผลลัพธ์ของ  $r / s$  จะเป็นรีเลชันที่มีสคีมา  $= R - S = (A_1, \dots, A_m)$

หรืออธิบายได้อีกแบบ ดังนี้

$$\text{ให้ } r(z) / s(x) \text{ โดยที่ } X \text{ เป็นชับเชตของ } Z$$

$$\text{ให้ } y = z - x \text{ (ดังนั้น } z = x \cup y)$$

จะได้ว่า  $y$  คือ เชตของแอตทริบิวต์ในรีเลชันที่เป็นผลลัพธ์ของ  $r / s$

บทที่ 7 พื้นฐานการจัดเริ่มพัฒนาและทดสอบระบบ

### ตัวอย่าง

รูปที่ 93 แสดงขั้นตอนและผลลัพธ์ของ ssns  $\leftarrow$  ssn\_pnos ÷ smith\_pnos

SSN_PNOS	ESSN	PNO	SMITH_PNOS	PNO
123456789	1			1
123456789	2			2
666884444	3			
453453453	1			
453453453	2			
333445555	2			
333445555	3			
333445555	10			
333445555	20			
999887777	30			
999887777	10			
987987987	10			
987987987	30			
987654321	30			
987654321	20			
888665555	20			

SSNS	SSN
123456789	
453453453	

รูปที่ 93 ขั้นตอนและผลลัพธ์ของ DIVISION ssns

รูปที่ 94 แสดงผลลัพธ์ของ t  $\leftarrow$  r ÷ s

R	A	B	S	A	T	B
a1	b1		a1	b1		
a2	b1		a2	b1		
a3	b1		a3	b1		
a4	b1		a4	b1		
a1	b2		a1	b2		
a3	b2		a3	b2		
a2	b3		a2	b3		
a3	b3		a3	b3		
a4	b3		a4	b3		
a1	b4		a1	b4		
a2	b4		a2	b4		
a3	b4		a3	b4		

รูปที่ 94 ขั้นตอนและผลลัพธ์ของ DIVISION R/S

## 7.2.9 JOIN

เมื่อต้องการเชื่อมสองรีเลชันเข้าด้วยกัน ที่ไม่จำเป็นต้องเป็นไฟป์เดียวกัน เรา เชื่อมได้ด้วยการ CARTESIAN แล้วตามด้วย SELECT ซึ่งก็คือโอบอเรชัน JOIN ที่รวมคำสั่ง CARTESIAN ตามด้วย SELECT เข้าไว้ด้วยกันนั่นเอง

กรณีที่รีเลชันทั้งสองฝ่ายมีคอมมอนแอตทริบิวต์หนึ่งตัว หมายถึง มีแอตทริบิวต์ คู่หนึ่งที่มีโดเมนเดียวกัน จะ JOIN โดยทำคำสั่ง CARTESIAN บนสองรีเลชัน แล้ว SELECT ข้อมูลทุกๆเพลทที่แอตทริบิวต์คูนั้นมีค่าเท่ากัน

กรณีที่รีเลชันทั้งสองฝ่ายมีคอมมอนแอตทริบิวต์ ผลลัพธ์ก็คือ CARTESIAN ของสองรีเลชันนั้น

กรณีที่รีเลชันทั้งสองฝ่ายมีคอมมอนแอตทริบิวต์มากกว่าหนึ่งตัว ผลลัพธ์จาก การ JOIN จะ SELECT ทุกๆเพลทที่ทุกคอมมอนแอตทริบิวต์มีค่าเท่ากัน

รูปแบบการ JOIN เขียนได้ดังนี้

$r \bowtie_{<\text{join condition}>} S$

สัญลักษณ์  $\bowtie$  อ่านว่า จอยน์ (join)

การทำงานของ JOIN จะเริ่มด้วยการคราร์ที่เขียนแล้วตามด้วย SELECT ด้วย แอตทริบิวต์อ้างอิงเสมอ

การ JOIN นี้ แบ่งได้หลายแบบ คือ

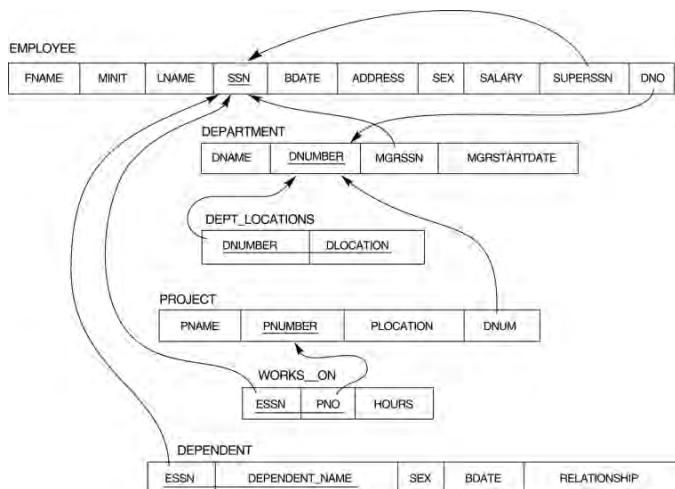
### 7.2.9.1 EQUI JOIN

ໂອເປ່ອເຮັດວຽກ EQUI JOIN ເປັນການເຂື້ອມຮີເລີ້ນນີ້ເມື່ອຢ່າງນ້ອຍໜຶ່ງແວດທຣິບົວຕໍ່  
ຈາກສອງຮີເລີ້ນທີ່ມີຄ່າເທົກນເທົກນ (equality comparison)

ຜລລັບຮົບຂອງ EQUI JOIN ຈະຕ້ອງມີແວດທຣິບົວຕໍ່ຢ່າງນ້ອຍ 1 ຄູ່ຈາກແຕ່ລະຮີເລີ້ນ  
ທີ່ມີຄ່າໃນຫຼຸ່ມເພີ້ມເທົກນທີ່ສອງຕາງໆ

ທັງນີ້ ຂຶ້ອຂອງແວດທຣິບົວຕໍ່ໄມ້ຈຳເປັນຕ້ອງເໝືອນກັນ ແຕ່ຕ້ອງມີໂດມເນັດີຍວກັນ

ຕ້ວຍຢ່າງ



ຮູບທີ 95 ຮີເລີ້ນບຣິຊ້ທີ່ບີ

ຈາກຮູບທີ 95 ສມມູດວ່າເຮົາຕ້ອງການສອບຄາມຂໍ້ອອງ Manager ຂອງແຕ່ລະ  
Department ເຮົາຈໍາເປັນຕ້ອງເຂື້ອມຮີເລີ້ນ DEPARTMENT ກັບຮີເລີ້ນ

EMPLOYEE เพราะ ใน DEPARTMENT มีแต่ MGRSSN ไม่มีชื่อ EMPLOYEE ดังนั้น เราจึงสนใจจะเชื่อมด้วยเงื่อนไขว่า MGRSSN ในรีเลชัน DEPARTMENT มีค่าเท่ากับ SSN ในรีเลชัน EMPLOYEE โดยจะเขียนโฉเปอเรชัน EQUI JOIN ได้ว่า

$\text{dept\_mgr} \leftarrow \text{department} \bowtie_{\text{MGRSSN}=\text{SSN}} \text{employee}$

จะได้ผลลัพธ์ดังนี้

DEPT_MGR	DNAME	DNUMBER	MGRSSN	...	FNAME	MINT	LNAME	SSN	...
Research	5	333445555	...	Proadpran	P	Punyabukkana	333445555	...	
Admin	4	123456789	...	Kawin	T	Meesiruitakul	123456789	...	
HR	1	001001001	...	Sanchai	T	Jakteerangkur	001001001	...	

รูปที่ 96 ผลลัพธ์ของ EQUI JOIN

สังเกตว่าคอลัมน์ MGRSSN และ SSN จะเหมือนกันทุกที่เพิ่ล ซึ่งคือความซ้ำซ้อนที่ฟุ่มเฟือย (superfluous) เกินจำเป็น

#### 7.2.9.2 NATURAL JOIN

เป็นการเชื่อมรีเลชันด้วยแอตทริบิวต์จากสองรีเลชันที่ชื่อเหมือนกันและโดยเม้นเดียวกัน หากชื่อแอตทริบิวต์ที่ใช้เชื่อมไม่เหมือนกัน ให้ RENAME เสียก่อน ผลลัพธ์ของ NATURAL JOIN ต่างกับ EQUI JOIN คือจะตัดแอตทริบิวต์หรือคอลัมน์ซ้ำทิ้งไป ทำให้รีเลชันผลลัพธ์ไม่พบปัญหาข้อมูลฟุ่มเฟือยและไม่จำเป็น (superfluous) ที่เกิดในกรณีของ EQUI JOIN

การเขียนโฉเปอเรชัน NATURAL JOIN สามารถใช้  $\bowtie$  ก็ได้ หรือในบางตำราให้ใช้ \* (สัญกรณ์ star) [2] เพื่อให้ต่างจาก EQUI JOIN

## ตัวอย่าง

$$R = (A, B, C, D)$$

$$S = (E, B, D)$$

$$r \bowtie s = \pi_{r.A, r.B, r.C, r.D, s.E} (\sigma_{r.B = s.B \ r.D = s.D} (r \times s))$$

$$\text{สคีมาของผลลัพธ์} = (A, B, C, D, E)$$

(a) รูปที่ 97 แสดงขั้นตอนและผลลัพธ์ของ NATURAL JOIN

$\text{proj\_dept} \leftarrow \text{project} \bowtie \text{dept}$

(b)  $\text{dept\_locs} \leftarrow \text{department} \bowtie \text{dept\_location}$

(a)	PROJ_DEPT	PNAME	PNUMBER	PLOCATION	DNUM	DNAME	MGRSSN	MGRSTARTDATE
	ProductX	1	Bellaire	5	Research	333445555	1988-05-22	
	ProductY	2	Sugarland	5	Research	333445555	1988-05-22	
	ProductZ	3	Houston	5	Research	333445555	1988-05-22	
	Computerization	10	Stafford	4	Administration	987654321	1995-01-01	
	Reorganization	20	Houston	1	Headquarters	888665555	1981-06-19	
	Newbenefits	30	Stafford	4	Administration	987654321	1995-01-01	

(b)	DEPT_LOCS	DNAME	DNUMBER	MGRSSN	MGRSTARTDATE	LOCATION
	Headquarters	1	888665555	1981-06-19	Houston	
	Administration	4	987654321	1995-01-01	Stafford	
	Research	5	333445555	1988-05-22	Bellaire	
	Research	5	333445555	1988-05-22	Sugarland	
	Research	5	333445555	1988-05-22	Houston	

รูปที่ 97 ขั้นตอนและผลลัพธ์ของ NATURAL JOIN

### 7.2.9.3 THETA JOIN

ໂອເປ່ອເຮັນ THETA JOIN ອີກາຣ ແຈນ ທີ່ມີຈຳກັດເຊີພະເຈືອນໄຂທ່າກັນ ຈຶ່ງ  
ເປັນກາຣ ແຈນ ທີ່ເປີດກວ່າກ່າວ່າ NATURAL JOIN ແລະ EQUI JOIN

$$r \bowtie_c s = \sigma_c(r \times s)$$

$c$  คือเงื่อนไขการ JOIN ซึ่งจะเป็นเพรดิเคตได้ ๆ ก็ได้

หากมีทูเพิลชั้ก็จะถูกลดไป แต่คอลัมน์ชั้จะยังคงอยู่

### 7.2.10 OUTER JOIN

โอเปอเรชัน JOIN ได ๆ ที่ผ่านมา ได้แก่ NATURAL JOIN, EQUI JOIN และ THETA JOIN จะได้ผลลัพธ์คือทูเพิลที่พบในทั้งสองรีเลชันที่ JOIN กันเท่านั้น ซึ่งอาจทำให้มีความสามารถดึงข้อมูลที่ต้องการออกมากได้ในบางกรณี หรือเมื่อทูเพิลที่ JOIN มีค่าว่างที่例外ทริบิวต์ที่ใช้ JOIN ก็จะไม่ปรากฏในผลลัพธ์ เช่นกัน ดังนั้น เมื่อต้องการเก็บข้อมูลจากบางรีเลชันไว้แม้จะ join แล้วไม่พบ เราจะใช้โอเปอเรเตอร์ OUTER JOIN ที่จะเก็บค่าทั้งหมดที่เกิดจากโอเปอเรชัน JOIN แม้ JOIN ไม่พบไว้ได โดยถ้าค่าในข้อมูลได้มีก็จะกำหนดให้เป็นค่าว่าง

#### 7.2.10.1 LEFT OUTER JOIN

$$R \bowtie S$$

หรือ  $R =x S$

อ่านว่า R LEFT-OUTER-JOIN S ส่งผลให้ผลลัพธ์การ JOIN นี้เก็บทุกทูเพิลข้างซ้ายของโอเปอเรชันไว้ คือทุกทูเพิลใน R จะคงอยู่ และถ้าไม่พบว่ามีทูเพิลใน S ที่ JOIN ได้ ก็กำหนดค่าว่างให้ ดังแสดงในรูปที่ 98

ตัวอย่าง

Employee

Name	EmpID	DeptName
Harry	3415	Finance
Sally	2241	Sales
George	3401	Finance
Harriet	2202	Sales

Employee =X Dept

Name	EmpID	DeptName	Mgr
Harry	3415	Finance	Null
Sally	2241	Sales	Harriet
George	3401	Finance	Null
Harriet	2202	Sales	Harriet

Dept

DeptName	Mgr
Sales	Harriet
Production	Charles

รูปที่ 98 ขั้นตอนและผลลัพธ์การ LEFT OUTER JOIN

### 7.2.10.2 RIGHT OUTER JOIN

R  $\bowtie$  S

หรือ      R x= S

อ่านว่า R RIGHT-OUTER-JOIN S ส่งผลให้ผลลัพธ์การ JOIN นี้เก็บทุกทูเพิลข้างขวาของโอเปอเรชันไว้ คือทุกทูเพิลใน S จะคงอยู่ และถ้าไม่พบว่ามีทูเพิลใน R ที่ JOIN ได้ ก็กำหนดค่าว่างให้ ดังแสดงในรูปที่ 99

ตัวอย่าง

**Employee**

Name	EmplD	DeptName
Harry	3415	Finance
Sally	2241	Sales
George	3401	Finance
Harriet	2202	Sales

**Dept**

DeptName	Mgr
Sales	Harriet
Production	Charles

**Employee X= Dept**

Name	EmplD	DeptName	Mgr
Sally	2241	Sales	Harriet
Harriet	2202	Sales	Harriet
Null	Null	Production	Charles

รูปที่ 99 ขั้นตอนและผลลัพธ์การ RIGHT OUTER JOIN

#### 7.2.10.3 FULL OUTER JOIN

R  $\bowtie$  S

หรือ R =x= S

อ่านว่า R FULL-OUTER-JOIN S ส่งผลให้ผลลัพธ์การ JOIN นี้เก็บทุกทุกเพิล  
ทั้งจาก R และ S พร้อมเติมค่าว่างให้กับค่าที่ JOIN ไม่พบ ดังแสดงในรูปที่

100

ตัวอย่าง

Employee			
Name	EmpID	DeptName	
Harry	3415	Finance	
Sally	2241	Sales	
George	3401	Finance	
Harriet	2202	Sales	

Dept	
DeptName	Mgr
Sales	Harriet
Production	Charles

Employee =X= Dept			
Name	EmpID	DeptName	Mgr
Harry	3415	Finance	Null
Sally	2241	Sales	Harriet
George	3401	Finance	Null
Harriet	2202	Sales	Harriet
Null	Null	Production	Charles

รูปที่ 100 ขั้นตอนและผลลัพธ์การ FULL OUTER JOIN

### 7.2.11 แอกรีเกตฟังก์ชันและการกรุป

ความต้องการในการสอบถามข้อมูลจากฐานข้อมูลบางอย่างไม่สามารถทำได้ด้วยนิพจน์พีชคณิตเชิงสัมพันธ์ เช่น การหาผลรวม การนับ หรือการหาค่าเฉลี่ย ซึ่งต้องอาศัยการคำนวณข้อมูลจากตารางด้วย ดังนั้น จึงได้มีแอกรีเกตฟังก์ชันทางคณิตศาสตร์ (mathematical aggregate function) ที่ทำให้สามารถคำนวณข้อมูลในรีเลชันได้

แอกรีเกตฟังก์ชันที่ใช้บ่อย ได้แก่ การหาผลรวม การหาค่าเฉลี่ย การหาค่าสูงสุด การหาค่าต่ำสุด และการนับ

มีรูปแบบการเขียนดังนี้

การหาค่าสูงสุด (maximum) ของแอ็ตทริบิวต์ในรีเลชัน ใช้คำสั่ง MAX สามารถเขียนได้เป็น

$$\Sigma_{\text{MAX } <\text{Attribute}>} (r)$$

การหาค่าต่ำสุด (minimum) ของแอ็ตทริบิวต์ในรีเลชัน ใช้คำสั่ง MIN สามารถเขียนได้เป็น

$$\Sigma_{\text{MIN } <\text{Attribute}>} (r)$$

การหาค่าผลรวม (summation) ของแอ็ตทริบิวต์ในรีเลชัน ใช้คำสั่ง SUM สามารถเขียนได้เป็น

$$\Sigma_{\text{SUM } <\text{Attribute}>} (r)$$

การนับจำนวนทุเพลิ (count) ในรีเลชันใช้คำสั่ง COUNT สามารถเขียนได้เป็น

$$\Sigma_{\text{COUNT } <\text{Attribute}>} (r)$$

ส่วนการกรุ๊ป (group) ให้เขียนชื่อแอ็ตทริบิวต์ที่ต้องการกรุ๊ปไว้หน้า  $\Sigma$  เช่น

$$DNO \Sigma_{\text{COUNT}_{\text{SSN}} (\text{employee})}$$

ตัวอย่าง

$$\sigma_{r(DNO, NO\_OF\_EMPLOYEES, AVERAGE\_SAL)DNO} \bowtie COUNT_{SSN}, AVERAGE_{SALARY}(employee))$$

ได้ผลลัพธ์ดังแสดงในรูปที่ 101 ผลลัพธ์การแยกกรุ๊ปที่ 101

R	DNO	NO_OF_EMPLOYEES	AVERAGE_SAL
	5	4	33250
	4	3	31000
	1	1	55000

รูปที่ 101 ผลลัพธ์การแยกกรี๊เกต

$$DNO \bowtie COUNT_{SSN}, AVERAGE_{SALARY}(employee)$$

ได้ผลลัพธ์ดังแสดงในรูปที่ 102

COUNT_SSН	AVERAGE_SALARY
8	35125

รูปที่ 102 ผลลัพธ์การกรุ๊ป

### 7.3 เขตสมบูรณ์ของโอเปอเรชันเชิงสัมพันธ์

โอเปอเรชันเชิงสัมพันธ์ประกอบด้วย SELECT  $\sigma$ , PROJECT  $\pi$ , UNION  $\cup$ , INTERSECTION  $\cap$ , MINUS – และ CARTESIAN X ถือว่าเป็นเขตสมบูรณ์ เพราะโอเปอเรชันอื่นได้ในพื้นฐานเชิงสัมพันธ์ล้วนเป็นการผนวก 6 โอเปอเรชันนี้ทั้งสิ้น

ตัวอย่าง

$$R \cap S \text{ มีค่าเท่ากับ } (R \cup S) - ((R - S) \cup (S - R))$$

$$R \bowtie \langle \text{join condition} \rangle S \text{ มีค่าเท่ากับ } \sigma_{\langle \text{join condition} \rangle}(R \times S)$$

## 7.4 แคลคูลัสเชิงสัมพันธ์

แคลคูลัสเชิงสัมพันธ์ (relational calculus) เป็นภาษาสอบถามแบบฟอร์มัล (formal query language) สำหรับแบบจำลองเชิงสัมพันธ์

การเขียนนิพจน์ด้วยภาษาที่จะเป็นลักษณะพรรณา ไม่สนใจลำดับการทำงาน แบบพีชคณิตเชิงสัมพันธ์ ผู้เขียนสามารถเขียนนิพจน์อธิบายผลลัพธ์ที่ต้องการ โดยไม่ต้องระบุขั้นตอนใด ๆ

ตัวอย่างแนวคิดแคลคูลัสเชิงสัมพันธ์เปรียบเทียบกับพีชคณิตเชิงสัมพันธ์ เป็นต้น

สมมุติว่าเราต้องการค้นหาชื่อและหมายเลขโทรศัพท์ของร้านหนังสือที่วางขายหนังสือแฮร์รี พอตเตอร์

แคลคูลัสเชิงสัมพันธ์ แบ่งได้เป็น 2 แนวคิดคือ

1. แคลคูลัสเชิงสัมพันธ์อิงทูเพิล (tuple relational calculus)
2. แคลคูลัสเชิงสัมพันธ์อิงโดเมน (domain relational calculus)

### 7.4.1 แคลคูลัสเชิงสัมพันธ์อิงทุเพิล

ดร. เอดการ์ เอฟ คอตต์ เป็นผู้เสนอแนวคิดและกรรมวิธีแคลคูลัสเชิงสัมพันธ์ อิงทุเพิล เมื่อ ค.ศ. 1972 [33]

แคลคูลัสเชิงสัมพันธ์อิงทุเพิล มีรูปแบบดังนี้

$$\{t | p(t)\}$$

โดย  $t$  คือ ตัวแปรทุเพิล

$p(t)$  คือ เงื่อนไขของ  $t$

ผลลัพธ์ของนิพจน์นี้คือรีเลชันที่ประกอบด้วยทุเพิล  $t$  ที่  $t$  เป็นจริงตามเงื่อนไข  $p(t)$  ที่กำหนด

ตัวอย่าง

เมื่อต้องการค้นหา EMPLOYEE ที่เงินเดือนมากกว่า 30000 จากรีเลชัน  $S$  เขียนได้ดังนี้

$$\{S.salary | S.salary > 30000\}$$

สำหรับเงื่อนไข เราสามารถใช้ตัวปงปริมาณ (quantifier) ได้สองตัวคือ

1. existential quantifier  $\exists$  (there exists/for some)
2. universal quantifier  $\forall$  (For all)

ตัวอย่าง

ให้รีเลชัน BOOK (B) และ รีเลชัน STORE (S)

$\{S.Name, S.Phone | \exists B(B.StoreID=S.StoreID \wedge B.Title='Harry Potter')\}$

เงื่อนไขที่ต้องเป็นจริงคือ มีบางทูเพิลใน รีเลชัน BOOK ที่มีรหัสร้านเท่ากับ รหัสร้านในรีเลชัน STORE และ ชื่อหนังสือ Title มีค่าเท่ากับ ‘Harry Potter’

ผลลัพธ์ที่ได้คือชื่อร้านหนังสือและหมายเลขโทรศัพท์ของร้านที่มีหนังสือแฮรี่ พอตเตอร์วางขาย

#### 7.4.2 แคลคูลัสเชิงสัมพันธ์องค์ความนิยม

แคลคูลัสเชิงสัมพันธ์องค์ความนิยมนี้ เสนอโดย Michel Lacroix and Alain Pirotte [45] นิพจน์แคลคูลัสเชิงสัมพันธ์องค์ความนิยมรูปแบบดังนี้

$$\{\langle x_1, x_2, \dots, x_n \rangle | p[\langle x_1, x_2, \dots, x_n \rangle]\}$$

โดยแต่ละ  $x_i$  คือตัวแปรโดเมน (domain variable) และ  $p[\langle x_1, x_2, \dots, x_n \rangle]$  คือเงื่อนไข (condition) ของตัวแปรโดเมน เรียกว่า ฟอร์มูลา

ผลลัพธ์ของนิพจน์นี้ จะได้ทຸກຫຼັບ  $\langle x_1, x_2, \dots, x_n \rangle$  ที่ทำให้ฟอร์มูลา  $p[\langle x_1, x_2, \dots, x_n \rangle]$  เป็นจริง

ฟอร์มูลาอาจซับซ้อนได้ด้วยการฟอร์มูลาซ้อนกัน สามารถใช้ logical connector  $\wedge$  (and),  $\vee$  (or),  $\neg$  (not) ได้

ฟอร์มูลาที่เล็กที่สุดเรียกว่า อะตอมมิกฟอร์มูลา s อาจอยู่ในรูปแบบ

$$\langle x_1, x_2, \dots, x_n \rangle \in r$$

หรือ

$$x \text{ op } y$$

หรือ

$$x \text{ op } \text{ค่าคงที่} \text{ (constant)}$$

$$\text{op} \text{ ที่ใช้ได้คือ } <, \leq, >, \geq, =, \neq$$

หนึ่งอะตอมมิกฟอร์มูลาคือหนึ่งเงื่อนไข

ส่วนฟอร์มูลานั้น อาจเป็นอะตอมมิกฟอร์มูลา หรือ อะตอม

$\neg p, p \wedge q, p \vee q$  เมื่อ  $p, q$  เป็นฟอร์มูลา หรือ

$\exists x(p(x))$  เมื่อ  $x$  เป็นตัวแปรทุกเพล้อิสระ

$\forall x(p(x))$  เมื่อ  $x$  เป็นตัวแปรทุกเพล้อิสระ

ตัวแปรที่เพลอิสระ (free tuple variable) หมายถึง ตัวแปรที่ไม่มี  $\exists$  หรือ  $\forall$  กำกับ

หรืออธิบายได้ว่า  $\exists x$  และ  $\forall x$  ยึดเหนี่ยว (bind)  $x$

ตัวแปรใด ๆ ที่ไม่ได้ถูกยึดเหนี่ยว ให้ถือว่าอิสระ

ดังนั้น ตัวแปร  $x_1, x_2, \dots, x_n$  ที่อยู่ข้างซ้ายของเครื่องหมาย  $|$  เท่านั้น ที่เป็นตัวแปรอิสระในฟอร์มula  $p(\dots)$  ได้

## 7.5 เปรียบเทียบพีชคณิตเชิงสัมพันธ์กับแคลคูลัสเชิงสัมพันธ์

ขอเปรียบเทียบนิพจน์พีชคณิตเชิงสัมพันธ์กับแคลคูลัสเชิงสัมพันธ์จากโจทย์ว่า จงหาชื่อลูกค้าทุกคนที่กู้เงินจากธนาคารสาขาจามจุรี

นิพจน์พีชคณิตเชิงสัมพันธ์

$$\pi_{\text{customer-name}}(\sigma_{\text{branch-name} = \text{'Chamchuri'}}(\sigma_{\text{borrower.loan-number} = \text{loan.loan-number}}(\text{borrower} \times \text{loan})))$$

นิพจน์แคลคูลัสเชิงสัมพันธ์

$$\{\langle X \rangle | \langle X, Y \rangle \in \text{borrower} \wedge \exists A, B, C (\langle A, B, C \rangle \in \text{loan} \wedge B = \text{'Chamchuri'} \wedge Y = A)\}$$

## 7.6 บทสรุปและเรื่องชวนคิด

เราได้เรียนรู้ว่าฐานข้อมูลเชิงสัมพันธ์มีวิธีการคำนวณอย่างไร ทำไม่จำเป็นต้องเรียนรู้ ก็เพื่อให้ในขั้นตอนการออกแบบได้คำนึงถึงการคำนวณที่เหมาะสมที่สุด หากออกแบบได้ไม่ดีแล้ว จะส่งผลให้ขั้นตอนการคำนวณซับซ้อน เพิ่มเวลาในการประมวลผลโดยไม่จำเป็น วิศวกรคอมพิวเตอร์และโปรแกรมเมอร์ที่เชี่ยวชาญเรื่องฐานข้อมูลจะสามารถปรับเปลี่ยนนิพจน์พีชคณิตที่มีประสิทธิภาพได้ หรือใช้ query optimizer ให้เลือกนิพจน์ที่ดีที่สุดก็ได้ จึงอยากชวนผู้เรียนไปศึกษาว่า optimizer ทำงานอย่างไร และเราอาจมีวิธีปรับปรุงให้ optimizer เก่งขึ้นได้อย่างไรบ้าง

## แบบฝึกหัดท้ายบท

- ผลลัพธ์ของนิพจน์พีชคณิตเชิงสัมพันธ์ หรือนิพจน์แคตคูลัสเชิงสัมพันธ์ มีความเหมือน หรือต่างกัน ซึ่งผลลัพธ์นั้นคืออะไร
- จะอธิบายความแตกต่างของโอบอเรเตอร์ SELECT และ PROJECT โอบอเรเตอร์ SELECT จะเป็นการเลือกบางแถวของรีเลชัน ส่วน โอบอเรเตอร์ PROJECT จะเป็นการเลือกบางคอลัมน์ของรีเลชัน
- ผู้เรียนคิดว่า การดำเนินการโอบอเรเตอร์ SELECT ก่อน PROJECT และ PROJECT ก่อน SELECT มีความแตกต่างกันหรือไม่ อย่างไร  $\sigma_p(\pi_q(r))$  และ  $\pi_q(\sigma_p(r))$  มีความแตกต่างกันหรือไม่ อย่างไร)
- กำหนดรีเลชัน MOVIE เก็บข้อมูลเกี่ยวกับภาพยนตร์ในระบบชุมชนภาพยนตร์ออนไลน์แห่งหนึ่ง ซึ่งอธิบายโดย

MOVIE (TITLE, SYNOPSIS, DURATION, GENRE, DIRECTOR, YEAR, COMPANY)

จงเขียนนิพจน์พีชคณิตเชิงสัมพันธ์ที่ให้ผลลัพธ์ดังต่อไปนี้

- แสดงรายละเอียดทั้งหมดของภาพยนตร์ที่เป็นประเภท comedy
- แสดงเฉพาะชื่อภาพยนตร์ทั้งหมด
- แสดงเฉพาะชื่อภาพยนตร์ที่เป็นของบริษัท Marvel เท่านั้น
- แสดงชื่อ และเรื่องย่อภาพยนตร์ ที่มีประเภท horror และระยะเวลาของภาพยนตร์ไม่เกิน 120 นาที
- แสดงชื่อภาพยนตร์ และชื่อผู้กำกับของทุกภาพยนตร์ โดยแสดงเป็นรีเลชันใหม่ ชื่อ DIRECTOR ที่มีคอลัมน์ชื่อว่า Movie และ Director

## 5. กำหนดรีเลชันดังต่อไปนี้

INSTRUCTOR (FNAME, LNAME, INSTRUCTOR\_ID, YEAR)

STUDENT (FNAME, LNAME, STUDENT\_ID, YEAR)

แทนข้อมูลอาจารย์ และนิสิตของมหาวิทยาลัยแห่งหนึ่ง จังหวัดนนิพจน์  
พีชคณิตเชิงสัมพันธ์ที่ให้ผลลัพธ์ดังต่อไปนี้

5.1. แสดงข้อมูลทั้งหมดของอาจารย์ และนักเรียนทุกคน รวมไว้ใน  
รีเลชันใหม่ที่อธิบายโดย

PEOPLE (FNAME, LNAME, PEOPLE\_ID, YEAR)

5.2. แสดงข้อมูลเฉพาะชื่อ นามสกุล ของคนที่เป็นทั้งอาจารย์ และ  
นักเรียน

5.3. แสดงข้อมูลเฉพาะชื่อ นามสกุล ของคนที่เป็นอาจารย์ หรือ  
นักเรียน อายุ่ได้อย่างหนึ่ง

5.4. แสดงข้อมูลเฉพาะชื่อ นามสกุลของอาจารย์ ที่มีญาติเป็นนักเรียน  
(ยกเว้นตัวเอง) กล่าวคือ เป็นอาจารย์ที่มีนักเรียนนามสกุลเดียวกับ  
อาจารย์ แต่ชื่อต่างกัน

## 6. กำหนดรีเลชันดังต่อไปนี้

MENU (MENU\_ID, MENU\_NAME, MENU\_PRICE)

ORDER (ORDER\_ID, CUSTOMER\_NAME)

ORDER\_ITEM (ORDER\_ID, MENU\_ID)

RECOMMENDED\_MENU (MENU\_ID)

แทนข้อมูลเมนู ข้อมูลการสั่งอาหาร รายการอาหารที่สั่ง และเมนูแนะนำ ของร้านอาหารแห่งหนึ่ง จงเขียนนิพจน์พีชคณิตเชิงสัมพันธ์ที่ให้ผลลัพธ์ดังต่อไปนี้

- 6.1. แสดงรายการชื่อเมนู และหมายเลขอารสารสั่งซื้อ ที่ตรงกัน (ORDER\_ID และ MENU\_NAME)
- 6.2. แสดงรายชื่อเมนู ราคา ที่ลูกค้าสั่ง ตามรายชื่อของลูกค้า (CUSTOMER\_NAME, MENU\_NAME และ MENU\_PRICE)
- 6.3. แสดงรายชื่อลูกค้าทั้งหมด ที่เคยสั่งเมนูแนะนำครบทุกเมนู
- 6.4. แสดงราคารวม แยกตามแต่ละ ORDER\_ID
- 6.5. แสดงชื่อลูกค้า และจำนวนเงินที่จ่ายไป สำหรับลูกค้าที่ใช้จ่ายรวมมากที่สุดของร้าน
7. จงแปลงนิพจน์พีชคณิตเชิงสัมพันธ์ต่อไปนี้ เป็นนิพจน์แคลคูลัสเชิงสัมพันธ์
  - 7.1.  $\pi_{NAME, SALARY} (\sigma_{SALARY > 20000} (employee))$
  - 7.2.  $\pi_{STUDENT_NAME, COURSE_NAME} (student \bowtie enrollment \bowtie course)$   
โดยที่ student.id = enrollment.student\_id และ enrollment.course\_id = course.id



## บทที่ 8

### ภาษาอีสานและ

#### วัตถุประสงค์

- เพื่อให้รู้จักภาษาอีสานและเป็นต้นจากตัวอย่าง
- เพื่อให้ใช้ภาษาอีสานและสอบถามข้อมูลจากฐานข้อมูลแบบง่ายๆ

ภาษาเอสคิวแอล (structured query language : SQL) เป็นภาษาชนิดโดเมนจำเพาะ (domain-specific) กล่าวคือ เป็นภาษาที่ออกแบบมาเพื่องานบางอย่างโดยเฉพาะ

ภาษาเอสคิวแอลถูกออกแบบมาเพื่อให้ผู้ใช้สามารถจัดการข้อมูลในระบบ การจัดการฐานข้อมูลเชิงสัมพันธ์ และเหมาะสมกับข้อมูลแบบมีโครงสร้างและมีความสัมพันธ์ต่อกัน สามารถเข้าถึงข้อมูลหลายพิลพิลพร้อม ๆ กันในเพียงคำสั่งเดียว โดยที่ผู้ใช้เอสคิวแอลไม่จำเป็นต้องรู้ว่าข้อมูลถูกจัดเก็บอย่างไร หรืออยู่ที่ไหน [46-50] ภาษาเอสคิวแอลสร้างมาบนพื้นฐานของแคลคูลัสเชิงสัมพันธ์

วัตถุประสงค์ของบทนี้คือให้รู้จักภาษาเอสคิวแอลเบื้องต้นเท่านั้น เน้นตัวอย่างคำสั่งพื้นฐานเพื่อให้เห็นรูปแบบของภาษา หากผู้เรียนต้องการเรียนรากยสัมพันธ์ (syntax) หรือคำสั่งภาษาเอสคิวแอลอย่างละเอียด ขอแนะนำให้ศึกษาจากตำราเอสคิวแอลต่อไป

## 8.1 รู้จักภาษาเอสคิวแอล

เอสคิวแอลเขียนง่าย มีรากยสัมพันธ์และไวยากรณ์ยืดหยุ่น ไม่ซับซ้อน แต่ทรงพลังในการค้นหาและจัดการข้อมูล เราสามารถเข้าใจและเขียนเอสคิวแอลได้อย่างเป็นธรรมชาติ (intuitive) เพราะเป็นภาษาที่ใกล้เคียงกับภาษามนุษย์ เมื่อเราเขียนคำสั่งเอสคิวแอลแล้วก็จะเป็นหน้าที่ของระบบการจัดการฐานข้อมูลที่จะประมวลผลและนำผลลัพธ์มาแสดง นอกเหนือจากนั้น เรายังสามารถใช้ตัวอปทิเมอร์คิวเรีย (query optimizer) จัดเรียงคำสั่งเอสคิวแอลให้ได้ประสิทธิภาพที่สุดแล้วความหมายคงเดิมได้ด้วย

ภาษาเอสคิวแอล เสนอโดยบริษัทไอบีเอ็มเมื่อช่วง ค.ศ. 1974 ซึ่งว่า Structured Query Language หรือ SEQUEL เพื่อใช้กับระบบจัดการฐานข้อมูลเชิงสัมพันธ์ของไอบีเอ็ม ซึ่ง System R ต่อมาเปลี่ยนชื่อเป็น SQL

เมื่อเอสคิวแอลได้รับความนิยมอย่างแพร่หลาย จึงจำเป็นต้องกำหนด มาตรฐาน ในที่สุด เมื่อ ค.ศ. 1986 สถาบันมาตรฐานแห่งสหรัฐอเมริกา (American National Standards Institute : ANSI) ได้ให้การรับรอง มาตรฐานเอสคิวแอล เรียกว่า SQL-86 ต่อมาก็ ISO ก็รับรองมาตรฐาน เอสคิวแอลใน ค.ศ. 1987 จากนั้นต่อไป มาตรฐานนี้ก็ได้ถูกปรับปรุงมาเรื่อยๆ โดยหน่วยงานมาตรฐานข้ามชาติหลายหน่วยงานร่วมกัน

### ภาษาเอสคิวแอล ทำให้ผู้ใช้

- สามารถสร้างฐานข้อมูลโดยกำหนดรีเลชันได้
- ทำให้ผู้ใช้สามารถจัดการข้อมูลได้ ทั้งการแทรกข้อมูล ลบข้อมูล หรือแก้ไขข้อมูล
- ทำให้ผู้ใช้สามารถสอบถามข้อมูลได้ง่ายอีกด้วย

รูปแบบในการเขียนคำสั่งเอสคิวแอล ให้ใช้ตัวอักษรตัวใหญ่เมื่อเขียน รายการสัมพันธ์ที่เป็นคำส่วนในเอสคิวแอล (reserved word) และใช้ตัวอักษร ตัวเล็กเมื่อเขียนศัพท์ที่อื่นใดที่ผู้เขียนกำหนดเอง เช่น ชื่อตาราง ชื่อฟิลเตอร์

ทั้งนี้ ขอเทียบศัพท์ที่ใช้ในแบบจำลองเชิงสัมพันธ์และพีชคณิตเชิงสัมพันธ์กับ ศัพท์ที่ใช้ในเอสคิวแอลที่อาจทำให้สับสนได้ดังแสดงในตารางที่ 3

ตารางที่ 3 ศัพท์ที่ใช้ในแบบจำลองเชิงสัมพันธ์และพีชคณิตเชิงสัมพันธ์กับศัพท์ที่ใช้ในอีสติวแอล

ศัพท์แบบจำลองเชิงสัมพันธ์และพีชคณิตเชิงสัมพันธ์	ศัพท์อีสติวแல
รีเลชัน (relation)	ตาราง (table)
แอ็ตทริบิวต์ (attribute)	ฟิลด์ (field)
PROJECT	SELECT
SELECT	WHERE

ตัวอย่าง

STUDENTS	sid	name	login	age	gpa
	6120056721	Chonlathorn	chonlathorn.k@student.chula.ac.th	23	4.00
	6120061221	Kailerk	kailerk.t@student.chula.ac.th	18	3.80
	6121005121	Kawin	kawin.m@student.chula.ac.th	18	3.50
	6121007721	Panusorn	panusorn.p@student.chula.ac.th	24	3.75
	6121010121	Sanchai	sanchai.j@student.chula.ac.th	25	4.00
	6121026321	Thanyanuch	thanyanuch.c@student.chula.ac.th	28	3.98

รูปที่ 103 รีเลชัน STUDENTS

จากรีเลชัน STUDENTS ในรูปที่ 103 หากเราต้องการหาชื่อนักเรียนที่อายุ 18 ปี เราสามารถเขียนคำสั่งເອສົກວແລດັ່ງນີ້

```
SELECT *
FROM Students S
WHERE S.age = 18
```

ผลลัพธ์คือ

STUDENTS	sid	name	login	age	gpa
	6120061221	Kailerk	kailerk.t@student.chula.ac.th	18	3.80
	6121005121	Kawin	kawin.m@student.chula.ac.th	18	3.50

รูปที่ 104 ผลลัพธ์การ SELECT \*

หรือหากต้องการได้ผลลัพธ์เฉพาะชื่อและ login ให้แก้คำสั่งบรรทัดแรกเป็น

```
SELECT S.name, S.login
FROM Students S
WHERE S.age = 18
```

ตัวอย่าง

จากรีเลชัน STUDENTS ในรูปที่ 103 และรีเลชัน ENROLL ในรูปที่ 105

ENROLL	sid	cid	gpa
	6120056721	Database201	4.00
	6120061221	Java101	3.80
	6121005121	SA112	3.50
	6121007721	SE113	3.75

รูปที่ 105 รีเลชัน ENROLL

หากต้องการสอบถามนักเรียนที่ได้ gpa = 4.0 จะเขียนได้ดังนี้

```
SELECT S.name, E.cid
FROM Students S, Enroll E
WHERE S.sid = E.sid AND E.gpa = 4.0
```

ผลลัพธ์แสดงในรูปที่ 106

S.name	E.cid
Chonlathorn	Database201

รูปที่ 106 ผลลัพธ์นักเรียน ที่ gpa = 4.0

จะเห็นได้ว่า เราสามารถเข้าใจคำสั่งได้โดยไม่ต้องอธิบาย

## 8.2 คำสั่งที่ใช้บ่อย

ในบทนี้ เราจะนำเสนอคำสั่งที่ใช้บ่อย โดยแสดงตัวอย่างคำสั่งเบสิคิวแอล เพราะเชื่อว่าไม่จำเป็นต้องอธิบายวากยสัมพันธ์ (syntax) ก็สามารถเข้าใจได้ง่าย

### 8.2.1 สร้างรีเลชัน

เมื่อต้องการสร้างรีเลชัน ให้ใช้คำสั่ง CREATE TABLE และกำหนด  
แอ็ตทริบิวต์พร้อมทั้งไดเมน ดังนี้

```
CREATE TABLE Students      (sid CHAR(20),  
                           name CHAR(20),  
                           login CHAR(10),  
                           age INTEGER,  
                           gpa REAL)
```

```
CREATE TABLE Enrolled      (sid CHAR(20),  
                           cid CHAR(20),  
                           grade CHAR(2))
```

### 8.2.2 ลบรีเลชัน

เมื่อต้องการลบรีเลชัน ให้ใช้คำสั่ง DROP

```
DROP TABLE Students
```

คำสั่งนี้จะลบสคีมา Students และข้อมูลทุกๆเพิลทิ้งไป

### 8.2.3 แก้ไขรีเลชัน

การแก้ไขสคีมาของตาราง ใช้คำสั่ง ALTER เช่น ต้องการเพิ่มฟิลด์ จะส่งผลให้ทุกทูเพิลในตารางมีฟิลด์เพิ่มและให้ใส่ค่า null ไปที่ฟิลด์ใหม่นั้น

```
ALTER TABLE Students  
ADD COLUMN firstYear INTEGER
```

### 8.2.4 เพิ่มทูเพิล

เพิ่มทีละทูเพิล ใช้คำสั่ง INSERT

```
INSERT  
INTO Students (sid, name, login, age, gpa)  
VALUES ('5931072221', 'Sanchai',  
       'Sanchai.J@chula.ac.th', 18, 3.9)
```

### 8.2.5 ลบทูเพิล

ลบทูเพิลตามเงื่อนไขที่ระบุ ใช้คำสั่ง DELETE

เช่น ลบทูเพิลที่ name = Proadpran

```
DELETE  
FROM Students S  
WHERE S.name = 'Proadpran'
```

### 8.2.6 อัปเดตฐานข้อมูล

อัปเดตที่ลักษณะพิเศษ ใช้คำสั่ง UPDATE

```
UPDATE Students S  
SET      S.age = S.age + 1, S.gpa = S.gpa + 0.1  
WHERE    S.sid = '5930136421'
```

### 8.2.7 กำหนดคีย์หลักและคีย์คู่แข่ง

หากมีคีย์คู่แข่ง ให้เลือกคีย์หนึ่งเป็นคีย์หลัก ใช้คำสั่ง PRIMARY KEY และหากต้องการให้ค่าในแอตทริบิวต์ใด ๆ ห้ามซ้ำกัน ให้ใช้คำสั่ง UNIQUE

```
CREATE TABLE Enrolled (sid    CHAR(20)  
                      cid    CHAR(20),  
                      grade  CHAR(2),  
PRIMARY KEY (sid, cid))
```

กรณีนี้ นักเรียน 1 คน ลงทะเบียน 1 วิชา จะได้เกรด 1 เกรด แต่นักเรียนคนนี้จะลงทะเบียนวิชาเดิมไม่ได้

```
CREATE TABLE Enrolled (sid    CHAR(20)  
                      cid    CHAR(20),  
                      grade  CHAR(2),  
PRIMARY KEY (sid),  
UNIQUE (cid, grade))
```

ถ้าเขียนแบบนี้ นักเรียน 1 คน จะลงทะเบียนได้เพียงวิชาเดียว นอกเหนือจากนั้น จะไม่มีนักเรียนคนไหนได้เกรด ข้ากันด้วย

ดังนั้น การกำหนดคีย์นั้นต้องออกแบบให้ดี ตรงตามความหมายที่ต้องการ ให้รองรับข้อมูลตามที่ควรจะเป็น

### 8.2.8 กำหนดฟอร์เรนคีย์

การกำหนดฟอร์เรนคีย์ ใช้คำสั่ง FOREIGN KEY และ REFERENCES เพื่อโยงความสัมพันธ์ไปยังรีเลชันที่เชื่อมโยง

```
CREATE TABLE Enrolled (sid CHAR(20),
                        cid CHAR(20),
                        grade CHAR(2),
                        PRIMARY KEY (sid, cid),
                        FOREIGN KEY (sid)
                        REFERENCES Students)
```

```
CREATE TABLE Works_In (ssn      CHAR(13),
                       did     INTEGER,
                       since   DATE,
                       PRIMARY KEY (ssn, did),
                       FOREIGN KEY (ssn)
                       REFERENCES Employees,
                       FOREIGN KEY (did)
                       REFERENCES Departments)
```

### 8.2.9 กำหนดทางเลือกเมื่อลบหรืออปเดตข้อมูลอ้างอิงข้ามตาราง

มีทางเลือก 4 ทางคือ

- NO ACTION ปฏิเสธการลบหรืออปเดต
- CASCADE ลบทุกทุเพลิที่อ้างอิงทุเพลิที่ถูกลบ
- SET NULL กำหนดค่าไว้ให้ฟอร์เนนคีย์ของทุเพลิที่จะลบ
- SET DEFAULT กำหนดค่าดีฟอลต์ให้ฟอร์เนนคีย์ของทุเพลิที่จะลบ

```
CREATE TABLE Enrolled (sid    CHAR(20),
                      cid    CHAR(20),
                      grade  CHAR(2),
                      PRIMARY KEY (sid,cid),
                      FOREIGN KEY (sid)
                      REFERENCES Students
                      ON DELETE CASCADE
                      ON UPDATE SET DEFAULT)
```

### 8.2.10 สร้างวิว

คำสั่ง VIEW ใช้เพื่อกำหนดวิวให้ผู้ใช้

```
CREATE VIEW YoungActiveStudents (name, grade)
AS SELECT S.name, E.grade
      FROM Students S, Enrolled E
      WHERE S.sid = E.sid AND S.age < 21
```

### 8.3 บทสรุปและเรื่องชวนคิด

ภาษาอีสานเป็นภาษาที่เข้าใจง่าย เรียนรู้ง่าย เขียนง่าย เพราะเป็นธรรมชาติ คล้ายภาษามนุษย์ สร้างบนพื้นฐานของแคลคูลัสเชิงสัมพันธ์ ผู้ใช้ภาษาไม่สามารถสอบถามและจัดการข้อมูลในฐานข้อมูลได้ไม่ยาก ขอให้ผู้เรียนได้ลองคิดดูว่ามีเหตุการณ์ใดที่ภาษาอีสานไม่สามารถรองรับได้บ้าง

## แบบฝึกหัดท้ายบท

ระบบฐานข้อมูลนักเรียน อาจารย์ และการลงทะเบียนเรียนของมหาวิทยาลัย หนึ่ง ประกอบด้วย 5 รีเลชัน ซึ่งมีตัวอย่างข้อมูลตั้งแสดงด้านล่างนี้ จงเขียนภาษาอังกฤษเพื่อทำตามคำสั่งที่กำหนดต่อไปนี้

### STUDENT

STUDENT_ID (PK)	STUDENT_FNAME	STUDENT_LNAME	YEAR
6030000121	เกษม	สวัสดิ์	2017
6030000221	ชรัสุขนก	ใจดี	2017
5930000321	คำมิน	สวัสดิ์	2016
5930000421	จิรัชฎ์	สุชี	2016

### INSTRUCTOR

INSTRUCTOR_ID (PK)	INSTRUCTOR_NAME	INSTRUCTOR_LNAME
100041	ปูเจ	เตียงเสนา
100042	เอก	ชัยชนะ
100043	ณภัทร	หมื่นลี้
100044	เมธส	นานะวงศ์

**COURSE**

COURSE_ID (PK)	COURSE_NAME
2110322	DATABASE SYSTEMS
2110221	COMP ENG ESS
2110202	DISCRETE STRUC COM

**COURSE\_SECTION**

SECTION_ID (PK)	COURSE_ID (FK)	INSTRUCTOR_ID (FK)	YEAR	SEMESTER	SECTION	DAY	TIME	CAPACITY
40001	2110322	100041	2016	1	1	MON	9:00 – 12:00	30
40002	2110322	100042	2016	1	2	TUE	13:00 – 16:00	30

**ENROLLMENT**

STUDENT_ID (PK, FK)	SECTION_ID (PK, FK)	GRADE
6030000121	40001	4.00
6030000221	40002	3.50
5930000321	40001	2.50

1. จงเขียนคำสั่งเพื่อสร้างตาราง ENROLLMENT โดยมีข้อกำหนดเพิ่มเติม คือ STUDENT\_ID มีความยาว 10 อักขระ และ SECTION\_ID มีความยาว 5 อักขระ และกำหนดให้ ถ้าหากข้อมูลนักเรียนในตาราง

STUDENT ลูก\_lb\_ทิ้ง หรือ ข้อมูล SECTION ในตาราง COURSE\_SECTION ลูก\_lb\_ทิ้ง ข้อมูล ENROLLMENT ที่เกี่ยวข้องจะลูก\_lb\_ทิ้งด้วย

2. เมื่อใช้งานระบบไปนานวันเข้า ความยาวของ SECTION\_ID เพียง 5 อักษรไม่พอในการใช้งาน จะเขียนคำสั่งเพื่อแก้ไขรีเลชัน COURSE\_SECTION โดยแก้ไขให้ SECTION\_ID มีความยาว 6 อักษร
3. ในปีการศึกษาใหม่ จะมีนักเรียนเข้าเรียนใหม่อุ่นเสมอ จึงเขียนคำสั่งเพื่อเพิ่มข้อมูลนักเรียน ที่มีรหัสนักเรียน 6039999921 ชื่อ นงนุช รักเรียน ซึ่งเข้าเรียนปี 2016 ลงไปในตาราง STUDENT
4. เมื่อนักเรียนมีการเปลี่ยนชื่อ หรือนามสกุล จะต้องแจ้งผู้ดูแลระบบเพื่อแก้ไขชื่อ姓名สกุลในระบบให้ถูกต้อง จึงเขียนคำสั่งเพื่อแก้ไขข้อมูลชื่อ นักเรียน ที่มีรหัสนักเรียน 6039999921 โดยเปลี่ยนแปลงเป็นชื่อ ชวัญ ชนก
5. เมื่อนักเรียนลาออกจาก ผู้ดูแลระบบจะดำเนินการลบข้อมูลนักเรียนออกจากระบบ จึงเขียนคำสั่งเพื่อลบบันก์เรียนที่มีรหัสนักเรียน 6039999921
6. จงเขียนคำสั่งเพื่อแสดงเฉพาะรหัสนักเรียน ชื่อ นามสกุล ของนักเรียนที่เข้าเรียนในปี 2017 ทุกคน
7. จงเขียนคำสั่งเพื่อแสดงเฉพาะรหัสนักเรียน ชื่อ นามสกุล ของนักเรียนที่เข้าเรียนในปี 2017 ลำดับที่ 16 – 25
8. จงเขียนคำสั่งเพื่อแสดง section ชื่อ-นามสกุลของอาจารย์ที่สอน วัน เวลา และจำนวนนิสิตที่รับทั้งหมดของวิชา 2110322 ในปีการศึกษา 2016 เทอม 1

9. จงเขียนคำสั่งเพื่อแสดงจำนวนนักเรียนที่ซื้อช้ากัน โดยแสดง ชื่อ และ จำนวนคนที่ซื้ากัน เรียงลำดับตามจำนวนที่ซื้ากัน หากมีจำนวนที่ซื้ากัน มากกว่า 1 อันดับ
10. จงเขียนคำสั่งเพื่อแสดงเกรดเฉลี่ยของรายวิชา โดยแสดง รหัสวิชา (COURSE\_ID) และเกรดเฉลี่ยรวมของรายวิชานั้น โดยเรียงลำดับตาม เกรดเฉลี่ยมากไปหน้าย

## บทที่ 9

### การจัดเก็บและการทำดัชนี

#### วัตถุประสงค์

- เพื่อให้ทราบวิธีการจัดเก็บข้อมูลของฐานข้อมูล
- เพื่อให้คำนวนเวลาที่ใช้ในการทำงานของໂອເປົວເຮັນ
- เพื่อให้ทราบประเภทของดัชนีและเดือกໃຫ້ได้เหมาะสม

เมื่อเรามีข้อมูลในฐานข้อมูลจำนวนมาก เราจะต้องคำนึงถึงวิธีการจัดเก็บที่เหมาะสมด้วย โดยมีเป้าหมายหลักคือให้ตอบสนองผู้ใช้ได้รวดเร็ว ทันกาล วิธีการจัดเก็บข้อมูลมีหลายวิธี เราต้องออกแบบตามลักษณะการใช้งานของฐานข้อมูลนั้น ๆ นอกเหนือจากนั้น เรายังสามารถเพิ่มประสิทธิภาพในการเรียกใช้ให้ได้รวดเร็วด้วยกันเพิ่มด้วยให้กับฐานข้อมูลอีกด้วย

ในบทนี้ เราจะพูดเรื่องการจัดเก็บข้อมูลและการทำดัชนีในฐานข้อมูล เพื่อให้การเข้าถึงข้อมูลมีประสิทธิภาพ เหมาะกับรูปแบบการสอบถามและการใช้งาน โดยเฉพาะงานที่ใช้บ่อย ดังนั้น เราจึงต้องเรียนรู้การใช้งานข้อมูลเป็นอย่างไร จึงจะออกแบบการจัดเก็บให้เหมาะสม และทำดัชนีที่ส่งผลให้ประสิทธิภาพการใช้งานของฐานข้อมูลที่ดีด้วย อย่างไรก็ดี ไม่ว่าเราจะออกแบบด้วยความระมัดเพียงใด เมื่อนำฐานข้อมูลไปใช้งานจริงก็ยังมีโอกาสที่จะต้องปรับวิธีการจัดเก็บและปรับตั้งนิตามสถานการณ์ที่เปลี่ยนไป เพื่อให้เหมาะสมกับการใช้งานและตอบสนองความต้องการของผู้ใช้อยู่เสมอ หน้าที่ของผู้ดูแลฐานข้อมูล จึงเป็นผู้ที่มีความสำคัญอย่างยิ่ง โดยเฉพาะในองค์กรที่ใช้ฐานข้อมูลขนาดใหญ่และมีผู้ใช้ฐานข้อมูลจำนวนมาก

ตำนานอ้างอิงวิธีการดัชนีทั้งบทนี้จาก [1]

รูปที่ 107 แสดงโครงสร้างของระบบการจัดการฐานข้อมูล ประกอบด้วย 5 ขั้นคือ

- 1) การอปติไมซ์การสอบถาม (query optimization)
- 2) การคำนวณเชิงสัมพันธ์ (relational operators)
- 3) ไฟล์และวิธีการเข้าถึง (file and access method)

4) การจัดการบัฟเฟอร์ (buffer management)

5) การจัดการพื้นที่ดิสก์ (disk space management)

ส่วนที่เกี่ยวข้องการการจัดเก็บและตั้งค่าคือ 3 ขั้นหลัง ได้แก่ ขั้นไฟล์และวิธีการเข้าถึง



รูปที่ 107 โครงสร้างของระบบการจัดการฐานข้อมูล

เมื่อเราต้องการจะเก็บข้อมูล ระบบการจัดการฐานข้อมูลจะส่ง request ไปยังส่วนการจัดการบัฟเฟอร์ เพื่อจัดสรร (allocate) พื้นที่ให้ หรือเมื่อต้องการอ่านข้อมูลใด ส่วนการจัดการบัฟเฟอร์ก็จะทำหน้าที่หาว่าข้อมูลนั้นอยู่ที่ตำแหน่งใด แล้วดึงข้อมูลนั้นมาเก็บไว้ในหน่วยความจำหลัก (main memory)

## 9.1 หน่วยเก็บข้อมูลภายนอก

การทำงานในระบบการจัดการฐานข้อมูลส่วนใหญ่คือการอ่าน (read) ข้อมูล และเขียน (write) ข้อมูลลงบนจากหน่วยเก็บข้อมูลภายนอก (external storage)

หน่วย (unit) วัดความเร็วในการอ่านและเขียนหรืออินพุตเอาต์พุต ย่อว่าไอโอ (input/output – i/o) จึงไม่คิดเป็นหน่วยเวลา ไม่นับเป็นฟล็อป (flop) แบบที่วัดความเร็วของระบบปฏิบัติการคอมพิวเตอร์ทั่วไป แต่จะคิดเป็น เพจ (page) โดยเฉพาะเจาะจงคือเพจอินพุตเอาต์พุต เรียกย่อว่า เพจไอโอ (page i/o) ว่ามีกี่ครั้งที่เราจะต้องเข้าถึงเพจ หรือเข้าถึงหน่วยจัดเก็บรอง (secondary storage) เพราะการอ่านเขียนข้อมูลเป็นงานที่กินเวลา เราจึงคำนวณเวลาที่ระบบปฏิบัติการฐานข้อมูลใช้เป็นหน่วยเพจ เพราะเวลาที่ซีพียูใช้จะไม่ต่างมากเท่ากับไอโอ ดังนั้น เพื่อให้ระบบฐานข้อมูลตอบสนองได้เร็ว เราจึงมีเป้าหมายลดเพจไอโอให้ต่ำที่สุด

### 9.1.1 ประเภทของหน่วยเก็บข้อมูลภายนอก

หน่วยเก็บข้อมูลภายนอก แบ่งออกเป็น 2 ประเภทได้แก่

1. ดิสก์ (disk) เป็นอุปกรณ์ที่บันทึกข้อมูลได้จำนวนมาก ใช้เทคโนโลยีแม่เหล็ก หรือแสง หรือทั้งสองอย่าง เช่น แมกнетิกดิสก์ แผ่นโซลิฟิวเดต ดิสก์ และหน่วยความจำไฮบริดสเตต ข้อดีคือ เก็บข้อมูลได้มาก และสามารถเข้าถึงข้อมูลด้วยแบบสุ่ม (random access) คือเข้าถึงข้อมูลตรงที่ที่ข้อมูลถูกบรรจุอยู่ได้ทันที
2. เทป (tape) เป็นหน่วยเก็บข้อมูลที่ใช้กันมานานตั้งแต่คอมพิวเตอร์ยุคต้น ปัจจุบันได้รับความนิยมน้อยลง มีลักษณะเป็นม้วน หรือตลับ (cartridge) การเข้าถึงข้อมูลเป็นแบบลำดับ (sequential access) จึงเข้าถึงได้ช้า ข้อดีคือราคาถูกกว่าดิสก์มากและเคลื่อนย้ายง่าย จึงเหมาะสมกับการเก็บข้อมูลที่ไม่ได้ใช้เป็นประจำ เช่นใช้สำรองข้อมูล (backup) หรือใช้เก็บข้อมูลถาวร (archive) ที่เมื่อนำกลับมาใช้ก็ให้อ่านข้อมูลทั้งหมดจากเทป

การอ่านข้อมูลแต่ละเพจจากดิสก์จะใช้เวลาเท่ากัน (fixed cost) ดังนั้น การอ่านข้อมูลจากเพจแบบต่อเนื่อง (consecutive pages) จะจะสิ้นเปลืองเวลาน้อยกว่าการแรนดอมอ่านที่ละเพจอยู่มาก

## 9.2 ไฟล์และวิธีการเข้าถึง

การเก็บเรียบข้อมูลในหน่วยเก็บข้อมูลจะเก็บในลักษณะไฟล์ของ เรกคอร์ด (record) ซึ่งเรกคอร์ดก็คือทูเพลิหรือแคร์ข้อมูลนั่นเอง การบันทึกเรียบข้อมูลนี้จะต้องระบุตัวเรกคอร์ด (Record ID) ที่มี รหัสเรกคอร์ด (Record ID) ย่อว่า RID ต่อจากนี้จะเรียก RID ทำหน้าที่เป็นพอยน์เตอร์หรือตัวชี้ (pointer) บอกเพจของเรกคอร์ดนั้น

ขั้นไฟล์และวิธีการเข้าถึงนี้ มีหน้าที่จัดเริ่มข้อมูลและระบุเพจของข้อมูล และติดตามสถานะการจัดสรรเพจให้แก่ไฟล์ ตรวจสอบเนื้อที่ว่างในแต่ละเพจที่ถูกจัดสรรแล้ว ขั้นตอนนี้จะเกิดขึ้นทุกครั้งที่มีการสร้าง (create) แทรก (insert) ลบ (delete) และ สแกน (scan) ข้อมูล ซึ่งการสแกนข้อมูลคือการค้นคืน (retrieve) ข้อมูลจำนวนมาก เช่น หาข้อมูลแล้วพิมพ์รายงานการลงทะเบียนของนิสิตทั้งหมดมหาวิทยาลัย เป็นต้น

### 9.3 การจัดโครงสร้างไฟล์

การจัดโครงสร้างไฟล์ (file organization) คือกระบวนการจัดระเบียบเรกคอร์ดในไฟล์เพื่อนำไปเก็บในดิสก์

ข้อมูลในดิสก์อาจถูกเก็บแบบเรียงลำดับ (sorted) หรือไม่เรียงลำดับ (unsorted) การที่จะเก็บแบบเรียงหรือไม่เรียงลำดับนั้น ให้คำนึงว่าเป้าหมายของผู้ออกแบบฐานข้อมูลคือต้องการลดจำนวนเพจไ้อิโวให้เหลือน้อยที่สุด เพื่อให้การทำงานของฐานข้อมูลใช้เวลาอ่านน้อยที่สุด ซึ่งจะขึ้นอยู่กับโอลีโอเปอร์เรชันที่ฐานข้อมูลนั้นใช้บ่อย โดยเฉพาะโอลีโอเปอร์เรชันการสอบถามข้อมูล

เราสามารถแบ่งโครงสร้างไฟล์ได้ 3 แบบ คือ

- ไฟล์แบบไม่เรียงลำดับ (unsorted file) หรือเรียกอีกชื่อว่าไฟล์ฮีป (heap file) คือโครงสร้างไฟล์แบบไม่เรียงลำดับ ซึ่งทำให้การเข้าถึงข้อมูลเป็นแบบลำดับ เหมาะกับโอลีโอเปอร์เรชันสแกน ที่ต้องการอ่านข้อมูลทุกรეกคอร์ด

2. ไฟล์แบบเรียงลำดับ (sorted file) ข้อดีคือจะค้นหา (search) ข้อมูลได้เร็ว โดยเฉพาะการค้นหาข้อมูลเฉพาะช่วง (range search) เพราะจัดเรียงข้อมูลไว้แล้ว จึงไม่ต้องอ่านทุกรეกอร์ด ส่วนข้อเสียคือ ทุกรังที่เพิ่มหรือแก้ไขข้อมูล จะต้องจัดเรียงข้อมูลทั้งไฟล์ใหม่เสมอเพื่อให้ลำดับการเรียงถูกต้อง
3. ดัชนี (index) คือการสร้างดัชนีแยกไว้ต่างหากจากข้อมูลในตาราง ซึ่งอาจสร้างดัชนีหลายชุด ส่งผลให้การค้นหาทำได้รวดเร็ว และเมื่อเพิ่มหรืออัปเดตเรกอร์ดก็จะทำงานเร็วกว่าแบบไฟล์เรียงลำดับมาก เพราะอัปเดตเฉพาะดัชนี ไม่ใช่ทั้งไฟล์ข้อมูล โครงสร้างดัชนีมีแบบต้นไม้ (tree) กับแบบแฮช (hash)

#### 9.4 ดัชนี

ตัวอย่างของดัชนีในชีวิตจริงที่ไม่ใช่ในระบบฐานข้อมูล ขอให้นึกถึงสารบัญหนังสือ ที่ซึ่งไปยังหน้าที่เราสนใจ หรือดัชนีท้ายเล่มที่มีไว้ค้นหาเนื้อหาตามคำที่สนใจ

ดัชนีในฐานข้อมูลก็ใช้หลักการเดียวกัน คือมีการสร้างดัชนีไว้นอกเหนือข้อมูลจริง เพื่อให้เข้าถึงข้อมูลที่ต้องการได้รวดเร็วขึ้น โดยไม่จำเป็นจะต้องค้นหาข้อมูลทั้งตาราง ฐานข้อมูลสามารถจะมีดัชนีได้หลายชุดตามความต้องการด้วย

อย่างไรก็ตาม เราต้องเข้าใจว่าจะต้องมาจากการสร้างดัชนีด้วย คือการทำดัชนีต้องใช้พื้นที่เพิ่มขึ้น และเมื่อเกิดโภ泊เรียนใด ๆ กับข้อมูล ไม่ว่า UPDATE, INSERT หรือ DELETE ระบบการจัดการฐานข้อมูลก็จะคำนวณ

และจัดเรียงตัวนี้ใหม่ทุกครั้ง ทำให้สิ้นเปลืองเวลาในการประมวลผลด้วยตัวนั้น เราจึงต้องออกแบบการจัดเก็บและตัวนี้ และปรับปรุงให้เหมาะสมอยู่เสมอ

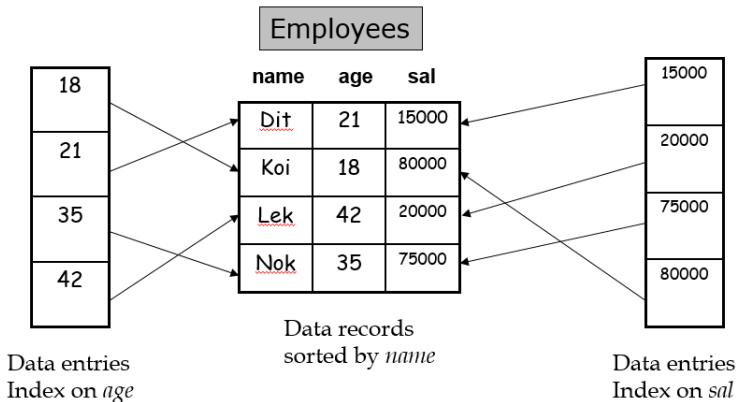
## 9.5 คีย์ค้นหาและรายการข้อมูล

เมื่อใช้ตัวนี้ เราสามารถค้นหาข้อมูลด้วย คีย์ค้นหา (search key) โดยระบุฟิลด์ใดฟิลด์หนึ่งในตารางให้เป็นคีย์ค้นหา ศัพท์คำว่า “คีย์ค้นหา” นี้ไม่เกี่ยวข้องกับคีย์ในบทอื่นที่ผ่านมา อาจสับสนได้ คือคีย์ค้นหาไม่มีคุณสมบัติที่ทำให้เพิ่ลหรือเรกคอร์ดไม่ซ้ำกัน คีย์ค้นหาคือฟิลด์ที่ผู้ออกแบบฐานข้อมูลเลือกระบุว่าเป็นฟิลด์ที่ใช้ค้นคืนข้อมูลบ่อย และต้องการสร้างตัวนี้สำหรับคีย์ค้นหานั้น

ให้แทนคีย์ค้นหาด้วย  $k$

ผลการค้นหาด้วย  $k$  จะได้รายการข้อมูล (data entries)  $k^*$  ซึ่งหมายถึงข้อมูลของคีย์ค้นหาที่นำมาทำตัวนี้

รูปที่ 108 แสดงตัวอย่างรีเลชัน Employees ประกอบด้วย 3 ฟิลด์คือ name, age และ salary และเรียงตาม name จากน้อยไปมาก จะเห็นว่ามี data entries  $k^*$  2 ชุดที่สร้างขึ้นมาเป็นตัวนี้ ได้แก่ data entries ที่ตัวนี้ด้วย  $k=age$  กับ data entries ที่ตัวนี้ด้วย  $k=sal$  ซึ่ง  $k^*$  แต่ละชุดจะจัดเรียงข้อมูลตาม  $k$  และเก็บพอยน์เตอร์ ซึ่งไปยังที่อยู่ของข้อมูลนั้น ซึ่งในรูปแสดงด้วยลูกศร



รูปที่ 108 รีเลชัน Employees

การเรียงข้อมูลจะทำทีละฟิลเตอร์ หากกำหนดไว้หลายฟิลเตอร์ ก็จะเรียงข้อมูลในฟิลเตอร์แรกก่อนแล้วค่อยเรียงฟิลเตอร์ต่อ ๆ มา

## 9.6 วิธีการสร้างรายการข้อมูล

การสร้างรายการข้อมูล หรือ data entries มี 3 วิธี

### 9.6.1 วิธีที่ 1 โครงสร้างไฟล์แบบตัวชี้

ให้นำข้อมูลจริงทุกrecordมาสร้างตัวชี้นี้ หรือพูดว่า  $k^*$  คือข้อมูลจริงทั้งหมด มาจัดเรียงตามคีย์ค้นหา  $k$  ที่กำหนด นั่นหมายความว่า  $k^*$  ก็จะมีได้ชุดเดียว เพราะเราเรียงข้อมูลจริงได้เพียงแบบเดียว หากมีrecordมาก จะส่งผลให้จำนวนเพจใน  $k^*$  สูงตามไปด้วย เราเรียกชื่อวิธีที่ 1 นี้ว่า โครงสร้างไฟล์แบบตัวชี้ (indexed file organization)

### 9.6.2 วิธีที่ 2 คู่ $\langle k, \text{rid} \rangle$

ให้สร้างรายการข้อมูล  $k^*$  ขึ้น โดยกำหนดคีย์ค้นหาและรหัสเรกคอร์ด  $\langle k, \text{rid} \rangle$  ที่ซึ่งไปยังเรกคอร์ดในข้อมูลจริง ดังนั้น  $k^*$  จะมีคู่  $\langle k, \text{rid} \rangle$  สำหรับทุก  $k$  ในตารางข้อมูลจริง จะเห็นว่าวิธีนี้เราจะไม่ยุ่งเกี่ยวกับข้อมูลที่อยู่ในไฟล์จริง มีเพียงพอยน์เตอร์จาก  $\text{rid}$  ซึ่งไปยังเรกคอร์ดในข้อมูลจริง

### 9.6.3 วิธีที่ 3 $\langle k, \text{rid-list} \rangle$

คล้ายวิธีที่ 2 แต่แทนที่จะสร้าง  $k$  กับ  $\text{rid}$  เป็นคู่ ก็ให้สร้างเป็นรายการหรือ  $\text{list}$   $\langle k, \text{rid-list} \rangle$  นั่นหมายถึง  $k^*$  วิธีที่ 3 นี้ สำหรับ  $k$  เดียวกัน อาจมี  $\text{rid}$  หลายค่าได้ ซึ่งไม่ปรากฏเป็นคู่เหมือนวิธีที่ 2 ดังนั้น วิธีนี้จะเหมาะสมกับ  $k$  ที่ปรากฏอยู่หลายที่ในตารางข้อมูลจริง

สรุปคือ วิธีที่ 2 และวิธีที่ 3 จะได้  $k^*$  ที่อิสระจากชุดข้อมูลจริง วิธีที่ 3 จะประหยัดพื้นที่กว่าวิธีที่ 2

## 9.7 ประเภทของดัชนี

เราสามารถแบ่งดัชนีได้หลายประเภท ขึ้นอยู่กับมุมมอง

หากพิจารณาจากคีย์ค้นหา จะแบ่งดัชนีได้ 2 ประเภท คือ

4. ดัชนีหลัก (primary index) หมายถึงคีย์ค้นหานั้นคือคีย์หลักหรือส่วนหนึ่งของคีย์หลักของรีเลชัน

5. ดัชนีรอง (secondary index) หมายถึงคีย์คันหนันไม่ใช่คีย์หลักของรีเลชัน รวมทั้งคีย์คู่ไปด้วย

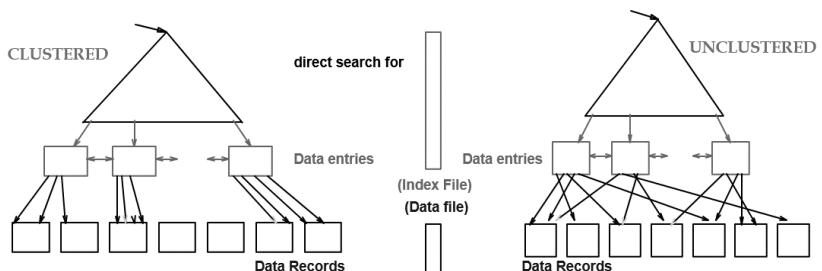
หากพิจารณาจากการเรียงของข้อมูลในดัชนี จะแบ่งดัชนีได้ 2 ประเภท คือ

6. ดัชนีคลัสเตอร์ (clustered index) หมายถึง ดัชนีที่รายการข้อมูล  $k^*$  มีลำดับการเรียงตัวเหมือนหรือคล้ายกันกับเรกคอร์ดในรีเลชัน
7. ดัชนีไม่คลัสเตอร์ (unclustered index)

จะเห็นได้ว่า การสร้างรายการข้อมูลด้วยวิธีที่ 1 คือแบบโครงสร้างไฟล์แบบดัชนีนั้น จะอนุญาตให้เป็นดัชนีแบบคลัสเตอร์แน่นอน

เวลาที่ใช้ในการค้นคืนข้อมูลอาจต่างกันได้มากเมื่อดัชนีเป็นแบบคลัสเตอร์ หรือไม่คลัสเตอร์

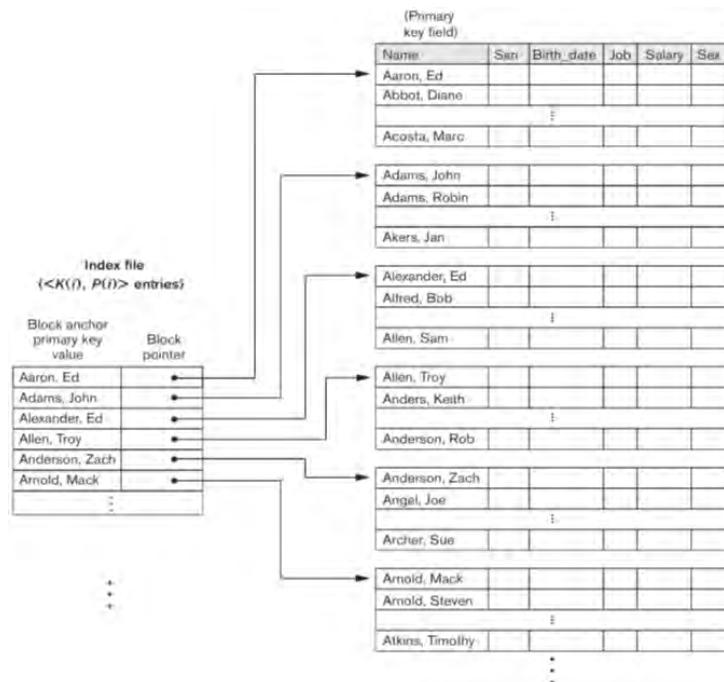
รูปที่ 109 เปรียบเทียบดัชนีคลัสเตอร์กับดัชนีไม่คลัสเตอร์



รูปที่ 109 ดัชนีคลัสเตอร์กับดัชนีไม่คลัสเตอร์

## ตัวอย่าง

รูปที่ 110 จาก [2] แสดงตัวอย่างดัชนีหลัก หมายถึง  $k^*$  ของดัชนีนี้ คือคีย์หลัก ของเริ่บซัน สังเกตว่า Name เป็นคีย์หลักของเริ่บซัน ดังนั้น  $k^*$  ก็คือ Name จากนั้นให้สังเกตว่ารายการข้อมูล  $k^*$  ซึ่งประกอบด้วย  $\langle \text{name}, \text{rid} \rangle$  นั้น ก็เรียงตาม name จึงทำให้ดัชนีเป็นแบบคลัสเตอร์โดยปริยาย



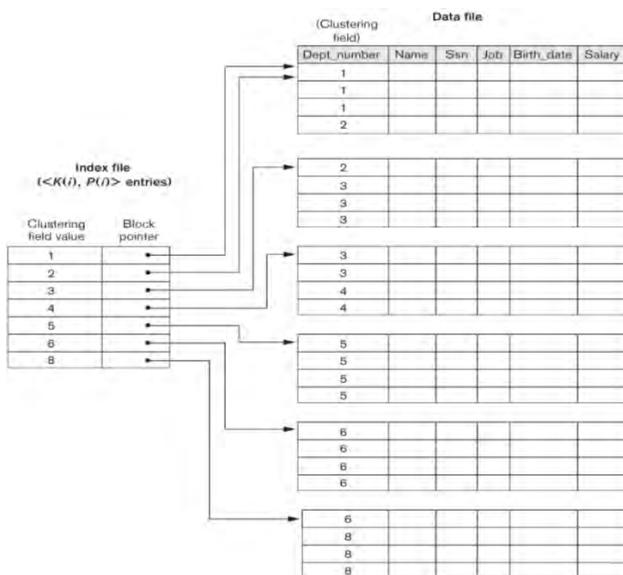
รูปที่ 110 ดัชนีหลัก

สังเกตรูปที่ 110 ต่อ จะเห็นว่า name ใน  $k^*$  นั้น สัมพันธ์กับข้อมูลแควรในเพจของเรกคอร์ดข้อมูล ดัชนีแบบนี้เรียกว่า ดัชนีเบาบาง (sparse index)

ที่  $k^*$  จะเก็บเฉพาะ rekord แรกของแต่ละเพจเท่านั้น เมื่อค้นหาข้อมูล ก็จะคำนวณหาที่อยู่ที่ต้นเพจแล้วค่อยหาในเพจต่อไปที

ดัชนีที่ตรงข้ามกับดัชนีเบาบาง เรียกว่า ดัชนีหนาแน่น (dense index) ซึ่ง  $k^*$  ที่เป็นดัชนีหลักนั้น จะต้องมีจำนวนเท่ากับจำนวนเรกคอร์ดในรีเลชัน และมี  $rid$  ซึ่งปัยหุ้นเรกคอร์ดที่  $k$  มีค่าเท่ากัน ดังนั้นจะมีจำนวนคู่  $\langle k, rid \rangle$  หรือ  $k^*$  จึงมีมากกว่าดัชนีเบาบางมาก แต่จะส่งผลให้ค้นหาข้อมูลได้เร็วกว่ามาก เช่นกัน ข้อเสียเมื่อเทียบกับดัชนีเบาบางก็คือ ดัชนีแน่นหนาจะใช้พื้นที่จัดเก็บใหญ่มาก

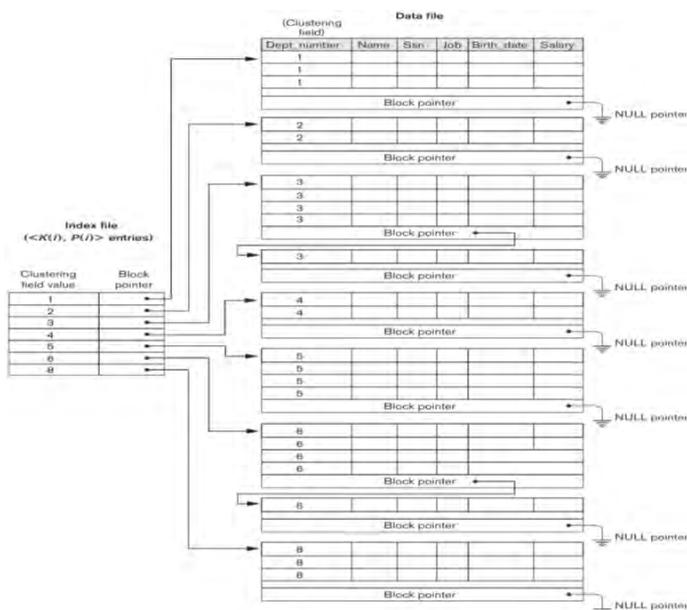
## ตัวอย่าง



รูปที่ 111 ดัชนีคลัสเตอร์

รูปที่ 111 จาก [2] แสดงตัวอย่างดัชนีคลัสเตอร์ แต่ไม่ใช่แบบดัชนีหลัก จะเห็นว่าลำดับของเรกคอร์ดในรีเลชันเรียงตาม Dept\_number และสร้างดัชนีให้ k คือ Dept\_number จะเห็นว่าเป็นดัชนีเบABAงอีกเช่นกัน เมื่อต้องการค้นหาคนที่อยู่ Dept\_number 1 จะมีบล็อกพอยน์เตอร์ซึ่งไปยังเรกคอร์ดแรกของเพจที่ Dept\_number = 1 เมื่อ เมื่อต้องการค้นหาคนที่อยู่ Dept\_number 2 บล็อกพอยน์เตอร์ก็ซึ่งไปยังเรกคอร์ดแรกของเพจที่ Dept\_number = 2 ซึ่งก็ยังเป็นเพจแรก เพราะเรกคอร์ดแรกที่ Dept\_number = 2 อยู่ในเพจนั้น ดังนั้น ดัชนีจึงเป็นแบบเบABAง

ตัวอย่าง



รูปที่ 112 ดัชนีคลัสเตอร์

ในรูปที่ 112 จาก [2] จะเห็นว่ามีการใช้ดัชนีคล้ายตัวอย่างก่อนหน้า ซึ่ง k คือ Dept\_number เช่นเดียวกัน เป็นดัชนีคลัสเตอร์เหมือนกัน ส่วนที่ต่างคือ แต่ละเพจนั้นจะเก็บเรกคอร์ดที่มีค่าเท่ากัน เช่นเพจแรกสุดเก็บเรกคอร์ด Dept\_number 1 เพจต่อมาเก็บเรกคอร์ด Dept\_number 2 ไปเรื่อย ๆ ถ้าเพจนั้นเต็มแล้วยังเก็บข้อมูลไม่พอ หมายถึงยังมี Dept\_number เท่าเดิมคงเหลืออยู่ ก็ให้อเวอร์โฟลลงไปเพจถัดไป แล้วสร้างพอยน์เตอร์ชี้ต่อไปยังเพจเพิ่มเติมนี้ด้วย เมื่อ Dept\_number มีค่าต่างไป ก็ให้ขึ้นเพจใหม่และย้อนกลับอยู่ที่ว่างในเพจนั้น พร้อมกับเซตตัลพอยน์ต่อรีวิวที่ท้ายเพจ วิธีดัชนีแบบนี้ ส่งผลให้การค้นหาข้อมูลทำได้เร็วขึ้น จากนั้นบล็อกพอยน์เตอร์จะชี้ตรงไปยังต้นเพจที่มี Dept\_number ที่ต้องการค้นหาได้ทันที แต่ใช้พื้นที่มากกว่า

## 9.8 โครงสร้างของดัชนี

ที่ผ่านมาเราเห็นตัวอย่างของดัชนีแบบคลัสเตอร์ ซึ่งตัวดัชนีจะเก็บคีย์ค้นหา k กับ rid แต่ถ้าเราเลือกทำดัชนีของรีเลชันแบบไม่คลัสเตอร์แล้ว เราจะต้องเลือกโครงสร้างของตัวดัชนี คือรายการข้อมูล  $k^*$  ด้วย ซึ่งตำรานี้จะกล่าวถึงโครงสร้างของดัชนี 2 แบบ คือ

1. โครงสร้างดัชนีแบบแฮช
2. โครงสร้างดัชนีแบบต้นไม้

### 9.8.1 โครงสร้างดัชนีแบบแฮช

โครงสร้างดัชนีแบบแฮช (hash-based index) เป็นการจัดเรียงข้อมูลโดยแยกเป็นถัง (bucket) แล้วแฮชรายการข้อมูล  $k^*$  ค่าด้วยคีย์ค้นหา k ดัชนีก็

จะเก็บรายการตั้งชื่อประกอบด้วย เพจหลัก (primary page) และพอยน์เตอร์ ที่เชื่อมไปยังโอลิวอร์ฟลูเพจถ้ามี

การจะหาว่าเรกคอร์ดที่ต้องการนั้นอยู่ที่ตั้งใด ให้คำนวณด้วยฟังก์ชันแฮช (Hashing function)

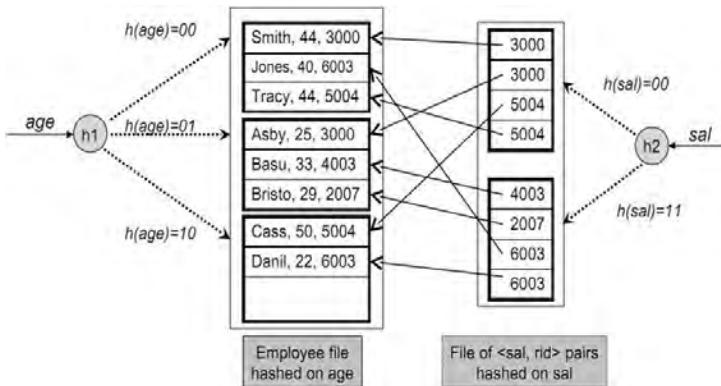
$$h: h(r)$$

จะได้หมายเลขของตั้งที่เก็บเรกคอร์ด  $r$

ในกรณีที่การสร้างรายการข้อมูลใช้วิธีที่ 1 คือใช้โครงสร้างไฟล์แบบดัชนีนั้น ในถังก็จะเก็บเรกคอร์ดข้อมูลจริง ถ้าใช้วิธีคู่  $\langle k, rid \rangle$  ข้อมูลในถังก็คือ คู่  $\langle k, rid \rangle$  ถ้าใช้วิธีคู่  $\langle k, rid-list \rangle$  ข้อมูลในถังก็คือ คู่  $\langle k, rid-list \rangle$

วิธีนี้หมายกับการค้นหาข้อมูลแบบเท่ากับ (equality search) จะทำได้เร็ว เพราะเราใช้โอเปเรชันเดียว แข็งคืบค้นหา แล้วจะทราบค่าถังของข้อมูลนั้น ทันที

อย่างไรก็ดี ดัชนีแบบแฮชไม่เหมาะสมกับการค้นหาข้อมูลแบบส่วน (range search) อย่างยิ่ง เพราะผลลัพธ์ของการแข็งค่าที่ใกล้เคียงกันนั้น อาจทำให้ ข้อมูลอยู่ห่างกันก็ได้



รูปที่ 113 พังก์ชันแฮช

รูปที่ 113 จาก [2] แสดงการใช้พังก์ชันแฮชเพื่อเก็บข้อมูล โดยแฮช 2 คีย์ค้นหา คือแฮช  $\text{age}$  และตามด้วยแฮช  $\text{sal}$

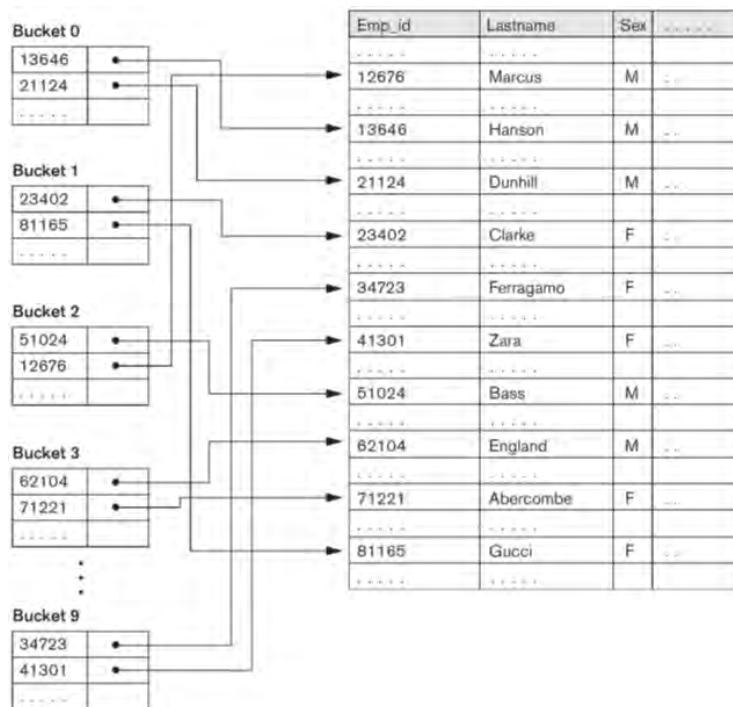
$h1: h(\text{age})$

$h2: h(\text{sal})$

ในการแฮช  $\text{age}$  นั้น ตัวอย่างนี้แฮชโดยการแปลง  $\text{age}$  เป็นเลขไบนาเรียแล้วตัดเลขมา 2 หลักสุดท้าย เช่น อายุ = 44 หรือ 40 จะได้บินาเรีย = 00 ( $x \bmod 4 = 0$ ) และนำ  $\text{age}$  ใด ๆ ที่ได้ค่าแฮช = 00 ไปเก็บไว้ในถังเดียวกัน ต่อมากะ เห็นว่า  $\text{age} = 25, 33, 29$  จะแฮชได้ 01 จึงนำข้อมูลที่แฮชแล้วได้ 01 ทั้งหมดไปไว้ในถังเดียวกัน และทำอย่างนี้ไปเรื่อย ๆ จะเห็นว่าข้อมูลในแต่ละถังนั้น จะได้  $\text{age}$  ที่กระจายกันไป ซึ่งจะส่งผลให้การกระจายตัวของจำนวนข้อมูลในเพจแต่ละเพจ ส่วนการแทรกข้อมูลเข้าไปในถังนั้น จะใช้วิธีแอปเพนด์ (append) คือผ่อนวงข้อมูลนั้นต่อท้ายโดยไม่ต้องจัดเรียงได ๆ

เมื่อต้องการค้นหาข้อมูล ก็ให้คำนวณว่าข้อมูลอยู่ที่ถังใด แล้วเข้าไปค้นหาในถังอีกที

มาลอง查詢 sal กันบ้าง จากตัวอย่างจะเห็นว่าเราใช้ฟังก์ชัน查詢เดียวกับที่ใช้กับ age ได้ผลการ查詢ที่เป็นได้ 2 ค่า คือ 00 กับ 01 จึงจัดข้อมูลในถังตามผล查詢



รูปที่ 114 ดัชนีหลัก

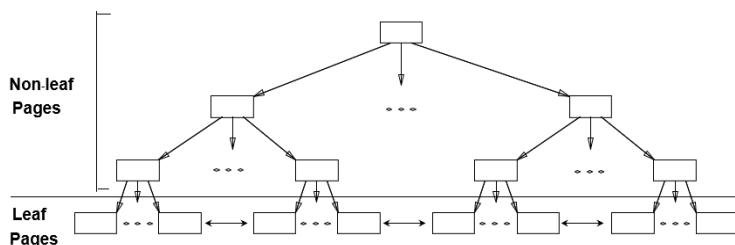
การ查詢ส่วนมากแล้วจะเป็นดัชนีรอง (secondary index) แต่ตัวอย่างในรูปที่ 114 จาก [2] นี้จะเห็นว่าเรียกอร์ดในรีเลชันเรียงตาม Emp\_id ซึ่งเป็นคีย์หลัก ดัชนีในกรณีจึงเป็นดัชนีหลัก ดังนั้นมือเราขอใช้ Emp\_id จะทำให้ Emp\_id ที่เดิมเรียงอยู่ติดกันกลับกระจายตัวออกด้วยผลของการ查詢 ผลก็จะได้ดัชนีแบบไม่คลัสเตอร์

### 9.8.2 โครงสร้างดัชนีแบบต้นไม้

โครงสร้างดัชนีแบบต้นไม้ (tree-based index) ที่ทำนานี้จะครอบคลุมคือ ต้นไม้บีพลัส (B+ tree) ซึ่งต่างจากโครงสร้างแบบแฮชอย่างมาก เพราะรายการข้อมูล  $k^*$  ของต้นไม้บีพลัสจะเรียงกันตามคีย์คันหาอยู่ก่อนแล้ว จากนั้นเราจะทำโครงสร้างต้นไม้ครอบรายการข้อมูลนี้อีกชั้น

โครงสร้างแบบต้นไม้บีพลัสนี้ จะมีความสมดุลอยู่เสมอ หมายถึงความสูงของต้นไม้จากราก (root) ถึงใบ (leaf) นั้นจะมีความยาวเท่ากันเสมอ คือเมื่อเพจนั้น ๆ เต็ม ก็จะแบ่งเพจ (r-split) และจัดเรียงไฟล์ใหม่ (reorganize)

ต่อไป ขอทับศพท์ ลีฟ (leaf) ดังนั้น เพจใบ จะเรียกว่า ลีฟเพจ (leaf page)

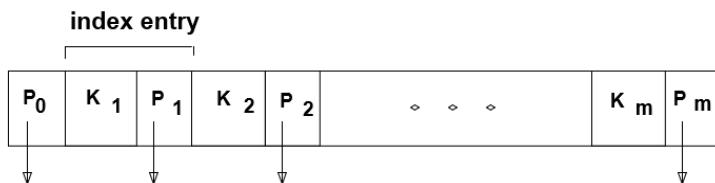


รูปที่ 115 โครงสร้างแบบต้นไม้บีพลัส

รูปที่ 115 จาก [2] แสดงอธิบายความสัมพันธ์ของลีฟเพจกับnonลีฟเพจ (non-leaf page) ในโครงสร้างต้นไม้บัลลส์

ข้อมูลในลีฟเพจอาจเป็นรายการข้อมูล (data entries)  $k^*$  ที่มีการโยงถึงกัน (chained) ในกรณีที่การสร้างรายการข้อมูลเป็นวิธีที่ 1 คือโครงสร้างไฟล์แบบดัชนี ข้อมูลตรงลีฟเพจจะเป็นเรกคอร์ดจริง

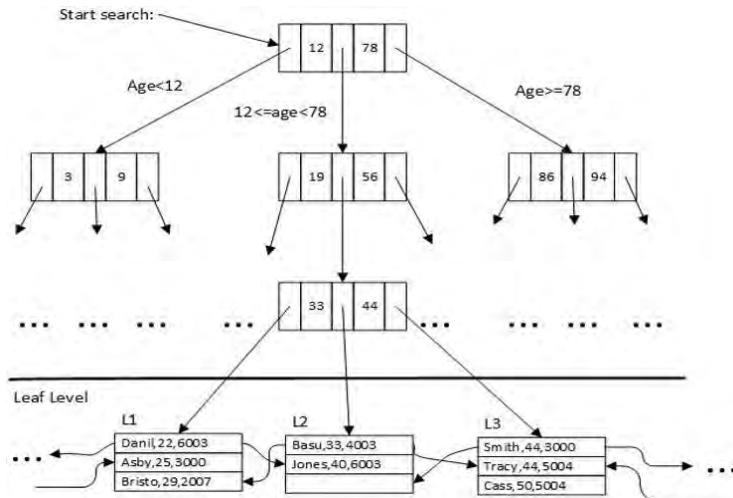
ส่วนข้อมูลที่อยู่ในnonลีฟเพจนั้นก็คือรายการดัชนี (index entries) ดังแสดงในรูปที่ 116 ประกอบด้วยพอยน์เตอร์คั่นด้วยคีย์คันหา  $k$  เพื่อให้เข้าไปยังลีฟเพจที่ถูกต้อง พอยน์เตอร์ซึ่งไปยังชั้นถัดไปด้านซ้ายเพื่อนำไปยังรายการข้อมูลมีค่าน้อยกว่า  $k$  และไปด้านขวาเมื่อรายการข้อมูลมากกว่า  $k$



รูปที่ 116 รายการดัชนี

รายการดัชนีแต่ละเพจ อาจมีขนาดใหญ่มากได้ อาจมากกว่า 100 รายการได้ ในทางปฏิบัติใช้กันอยู่ที่ 50-100 รายการ และมีพอยน์เตอร์จำนวนมากตามไปด้วย ในรูปแสดงด้วยลูกศร ทำให้สามารถรองรับเรกคอร์ดได้จำนวนมหาศาลโดยที่ความสูงของต้นไม้มีค่าน้อยได้ เช่นสามารถรองรับเรกคอร์ดจำนวนเป็นหลักล้านได้ด้วยความสูงของต้นไม้มีเงิน 4 ชั้น

คุณสมบัติสำคัญของต้นไม้บีเพลสคือจะมีฟิลแฟคเตอร์ (fill factor) อยู่ที่แต่ละเจเจ เพื่อรับรองจำนวนข้อมูลในเพจนั้น ๆ ให้มากกว่าที่ผู้ใช้กำหนด (user-defined) เช่น กำหนดว่าให้มีข้อมูลไม่ต่ำกว่า 50% ของเพจ เป็นต้น หากจำนวนข้อมูลน้อยกว่าที่กำหนด ก็จะรวมเพจเข้าด้วยกัน



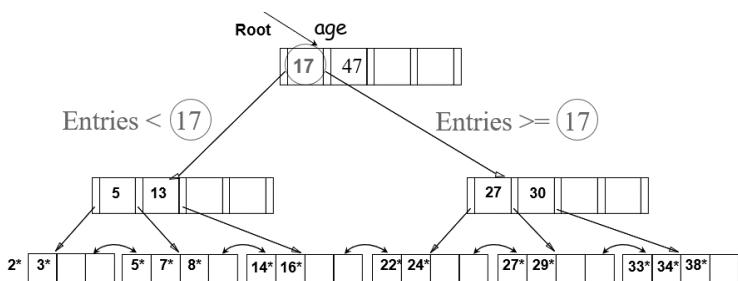
รูปที่ 117 การใช้ต้นไม้บีเพลส ด้วยตัวอย่าง

รูปที่ 117 จาก [2] แสดงตัวอย่างการใช้ต้นไม้บีเพลส ด้วยตัวอย่าง `age` เริ่มจาก根ที่มีช่องเก็บค่าคีย์ค้นหาได้ 2 ค่า สมมุติว่าเป็นค่าอายุ (`age`) จะเห็นว่าที่根นั้นเก็บค่า 12 กับ 78 และมีพอยน์เตอร์ชี้ไปยังชั้นถัดไป ถ้าซึ่งทางซ้ายหมายความว่าข้อมูลในเพจนั้นมีค่าน้อยกว่า 12 ไปด้านขวาหมายความว่ามากกว่า 78 ไปเรื่อย ๆ ดังนี้ จนกว่าจะถึงชั้นลีฟเพจ ก็จะพบรายการข้อมูลที่ต้องการ หากรายการนั้นมีอยู่ในฐานข้อมูล แต่ถ้าไม่พบรายการที่ต้องหา ณ เพจที่ควรจะอยู่ ก็สรุปได้ว่าเรกคอร์ดนั้นไม่ปรากฏในรีเลชัน คือไม่มีความ

จำเป็นได้ ๆ ที่จะค้นหาข้อมูลในเพจอื่นต่อไป เพราะข้อมูลถูกจัดเรียงไว้อยู่แล้ว

สังเกตพอยน์เตอร์ในระดับลิฟนั้นจะมีทั้งพอยน์เตอร์ที่ซ้ายไปยังเพจถัดไปและซ้ายกลับไปยังเพจก่อนหน้าด้วย

ตัวอย่าง



รูปที่ 118 ต้นไม้บีพลัส ให้  $age$  เป็น  $k$

รูปที่ 118 แสดงตัวอย่างต้นไม้บีพลัสที่ให้  $age$  เป็น  $k$  สังเกต \* ที่ชั้นลิฟหรือชั้นล่างสุด นั่นแสดงว่าข้อมูลชั้นนี้คือรายการข้อมูล

สมมุติว่าเราต้องการค้นหาคนที่อายุ 28 คือ  $k^* = 28$  ให้เริ่มดูที่ราก จะพบว่า 28 มากกว่า 17 แต่น้อยกว่า 47 จึงตามต้นไม้มงมาชั้นถัดไปด้านขวา ซึ่ง  $k = 27, 30, \dots$  ก็ให้ตามต้นไม้มงมาชั้นถัดไปด้านขวาอีกครั้ง เพราะ 28 มากกว่า 27 แต่น้อยกว่า 30 จึงพบ  $k^* = 27^*$  ตามด้วย 29 ก็ให้สรุปได้ทันทีว่าไม่มี  $k^*$  ที่มีค่าเท่ากับ 28\*

สมมุติว่าเราจะค้นหาช่วงของข้อมูล (range search) เช่น ต้องการค้นหาคนที่อายุมากกว่า 15 หรือ  $k^* > 15$  ก็เริ่มจาก根 เชนเดียวกัน แล้วตามต้นไม้ชั้นถัดไปด้านซ้าย เนื่องจาก  $15 < 17$  แล้วแยกต่อลงไปด้านขวาของ 13 พบรั้นลีฟ มี  $k^* = 16^*$  ก็จะเริ่มรู้คำตอบว่า  $k^*$  ตัวแรกคือ 16 แล้วให้ตามพอยน์เตอร์ไปจนจบไฟล์ (eof) ถือเป็นคำตอบ เพราะข้อมูลจัดเรียงไว้เรียบร้อยแล้ว

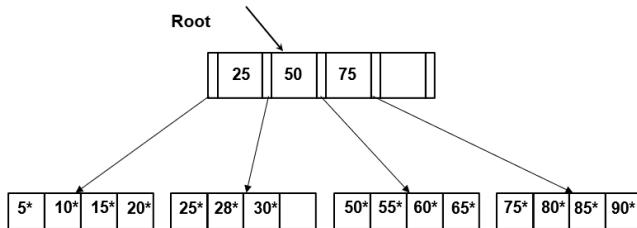
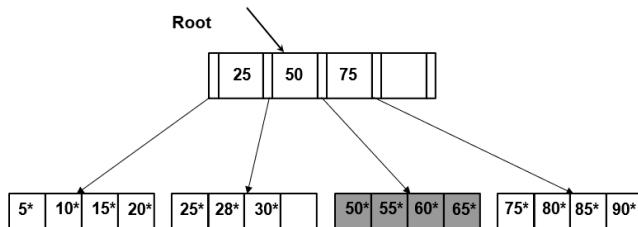
สมมุติว่าเราจะค้นหาช่วงของข้อมูล (range search) เช่น ต้องการค้นหาคนที่อายุมากกว่า 15 แต่น้อยกว่า 30 หรือ  $15 < k^* < 30$  ก็ให้ใช้วิธีเดิมหาจุดเริ่มของ  $k^* = 15^*$  และ  $30^*$  แล้วภาวดคำตอบพึ่พบตามพอยน์เตอร์ของทั้งสองค่านั่นเอง

จะเห็นว่าวิธีต้นไม้บีเพลสนี้ หมายความว่าการค้นหาช่วงของข้อมูล (range search) อย่างยิ่ง

คุณสมบัติอีกอย่างของต้นไม้บีเพลสคือมีแฟนเอาต์ (fan-out) หรือจำนวนเฉลี่ยของพอยน์เตอร์ต่อโหนด (node) ในชั้นบนลีฟ จำนวนมาก เมื่อเทียบกับต้นไม้บีเบนาเรียที่มีค่าแฟนเอาต์เท่ากับ 2

## 9.9 การปรับความสมดุลของต้นไม้บีเพลส

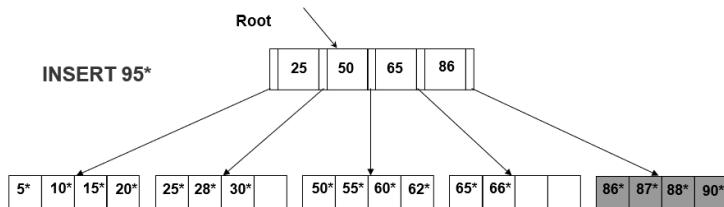
ทุกครั้งมีการอปเดต ลบ หรือแทรกข้อมูลลงในฐานข้อมูลที่ใช้ด้ชนี จะส่งผลให้ต้องจัดเรียงตัวชั้นใหม่ เนื่องจากคุณสมบัติของต้นไม้บีเพลสกำหนดว่าต้องมีความสมดุลเสมอ เราจึงต้องมีอัลกอริทึมในการปรับความสมดุลของต้นไม้บีเพลส ให้ดูตัวอย่างการแทรกข้อมูลในรูปที่ 119

**INSERT 70\***รูปที่ 119 แทรกข้อมูล  $age^* = 70^*$ 

สมมุติว่าเราจะแทรกข้อมูล  $age^* = 70^*$  ให้เริ่มจาก根 พบว่าจะต้องเพิ่มข้อมูลที่เพิ่งที่ 3 ของลีฟเพจ คือเพจที่มี  $\{50^*, 55^*, 60^*, 65^*\}$  ก็พบว่าเพจนี้เต็มแล้ว จึงให้แบ่งโหนด (split node) ที่เต็มนั้นออกจากกัน โดยคำนึงถึงค่าพิลแพคเตอร์ที่กำหนด สมมุติกำหนดเป็น 50% คือทุกโหนดต้องถูกใช้อย่างน้อย 50% จึงแบ่งได้เป็น  $\{50^*, 55^*, \dots\}$  กับ  $\{60^*, 65^*, \dots\}$  เพราะ 50% ของช่องสำหรับแต่ละโหนดในรูปนี้คือ 4 คีย์คันหา หรือ 5 พอยน์เตอร์ จากนั้นก็สามารถแทรก  $K^* = 70^*$  จะได้  $\{50^*, 55^*, \dots\}$  กับ  $\{60^*, 65^*, 70^*, \dots\}$  สุดท้าย เราจะต้องปรับดัชนีของโหนดที่มายังลีฟโหนดที่ถูกแบ่งด้วย ซึ่งเดิมนั้น โหนดที่ซึ่มามีค่า  $\{25, 50, 75, \dots\}$  จึงอาจเลือกค่าใด ๆ ระหว่าง 55 กับ 60 ก็ได้ แต่วิธีที่แนะนำคือให้เลือกกำหนดเป็น  $\{25, 50, 60, 75\}$  หมายถึงให้ใช้ค่าน้อยที่สุดจากโหนดที่ถูกแบ่งมา แต่จะเห็นได้ว่าการที่เพิ่ม 60 ไปใน

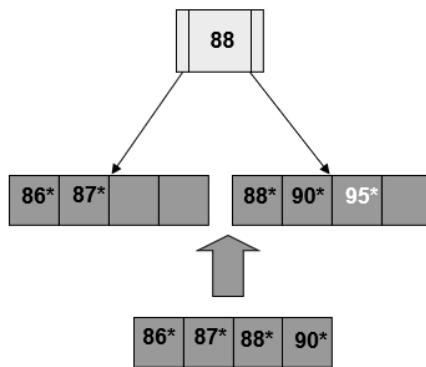
ดัชนีนี้ เราจำเป็นต้องจัดเรียงให้ลำดับถูกต้อง จึงต้องเลื่อน 75 ออกไปก่อนที่จะแทรก 60 และด้วยการย้ายพอยน์เตอร์ให้สอดคล้องกัน

ตัวอย่าง



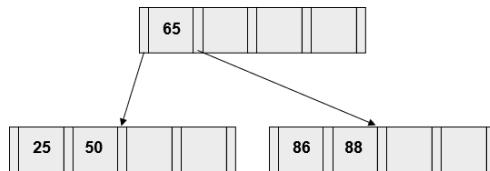
รูปที่ 120 แทรกข้อมูล  $age^* = 95^*$

ตัวอย่างจากรูปที่ 120 นี้จะแสดงเหตุการณ์เมื่อจำเป็นต้องเพิ่มขั้นของต้นไม้ในกรณีนี้ สมมุติว่าเราอยากราบรากข้อมูล  $age^* = 95^*$  ซึ่งพบว่า ต้องเพิ่มที่ลิฟ เพจสุดท้าย แต่เพจเต็ม ก็ให้แบ่งเพจนั้น ดังรูปที่ 121



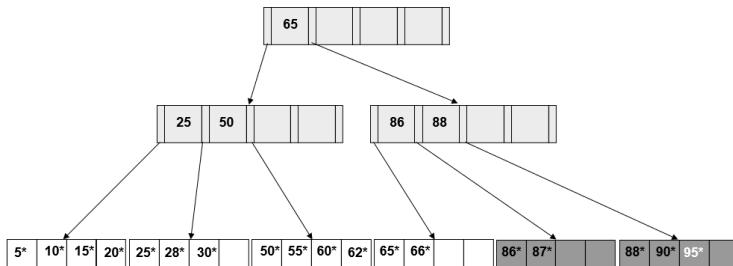
รูปที่ 121 เพิ่มขั้นของต้นไม้

หลังจากนั้น ให้เพิ่ม 88 ซึ่งคือค่า�้อยที่สุดของเพจใหม่ ไปยังโนนดที่ชื่\_ma เพจที่ 88 อยู่ ซึ่งก็พบว่าเต็ม จึงต้องแบ่งอีก ดังรูปที่ 122



รูปที่ 122 โนนดเพิ่มขั้น

จะเห็นว่าขั้นตอนนี้ทำให้เกิดโนนดเพิ่มขั้น เพราะหากถูกแบ่ง และในกรณีนี้ การกำหนดค่าให้พอยน์เตอร์จะทำต่างจากวิธีเดิม สังเกตว่า ถ้าเป็นวิธีเดิมนั้น k แรกของโนนดด้านขวาจะมีค่า 65 เพราะเราแบ่ง  $\{25, 50, 65, 86\}$  ออกจากกัน แล้วครรจะได้  $\{25, 50, \dots\} \{65, 86, \dots\}$  แต่ถ้าเป็นกรณีการเกิดขั้นเพิ่ม คือถูกแบ่ง เราจะผลักค่า k น้อยสุดจากโนนดที่ถูกแบ่งขึ้นไปอยู่ขั้นต้นไม่ที่สูงขึ้นนั้นได้ทันที ผลลัพธ์จึงได้อวย่างที่แสดงในรูปที่ 123



รูปที่ 123 ต้นไม้บีพลัสสมดุล

ด้วยวิธีนี้ ต้นไม้ปีพลัสก็จะสมดุลได้ตลอดเวลา อย่างไรก็ดี หากมีการลบข้อมูลไปถึงระดับหนึ่ง ก็มีความเป็นไปได้ว่าโหนดจะถูกเชื่อมเพราะมีจำนวนข้อมูลน้อยกว่าพิลแฟคเตอร์ และอาจส่งผลให้ชั้นของต้นไม้ลดลงได้ แต่ก็จะยังสมดุลเสมอ

## 9.10 การคำนวณเวลา

การจัดข้อมูลและการสร้างตัวชี้นี้แต่ละแบบ จะส่งผลให้การทำงานของฐานข้อมูลต่างไปโดยเฉพาะในด้านของความเร็ว เราจะวิเคราะห์ความสัมเปลือง (cost) หรือเวลา ในแบบเพจໄโอโอะแบบเฉลี่ย และจะไม่นำเวลาซีพียูมาคำนวณในขั้นตอนนี้ ตัวแปรที่เราจะนำมาใช้คำนวณเวลามี 3 ตัวประดิษฐ์

B : จำนวนเพจข้อมูล (data page)

R : จำนวนเรกคอร์ดต่อเพจ

D : เวลาเฉลี่ยในการอ่านหรือเขียนคิสก์

ในการวิเคราะห์แบบเฉลี่ย (average case analysis) นี้จะไม่คำนึงกรณีที่ข้อมูลบางเพจอยากถูกดึงมาพร้อมไปด้วยนั้นเพจໄโอโอก็จะเป็นค่าประมาณเท่านั้น

การจัดโครงสร้างไฟล์ที่เราคำนวณมาพิจารณาคำนวณ มีอยู่ด้วยกัน 5 แบบคือ

- ไฟล์ชีป หรือไฟล์แบบไม่เรียงลำดับ หมายถึงข้อมูลในไฟล์ไม่มีการจัดเรียงใดๆ (random order) การแทรกข้อมูลให้แทรกที่ท้ายไฟล์เสมอ
- ไฟล์แบบเรียงลำดับ เช่น เรียงลำดับตาม `<age, sal>`

3. ไฟล์แบบคลัสเตอร์ด้วยต้นไม้บีพลัส คือใช้โครงสร้างไฟล์แบบดั้งนี้ และมีคีย์ค้นหา เช่น  $\langle age, sal \rangle$
4. ไฟล์ชีปแบบไม่คลัสเตอร์และดัชนีแบบต้นไม้บีพลัส ด้วยคีย์ค้นหา เช่น  $\langle age, sal \rangle$
5. ไฟล์ชีปแบบไม่คลัสเตอร์และดัชนีแบบแฮชด้วยคีย์ค้นหา เช่น  $\langle age, sal \rangle$

โอเปอเรชันที่นำมาคำนวณ มี 5 โอเปอเรชัน คือ

- a) สแกน (scan) คือให้ดึงนำเรกคอร์ดทั้งหมดจาก迪สก์
- b) ค้นหาแบบเท่ากับ (equality search)
- c) ค้นหาแบบช่วง (range selection)
- d) แทรกเรกคอร์ด (insert a record)
- e) ลบเรกคอร์ด (delete a record)

ในการคำนวณนี้ เพื่อให้มีขับซ้อนเกินไป ขอตั้งสมมุติฐานดังนี้

1. ไฟล์ชีป
  - ค้นหาค่าเท่ากันด้วยคีย์ค้นหา เมื่อค้นพบเพียงตัวแรกก็ให้หยุดนับเวลา
2. ไฟล์แบบเรียงลำดับ
  - ไฟล์จะถูกกระซับหรืออัดแน่น (compact) หลังการลบเพื่อไม่ให้มีช่องโหว่ในไฟล์

### 3. ดัชนี

- สำหรับวิธีที่ 2 และวิธีที่ 3 ให้ขนาดรายการข้อมูลเป็น 10% ของเรกคอร์ด นั่นเป็นเพราะว่าดัชนีความมีขนาดเล็กกว่าข้อมูลจริงมาก ๆ และไม่ได้ต้องนำมาใช้ทั้งหมด นั่นคือเราเลือกใช้ดัชนีแบบเบาบาง (sparse index)
- การแข็ง เพื่อป้องกันการเกิดโอลูฟลู ให้ออกแบบไม่ให้มีข้อมูลเต็มเพจตั้งแต่แรก คือให้เหลือเนื้อที่ว่างพอให้แทรกข้อมูลได้ เราเรียกสถานการณ์นี้ว่า อัตราการครอบครองพื้นที่ (occupancy rate) เพื่อให้การคำนวนไม่ซับซ้อน เราจะใช้อัตราการครอบครองพื้นที่ที่ 80% ซึ่งจะส่งผลให้ไฟล์ (file size) มีขนาด 1.25 เท่าของขนาดข้อมูล (data size) ซึ่งคิดมาจากว่าขนาดไฟล์ต้องมีขนาดเท่าใดจึงจะรองรับข้อมูลได้ทั้งหมด ก็คือขนาดไฟล์ =  $100 \times 100 / 80 = 1.25$
- เมื่อใช้ดันไม้ ก็ต้องคำนึงถึงพิลแฟคเตอร์ ว่าแต่ละเพจควรใช้พื้นที่อย่างน้อยเท่าใด หรือมีอัตราการครอบครองพื้นที่เท่าใดในกรณีนี้ เราจะใช้อัตราการครอบครองพื้นที่ที่ 2/3 หรือ 67% ดังนั้นไฟล์จะมีขนาดเท่ากับ 1.5 เท่าของขนาดข้อมูล หรือมองว่าขนาดไฟล์ =  $3/2$  หรือ 1.5 เท่าของขนาดข้อมูล

ตารางที่ 4 แสดงผลการวิเคราะห์เวลาที่ใช้สำหรับแต่ละโอลูเรชัน a, b, c, d, e ในการทำงานกับไฟล์ที่มีโครงสร้างต่างกัน 5 แบบ 1, 2, 3, 4, 5

## ตารางที่ 4 ผลการวิเคราะห์เวลา

	(a) Scan	(b) Equality	(c) Range	(d) Insert	(e) Delete
(1) Heap	BD	0.5BD	BD	2D	Search +D
(2) Sorted	BD	Dlog 2B	Dlog 2 B + # matches	Search + BD	Search +BD
(3) Clustered	1.5BD	Dlog F 1.5B	Dlog F 1.5B + # matches	Search + D	Search +D
(4) Unclustered Tree index	BD(R+0.15)	D(1 + log F 0.15B)	Dlog F 0.15B + # matches	D(3 + log F 0.15B)	Search + 2D
(5) Unclustered Hash index	BD(R+0.125)	2D	BD	4D	Search + 2D

1(a) การสแกนไฟล์อีป สำหรับไฟล์อีปนั้น เนื่องจากไม่มีการเรียงข้อมูลใด ๆ การสแกนจึงต้องอ่านข้อมูลทั้งหมด คือ B เพจ แต่ละเพจใช้เวลาอ่านเท่ากับ D ดังนั้น การสแกนไฟล์อีปจึงใช้เวลา BD ที่ไม่ต้องคำนึงถึง R เป็น เพราะเราต้องอ่านทั้งเพจไม่ว่าจะมีกี่เรกคอร์ด

1(b) การค้นหาข้อมูลแบบเท่ากับจากไฟล์อีป เนื่องจากเป็นไฟล์อีปที่ไม่มีการจัดเรียง เราจึงจำเป็นต้องค้นหาตั้งแต่ต้นไฟล์ (sequential search) จนกว่าจะพบข้อมูลที่ต้องการ กรณีที่ต้องสุดคืออ่านเพจแรกแล้วพบข้อมูลที่ต้องการทันที คืออ่าน 1 เพจ กรณีเมื่อสุดคืออ่านจนเพจสุดท้าย คืออ่าน B เพจ แต่ด้วยว่าเราใช้วิเคราะห์แบบเฉลี่ย จึงเฉลี่ยว่าต้องอ่านข้อมูล 0.5B จึงจะพบข้อมูลที่ต้องการ แต่ละเพจใช้เวลาอ่านเท่ากับ D ดังนั้น การค้นหาข้อมูลแบบเท่ากับจากไฟล์อีปจึงใช้เวลา 0.5BD

1(c) การค้นหาข้อมูลแบบช่วงจากไฟล์อีป เนื่องจากไม่มีการเรียงข้อมูลใด ๆ การค้นหาข้อมูลแบบช่วงจึงต้องอ่านข้อมูลทุกเพจ คือ B เพจ เพราะข้อมูลที่

ต้องการอ่านอยู่ท้ายไฟล์ได้ แต่ละเพจใช้เวลาอ่านเท่ากับ D ดังนั้น การค้นหาข้อมูลแบบช่วงไฟล์ชีปจึงใช้เวลา BD

1(d) การแทรกข้อมูลในไฟล์ชีป เนื่องจากไฟล์ไม่จัดเรียง การแทรกข้อมูลจึงทำได้ด้วยการต่อท้ายไฟล์ (append at eof) ได้ทันที ในโครงสร้างของไฟล์ชีปจะมียอดเดอร์ (header) ระบุที่อยู่ของเพจ ทำให้เราสามารถเข้าถึงเพจที่มีที่ว่างได้ทันที การจะแทรกข้อมูลได้นั้น จะต้องอ่านเพจที่มีที่ว่างนั้นมาใส่ไว้ในหน่วยความจำก่อน การอ่านใช้เวลาเท่ากับ D และหากแทรกข้อมูลไว้ที่เพจนั้น ขั้นต่อไปก็ต้องเขียนเพจนั้นกลับลงบน迪สก์ การเขียนใช้เวลาเท่ากับ D สรุปโดยเปอเรชันนี้จึงใช้เวลาทั้งหมด 2D

1(e) การลบข้อมูลในไฟล์ชีป ก่อนจะลบข้อมูลได้ ก็ต้องค้นหาข้อมูลที่จะลบเสียก่อน จึงใช้เวลาในการค้นหาเท่ากับ BD เมื่อพบข้อมูลที่จะลบแล้วก็ทำโอเปอเรชันลบได้พระโภเปอเรชันค้นหาได้นำเพจนั้นมาอยู่ในหน่วยความจำแล้ว เสร็จแล้วก็ต้องเขียนเพจนั้นกลับ ใช้เวลาเท่ากับ D โดยรวมโอเปอเรชันนี้ใช้เวลาเท่ากับ Search + D หรือ BD + D

2(a) การสแกนไฟล์แบบเรียงลำดับ อย่างไรก็ต้องสแกนทั้งไฟล์ จึงต้องอ่านข้อมูลทั้งหมด คือ B เพจ แต่ละเพจใช้เวลาอ่านเท่ากับ D ดังนั้น โอเปอเรชันนี้จึงใช้เวลา BD

2(b) การค้นหาข้อมูลแบบเท่ากับจากไฟล์แบบเรียงลำดับ เนื่องจากเป็นไฟล์ที่เรียงข้อมูลมาแล้ว จึงสามารถค้นหาข้อมูลได้จ่าย ถ้าเรามีทั้งหมด B เพจ แล้วค้นหาแบบไบนารี ก็จะต้องอ่านเพจ  $\log_2 B$  เพจ ใช้เวลาเพจละ D จึงใช้เวลาทั้งหมด  $D(\log_2 B)$

2(c) การค้นหาข้อมูลแบบช่วงจากไฟล์แบบเรียงลำดับ ให้เริ่มหาข้อมูลตัวแรก ด้วยวิธีเดียวกับการค้นหาข้อมูลแบบเท่ากับ จึงใช้เวลาค้นหาตัวแรกเท่ากับ  $D(\log_2 B)$  จากนั้นก็ยังต้องค้นหาข้อมูลต่อ ถ้าข้อมูลที่ต้องการอยู่ที่เพจเดียวกัน เวลาที่ใช้หั้งหมวดก็จะเท่าเดิม แต่เนื่องจากเราไม่สามารถคาดเดาได้ว่าข้อมูลที่อยู่ในช่วงที่ต้องการนั้นมีจำนวนเท่าใดและอยู่ในกี่เพจ จึงคำนวณเวลาแบบคร่าวๆ ไว้ว่า เวลาหั้งหมวดที่ใช้ในโอบีอูนี้คือ  $D(\log_2 B)$  รวมกับจำนวนข้อมูลที่พบ หรือ  $D(\log_2 B) + \#matches$

2(d) การแทรกข้อมูลในไฟล์แบบเรียงลำดับ เนื่องจากไฟล์จัดเรียงไว้ การแทรกข้อมูลจึงทำไม่สามารถต่อท้ายไฟล์ได้ เราจำเป็นต้องหาที่ที่จะแทรกข้อมูล ซึ่งการค้นหาที่แทรกจะใช้เวลาการค้นหาข้อมูลแบบเท่ากับ แต่ก่อนจะแทรกข้อมูลได้ เราจะต้องสร้างช่องว่างให้แทรกข้อมูลใหม่นั้นได้ สมมุติว่าที่แทรกที่่เหมาะสมนั้นอยู่ท้ายไฟล์ เราจะไม่ต้องสร้างช่องว่างเลย แต่ถ้าที่แทรกอยู่ต้นไฟล์ นั่นหมายความว่าเราต้องเลื่อนข้อมูลทั้งไฟล์ให้มีช่องว่างต้นไฟล์ให้ได้ก่อนจะแทรก ในกรณีที่เราใช้เคราะห์แบบแลี่ย ให้คิดว่าเราจะแทรกที่กลางไฟล์ คือต้องเลื่อนข้อมูลครึ่งไฟล์ออกทีละหนึ่ง ทุกครั้งที่เลื่อนข้อมูลเพื่อสร้างช่องนี้ เราจะต้องอ่านเพจแล้วเขียนกลับ ดังนั้น เราอ่านเพจ 0.5B เพจ และเขียน 0.5B เพจ รวมกับได้ B เพจ ใช้เวลา D ต่อเพจ จึงจะใช้เวลาหั้งหมวดคือ Search + BD

2(e) การลบข้อมูลในไฟล์แบบเรียงลำดับ ก่อนจะลบข้อมูลได้ ก็ต้องค้นหาข้อมูลที่จะลบเสียก่อน จึงใช้เวลาในการค้นหาแบบเท่ากับ จากนั้น เมื่อพบข้อมูลที่จะลบแล้วก็ทำโอบีอูนลบได้ ซึ่งก็จะทำให้เกิดช่องว่างในเพจ เพราะข้อมูลถูกลบไป ดังนั้น เราจึงต้องกระชับไฟล์ด้วยวิธีเดียวกับการแทรก

คือเลื่อนข้อมูลทีละตัวจนครบ ต้องอ่าน 0.5B และเขียนทีละเพจ 0.5B โดย  
เฉลี่ยแล้วจึงใช้เวลาเท่ากับการแทรกข้อมูล คือ Search + BD

3(a) การสแกนไฟล์แบบคลัสเตอร์ ไฟล์แบบคลัสเตอร์นี้ลักษณะคล้ายต้นไม้  
บีพลัสที่ลีฟเพจนั้นเก็บเรกคอร์ดจริง ดังนั้นเนื่องจากโครงสร้างเป็นแบบต้นไม้  
เราจะต้องคำนึงถึงพิลแฟคเตอร์ที่เรากำหนดให้เป็น 2/3 หรือ 67% ทำให้  
จำนวนเพจที่ต้องสแกนหักหมดเพิ่มเป็น 1.5 เท่าจากของเดิม คือ 1.5B  
การอ่านข้อมูลใช้เวลาเพจละ D จึงทำให้อีกเพชันนี้ใช้เวลาหักหมด 1.5BD

3(b) การค้นหาข้อมูลแบบเท่ากับจากไฟล์แบบคลัสเตอร์ จะคล้ายกับ<sup>จะคล้ายกับ</sup>  
การค้นหาข้อมูลแบบช่วงจากไฟล์แบบเรียงลำดับ แต่เนื่องจากเราไม่สามารถ  
ใช้การค้นหาแบบใบnaire แล้ว ด้วยเหตุผลว่าต้นไม้เป็นแบบบีพลัส ทำให้  
พอยน์เตอร์ไม่เท่ากับสองเหมือนเดิม ซึ่งต้นไม้แบบบีพลัสหรือพอยน์เตอร์  
เท่ากับจำนวนแฟ้มเอกสาร แทนด้วย F ดังนั้น เวลาที่ใช้ในการค้นหา B เพจ จะ  
เป็น  $\log_F B$  แทน  $\log_2 B$  นอกจากนี้จากการคำนวณ จำนวนเพจในกรณีนี้เป็น 1.5B  
การอ่านข้อมูลใช้เวลาเพจละ D จึงทำให้เวลาหักสิ้นมีค่าเท่ากับ D( $\log_F 1.5B$ )

3(c) การค้นหาข้อมูลแบบช่วงจากไฟล์แบบคลัสเตอร์ ให้คิดเหมือนกับ<sup>จะคล้ายกับ</sup>  
การค้นหาข้อมูลแบบเท่ากับจากไฟล์แบบคลัสเตอร์ แล้วเวลาในการหา  
ค่าอื่นในช่วง ทำให้เวลาหักสิ้นมีค่าเท่ากับ D( $\log_F 1.5B$ ) + #matches

3(d) การแทรกข้อมูลในไฟล์แบบคลัสเตอร์ ก็ต้องเริ่มจากการค้นหาแบบ  
เท่ากับเสียก่อน พบแล้วก็แทรกข้อมูลได้เลย เพราะเราได้ເຝື່ອທີ່ວ່າງຈາກ  
พิลแฟคเตอร์ໄວ້แล้ว การเขียนข้อมูลคืนจะใช้เวลาเท่ากับ D จึงทำให้เวลา  
หักสิ้นมีค่าเท่ากับ Search + D

3(e) การลบข้อมูลในไฟล์แบบคลัสเตอร์ เนื่องจากข้อมูลในไฟล์แบบคลัสเตอร์ทุกรอบวนการ เพราะเมื่อค้นหาข้อมูลที่จะลบได้แล้วก็ทำไปเป็นรูปแบบนี้ได้ จากนั้นก็ต้องเขียนข้อมูลคืน ใช้เวลาเท่ากับ D แต่ไม่ต้องเลื่อนข้อมูลเพื่อกระชับไฟล์ เนื่องจากเป็นต้นไม้และมีพิลแฟคเตอร์ ทำให้เวลาที่ต้องใช้สิ้นเปลืองกับ Search + D

4(a) การสแกนไฟล์แบบไม่คลัสเตอร์ที่ใช้ดัชนีต้นไม้บีพลัสด ให้คิดว่าข้อมูลจริงคือไฟล์ชิป แล้วเราสร้างดัชนีแบบต้นไม้เอาไว้ เนื่องจากไฟล์นี้ไม่คลัสเตอร์ นั่นหมายความว่าการเรียงของเรกคอร์ดไม่สอดคล้องกับรายการข้อมูลในดัชนีต้นไม้ และเนื่องจากเรากำหนดให้ต้นไม้บีพลัสมีพิลแฟคเตอร์ 2/3 นั่นคือจำนวนเพจก็จะเป็น 1.5B เพจ แต่เนื่องจากเรากำหนดไว้ว่าเมื่อใช้วิธีดัชนีเบางาน คือจะเลือกเพียง 10% ของเรกคอร์ดจริงมาทำดัชนี ทำให้ขนาดเพจที่เราต้องสแกนจะเหลือเพียง 10% ของ 1.5B คือ 0.15B คูณด้วยเวลา D ต่อเพจ จึงได้เวลาที่ใช้ในการอ่านดัชนีคือ 0.15BD

แต่เนื่องจากการทำงานแบบใช้ดัชนีนี้ เรายังคงสแกนดัชนีก่อนแล้วก็ตามไปอ่านเรกคอร์ดที่ดัชนีนั้นชี้ไป และเพราะเป็นไฟล์แบบไม่คลัสเตอร์ การอ่านจึงเกิดขึ้นที่ละเรกคอร์ด ไม่ใช่อ่านที่ละเพจ มีจำนวน R เรกคอร์ดต่อเพจ อ่านทั้งหมด B เพจ จึงจะต้องใช้เวลาในการอ่านเรกคอร์ดทั้งหมดเท่ากับ RBD

รวมทั้งการสแกนดัชนีและอ่านเรกคอร์ดทั้งหมด จะใช้เวลาทั้งสิ้น RBD + 0.15BD หรือ BD(R+0.15)

4(b) การค้นหาข้อมูลแบบเท่ากับจากไฟล์แบบไม่คลัสเตอร์ที่ใช้ดัชนีต้นไม้บีพลัส ขั้นตอนการค้นหาข้อมูลก็ไม่ต่างไปจากการค้นหาข้อมูลแบบเท่ากับจากไฟล์แบบคลัสเตอร์ แต่เนื่องจากเราเริ่มต้นที่  $0.15B$  เพจ แทน  $1.5B$  เพจ แบบก่อนหน้า ทำให้เวลาในการค้นหาเป็น  $D(\log_F 0.15B)$  ตอนนี้เราจะได้พอยน์เตอร์จากดัชนีซึ่งปะยังข้อมูลที่ต้องการค้นหา ดังนั้น เราจึงใช้เวลาในการอ่านเพียงเพจนั้นเพจเดียวเท่ากับ  $D$  ส่งผลให้เวลาทั้งสิ้นเป็น  $D + D(\log_F 0.15B)$  หรือ  $D(1+\log_F 0.15B)$

4(c) การค้นหาข้อมูลแบบช่วงจากไฟล์แบบไม่คลัสเตอร์ที่ใช้ดัชนีต้นไม้บีพลัส ให้คิดเหมือนกับการค้นหาข้อมูลแบบเท่ากับจากไฟล์แบบไม่คลัสเตอร์ แล้ว บวกเวลาในการหาค่าอื่นในช่วง ทำให้เวลาทั้งสิ้นมีค่าเท่ากับ  $D(\log_F 0.15B) + \#matches$

4(d) การแทรกข้อมูลในไฟล์แบบไม่คลัสเตอร์ที่ใช้ดัชนีต้นไม้บีพลัส เนื่องจาก rekord เป็นไฟล์อิป การแทรกใช้เวลาเท่ากับ  $2D$  เพื่อเขียนข้อมูลเข้าที่ท้ายไฟล์ ใช้เวลาอ่านเท่ากับ  $D$  และเขียนเท่ากับ  $D$  จากนั้น ก็ต้องมาอัปเดตพอยน์เตอร์ที่ต้นไม้ให้มีพอยน์เตอร์ซึ่งปะยังข้อมูลที่แทรกใหม่ให้ได้ จึงต้องค้นหาเพจที่ต้องอัปเดตพอยน์เตอร์ ใช้เวลาค้นหาเท่ากับ  $D(1+\log_F 0.15B)$  ได้เพจที่ต้องอัปเดตมาอยู่ในหน่วยความจำ แล้วเขียนพอยน์เตอร์ใช้เวลาเท่ากับ  $D$  ทำให้เวลาทั้งสิ้นมีค่าเท่ากับ  $D + D(1+\log_F 0.15B) + 2D$  หรือ  $D(3+\log_F 0.15B)$

4(e) การลบข้อมูลในไฟล์แบบไม่คลัสเตอร์ที่ใช้ดัชนีต้นไม้บีพลัส ก็เริ่มตัวยการค้นหาเสียก่อน ก็จะได้ rekord ที่จะลบมาอยู่ในหน่วยความจำ เมื่อลบแล้วก็ต้องเขียนกลับ ใช้เวลาเขียน =  $D$  จากนั้นก็ต้องไปอัปเดตพอยน์เตอร์ในดัชนี ใช้เวลา  $D$  จึงทำให้เวลาทั้งสิ้นมีค่าเท่ากับ  $\text{Search} + 2D$

5(a) การสแกนไฟล์แบบไม่คลัสเตอร์ที่ใช้ดัชนีแบบแซช ข้อมูลเรกคอร์ดจริงคือไฟล์ชิปเหมือนแบบที่แล้ว แต่เราสร้างดัชนีแบบแซชไว้ เนื่องจากไฟล์นี้ไม่คลัสเตอร์ นั่นหมายความว่าการเรียงของเรกคอร์ดไม่สอดคล้องกับรายการข้อมูลในดัชนีแบบแซช และเนื่องจากเรากำหนดให้แซชมีอัตราการครอบครองพื้นที่เป็น 80% จึงส่งผลให้จำนวนเพจเพิ่มเป็น 1.5B เพจ แต่เนื่องจากเรากำหนดไว้ว่าเมื่อใช้วิธีดัชนีเบาบาง คือจะเลือกเพียง 10% ของเรกคอร์ดจริงมาทำดัชนี ทำให้ขนาดเพจที่เราต้องสแกนจะเหลือเพียง 10% ของ 1.25B เพจ คือ 0.125B เพจ คุณด้วยเวลา D ต่อเพจ จึงได้เวลาที่ใช้ในการสแกนดัชนีคือ 0.125BD

เข่นเดียวกับแบบที่แล้ว เมื่อเราสแกนดัชนีแล้วก็ต้องตามไปอ่านเรกคอร์ดที่ดัชนีนั้นแซชแล้วซึ่งไป และเพราเป็นไฟล์แบบไม่คลัสเตอร์ การอ่านจึงเกิดขึ้นที่ลําเรกคอร์ด ไม่ใช้อ่านทีละเพจ มีจำนวน R เรกคอร์ดต่อเพจ อ่านทั้งหมด B เพจ จึงจะต้องใช้เวลาในการอ่านเรกคอร์ดทั้งหมดเท่ากับ RBD

รวมทั้งการสแกนดัชนีและอ่านเรกคอร์ดทั้งหมด จะใช้เวลาทั้งสิ้น  $RBD + 0.125BD$  หรือ  $BD(R+0.125)$

5(b) การค้นหาข้อมูลแบบเท่ากับจากไฟล์แบบไม่คลัสเตอร์ที่ใช้ดัชนีแบบแซช ขั้นแรกก็ค้นหาดัชนีด้วยฟังก์ชันแซช ก็จะได้ผลลัพธ์ว่าข้อมูลนั้นอยู่ถังใด ก็จะไปอ่านดัชนีเพจนั้นมาเก็บไว้ก่อน ใช้เวลาอ่านดัชนีเท่ากับ D แล้วก็ตามไปอ่านเรกคอร์ดจริง ใช้เวลาอ่านเรกคอร์ดเท่ากับ D ส่งผลให้เวลาทั้งสิ้นเป็น  $2D$

5(c) การค้นหาข้อมูลแบบซึ่งจากไฟล์แบบไม่คลัสเตอร์ที่ใช้ดัชนีแบบแซช กรณีนี้เป็นกรณีที่แปลกออกไป เพราะถ้าคำนวณการทำงานโดยใช้

ฟังก์ชันแซฟแล้วจะส่งผลให้ใช้เวลามาก ดังนั้น ในทางปฏิบัติในกรณีนี้จึงเลือกที่จะไม่ใช้แซฟแล้วเรียกว่าเดียวิกับไฟล์อีป จึงได้เวลาเวลาทั้งสิ้นเท่ากับไฟล์อีป คือ BD

5(d) การแทรกข้อมูลในไฟล์แบบไม่คลัสเตอร์ที่ใช้ดัชนีแบบแซฟ เนื่องจากเรกคอร์ดเป็นไฟล์อีป การแทรกใช้เวลาเท่ากับ 2D เพื่อเขียนข้อมูลเข้าที่หัวไฟล์ใช้อ่านเท่ากับ D และเขียนเท่ากับ D จากนั้น ก็ต้องมาอัปเดตพอยน์เตอร์ ซึ่งทำได้จำกัดเฉพาะสามารถคำนวณจากฟังก์ชันแซฟได้ทันทีไม่ต้องเสียเวลาค้นหาดัชนี จึงทำให้ไปอ่านเพจ ใช้เวลาเท่ากับ D และเขียนกลับเท่ากับ D รวมเวลาทั้งสิ้นเท่ากับ 4D

5(e) การลบข้อมูลในไฟล์แบบไม่คลัสเตอร์ที่ใช้ดัชนีแบบแซฟ ก็เริ่มด้วยการใช้ฟังก์ชันแซฟหาข้อมูลที่จะลบ แล้วอ่านข้อมูลมา ใช้เวลาเท่ากับ D เมื่อลบแล้วเขียนคืน ใช้เวลาเท่ากับ D ค้นหาเสียก่อน ก็จะได้เรกคอร์ดที่จะลบมาอยู่ในหน่วยความจำ เมื่อลบแล้วก็ต้องเขียนกลับ ใช้เวลาเขียนเท่ากับ D จากนั้นก็ต้องไปอัปเดตพอยน์เตอร์ในดัชนี ใช้เวลา D จึงทำให้เวลาทั้งสิ้นมีค่าเท่ากับ Search + 2D

## 9.11 ภาระของฐานข้อมูล

เนื่องจากการทำดัชนีแต่ละแบบจะส่งผลต่อโอเปอเรชันแตกต่างกัน จึงต้องตรวจสอบเวิร์คโหลด (workload) ซึ่งหมายถึง ภาระของฐานข้อมูล ว่าภาระมักเกิดขึ้นที่รีเลชันใด โดยตรวจสอบผ่านการสอบถามที่เกิดขึ้นบ่อย

### 9.11.1 ภาระการสอบถาม

ให้ดูว่าการสอบถามข้อมูลส่วนใหญ่ภาระตกอยู่ที่ใด ประเดิมที่ต้องตรวจสอบเมื่อมีการสอบถามได้แก่

- รีเลชันใดมีการเข้าถึงบ่อยบ้าง
- แอ็ตทริบิวต์ใดที่ถูกค้นคืน
- คำสั่ง SELECT และ JOIN นั้นมักมีแอ็ตทริบิวต์ใดเกี่ยวข้องบ้าง และแอ็ตทริบิวต์ตัวใดส่งผลให้การคัดเลือกข้อมูลต่างกันอย่างไร ถ้าเลือกข้อมูลได้ดี ก็นับว่ามีความซีเลกทีฟ (selective) สูง

### 9.11.2 ภาระการอัปเดต

กรณีถัดไป ให้ดูว่าเมื่อมีการอัปเดตแล้ว ภาระตกอยู่ที่ใด

- เมื่อมีการอัปเดตพร้อมกับใช้คำสั่ง SELECT หรือ JOIN นั้น มักมีแอ็ตทริบิวต์ใดเกี่ยวข้องบ้าง และแอ็ตทริบิวต์ตัวใดมีความซีเลกทีฟ (selective) อย่างไร
- ในการอัปเดตนั้น มักเป็นการแทรก หรือลบ หรือแก้ไขข้อมูล และมีแอ็ตทริบิวต์ใดเกี่ยวข้องบ้าง

เมื่อเรารู้ภาระของรีเลชันและแอ็ตทริบิวต์ที่ถูกเรียกใช้บ่อย ๆ ผ่านการสอบถามหรือการอัปเดต ก็ให้พิจารณาทำดัชนีให้กับรีเลชันเหล่านั้นด้วย แอ็ตทริบิวต์ที่มักใช้ในการ SELECT หรือ JOIN นั้นเอง

การทำดัชนีไม่ได้ส่งผลกระทบบางส่วนไป เพราะการทำดัชนีนั้นก็เพื่อให้การสอบถามได้ผลเร็วขึ้น แต่ก็จะทำให้เกิดภาระทุกครั้งที่อัปเดตฐานข้อมูล จึงต้องพิจารณาและออกแบบอย่างรอบคอบ นอกจากนั้น ผู้ดูแลฐานข้อมูลก็ต้องติดตามประเมินประสิทธิภาพและปรับแก้ดัชนีให้เหมาะสมอยู่เสมอ

## 9.12 ข้อแนะนำในการเลือกดัชนี

ขั้นตอนแรกให้สังเกตที่คำสั่ง WHERE เพื่อรับแอตทริบิวต์ที่อาจนำมากำหนดเป็นดัชนี

- กรณีของการค้นหาแบบเท่ากับ ให้ใช้ดัชนีแบบแยก
- สำหรับการค้นหาแบบช่วง ให้ใช้ดัชนีแบบตันไม้ พร้อมดัชนีแบบคลัสเตอร์เรโน莫
- หากกรณีของการค้นหาแบบเท่ากับ ที่มีข้อมูลซ้ำจำนวนมาก ควรให้ใช้ดัชนีแบบตันไม้ พร้อมดัชนีแบบคลัสเตอร์ร่วมด้วย

เมื่อคำสั่ง WHERE มีเงื่อนไขการคัดเลือกข้อมูลมากกว่าหนึ่งเงื่อนไข ควรพิจารณาใช้คีย์ค้นหาที่มีแอตทริบิวต์มากกว่า 1 ตัวได้

- สำหรับการค้นหาแบบช่วงนั้น ลำดับการจัดเรียงแอตทริบิวต์จะส่งผลต่างกัน
- อาจพิจารณาเลือกใช้ดัชนีแบบอินเด็กซ์โอนลี (index-only) สำหรับการสอบถามที่สำคัญ ๆ ซึ่งหากเลือกใช้อินเด็กซ์โอนลี ก็ไม่ต้องใช้ดัชนีแบบคลัสเตอร์

อย่างไรก็ได้ เรายังเลือกสร้างดัชนีที่ส่งผลดีต่อการสอบถามให้มากที่สุด ไม่ใช่ จำเพาะเฉพาะเจาะจงที่การสอบถามหนึ่ง ๆ และเนื่องจากการทำดัชนีแบบคลัสเตอร์นั้นสามารถทำได้เพียงดัชนีเดียวต่อหนึ่งรีเลชัน เราจึงต้องพิจารณาให้ถี่ถ้วนว่าควรเลือกทำดัชนีกับแอตทริบิวต์ใดที่จะส่งผลดีต่อการสอบถามส่วนใหญ่

ตัวอย่าง

```
SELECT E.dno  
FROM Emp E  
WHERE E.age>40
```

ขั้นแรก ให้ตรวจสอบแอตทริบิวต์ในคำสั่ง WHERE ซึ่งในกรณีนี้พบ 萼ุลจากนั้นให้พิจารณาว่าควรสร้างดัชนีด้วยแอตทริบิวต์ age หรือไม่ ด้วยการดูว่า เงื่อนไขของ WHERE นี้ มีความซีเล็กทีฟมากน้อยเพียงใด เช่น ถ้าพนักงาน ส่วนใหญ่ อายุมากกว่า 40 การทำดัชนีด้วย age ก็จะไม่มีความซีเล็กทีฟ ดังนั้น สมมุติว่าพนักงานส่วนใหญ่ อายุน้อยกว่า 40 ก็ให้ใช้ดัชนีแบบต้นไม้ พร้อมกับดัชนีแบบคลัสเตอร์

ตัวอย่าง

```
SELECT E.dno, COUNT (*)  
FROM Emp E  
WHERE E.age>10  
GROUP BY E.dno
```

กรณีนี้มีคำสั่ง WHERE และ GROUP BY ดังนั้น เมื่อพิจารณา WHERE แต่เพียงอย่างเดียว ก็จะเลือกใช้ `age` ทำดัชนีแบบตันไม้ พร้อมดัชนีแบบคลัสเตอร์ แต่ในกรณีนี้ ด้วยคำสั่ง GROUP จะส่งผลให้ໂປຣເຊັ່ນ COUNT ทำได้ยาก เพราะต้องอ่านข้อมูลที่จัดเรียงตามคำสั่ง GROUP นั่นคือเรียงตาม `dno` หรือหมายເລີດພາຣທມັນຕໍ ดังนั้น จะต้องใช้ `dno` ทำดัชนีแบบตันไม้ พร้อมดัชนีแบบคลัสเตอร์

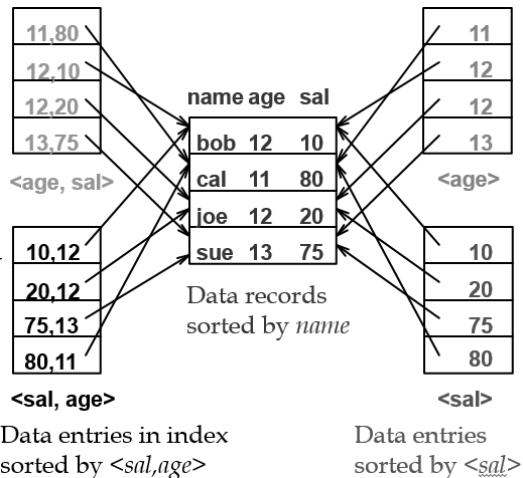
ตัวอย่าง

```
SELECT E.dno  
FROM Emp E  
WHERE E.hobby=Stamps
```

ในกรณีนี้ เนื่องจากเป็นการค้นหาแบบเท่ากับ จึงดูเหมือนว่าควรจะเลือกใช้ดัชนีแบบแยก แต่ถ้าพบว่า `hobby = Stamps` มีช้าจำนวนมาก เพราะไม่ใช่คีย์คุ้มแข็ง ควรใช้ดัชนีแบบตันไม้พร้อมดัชนีแบบคลัสเตอร์แทน

ตัวอย่าง

เมื่อคีย์ค้นหาเป็นแบบคีย์ค้นหาร่วม (composite search key) ดังแสดงในรูปที่ 124 จะเห็นว่า `<age, sal>` และ `<sal, age>` เป็นคีย์ค้นหาร่วม ซึ่งจะส่งผลดีต่อการค้นหาแบบช่วง และลำดับของคีย์ค้นหาอาจเรียงตามตัวอักษร (lexicographic order) หรือตามตำแหน่งของข้อมูล (spatial order) อย่างไรก็ได้ ดัชนีแบบคีย์ค้นหาร่วมจะมีขนาดใหญ่และอับเดตบ่อยกว่า



รูปที่ 124 ดัชนีแบบคีย์ค้นหาร่วม

ตัวอย่าง

จากรูปที่ 124 เมื่อต้องการค้นหาพนักงานที่อายุเท่ากับ 30 และเงินเดือนเท่ากับ 75 ควรใช้ดัชนี  $\langle \text{age}, \text{sal} \rangle$

ถ้าต้องการค้นหา  $20 < \text{age} < 30$  และ  $70 < \text{sal} < 90$  ให้ใช้ดัชนีแบบตั่นไม้ควบคู่กับดัชนีแบบคลัสเตอร์ โดยให้คีย์ค้นหาร่วมเป็น  $\langle \text{age}, \text{sal} \rangle$  หรือ  $\langle \text{sal}, \text{age} \rangle$  ก็ได้

หากต้องการค้นหา  $\text{age} = 30$  และ  $70 < \text{sal} < 90$  ให้ใช้ดัชนีแบบคลัสเตอร์ โดยให้คีย์ค้นหาร่วมเป็น  $\langle \text{age}, \text{sal} \rangle$  ซึ่งจะให้ผลลัพธ์เร็วกว่า  $\langle \text{sal}, \text{age} \rangle$ มาก

### 9.13 อินเด็กซ์ออนไลน์

อินเด็กซ์ออนไลน์ (Index only) หมายถึงสถานการณ์ที่สามารถค้นหาข้อมูลในฐานข้อมูลจากดัชนี โดยไม่ต้องเข้าถึงข้อมูลจริง ซึ่งจะทำให้เข้าถึงข้อมูลที่ต้องการได้เร็วมาก

ตัวอย่าง

```
SELECT D.mgr  
FROM Dept D, Emp E  
WHERE D.dno=E.dno
```

ให้ใช้ <E.dno> สร้างดัชนีแบบตันไม้

```
SELECT D.mgr, E.eid  
FROM Dept D, Emp E  
WHERE D.dno=E.dno
```

ให้ใช้ <E.dno,E.eid> สร้างดัชนีแบบตันไม้

```
SELECT E.dno, COUNT(*)  
FROM Emp E  
GROUP BY E.dno
```

ให้ใช้ <E.dno> สร้างดัชนีแบบตันไม้

```
SELECT      E.dno, MIN(E.sal)
FROM        Emp E
GROUP BY    E.dno
```

ให้ใช้ <E.dno,E.sal> สร้างดัชนีแบบตันไม้

```
SELECT      AVG(E.sal)
FROM        Emp E
WHERE       E.age=25 AND
           E.sal BETWEEN 3000 AND 5000
```

ให้ใช้ <E.age,E.sal> หรือ <E.sal, E.age> สร้างดัชนีแบบตันไม้

## 9.14 บทสรุปและเรื่องชวนคิด

การเรียนรู้การจัดเก็บข้อมูลจะทำให้ผู้ดูแลฐานข้อมูลสามารถออกแบบหรือปรับปรุงฐานข้อมูลให้ทำงานได้อย่างมีประสิทธิภาพ ซึ่งเป็นสิ่งที่จำเป็นอย่างยิ่งสำหรับวิศวกรคอมพิวเตอร์ที่จะต้องเข้าใจการทำงานของระบบโดยรวม สำหรับบทนี้ อยากรู้ว่าเรียนให้คิดต่อว่าวิธีการจัดเก็บข้อมูลและการทำดัชนีที่ได้เรียนในบทนี้สามารถนำไปใช้กับข้อมูลยุคบีก้าต้าได้หรือไม่ ถ้าได้ เพราะอะไร หรือถ้าคิดว่าไม่ได้ก็ขอให้หาความรู้เพิ่มพูนว่ามีวิธีการอะไรอีกบ้างในการจัดเก็บข้อมูลและการทำดัชนี

## แบบฝึกหัดท้ายบท

1. เหตุใดที่เราต้องคำนึงถึงวิธีการจัดเก็บฐานข้อมูลที่เหมาะสม และเราว่า  
คำนึงถึงประเด็นใดบ้าง
2. การวัดความเร็วในการอ่านเขียนฐานข้อมูลวัดเป็นอะไร และการที่ระบบ  
ฐานข้อมูลตอบสนองได้ไว จะมีค่าดังกล่าวเป็นอย่างไร
3. จงเปรียบเทียบข้อดีข้อเสีย ของโครงสร้างไฟล์แบบไม่เรียงลำดับ  
ไฟล์แบบเรียงลำดับ และดัชนี
4. หากข้อมูลที่ต้องการสร้างดัชนีมีลักษณะคือ เมื่อค้นหาตามดัชนีแล้ว  
อาจพบ rekord มากกว่า 1 rekord ได้ เรายังจะใช้วิธีการสร้าง  
รายการข้อมูลแบบใด เพราะอะไร
5. จงเปรียบเทียบระหว่างดัชนีเบาบาง และดัชนีหนาแน่น ในประเด็นของ  
ความเร็วในการค้นหา และขนาดพื้นที่จัดเก็บ
6. จงเปรียบเทียบระหว่างโครงสร้างดัชนีแบบแยก และโครงสร้างดัชนีแบบ  
ตันไม้ ในประเด็นของการค้นหาข้อมูล



## บทส่งท้าย

เราเดินทางมาจนถึงบทสุดท้ายของตำนานนี้ เพื่อเรียนรู้เรื่องฐานข้อมูลโดยเฉพาะฐานข้อมูลเชิงสัมพันธ์ ความรู้ในระดับนี้นับว่าเป็นความรู้พื้นฐานที่ผู้เรียนวิชาฯว่าด้วยฐานข้อมูลในระดับปริญญาตรีต้องเข้าใจได้ อย่างไรก็ต้องความรู้ด้านฐานข้อมูลยังมีอีกมาก เช่น การจัดการความปลอดภัยของข้อมูล การปรับสมรรถภาพฐานข้อมูล การประเมินขนาดฐานข้อมูล และอื่นๆ อีกมากมาย อีกทั้งยังมีรูปแบบฐานข้อมูลที่ต่างไปจากเดิม สามารถเรียนได้จนถึงปริญญาเอก เพื่อเสนอวิธีการที่จะพัฒนาให้ฐานข้อมูลก้าวหน้าไปอีกผู้เรียนเชื่อว่าพื้นฐานเหล่านี้เพียงพอที่จะให้ผู้เรียนสามารถทำงานด้านฐานข้อมูลในองค์กรหรือเรียนต่อได้



## បរវត្ថុក្រណី

- [1] R. Elmasri and S. Navathe, *Fundamentals of Database Systems - 7th edition*. Pearson London, 2016.
- [2] R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems - 5th edition* (Database). 2003.
- [3] R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems - 6th Edition*. 2011.
- [4] J. Martin and James, *Computer data-base organization*. Prentice-Hall, 1977, pp. 713-713.
- [5] J. Martin and J. Leben, *Client/server databases : enterprise computing*. Prentice Hall PTR, 1995, pp. 352-352.
- [6] B. Boehm, "A view of 20th and 21st century software engineering," in *Proceedings of the 28th international conference on Software engineering*, 2006, pp. 12-29: ACM.
- [7] C. Ghezzi, M. Jazayeri, and D. Mandrioli, *Fundamentals of software engineering*. Prentice Hall PTR, 2002.
- [8] W. S. Humphrey, *A discipline for software engineering*. Addison-Wesley Longman Publishing Co., Inc., 1995.
- [9] *The database development life cycle: 1.1 Introduction - OpenLearn - Open University - M359\_1*. Available:

- <http://www.open.edu/openlearn/science-maths-technology/computing-and-ict/information-and-communication-technologies/the-database-development-life-cycle/content-section-1.1>
- [10] W. Adrienne and A. Watt, "Database Design - 2nd Edition," ed.
  - [11] H. Garcia-Molina *et al.*, *Database Systems: A Practical Approach to Design, Implementation, and Management*. 2010.
  - [12] M. L. Gillenson, *Fundamentals of database management systems*. Wiley, 2005, pp. 366-366.
  - [13] R. Ramakrishnan and J. Gehrke, *Database Management Systems*. 2003.
  - [14] T. Risch *et al.*, "Database Management System," 2009.
  - [15] C. Coronel and S. Morris, *Database systems: design, implementation, & management*. Cengage Learning, 2016.
  - [16] Y. E. Gelogo and S. Lee, "Database management system as a cloud service," *International Journal of Future Generation Communication and Networking*, vol. 5, no. 2, pp. 71-76, 2012.
  - [17] S. Ramanathan, S. Goel, and S. Alagumalai, "Comparison of cloud database: Amazon's SimpleDB and Google's Bigtable," in *Recent Trends in Information Systems*

- (ReTIS), 2011 International Conference on, 2011, pp. 165-168: IEEE.
- [18] B. Alam, M. Doja, M. Alam, and S. Mongia, "5-Layered Architecture of Cloud Database Management System," *AASRI Procedia*, vol. 5, pp. 194-199, 2013.
- [19] D. M. DeKimpe, W. E. Malloy, and C. R. Tomlyn, "Extension of data definition language (DDL) capabilities for relational databases for applications issuing DML and DDL statements," ed: Google Patents, 2002.
- [20] R. Elmasri and S. B. Navathe, "Fundamentals of Database Systems 5th edition," *Database*, 2003.
- [21] *Types of Database Management Systems*. Available: <https://www.sharpcorner.com/UploadFile/65fc13/types-of-database-management-systems/>
- [22] *Types of Databases and Dbms (With Examples)*. Available: <https://codebots.com/data-management/types-of-databases-and-dbms-with-examples>
- [23] R. Elmasri, J. Weeldreyer, and A. Hevner, "The category concept: An extension to the entity-relationship model," *Data and Knowledge Engineering*, 1985.
- [24] H. Garcia-Molina, J. D. Ullman, and J. Widom, "Database Systems: The Complete Book," *Education*, 2008.

- [25] J. A. Hoffer, *Modern Database Management*, 10/e. Pearson Education India, 2011.
- [26] I.-Y. Song, M. Evans, and E. K. Park, "A comparative analysis of entity-relationship diagrams," *Journal of Computer and Software Engineering*, vol. 3, no. 4, pp. 427-459, 1995.
- [27] C. Liang *et al.*, "MAGIC-SPP: a database-driven DNA sequence processing package with associated management tools," *BMC bioinformatics*, vol. 7, no. 1, p. 115, 2006.
- [28] G. Chen and E. Kerre, "Extending ER/EER concepts towards fuzzy conceptual data modeling," in *Fuzzy Systems Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, 1998, vol. 2, pp. 1320-1325: IEEE.
- [29] R. Elmasri and S. Navathe, *Fundamentals of database systems*. Addison-Wesley Publishing Company, 2010.
- [30] *Top 6 Data Modeling Tools - Data Science Central*. Available:  
<https://www.datasciencecentral.com/profiles/blogs/top-6-data-modeling-tools>
- [31] *Comparison of data modeling tools - Wikipedia*. Available:

[https://en.wikipedia.org/wiki/Comparison\\_of\\_data\\_modeling\\_tools](https://en.wikipedia.org/wiki/Comparison_of_data_modeling_tools)

- [32] E. F. Codd, "A relational model of data for large shared data banks," *Communications of the ACM*, vol. 13, no. 6, pp. 377-387, 1970.
- [33] E. F. Codd, "A data base sublanguage founded on the relational calculus," in *Proceedings of the 1971 ACM SIGFIDET (now SIGMOD) Workshop on Data Description, Access and Control*, 1971, pp. 35-68: ACM.
- [34] E. F. Codd, "Normalized data base structure: A brief tutorial," in *Proceedings of the 1971 ACM SIGFIDET (now SIGMOD) Workshop on Data Description, Access and Control*, 1971, pp. 1-17: ACM.
- [35] E. F. Codd, *Relational completeness of data base sublanguages*. Citeseer, 1972.
- [36] E. F. Codd, "Extending the database relational model to capture more meaning," *ACM Transactions on Database Systems (TODS)*, vol. 4, no. 4, pp. 397-434, 1979.
- [37] E. F. Codd, "Data models in database management," *ACM Sigmod Record*, vol. 11, no. 2, pp. 112-114, 1981.
- [38] E. F. Codd, "Missing information (applicable and inapplicable) in relational databases," *ACM Sigmod Record*, vol. 15, no. 4, pp. 53-53, 1986.

- [39] E. F. Codd, "Relational database: a practical foundation for productivity," in *Readings in Artificial Intelligence and Databases*: Elsevier, 1988, pp. 60-68.
- [40] E. F. Codd, *The relational model for database management: version 2*. Addison-Wesley Longman Publishing Co., Inc., 1990.
- [41] M. Campbell-Kelly, "Edgar Codd," *The Independent*, 2003.
- [42] E. F. Codd, 1981 *Turing Award Lecture – Relational Database: A Practical Foundation for Productivity*. 1981.
- [43] C. J. Date, *The database relational model : a retrospective review and analysis : a historical account and assessment of E.F. Codd's contribution to the field of database technology*. Addison-Wesley, 2001, pp. 152-152.
- [44] *Database Course, graduate database course, databases, DBMS, Graduate Database, Oracle*. Available: <http://sce.uhcl.edu/boetticher/courses.html>
- [45] M. Lacroix and A. Pirotte, "Domain-oriented relational languages," in *Proceedings of the third international conference on Very large data bases-Volume 3*, 1977, pp. 370-378: VLDB Endowment.
- [46] A. Beaulieu, *Learning SQL*, 2nd ed. Sebastopol, CA, USA: O'Reilly, 2009, pp. 320-320.

- [47] M. Chapple, "SQL Fundamentals," *Databases*.
- [48] M. Chatham, *Structured query language by example - volume i : data query language*. Lulu Com, 2012, pp. 8-8.
- [49] C. J. Date and H. Darwen, *A guide to the SQL standard : a user's guide to the standard database language SQL*. Addison-Wesley, 1997, pp. 522-522.
- [50] J. Melton and A. R. Simon, *Understanding the new SQL : a complete guide*. Morgan Kaufmann Publishers, 1993, pp. 536-536.

# ଡାଟାବେସ୍

## 1

1:I, 61  
1:N, 61  
1NF, 143

BCNF, 154  
binary, 59  
binary relational operations, 161  
Boyce-Codd Normal Form, 154

## 2

2NF, 147

candidate key, 99  
cardinality, 61, 95  
CARTESIAN, 173  
cascade, 103  
catalog, 11  
Category, 81  
client/server, 26

## 3

3NF, 151

clustered index, 225  
column, 89  
commutative, 164, 172  
completeness constraint, 77  
composite attribute, 54  
conceptual data model, 45  
Conceptual data model, 28  
conceptual level, 108  
conceptual model, 12  
conceptual modeling, 83  
constraint, 9, 96  
COUNT, 187  
CREATE, 205  
cross reference, 114  
cross-product, 173

## A

aggregate function, 186  
ALTER, 206  
Amazon Web Service : AWS, 32  
anomaly, 130  
Armstrong's axioms, 136  
associative, 172  
atomic, 97  
attribute, 46, 89, 92, 95  
attribute defined-specialization, 75, 121  
augmentation, 137

## B

B+ tree, 233  
backup, 219  
Batch, 15

**D**

data, 3  
data conversion tool, 39  
data definition language, 33  
data dictionary, 13, 30  
data entries, 222, 223  
data manipulation language, 33  
data model, 12  
data type, 9, 92, 95  
database, 3  
Database Administrator, 38  
database management system, 4  
database system, 5  
DBA, 38  
DBMS, 4  
DDL, 33  
decrypt, 31  
degree, 95  
degree of relationship type, 59  
delete, 102, 103  
DELETE, 206  
deletion, 9  
dense index, 227  
dependency-preserving, 143  
derived attribute, 56  
disaster, 7  
disjoint, 76  
disjointness constraint, 76  
disk, 219  
DIVISION, 177  
DML, 33  
Document stores, 37  
domain, 90, 92, 95  
domain constraint, 96  
domain relational calculus, 189  
Dr. Edgar F. Codd, 88

## DROP, 205

**E**

EER model, 69  
encrypt, 31  
Enhanced ER model, 69  
entity, 46, 49  
entity integrity constraint, 96, 99  
entity type, 50  
Entity-Relationship diagram, 46  
Entity-Relationship model, 46  
equality search, 242  
EQUI JOIN, 180  
ER diagram, 46  
ER model, 46  
Existential quantifier, 190  
Extended ER model, 69  
extension, 94

**F**

$F^+$ , 136, 137  
F-closure, 136, 137  
field, 89, 95  
File Access Method, 15  
file organization, 220  
fill factor, 235  
First Normal Form, 143  
foreign key, 100  
FOREIGN KEY, 208  
free tuple variable, 193  
full functional dependency, 148  
Full Outer JOIN, 185  
functional dependency, 132  
Functional dependency, 101

## G

generalization, 69, 73  
Graph database, 37  
group, 187

## H

hash-based index, 229  
heap file, 220  
hierarchical database, 35

## I

implementation data model, 45  
Implementation data model, 28  
index, 221  
index entries, 234  
Index sequential, 15  
indexed file organization, 223  
inheritance, 69, 72  
insert, 102  
INSERT, 206  
insertion, 9  
integrity constraint, 13  
intension, 94, 95  
intersect, 170

## J

join, 179  
JOIN, 175, 179

## K

$k^*$ , 222  
key, 97  
Key, 98

key constraint, 13, 96  
Key-value stores, 37

## L

leaf, 233  
leaf page, 233  
LEFT-OUTER-JOIN, 184  
Logical file, 15  
logical level, 108  
lossless-join, 143

## M

M:N, 61  
maintenance, 9  
mapping, 28  
mathematical concept of set, 88  
MAX, 187  
meta-data, 8  
MIN, 187  
MINUS, 170  
modeling tool, 83  
modification, 9  
modify, 102  
monitor, 17  
multimedia, 4  
multiple inheritance, 80  
Multiple-key, 15  
multi-valued attribute, 55

## N

N:1, 61  
n-ary, 60  
n-ary relation operations, 161  
NATURAL JOIN, 181  
network database, 35

non-leaf page, 234

normalization, 142

NoSQL, 19, 36

notation, 66

Not-only-SQL, 19

Not-Only-SQL, 36

null, 93

## R

range search, 221, 237

range selection, 242

record, 89, 95, 219

Record ID, 219

recovery, 7

recursive relationship, 59

redundancy, 10, 130

referenced relation, 100

REFERENCES, 208

referencing relation, 100

referential integrity, 100

referential integrity constraint, 96

reflexivity, 137

regular entity type, 50

relation, 89, 95

relational algebra expression, 160

relational calculus, 189

relational database, 35, 88

relational database system, 17

relational DBMS, 18, 36

relational operation, 161

relationship, 46

relationship type, 58

RENAME, 167

report, 9

Representation data model, 28

representational data model, 45

rho, 167

RID, 219

Right Outer JOIN, 184

root, 79, 233

row, 89

## S

scan, 242

schema, 93, 95  
search key, 222  
Second Normal Form, 148  
secondary index, 225  
secondary storage, 9  
SELECT, 162  
selective inheritance, 81  
semantic, 45  
sequential access, 219  
Serial access, 15  
server, 26  
set difference, 170  
set theory, 160  
Shared subclass, 80  
simple attribute, 54  
single inheritance, 79  
single-valued attribute, 55  
sparse index, 226  
specialization, 69, 73  
specialization hierarchies, 79  
Specialization Lattice, 80  
SQL, 200  
stored attribute, 56  
stored database, 8  
structured query language, 200  
subclass/superclass, 69, 70  
SUM, 187  
superkey, 97  
surrogate key, 123

## T

table, 89, 95  
tape, 219  
ternary, 60  
theory of relation, 88  
Third Normal Form, 151

three-schema architecture, 26  
total participation, 63, 112  
transaction, 12  
Transitive dependency, 151  
transitivity, 137  
tree-based index, 233  
tuple, 91, 95  
tuple relational calculus, 189  
type compatibility, 169

## U

UML, 66  
unary relational operations, 161  
unclustered index, 225  
Unified Modeling Language, 66  
union, 81  
UNION, 168  
UNIQUE, 207  
Universal quantifier, 190  
unsorted file, 220  
update, 102, 103  
UPDATE, 207

## V

view, 12  
VIEW, 209

## W

weak entity type, 50, 111  
weak relationship, 58  
Wide-column stores, 37

## X

$X^+$ , 136, 140

- X-closure, 136, 140
- ก**
- กฎการแต่งเติม, 137
  - กฎการถ่ายทอด, 137
  - กฎการย่อยี, 137
  - กฎการซูเนียน, 137
  - กฎการลงทะเบียน, 137
  - กฎของคอดต์, 160
  - กระบวนการประมวลผลแบบบกุม, 14
  - กระบวนการย่อ, 130
  - กระบวนการสเปเชียลไดเซ็นแบบแอตทิว  
บิวต์กำหนด, 75
  - กลุ่มแอนทิตี, 50
  - การแก้ไข, 9, 102
  - การเข้ากันแบบทั้งหมด, 148
  - การเข้ากึ่งข้อมูลแบบตามลำดับ, 15
  - การเข้ากึ่งข้อมูลแบบสุ่ม, 15
  - การเข้ากึ่งข้อมูลเป็นแบบลำดับ, 219
  - การเข้ากึ่งไฟล์, 15
  - การเข้าร่วมแบบบังคับ, 63
  - การเข้าร่วมแบบไม่บังคับ, 63
  - การคืนสภาพ, 30
  - การจัดการความปลอดภัย, 17
  - การจัดโครงสร้างไฟล์, 220
  - การจัดทำรายงาน, 9
  - การทำเหมินการตามที่ผู้ให้ระบุ, 44
  - การทำเหมินการบนแบบจำลองฐานข้อมูล, 44
  - การทำเหมินการระดับพื้นฐาน, 44
  - การทำรายการ, 12
  - การแทรก, 102
- การแทรกข้อมูล, 9
  - การนับ, 186
  - การบำรุงรักษา, 17
  - การปรับปรุงและบำรุงรักษาฐานข้อมูล, 39
  - การแปลงข้อมูล, 39
  - การเมี้ยป, 28
  - การรับทอด, 69, 72
  - การรับทอดแบบเดี่ยว, 79
  - การรับทอดแบบเลือก, 81
  - การรับทอดแบบหลาย, 80
  - การลบ, 9, 102, 103
  - การสร้างรายงาน, 39
  - การสอบถาม, 9, 160, 161
  - การสำรองข้อมูล, 39
  - การหาค่าเฉลี่ย, 186
  - การหาค่าต่ำสุด, 186
  - การหาค่าสูงสุด, 186
  - การหาผลรวม, 186
  - กำหนด, 132
- บ**
- ข้อความ, 52
  - ข้อคำถาม, 8
  - ข้อบังคับของคีย์, 13
  - ข้อบังคับความเป็นบูรณาภิพ, 13
  - ข้อมูล, 3
  - ข้อมูลเชิงตรรกะ, 14
  - ข้อมูลแปลงกลอม, 143, 174
  - ข้อมูลผิดพลาด, 130
  - ข้อมูลเพียง, 130

- ข้อมูลมัลติมีเดีย, 4  
 ข้อมูลไม่เกิน, 143  
 ขั้นกับ, 132  
 เข้าถึงข้อมูลตรง ๆ, 15  
 เข้าร่วมแบบบังคับ, 112  
 เข้าร่วมแบบไม่บังคับ, 112  
 เข้ารหัส, 31
- ค**
- ค้นหาช่วงของข้อมูล, 237  
 ค้นหาแบบท่ากับ, 242  
 ครอสโปรดักต์, 173  
 คลาสออย, 70  
 คลาสหลัก, 70  
 ความขึ้นต่อ กันแบบส่งผ่าน, 151  
 ความชี้ช้อน, 10, 130  
 ความมีส่วนร่วมของขั้นคลาส, 76  
 ความสมบูรณ์ของความสัมพันธ์, 77  
 ความสัมพันธ์ข้อนใน, 143  
 ความสัมพันธ์แบบเรียนเกิด, 59  
 ความสัมพันธ์อีสเอ, 71  
 คงลัมป์, 89, 95  
 คัลซานตรา, 19  
 ค่า, 52  
 ค่าต่างของเซต, 170  
 คาร์ดินาลิตี้, 61  
 คาร์ดินาลิตี้, 61, 95  
 คาร์ทีเชียน, 173  
 ค่าว่าง, 56, 93  
 ค่าอะตอมมิก, 97  
 คำนวนเวลา, 241
- คีย์, 53, 97, 98  
 คีย์ค้นหา, 222  
 คีย์คุณ, 99, 133  
 คีย์เชอร์เกต, 123  
 คีย์บางส่วน, 51  
 คีย์หลัก, 99, 110  
 คีย์แอ็ตทริบิวต์, 53  
 คุณสมบัติการเรียงลำเบ列表, 164  
 คู่ <k, id>, 224  
 เครื่องมือสร้างแบบจำลองข้อมูล, 83  
 แคดดาลีก, 11  
 แคทากอรี, 81, 123  
 แคนติเดตคีย์, 99  
 แคลคูลัสเชิงสัมพันธ์, 160, 189  
 แคลคูลัสเชิงสัมพันธ์อิงโดเมน, 189, 191  
 แคลคูลัสเชิงสัมพันธ์อิงเพล, 189, 190  
 แคลสิก, 103  
 โครงสร้างของข้อมูล, 9  
 โครงสร้างของฐานข้อมูล, 34  
 โครงสร้างข้อมูลทางกายภาพ, 14  
 โครงสร้างด้วยแบบตัวนี้ไม้, 233  
 โครงสร้างด้วยนี้แบบแซช, 229  
 โครงสร้างตาราง, 95  
 โครงสร้างไฟล์แบบด้วยนี้, 223
- ง**
- เงื่อนไขคีย์, 96, 97, 103  
 เงื่อนไขโดเมน, 96, 97, 103  
 เงื่อนไขบังคับ, 9, 96  
 เงื่อนไขบังคับเข้าร่วม, 61, 63  
 เงื่อนไขบูรณาการอ้างอิง, 96, 100

เงื่อนไขบูรณาภาพอาณทิติ, 96, 99, 103

ฐานข้อมูลแบบกราฟ, 37

ฐานข้อมูลแบบเอกสาร, 37

## จ

จอยน์, 179

จำนวนเต็ม, 52

เจเนอเรล์ໄลเซชัน, 69, 73

## ก

ชนิดของข้อมูล, 9, 52

## ก

ขับคลาส, 70

ขับคลาส/ชูเปอร์คลาส, 69, 70

ขับคลาสที่ผู้ใช้กำหนด, 76

ขับคลาสร่วม, 80

ขับเขต, 97

ชูเปอร์คลาส, 70

ชูเปอร์คลีฟ, 97

เขตสมบูรณ์, 188

## ก

ฐานข้อมูล, 3

ฐานข้อมูลคลอสัมบัน្តกว้าง, 37

ฐานข้อมูลค่าคีย์, 37

ฐานข้อมูลเชิงเครือข่าย, 35

ฐานข้อมูลเชิงคำดับขั้น, 35

ฐานข้อมูลเชิงวัดๆ, 11, 35

ฐานข้อมูลเชิงสัมพันธ์, 35, 36

ฐานข้อมูลในเอกสารค่าวาอล, 36

## ด

ดร. เอด加ร์ เอฟ คอดต์, 88

ตั้งนี, 13, 221

ตั้งนีคลัสเตอร์, 225

ตั้งนีเบาบาง, 226

ตั้งนีไม่คลัสเตอร์, 225

ตั้งนีร่อง, 225

ตั้งนีหนาแน่น, 227

ตั้งนีหลัก, 224

ดำเนินการในโอลิเมอร์ชั้นเชิงสัมพันธ์จาก 2

รีเลชัน, 168

ติสก์, 219

ติสกอยท์, 76

ติกวี, 95

ติกวีความสัมพันธ์, 59

โดเมน, 90, 91, 92, 95

## ต

ต้นไม้บีเพลส, 233

ตัวชี้, 219

ตัวบ่งปริมาณ, 190

ตัวแปรทุเพลิอิสระ, 193

ตัวเลข, 52

ตาราง, 89, 95

**ก**

ဂიტრაჟ, 31

ແຄາ, 89, 95

**ທ**

ທດສອບແລະປະເມີນຮຽນຂໍ້ອມຸລ, 24

ທຮງ, 133

ທຖານີ່ເຊືດ, 160

ທຖານີ່ເລີ້ນ, 88

ທູພຶລ, 91, 95

ເຫັປ, 219

ເຫອນໄວ້, 60, 117

ແທຣກ, 102

ແທຣກເຮັດວຽກ, 242

ໄທປີເຕີຍວັກນ, 169, 171, 173

**ນ**

ນອນລື່ີເພິຈ, 234

ນອວັມລົ່ວ່າງທີ 1, 143

ນອວັມລົ່ວ່າງທີ 2, 147

ນອວັມລົ່ວ່າງທີ 3, 151

ນອວັມລົ່ວ່າງທີ 4, 154

ນອວັມລົ່ວ່າງທີ 5, 160

ນັດ, 56

ນຳກັບມາໃໝ່, 6

ນິພຈນີ່ພື້ນຍົດເຊີງສັນພັນນີ້, 160

ນິຍາມຂໍ້ອມຸລ, 8

ແນວົດເຊີງວັດຖຸ, 72

ແນວົດຮຽນຂໍ້ອມຸລເຊີງສັນພັນນີ້, 88

ໂນເສດວິແລລ, 19

**ປ**

ບຣິກາຣຄລາວດີ, 26

ບຣິກາຣເວີບຂອງອມື່ອນ, 32

ບໍາຮຸຮັກຂໍ້ອມຸລ, 24

ບຶກດາດ້າ, 19

ບຶຈີເອັນເອົ່າ, 154

ບຸຮັນກາພຂອງຂໍ້ອມຸລ, 16

ບຸຮັນກາພອ້າງອີງ, 103

ແບບກະຈາຍ, 34

ແບບຄວາມສັນພັນນີ້, 58

ແບບຈຳລອງຂໍ້ອມຸລ, 12, 44, 45

ແບບຈຳລອງຂໍ້ອມຸລເຊີງກາຍກາພ, 45

ແບບຈຳລອງຂໍ້ອມຸລເຊີງແນວົດ, 28, 45

ແບບຈຳລອງຂໍ້ອມຸລເຊີງປົງປັບຕິກາຣ, 28, 45

ແບບຈຳລອງເຊີງສັນພັນນີ້, 189

ແບບຈຳລອງຮະຕັບຮຽກ, 108

ແບບຈຳລອງຮະຕັບແນວົດ, 83, 108

ແບບຈຳລອງອີ້ວັກ, 46

ແບບຈຳລອງອີ້ວັກແບບຂາຍ, 69

ແບບຈຳລອງເອັນທີ່ເລີ້ນເຂົ້າປິປີ, 46

ແບບວິວິຫພັນນີ້, 34

ແບບສທັນນີ້, 34

ແບບເອກພັນນີ້, 34

ໄບນາໄວ້, 59

**ຈ**

ປະມະລຸຜລພວ່ນກັນແລະໃຫ້ຂໍ້ອມຸລຮ່ວມກັນ, 9

- ปรับความสมดุลของต้นไม้บีเพลส, 237
- โปรแกรมลูกข่าย, 30
- ก**
- ผู้ใช้เดียว, 34
- ผู้ดูแลฐานข้อมูล, 38
- แผนภาพความสัมพันธ์ระหว่างเออนทีกับบริเลชั่นซิป, 46
- แผนภาพอีอาร์, 47
- แผนภาพอีอาร์, 69
- ก**
- ภาษาเชิงวัฒนธรรม, 18, 69
- ภาษาติดีแออล, 33
- ภาษาตีเอ้มแออล, 33
- ภาษาตีแออล, 33
- ภาษาสอนภาษา, 8, 33
- ภาษาอีสติกวัฒนธรรม, 200
- ภาษาอีสตีแออล, 33
- ก**
- พจนา奴กรณ์ข้อมูล, 13, 30
- พอยน์เตอร์, 219
- พัฒนาฐานข้อมูล, 24
- พาย, 165
- พิชคณิตเชิงล้มเหลว, 160
- เพจipe, 233
- เพจไอโอ, 220
- เพรดิคเตอร์หนา, 121
- ไฟร์แน็ตทริบิวต์, 148
- ไฟร์มาร์คิย์, 99
- ก**
- ฟอร์เจนคิร์, 100
- ฟังก์ชันนักดีเพนเดนซี, 101, 130, 132
- ฟังก์ชันนักดีเพนเดนซียังคงสภาพ, 143
- ฟิลแฟคเตอร์, 235
- ฟิล์ต, 16, 89, 95
- ไฟล์ข้อมูลจริง, 15
- ไฟล์เชิงตรรกะ, 15
- ไฟล์แบบไม่เรียงลำดับ, 220
- ไฟล์แบบเรียงลำดับ, 221
- ไฟล์อีปี, 220
- ก**

**ย**

ყูเนียน, 81, 168

ลำดับเชิงตัวนี้, 15

ลำดับอนุกรม, 14

ลีฟ, 233

ลีฟเพจ, 233

ลูกข่ายแม่ข่าย, 26

เล็กที่สุด, 133

**ร**

รหัสเรกคอร์ด, 219

ระบบการจัดการฐานข้อมูล, 4

ระบบการจัดการฐานข้อมูลเชิงสัมพันธ์, 18

ระบบการจัดการฐานข้อมูลแบบกระจาย, 34

ระบบเกิดความคาดเดาของ, 7

ระบบฐานข้อมูล, 5

ระบบฐานข้อมูลเชิงสัมพันธ์, 17

ராக, 79, 233

ரங்வீத் வரி, 88

ரயகரி, 222, 223

ரயகரிதழ்நி, 234

ரீலெக்ஷன், 88, 89, 95

ரீலெக்ஷன்ชิป, 46, 57

ரீலெக்ஷันชิปไทยบุ, 51

ரீலெக்ஷันผู้อ้าง, 100

ரீலெக்ஷันรับอ้าง, 100

ரேகคอร์ด, 219

ரேகคอร์ด, 89, 95

ரெடிஸ, 19

**ต**

วันทูเมนี, 61

วันทูวน, 61

วิเคราะห์ฐานข้อมูล, 24

วิธีกรอเสริฟ์เรนซ์, 114

วิว, 28

วิศวกรรมซอฟต์แวร์, 22

วิเครียลีเซ็นชิป, 58

วีคเอนทิตี้ไทป์, 50, 111

**ถ**

สแกน, 242

สคีมา, 93, 95

สคีมาแนววิดิค, 28

สคีมาภายนอก, 28

สคีมาภายใน, 28

สถาปัตยกรรมแบบคลาวด์, 26, 32

สถาปัตยกรรมแบบทรีสคีมา, 26

สถาปัตยกรรมแบบรวมมุนญ์, 26

สถาปัตยกรรมแบบลูกข่ายแม่ข่าย, 26, 29

สเปเชียลไลเซชัน, 69, 73

สเปเชียลไลเซชันแบบลำดับขั้น, 79

สเปเชียลไลเซชันแบบแลตทิช, 80

**ก**

ຄப, 103, 171

ຄບເຮກໂຄຣດ, 242

ຄືອກ, 2

ຄົກຍະນະຂໍ້ມູນ, 92, 95

- สเปเชียลไลเซชันแบบแอ็ตทริบิวต์กำหนด,  
121
- สมบัติการเปลี่ยนกลุ่ม, 172
- สมบัติการสลับที่, 172
- สังจันของอาرمสตรอง, 136, 137
- สัญกรณ์, 66
- สัญกรณ์คณิตศาสตร์, 93
- สัญกรณ์เพิ่มเติม, 66
- สัญกรณ์ยุเมะอล, 66
- สัญลักษณ์-, 171
- สัญลักษณ์π, 165, 168
- สารบัญ, 11
- สำรองข้อมูล, 219
- ॥**
- หน่วยเก็บข้อมูล, 8
- หน่วยเก็บข้อมูลภายนอก, 218
- หน่วยเก็บร่อง, 9
- หนึ่งต่อหนึ่ง, 61
- หนึ่งต่อหลาย, 61
- หลักการจากคณิตศาสตร์เรื่องเซต, 88
- หลายต่อหนึ่ง, 61
- หลายต่อหลาย, 61
- ⌚**
- ออกแบบฐานข้อมูล, 24
- อ oranacikl, 4, 18
- อะดอมมิก, 103
- อินเตอร์เชก, 170
- อินเตอร์เชกชัน, 170
- อินเทนชัน, 94, 95
- อินสแตนซ์, 133
- เอกสารเทนชัน, 50, 94, 95
- เอนกิตี, 46, 49
- เอนกิตีไทยปี, 50, 110
- เอนกิตีเจ้าของ, 51
- เอนกิตีไทยปักติ, 50
- เอนกิตีไทยบุ, 51
- เอนนารี, 60, 117
- เอฟไอ, 30
- ເອໂຄລເຊ່ວງ, 137
- ເສຄົວແອລເຊີ່ວໄວ່ວັນ, 200
- ເສຄົວແອລເຊີ່ວໄວ່ວັນເວົ້ວ, 18
- ແອກກວິເກຕິພັງກົ່ນ, 186
- ແອດທຣິບົວຕົ່ວມາວຸນ, 56
- ແອດທຣິບົວຕຳເຫິວ, 55
- ແອດທຣິບົວຕຳບັນທຶກ, 56
- ແອດທຣິບົວຕົ່ວພະຍະ, 71
- ແອດທຣິບົວຕົ່ວວົມ, 54
- ແອດທຣິບົວຕໍ່ຫລາຍຄ່າ, 54, 55
- ແອດທຣິບົວຕໍ່ອ່າຍຈ່າຍ, 54
- ໂອສີເປີເປີ, 30
- ໂອເປົວເຮັນເຈິງສັນພັນຮັກ, 161
- ໂອເປົວເຮັນເຈິງສັນພັນຮັກ 2 ຮື່ເລັ້ນ, 161
- ໂອເປົວເຮັນເຈິງສັນພັນຮັກຈາກກວິເເລັ້ນເດີຍວັນ, 161
- ໂອເປົວເຮັນເຈິງສັນພັນຮັກຫລາຍຮື່ເລັ້ນ, 161
- ໂອເປົວເຮັດອົງ, 161
- ໂອວົງແລປ, 76

**ඇ**

හෙතු, 229

การออกแบบฐานข้อมูลเป็นทักษะที่สำคัญทุกคนในวงการไอที  
 เพราะองค์กรไม่ว่าขนาดเล็กหรือใหญ่ล้วนจำเป็นต้องใช้ฐานข้อมูล  
 ในการดำเนินธุรกิจหรือกิจกรรมใด ๆ ที่มีข้อมูลจำนวนมาก  
 หากออกแบบได้ไม่เหมาะสมอาจส่งผลให้การทำงานของระบบฐานข้อมูลนั้นล้มเหลวได้ง่าย<sup>1</sup>  
 ตามที่พื้นฐานการออกแบบระบบฐานข้อมูลนี้รวมความรู้พื้นฐานแนวคิด เครื่องมือ วิธีการ  
 และตัวอย่างการออกแบบระบบฐานข้อมูลเชิงสัมพันธ์

หมายเหตุ  
 1. สำหรับนิสิตระดับปริญญาตรีที่สนใจความรู้ด้านคอมพิวเตอร์  
 ที่ต้องเข้าใจการทำงานและการคำนวณภายในฐานข้อมูล  
 จึงจะออกแบบฐานข้อมูลได้อย่างมีประสิทธิภาพ  
 และมีพื้นฐานที่เข้มแข็งพอที่จะพัฒนาองค์ความรู้ใหม่ต่อไป  
 หรือประกอบวิชาชีพในองค์กรที่ใช้ฐานข้อมูลโดยเฉพาะฐานข้อมูลเชิงสัมพันธ์