บทที่ 3

การออกแบบฐานข้อมูลแบบเชิงสัมพันธ์ (Relational Database Design)

จากการวางแผนและวิเคราะห์ข้อมูล เมื่อได้แบบจำลองข้อมูลดังภาพในภาคผนวก ก. ซึ่งเป็นความสัมพันธ์ระหว่างเอนทิตีแล้ว ขั้นต่อมาก็ต้องทำการแปลงแบบจำลองข้อมูลให้เป็นฐานข้อมูลแบบเชิงสัมพันธ์ โดยอาศัยขั้นตอน และวิธีการจากเฟรมมิง [4] ดังนี้

การแปลงจากโครงร่างข้อมูลเชิงตรรกภาพ ให้เป็นฐานข้อมูลแบบเชิงสัมพันธ์ในเบื้องต้น (Translate The Logical Data Structure)

ในขั้นตอนนี้จะพิจารณาความเป็นไปได้ และองค์ประกอบต่าง ๆ ว่าครบถ้วนหรือไม่ ซึ่งจะกระทำได้ ดังนี้

1. กำหนดตาราง (Identify Tables)

สำหรับแต่ละเอนทิตีนั้นจะสามารถนำมาสร้างตารางได้ 1 ตาราง ดังนี้

ชื่อตาราง	ความหมาย	เอนทิตี
ABS_ANUAL	การลาประจำปีของบุคลากร	HIS_ANY_ABSENT
ABS_EDUCATION	การลาศึกษาต่อของบุคลากร	HIS_ABSENT
ABS_QUATAS	โควต้าของจำนวนวันลาของบุคลากร	ABS_QUATAS
ABS_TYPE	รายละเอียดประเภทของการลา	ANY_ABSENT
ADDRESS	ประวัติที่อยู่อาศัยของบุคลากร	PERS_ADDRESS
CHILD	ประวัติบุตร/ธิดาของบุคลากร	CHILD
CMD_TYPE	รายละเอียดประเภทของคำสั่ง	CMD_TYPE
COMMAND	ข้อมูลของคำสั่งต่าง ๆ	COMMAND
CONTACT	ประวัติผู้ที่สามารถติดต่อกับบุคลากรได้	CONTACT
COUNTRY	รายละเอียดของประเทศ	COUNTRY
CURRICULUM	รายละเอียดของหลักสูตรการศึกษา	CURRICULUM
DEPARTMENT	รายละเอียดของภาควิชาในคณะ	DEPARTMENT
DIVISION	รายละเอียดของฝ่ายงาน	DIVISION
DUTY	รายละเอียดของชื่อตำแหน่ง	DUTY

ตารางที่ 3.1 แสดงรายละเอียดของเอนทิตี กับตาราง

ชื่อตาราง	ความหมาย	เอนทิตี
EDUCATION	ประวัติการศึกษาของบุคลากร	HIS_EDUCATION
EDUCATION_LEVEL	รายละเอียดระดับการศึกษา	EDU_LEVEL
EXT_POSITION	ข้อมูลการดำรงตำแหน่งนอกมหาวิทยาลัย	EXT_POSITION
FACULTY	รายละเอียดของคณะ	FACULTY
FUND	รายละเอียดของแหล่งทุนการศึกษา	FUND
LOAN	รายละเอียดของวัตถุประสงค์การกู้	LOAN
MARITAL_STATUS	รายละเอียดของสถานภาพการสมรส	MARITAL_STATUS
NATION	รายละเอียดของสัญชาติ	NATION
OLD_NAME	ข้อมูลการเปลี่ยนชื่อ-สกุลของบุคลากร	OLD_NAME
ORGANIZATION	รายละเอียดขององค์กร	ORGANIZATION
PARENT	ประวัติบิดา/มารดาของบุคลากร	PARENT
PC_SALARY	ตารางระดับขั้น และขั้นเงินเดือน	PC_SALARY
PERSON	ประวัติส่วนตัวของบุคลากร	PERSON
PERSON_POSITION	ข้อมูลการดำรงตำแหน่งของบุคลากร	HIST_POSITION
PERSON_PROJECT	ข้อมูลการทำผลงานวิชาการของบุคลากร	PERS_PROJECT
PERSON_RESEARCH	ข้อมูลการทำผลงานวิจัยของบุคลากร	PERS_RESEARCH
PERSON_SALARY	ข้อมูลขั้นเงินเดือนของบุคลากร	HIS_SALARY
POSITION	ข้อมูลรายละเอียดของตำแหน่ง	POSITION
POSITION_STATUS	รายละเอียดของสถานภาพของตำแหน่ง	POS_STATUS
POS_LEVEL	รายละเอียดของระดับตำแหน่ง	POS_LEVEL
PROFESSION	รายละเอียดของประเภทอาชีพ	PROFESSION
PROJECT	ข้อมูลของผลงานทางวิชาการ	PROJECT
PROJECT_TYPE	รายละเอียดของประเภทผลงานทางวิชาการ	PROJ_TYPE
PROVINCE	รายละเอียดของจังหวัด	PROVINCE
PUNISHMENT	ข้อมูลการได้รับโทษทางวินัยของบุคลากร	HIS_PUNISHMENT
PUNISH_TYPE	รายละเอียดของประเภทการลงโทษทางวินัย	PUNISH_TYPE
RACE	รายละเอียดของเชื้อชาติ	RACE
RECEPT_REWARD	ข้อมูลการได้รับเครื่องราชฯ ของบุคลากร	HIS_RECEPT_REW
RELATION	รายละเอียดของความสัมพันธ์	RELATION
RELIGION	รายละเอียดของศาสนา	RELIGION
RESEARCH	ข้อมูลของผลงานวิจัย	RESEARCH
RESPONSE	ข้อมูลความรับผิดชอบของบุคลากร	HIS_RESPONSE

ตารางที่ 3.1(ต่อ)

ชื่อตาราง	ความหมาย	เอนทิตี
RES_FINISH_STATUS	รายละเอียดของสถานภาพการปิดผลงานวิจัย	RES_FINISH_STATUS
REWARD	รายละเอียดของเครื่องราชอิสริยาภรณ	REWARD
SALARY_GRADE	ตารางขั้นเงินเดือน และเงินเดือน	SALARY_TABLE
SPOUSE	ประวัติการสมรสของบุคลากร	HIS_MARITAL
SPOUSE_STATUS	รายละเอียดสถานภาพของคู่สมรส	SPOUSE_STATUS
TERMINATION	ข้อมูลการออกจากราชการของบุคลากร	HIS_TERMINATION
TERM_REASON	รายละเอียดของเหตุผลในการออกจากราชการ	TERM_REASON
WELFARE	ข้อมูลการกู้เคหะสงเคราะห์ของบุคลากร	HIS_WELFARE
WORK_TYPE	รายละเอียดของประเภทงานรับผิดชอบ	WORK_TYPE

ตารางที่ 3.1(ต่อ)

2. กำหนดคอลัมน์ (Identify Column)

ในแต่ละเอนทิตีจะมีแอตตริบิวต์ ซึ่งจะนำมาใช้ในการกำหนดเป็นคุณสมบัติของคอลัมน์ (ดูรายละเอียดได้ที่ภาคผนวก ข.)

จะเห็นได้ว่าในเบื้องต้นนี้แต่ละตาราง จะมีคอลัมน์ซึ่งประกอบด้วย คีย์หลัก คีย์รอง ฟอร์เรนจ์คีย์ รวมทั้งคอลัมน์ที่ไม่ใช่คีย์ ซึ่งการกำหนดคอลัมน์ของฟอร์เรนจ์คีย์นั้นมาจากการแปลงความสัมพันธ์ที่ได้จากแบบจำลองข้อมูลนั่นเอง และขณะนี้ตารางที่ได้อยู่ในรูปแบบนอร์มัล เพราะว่าแบบจำลองข้อมูลอยู่ในรูปแบบนอร์มัลแล้ว

3. ปรับโครงสร้างข้อมูลให้เข้ากับสภาพแวดล้อมที่นำมาใช้ (Adapt Data Structure to Product Environment)

ทุกระบบของฐานข้อมูลแบบเชิงสัมพันธ์ (RDBMS) จะมีการให้คำจำกัดความของตาราง โดยใช้ประโยคดีดีแอล (DDL=Data Definition Language) ซึ่งมีวากยสัมพันธ์ (Syntax) ที่ใช้อาจต่างกัน ในแต่ละระบบจึงต้องมีการปรับให้เข้ากับระบบที่ต้องการ เช่น พิจารณาการสร้างตารางเก็บในระบบฐานข้อมูล การเรียงคอลัมน์ ในตารางจากซ้ายไปขวา เป็นต้น ในการกำหนดตารางให้กับระบบการจัดการฐานข้อมูล (DBMS) นั้นยังต้องพิจารณาถึงพารามิเตอร์ที่มีผลกระทบกับการจัดสรร (allocate) และการจัดการของหน่วยเก็บ รวมทั้งเนื้อที่ที่ว่างด้วยซึ่งเนื้อที่ต้องมากเพียงพอ

แปลความเป็นบูรณภาพของข้อมูลเชิงตรรก (Translate the Logical Data Integrity)

กระบวนการแปลงในที่นี้คือ การบังคับให้เป็นไปตามกฎธุรกิจต่าง ๆ ซึ่งบางครั้งระบบจัดการฐานข้อมูลที่นำมาใช้ อาจจะสนับสนุนการทำงานบางอย่าง หรืออาจใช้กลไกอื่น ๆ ช่วย สำหรับขั้นตอนต่าง ๆ ในขั้นนี้ ได้แก่

1. การออกแบบคุณสมบัติเชิงตรรกของคีย์หลักของเอนทิตี (Design for Business Rules About Entities)

ในการออกแบบคุณสมบัติของคีย์หลัก เพื่อนำมาใช้งาน เช่น ความเป็นเอกลักษณ์ (Uniqueness) หรือการไม่ยอมให้มีค่าว่าง (Mandatory) ฯลฯ เป็นต้น โดยอาจกำหนดคุณสมบัติเหล่านี้โดยใช้ประโยคดีดีแอล หรือกำหนดโดยใช้เทคนิคการใช้ข้อกำหนดของโดเมน ซึ่งในระบบนี้ได้กำหนดคุณสมบัติเชิงตรรกของคีย์หลักของเอนทิตีไว้แล้วในการกำหนดคอลัมน์

2. การออกแบบคุณสมบัติเชิงตรรกของความสัมพันธ์ (Design for Business Rules About Relationships)

ในการออกแบบนี้เกี่ยวข้องกับข้อบังคับในการเพิ่ม ลบ แก้ไข คีย์แอตตริบิว โดยอาจกำหนดโดยใช้ประโยคดีดีแอล หรือใช้เทคนิคการทำทริกเกอร์ (Trigger) มิฉะนั้นอาจเป็นหน้าที่ หรือความรับผิดชอบของผู้ใช้ที่จะต้องปฏิบัติตามข้อบังคับ โดยคุณสมบัตินี้พิจารณาได้ดังนี้

- การเพิ่ม เป็นการพิจารณาสถานการณ์ เงื่อนไขที่สมเหตุสมผล ถูกต้อง เมื่อเราจะเพิ่มเอนทิตีลูก หรือแก้ไขฟอร์เรนจ์คีย์ในแต่ละสมาชิกในเอนทิตี ซึ่งสามารถจำแนกเงื่อนไขได้เป็น 6 แบบ ดังนี้

(1) โดยขึ้นต่อกัน (Dependent) กล่าวคือ จะยอมให้มีการเพิ่มสมาชิกของเอนทิตีลูก เมื่อสมาชิกในเอนทิตีแม่ที่ตรงกันมีอยู่

(2) โดยอัตโนมัติ (Automatic) กล่าวคือ ให้มีการเพิ่มสมาชิกของเอนทิตีลูกได้เสมอ ถ้าสมาชิกในเอนทิตีแม่ที่ตรงกันยังไม่มีสร้างขึ้นมา

(3) โดยให้ค่าว่าง (Nullify) กล่าวคือ ให้มีการเพิ่มสมาชิกของเอนทิตีลูกได้เสมอ ถ้าสมาชิกในเอนทิตีแม่ที่ตรงกันไม่มีอยู่ ให้ตั้งค่าฟอร์เรนจ์คีย์ในเอนทิตีลูกเป็นค่าว่าง

(4) โดยให้ค่าปริยาย (Default) กล่าวคือ ให้มีการเพิ่มสมาชิกของเอนทิตีลูกได้เสมอ ถ้าสมาชิกในเอนทิตีแม่ที่ตรงกันไม่มีอยู่ ให้ตั้งค่าฟอร์เรนจ์คีย์ในเอนทิตีลูกเป็นค่าที่กำหนดไว้ก่อน(ค่าปริยาย)

(5) โดยตามธรรมเนียม (Customized) กล่าวคือ ยอมให้มีการเพิ่มสมาชิกในเอนทิตีลูกก็ต่อเมื่อตรวจสอบถูกต้องตรงกับสถานการณ์ที่กำหนด

(6) ไม่มีผลกระทบใด ๆ (No Effect) กล่าวคือ ให้มีการเพิ่มสมาชิกของเอนทิตีลูกได้เสมอ โดยสมาชิกในเอนทิตีแม่ที่ตรงกันไม่จำเป็นต้องมีอยู่ และไม่ต้องมีการตรวจสอบใด ๆ

- การลบ เป็นการพิจารณาสถานการณ์ เงื่อนไขที่เป็นไปได้ที่จะสามาถลบเอนทิตีแม่ได้ถูกต้อง หรือแก้ไขคีย์หลักที่ถูกอ้างถึง โดยฟอร์เรนจ์คีย์ซึ่งสามารถจำแนกเงื่อนไขได้เป็น 6 แบบ ดังนี้

(1) โดยจำกัด (Restrict) กล่าวคือ จะยอมให้มีการลบสมาชิกของเอนทิตีแม่ เมื่อไม่มีสมาชิกในเอนทิตีลูกที่ตรงกันอยู่

(2) โดยต่อเนื่อง (Cascade) กล่าวคือ ให้มีการลบสมาชิกของเอนทิตีแม่ได้เสมอ และให้ลบสมาชิกในเอนทิตีลูกที่ตรงกันทุกสมาชิก

(3) โดยให้ค่าว่าง (Nullify) กล่าวคือ ให้มีการลบสมาชิกของเอนทิตีแม่ได้เสมอ ถ้ามีสมาชิกในเอนทิตีแม่ที่ตรงกัน ก็ให้ตั้งค่าฟอร์เรนจ์คีย์ในเอนทิตีนั้นเป็นค่าว่าง

(4) โดยให้ค่าปริยาย (Default) กล่าวคือ ให้มีการลบสมาชิกของเอนทิตีแม่ได้เสมอ ถ้ามีสมาชิกในเอนทิตีลูกที่ตรงกันอยู่ ก็ให้ตั้งค่าฟอร์เรนจ์คีย์ในเอนทิตีเป็นค่าที่กำหนดไว้ก่อน

(5) โดยตามธรรมเนียม (Customized) กล่าวคือ ยอมให้มีการลบสมาชิกในเอนทิตีแม่ก็ต่อเมื่อตรวจสอบตรงกับสถานการณ์ที่กำหนด

(6) ไม่มีผลกระทบใด ๆ (No Effect) กล่าวคือ ให้มีการลบสมาชิกของเอนทิตีแม่เสมอ โดยอาจมี หรือไม่มีสมาชิกในเอนทิตีลูกที่ตรงกัน และไม่จำเป็นต้องมีการตรวจสอบใด ๆ

ในการกำหนดกฎต่าง ๆ ควรกำหนดให้ตรงกับภาพในธุรกิจ ซึ่งในการออกแบบข้อบังคับของความสัมพันธ์ต่าง ๆ ในระบบนี้มีรายละเอียดดังนี้

PARENT	CHILD	CHILD Insert Rule	PARENT Delete Rule
PERSON	ADDRESS	Dependent	Restrict
PERSON	CONTACT	Dependent	Restrict
PERSON	PARENT	Dependent	Restrict
PERSON	SPOUSE	Dependent	Restrict
PERSON	CHILD	Dependent	Restrict
PERSON	EDUCATION	Dependent	Restrict
PERSON	OLD_NAME	Dependent	Restrict
PERSON	WELFARE	Dependent	Restrict
PERSON	PERSON_PROJECT	Dependent	Restrict
PERSON	PERSON_RESEARCH	Dependent	Restrict
PERSON	PERSON_POSITION	Dependent	Restrict
PERSON	PERSON_SALARY	Dependent	Restrict
PERSON	PUNISHMENT	Dependent	Restrict
PERSON	TERMINATION	Dependent	Restrict
PERSON	ABS_EDUCATION	Dependent	Restrict
PERSON	ABS_ANUAL	Dependent	Restrict
PERSON	ABS_QUATAS	Dependent	Restrict
PERSON	EXT_POSITION	Dependent	Restrict
PERSON	RESPONSE	Dependent	Restrict
PERSON	RECEPT_REWARD	Dependent	Restrict
POSITION	PERSON_POSITION	Dependent	Restrict
COMMAND	PERSON_POSITION	Dependent	Cascade
COMMAND	PUNISHMENT	Dependent	Cascade
COMMAND	TERMINATION	Dependent	Cascade
COMMAND	ABS_EDUCATION	Dependent	Cascade
COMMAND	PERSON_SALARY	Dependent	Cascade
REWARD	RECEPT_REWARD	Dependent	Restrict
PROJECT	PERSON_PROJECT	Dependent	Restrict
RESEARCH	PERSON_RESEARCH	Dependent	Restrict

รูปที่ 3.1 แสดงคุณสมบัติเชิงตรรกของความสัมพันธ์

3. ออกแบบกฎธุรกิจของแอตตริบิวต์ (Design for Additional Business Rules About Attributes)

การออกแบบกฎธุรกิจนี้อาจทำได้โดยใช้ทริกเกอร์ ประโยคดีดีแอล หรือใช้ข้อกำหนดของโดเมน และโดยปกติไม่ควรยอมให้คอลัมน์บางคอลัมน์เป็นค่าว่าง ได้แก่ คอลัมน์ที่เป็นคีย์หลัก ฟอร์เรนจ์คีย์ ซึ่งมักถูกใช้ในการเปรียบเทียบ และเชื่อมโยง (join) กับคอลัมน์ที่ใช้ในเอสคิวแอล (SQL=Structure Query Language) ที่มีการใช้ WHERE และคอลัมน์ที่เป็นตัวเลข เพราะจะเป็นปัญหาในการคำนวณ ซึ่งในการออกแบบระบบนี้ได้กำหนดกฎธุรกิจของแอตตริบิวต์ไว้พร้อม ๆ กับการสร้างคอลัมน์

<u>การปรับแต่งการออกแบบ (Tuning)</u>

เราสามารถใช้วิธีการหลายอย่าง เพื่อช่วยในการปรับเพื่อเพิ่มประสิทธิภาพ ในการทำงานได้ดังนี้

1. การวิเคราะห์กลไกในการเข้าถึง (Tuning Access Mechanism) โดยหาทางเลือกที่ให้ประโยชน์สูงสุด

2. การจัดรวบรวมข้อมูลให้เป็นหมู่ หรือกลุ่มเดียวกันไว้ด้วยกัน (Define Clustering Sequence) ในการจัดกลุ่มข้อมูลนี้จะพิจารณาทำกับตารางขนาดกลาง จนถึงขนาดใหญ่ (เช่น ตารางที่มีขนาดใหญ่กว่า 6 physical blocks) แต่อย่าทำการจัดกลุ่มข้อมูลให้กับตารางที่จะไปขัดขวางการทำงานบางอย่าง เช่น การทำให้เกิด Overhead ในการเพิ่ม แก้ไข หรือลบข้อมูล และไม่ควรทำการจัดกลุ่มข้อมูลตารางขนาดเล็ก นอกจากนี้ควรทำการจัดกลุ่มข้อมูลของตารางที่เชื่อมระหว่างคีย์หลัก และฟอร์เรนจ์คีย์ เพื่อประโยชน์ในการเชื่อม (Join) และควรเก็บสถิติการทำงานของตารางที่ได้จัดกลุ่มข้อมูล เพื่อใช้ในการปรับแต่งในภายหลัง แต่อย่างไรก็ตามก่อนที่จะทำการจัดกลุ่มข้อมูลนั้น ควรจะตอบคำถามเหล่านี้ให้ได้ก่อน เพื่อจะได้เป็นประโยชน์ในการใช้งานต่อไป

- ตารางอะไรที่ควรจะทำการจัดกลุ่มข้อมูล ?
- ควรจะจัดกลุ่มข้อมูลตามคอลัมน์ใด ?
- จะใช้เทคนิค หรือวิธีการใด เพื่อให้ประสิทธิภาพในการทำงานดีขึ้น และช่วยลด ค่าใช้จ่ายด้วย ?

3. การสร้างดัชนีเพิ่ม (Add Indexes) ในการสร้างดัชนีมีข้อแนะนำในการสร้างคือ ควรสร้างดัชนีในตารางที่มีขนาดกลาง จนถึงขนาดใหญ่ หรือถ้าในการเรียกใช้ หรือดึงข้อมูลในตารางนั้น ๆ แล้วได้ผลลัพธ์ประมาณ 20% ของจำนวนแถว และในการประเมินประสิทธิภาพของดัชนีนั้นจะต้องพิจารณาผลกระทบในการเพิ่ม ลบ แก้ไข ตลอดจนการประมวลผลต่าง ๆ นอกจากนี้แล้วยังต้องพิจารณาค่าใช้จ่ายเนื่องจากความต้องการหน่วยเก็บอีกด้วย

4. การยอมให้ข้อมูลมีการซ้ำซ้อนโดยมีการควบคุม (Tune by Introducing Redundancy Controlled Data) สิ่งที่สำคัญที่ต้องพิจารณาการยอมให้มีข้อมูลซ้ำ คืออาจมีเหตุผลที่ทำให้ เอนทิตีหนึ่ง ๆ ที่วิเคราะห์ออกมาอาจจะไม่กลายเป็นตาราง 1 ตาราง ในฐานข้อมูลเสมอไป ซึ่งมีเหตุผล 2 ประการ

โดยทั่วไป คือ

(1) เอนทิตีที่ถูกกำหนดขึ้นมาไม่ได้เกี่ยวพันกับการใช้งานด้วยคอมพิวเตอร์ ขณะที่กำลังทำงานอยู่ ซึ่งเราอาจจะตัดสินใจ หรือไม่ก็ได้ว่าจะยกเอนทิตีนี้เป็นตารางต่อไป ถ้าหากมีการขยายการใช้งาน

(2) อาจต้องการรวมเอนทิตี 2 เอนทิตี หรือมากกว่าเป็นเพียง 1 ตาราง เพื่อเพิ่มประสิทธิภาพในการทำงาน เช่น การที่กำหนดเอนทิตีเป็นสิ่งสำคัญที่มองแยกออกไป แต่ในโครงสร้างของตารางจะรวมกันให้อยู่ในรูปดีนอร์มัล (Denormal) ซึ่งจะก่อให้เกิดความซ้ำซ้อนของฐานข้อมูลขึ้นใหม่ แต่ก่อนที่จะพิจารณาการดีนอร์มัล จะต้องผ่านกระบวนการทำนอร์มัลในระดับที่ 3 ก่อน และจะต้องตัดสินใจมากขึ้น เพื่อให้แน่ใจว่าข้อมูลที่ซ้ำกันนั้นไม่ลักหลั่นกัน ซึ่งถ้าทำให้อยู่ในรูปดีนอร์มัลแล้ว อาจทำให้การพัฒนางานง่ายขึ้น และทำให้เพิ่มประสิทธิภาพการเรียกดู หรือค้นหาข้อมูล ไม่ต้องสืบค้นหลายตาราง หรือไม่เสียเวลาในการคำนวณ เป็นต้น แต่อย่างไรก็ตามต้องพิจารณาผลดี ผลเสีย ที่เกิดขึ้น ซึ่งจะมีผลกระทบกับหน่วยความจำ การดึงข้อมูลด้วยประโยคดีเอ็มแอล (DML = Data Manipulate Language) ที่ใช้ความเป็นบูรณภาพของข้อมูล ตลอดจนประสิทธิภาพในการใช้งาน ดังนั้นคอลัมน์ที่ซ้ำซ้อนอาจได้มาจาก

- คอลัมน์ที่ลอกมาจากคอลัมน์ที่มีอยู่แล้วในตารางอื่น เช่น ถ้ามีตารางเก็บประวัติการทำงาน แต่เราอาจเก็บการทำงานปัจจุบันในตารางพนักงานอีกด้วย ทำให้มีข้อมูลปรากฏ 2 แห่ง เป็นต้น
- คอลัมน์ที่เป็นผลสรุปมาจากคอลัมน์อื่น ๆ (derived column) ซึ่งมีความสำคัญต่อผู้ใช้ทำให้สามารถตอบคำถามในการสอบถามได้เร็วขึ้น แต่การยอมให้ผู้ใช้แก้ไข เปลี่ยนแปลงค่า ควรกระทำที่คอลัมน์ที่เป็นตัวกำเนิด หรือที่มาเท่านั้น
- คอลัมน์ที่เป็นกลุ่มซ้ำ เมื่อกลุ่มซ้ำนั้นมีจำนวนแน่นอนและกลุ่มซ้ำนั้นมักถูกเรียกใช้ หรือแก้ไข พร้อม ๆ กัน เช่น ถ้าทราบว่าพนักงานแต่ละคนมีทักษะ 2 อย่างเสมอ เราอาจเก็บข้อมูลเป็น 2 คอลัมน์ในตาราง แต่ปัญหาก็คือ หากภายหลังต้องการเพิ่มประเภททักษะชนิดที่ 3 จะจัดการอย่างไร ดังนั้นต้องพิจารณาผลดี ผลเสีย ในการจำกัดความยืดหยุ่น
- คอลัมน์อย่างย่อที่สร้างขึ้นมาเพื่อแทนคอลัมน์ที่มีอยู่ เช่น ในกรณีที่คีย์หลัก ประกอบด้วยหลายคอลัมน์และยาว อาจสร้างเป็นคอลัมน์เพื่อแทนคีย์หลักนั้น ๆ โดยอาจใช้เป็นรหัสในการอ้างถึง

5. พิจารณาโครงสร้างของฐานข้อมูลใหม่ (Tune by Redefining the Relation Database Structure) มีวิธีการดำเนินการอยู่ 2 ลักษณะ คือ

(1) การกำหนดคอลัมน์ใหม่ (Redefine Columns) เนื่องจากมีความจำเป็นต้องปรับโครงสร้างให้เข้ากับข้อจำกัดของดีบีเอ็มเอสที่นำมาใช้งาน เช่น การกำหนดชนิดข้อมูลให้คอลัมน์เป็นแบบ long หรือมีความยาวแบบไม่จำกัด (variable length) กระทำไม่ได้ เป็นต้น จึงต้องหาวิธีแก้ไข โดยการสร้างตารางเพื่อรองรับขึ้นมา

ดังเช่นในการออกแบบระบบนี้ เนื่องจากต้องใช้ภาษาไทย ข้อมูลที่เป็นวันที่จึงต้องการให้อยู่ในรูปแบบของวันที่แบบพุทธศักราช แต่เนื่องจาก ORACLE RDBMS ที่นำมาใช้งานนั้นไม่รองรับการเก็บข้อมูลวันที่ที่เป็นแบบพุทธศักราช แต่รองรับวันที่ที่เป็นแบบคริสตศักราช จึงทำให้ต้องมีการกำหนด

คอลัมน์ที่มีชนิดของข้อมูลเป็นวันที่ใหม่ โดยให้เป็นข้อมูลชนิดอักขระ (Character) แทน เพื่อที่จะได้สามารถบันทึกข้อมูลที่เป็นวันที่แบบพุทธศักราชได้ ส่วนการตรวจสอบความถูกต้องนั้น กระทำโดยการใช้ฟอร์ม (Form) ที่ใช้ในการบันทึกข้อมูล แต่ถ้าผู้ใช้ไม่ได้นำข้อมูลเข้าโดยการใช้ฟอร์มแล้ว ผู้ใช้ต้องตรวจสอบความถูกต้องของข้อมูลเอง

(2) การกำหนดตารางใหม่ (Redefine Tables) เพื่อทำให้การทำงานเร็วขึ้น ซึ่งมีวิธีการต่าง ๆ เช่น การเพิ่มตาราง ได้แก่ ตารางที่สรุปมาจากตารางอื่น เป็นต้น การแบ่งตารางออกไป (Segmenting) ตามแนวนอน หรือแนวตั้ง ซึ่งบางครั้งทำให้การใช้ประโยคดีเอ็มแอลซับซ้อนขึ้น อีกทั้งต้องระบุกฎเงื่อนไขต่าง ๆ เพิ่ม นอกจากนี้ยังมีวิธีการรวมตารางเข้าด้วยกัน อาจเป็นในลักษณะแนวนอนหรือแนวตั้ง คือรวมตารางแม่ กับลูก ซึ่งการที่จะควบคุมความเป็นบูรณภาพของข้อมูลยุ่งยากขึ้น เพราะไม่ได้อยู่ในรูปแบบนอร์มัล และคอลัมน์อาจจะมีค่าว่างเกิดขึ้น

ดังนั้นจากการออกแบบระบบนี้ทำให้เกิดการแยกตาราง PERSON ออกมาเป็น 2 ตาราง ทั้งนี้เนื่องจากตาราง PERSON นี้ มีการค้นหารหัสบุคลากร ชื่อ และนามสกุล บ่อยมาก ซึ่งจะเห็นได้ว่าตารางนี้มีคอลัมน์มาก และคอลัมน์ที่นอกเหนือจากรหัสบุคลากร ชื่อ และ นามสกุล มักจะไม่ได้ถูกค้นหาบ่อย จึงทำให้การค้นหาเป็นไปได้ช้า ฉะนั้นจึงทำการเพิ่มตารางขึ้นมาใหม่ โดยแยกคอลัมน์ที่เกี่ยวกับข้อมูลอ้างอิงอื่น ได้แก่ เลขประจำตัวบัตรประชาชน (CITIZEN_ID) วันเกิด (BIRTH_DAY) รหัสจังหวัดที่เกิด (BIRTH_PLACE) เลขประจำใบสูติบัตร (BIRTH_ID) รหัสเชื้อชาติ (RACE_CODE) รหัสสัญชาติ (NATION_CODE) รหัสศาสนา (RELIGION_CODE) เลขประจำบัตรผู้เสียภาษี (TAX_ID) รหัสสถานภาพการสมรส (MARITAL_STATUS) รหัสจังหวัดภูมิลำเนา (DOMICLINE_CODE) รหัสโซนของห้องทำงาน (ROOM_ZONE_CODE) รหัสตึกของห้องทำงาน (ROOM_BUILD_CODE) รหัสชั้นของห้องทำงาน (ROOM_FLOOR_CODE) รหัสห้องทำงาน (ROOM_CODE) หมายเลขโทรศัพท์ของที่ทำงาน (OFFICE_TEL) ปีงบประมาณที่เกษียณอายุราชการ (EXPIRE_YEAR) เงินช่วย พ.ส.ร. (PSR) ออกมาเป็นอีกตารางหนึ่งโดยให้มีชื่อว่า **"HISTORY_PERSON"**

<u>การอนุญาตให้เข้าถึงข้อมูลโดยการจำกัดมุมมองข้อมูลของผู้ใช้</u>

นอกจากหัวข้อในการออกแบบดังที่กล่าวมาข้างต้น ยังต้องมีการพิจารณาในส่วนอื่น ๆ อีกเกี่ยวกับการอนุญาตให้เข้าถึงข้อมูลโดยใช้วิว (มุมมองข้อมูลของผู้ใช้) วิวเป็นเสมือนหน้าต่างบนฐานข้อมูล ซึ่งแสดงในลักษณะตารางความสัมพันธ์ 2 มิติ (แถว/คอลัมน์) โดยเป็นผลมาจากการเชื่อมโยง (join) หรือการปฏิบัติการแบบเชิงสัมพันธ์ (relational operation) ที่มาจากตารางจริง (base table) ในฐานข้อมูล หรืออาจได้มาจากวิวอีกที่ก็ได้ แต่ค่าของข้อมูลจะเก็บในตารางจริงเท่านั้น การให้ผู้ใช้ดึงข้อมูลโดยใช้วิวมีประโยชน์ดังนี้

(1) วิวสามารถแทนตารางให้กับผู้ใช้ในลักษณะใดก็ได้ โดยไม่มีผลกระทบต่อโครงสร้างข้อมูลจริง เช่น อาจทำให้ข้อมูลที่อยู่ในรูปแบบนอร์มัลดูเหมือนดีนอร์มัล เป็นต้น

019507

(2) ทำให้การเรียกดูข้อมูลง่ายขึ้น วิวสามารถถูกกำหนดให้มาจากการเชื่อมตารางหลายตาราง ทำการคำนวณต่าง ๆ หรือเลือกเฉพาะคอลัมน์ หรือแถวที่ต้องการก็ได้

(3) สามารถใช้วิวในการกำหนดคำเหมือน (synonym) สำหรับตาราง รวมทั้งสำหรับคอลัมน์ และข้อมูลที่คำนวณได้ต่าง ๆ

(4) ทำให้สามารถจำกัดอำนาจในการเข้าถึงข้อมูล

แต่อย่างไรก็ตามการสร้างวิว โดยไม่มีการจัดการที่ดีจะทำให้เกิดปัญหาต่าง ๆ ได้ เช่น การเกิดวิวจำนวนมาก ยากแก่การดูแล ชื่อข้อมูลมีมาก อาจทำให้เกิดการสับสน และต้องมีการสร้างโปรแกรมเพิ่มขึ้นในการจัดการกับวิวต่าง ๆ

โดยทั่วไปจะใช้วิว และคำสั่งดีซีแอล เพื่อป้องกันในการเข้าถึงข้อมูล โดยที่วิวจะระบุแถว หรือคอลัมน์ที่เฉพาะตามที่ต้องการ และคำสั่ง GRANT/REVOKE จะระบุฟังก์ชันสำหรับผู้ใช้ในการใช้วิว หรือตาราง ถ้าต้องการจำกัด หรือให้อำนาจในการใช้ตารางทั้งตาราง เราอาจอนุญาตผ่านวิวหลัก (master view) หมายถึง วิวซึ่งรวมทุกคอลัมน์ทุกแถวจาก 1 ตาราง และถ้าต้องการควบคุมมากขึ้นที่ระดับแถว หรือคอลัมน์ ก็อาจกำหนดวิวในระดับถัดมา คือวิวในระดับที่ 2 เหนือวิวหลักขึ้นไป กล่าวคือ เป็นวิวที่ถูกกำหนดมาจากวิวอีกทีหนึ่ง ซึ่งจะจำกัดการใช้งานแล้วแต่ความเหมาะสม และนอกจากนี้ปกติความต้องการความปลอดภัยจะเป็นการจำกัดการเข้าถึงแถวของตาราง โดยขึ้นกับรหัสผู้ใช้ (user id) ตัวอย่างเช่น ในการดึงข้อมูลจากตารางบุคลากรโดยจำกัดตามแต่ละหน่วยงาน ให้เรียกใช้ได้เฉพาะข้อมูลในส่วนของหน่วยงานของตนเอง ซึ่งวิธีการดำเนินการ คือการกำหนดวิวต่าง ๆ สำหรับแต่ละหน่วยงานนั้น ถ้ามีหน่วยงานมากวิวก็ย่อมมาก ทำให้ยุ่งยาก และการออกแบบโปรแกรมจะซับซ้อนขึ้นด้วย ดังนั้นการแก้ปัญหานี้ ก็คือการใช้วิธีกำหนดตารางระบุสิทธิในการทำงาน จากนั้นจึงกำหนดวิวซึ่งมาจากการเชื่อม (join) จากตารางระบุสิทธิการใช้งานของผู้ใช้ และรหัสคณะที่ผู้ใช้นั้น ๆ สังกัดอยู่กับตาราง หรือวิวหลักที่ต้องการใช้ ดังนั้นจากการเชื่อมจะได้เพียงแถวที่ผู้ใช้คนนั้นมีสิทธิในการเรียกใช้เท่านั้น และจะไม่ทำให้จำนวนวิวเพิ่มขึ้น ถึงแม้ว่าจะมีหน่วยงานเพิ่มขึ้นก็ตาม