การประยุกต์ภาษาเอสคิวแอลกับข้อมูลเชิงเวลาโดยใช้ ORACLE FLASHBACK DATA ARCHIVE

THE APPLICATION OF SQL ON TEMPORAL DATA USING ORACLE FLASHBACK DATA ARCHIVE

นิธี นวลแก้ว

พิชามญชุ์ ยินดีพน

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2557

การประยุกต์ภาษาเอสคิวแอลกับข้อมูลเชิงเวลาโดยใช้ ORACLE FLASHBACK DATA ARCHIVE

THE APPLICATION OF SQL ON TEMPORAL DATA USING ORACLE FLASHBACK DATA ARCHIVE

นิธี นวลแก้ว

พิชามญชุ์ ยินดีพบ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2557

ปริญญานิพนธ์ปีการศึกษา 2557

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การประยุกต์ภาษาเอสคิวแอลกับข้อมูลเชิงเวลาโดยใช้ ORACLE FLASHBACK DATA ARCHIVE

THE APPLICATION OF SQL ON TEMPORAL DATA USING ORACLE FLASHBACK DATA ARCHIVE

ผู้จัดทำ

1. นายนิธี นวลแก้ว รหัสนักศึกษา 54010714
2. นางสาวพิชามญชุ์ ยินดีพบ รหัสนักศึกษา 54010917

อาจารย์ที่ปรึกษา

(รศ. ดร. ศุภมิตร จิตตะยโศธร)

การประยุกต์ภาษาเอสคิวแอลกับข้อมูลเชิงเวลาโดยใช้ ORACLE FLASHBACK DATA ARCHIVE

นายนิธี นวลแก้ว 54010714

นางสาวพิชามญชุ์ ยินดีพบ 54010917

รศ. ดร. ศุภมิตร จิตตะยโศธร อาจารย์ที่ปรึกษา

ปีการศึกษา 2557

บทคัดย่อ

คลังข้อมูล (Data Warehouse) ได้เข้ามามีบทบาทและมีส่วนช่วยในการเพิ่มศักยภาพการตัดสินใจทางธุรกิจ (Business Intelligence) ซึ่งมีความสำคัญอย่างยิ่งสำหรับองค์กรธุรกิจในปัจจุบัน โดยทั่วไปการสร้างคลังข้อมูลเป็นการรวบรวมข้อมูลจากฐานข้อมูลระบบงานประจำวัน (Operational database) ที่เป็นฐานข้อมูลปกติ (Non-temporal database) ซึ่งฐานข้อมูลชนิดนี้จะเก็บเพียงแค่ข้อมูลที่เป็นเวอร์ชันปัจจุบันเท่านั้น ก่อนจะสร้างคลังข้อมูลจำเป็นที่จะต้องทำการดึงข้อมูล (extract) เวอร์ชันต่างๆ ของฐานข้อมูลเก็บไว้ในไฟล์หรือหน่วยความจำอื่นตามช่วงเวลาต่างๆ ซึ่งจะมีคาบเวลาในการดึงข้อมูลที่ค่อนข้างแน่นอน เมื่อทำการสร้างคลังข้อมูลก็เพียงดึงข้อมูลเวอร์ชันเหล่านั้นจากฐานข้อมูลตามที่ต้องการมารวมไว้ในฐานข้อมูลเดียวกันและนำวิเคราะห์ร่วมกัน แต่หากฐานข้อมูลระบบงานประจำวันเป็นฐานข้อมูลเชิงเวลา (Temporal database) จะไม่มีความจำเป็นที่ต้องทำการดึงข้อมูลเวอร์ชันปัจจุบันของฐานข้อมูลไปลงไฟล์ เนื่องจากฐานข้อมูลเชิงเวลามีความสามารถในการสืบค้นข้อมูลที่เป็นปัจจุบันหรือข้อมูลในอดีต ดังนั้นเพียงแค่คิดค้นอัลกอริทึมในการสรุปข้อมูลเหล่านั้นไปสร้างเป็นคลังข้อมูลก็เพียงพอแล้ว อีกทั้งคลังข้อมูลที่สร้างขึ้นยังสามารถใช้แหล่งข้อมูล (Data source) ที่เป็นข้อมูลเชิงเวลา (Temporal data) ได้อีกด้วย

โครงงานนี้จึงได้นำเสนอการสร้างคลังข้อมูลที่ข้อมูลทั้งหมดมาจากฐานข้อมูลระบบงานประจำวันเดียวกัน ซึ่งประกอบไปด้วยแหล่งข้อมูลที่เป็นฐานข้อมูลเชิงเวลาและแหล่งข้อมูลที่เป็นฐานข้อมูลปกติ โดยมีวิธีในการสรุปข้อมูลตามระดับของเวลา (Granularity) ที่แตกต่างกันเพื่อให้ได้ข้อสรุปการวิเคราะห์ข้อมูลตามเรื่องที่ผู้ใช้สนใจ โดยได้นำคุณสมบัติของ Oracle Flashback Data Archive ของผลิตภัณฑ์ Oracle database ที่มีความสามารถในการเก็บเวอร์ชันของข้อมูลในช่วงเวลาต่างๆมาใช้ จึงลดความยุ่งยากในขั้นตอนการทำ ETL ของคลังข้อมูลรูปแบบปกติได้ อีกทั้งยังเป็นการเพิ่มขีดความสามารถให้กับคลังข้อมูลรูปแบบปกติอีกด้วย

THE APPLICATION OF SQL ON TEMPORAL USING ORACLE FLASHBACK DATA ARCHIVE

Ms. Nithee Nuankaew 54010714

Miss Pichamol Yindeephop 54010917

Assoc. Prof. Dr. Suphamit Chittayasothorn Advisor

Academic Year 2014

ABSTRACT

Nowadays, Data Warehouse has been playing a major role in Business Intelligence which is especially important in management business process. Data Warehouse can be optimized to consolidate data from many operational databases, generally non-temporal databases also has a current version. Before data are loaded into the data warehouse, the version of extracted data from those source system is stored in files or memory at period of time, which are almost certainly defined time to extract. When creating the data warehouse, application sides load version of data from the database to include data in the same system for analysis together. If operational database is database with built-in support for handling data involving time called temporal database, it would not be necessary to extract current version from database into files due to the temporal database has the ability to retrieve both current and historical data. Consequently, algorithm for data analysis conclusion is required to create data warehouse. Moreover, the data warehouse which is created can also use data as a temporal data from system source.

This project proposes to build data warehouse that all data is loaded from the same operational database, which may be include either temporal database source, non-temporal database source database or both. These have approach of data summary according time granularity in order to obtain data analysis for subject area. Flashback Data Archive feature in Oracle database which has been applied in this application has the ability to store version of data at period of time, so reduce complexity of the ETL process in general data warehouse. Furthermore, it makes system more efficient.

กิตติกรรมประกาศ

โครงงานการประยุกต์ภาษาเอสคิวแอลกับข้อมูลเชิงเวลาโดยใช้ Oracle Flashback Data Archive ฉบับนี้สำเร็จลุล่วงไปได้ด้วยดี เนื่องจากได้รับอนุเคราะห์อย่างดียิ่งจาก รศ.ดร.ศุภมิตร จิตตะยโศธร ซึ่งเป็นอาจารย์ที่ปรึกษาโครงงาน ผู้ซึ่งให้คำแนะนำ คำสั่งสอน อีกทั้งยังชี้แนะแนวทางการทำงาน ชี้ข้อปรับปรุงแก้ไข และติดตามความก้าวหน้าของโครงงานนี้อย่างสม่ำเสมอมาโดยตลอด

ขอขอบพระคุณคณาจารย์ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังทุกท่าน ที่ได้อบรมสั่งสอนวิชาความรู้ต่างๆ แก่ข้าพเจ้ามาโดยตลอด ทำให้ข้าพเจ้าสามารถนำความรู้เหล่านั้นมาใช้พัฒนาโครงงานนี้จนสำเร็จลุล่วงไปได้ด้วยดี

ขอขอบคุณรุ่นพี่ รวมทั้งเพื่อนๆ ทุกคนที่มีส่วนร่วมประติดประต่อความรู้ความสามารถแลกเปลี่ยนความคิดเห็นและแนะนำการทำโครงงานในด้านต่างๆ ตลอดมา

ขอขอบคุณบิดา มารดาและครอบครัว ที่ให้การอบรมสั่งสอน เลี้ยงดู ให้โอกาสทางการศึกษา และให้การสนับสนุนในทุกๆ ด้านมาโดยตลอด

สุดท้ายนี้ขอขอบพระคุณสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ที่ให้ข้าพเจ้าได้เข้ามาศึกษาหาความรู้ที่นี่ ข้าพเจ้ารู้สึกเป็นเกียรติอย่างยิ่ง คุณความดีใดๆ ที่ปรากฏในโครงงานนี้ ข้าพเจ้าขอมอบแด่ผู้มีพระคุณทุกท่านมา ณ ที่นี้

นิธี นวลแก้ว

พิชามญชุ์ ยินดีพบ

สารบัญ

หน้า

บทคัดย่อ I

ABSTRACT II

สารบัญ IV

สารบัญตาราง VI

สารบัญรูป VIII

บทที่ 1 บทนำ 1

1.1 ความสำคัญและที่มาของโครงงาน 1

1.2 วัตถุประสงค์ของโครงงาน 2

1.3 ขอบเขตของโครงงาน 3

1.4 วิธีการดำเนินงาน 4

1.5 ประโยชน์ที่คาดว่าจะได้รับ 4

บทที่ 2 แนวคิดและทฤษฎีที่เกี่ยวข้อง 5

2.1 การทบทวนวรรณกรรม (Literature Review) 5

2.2 ฐานข้อมูลเชิงเวลา (Temporal Database) 8

2.3 การจัดการกับฐานข้อมูลเชิงเวลาด้วยมาตรฐานภาษาเอสคิวแอล 2011 (Temporal Feature in SQL 2011) 32

2.4 เทคโนโลยีของ Oracle Flashback (Oracle Flashback Technology) 49

2.5 ฐานข้อมูลเชิงเวลาใน Oracle Database 12c 61

2.6 ระบบฐานข้อมูลที่ใช้ในองค์กร 67

2.7 คลังข้อมูล 69

บทที่ 3 การออกแบบและพัฒนาซอฟต์แวร์ 76

3.1 Oracle Flashback Data Archive 76

3.2 Slowly Changing Dimensions ใน Oracle Flashback Data Archive 78

3.3 การสรุปข้อมูลที่เป็นแหล่งข้อมูลสำหรับการทำคลังข้อมูลเมื่อเวลาผ่านไป 80

สารบัญ(ต่อ)

หน้า

3.4 ขั้นตอนในการสรุปข้อมูลจาก Oracle Flashback Data Archive มาสร้างเป็น คลังข้อมูล 87
บทที่ 4 การทดลองและผลการทดลอง.....96
4.1 การทดลองการสรุปข้อมูลในคลังข้อมูลที่ได้รวบรวมข้อมูลมาจากแหล่งข้อมูล ประเภทต่างๆ 96
บทที่ 5 บทสรุปและข้อเสนอแนะ.....112
5.1 บทสรุปของโครงงาน 112
5.2 ปัญหาอุปสรรคและแนวทางแก้ไข.....113
5.3 แนวทางในการพัฒนาต่อ 114

บรรณานุกรม

สารบัญตาราง

ตาราง หน้า

2.1 สถานะของคนไข้ในโรงพยาบาลแห่งหนึ่ง [ที่มา Managing Temporal Data A Five-Part Series] 9

2.2 ตัวอย่างความซ้ำซ้อนแบบ Nonsequenced duplicates [ที่มา Managing Temporal Data A Five-Part Series] 10

2.3 ตัวอย่างความซ้ำซ้อนแบบ Value-equivalent duplicates [ที่มา Managing Temporal Data A Five-Part Series] 10

2.4 ตัวอย่างความซ้ำซ้อนแบบ Current duplicates [ที่มา Managing Temporal Data A Five-Part Series] 11

2.5 ตัวอย่างความซ้ำซ้อนแบบ Sequenced duplicates [ที่มา Managing Temporal Data A Five-Part Series] 11

2.6 รายละเอียดของฝูงวัว [ที่มา Managing Temporal Data A Five-Part Series] 12

2.7 ผลลัพธ์ของการสอบถามข้อมูลทั้งสองแบบข้างต้น [ที่มา Managing Temporal Data A Five-Part Series] 13

2.8 ผลลัพธ์จากการทำ Sequence Query [ที่มา Managing Temporal Data A Five-Part Series] 14

2.9 สถานะของฝูงวัว [ที่มา Managing Temporal Data A Five-Part Series] 18

2.10 สถานะของฝูงวัวใช้สำหรับตัวอย่าง Current Delete [ที่มา Managing Temporal Data A Five-Part Series] 19

2.11 ผลลัพธ์สถานะของฝูงวัว หลังจากการทำ Current Delete [ที่มา Managing Temporal Data A Five-Part Series] 20

2.12 WDS บันทึกตำแหน่งและความสว่างของดวงดาว [ที่มา Managing Temporal Data A Five-Part Series] 30

2.13 Audit log ของ WDS (WDS_TT) [ที่มา Managing Temporal Data A Five-Part Series] 30

2.14 Bitemporal (WDS_B) [ที่มา Managing Temporal Data A Five-Part Series] 31

2.15 ผลลัพธ์จากการจัดเก็บข้อมูล ในตาราง Application-time period ชื่อ Emp [ที่มา Temporal features in SQL: 2011] 34

2.16 ผลลัพธ์ของการปรับปรุงข้อมูล ในตาราง Application-time period ชื่อ Emp [ที่มา Temporal features in SQL: 2011] 35

สารบัญตาราง(ต่อ)

ตาราง หน้า

2.17 ผลลัพธ์ของการลบข้อมูล ในตาราง Application-time period ชื่อ Emp [ที่มา Temporal features in SQL: 2011]....36

2.18 ตาราง Application-time period ชื่อ Emp สำหรับเรื่อง Primary keys [ที่มา Temporal features in SQL: 2011]....37

2.19 ตาราง Application-time period ชื่อ Emp สำหรับเรื่อง Referential constraints [ที่มา Temporal features in SQL: 2011]....38

2.20 ตัวอย่างตาราง Dept [ที่มา Temporal features in SQL: 2011]....38

2.21 ผลลัพธ์จากการ Insert ในตาราง System-versioned ชื่อ Emp [ที่มา Temporal features in SQL: 2011]....43

2.22 ผลลัพธ์จากการปรับปรุง ในตาราง System-versioned ชื่อ Emp [ที่มา Temporal features in SQL: 2011]....44

2.23 ผลลัพธ์จากการ Delete ในตาราง System-versioned ชื่อ Emp [ที่มา Temporal features in SQL: 2011]....45

2.24 รายละเอียดเกี่ยวกับ Undo Data ใน Tablespace [ที่มา **http://docs.oracle.com/databas**e/121/ADMIN/undo.htm#ADMIN10180]....57

สารบัญรูป

รูป หน้า

1.1 โครงสร้างของคลังข้อมูลที่ใช้ Flashback Data Archive 3
2.1 แสดงลักษณะของกรณีที่ 1 (ที่มา Managing Temporal Data A Five-Part Series) 14
2.2 แสดงลักษณะของกรณีที่ 2 (ที่มา Managing Temporal Data A Five-Part Series) 15
2.3 แสดงสถานะของวัวก่อนถูกตอนแยกตามเพศ (ที่มา Managing Temporal Data A Five-Part Series) 18
2.4 กรณีของการปรับปรุงเวลาของการเปลี่ยนแปลงในช่วงเวลาปัจจุบัน (ที่มา Managing Temporal Data A Five-Part Series) 21
2.5 กรณีของลบช่วงเวลาแบบ Sequence (Sequence delete) ทั้ง 4 กรณี (ที่มา Managing Temporal Data A Five-Part Series) 23
2.6 กรณีของปรับปรุงช่วงเวลาแบบ Sequence (Sequence Update) ทั้ง 4 กรณี (ที่มา Managing Temporal Data A Five-Part Series) 25
2.7 Allen's interval algebra (ที่มา SQL for date ranges, gaps and overlaps) 41
2.8 การทำงานของ Flashback Data Archive (ที่มา Oracle Total Recall with Oracle Database 11g Release 2) 51
2.9 ผลลัพธ์จากการใช้ Flashback Query 55
2.10 ผลลัพธ์จากการใช้ Flashback Versions Query 56
2.11 การสร้าง Flashback Data Archive (ที่มา Oracle Total Recall Tips) 58
2.12 การเปิดการใช้งาน Flashback Data Archive เข้ากับตารางที่ต้องการจะเก็บข้อมูลในอดีต (ที่มา Oracle Total Recall Tips) 58
2.13 การเรียกดูเวลาที่มีข้อมูลอยู่ในตารางแล้ว (ที่มา Oracle Total Recall Tips) 59
รูป2.14 การเรียกดู Undo tablespace อันเดิม ก่อนจะสร้างอันใหม่แทนที่ (ที่มา Oracle Total Recall Tips) 59
2.15 การสร้าง Undo tablespace อันใหม่พร้อมทั้งเปิดการ เริ่มใช้งาน Undo tablespace อันใหม่นี้ (ที่มา Oracle Total Recall Tips) 59
2.16 การใช้ Flashback Query (ที่มา Oracle Total Recall Tips) 60
2.17 การกำหนดระยะเวลาในการจัดเก็บข้อมูลใน Flashback Data Archive 60
2.18 ช่วงเวลาที่ชนิดข้อมูลเป็นคาบเวลาในฐานข้อมูล 62

สารบัญรูป(ต่อ)

รูป หน้า

2.19 ผลลัพธ์ของการสอบถามด้วยคำสั่ง Select ข้างต้น....62
2.20 ตัวอย่างการใช้ฟังก์ชันมาตรฐาน....66
2.21 การทำฐานข้อมูลกลาง....68
2.22 ตัวอย่างโปรแกรมประยุกต์และเรื่องที่ต้องการวิเคราะห์ในคลังข้อมูล (ที่มา W.H Inmon Buiding the Data Warehouse Third Edition)....69
2.23 การรวมข้อมูลจากแต่ละโปรแกรมประยุกต์ไปเป็นเรื่องที่สนใจในคลังข้อมูล (ที่มา W.H Inmon Buiding the Data Warehouse Third Edition)....70
2.24 ความแตกต่างระหว่างฐานข้อมูลและคลังข้อมูล (ที่มา W.H Inmon Buiding the Data Warehouse Third Edition)....71
2.25 Day-1 to Day-n pheromenon (ที่มา W.H Inmon Buiding the Data Warehouse Third Edition)....71
2.26 การรวบรวมข้อมูลก่อนที่จะนำเข้าสู่คลังข้อมูล (ที่มา W.H Inmon Buiding the Data Warehouse Third Edition)....73
2.27 การจัดเก็บข้อมูลแบบระยะสั้นในฐานข้อมูลประจำวัน และระยะยาว ในคลังข้อมูล (ที่มา W.H Inmon Buiding the Data Warehouse Third Edition)....74
2.28 ลักษณะของการจัดเก็บข้อมูลแบบไม่มีการเปลี่ยนแปลง (ที่มา W.H Inmon Buiding the Data Warehouse Third Edition)....74
2.29 องค์ประกอบของคลังข้อมูล (ที่มา Oracle Database Data Warehousing Guide)....75
2.30 รูปแบบของโครงสร้างแบบดวงดาว (ที่มา https://pythonhosted.org/cubes/backends/sql.html)....77
2.31 รูปแบบของโครงสร้างแบบเกล็ดหิมะ (ที่มา https://pythonhosted.org/cubes/backends/sql.html)....77
2.32 การจัดการปัญหา Slowly Changing Dimension รูปแบบที่ 1 (ที่มา http://datawarehouse4u.info/SCD-Slowly-Changing-Dimensions.html)....79
2.33 การจัดการปัญหา Slowly Changing Dimension รูปแบบที่ 2 (ที่มา http://datawarehouse4u.info/SCD-Slowly-Changing-Dimensions.html)....79

สารบัญรูป(ต่อ)

รูป หน้า

2.34 การจัดการปัญหา Slowly Changing Dimension รูปแบบที่ 3 (ที่มา http://datawarehouse4u.info/SCD-Slowly-Changing-Dimensions.html) 80
2.35 การจัดการปัญหา Slowly Changing Dimension รูปแบบที่ 4 (ที่มา http://datawarehouse4u.info/SCD-Slowly-Changing-Dimensions.html) 80
3.1 ลักษณะการเก็บข้อมูลใน Oracle Flashback Data Archive รูปแบบที่ 1 76
3.2 ลักษณะการเก็บข้อมูลใน Oracle Flashback Data Archive รูปแบบที่ 2 77
3.3 โครงสร้างแบดวงดาวของการขาย....77
3.4 ข้อมูลของ Dimension Table ที่มีการเปลี่ยนแปลง....79
3.5 การใช้ Oracle Flashback Query ในการหยิบเอาข้อมูลใน ตาราง Employee ของทุกๆเดือนมารวมกัน....79
3.6 ผลลัพธ์จากการใช้ Oracle Flashback Query....80
3.7 การสรุปข้อมูลตามช่วงเวลาตามที่กำหนดไปเก็บไว้ยังส่วนหนึ่งของฐานข้อมูล ที่ไม่ใช่เนื้อที่ส่วนเดียวกับ Oracle Flashback Data Archive....81
3.8 การสรุปข้อมูลรวมกันทุก 5 และ 30 นาที ตามลำดับ....82
3.9 การรวมข้อมูลเมื่อเวลาครบ 5 นาที....83
3.10 ข้อมูลของตารางเมื่อเวลาผ่านไป 10 นาที หักล้างกับข้อมูลของ ตารางเมื่อเวลาผ่านไป 5 นาที....84
3.11 การสรุปข้อมูล ณ เวลา 10 นาที....84
3.12 การสรุปข้อมูล ณ เวลา 30 นาที....84
3.13 การสรุปข้อมูลจากตารางที่เก็บการสรุปข้อมูลทุก 5 นาทีไปเป็น 1 ชุดข้อมูล ในตารางที่สรุปข้อมูลทุกๆ 30 นาที....85
3.14 ตารางข้อมูลที่มีทั้งการจัดเก็บข้อมูลและการปรับปรุงข้อมูลเมื่อเวลาผ่านไป....86
3.15 การสรุปข้อมูลของตารางที่มีทั้งการจัดเก็บและการปรับปรุงข้อมูล....87
3.16 การสรุป Fact source ไปเป็น Fact table ในคลังข้อมูล....89
3.17 ข้อมูลที่มีในตาราง Valid time state....90
3.18 ข้อมูลการสอบถามจาก Oracle Flashback Data Archive วันที่ 2 Feb. 1998 ถึงวันที่ 9 Feb. 1998....91

สารบัญรูป(ต่อ)

รูป หน้า

3.19 ข้อมูลการสอบถามจาก Oracle Flashback Data Archive วันที่ 10 Feb. 1998 ถึงวันที่ 15 Feb. 1998 92
3.20 การใช้ Oracle Flashback Query ดึงข้อมูลจากช่วงเวลาต่างๆมาไว้ ณ ตารางเดียวกัน 92
3.21 การตอบคำถามด้วยระดับเวลา 3 วัน 93
3.22 การตอบคำถามด้วยระดับเวลา 1 สัปดาห์ 93
3.23 ผลลัพธ์จากการใช้ Oracle Flashback Query สรุปข้อมูลจากแหล่งข้อมูลที่เป็นตาราง Transaction time state 94
3.24 การสรุปสร้างคลังข้อมูลที่ข้อมูลทั้ง Fact source และ Dimension source เป็นข้อมูลเชิงเวลา ..95
4.1 โครงสร้างแบบดวงดาวสำหรับคลังข้อมูลที่ Dimension table และ Fact table ไม่ได้มาจากข้อมูลเชิงเวลา 98
4.2 เมื่อเลือกระดับของเวลาเป็น Summer สำหรับ Fact table ชื่อ Medallists 98
4.3 ผลลัพธ์ Fact table ชื่อ Medallists ตามระดับของเวลาแบบ Summer 99
4.4 เมื่อเลือกระดับของเวลาเป็น Winter สำหรับ Fact table ชื่อ Medallists 99
4.5 ผลลัพธ์ Fact table ชื่อ Medallist ตามระดับของเวลาแบบ Summer 99
4.6 การเข้าใช้งาน เมื่อผู้ใช้ต้องการทราบผลลัพธ์ในรูปกราฟที่ตอบคำถามทาง คลังข้อมูลตามระดับของเวลาที่เตรียมไว้ 100
4.7 การเข้าดูผลลัพธ์แบบกราฟจากคำถามแนวโน้มการได้รับเหรียญทุกๆ รางวัลของแต่ละทวีป ในปีที่จัดตรงกับ Summer เป็นอย่างไร 100
4.8 ผลลัพธ์แบบกราฟจากคำถามแนวโน้มการได้รับเหรียญทุกๆรางวัลของแต่ละทวีป ในปีที่จัดตรงกับ Summer เป็นอย่างไร 101
4.9 การเข้าดูผลลัพธ์แบบกราฟจากคำถามแนวโน้มการได้รับเหรียญทุกๆรางวัลของแต่ละทวีป ในปีที่จัดตรงกับ Winter เป็นอย่างไร 101
4.10 ผลลัพธ์แบบกราฟจากคำถามแนวโน้มการได้รับเหรียญทุกๆรางวัลของแต่ละทวีป ในปีที่จัดตรงกับ Winter เป็นอย่างไร 101
4.11 การเข้าดูผลลัพธ์แบบกราฟจากคำถามแสดงจำนวนเหรียญรางวัล ที่ประเทศไทยได้รับทั้งหมดในแต่ละปีที่ผ่านมา 102

สารบัญรูป(ต่อ)

รูป หน้า

4.12 ผลลัพธ์แบบกราฟจากคำถามแสดงจำนวนเหรียญรางวัล ที่ประเทศไทยได้รับทั้งหมดในแต่ละปีที่ผ่านมา 102
4.13 การเข้าดูผลลัพธ์แบบกราฟจากคำถามแสดงจำนวนเหรียญรางวัล ที่สหรัฐอเมริกาได้รับทั้งหมดในแต่ละปีที่ผ่านมา 102
4.14 ผลลัพธ์แบบกราฟจากคำถามแสดงจำนวนเหรียญรางวัล ที่สหรัฐอเมริกาได้รับทั้งหมดในแต่ละปีที่ผ่านมา 103
4.15 การเข้าดูผลลัพธ์แบบกราฟจากคำถามแสดงจำนวนเหรียญรางวัล ที่ญี่ปุ่นได้รับทั้งหมดในแต่ละปีที่ผ่านมา 103
4.16 ผลลัพธ์แบบกราฟจากคำถามแสดงจำนวนเหรียญรางวัล ที่ญี่ปุ่นได้รับทั้งหมดในแต่ละปีที่ผ่านมา 103
4.17 โครงสร้างแบบดวงดาวสำหรับคลังข้อมูล ที่ Fact table มาจากข้อมูลที่ไม่ชิ่งเวลาและ Dimension table มาจากข้อมูลเชิงเวลา 105
4.18 โครงสร้างแบบดวงดาวสำหรับคลังข้อมูลที่ Dimension table มาจากข้อมูลที่ไม่ใช่เงเวลาและ Fact table มาจากข้อมูลเชิงเวลา 107
4.19 เมื่อเลือกระดับของเวลาให้สรุปการเปลี่ยนแปลงทุกๆ 10 นาที ตั้งแต่ 12:00 น. ถึง 22:00 น. สำหรับ Fact table ชื่อ LotLoc 108
4.20 ผลลัพธ์ Fact table ชื่อ LotLoc ตามระดับของเวลา ให้สรุปการเปลี่ยนแปลงทุกๆ 10 นาที ตั้งแต่ 12:00 น. ถึง 22:00 น 108
4.21 เมื่อเลือกระดับของเวลาให้สรุปการเปลี่ยนแปลงทุกๆ 30 นาที ตั้งแต่ 12:00 น. ถึง 22:00 น. สำหรับ Fact table ชื่อ LotLoc 108
4.22 ผลลัพธ์ Fact table ชื่อ LotLoc ตามระดับของเวลา ให้สรุปการเปลี่ยนแปลงทุกๆ 30 นาที ตั้งแต่ 12:00 น. ถึง 22:00 น 109
4.23 เมื่อเลือกระดับของเวลาให้สรุปการเปลี่ยนแปลงทุกๆชั่วโมง ตั้งแต่ 12:00 น. ถึง 22:00 น. สำหรับ Fact table ชื่อ LotLoc 109
4.24 ผลลัพธ์ Fact table ชื่อ LotLoc ตามระดับเวลาให้ สรุปการเปลี่ยนแปลงทุกๆชั่วโมง ตั้งแต่ 12:00 น. ถึง 22:00 น 109

สารบัญรูป(ต่อ)

รูป **หน้า**

4.25 โครงสร้างแบบดวงดาวสำหรับคลังข้อมูลที่ Dimension table และ Fact table มาจากข้อมูลเชิงเวลา 111

บทที่ 1
บทนำ

1.1 ความสำคัญและที่มาของโครงงาน

คลังข้อมูลในปัจจุบันนั้นได้เข้ามามีบทบาทสำคัญในการดำเนินงานทางธุรกิจเป็นอย่างมาก เนื่องจากสามารถใช้ในการทำกลยุทธ์ทางธุรกิจเพื่อวิเคราะห์ข้อมูลเชิงธุรกิจ อีกทั้งยังเป็นการเตรียมข้อมูลเพื่อที่จะนำไปทำเหมืองข้อมูล (Data Mining) ต่อยอดได้อีกด้วย แต่ความยุ่งยากในการทำคลังข้อมูลนั้นคือการที่ผู้สร้างจะต้องดึงข้อมูลที่จะนำไปสร้างคลังข้อมูลจากฐานข้อมูลระบบงานประจำวันไปจัดเก็บไว้ในไฟล์ ซึ่งจะต้องกำหนดระยะเวลาและคาบเวลา (Period) ที่จะจัดเก็บข้อมูลเหล่านั้นอย่างชัดเจน ซึ่งข้อเสียของการทำคลังข้อมูลทั่วไปคือ เราไม่สามารถที่จะดึงข้อมูลโดยมีคาบเวลาในการดึงข้อมูลที่สั้นได้ กล่าวคือคาบเวลาในการดึงข้อมูลอาจจะถูกจำกัดจากผู้ใช้ว่าจะให้คลังข้อมูลนั้นมีความสามารถในการดึงข้อมูลในช่วงเวลาที่ถี่ได้มากน้อยแค่ไหน อีกทั้งการรวมข้อมูลเหล่านั้นมาสร้างเป็นคลังข้อมูล ยังมีความยากลำบากเป็นอย่างมาก เนื่องจากเมื่อเวลาผ่านไปรูปแบบในการจัดเก็บข้อมูลอาจจะมีการเปลี่ยนแปลง ผู้สร้างคลังข้อมูลจึงจำเป็นต้องทำการคัดกรองข้อมูลและกำหนดให้ข้อมูลที่โหลด (Load) มาจากทุกช่วงเวลาอยู่ในรูปแบบเดียวกันภายในคลังข้อมูล

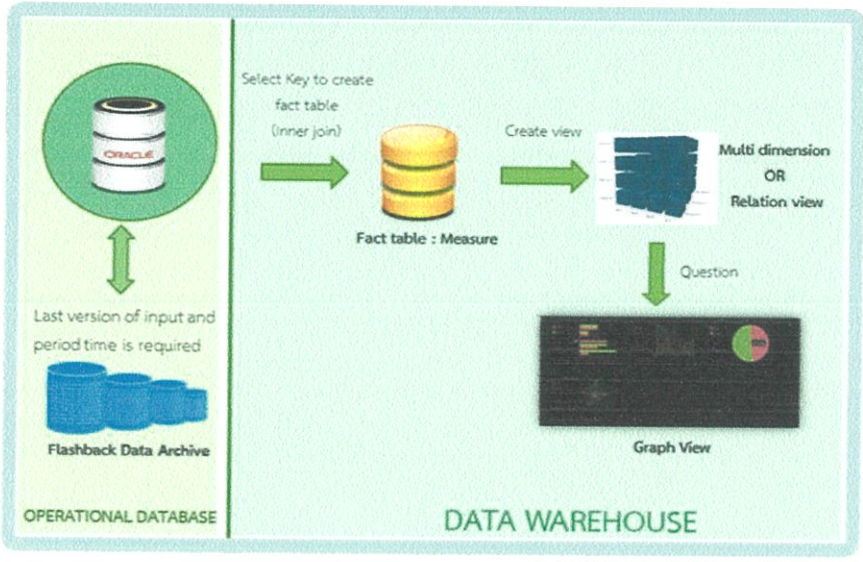
โครงงานนี้จึงจัดทำขึ้นเพื่อศึกษาทดลอง และนำความรู้ที่ได้จากการค้นคว้าเกี่ยวกับเทคโนโลยีที่มีอยู่ใน Oracle Database นั่นคือ Oracle Flashback Data Archive ที่มีความสามารถพิเศษในการจัดเก็บเวอร์ชันของตารางในช่วงเวลาต่างๆได้ มาประยุกต์ใช้ในการสร้างคลังข้อมูลที่ขจัดปัญหาในการดึงข้อมูลไปเก็บไว้ในไฟล์ได้ เนื่องจากเราสามารถหยิบเอาเวอร์ชันเหล่านั้นมาสรุปเป็นคลังข้อมูลได้เลย อีกทั้งคาบเวลาในการทำเวอร์ชันของข้อมูลด้วย Oracle Flashback Data Archive นั้นยังมีความละเอียดถึงระดับหลักวินาที ทำให้การสร้างคลังข้อมูลนั้นเป็นคลังข้อมูลที่มีความละเอียดของข้อมูลเป็นอย่างมาก ซึ่งลักษณะเช่นนี้จะไม่สามารถทำได้เลยในคลังข้อมูลรูปแบบทั่วไปและที่สำคัญของคลังข้อมูลที่พัฒนาขึ้นจาก Oracle Flashback Data Archive ยังสามารถนำข้อมูลเชิงเวลามาใช้งานในรูปแบบของคลังข้อมูลได้อีกด้วย

ทั้งนี้นอกจากจะใช้ความสามารถในการเรียกใช้เวอร์ชันของฐานข้อมูลใน Oracle Flashback Data Archive มาแก้ปัญหาในกระบวนการดึงข้อมูลในการทำคลังข้อมูลรูปแบบปกติแล้ว ยังสามารถช่วยแก้ไขความยุ่งยากในการรวมข้อมูลจากข้อมูลที่ได้ทำการดึงข้อมูลไปเก็บไว้ในรูปแบบไฟล์ได้อีกด้วย เนื่องจากเดิมที่เราต้องทำการโหลดข้อมูลเหล่านั้นจากไฟล์ต่างๆที่เราได้ทำการดึงเก็บไว้มารวมกันพร้อมทั้งต้องจัดรูปแบบของข้อมูลให้อยู่ในรูปแบบที่ตรงกัน ซึ่งยิ่งข้อมูลมี

ระยะเวลาในการเก็บไว้นานมากเท่าไร ยิ่งเพิ่มความยุ่งยากในการรวมข้อมูลมากเท่านั้น เหตุเพราะเมื่อเวลาผ่านไป ข้อมูลต่างๆอาจจะไม่ได้อยู่ในรูปแบบเดิม แต่สำหรับ Oracle Flashback Data Archive แล้วจะไม่เกิดปัญหาเหล่านี้เนื่องจากข้อมูลทุกๆ เวอร์ชันล้วนอยู่ในฐานข้อมูลเดียวกัน ดังนั้นหากข้อมูลชุดปัจจุบันที่เก็บอยู่ในฐานข้อมูลระบบงานประจำวันเปลี่ยนแปลงรูปแบบข้อมูลไป ก็จะส่งผลต่อทุกเวอร์ชันใน Oracle Flashback Data Archive โดยอัตโนมัติ เราจึงสามารถลดความยุ่งยากในการรวบรวมข้อมูลตรงนี้ได้

1.2 วัตถุประสงค์ของโครงงาน

1) เพื่อศึกษาขีดความสามารถของ Oracle Flashback Data Archive
2) เพื่อเพิ่มขีดความสามารถให้แก่การทำคลังข้อมูลรูปแบบทั่วไป ในแง่ของการใช้ข้อมูลเชิงเวลามาประยุกต์ใช้กับคลังข้อมูล
3) เพื่อลดความซับซ้อนในกระบวนการ ETL นั่นคือไม่จำเป็นต้องตรวจสอบความถูกต้องของรูปแบบข้อมูล (Data Format Cleansing) ในการปรับเปลี่ยนข้อมูลให้มีรูปแบบเดียวกันก่อนนำมารวมกันในการสร้าง คลังข้อมูล
4) เพื่อให้ผู้ใช้งานสามารถวิเคราะห์ข้อมูลโดยสรุปเป็นระดับเวลาต่างๆ (Granularity) ตามที่ผู้ใช้ต้องการได้โดยใช้มุมมองของเวลา (Time dimension) ได้เสมือนกับคลังข้อมูลทั่วไป
5) เพื่อให้สามารถนำข้อมูลในอดีต ปัจจุบัน อนาคต (ข้อมูลที่เป็นการวางแผนล่วงหน้า) มาประกอบการวิเคราะห์ตามเรื่องที่องค์กรสนใจ โดยไม่ได้มีเพียงข้อมูลที่เป็นค่าปัจจุบันเพียงอย่างเดียว
6) เพื่อให้ผู้ใช้สามารถสร้างคลังข้อมูลที่มีแหล่งข้อมูลต่างกันคือข้อมูลเชิงเวลาและไม่ใช่ข้อมูลเชิงเวลา โดยที่ไม่จำเป็นต้องไปยุ่งยากกับการรวมข้อมูลจากไฟล์หลายๆเวอร์ชันที่เคยดึงข้อมูลเอาไว้ อีกทั้งยังสามารถตอบคำถามที่ช่วยในการตัดสินทางธุรกิจ และแสดงผลในรูปแบบกราฟได้ตามที่ต้องการ



รูป 1.1 โครงสร้างของคลังข้อมูลที่ใช้ Flashback Data Archive

1.3 ขอบเขตของโครงงาน

โครงงานเรื่องการประยุกต์ภาษาเอสคิวแอลกับข้อมูลเชิงเวลาโดยใช้ Oracle Flashback Data Archive นั้น ในขั้นแรกจะศึกษาขีดความสามารถของ Oracle Flashback Data Archive ในการที่จะทำเวอร์ชันของข้อมูลในช่วงเวลาต่างๆว่ามีรายละเอียดการใช้งานและข้อจำกัดเป็นอย่างไร จากนั้นนำความรู้ที่ได้จากการศึกษามาพัฒนาคลังข้อมูลที่มีแหล่งข้อมูลและคลังข้อมูลอยู่บนฐานข้อมูลระบบงานประจำวันเดียวกัน โดยจะไม่มีการโหลดแหล่งข้อมูลมาจากแหล่งเก็บข้อมูลอื่นเลย ซึ่งแหล่งข้อมูล ดังกล่าวนี้จะแบ่งย่อยได้ 4 ลักษณะคือ

1) แหล่งข้อมูลของคลังข้อมูลที่ทั้ง Dimension source และ Fact Source เป็นแหล่งข้อมูลปก
2) แหล่งข้อมูลของคลังข้อมูลที่มี Fact Source เป็นแหล่งข้อมูลเชิงเวลาแต่ Dimension source เป็นแหล่งข้อมูลปกติ
3) แหล่งข้อมูลของคลังข้อมูลที่มี Fact Source เป็นแหล่งข้อมูลปกติแหล่งข้อมูล แต่ Dimension source เป็นแหล่งข้อมูลเชิงเวลา
4) แหล่งข้อมูลของคลังข้อมูลที่ทั้ง Dimension source และ Fact Source เป็นแหล่งข้อมูลเชิงเวลา

จากนั้นพัฒนาโปรแกรมประยุกต์ที่นำแหล่งข้อมูลทั้ง 4 แบบนี้มาสรุปเป็นคลังข้อมูลที่ประกอบไปด้วย Fact Table และ Dimension Table ตามหลักเกณฑ์ในการสร้างคลังข้อมูลทั่วไปพร้อมทั้งสามารถตอบคำถามเชิงธุรกิจได้และนำเสนอคำตอบนั้นในรูปแบบของกราฟ

1.4 วิธีการดำเนินงาน

1) ศึกษาทฤษฎีเรื่องฐานข้อมูลเชิงเวลา
2) ศึกษาการใช้งาน Oracle Flashback Data Archive
3) ศึกษาทฤษฎีการออกแบบและการใช้งานคลังข้อมูล
4) ศึกษาคิดค้นอัลกอริทึมในการสรุปแหล่งข้อมูลที่นำมาสร้างเป็นคลังข้อมูล
5) วิเคราะห์และออกแบบโปรแกรมประยุกต์ที่จะใช้สร้างคลังข้อมูล
6) พัฒนาโปรแกรมประยุกต์ในการใช้ข้อมูลเชิงเวลาในการสร้างคลังข้อมูล
7) เขียนการทบทวนวรรณกรรม (Literature Review) ควบคู่ไปกับการทำโครงงานเพื่อที่จะได้ทราบว่ามีผู้ที่คิดโจทย์ในการแก้ปัญหาเช่นเดียวกับเราหรือไม่ และมีวิธีแก้ไขปัญหานั้นอย่างไร

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1) ได้รับความรู้ ความเข้าใจเกี่ยวกับฐานข้อมูลเชิงเวลา
2) ได้รับความรู้ ความเข้าใจเกี่ยวกับขีดความสามารถของ Oracle Flashback Data Archive
3) ได้รับความรู้ ความเข้าใจเกี่ยวกับทฤษฎีคลังข้อมูล
4) สามารถนำความรู้ที่ได้ไปออกแบบ และพัฒนาคลังข้อมูลที่มีขีดความสามารถมากกว่าคลังข้อมูลแบบทั่วไปได้
5) สามารถนำโปรแกรมประยุกต์ที่พัฒนาขึ้นไปใช้งานที่เกี่ยวข้องกับเพิ่มศักยภาพการตัดสินใจทางธุรกิจได้

บทที่ 2
แนวคิดและทฤษฎีที่เกี่ยวข้อง

2.1 การทบทวนวรรณกรรม (Literature Review)

ในด้านของคลังข้อมูลโดยทั่วไปที่พบเห็นและมีการศึกษากันนั้น มักจะเป็นคลังข้อมูลที่มีการนำฐานข้อมูลแบบปกติหลากหลายฐานข้อมูล ซึ่งอาจจะมีลักษณะแตกต่างกันทั้งในเรื่องของรูปแบบการจำลองข้อมูล, ชื่อแอตทริบิวท์, ชนิดของข้อมูล เป็นต้น นอกจากนี้ฐานข้อมูลต่างๆเหล่านั้น อาจกระจายอยู่คนละส่วนภายในองค์กร ไม่ได้มาจากฐานข้อมูลเดียวกัน จึงต้องใช้ระยะเวลาในรวบรวม (Data integration) และทำกระบวนการ ETL เพื่อดึงข้อมูลจากฐานข้อมูลเหล่านั้นเข้าสู่คลังข้อมูล เมื่อต้องการจะสรุปข้อมูลในคลังข้อมูลให้อยู่ในช่วงเวลาต่างๆ (Time granularity) ตามที่ผู้ใช้ต้องการทราบ ก็จะต้องทำการดึงข้อมูลมาลงไฟล์เป็นประจำ ยิ่งถ้าต้องการเห็นการเปลี่ยนแปลงของข้อมูลที่อาจจะเกิดขึ้นภายในคลังข้อมูล โดยยิ่งมีการเปลี่ยนแปลงถี่เท่าไหร่ ผู้ใช้ก็จะยิ่งต้องดึงข้อมูลเพื่อจะนำมาสรุปผลบ่อยเท่านั้น โดยส่วนใหญ่มีการศึกษากันแค่เพียงในคลังข้อมูลที่ภายในไม่มีการเปลี่ยนแปลงของข้อมูลหรือมีลักษณะเป็นคลังข้อมูลแบบอ่านอย่างเดียว (Read only) เท่านั้น แต่ที่มีการศึกษากันน้อยคือเมื่อฐานข้อมูลที่จะถูกดึงนำมาใช้ในคลังข้อมูลมีการเปลี่ยนแปลงของข้อมูลตามช่วงเวลาและฐานข้อมูลเหล่านั้นต่างอยู่ในฐานข้อมูลเดียวกัน เมื่อต้องการจะสรุปข้อมูลตามช่วงเวลาต่างๆ จะต้องมีการจัดการอย่างไร

แต่ประเด็นเรื่องการทำคลังข้อมูลเชิงเวลานั้นมีหลักฐานจากการศึกษาจำนวนมากเริ่มทำการศึกษากันมากขึ้นและชี้ชัดไปในทางเดียวกันว่าภายในคลังข้อมูลควรจะมีประเภทของเวลาต่างๆเข้ามาเกี่ยวข้องอยู่ภายในฐานข้อมูลเชิงเวลาและคลังข้อมูลมีความสัมพันธ์เชิงเวลาในทางความหมายอย่างไรบ้าง แต่ยังไม่มีหลักฐานบอกได้ว่าเราควรจะจัดการอย่างไร ถ้าฐานข้อมูลที่เป็นแหล่งข้อมูลมีลักษณะเป็นฐานข้อมูลเชิงเวลา และจำเป็นจะต้องดึงข้อมูลจากฐานข้อมูลนั้นมาใช้เป็นส่วนหนึ่งของโครงสร้างข้อมูลแบบดวงดาว (Star Schema) ที่จะนำไปใช้วิเคราะห์ข้อมูล สรุปเป็นข้อมูลตามช่วงเลาต่างๆ เพราะยังไม่ได้มีการศึกษาลงลึกไปถึงการจัดสร้างระบบสำหรับการคลังข้อมูลที่ดึงข้อมูลมาจากฐานข้อมูลเชิงเวลา และทำอย่างไรถึงจะเห็นข้อมูลที่เปลี่ยนแปลงตามช่วงเวลาต่างๆ ในรูปแบบระดับ นาที ชั่วโมง วัน สัปดาห์ เดือน ปี เป็นต้น

ในเมื่อยังไม่มีการศึกษาที่แสดงหลักฐานว่าควรจะมีการจัดการภายในคลังข้อมูลอย่างไร ถ้าข้อมูลที่จะนำมาใช้ต้องดึงจากหลายฐานข้อมูลที่อยู่ในฐานข้อมูลเดียวกันและฐานข้อมูลที่ต้องการนำข้อมูลภายในมาใช้เหล่านั้น บางฐานข้อมูลเป็นฐานข้อมูลเชิงเวลา บางฐานข้อมูลเป็นฐานข้อมูล

แบบปกติ เราจะดึงฐานข้อมูลเหล่านั้นมาสร้างเป็นโครงสร้างแบบดวงดาวที่มี Fact table และ Dimension table หน้าตาเป็นอย่างไร เมื่อถึงขั้นตอนการสรุปข้อมูล จะต้องสรุปข้อมูลอย่างไรให้สอดคล้องกับการเปลี่ยนแปลงตามรูปแบบระยะเวลาที่ต้องการทราบหรือใช้เปรียบเทียบผลการศึกษานี้จึงสนใจที่จะหาคำตอบดังกล่าว โดยคำตอบนี้จะช่วยสนับสนุนองค์กรที่สามารถนำคลังข้อมูลรูปแบบนี้ไปใช้ต่อยอดทางธุรกิจ เช่น การศึกษาแนวโน้มเชิงการทำเหมืองข้อมูลเพราะในความจริงแล้ว ภายในคลังข้อมูลอาจมีข้อมูลต่างๆที่สามารถเปลี่ยนแปลงตามช่วงเวลาได้ ไม่ได้เป็นคลังข้อมูลแบบอ่านอย่างเดียวเสมอไป

จากบทความมากมายที่ทำการศึกษาเกี่ยวกับคลังข้อมูลและฐานข้อมูเชิงเวลาดังกล่าวก็ยังไม่คลอบคลุมในเนื้อหาที่ต้องการจะศึกษา เช่น ใน [3] ได้กล่าวว่าเนื้อหาของสาขาคลังข้อมูลและฐานข้อมูลเชิงเวลาต่างก็มีความเกี่ยวข้องกับเวลาทั้งคู่ โดยทั้งสองสาขาต่างมีความหมายที่คล้ายคลึงกันมาก ความคล้ายคลึงกันนี้ทำให้เป็นไปได้ที่จะกำหนดนิยามของคลังข้อมูลเสียใหม่ ให้เป็นความหมายในแง่ของฐานข้อมูล Bitemporal เสียก็ยังได้ บทความนี้จะเน้นไปที่การนำเสนอโครงสร้างการจัดเก็บสำหรับคลังข้อมูลแบบ bitemporal (Bitemporal Data Warehouse) ที่มีทั้ง Valid time (VT) และ Transaction time (TT) วิธีการที่ผู้เขียนเสนอคือ ควรจะมี Transaction time ประกอบเป็นส่วนหลักของโครงสร้าง โดยแหล่งข้อมูลที่จะโหลดเข้ามาสู่คลังข้อมูลเป็น delta files ซึ่งมีแต่ transaction time อยู่ภายใน ซึ่งได้กล่าวว่าเป็นแหล่งข้อมูลที่มักพบเจอบ่อย และนำมาวิเคราะห์ว่าเวลา TT ของแหล่งข้อมูล เมื่อโหลดเข้าสู่คลังข้อมูลแล้ว เวลา VT และ TT ของคลังข้อมูลจะเป็นอย่างไร และรูปแบบในการจัดเก็บข้อมูลใช้แบบจำลองข้อมูลเชิงวัตถุ (Object-oriented Data Model) โดยกล่าวว่ามีความเหมาะสมและยืดหยุ่นกว่าแบบจำลองเชิงมิติ Mulitidimensional Model เพราะช่วยเพิ่มคุณลักษณะการรวบรวมข้อมูล (Integration) และกำหนดหัวข้อที่สนใจ (Subject-orientation) ภายในคลังข้อมูล ส่วนแบบจำลองเชิงมิติเหมาะกับคลังข้อมูลขนาดเล็ก และ Star schema เป็นรูปแบบที่มีความเข้มงวดมากเกินไป มีการ query ที่ซับซ้อน บทความนี้จึงไม่ได้กล่าวถึงการออกแบบ Fact table และ Dimension table เมื่อแหล่งข้อมูลสัมพันธ์กับเวลาเลย ซึ่งเป็นส่วนที่พวกเราต้องการจะศึกษา ใน [4] บทความนี้เป็นการศึกษาเชิงเวลาของแหล่งข้อมูลที่แตกต่างกันและโหลดเข้าสู่คลังข้อมูล มีการวิเคราะห์ความสอดคล้องกันระหว่างแอตทริบิวต์เชิงเวลาในแหล่งข้อมูลและแอตทริบิวส์เชิงเวลาในคลังข้อมูล โดยกล่าวว่าคลังข้อมูลมีโครงสร้างภายในเป็นฐานข้อมูล Bitemporal ทำให้สามารถติดตามการเปลี่ยนแปลงในอดีตที่ผ่านมาได้ จึงมีการระบุเกี่ยวกับมุมมองเชิงเวลา ได้แก่ Valid time dimension และ Transaction time dimension เป็นสิ่งที่ควรจะมีในคลังข้อมูลแบบ Bitemporal การใช้โครงสร้างการจัดเก็บเป็น bitemporal เพื่อแทนข้อมูลเชิงเวลาจากแหล่งข้อมูลที่ได้รับมาจากแหล่งข้อมูลชนิดต่างๆ ที่แตกต่างกัน โดยได้ทำการการวิเคราะห์แหล่งข้อมูลที่แตกต่างกันเหล่านี้ว่าในแหล่งข้อมูลมีค่า VT และ TT

เป็นอย่างไร เมื่อโหลดข้อมูลจากแหล่งข้อมูลประเภทต่างๆเหล่านั้นแล้ว จะได้ค่า VT และ TT ภายในคลังข้อมูลเป็นอย่างไร ซึ่งไม่ได้กล่าวถึงการออกแบบเพื่อสร้างแบบจำลองเชิงมิติในการวิเคราะห์ข้อมูลเลย อีกทั้งไม่ได้กล่าวถึงชนิดของแหล่งข้อมูลที่เป็นฐานข้อมูลเชิงเวลาตาม[1] ที่ซึ่งอาจปะปนกับฐานข้อมูลแบบปกติในฐานข้อมูลระบบงานประจำวันอย่างที่ข้าพเจ้าต้องการศึกษา ใน [5] ผู้เขียนได้กล่าวว่าฐานข้อมูลเชิงเวลาและคลังข้อมูลมีแนวคิดที่มีความคล้ายคลึงกัน ตัวอย่างเช่น แนวคิดของ Time-variance ในคลังข้อมูลคล้ายกับ valid time ในฐานข้อมูลเชิงเวลาโดย Time-variance หมายถึง ทุกๆ แถวในคลังข้อมูลจะต้องมีความเกี่ยวเนื่องและเกิดขึ้นจริงในเวลาหนึ่งๆ ส่วน Valid time ก็เป็นเวลาเมื่อ Fact เป็นจริง ดังนั้นจะเห็นได้ว่าทั้งสองคำสำคัญเกี่ยวข้องซึ่งกันและกันในแง่ของเวลาที่ขึ้นกับความเป็นจริง นอกจากนั้นยังมีคำศัพท์ที่สำคัญ คือคำว่า Non-volatility หมายถึงการที่ Fact มีการเปลี่ยนแปลงได้ในคลังข้อมูลซึ่งจะมีการจัดเก็บรูปแบบเป็นเวอร์ชันของข้อมูลต่างๆอ้างอิงตามช่วงเวลาที่เปลี่ยนแปลง (Time-variant snapshot) เมื่อมีการปรับปรุงข้อมูล ก็จะมีเวอร์ชันใหม่ถูกเพิ่มเข้ามาในคลังข้อมูลโดยจะมีความคล้ายคลึงกับ Transaction time ซึ่งก็คือเวลาที่ระบุว่า Fact เหล่านั้นมีอยู่จริงในฐานข้อมูล เช่น โดยได้แนะนำว่าควรจะมีประเภทของเวลาได้แก่ Valid time และ Transaction time เป็นส่วนประกอบอยู่ภายในคลังข้อมูล เช่นดังสองบทความข้างต้น แต่บทความนีเน้นไปที่การอธิบายสิ่งที่เกี่ยวกับเวลาภายในคลังข้อมูล เช่น ภาษาฐานข้อมูลเชิงเวลาใช้ในคลังข้อมูลอย่างไร โดยภายในคลังข้อมูลเป็น Bitemporal และยืนยันว่าควรใช้แนวติดแบบจำลองเชิงวัตถุ ซึ่งจาก 3 บทความข้างต้นที่เขียนจากผู้เขียนคนเดียวกัน ข้าพเจ้ามีความคิดว่ายังมีการอธิบายเหตุผลในข้อความที่ว่าการใช้แบบจำลองเชิงวัตถุเหมาะสมมากกว่าการใช้แบบจำลองเชิงมิติไม่ชัดเจนเพียงพอ

การศึกษาของโครงงานชิ้นนี้ก็เป็นการอธิบายฐานข้อมูลเชิงเวลาที่สัมพันธ์กับคลังข้อมูลเช่นเดียวกัน แต่แหล่งข้อมูลที่ใช้เพื่อโหลดข้อมูลเข้าสู่คลังข้อมูล ล้วนแต่อยู่บนฐานข้อมูลเดียวกันแตกต่างกันที่แหล่งข้อมูลเป็นแหล่งข้อมูลที่เก็บข้อมูลเชิงเวลาบ้าง หรืออาจจะเป็นแหล่งข้อมูลที่จัดเก็บข้อมูลที่ไม่ใช่เชิงเวลาบ้าง ซึ่งมีวิธีการจัดการมากมายในการที่จะสรุปข้อมูลที่มีการเปลี่ยนแปลงต่างๆ โดยนำแหล่งข้อมูลโหลดเข้าคลังข้อมูลและออกแบบเป็นโครงสร้างแบบดวงดาวเพื่อนำข้อมูลนั้นมาวิเคราะห์และสรุปเป็นระดับเวลาตามที่ผู้ใช้ต้องการ ซึ่งทำให้ผู้ใช้สะดวกมากขึ้นไม่ต้องยุ่งยากกับการเตรียมข้อมูลก่อนโหลดเข้าคลังข้อมูล อีกทั้งยังสามารถติดตามข้อมูลที่เคยเปลี่ยนแปลงไปซึ่งอาจจะมีความสำคัญมาก อาจเป็นสิ่งที่ละเลยไม่ได้ในการทำงานด้านธุรกิจ เป็นต้น

2.2 ฐานข้อมูลเชิงเวลา (Temporal Database)

ฐานข้อมูลเชิงเวลาคือ ฐานข้อมูลที่มีความสามารถในการจัดเก็บข้อมูลที่เปลี่ยนแปลงไปตามเวลา เช่น ชื่อ ที่อยู่ สถานที่ทำงาน เป็นต้น ซึ่งในเวลาต่อมาได้มีการถกเถียงกันว่า ข้อมูลบางอย่างไม่ได้เป็นข้อมูลที่เปลี่ยนแปลงไปตามเวลาแต่เปลี่ยนแปลงไปตามความสามารถหรือเทคโนโลยีในการประเมินค่าของมนุษย์ เช่น ความสูงของยอดเขา ความสว่างของดวงดาวเป็นต้น ก็สมควรที่จะจัดเก็บด้วยฐานข้อมูลเชิงเวลาได้ ดังนั้นความหมายที่อธิบายคำว่าระบบฐานข้อมูลเชิงเวลาได้ดีที่สุดคือ ฐานข้อมูลที่นำเสนอแง่มุมทางเวลา โดยไม่ถือรวมเวลาที่ประกอบเป็นความจริงส่วนหนึ่งของข้อมูล

ในยุคเริ่มแรกได้มีการถกเถียงกันเกี่ยวกับแอตทริบิวต์เชิงเวลา (Timing attribute) ว่าสมควรจะมองเสมือนแอตทริบิวต์ธรรมดาหรือควรมองแยกออกมาเป็นอีกมิติและให้ภาระการจัดการข้อมูลเชิงเวลาตกเป็นของระบบจัดการฐานข้อมูล (Database Management System :DBMS) ทั้งหมด ซึ่งแนวคิดอันหลังนี้ค่อนข้างจะเป็นที่นิยมกว่าเนื่องจากการบริหารจัดการข้อมูลเกี่ยวกับช่วงเวลานั้นมีความยุ่งยากเป็นอย่างมาก ซึ่งจะนำเสนอให้เห็นภาพดังต่อไปนี้

ในประวัติศาสตร์ประมาณปี 1996 ประเทศอเมริกาได้พบว่าเนื้อวัวที่ส่งขายจำนวนหนึ่งได้ติดเชื้อโรคและเชื้อโรคได้ระบาดในเนื้อวัวเหล่านั้น ซึ่งผู้จำหน่ายเนื้อวัวเหล่านั้นไม่สามารถทราบได้เลยว่าเนื้อจำนวนนั้น มาจากโรงฆ่าสัตว์ไหน และมีวัวตัวไหนที่อยู่คอกเดียวกันในเวลาเดียวกันบ้าง โดยตั้งสมมติฐานที่ว่า วัวที่อยู่คอกเดียวกัน น่าจะเป็นวัวที่ติดเชื้อชนิดเดียวกัน ซึ่งผู้จำหน่ายเนื้อวัวเหล่านี้ไม่สามารถตอบคำถามเหล่านี้ได้เลย เป็นผลให้ต้องเรียกคืนเนื้อวัวทั้งหมดที่จำหน่ายโดยบริษัทนี้ในประเทศอเมริกาทำให้บริษัทนี้ สูญเสียรายได้เป็นอย่างมาก จึงเกิดแนวคิดขึ้นว่าหากเราสามารถสอบถามข้อมูลย้อนกลับไปได้ว่าเนื้อแต่ละชิ้นมาจากวัวตัวไหน คอกไหน หรือแม้แต่กระทั้งวัวตัวใดเคยอยู่คอกเดียวกันกับวัวตัวที่ติดเชื้อเหล่านี้บ้าง จะทำให้ลดการสูญเสียรายได้มหาศาลของบริษัทนี้ลงได้

จากเหตุการณ์ดังกล่าวเราสามารถนำฐานข้อมูลเชิงเวลาเข้ามาแก้ได้ปัญหาที่เกิดจากเหตุการณ์นี้ได้ดังจะกล่าวถึงต่อไป แต่เริ่มแรกต้องเข้าใจรูปแบบของฐานข้อมูลเชิงเวลาก่อน ดังนี้

2.2.1 ประเภทและความหมายของเวลา

ในการจัดการกับฐานข้อมูลเชิงเวลานั้น จะต้องมีการระบุเวลาในฐานข้อมูล ซึ่งประเภทของเวลาที่ระบุในฐานข้อมูลมี 3 ประเภท ดังนี้

1) Valid Time คือ เวลาที่ Fact เป็นจริง หรือเวลาที่ข้อมูลนั้นเป็นจริงในฐานข้อมูล ซึ่งต้องกำหนดว่าเป็นจริงตั้งแต่เมื่อไหร่ถึงเมื่อไหร่
2) Transaction Time คือ เวลาที่ข้อมูลถูกบันทึกลงในฐานข้อมูล ซึ่งจัดการโดยระบบจัดการฐานข้อมูลในการบันทึกข้อมูล
3) User-defined Time คือ เวลาที่เป็นส่วนหนึ่งของ Fact ถึงแม้จะเป็นข้อมูลประเภทเวลา แต่ระบบจะมองเป็นแค่ข้อมูลธรรมดา เช่น วันเกิด วันจบการศึกษา เป็นต้น

2.2.2 ความซ้ำซ้อนของข้อมูลเชิงเวลา

ฐานข้อมูลเชิงเวลาที่ดีนั้น ไม่ควรจะมีความซ้ำซ้อนของข้อมูลเกิดขึ้น เพราะจะส่งผลให้เกิดปัญหาในการจัดเก็บข้อมูล (Insert) การปรับปรุงข้อมูล (Update) และการลบข้อมูล (Delete) เพื่อแสดงให้เห็นความซ้ำซ้อนของข้อมูลในแบบต่างๆ จะนำเสนอโดยใช้ตัวอย่างตารางที่แสดงสถานะของคนไข้ ดังนี้

ตาราง 2.1 สถานะของคนไข้ในโรงพยาบาลแห่งหนึ่ง [ที่มา Managing Temporal Data A Five-Part Series]

Name	Status	From_date	To_date
Kenneth Robert	Serious	1997-11-19	1997-11-21
Alexis May	Serious	1997-11-19	1997-11-27
Natalie Sue	Serious	1997-11-19	1997-11-25
Kelsey Ann	Serious	1997-11-19	1997-11-26
Brandon James	Serious	1997-11-19	1997-11-26
Nathan Roy	Serious	1997-11-19	1997-11-28
Joel Steven	Critical	1997-11-19	1997-11-20
Joel Steven	Serious	1997-11-20	1997-11-26
Kenneth Robert	fair	1997-11-21	1998-01-03
Alexis May	Fair	1997-11-27	1998-01-11
Alexis May	Fair	1997-12-02	9999-12-31
Alexis May	Fair	1997-12-02	9999-12-31

จากตารางข้างต้นเป็นตารางเชิงเวลาที่แสดงสถานะของผู้ป่วยเด็กเล็ก โดยที่ Name แทนชื่อผู้ป่วย, Status แทนสถานะอาการ, From_date แทนวันที่เริ่มต้นที่เกิดอาการ และ To_date แทนวันที่สิ้นสุดการเป็นอาการดังกล่าว

จะเห็นได้ว่า Kenneth Robert มีสถานะ Serious ตั้งแต่วันที่ 1997-11-19 ถึง 1997-11-21 จริงแล้ว Robert มีสถานะ Serious จริงตั้งแต่วันที่ 1997-11-19 ถึง 1997-11-20 ต่อมาในวันที่ 1997-11-21 Robert มีอาการดีขึ้น กล่าวคือเป็นการใช้รูปแบบแสดงช่วงเวลาแบบ Closed-open การที่ใส่ To_date ของสถานะ Serious เป็นวันเดียวกับ From_date ของสถานะ Fair ก็เพื่อให้เห็นความต่อเนื่องของเวลาและตรวจสอบได้ง่ายนั่นเองโดยตารางตัวอย่างข้างต้นได้แสดงความซ้ำซ้อนของข้อมูลทั้งหมด 4 รูปแบบ คือ

1) Nonsequenced duplicates คือ ความซ้ำซ้อนของข้อมูลที่เหมือนกันทุกประการ กล่าวคือเหมือนกันทุกคอลัมน์ โดยไม่สนใจคอลัมน์เวลา หรือถือคอลัมน์เวลาเป็นคอลัมน์พิเศษ

ตาราง 2.2 ตัวอย่างความซ้ำซ้อนแบบ Nonsequenced duplicates [ที่มา Managing Temporal Data A Five-Part Series]

Name	Status	From_date	To_date
Alexis May	Fair	1997-12-02	9999-12-31
Alexis May	Fair	1997-12-02	9999-12-31

ซึ่งสามารถแก้ไขความซ้ำซ้อนได้โดยกำหนดทั้ง 4 คอลัมน์เป็น Primary key ร่วมกัน ซึ่งจะทำให้คนไข้ไม่สามารถมีหลายสถานะเกิดขึ้นในเวลาเดียวกัน แต่การแก้ไขปัญหาโดยวิธีนี้ยังไม่เพียงพอ

2) Value-equivalent duplicates คือ ความซ้ำซ้อนของข้อมูลที่ทุกคอลัมน์มีค่าเหมือนกัน ยกเว้นคอลัมน์ช่วงเวลา

ตาราง 2.3 ตัวอย่างความซ้ำซ้อนแบบ Value-equivalent duplicates [ที่มา Managing Temporal Data A Five-Part Series]

Name	Status	From_date	To_date
Alexis May	Fair	1997-11-27	1998-01-11
Alexis May	Fair	1997-12-02	9999-12-31
Alexis May	Fair	1997-12-02	9999-12-31

ซึ่งสามารถแก้ไขความซ้ำซ้อนได้โดยกำหนดคอลัมน์ Name และ Status เป็น Primary key ร่วมกัน ทำให้ผู้ป่วยมีได้สถานะเดียวเท่านั้น

3) Current duplicates คือ ความซ้ำซ้อนของข้อมูลที่ทุกคอลัมน์มีค่าเหมือนกัน ในช่วงเวลา ในเวลาปัจจุบัน หรือกล่าวได้ว่ามี Fact ซ้ำกันในเวลาปัจจุบัน สมมติให้วันนี้เป็นวันที่ 10 มกราคม 1998

ตาราง 2.4 ตัวอย่างความซ้ำซ้อนแบบ Current duplicates [ที่มา Managing Temporal Data A Five-Part Series]

Name	Status	From_date	To_date
Alexis May	Fair	1997-11-27	1998-01-11
Alexis May	Fair	1997-12-02	9999-12-31
Alexis May	Fair	1997-12-02	9999-12-31

ซึ่งสามารถแก้ไขความซ้ำซ้อนได้โดยการบันทึกข้อมูลให้ถูกต้องตั้งแต่แรก

4) Sequenced duplicates คือ ความซ้ำซ้อนของข้อมูลที่ทุกคอลัมน์มีค่าเหมือนกัน ยกเว้นคอลัมน์ช่วงเวลา โดยซ้ำซ้อนในช่วงเวลาใดๆ ไม่จำเป็นต้องซ้ำซ้อนในเวลาปัจจุบัน

ตาราง 2.5 ตัวอย่างความซ้ำซ้อนแบบ Sequenced duplicates [ที่มา Managing Temporal Data A Five-Part Series]

Name	Status	From_date	To_date
Alexis May	Fair	1997-11-27	1998-01-11
Alexis May	Fair	1997-12-02	9999-12-31
Alexis May	Fair	1997-12-02	9999-12-31

ซึ่งสามารถแก้ไขความซ้ำซ้อนได้โดยกำหนดเงื่อนไขในการบันทึกข้อมูล เพื่อไม่ให้มีการบันทึกสถานะอาการที่ซ้ำในช่วงเวลาใดๆ

2.2.3 ตาราง Valid Time State

ตาราง Valid Time State คือตารางที่เพิ่ม valid time เข้ามาเพื่อบอกว่าข้อมูลใดที่ยังเป็นจริงอยู่ในฐานข้อมูลเชิงเวลาบ้าง ดังตัวอย่างต่อไปนี้ (ตัวอย่างจำเป็นต้องเพิ่มคอลัมน์เวลามาเพื่อช่วยในการอธิบาย แต่พึงระลึกไว้เสมอว่าสิ่งผู้เขียนพยายามจะนำเสนอคือ เวลาจะถูกซ่อนเป็นอีกมิติให้ระบบจัดการฐานข้อมูลเป็นผู้จัดการ)

ตาราง 2.6 รายละเอียดของฝูงวัว [ที่มา Managing Temporal Data A Five-Part Series]

FDYD_ID	LOT_ID_NUM	PEN_ID	HD_CNT	FROM_DATE	TO_DATE
1	137	1	17	1998-02-07	1998-02-18
1	219	1	43	1998-02-25	1998-03-01
1	219	1	20	1998-03-01	1998-03-14
1	219	2	23	1998-03-01	1998-03-14
1	219	2	43	1998-03-14	9999-12-31
1	374	1	14	1998-02-20	9999-12-31

ตารางนี้นำเสนอรายละเอียดของวัวแต่ละฝูงว่าเคยอยู่คอกใดในช่วงเวลาใดบ้างโดยที่ FDYD_ID แทนลานให้อาหารของวัว, LOT_ID_NUM แทนหมายเลขฝูงของวัว, PEN_ID แทนคอกย่อยของวัว, HD_CNT แทนจำนวนวัวในฝูงนั้น และ FROM_DATE, TO_DATE แทนวันเวลาที่ข้อมูลนั้นยังเป็นจริง หาก TO_DATE เป็น 9999-12-31 ซึ่งหมายความ ข้อมูลนั้นยังเป็นจริงมาจนถึงปัจจุบัน

จากตางรางข้างต้นแสดงตัวอย่างให้เห็นว่า วัวฝูง 219 นั้นเคยอยู่คอกที่ 1 เมื่อวันที่ 1998-02-25 จนถึงวันที่ 1998-02-28 จำนวนทั้งสิ้น 43 ตัว จากนั้นวันที่ 1998-03-01 ได้แยกวัวฝูงนี้กระจายไปเป็น 2 คอก คือคอกที่ 1 จำนวน 20 ตัวและคอกที่ 2 จำนวน 23 ตัวจนถึงวันที่ 1998-03-13 จากนั้นได้จับวัวฝูงนี้มาอยู่รวมที่คอกที่ 2 ด้วยกันอีกครั้งจนถึงปัจจุบัน จะเห็นว่ารายละเอียดการย้ายคอกของวัวฝูงนี้จะไม่สามารถแสดงได้เลยหากเราใช้ฐานข้อมูลแบบปกติ

2.2.3.1 ตัวอย่างคำถามเชิงเวลาสำหรับข้อมูลเชิงเวลา

1) Temporal Selection เป็นการสอบถามข้อมูลเชิงเวลาโดยไม่มีการรวมข้อมูล (Join) ระหว่าง 2 ตาราง (ข้อมูลทุกอย่างปรากฏครบถ้วนในตารางเดียว)

- Current Query (สอบถามคำถามที่ยังเป็นจริงอยู่ในปัจจุบัน) ตัวอย่างคำถามประเภทนี้เช่น "ในแต่ละคอกย่อยของลานให้อาหารที่ 1 มีวัวฝูง 219 อยู่คอกใดและมีจำนวนกี่ตัวบ้าง" จะเห็นได้ว่าจากคำถามดังกล่าวหากเป็นระบบฐานข้อมูลปกติสามารถใช้คำสั่งภาษาฐานข้อมูลเอสคิวแอลธรรมดาได้ง่ายดาย ดังนี้

```
SELECT PEN ID, HD CNT FROM LOT LOC

WHERE FDYD ID = 1 AND LOT ID NUM = 219
```

แต่หากเป็นฐานข้อมูลเชิงเวลาจะต้องระบุช่วงเวลาที่เป็นปัจจุบันไปด้วยดังนี้

```
SELECT PEN ID, HD CNT FROM LOT LOC

WHERE FDYD ID = 1 AND LOT ID NUM = 219 AND TO DATE = DATE '9999-12-31'
```

ซึ่งผลลัพธ์ของการสอบถามข้อมูลทั้งสองแบบจะได้ผลดังนี้

ตาราง 2.7 ผลลัพธ์ของการสอบถามข้อมูลทั้งสองแบบข้างต้น

[ที่มา Managing Temporal Data A Five-Part Series]

PEN_ID	HD_CNT
2	43

- Sequence Query (การสอบถามถึงข้อมูลต่างๆที่เปลี่ยนแปลงไปในอดีต) คำถามประเภทนี้จะเป็นการถามโดยการระบุช่วงเวลาในอดีตจนถึงปัจจุบันเพื่อต้องการทราบประวัติการเปลี่ยนแปลงของข้อมูลเช่น "ในเวลาที่ผ่านมา วัวฝูง 219 เคยอยู่ในคอกย่อยใดบ้างและคอกล่ะกี่ตัว" สามารถเขียนคำสั่งภาษาฐานข้อมูลเอสคิวแอลธรรมดาเพื่อตอบคำถามดังกล่าวได้ดังนี้

```
SELECT PEN ID, HD CNT, FROM DATE, TO DATE FROM LOT LOC

WHERE FDYD ID = 1 AND LOT ID NUM = 219
```

จะเห็นว่าหากเราไม่ระบุเวลาในฐานข้อมูลเชิงเวลาจะมีความหมายเทียบเท่าช่วงเวลาทั้งหมดจากอดีตจนถึงปัจจุบัน ซึ่งจะให้ผลลัพธ์ดังนี้

ตาราง 2.8 ผลลัพธ์จากการทำ Sequence Query

[ที่มา Managing Temporal Data A Five-Part Series]

PEN_ID	HD_CNT	FROM_DATE	TO_DATE
1	43	1998-02-25	1998-03-01
1	20	1998-03-01	1998-03-14
2	23	1998-03-01	1998-03-14
2	43	1998-03-14	9999-12-31

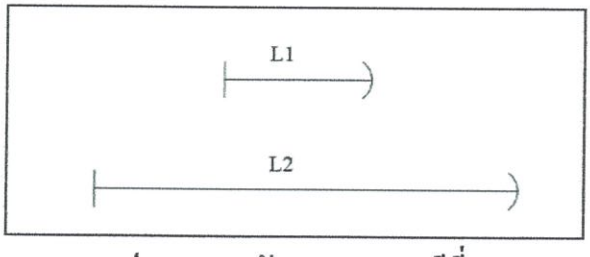
ซึ่งคำถามลักษณะนี้ หากเป็นการเก็บลงฐานข้อมูลปกติ จะไม่มีทางตอบคำถามได้อย่างแน่นอน

2) Temporal Joins ในบางครั้งตารางเพียง 1 ตารางไม่เพียงพอจะให้ข้อมูลในการตอบคำถามทั้งหมด จึงจำเป็นต้องมีการรวมข้อมูลระหว่างตารางเกิดขึ้นซึ่งจะแสดงให้เห็นดังตัวอย่างคำถามต่อไปนี้ "มีวัวฝูงใดบ้างที่อยู่คอกเดียวกัน" พิจารณาจากคำถามนี้ หากใช้ฐานข้อมูลทั่วไป ไม่มีทางเลยที่จะตอบได้แต่หากเป็นฐานข้อมูลเชิงเวลา จะตอบคำถามดังกล่าวได้ จะต้องคิดพิจารณาถึงเหตุการณ์การอยู่ร่วมกันของวัว 2 ฝูงดังนี้

1. วัว 2 ฝูงเคยอยู่ร่วมกันจริงในอดีต (Sequence Query)
2. วัว 2 ฝูงอยู่ร่วมกันจริงในปัจจุบัน (Current Query)
3. วัว 2 ฝูงเคยอยู่คอกเดียวกัน แต่คนละเวลา (Non-Sequence Query)

<u>คำตอบที่ 1</u> ฝูงเคยอยู่ร่วมกันจริงในอดีต หากพิจรณาเป็นเส้นเวลาจะสามารถเขียนได้ดังนี้

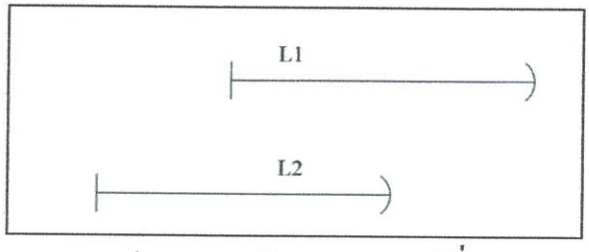
กรณีที่ 1 วัวฝูงที่ 1 มาอยู่คอกเดียวกันในช่วงเวลาที่ฝูงที่ 2 ยังคงอยู่คอกเดิม และออกจากคอกนี้ไปก่อนฝูง 2 จะออกดังนี้



รูป 2.1 แสดงลักษณะของกรณีที่ 1

(ที่มา Managing Temporal Data A Five-Part Series)

กรณีที่ 2 วัวฝูงที่ 1 มาอยู่ร่วมคอกเดียวกับ วัวฝูงที่ 2 โดยที่วัวฝูงที่ 2 นั้นมาอยู่ก่อน วัวฝูงที่ 1 มาอยู่ตามหลังและวัวฝูงที่ 2 ได้ออกไปจากคอกก่อนที่วัวฝูงที่ 1 จะออกตามไปดังแสดงให้เห็นในแผนภาพดังนี้



รูป 2.2 แสดงลักษณะของกรณีที่ 2

(ที่มา Managing Temporal Data A Five-Part Series)

กรณีที่ 3 คล้ายกรณีที่ 2 แต่ต่างกันที่วัวฝูงที่ 2 มาอยู่ร่วมคอกเดียวกับ วัวฝูงที่ 1 โดยที่วัวฝูงที่ 1 นั้นมาอยู่ก่อน วัวฝูงที่ 2 มาอยู่ตามหลังและวัวฝูงที่ 1 ได้ออกไปจากคอกก่อนที่วัวฝูงที่ 2 จะออกตามไป

กรณีที่ 4 คล้ายกรณีที่ 1 แต่ต่างกันที่วัวฝูงที่ 2 มาอยู่คอกเดียวกันในช่วงเวลาที่ฝูงที่ 1 ยังคงอยู่คอกเดิม และออกจากคอกนี้ไปก่อนฝูง 1 จะออก

เมื่อพิจารณากรณีที่เป็นไปได้ทั้งหมดแล้วต้องเขียนคำตอบสำหรับคำตอบนี้คำตอบเดียวถึง 4 กรณี ดังแสดงให้เห็นด้วยภาษาฐานข้อมูลเอสคิวแอล 2 (SQL 2 หรือ SQL-92) ดังนี้

```
SELECT L1.LOT ID NUM, L2.LOT ID NUM, L1.PEN ID, L1.FROM DATE, L1.TO DATE
FROM LOT LOC AS L1, LOT LOC AS L2 WHERE L1.LOT ID NUM < L2.LOT ID NUM AND
L1.FDYD ID = L2.FDYD ID AND L1.PEN ID = L2.PEN ID AND L2.FROM DATE <= L1.FROM
DATE AND L1.TO DATE <= L2.TO DATE

UNION SELECT L1.LOT ID NUM, L2.LOT ID NUM, L1.PEN ID, L1.FROM DATE, L2.TO
DATE FROM LOT LOC AS L1, LOT LOC AS L2 WHERE L1.LOT ID NUM < L2.LOT ID NUM
AND L1.FDYD ID = L2.FDYD ID AND L1.PEN ID = L2.PEN ID AND L1.FROM DATE >
L2.FROM DATE AND L2.TO DATE < L1.TO DATE AND L1.FROM DATE < L2.TO DATE

UNION SELECT L1.LOT ID NUM, L2.LOT ID NUM, L1.PEN ID, L2.FROM DATE, L1.TO
DATE FROM LOT LOC AS L1, LOT LOC AS L2 WHERE L1.LOT ID NUM < L2.LOT ID NUM
AND L1.FDYD ID = L2.FDYD ID AND L1.PEN ID = L2.PEN ID AND L2.FROM DATE >
L1.FROM DATE AND L1.TO DATE < L2.TO DATE AND L2.FROM DATE < L1.TO DATE

UNION SELECT L1.LOT ID NUM, L2.LOT ID NUM, L1.PEN ID, L2.FROM DATE, L2.TO
DATE FROM LOT LOC AS L1, LOT LOC AS L2 WHERE L1.LOT ID NUM < L2.LOT ID NUM
AND L1.FDYD ID = L2.FDYD ID AND L1.PEN ID = L2.PEN ID AND L2.FROM DATE >=
L1.FROM DATE AND L2.TO DATE <= L1.TO DAT
```

จะเห็นได้ว่าแค่คำตอบเดียว จำเป็นต้องเขียนภาษาฐานข้อมูลเอสคิวแอลมากมายเพื่อหาคำตอบให้ครอบคลุมถึง 4 กรณีด้วยกัน ซึ่งมันอาจจะไม่ใช่เรื่องยาก แต่มันเป็นเรื่องยุ่งที่จะต้องเขียนโดยไม่ให้มีข้อผิดพลาด

คำตอบที่ 2 วัว 2 ฝูงอยู่ร่วมกันจริงในปัจจุบัน

```
SELECT L1.LOT ID NUM, L2.LOT ID NUM, L1.PEN ID FROM LOT LOC AS L1, LOT
LOC AS L2 WHERE L1.LOT ID NUM < L2.LOT ID NUM AND L1.FDYD ID = L2.FDYD
ID AND L1.PEN ID = L2.PEN ID AND L1.TO DATE = DATE '9999-12-31' AND L2.TO
DATE = DATE '9999-12-31'
```

คำตอบที่ 3 วัว 2 ฝูงเคยอยู่คอกเดียวกันแต่คนละเวลา

```
SELECT L1.LOT ID NUM, L2.LOT ID NUM, L1.PEN ID FROM LOT LOC AS L1, LOT LOC
AS L2 WHERE L1.LOT ID NUM < L2.LOT ID NUM AND L1.FDYD ID = L2.FDYD ID AND
L1.PEN ID = L2.PEN ID
```

สุดท้ายจะต้องนำคำตอบทั้งหมดของแต่ละคำถามมายูเนียน (Union) กันเพื่อตอบคำถามที่ว่า "วัวตัวใดเคยอยู่คอกเดียวกัน" จากความยุ่งยากดังกล่าวอาจารย์ Richard Snodgrass ผู้เขียนบทความได้พยายามจะผลักดันให้เกิดเป็นคุณลักษณะใหม่ในภาษาฐานข้อมูลเอสคิวแอล เพื่อลดความยุ่งยากในการพัฒนาโปรแกรมประยุกต์เชิงเวลาลงดังนี้

คำถามเดิม "วัวคอกไหนเคยอยู่คอกเดียวกันบ้าง" สำหรับคำตอบที่ว่า วัวต้องเคยอยู่คอกเดียวกันจริงไม่ว่าจะช่วงเวลาใดในอดีตสามารถเขียนได้โดยเพิ่มคำสำคัญ (keyword) "VALIDTIME" ดังนี้

```
VALIDTIME SELECT L1.LOT ID NUM, L2.LOT ID NUM, L1.PEN ID FROM LOT LOC AS
L1, LOT LOC AS L2 WHERE L1.LOT ID NUM < L2.LOT ID NUM AND L1.FDYD ID =
L2.FDYD ID AND L1.PEN ID = L2.PEN ID
```

ระบบจัดการฐานข้อมูลจะจัดการให้ครอบคลุมทั้ง 4 กรณีที่เกิดขึ้นให้เองโดยที่ผู้พัฒนาโปรแกรมประยุกต์ไม่จำเป็นต้องรู้ว่าระบบฐานข้อมูลมีคอลัมน์เวลาอยู่ด้วยและหากต้องการคำตอบที่ว่า วัวเคยอยู่คอกเดียวกันแต่คนละช่วงเวลา ก็สามารถทำได้เพียงแค่ระบุคำสำคัญ "NONSEQUENCED" ก่อนหน้า "VALIDTIME" เท่านั้น

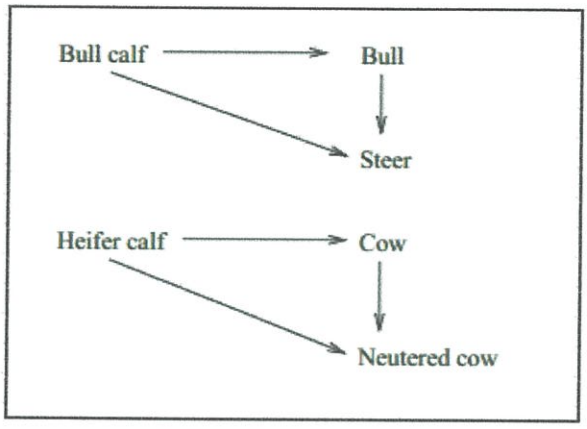
2.2.3.2 Current Modification

จะเห็นว่าการสอบถามข้อมูลเชิงเวลานั้นมีความยุ่งยากมากพอสมควรในการปรับปรุงข้อมูลเชิงเวลาก็มีความยุ่งยากมากไม่แพ้กัน โดยการแก้ไขข้อมูลในปัจจุบันนั้นมีการแบ่งการแก้ไขออกเป็น

- General scenario เป็นการอนุญาตให้แทรกข้อมูล ปรับปรุงข้อมูล ลบข้อมูลในช่วงเวลาไหนก็ได้

- Restricted scenario เป็นการอนุญาตให้แก้ไขข้อมูลตารางเฉพาะช่วงเวลาในปัจจุบันของข้อมูลเท่านั้นไม่สามารถแก้ไขเปลี่ยนแปลงอดีตได้

โดยจะขอใช้ตัวอย่างตารางแสดงสถานะของฝูงวัว ซึ่งมีแผนภาพสถานะของวัวเพื่อประกอบการอธิบาย ดังนี้



รูป 2.3 แสดงสถานะของวัวก่อนถูกตอนแยกตามเพศ
(ที่มา Managing Temporal Data A Five-Part Series)

ตาราง 2.9 สถานะของฝูงวัว [ที่มา Managing Temporal Data A Five-Part Series]

LOT_ID_NUM	GNDR_CODE	FROM_DATE	TO_DATE
101	C	1998-01-01	1998-03-23
101	S	1998-03-23	9999-12-31
234	C	1998-02-17	9999-12-31
799	S	1998-03-12	9999-12-31

จากตารางจะเห็นได้ว่าฝูง 101 เป็น calf ตั้งแต่วันที่ 1998-01-01 ถึง 1998-03-23 จากนั้นก็ถูกตอนตั้งแต่วันที่ 1998-03-23 จนถึงปัจจุบัน

1) Current Insert

ถ้าต้องการจัดเก็บ Fact ที่เป็นจริงตั้งแต่วันนี้ สามารถทำได้ ตัวอย่างเช่น

```
INSERT INTO LOT

VALUES (433, 'h', CURRENT DATE, DATE '9999-12-31')
```

2) Current Delete

ถ้าต้องการลบข้อมูล LOT101 ในปัจจุบัน จะไม่สามารถใช้การลบข้อมูลแบบปกติดังนี้ได้

```
DELETE FROM LOT
WHERE LOT ID NUM = 101
```

จะต้องพิจารณาเวลาที่คาบเกี่ยวกันด้วย เพื่อไปลบเฉพาะเวลาในปัจจุบัน โดยใช้ตัวอย่างของตารางแสดงสถานะของฝูงวัว ดังนี้

ตาราง 2.10 สถานะของฝูงวัวใช้สำหรับตัวอย่าง Current Delete [ที่มา Managing Temporal Data A Five-Part Series]

LOT_ID_NUM	GNDR_CODE	FROM_DATE	TO_DATE
101	C	1998-01-01	1998-03-23
101	S	1998-03-23	9999-12-31
234	C	1998-02-17	1998-10-17
234	S	1998-10-17	9999-12-31
799	S	1998-03-12	9999-12-31

จากตารางข้างต้น ฝูง 234 เป็น Calf ตั้งแต่วันที่ 1998-02-17 ถึง 1998-10-17 และวางแผนว่าจะตอนในวันที่ 1998-10-17 โดยสมมติว่าวันนี้เป็นวันที่ 1998-07-29 ถ้าต้องการลบข้อมูลฝูง 234 จะต้องแก้ไขให้ข้อมูลสิ้นสุดที่วันนี้ และอนาคตต้องไม่มีข้อมูลนั้นอยู่แล้ว ดังนี้

```
UPDATE LOT
SET TO DATE = CURRENT DATE
WHERE LOT ID NUM = 234
AND TO DATE >= CURRENT DATE
AND FROM DATE < CURRENT DATE


DELETE FROM LOT
WHERE LOT ID NUM = 234
AND FROM DATE > CURRENT DATE
```

คำสั่งดังกล่าวเป็นการแก้ไขฝูง 234 C 1998-02-17 1998-10-17 ซึ่งเป็นแถวที่มีเวลาคาบเกี่ยวกับเวลาปัจจุบัน แก้ไขโดยปรับ TO_DATE เป็นวันที่ปัจจุบันและลบ 234 S 1998-10-17 9999-12-31 ที่เป็นเวลาในอนาคตทิ้งไป สุดท้ายจะได้ผลลัพธ์จากการทำ Current Delete ดังนี้

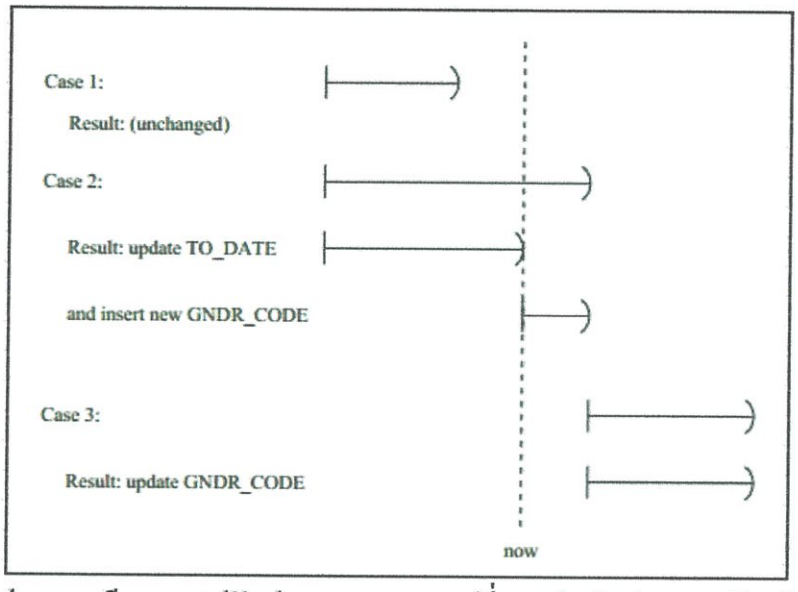
ตาราง 2.11 ผลลัพธ์สถานะของฝูงวัว หลังจากการทำ Current Delete

[ที่มา Managing Temporal Data A Five-Part Series]

LOT_ID_NUM	GNDR_CODE	FROM_DATE	TO_DATE
101	C	1998-01-01	1998-03-23
101	S	1998-03-23	9999-12-31
234	C	1998-02-17	1998-07-29
799	C	1998-03-12	9999-12-31

3) Current Update

ในการปรับปรุงข้อมูลก็ต้องพิจารณาช่วงเวลาที่คาบเกี่ยวกันด้วยเช่นกัน ไม่สามารปรับปรุงโดยวิธีปกติได้ โดยต้องตรวจสอบช่วงเวลาระหว่างเวลาเดิมก่อนการปรับปรุงกับเวลาที่ต้องการให้เป็นว่ามันคาบเกี่ยวกับแบบไหน ซึ่งสามารถพิจาณาได้ 3 กรณีดังนี้



รูป 2.4 กรณีของการปรับปรุงเวลาของการเปลี่ยนแปลงในช่วงเวลาปัจจุบัน

(ที่มา Managing Temporal Data A Five-Part Series)

กรณีที่ 1 ปรับปรุงข้อมูลในอดีตในเวลาปัจจุบัน

โดยแก้ไข TO_DATE เป็นเวลาในปัจจุบัน

กรณีที่ 2 ปรับปรุงข้อมูลที่ยังเป็นจริงอยู่ในเวลาปัจจุบัน

โดยการปรับปรุง TO_DATE ของแถวเดิมเป็นเวลาปัจจุบัน และแทรกแถวใหม่ที่มี FROM_DATE เป็นเวลาปัจจุบัน

กรณีที่ 3 ปรับปรุงข้อมูลในอนาคตในแวลาปัจจุบัน

2.2.3.3 Sequenced Modification

1) Sequence Insert

ถ้าจะจัดเก็บข้อมูล Sequence ต้องบอกด้วยว่า Fact นั้น valid ตั้งแต่เมื่อไหร่ถึงเมื่อไหร่ ตัวอย่างเช่น

```
INSERT INTO LOT

VALUES (426, 'h', DATE '1998-03-26', DATE '1998-04-14')
```

2) Sequence Delete

จากตัวอย่างได้มีการให้ฝูงวัว 234 ย้ายออกจากฟาร์ม 3 สัปดาห์แรกของเดือนตุลาคม ซึ่งเป็นช่วงเวลาที่มีการวางแผนจะเปลี่ยนฝูงวัว 234 ให้เป็นวัวตอน โดยช่วงเวลาที่ต้องการเปลี่ยนแปลงข้อมูล คือวันที่ '1998-10-01' ถึงวันที่ '1998-10-22' โดยการเปลี่ยนแปลงลักษณะนี้ทำให้ต้องพิจารณาการแปลี่ยนแปลงข้อมูล ซึ่งจะเกิดกรณีของการเปลี่ยนแปลงข้อมูลได้ 4 กรณี

PV คือ ช่วงเวลาของข้อมูลเดิม กล่าวคือเป็นช่วงเวลาที่ Fact เป็นจริงอยู่ในฐานข้อมูล

PA คือ ช่วงเวลาที่ต้องการเปลี่ยนแปลงข้อมูล กล่าวคือเป็นช่วงเวลาที่ถูกปฏิบัติ

กรณีที่ 1 PA เกิดทีหลังและสิ้นสุดก่อน PV

ผลของการลบข้อมูลจะเกิดการปรับปรุง 1 แถว โดย FROM_DATE เท่ากับ FROM_DATE ของ PV และเปลี่ยน TO_DATE เป็น FROM_DATE ของ PA รวมทั้งแทรกอีก 1 แถว โดยให้ FROM_DATE เท่ากับ TO_DATE ของ PA และ FROM_DATE เท่ากับ FROM_DATE ของ PV

กรณีที่ 2 PA เกิดขึ้นหลัง PV และสิ้นสุดหลัง PV

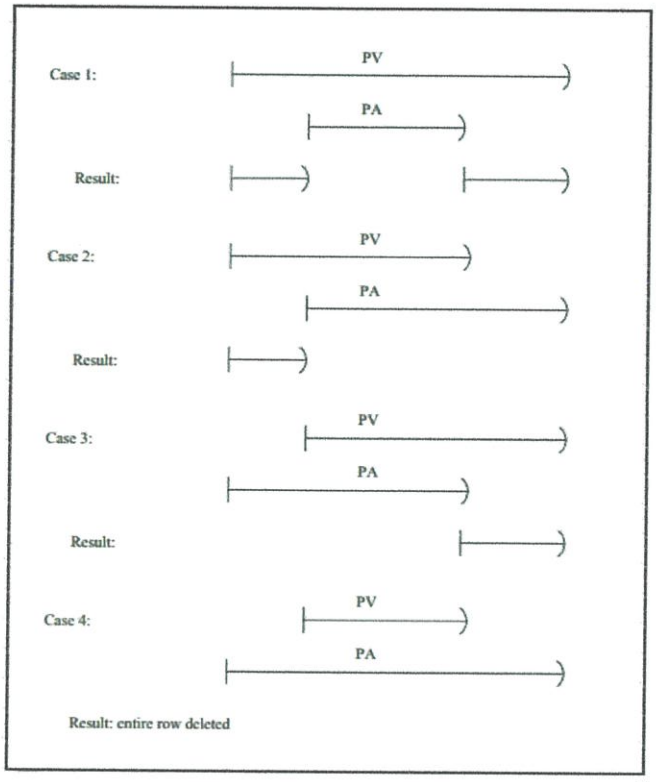
ผลของการลบข้อมูลจะเป็นการปรับปรุง 1 แถว โดย FROM_DATE เท่ากับ FROM_DATE ของ PV และ TO_DATE เท่ากับ FROM_DATE ของ PA

กรณีที่ 3 PA เกิดก่อน PV และสิ้นสุดก่อน PV

ผลของการลบข้อมูลจะเป็นการปรับปรุง 1 แถว โดย FROM_DATE เท่ากับ TO_DATE ของ PA และ TO_DATE เท่ากับ TO_DATE ของ PV

กรณีที่ 4 PA เกิดขึ้นก่อนและสิ้นสุดหลัง PV

ผลของการลบข้อมูลจะเป็นการลบแถวข้อมูลของ PA ออกจากฐานข้อมูล



รูป 2.5 กรณีของลบช่วงเวลาแบบ Sequence (Sequence delete) ทั้ง 4 กรณี (ที่มา Managing Temporal Data A Five-Part Series)

โดยจากตัวอย่าง ถ้าต้องการลบฝูง 234 ในช่วงเวลาตั้งแต่วันที่ 1998-10-01 ถึง 1998-10-22 จะต้องพิจารณาให้ครบ 4 กรณี โดยใช้คำสั่งภาษาฐานข้อมูลเอสคิวแอล ดังนี้

กรณีที่ 1

```
INSERT INTO LOT

SELECT LOT ID NUM, GNDR CODE, DATE '1998-10-22', TO
DATE

FROM LOT

WHERE LOT ID NUM = 234

AND FROM DATE <= DATE '1998-10-01'

AND TO DATE > DATE '1998-10-22'
```

กรณีที่ 2

```
UPDATE LOT
SET TO DATE = DATE '1998-10-01'
WHERE LOT ID NUM = 234
AND FROM DATE < DATE '1998-10-01'
AND TO DATE >= DATE '1998-10-01'
```

กรณีที่ 3

```
UPDATE LOT
SET FROM DATE = DATE '1998-10-22'
WHERE LOT ID NUM = 234
AND FROM DATE < DATE '1998-10-22'
AND TO DATE >= DATE '1998-10-22'
```

กรณีที่ 4

```
DELETE FROM LOT
WHERE LOT ID NUM = 234
AND FROM DATE >= DATE '1998-10-01'
AND TO DATE <= DATE '1998-10-22'
```

จะเห็นว่าการทำ Sequence Delete จะต้องเขียนคำสั่งเอสคิวแอลมาตรฐานถึง 4 คำสั่ง เพื่อให้เกิดการลบที่ถูกต้อง โดยในแต่ละคำสั่งผู้พัฒนาโปรแกรมจะต้องตรวจสอบเงื่อนไขของช่วงเวลาให้ถูกต้องทั้ง FROM_DATE และ TO_DATE ซึ่งเป็นเรื่องที่ซับซ้อนและยุ่งยากเป็นอย่างมาก ถ้าเขียนผิดก็อาจส่งผลให้เกิดการลบที่ไม่ถูกต้องได้

ซึ่งก็ยังมีคำสั่งเอสคิวแอลที่เขียนง่ายกว่า โดยระบุช่วงเวลาที่ต้องการลบ จากนั้นระบบจัดการฐานข้อมูล จะตรวจสอบเงื่อนไขทั้ง 4 ให้โดยอัตโนมัติ คำสั่งจะเป็นดังนี้

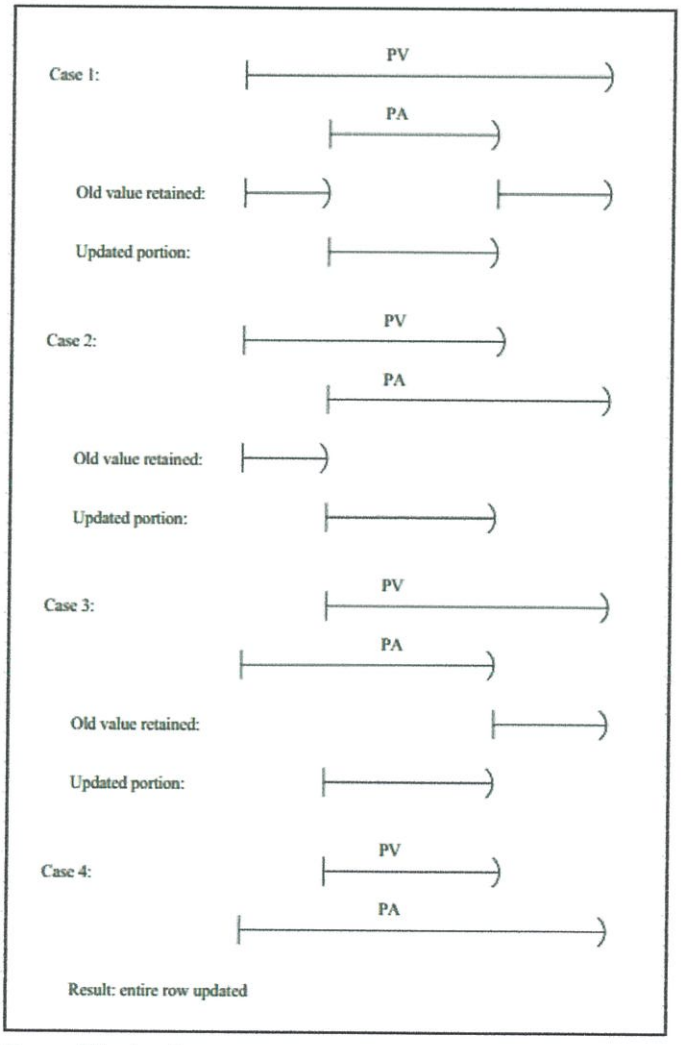
```
VALIDTIME PERIOD '[1998-10-01 - 1998-10-22)'

DELETE FROM LOT

WHERE LOT ID NUM = 234
```

3) Sequence Update

ตัวอย่าง ถ้าต้องการเปลี่ยนวัว LOT 799 เป็นวัวตอนตัวผู้ เฉพาะในเดือนมีนาคม การเปลี่ยนแปลงนี้จะต้องพิจารณา 4 กรณี ดังนี้



รูป 2.6 กรณีของปรับปรุงช่วงเวลาแบบ Sequence (Sequence Update) ทั้ง 4 กรณี (ที่มา Managing Temporal Data A Five-Part Series)

กรณีที่ 1 PA เกิดขึ้นหลังและสิ้นสุดก่อน PV

ผลของการปรับปรุงจะเป็นการเพิ่มแถว 2 แถวและ ปรับปรุง 1 แถว ดังนี้

- เพิ่มแถวแรกโดย FROM_DATE มีค่าเท่ากับ FROM_DATE ของ PV และ TO_DATE มีค่าเท่ากับ FROM_DATE ของ PA
- เพิ่มแถวที่สองโดย FROM_DATE มีค่าเท่ากับ TO_DATE ของ PA และ FROM_DATE มีค่าเท่ากับ FROM_DATE ของ PV
- ปรับปรุงแถวข้อมูลหนึ่งแถวโดยปรับปรุง FROM_DATE ของ PV ให้เป็น FROM_DATE ของ PA และ TO_DATE ของPV ให้มีค่าเป็น TO_DATE ของ PA และค่าของข้อมูลจะเท่ากับค่าของ PA

กรณีที่ 2 PA เกิดขึ้นและสิ้นสุดหลัง PV

ผลของการปรับปรุงจะเป็นการเพิ่มแถว 1 แถว และปรับปรุง 1 แถว

- เพิ่มแถวโดย FROM_DATE มีค่าเท่ากับ FROM_DATE ของ PV และ TO_DATE มีค่าเท่ากับ FROM_DATE ของ PA
- ปรับปรุงแถวโดยปรับปรุง FROM_DATE ของ PV ให้เป็น FROM_DATE ของ PA และ TO_DATE ของ PV มีค่าเท่ากับ TO_DATE ของ PA และค่าของข้อมูลจะเท่ากับค่าของ PA

กรณีที่ 3 PA เกิดขึ้นและสิ้นสุดก่อน PV

ผลของการปรับปรุงจะเป็นการเพิ่ม 1 แถวและปรับปรุง 1 แถว

- เพิ่มแถวโดย FROM_DATE มีค่าเท่ากับ TO_DATE ของ PA และ TO_DATE มีค่าเท่ากับ TO_DATE ของ PV
- ปรับปรุงแถวโดยปรับปรุง FROM_DATE ของ PV ให้เป็น FROM_DATE ของ PA และ TO_DATE ของ PV มีค่าเท่ากับ TO_DATE ของ PA และค่าของข้อมูลจะเท่ากับค่าของ PA

กรณีที่ 4 PA เกิดขึ้นก่อนและสิ้นสุดหลัง

ผลของการปรับปรุงจะเป็นการปรับปรุง 1 แถว คือ ปรับปรุงเฉพาะค่า PA โดย FROM_DATE และ TO_DATE ยังมีค่าเท่าเดิม

ซึ่งสามารถใช้คำสั่งเอสคิวแอลในการปรับปรุงข้อมูลแบบ Sequence ได้ดังนี้

```
INSERT INTO LOT

SELECT LOT ID NUM, GNDR CODE, FROM DATE, DATE '1998-03-01'

FROM LOT

WHERE LOT ID NUM = 799

AND FROM DATE < DATE '1998-03-01'

AND TO DATE > DATE '1998-03-01'


INSERT INTO LOT

SELECT LOT ID NUM, GNDR CODE, DATE '1998-04-01', TO DATE

FROM LOT

WHERE LOT ID NUM = 799

AND FROM DATE < DATE '1998-04-01'

AND TO DATE > DATE '1998-04-01'


UPDATE LOT

SET GNDR CODE = 's'

WHERE LOT ID NUM = 799

AND FROM DATE < DATE '1998-04-01'

AND TO DATE > DATE '1998-03-01'
```

```
UPDATE LOT
SET FROM DATE = DATE '1998-03-01'
WHERE LOT ID NUM = 799
AND FROM DATE < DATE '1998-03-01'
AND TO DATE > DATE '1998-03-01'

UPDATE LOT
SET TO DATE = DATE '1998-04-01'
WHERE LOT ID NUM = 799
AND FROM DATE < DATE '1998-04-01'
AND TO DATE > DATE '1998-04-01'
```

จะเห็นได้ว่าการทำ Sequence Update ต้องเขียนคำสั่งเอสคิวแอลมาตรฐานถึง 5 คำสั่ง เพื่อให้เกิดการปรับปรุงข้อมูลที่ถูกต้อง โดยในแต่ละคำสั่งผู้พัฒนาโปรแกรมจะต้องตรวจสอบเงื่อนไขของช่วงเวลาให้ถูกต้องทั้ง FROM_DATE และ TO_DATE ซึ่งเป็นเรื่องที่ซับซ้อนและยุ่งยากเป็นอย่างมาก ถ้าเขียนผิดก็อาจส่งผลให้เกิดการปรับปรุงที่ไม่ถูกต้องได้

ซึ่งก็ยังมีคำสั่งเอสคิวแอลที่เขียนง่ายกว่า โดยระบุช่วงเวลาที่ต้องการปรับปรุง จากนั้นระบบจัดการฐานข้อมูลจะตรวจสอบเงื่อนไขทั้ง 4 ให้โดยอัตโนมัติ คำสั่งจะเป็นดังนี้

```
VALIDTIME PERIOD '[1998-03-01 - 1998-04-01)'
UPDATE LOT
SET GNDR CODE = 's'
WHERE LOT ID NUM = 799
```

ความสามารถดังกล่าวนี้ได้ถูกเพิ่มเติมกลายเป็นมาตรฐานภาษาเอสคิวแอล 2011 เป็นที่เรียบร้อย แต่ด้วยความที่ Oracle เล็งเห็นว่าความสามารถด้านเวลาของระบบฐานข้อมูลนั้นมีความสำคัญเป็นอย่างมากต่อภาครัฐและภาคเอกชน Oracle จึงได้เพิ่มเติมความสามารถในการบริหารจัดการและตอบคำถามเกี่ยวกับข้อมูลเชิงเวลาขึ้นมาให้ผลิตภัณฑ์ของตัวเองภายใต้คุณลักษณะการทำงานพิเศษที่มีชื่อว่า Workspace Manage ซึ่งได้พัฒนาจนแล้วเสร็จก่อนที่จะมีมาตรฐานภาษาเอสคิวแอล 2011 ซึ่งการจัดการกับฐานข้อมูลเชิงเวลาด้วยมาตรฐานภาษาเอสคิวแอล 2011 นั้นได้ถูกนำเสนอโดย Krishna Kulkarni และ Jan-Eike Michels จากบริษัท IBM Corporation ทำให้ระบบจัดการฐานข้อมูลของ IBM ซึ่งคือ DB2 เวอร์ชัน 10 ที่คาดว่าได้ออกตามหลังมาตรฐานภาษาเอสคิวแอล 2011 จึงเป็นระบบจัดการฐานข้อมูลที่สามารถดูแลและจัดการความสามารถทางด้านเวลาได้อย่างครบถ้วนและตรงตามมาตรฐานภาษาเอสคิวแอล ซึ่งแน่นอนว่าทาง Oracle ผู้ซึ่งพัฒนาความสามารถด้านเวลานี้เสร็จสิ้นก่อนจะมีมาตรฐานภาษา ย่อมไม่ใช่สิ่งที่เป็นไปตามมาตรฐาน Oracle จึงจำเป็นที่จะต้องปรับปรุงสิ่งตนมีอยู่ให้เป็นไปตามมาตฐานภาษาเอสคิวแอล ดังนั้น Oracle จึงได้เพิ่มเติมความสามารถนี้ไปยัง Oracle รุ่นล่าสุดคือรุ่น 12c และรับประกันว่าความสามารถในการบริหารจัดการข้อมูลด้านเวลาของตนเองนั้นเป็นไปตามมาตรฐานภาษาเอสคิวแอล 2011 เป็นที่เรียบร้อย แต่ทางคณะผู้จัดทำก็ได้นำความสามารถเหล่านี้ไปทดลองตามเอกสารอ้างอิงมาตรฐานภาษาเอสคิวแอล 2011 กลับพบว่า Oracle 12c ยังไม่สามารถใช้คำสำคัญในการตอบคำถามและบริหารจัดการข้อมูลเชิงเวลาตามมาตรฐานได้ ทำได้เพียงแค่การสร้างตาราง (Create) และการจัดเก็บข้อมูลเชิงเวลาเท่านั้น

2.2.4 ตาราง Transaction-Time State

ตาราง Transaction-Time State เป็นตารางที่บันทึกช่วงเวลาที่บันทึกข้อมูลลงฐานข้อมูล ซึ่งเป็นเวลาที่เชื่อว่า Fact นั้นเป็นจริง (valid) ตัวอย่างเช่น ตารางบันทึกตำแหน่งและความสว่างของดวงดาว โดยแต่ก่อนค่าตำแหน่งและความสว่างจะถือว่าเป็นเพียงค่าหนึ่ง แต่พอต่อมามีเทคโนโลยีในการวัดที่แม่นยำและทันสมัยมากขึ้น จึงพบว่าตำแหน่งและค่าความสว่างของดวงดาวนั้นๆ ได้เปลี่ยนแปลงไป จึงมีการแก้ไข Fact ที่เคยมีในตารางหลัก และบันทึกค่าเก่าไว้ในตาราง Transaction-Time State แสดงตัวอย่างดังต่อไปนี้

ตาราง 2.12 WDS บันทึกตำแหน่งและความสว่างของดวงดาว [ที่มา Managing Temporal Data A Five-Part Series]

RA_Hour	RA_Min	RA_Sec	Dec_Degree	Dec_Minute	Discoverer	Mag_First
00	00	08	75	30	'A 1248'	10.5
05	57	40	00	02	'BU 1190'	6.5
04	13	20	50	32	'CHR 15'	15.5
01	23	70	-09	55	'HJ 3433'	10.5

จากตารางข้างต้น เป็นตารางหลักที่บันทึกตำแหน่งและความสว่างของดวงดาวที่เป็นข้อมูลที่เป็นจริงในปัจจุบัน โดย RA_Hour RA_Min และ RA_Sec เป็นคอลัมน์ในการบันทึกตำแหน่งของดวงดาวแบบลองจิจูดเป็น ชั่วโมง นาที และวินาทีตามลำดับ, Dec_Degree และ Dec_Minute เป็นคอลัมน์ในการบันทึกตำแหน่งของดวงดาวแบบละติจูด, Discoverer แทนชื่อของดวงดาว และ Mag_First แทนความสว่างของดวงดาว

ตาราง 2.13 Audit log ของ WDS (WDS_TT) [ที่มา Managing Temporal Data A Five-Part Series]

RA_ Hour	RA_ Min	RA_ Sec	Dec_ Degree	Dec_ Minute	Discoverer	Mag_ First	Trans_Start	Trans_Stop
00	00	08	75	30	'A 1248'	12.0	1989-03-12	1992-11-15
00	00	09	75	30	'A 1248'	12.0	1992-11-15	1994-05-18
00	00	09	75	30	'A 1248'	10.5	1994-05-18	1995-07-23
00	00	08	75	30	'A 1248'	10.5	1995-07-23	9999-12-31
05	57	40	00	02	'BU 1190'	6.5	1988-11-08	9999-12-31
04	13	20	50	32	'CHR 15'	15.5	1990-02-09	9999-12-31
01	23	70	-09	55	'HJ 3433'	10.5	1991-03-25	9999-12-31
02	33	10	-09	25	'LDS3402'	10.6	1993-12-19	1996-07-09

ตาราง Audit log ของ WDS เป็นตาราง Transaction-Time State สำหรับบันทึกการเปลี่ยนแปลงค่าของตำแหน่งและค่าความสว่างของดวงดาวที่เป็นค่าในอดีต จะเห็นได้ว่ามีการเพิ่ม Trans_Start และ Trans_Stop เพื่อบันทึกช่วงเวลาที่มีการบันทึกข้อมูลใหม่ซึ่งเป็นข้อมูลที่มีการเปลี่ยนแปลงและเป็นจริง กล่าวคืออาจเป็นเวลาที่ค้นพบการเปลี่ยนแปลงก็เป็นได้ ซึ่งการจัดการ

บันทึกเป็นหน้าที่ของระบบจัดการฐานข้อมูล เท่านั้น ผู้ใช้ฐานข้อมูลทั่วไปจะไม่สามารถเปลี่ยนแปลงค่าเวลาที่ถูกบันทึกในตาราง Transaction-Time State ข้างต้นนี้ได้

ถ้ามีการแทรกข้อมูลหรือปรับปรุงข้อมูลในตาราง WDS เมื่อใดก็ตามระบบจัดการฐานข้อมูลก็จะทำการเปลี่ยนแปลงลงในตาราง WDS_TT โดยอัตโนมัติ

2.2.5 ตาราง Bitemporal

เป็นตารางที่มีการเก็บข้อมูลทั้งในส่วนของ Valid time และ Transaction time กล่าวคือเป็นตารางที่เก็บช่วงเวลาที่ค่าข้อมูลนั้นยังคงเป็นจริงอยู่และเก็บช่วงเวลาที่ได้เปลี่ยนแปลงข้อมูลเหล่านั้น

ตาราง 2.14 Bitemporal (WDS_B) [ที่มา Managing Temporal Data A Five-Part Series]

Discoverer	Mag_ First	Trans_Start	Trans_Stop	Valid_From	Valid_To
'A 1248'	12.0	1989-03-12	1995-11-15	1922-05-14	9999-12-31
'A 1248'	12.0	1995-11-15	9999-12-31	1922-05-14	1994-10-16
'A 1248'	10.5	1995-11-15	9999-12-31	1994-10-16	9999-12-31

จากตารางจะเห็นว่าดวงดาว A-1248 วัดครั้งแรกเมื่อ 1998-03-12 โดยเชื่อว่าดาว A-1248 สว่าง 12.0 ตั้งแต่ 1922-05-14 ถึง 9999-12-31 ต่อมาเมื่อถึง 1995-11-15 มีการบันทึกใหม่สองแถว เนื่องจากค้นพบว่าดวงดาวนั้นมีความสว่างใหม่คือ 10.5 โดยแสดงว่าความสว่างเดิม 12.0 เป็นจริงตั้งแต่ 1922-05-14 ถึง 1994-10-16 และดวงดาวดวงนี้มีความสว่างใหม่คือ 10.5 เป็นจริงตั้งแต่วันที่ 1994-10-16 ถึงปัจจุบัน [1]

2.3 การจัดการกับฐานข้อมูลเชิงเวลาด้วยมาตรฐานภาษาเอสคิวแอล 2011 (Temporal Feature in SQL 2011)

ภาษาเอสคิวแอล 2011 เป็นภาษาที่สามารถบริหารจัดการและสามารถตอบคำถามเกี่ยวกับข้อมูลเชิงเวลาได้ ในปัจจุบันถือเป็นภาษาที่ได้รับการยอมรับจากคณะกรรมการมาตรฐาน ISO ซึ่งในภาษาเอสคิวแอล 2011 นี้ได้เพิ่มเติมคำสั่งในการการปรับปรุงข้อมูลเชิงเวลา (Update) การลบข้อมูลเชิงเวลา (Delete) และการสอบถามข้อมูลเชิงเวลา (Select) อีกทั้งในส่วนของการดำเนินการเชิงเวลา เช่น การสอบถามคำถามที่มีการคาบเกี่ยวกันของเวลาเป็นต้น โดยภาษาเอสคิวแอล 2011 ได้เพิ่มเติมส่วนของคำสั่งที่ช่วยให้นักพัฒนาสะดวกในการเขียนโปรแกรมในการบริหารจัดการกับความความถูกต้องเชิงเวลา รวมทั้งยังดูแลเรื่องกฎบังคับความถูกต้องของฐานข้อมูลเชิงเวลาหลังจากที่ได้ทำการปรับปรุงระบบฐานข้อมูลเชิงเวลาไปแล้วอีกด้วย ซึ่งหากไม่มีขีดความสามารถของภาษามารองรับ สิ่งเหล่านี้เป็นสิ่งที่ผู้พัฒนาโปรแกรมจะต้องจัดการดูแลเองทั้งหมด ซึ่งเป็นเรื่องที่มีความยากลำบากและซับซ้อนเป็นอย่างมาก

2.3.1 ช่วงเวลา (Period)

ในฐานข้อมูลเชิงเวลานั้นมีความสามารถในการกำหนดและเชื่อมโยงช่วงเวลา (Time periods) ของแถวต่างๆ ในตาราง โดยมีการกำหนดขอบเขตเวลาเริ่มต้น (Start time) ถึงเวลาสิ้นสุด (End time) เป็นข้อมูลชนิดช่วงเวลา (Period data type) ซึ่งปัจจุบันนี้ภาษาเอสคิวแอล 2011 ได้เพิ่มนิยามของช่วงเวลา (Period definition) เป็นส่วนหนึ่งของคำอธิบายข้อมูล (Metadata) ของตาราง สามารถระบุช่วงเวลาได้โดยใช้สองคอลัมน์ที่บ่งบอกเวลาเริ่มต้นและเวลาสิ้นสุด และใช้คำสั่งสร้างตาราง (CREATE TABLE) และคำสั่งแก้ไขตาราง (ALTER TABLE) ในการเพิ่มส่วนของช่วงเวลา

ภาษาเอสคิวแอล 2011 จะใช้ช่วงเวลารูปแบบ Closed-open ซึ่งเป็นรูปแบบช่วงเวลาที่นับเวลาเริ่มต้น และดำเนินต่อไปเรื่อยๆ โดยไม่นับรวมจุดเวลาสิ้นสุด กล่าวคือข้อมูลจะเป็นจริงตั้งแต่เวลาเริ่มต้นและก่อนเวลาสิ้นสุดหนึ่งวัน ในกรณีที่กำหนดให้ทั้ง 2 คอลัมน์เป็นข้อมูลชนิดวันที่ (Date) โดยเมื่อกำหนดช่วงเวลาที่เป็นจริงของข้อมูลในแถวนั้นๆ ต้องกำหนดเวลาสิ้นสุดให้มากกว่าเวลาเริ่มต้น ซึ่งการประกาศช่วงเวลาในตารางนั้นจะมีข้อจำกัดที่ใช้บังคับคุณสมบัตินี้อยู่แล้ว

ฐานข้อมูลเชิงเวลาตามมาตรฐานภาษาเอสคิวแอล 2011 นี้ได้เสนอข้อมูลเวลาสำหรับรองรับข้อมูลเชิงเวลา ดังนี้

- Valid time คือ ช่วงเวลาที่ Fact นั้นเริ่มเป็นจริงตั้งแต่เวลาเริ่มต้นถึงเวลาสิ้นสุด ซึ่งเป็นเวลาที่ผู้ใช้กำหนดได้เอง

- Transaction time คือ ช่วงเวลาที่ถูกเก็บโดยระบบ โดยจะเก็บเวลาที่ Transaction ปฏิบัติกับ Fact ในช่วงเวลาที่ Fact นั้นยังเป็นจริงอยู่ภายในฐานข้อมูล

การจัดการ Transaction time สามารถปฏิบัติได้ใน System-versioned tables ที่ประกอบด้วย System-time period ส่วนการจัดการ Valid time นั้นสามารถจัดการได้ในตารางซึ่งประกอบด้วย Application-time period โดยชื่อช่วงเวลาของ System-time period ถูกกำหนดเป็นมาตรฐานในชื่อ SYSTEM_TIME และชื่อช่วงเวลาของ Application-time period สามารถกำหนดได้เองโดยผู้ใช้ฐานข้อมูล โดยในหนึ่งตารางจะสามารถกำหนดชื่อ Application-time period ได้เพียงแค่หนึ่งชื่อ รวมทั้งชื่อของ System-time period ก็เช่นกัน สามารถมีได้เพียงหนึ่งชื่อต่อหนึ่งตาราง

ข้อดีของการมีข้อมูลชนิดช่วงเวลาและมีการจัดการเชิงเวลาเพิ่มเติมในภาษาเอสคิวแอล 2011 นั้นทำให้สามารถจัดการกับฐานข้อมูลที่เก็บข้อมูลช่วงเวลาได้ง่ายขึ้น โดยใช้คุณลักษณะเพิ่มเติมด้านเวลาของภาษาเอสคิวแอล 2011

2.3.2 ตาราง Application-Time Period

ตาราง Application-time period มีวัตถุประสงค์เพื่อสนองความต้องการของแอพพลิเคชั่นที่ต้องการใช้งานระบบฐานข้อมูลเชิงเวลา และต้องการพิจารณาช่วงเวลาที่ข้อมูลยังเป็นจริง

ความต้องการหลักของแต่ละแอพพลิเคชั่นคือผู้ใช้จำเป็นต้องตั้งค่าเวลาเริ่มต้นและสิ้นสุด ซึ่งต้องเป็นช่วงเวลาที่เป็นจริงในแต่ละแถวข้อมูล โดยผู้ใช้มีอิสระที่จะกำหนดค่าเวลาใดๆ สำหรับเวลาเริ่มต้นและสิ้นสุด ทั้งในอดีต ปัจจุบันหรือในอนาคต รวมทั้งผู้ใช้ยังสามารถปรับปรุงช่วงเวลาที่เป็นจริงของแถวข้อมูลที่ค้นพบข้อผิดพลาดหรือมีข้อมูลใหม่เพิ่มขึ้นได้อีกด้วย

ตาราง Application-time period จะประกอบด้วยนิยามช่วงเวลา ซึ่งมีชื่อของช่วงเวลาที่กำหนดโดยผู้ใช้ สามารถสร้างได้ ดังนี้

```
CREATE TABLE Emp(
ENo INTEGER,
EStart DATE,
EEnd DATE,
EDept INTEGER,
PERIOD FOR EPeriod (EStart, EEnd)
)
```

ผู้ใช้สามารถตั้งชื่อของช่วงเวลา ชื่อของคอลัมน์ที่เป็นจุดเริ่มต้นและจุดสิ้นสุดของช่วงเวลาได้ตามต้องการ ชนิดข้อมูลของคอลัมน์เวลาเริ่มต้นและเวลาสิ้นสุดสามารถเป็นได้ทั้งชนิด DATE หรือชนิด Timestamp แต่ชนิดข้อมูลของทั้งสองคอลัมน์จะต้องเหมือนกัน

สำหรับคำสั่ง INSERT ใช้ในการกำหนดค่าเริ่มต้นของคอลัมน์เวลาเริ่มต้นและเวลาสิ้นสุด ดังตัวอย่างต่อไปนี้ ใช้คำสั่ง INSERT จัดเก็บหนึ่งแถวลงไปในตาราง Emp

```
INSERT INTO Emp
VALUES (22217,
DATE '2010-01-01',
DATE '2011-11-12', 3)
```

จะได้ผลลัพธ์ดังนี้ (โดยกำหนดว่าตารางเคยว่างมาก่อนหน้านี้)

ตาราง 2.15 ผลลัพธ์จากการจัดเก็บข้อมูล ในตาราง Application-time period ชื่อ Emp [ที่มา Temporal features in SQL: 2011]

ENO	EStart	EEnd	EDept
22217	2010-01-01	2011-11-12	3

สำหรับคำสั่ง UPDATE ใช้ในการปรับปรุงแถวของตาราง Application-time period อีกทั้งยังรวมไปถึงการปรับปรุงเวลาเริ่มต้นและเวลาสิ้นสุด ในทำนองเดียวกันคำสั่ง DELETE ใช้ในการลบแถวของตาราง Application-time period

คุณลักษณะใหม่ที่มีในภาษาเอสคิวแอล 2011 คือความสามารถในการปรับเปลี่ยนช่วงเวลาที่ถูกกำหนดโดยอัตโนมัติ เพื่อให้ข้อมูลเหล่านั้นมีความสอดคล้องกันและยังคงเป็นจริงอยู่ในช่วงเวลาหนึ่ง ความสามารถนี้จะถูกใช้เมื่อมีการปรับปรุงข้อมูลและการลบข้อมูล

ตัวอย่างเช่น ต้องการเปลี่ยนแปลงแผนกของพนักงานที่มีหมายเลข 22217 ซึ่งได้ย้ายไปอยู่แผนก 4 ตั้งแต่วันที่ 3 กุมภาพันธ์ 2011 ถึง 10 กันยายน 2011 โดยในการปรับปรุงข้อมูลจะมีการใช้คำสั่ง `FOR PORTION OF` เพิ่มขึ้นมา

```
UPDATE Emp
 FOR PORTION OF EPeriod
  FROM DATE '2011-02-03'
  TO DATE '2011-09-10'
SET EDept = 4
WHERE ENo = 22217
```

ในการดำเนินการคำสั่งนี้ ระบบจัดการฐานข้อมูลจะหาแถวทั้งหมดที่มีช่วงเวลาคาบเกี่ยวกับช่วงเวลา P (P คือ ช่วงเวลาที่ต้องการให้เป็นเพราะมันคือช่วงเวลาที่เป็นจริงอยู่ ณ ขณะนั้น) โดยในความจริงแล้ว ช่วงเวลาต่างๆ ในภาษาเอสคิวแอล 2011 จะเป็นไปตามรูปแบบ Closed-open

ดังนั้นช่วงเวลา P จึงนับตั้งแต่ 3 กุมภาพันธ์ 2011 ไปจนถึงก่อนวันที่ 10 กันยายน 2011 แต่ไม่รวม 10 กันยายน 2011 ถ้าแถวใดๆ มีช่วงเวลาที่คาบเกี่ยวกับ P โดยมี Application-time period ของแถวนั้นเป็นส่วนหนึ่งของช่วงเวลา P จะง่ายต่อการปรับปรุงข้อมูล กล่าวคือสามารถปรับปรุงข้อมูลให้เป็นไปตามช่วงเวลาใหม่ได้เลย แต่ถ้าแถวใดๆ มีช่วงเวลาที่คาบเกี่ยวกับ P โดยมีแค่บางส่วนของ Application-time period ของแถวนั้นคาบเกี่ยวกับ P หรือมีช่วงเวลา P เป็นส่วนหนึ่งของ Application-time period ของแถวนั้น จะมีการดำเนินการแตกแถว (Split) แยกออกเป็นสองหรือสามแถวที่อยู่ติดกันขึ้นอยู่กับขอบเขตของการทับซ้อนกันของช่วงเวลาโดยระบบจัดการฐานข้อมูลจะดำเนินการให้อัตโนมัติ

จากตาราง 2.14 จะเห็นว่าแถวที่มี ENo เป็น 22217 นั้นเป็นแถวที่มีเวลาคาบเกี่ยวกับเวลาที่จะปรับปรุง P โดย P เป็นส่วนหนึ่งของเวลาดังกล่าว จะได้ผลลัพธ์ของการปรับปรุงเป็นข้อมูล 3 แถว ดังนี้

ตาราง 2.16 ผลลัพธ์ของการปรับปรุงข้อมูล ในตาราง Application-time period ชื่อ Emp [ที่มา Temporal features in SQL: 2011]

ENO	EStart	EEnd	EDept
22217	2010-01-01	2011-02-03	3
22217	2011-02-03	2011-09-10	4
22217	2011-09-10	2011-11-12	3

จากตัวอย่างข้างต้นนี้แถวที่ค่า EDept ถูกปรับปรุงเป็น 4 เรียกว่า แถวเดิม (Original row) และอีกสองแถวเป็นแถวที่เพิ่มเข้ามาใหม่ เช่นนี้เพราะพนักงานหมายเลข 22217 ได้ย้ายไปอยู่แผนก 4 แค่ช่วงเวลาหนึ่งเท่านั้น ย้ายไปไม่นาน ก็ย้ายกลับมาทำที่เดิม โดยพนักงานคนนี้อยู่แผนกที่ 3 ตั้งแต่วันที่ 1 มกราคม 2010 ถึง 3 กุมภาพันธ์ 2011 จากนั้นเขาก็ได้ย้ายไปทำงานที่แผนกที่ 4 ไม่นาน ตั้งแต่วันที่ 3 กุมภาพันธ์ 2011 ถึง 10 กันยายน 2011 ต่อจากนั้นก็กลับมาทำงานที่แผนกเดิมคือแผนกที่ 3 ตั้งแต่วันที่ 10 กันยายน 2011 ถึง 12 พฤศจิกายน 2011 จะเห็นว่าฐานข้อมูลเชิงเวลามีการจัดการกับช่วงเวลาที่คาบเกี่ยวกันเพื่อให้ผลของการปฏิบัติคำสั่งกับฐานข้อมูลโดยสุดท้ายแล้วนั้นข้อมูลจะสอดคล้องและเป็นจริง

คำสั่ง DELETE มีรูปแบบคำสั่งเพิ่มเติมเช่นเดียวกับคำสั่งในการปรับปรุง คือ FOR PORTION OF รวมทั้งในการลบข้อมูลยังมีการตรวจสอบช่วงเวลาที่คาบเกี่ยวกันเช่นเดียวกันกับการปรับปรุงข้อมูล

ตัวอย่างเช่น คำสั่งในการลบข้อมูลดังต่อไปนี้ ต้องการลบพนักงานที่มีหมายเลข 22217 ช่วงเวลาตั้งแต่ 3 กุมภาพันธ์ 2011 ถึง 10 กันยายน 2011 เนื่องจากพนักงานคนนี้อาจจะลาพักร้อนไม่นานก็กลับมาทำงานที่แผนกเดิมต่อ

```
DELETE Emp
   FOR PORTION OF EPeriod
     FROM DATE '2011-02-03'
     TO DATE '2011-09-10'
WHERE ENo = 22217
```

คล้ายกับตัวอย่างการปรับปรุงข้อมูล โดยแถวใดๆ ที่มี Application-time period อยู่ในช่วงเวลาเดียวกับ P คือตั้งแต่ 3 กุมภาพันธ์ 2011 ถึง 10 กันยายน 2011 จะถูกลบออก ถ้าแถวใดๆมีช่วงเวลาที่คาบเกี่ยวกับ P โดยมีแค่บางส่วนของ Application-time period ของแถวนั้นคาบเกี่ยวกับ P หรือมีช่วงเวลา P เป็นส่วนหนึ่งของ Application-time period ของแถวนั้น จะมีการดำเนินการแตกแถว แยกออกเป็นสองหรือสามแถวติดกันขึ้นอยู่กับขอบเขตของการทับซ้อนของช่วงเวลาและแถวที่มี Application-time period อยู่ใน P จะถูกลบออก

จากตาราง 2.14 จะเห็นว่าแถวที่มี ENo เป็น 22217 นั้นเป็นแถวที่มีเวลาคาบเกี่ยวกับเวลาที่จะลบ P โดย P เป็นส่วนหนึ่งของเวลาดังกล่าว จะได้ผลลัพธ์ของการลบเป็นข้อมูล 2 แถว ดังนี้

ตาราง 2.17 ผลลัพธ์ของการลบข้อมูล ในตาราง Application-time period ชื่อ Emp [ที่มา Temporal features in SQL: 2011]

ENO	EStart	EEnd	EDept
22217	2010-01-01	2011-02-03	3
22217	2011-09-10	2011-11-12	3

จากตัวอย่างข้างต้นนี้เป็นการลบแถวเดิมและจัดเก็บสองแถวใหม่ เนื่องจากเกิดการคาบเกี่ยวกันของ Application-time period กับ ช่วงเวลา P โดยช่วงเวลา P เป็นส่วนหนึ่งของ Application-time period ดังนั้นระบบจัดการฐานข้อมูลจะดำเนินการลบแถวเดิมทิ้ง และจัดเก็บสองแถวใหม่โดยแถวแรกมี EStart เป็นเวลาเริ่มต้นเดิม ส่วน EEnd ของแถวแรกคือเวลาเริ่มต้นของ P จากนั้นแทรกแถวที่สองโดยมี EStart เป็นเวลาสิ้นสุดของ P และ EEnd คือเวลาสิ้นสุดเดิม จะเห็นว่าฐานข้อมูลเชิงเวลามีการจัดการกับช่วงเวลาที่คาบเกี่ยวกันเพื่อให้ผลของการปฏิบัติคำสั่งกับฐานข้อมูลโดยสุดท้ายแล้วนั้นข้อมูลจะสอดคล้องและเป็นจริง

2.3.2.1 Primary keys ของตาราง Application-time period

ถ้าตาราง Emp มี ENo เป็น Primary key อย่างไรก็ตามสังเกตตัวอย่างผลลัพธ์ของคำสั่งที่ใช้ปรับปรุงข้อมูลในตาราง 2.15 ทั้งสามแถวเป็นของ ENo 22217 ทั้งหมด รวมทั้งในตัวอย่างผลลัพธ์ของคำสั่งการลบข้อมูลในตาราง 2.16 ทั้งสองแถวก็เป็นของ ENo 22217 ทั้งหมดเช่นกัน แสดงให้เห็นว่า ENo ไม่สามารถเป็น Primary key ได้ ดังนั้น Primary key จะต้องมีทั้ง 3 แอตทริบิวต์เป็น Primary key ร่วมกันคือ ENo, EStart และ EEnd โดยเพียงแค่เพิ่ม EStart และ EEnd เป็น Primary key ร่วมกับ ENo เป็นสิ่งที่ไม่เพียงพอ พิจารณาข้อมูลต่อไปนี้

ตาราง 2.18 ตาราง Application-time period ชื่อ Emp สำหรับเรื่อง Primary keys [ที่มา Temporal features in SQL: 2011]

ENO	EStart	EEnd	EDept
22217	2010-01-01	2011-09-10	3
22217	2010-02-03	2011-11-12	4

จะเห็นได้ว่า (22217, 2010-01-01, 2011-09-10) และ (22217, 2010-02-03, 2011-11-12) ไม่ซ้ำกัน ดังนั้นจึงยอมรับได้ที่มีทั้ง 3 คอลัมน์ เป็น Primary key ร่วมกัน แต่โปรดสังเกตว่า Application-time period ของแถวเหล่านี้คาบเกี่ยวกัน ซึ่งมีความหมายว่าพนักงานที่มี ENo 22217 ทำงานอยู่ 2 แผนกคือ 3 และ 4 ในระหว่างวันที่ 3 กุมภาพันธ์ 2010 ถึง 10 กันยายน 2011 เป็นไปได้ว่าบางทีผู้ใช้มีความประสงค์ที่จะอนุญาตให้พนักงานสามารถทำงานได้สองแผนกพร้อมกัน แต่อย่างไรก็ตามความต้องการขององค์กรโดยทั่วไป คือการที่พนักงานมีตำแหน่งหน้าที่ได้เพียง 1 แผนกในเวลาใดก็ตาม เพื่อเป็นการห้ามไม่ให้เกิดการคาบเกี่ยวกันของ Application-time period นั้นสามารถระบุด้วยคำสั่งนี้

```
ALTER TABLE Emp
ADD PRIMARY KEY (ENo,
   EPeriod WITHOUT OVERLAPS)
```

2.3.2 Referential constraints ของตาราง Application-time period

กำหนดให้มีตารางดังต่อไปนี้

```
CREATE TABLE Dept(
DNo INTEGER,
DStart DATE,
DEnd DATE,
DName VARCHAR (30),
PERIOD FOR DPeriod (DStart, DEnd),
PRIMARY KEY (DNo,
DPeriod WITHOUT OVERLAPS)
)
```

เป็นกฎข้อบังคับเพื่อให้แน่ใจว่าที่ทุกจุดเวลาอยู่ในช่วงเวลานั้น ทุกค่าในคอลัมน์ EDept สอดคล้องกับบางค่าของคอลัมน์ DNo ในตาราง Dept กล่าวคือ ช่วงเวลาที่พนักงานทำงานในแผนกนั้นๆจะต้องมีช่วงเวลาที่สอดคล้องกับช่วงเวลาของ DNo ในตาราง Dept ด้วย สมมติตาราง Emp มีแถวข้อมูลดังต่อไปนี้

ตาราง 2.19 ตาราง Application-time period ชื่อ Emp สำหรับเรื่อง Referential constraints [ที่มา Temporal features in SQL: 2011]

ENO	EStart	EEnd	EDept
22217	2010-01-01	2011-02-03	3
22217	2010-02-03	2011-11-12	4

สมมติตาราง Dept มีแถวข้อมูลดังต่อไปนี้

ตาราง 2.20 ตัวอย่างตาราง Dept [ที่มา Temporal features in SQL: 2011]

DNO	DStart	DEnd	DName
3	2009-01-01	2011-12-31	Test
4	2011-06-01	2011-12-31	QA

สังเกตค่าของคอลัมน์ EDept ของตาราง Emp และคอลัมน์ DNo ของตาราง Dept เราอาจสรุปได้ว่า Referential integrity constraint ในสองตารางนี้มีความถูกต้อง แต่โปรดสังเกตว่าพนักงานที่มี ENo 22218 ได้รับมอบหมายอยู่หน่วยงานที่มี DNO 4 ตั้งแต่ 3 กุมภาพันธ์ 2011 ถึง 12 พฤศจิกายน 2011 แต่ไม่มีแผนกที่มี DNO 4 ในช่วงเวลาตั้งแต่ 3 กุมภาพันธ์ 2011 ถึง 1 มิถุนายน 2011 เห็นได้ชัดว่าเป็นการฝ่าฝืนความต้องการขององค์กรที่ว่าทุกค่าของคอลัมน์ EDept ในตาราง Emp จะสอดคล้องกับบางค่าของคอลัมน์ DNO ในตาราง Dept ในทุกช่วงเวลา ซึ่งจะไม่อนุญาตให้สถานการณ์ดังกล่าวเกิดขึ้น ดังนั้นจึงต้องห้ามให้มีแถวในตารางลูกที่มี Application-time period

ไม่ได้เป็นส่วนหนึ่งใน Application-time period ของแถวที่สอดคล้องกัน (Match) ในตารางแม่ ซึ่งสามารถระบุด้วยคำสั่งดังนี้

```
ALTER TABLE Emp
ADD FOREIGN KEY
      (Edept, PERIOD EPeriod)
      REFERENCES Dept
       (DNo, PERIOD DPeriod)
```

โดยปกติสำหรับแถวลูกในตารางลูกนั้น ไม่จำเป็นว่าแถวที่สอดคล้องซึ่งอยู่ในตารางแม่จะต้องมี Application-time period เป็นส่วนหนึ่งใน Application-time period ของแถวลูกนั้นเสมอไป ตราบใดที่ Application-time period ของแถวในตารางลูกอยู่ใน Application-time period ที่ยูเนียนกันโดยแถวที่ติดกันของตารางแม่ ซึ่งก็ยังเป็นไปตาม Referential constraint

2.3.2.3 การสอบถามข้อมูลในตาราง Application-time period

ในภาษาเอสคิวแอล 2011 นั้น ตาราง Application-time period สามารถถูกสอบถาม โดยใช้คำสั่งการสอบถามข้อมูลปกติ ตัวอย่างเช่น ต้องการดูข้อมูลแผนกของพนักงานหมายเลข 22217 ที่ทำงานในช่วงวันที่ 2 มกราคม 2011 สามารถสอบถาม ได้ดังนี้

```
SELECT Name, Edept
FROM Emp
WHERE ENo = 22217
 AND EStart <= DATE '2011-01-02'
 AND EEnd > DATE '2011-01-02'
```

มีวิธีที่ง่ายกว่าเดิมในการสอบถามเช่นเดียวกับตัวอย่างข้างต้น โดยการใช้คำสั่งปฏิบัติการที่มีเงื่อนไขที่เกี่ยวข้องกับช่วงเวลา ซึ่งมีให้ในภาษาเอสคิวแอล 2011 ดังนี้ CONTAINS, OVERLAPS , EQUALS , PRECEDES , SUCCEEDS, IMMEDIATELY PRECEDES , และ IMMEDIATELY SUCCEEDS

ตัวอย่างเช่น การสอบถามโดยใช้ CONTAINS ดังนี้

```
SELECT Ename, Edept
FROM Emp
WHERE ENo = 22217 AND
  EPeriod CONTAINS DATE '2011-01-02'
```

ถ้าอยากจะทราบว่าทุกหน่วยงานที่พนักงานหมายเลข 22217 ทำงานในช่วงระยะเวลาตั้งแต่ 1 มกราคม 2010 ถึง 1 มกราคม 2011 กำหนดคำสั่งในการสอบถามเป็นดังนี้

```
SELECT Ename, Edept
FROM Emp
WHERE ENo = 22217
      AND EStart < DATE '2011-01-01'
      AND EEnd > DATE '2010-01-01'
```

โปรดทราบว่าช่วงเวลาที่กำหนดในการสอบถามข้างต้นจะใช้รูปแบบ Closed-open กล่าวคือ ช่วงระยะเวลานั้นจะรวม 1 มกราคม 2010 แต่ไม่รวมวันที่ 1 มกราคม 2011 อีกหนึ่งวิธีหนึ่งในการสอบถามแบบเดียวกันโดยใช้ OVERLAPS เป็นดังนี้

```
SELECT Ename, Edept
FROM Emp
WHERE ENo = 22217 AND
      EPeriod OVERLAPS
            PERIOD (DATE '2010-01-01',
                    DATE '2011-01-01')
```

รูปแบบการนำเสนอเวลาดังตัวอย่างข้างต้นนั้นมีรูปแบบคล้ายคลึงกับ Allen's interval operators โดยการแสดงช่วงเวลาของภาษาเอสคิวแอลและ Allen's operator มีความสอดคล้องกันดังนี้

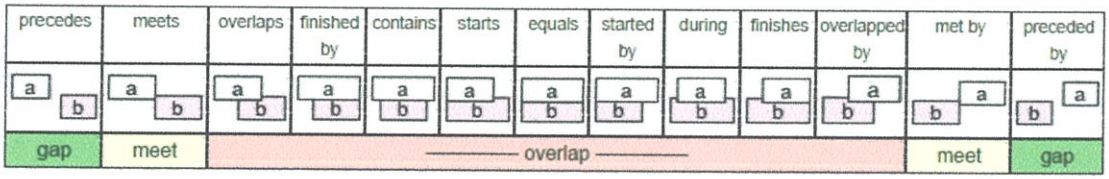
- "X OVERLAPS Y" ในภาษาเอสคิวแอล 2011 มีค่าเท่ากับนิพจน์บูลีนของ Allen's operators ดังนี้ "(X OVERLAPS Y) OR (X OVERLAPPED_BY Y) OR (X DURING Y) OR (X CONTAINS Y) OR (X STARTS Y) OR (X STARTED_BY Y) OR (X FINISHES Y) OR (X FINISHED_BY Y) OR (X EQUAL Y)"

  ข้อสังเกต: OVERLAPS ของ Allen's operators ไม่เหมาะนำมาใช้กับช่วงเวลาที่ทับซ้อนกัน โดยปกติ 2 ช่วงเวลาจะถือว่าทับซ้อนกัน ถ้า 2 ช่วงเวลานั้นมีจุดเวลาอย่างน้อยหนึ่งจุดที่เหมือนกัน แต่ใน Allen's overlaps operator ไม่ใช่ ในทางตรงกันข้าม OVERLAPS ในภาษาเอสคิวแอล 2011 เหมาะนำมาใช้กับช่วงเวลาที่ทับซ้อนกันมากกว่า โดยถ้า "X OVERLAPS Y" เป็นจริง แล้ว "Y OVERLAPS X" ก็จะเป็นจริงด้วย แต่ไม่เป็นจริงในกรณีที่ใช้ Allen's overlaps operator

- "X CONTAINS Y" ในภาษาเอสคิวแอล 2011 มีค่าเท่ากับนิพจน์บูลีนของ Allen's operators ดังนี้ "(X CONTAINS Y) OR (X STARTS Y) OR (X FINISHES Y) OR (X EQUAL Y)"

ข้อสังเกต: โดยปกติช่วงเวลา X ถูกพิจารณาว่ามีช่วงเวลา Y เป็นส่วนหนึ่งของช่วงเวลา X ถ้าทุกจุดเวลาใน Y อยู่ใน X สำหรับ CONTAINS ของ Allen's operators จะถือว่าไม่จริง ในทางตรงกันข้าม CONTAINS ในภาษาเอสคิวแอล 2011 เป็นคำสั่งที่เหมาะใช้กับช่วงเวลาที่อยู่ในอีกช่วงเวลาหนึ่งมากกว่า

- "X PRECEDES Y" ในภาษาเอสคิวแอล 2011 มีค่าเท่ากับนิพจน์บูลีนของ Allen's operators ดังนี้ "(X BEFORE Y) OR (X MEETS Y)".
- "X SUCCEEDS Y" ในภาษาเอสคิวแอล 2011 มีค่าเท่ากับนิพจน์บูลีนของ Allen's operators ดังนี้ "(X AFTER Y) OR (X MET_BY Y)".
- "X EQUALS Y", "X IMMEDI- ATELY PRECEDES Y", และ "X IMMEDIATELY SUCCEEDS Y" ในภาษาเอสคิวแอล 2011 มีค่าเท่ากับนิพจน์บูลีนของ Allen's operators ดังนี้ "X EQUAL Y", "X MEETS Y", และ "X MET_BY Y" ตามลำดับ



รูป 2.7 Allen's interval algebra (ที่มา SQL for date ranges, gaps and overlaps)

2.3.3 ตาราง System-versioned

ตาราง System-versioned มีวัตถุประสงค์เพื่อสนองความต้องการของโปรแกรมประยุกต์ที่ต้องการรักษาประวัติในการเปลี่ยนแปลงข้อมูล

ความต้องการของแต่ละโปรแกรมประยุกต์คือการปรับปรุงข้อมูลหรือลบแถวข้อมูลนั้นจะต้องมีการรักษาสถานะเก่าของ Fact ก่อนที่จะดำเนินการปรับปรุงข้อมูลหรือลบแถวข้อมูลนั้นไป ความต้องการอีกประการหนึ่งที่สำคัญคือความต้องการระบบเพื่อใช้ในการเก็บเวลาเริ่มต้นและสิ้นสุดของช่วงเวลาในแต่ละแถวข้อมูล ซึ่งผู้ใช้ฐานข้อมูลจะไม่สามารถแก้ไขเนื้อหาของแถวข้อมูลในอดีต (Historical rows) หรือช่วงเวลาที่เกี่ยวข้องกับแถวข้อมูลใดๆ ได้ การปรับปรุงช่วงเวลาของแถวข้อมูลในตาราง System-versioned จะสามารถทำได้โดยระบบเท่านั้น

ทุกตารางจะใส่นิยามตารางที่เป็นชื่อมาตรฐานซึ่งก็คือ SYSTEM_TIME ประกอบกับคำสำคัญ `WITH SYSTEM VERSIONING` ซึ่งเป็นสิ่งที่บ่งบอกว่ามันคือตาราง System-versioned สิ่งที่คล้ายกับตาราง Application-time period คือผู้ใช้สามารถตั้งชื่อคอลัมน์เวลาเริ่มต้นและเวลาสิ้นสุดของช่วงเวลา SYSTEM_TIME ได้ตามต้องการ ในคอลัมน์เวลาเริ่มต้นและสิ้นสุดสามารถใช้ชนิดข้อมูล

DATE หรือ Timestamp ก็ได้ แต่ชนิดข้อมูลของคอลัมน์ทั้งสองคอลัมน์ต้องเหมือนกัน แต่ในทางปฏิบัติการใช้งานส่วนใหญ่จะใช้ Timestamp เนื่องจากสามารถเก็บข้อมูลเวลาและวันที่ได้ ทำให้มีความแม่นยำในการเก็บช่วงเวลามากยิ่งขึ้น

ตัวอย่างเช่น

```
CREATE TABLE Emp
ENo INTEGER,
Sys_start TIMESTAMP(12) GENERATED
 ALWAYS AS ROW START,
Sys_end TIMESTAMP(12) GENERATED
 ALWAYS AS ROW END,
EName VARCHAR(30),
PERIOD FOR SYSTEM_TIME (Sys_start,
 Sys_end)
) WITH SYSTEM VERSIONING
```

System-time period ใช้ช่วงเวลารูปแบบ Closed-open period เช่นเดียวกันกับ Application-time period ที่จุดเวลาใดก็ตาม ถ้า System-time period ของแถวข้อมูลใดมีเวลาปัจจุบันอยู่ แถวข้อมูลในตาราง System-versioned นั้นจะเรียกว่าเป็นแถวของระบบที่เป็นปัจจุบัน (Current system row) ส่วนแถวข้อมูลที่ไม่ใช่แถวของระบบที่เป็นปัจจุบันจะเรียกว่าเป็นแถวของระบบในอดีต (Historical system row)

ตาราง System-versioned แตกต่างจากตาราง Application-time ดังนี้

1) ผู้ใช้จะไม่ได้รับอนุญาตในการกำหนดหรือเปลี่ยนแปลงค่าของคอลัมน์ SYS_START หรือ SYS_END ในตาราง System-versioned ซึ่งตรงกันข้ามกับตาราง Application-time โดยค่าดังกล่าวจะถูกกำหนดและเปลี่ยนแปลงอัตโนมัติโดยระบบจัดการฐานข้อมูล จึงกลายเป็นเหตุผลที่ว่าจะต้องมีคำสำคัญ GENERATED ALWAYS AS ROW START และ GENERATED ALWAYS AS ROW END ในนิยามของคอลัมน์ SYS_START หรือ SYS_END

2) การเพิ่มข้อมูลในตาราง System-versioned เกิดขึ้นโดยอัตโนมัติ ซึ่งจะมีการตั้งค่าของคอลัมน์ SYS_START เป็น Transaction timestamp กล่าวคือเป็นค่าพิเศษที่เกี่ยวข้องกับทุก Transaction และกำหนดค่าคอลัมน์ SYS_END ให้เป็นค่าสูงสุดของชนิดข้อมูลของคอลัมน์

ตัวอย่างเช่น สมมติว่า ใช้คำสั่ง INSERT ต่อไปนี้ดำเนินการใน Transaction ที่มี Transaction timestamp เป็น 2012-01-01 09:00:00

```
INSERT INTO Emp (ENo, EName)
    VALUES (22217, 'Joe')
```

ตารางผลลัพธ์ที่ได้มีลักษณะดังต่อไปนี้ (โดยกำหนดว่าตารางเคยว่างมาก่อนหน้านี้)

ตาราง 2.21 ผลลัพธ์จากการ Insert ในตาราง System-versioned ชื่อ Emp
[ที่มา Temporal features in SQL: 2011]

ENO	SYS_START	SYS_END	EName
22217	2012-01-01 09:00:00	9999-12-31 23:59:59	Joe

3) การปรับปรุงข้อมูลและลบข้อมูลในตาราง System-versioned จะปฏิบัติแถวของระบบที่เป็นปัจจุบันเท่านั้น ผู้ใช้จะไม่ได้รับอนุญาตในการปรับปรุงหรือลบแถวของระบบในอดีต รวมถึงผู้ใช้จะยังไม่ได้รับอนุญาตในการปรับเปลี่ยนเวลาเริ่มต้นหรือเวลาสิ้นสุดของทั้งแถวของระบบที่เป็นปัจจุบันและแถวของระบบในอดีต

4) การปรับปรุงข้อมูลและลบข้อมูลในตาราง System-versioned ให้ผลลัพธ์เป็นการจัดเก็บแถวของระบบในอดีตสำหรับทุกๆแถวของแถวของระบบที่เป็นปัจจุบันที่ถูกปรับปรุงหรือถูกลบไปโดยอัตโนมัติ

คำสั่งการปรับปรุงข้อมูลในตาราง System-versioned คือการเพิ่มสำเนาของแถวข้อมูลเก่าพร้อมระบุเวลาสิ้นสุดของ Fact นั้นลงไปด้วย ซึ่งเป็น Transaction timestamp จากนั้นปรับปรุงแถวโดยเปลี่ยนเวลาเริ่มต้นของระบบเป็น Transaction timestamp แสดงให้เห็นว่าแถวข้อมูลนั้นถูกปรับปรุงให้เป็นแถวของระบบที่เป็นปัจจุบันที่เวลา Transaction timestamp ตัวอย่างเช่น สมมติ แถวของระบบที่เป็นปัจจุบันของ ENo 22217 เป็นไปตามตาราง 2.20

ใช้คำสั่งการปรับปรุงข้อมูลต่อไปนี้เปลี่ยนชื่อของพนักงานที่มีหมายเลข 22217 จาก Joe เป็น Tom ดังนี้

```
UPDATE Emp
SET EName = 'Tom'
WHERE ENo = 22217
```

แถวของระบบในอดีตที่เป็นสถานะของแถวก่อนที่จะมีการปรับปรุงนั้น จะถูกจัดเก็บก่อนแล้วจึงค่อยปรับปรุง สมมติว่าคำสั่งดังกล่าวดำเนินการใน Transaction ที่มี Transaction timestamp เป็น 2012-02-03 10:00:00 ผลลัพธ์สุดท้ายจะได้แถวข้อมูล 2 แถว ดังนี้

ตาราง 2.22 ผลลัพธ์จากการปรับปรุง ในตาราง System-versioned ชื่อ Emp
[ที่มา Temporal features in SQL: 2011]

ENO	SYS_START	SYS_END	EName
22217	2012-01-01 09:00:00	2012-02-03 10:00:00	Joe
22217	2012-02-03 10:00:00	9999-12-31 23:59:59	Tom

ในตัวอย่างข้างต้นนี้ แถวของ Tom เป็นแถวที่ปรับปรุงใหม่และแถวของระบบในอดีตคือแถวของ Joe ที่เป็นสถานะก่อนถูกปรับปรุง ซึ่ง SYS_END ของ Joe จะกลายเป็น Transaction timestamp ซึ่งเป็นเวลาที่เริ่มปฏิบัติ Transaction เพื่อปรับปรุงข้อมูล SYS_START ของ Tom ก็คือ Transaction timestamp ซึ่งเป็นเวลาที่เริ่มปฏิบัติ Transaction เพื่อปรับปรุงข้อมูล และ SYS_END ของ Tom ก็คือเวลา ณ ปัจจุบัน สำหรับช่วงเวลาของแถวข้อมูลดังกล่าวจะไม่มีช่องว่างระหว่าง System-time periods

สำหรับคำสั่งการลบข้อมูลบนตาราง System- versioned ไม่ได้เป็นการไม่ลบแถวข้อมูลทิ้งไป แต่จะเป็นการเปลี่ยนแปลงเวลาสิ้นสุดของแถวเหล่านั้นให้เป็น Transaction timestamp แสดงให้เห็นว่าแถวนั้นสิ้นสุดความเป็นจริง ซึ่งเคยเป็นจริงในอดีต ตัวอย่างเช่น สมมติแถวของระบบที่เป็นปัจจุบันของ ENo 22217 เป็นไปตามตาราง 2.20

คำสั่งการลบข้อมูลเพียงแค่เปลี่ยนแปลง System-time period ที่เวลาสิ้นสุดในแถวของระบบที่เป็นปัจจุบันของพนักงานหมายเลข 22217 เป็น Transaction timestamp คำสั่งการลบข้อมูลดำเนินการดังนี้

```
DELETE FROM Emp
WHERE ENo = 22217
```

สมมติว่าคำสั่งการลบข้อมูลดังกล่าวดำเนินการใน Transaction ที่มี Transaction timestamp เป็น 2012-06-01 00:00:00 ผลลัพธ์สุดท้ายจะได้

ตาราง 2.23 ผลลัพธ์จากการ Delete ในตาราง System-versioned ชื่อ Emp [ที่มา Temporal features in SQL: 2011]

ENO	SYS_START	SYS_END	EName
22217	2012-01-01 09:00:00	9999-06-01 00:00:00	Joe

ในตาราง system-versioned นั้นไม่จำต้องใช้คำสั่ง FOR PORTION OF SYSTEM_TIME สำหรับคำสั่งในการปรับปรุงข้อมูลและการลบข้อมูล

2.3.3.1 Primary key และ Referential constraints ในตาราง system-versioned

การบังคับใช้ของ Constraint บนตาราง System-versioned ง่ายกว่าการบังคับใช้ Constraint ในตาราง Application-time period เป็นเพราะ Constraint ในตาราง System-versioned บังคับใช้เฉพาะแถวของระบบที่เป็นปัจจุบัน ซึ่งแถวของระบบในอดีตของตาราง System-versioned จะไม่เปลี่ยนแปลงทำให้เราสามารถดูอดีตที่ผ่านมาได้

Constraint ที่มีผลบังคับ เมื่อแถวของระบบในอดีตถูกสร้างขึ้นนั้นจะได้รับการตรวจสอบแล้วเมื่อแถวนั้นเป็นแถวของระบบที่เป็นปัจจุบัน ไม่จำเป็นต้องบังคับใช้ Constraint ในแถวของระบบในอดีตใดๆ เพราะฉะนั้นจึงไม่มีความจำเป็นที่จะต้องรวมคอลัมน์เวลาเริ่มต้นและสิ้นสุดหรือชื่อของช่วงเวลาตามความหมายของ Primary key และ Referential constraint ในตาราง System-versioned

ตัวอย่าง ใช้คำสั่งการแก้ไขตาราง ระบุให้คอลัมน์ ENo เป็น Primary key ของตาราง Emp

```
ALTER TABLE Emp
ADD PRIMARY KEY (ENo)
```

Constraint ดังกล่าวข้างต้นมั่นใจว่ามีอยู่หนึ่ง แถวของระบบที่เป็นปัจจุบันมีค่า ENo เช่นเดียวกันกับใช้ ALTER TABLE ระบุ Referential constraint ระหว่าง Emp และ Dept

```
ALTER TABLE Emp
ADD FOREIGN KEY (Edept)
   REFERENCES Dept (DNo)
```

Constraint ข้างต้น ถูกบังคับใช้เฉพาะแถวของระบบที่เป็นปัจจุบันของตาราง Emp และ Dept

2.3.3.2 การสอบถามข้อมูลของตาราง System-versioned

เนื่องจากตาราง System-versioned มีวัตถุประสงค์หลักสำหรับการติดตามการเปลี่ยนแปลงประวัติข้อมูล การสอบถามในตาราง System-versioned มักจะมีแนวโน้มเกี่ยวข้องกับการกู้เนื้อหาของตาราง ณ จุดเวลาหนึ่งหรือระหว่างสองจุดเวลา ในภาษาเอสคิวแอล 2011 มี 3 คำสั่ง เพื่อใช้ในการสอบถามข้อมูล คำสั่งเหล่านี้จะได้รับอนุญาตในการสอบถามบนตาราง System-versioned เท่านั้น

คำสั่งแรกคือ `FOR SYSTEM_TIME AS OF` ใช้สำหรับการสอบถามข้อมูลในตาราง ณ จุดเวลาหนึ่งๆ ตัวอย่างเช่น การสอบถามแถวข้อมูลของ Emp ที่เวลาปัจจุบันคือวันที่ 2 มกราคม 2011 ใช้คำสั่งดังนี้

```
SELECT ENo,EName,Sys_Start,Sys_End
FROM Emp FOR SYSTEM_TIME AS OF
TIMESTAMP '2011-01-02 00:00:00'
```

การสอบถามข้อมูลดังกล่าวจะให้ผลลัพธ์เป็นแถวทั้งหมดที่เวลาเริ่มต้นน้อยกว่าหรือเท่ากับ Timestamp ที่กำหนดไว้และเวลาสิ้นสุดมากกว่าเท่ากับ Timestamp ที่กำหนดไว้

คำสั่งที่สองและสามสำหรับการสอบถามเนื้อหาของตาราง System-versioned tables ระหว่างเวลาสองจุดใดๆ การสอบถามต่อไปนี้จะให้ผลลัพธ์เป็นแถวทั้งหมดที่เริ่มต้นจาก TIMESTAMP '2011-01- 02 00:00:00' ไปจนถึง TIMESTAMP '2011-12-31 00:00:00' แต่ไม่นับรวม TIMESTAMP '2011-12-31 00:00:00' ใช้คำสั่งดังนี้

```
SELECT ENo,EName,Sys_Start,Sys_End
FROM Emp FOR SYSTEM_TIME FROM
  TIMESTAMP '2011-01-02 00:00:00'TO
  TIMESTAMP '2011-12-31 00:00:00'
```

ในทางตรงกันข้าม การสอบถามต่อไปนี้จะให้ผลลัพธ์เป็นแถวทั้งหมดที่เริ่มต้นจาก TIMESTAMP '2011-01- 02 00:00:00' ไปจนถึง TIMESTAMP '2011-12-31 00:00:00' ซึ่งจะนับรวม TIMESTAMP '2011-12-31 00:00:00' ด้วย ใช้คำสั่งดังต่อไปนี้

```
SELECT ENo,EName,Sys_Start,Sys_End
FROM Emp FOR SYSTEM_TIME BETWEEN
 TIMESTAMP '2011-01-02 00:00:00'AND
 TIMESTAMP '2011-12-31 00:00:00'
```

จากข้างต้นจะสังเกตเห็นว่า ช่วงเวลาที่ระบุด้วย (FROM ... TO ...) สอดคล้องกับช่วงเวลารูปแบบ Closed-open ในขณะที่ช่วงเวลาที่ระบุด้วย (BETWEEN ... AND ...) สอดคล้องกับรูปแบบ Closed-closed

ถ้าการสอบถามข้อมูลในตาราง System-versioned ไม่ได้ระบุสามคำสั่งดังกล่าว แล้วการสอบถามข้อมูลนั้นจะเปรียบเสมือนว่าระบุด้วยคำสั่ง FOR SYSTEM_TIME AS OF CURRENT_TIMESTAMP ซึ่งเป็นค่าเริ่มต้น โดยการสอบถามข้อมูลจะให้ผลลัพธ์เฉพาะแถวที่จัดเป็นแถวปัจจุบันเท่านั้น ตัวอย่างเช่น การสอบถามข้อมูลต่อไปนี้จะให้ผลลัพธ์เฉพาะแถวของระบบที่เป็นปัจจุบันในตาราง Emp

```
SELECT ENo,EName,Sys_Start,Sys_End
FROM Emp
```

2.3.4 ตาราง Bitemporal

เป็นตารางที่ประกอบด้วยตาราง System-versioned และตาราง Application-time period เช่น ตัวอย่างดังนี้

```
CREATE TABLE Emp(
ENo INTEGER,
EStart DATE,
EEnd DATE,
EDept INTEGER,
PERIOD FOR EPeriod (EStart, EEnd),
Sys_start TIMESTAMP(12)GENERATED
   ALWAYS AS ROW START,
Sys_end TIMESTAMP(12)GENERATED
   ALWAYS AS ROW END,
EName VARCHAR(30),
PERIOD FOR SYSTEM_TIME
   (Sys_start, Sys_end),
PRIMARY KEY (ENo,
   EPeriod WITHOUT OVERLAPS),
FOREIGN KEY
   (Edept, PERIOD EPeriod)
 REFERENCES Dept
   (DNo, PERIOD DPeriod)
) WITH SYSTEM เวอร์ชันING
```

แต่ละแถวในตารางดังกล่าวจะเกี่ยวข้องกับทั้ง System-time period และ Application-time period สามารถใช้ดูช่วงเวลาที่ Facts เป็นจริง และช่วงเวลาที่ Facts เหล่านั้นถูกบันทึกไว้ในฐานข้อมูล ตัวอย่างเช่น ในขณะการจ้างงาน พนักงานอาจมีการเปลี่ยนแปลงชื่อ โดยทั่วไปแล้วชื่อจะ

เปลี่ยนแปลงอย่างถูกต้องตามกฎหมายในเวลาที่กำหนด (เช่น การแต่งงาน) แต่ชื่อจะไม่ถูกเปลี่ยนแปลงในฐานข้อมูลพร้อมกับการเปลี่ยนแปลงตามกฎหมาย ในกรณีนี้ตาราง System-time period จะบันทึกโดยอัตโนมัติ เมื่อชื่อเฉพาะอยู่ในฐานข้อมูลและ Application-time period จะบันทึกเมื่อชื่อมีผลถูกต้องตามกฎหมาย

ตาราง Bitemporal รวมความสามารถของทั้งสองตาราง เช่น ในตาราง Application-time period ผู้ใช้สามารถจัดการกับคอลัมน์เวลาเริ่มต้นและเวลาสิ้นสุดได้ ส่วนตาราง System-versioned นั้น การจัดเก็บข้อมูลลงในตารางดังกล่าวจะทำโดยอัตโนมัติด้วยการกำหนดค่าของคอลัมน์เริ่มต้นเป็น Transaction timestamp และค่าของคอลัมน์เวลาสิ้นสุดเป็นค่าสูงสุดของชนิดข้อมูลสำหรับคอลัมน์นี้

ในกรณีของตาราง Application-time period คำสั่งในการปรับปรุงข้อมูลใช้กับ FOR PORTION OF app-period โดย app-period คือชื่อของ Application-time period สามารถแก้ไขแถวข้อมูลของตาราง Bitemporal ได้ ในทำนองเดียวกัน คำสั่งการลบข้อมูลใช้กับ FOR PORTION OF app-period ใช้ลบแถวข้อมูลออกจากตาราง Bitemporal ส่วนในกรณีของตาราง System-versioned นั้น สามารถปรับปรุงหรือลบข้อมูลกับแถวปัจจุบันเท่านั้นในระบบ และแถวของระบบในอดีตจะถูกเพิ่มโดยอัตโนมัติ เมื่อทุกๆ แถวของระบบที่เป็นปัจจุบันนั้นถูกปรับปรุงหรือถูกลบข้อมูล

การสอบถามในตาราง Bitemporal ใช้ได้ทั้งสองตาราง เพื่อหาแถวข้อมูลที่จะถูกส่งเป็นผลลัพธ์ของการสอบถาม ตัวอย่างเช่น การสอบถามข้างล่างนี้จะให้ผลลัพธ์แผนกที่พนักงานหมายเลข 22217 ทำงานในช่วง 1 ธันวาคม 2010 บันทึกในฐานข้อมูล วันที่ 1 กรกฎาคม 2011 [2]

```
SELECT ENo, EDept
FROM Emp FOR SYSTEM_TIME AS OF
       TIMESTAMP '2011-07-01 00:00:00'
WHERE ENo = 22217 AND
  EPeriod CONTAINS DATE '2010-12-01'
```

2.4 เทคโนโลยีของ Oracle Flashback (Oracle Flashback Technology)

2.4.1 แนวคิดของเทคโนโลยี Oracle Flashback

Oracle Flashback มีแนวคิดหลักในการทำงานคือความต้องการที่จะลดเวลาในการกู้คืนข้อมูล (Recovery) ลงซึ่งเวลาในการกู้คืนข้อมูลนั้นไม่ได้หมายความว่าเป็นการกู้คืนข้อมูลที่เกิดจากความขัดข้องของระบบ (system failure) แต่อย่างใด แต่มุ่งเน้นไปที่การกู้คืนข้อมูลที่เป็นผลมาจากความไม่ตั้งใจของผู้ดูแลระบบ เช่น บังเอิญลบบางแถวข้อมูลของตารางโดยไม่ตั้งใจหรือแม้กระทั่งบังเอิญลบตารางทิ้งไป โดยให้สามารถกู้คืนข้อมูลเหล่านั้นกลับมาได้โดยง่าย ซึ่งเทคโนโลยีที่พบใน flashback นั้นแบ่งการใช้งานออกเป็น 2 ส่วน คือเทคโนโลยีในด้านการพัฒนาโปรแกรมประยุกต์ (Application development features) และเทคโนโลยีช่วยผู้ดูแลระบบฐานข้อมูล (Database Administration Features)

1) เทคโนโลยีช่วยผู้ดูแลระบบฐานข้อมูล ส่วนใหญ่จะเป็นรูปแบบพิเศษสำหรับการกู้คืนข้อมูล โดยปกติแล้วผู้ดูแลระบบฐานข้อมูลจะเป็นผู้ใช้งานในส่วนนี้ ซึ่งจะประกอบไปด้วย

- Flashback Database เป็นการกู้คืนข้อมูลของฐานข้อมูลทั้งหมดไปยังจุดจุดหนึ่งที่เคยมีอยู่ในอดีต โดยอาศัย Log file ที่มีชื่อว่า Oracle-Optimized Flashback Logs
- Flashback Table เป็นการลดระดับการกู้คืนข้อมูลจากการกู้คืนระดับทั้งฐานข้อมูล เหลือเพียงการกู้คืนระดับตาราง
- Flashback Drop เป็นการกู้คืนเหตุการณ์ที่ผู้ดูแลระบบบังเอิญไปทำการลบตารางทิ้ง นอกจาก flashback drop จะสามารถกู้คืนแถวข้อมูลในตารางได้แล้ว ยังสามารถกู้คืนข้อมูลสำคัญเกี่ยวกับตารางเช่น Index, Constrains ต่างๆมาให้อีกด้วย
- Flashback Transaction เป็นการยกเลิก (Undo) ผลที่เกิดจาก Transaction โดยจะต้องใช้งานร่วมกับ PL/SQL operation หรือ Enterprise Manager Wizard ของ Oracle

2) เทคโนโลยีในด้านการพัฒนาโปรแกรมประยุกต์ เป็นส่วนที่จะถูกใช้งานโดยผู้ออกแบบโปรแกรมประยุกต์ ซึ่งจะมีรูปแบบต่างๆดังนี้

- Flashback Transaction Query เป็นการเรียกดูทุกๆการเปลี่ยนแปลงที่เกิดขึ้นจากผลการทำงานของ Transaction โดยสามารถระบุ Transaction ID ในการเรียกดูผลการเปลี่ยนแปลงเฉพาะของแต่ละ Transaction ได้ ซึ่งจะมีประโยชน์มากหากข้อผิดพลาดที่เกิดขึ้นจาก Transaction นั้นมีผลกระทบต่อหลายแถวข้อมูลหรือหลายตาราง

- Flashback Query เป็นกลไกลที่สำคัญในการสอบถามข้อมูลที่เคยมีอยู่จริงในอดีตโดยสามารถไปค้นคืนได้ทั้งจาก Undo Log File หรือ Flashback Data Archive ก็ได้
- Flashback versions Query เป็นการสอบถามประวัติทุกอย่างที่เคยปฏิบัติกับแถวข้อมูลนั้นๆ ไม่ว่าจะเป็นการจัดเก็บข้อมูล การปรับปรุงข้อมูลและการลบข้อมูล
- Flashback Feature นั้นเสนอความสามารถในการสอบถามข้อมูลในอดีต (Historical data) ซึ่งความสามารถในส่วนนี้ทำให้เราสามารถแก้ไขการเปลี่ยนแปลงในปัจจุบันให้กลับไปยังอดีตได้

2.4.2 Tablespace

ใน Oracle database นั้นจะประกอบด้วย Tablespace ต่างๆมากมาย ซึ่ง Tablespace นี้จะแบ่งเป็น 2 ลักษณะคือ ส่วนหนึ่งจะสร้างพร้อมกับการสร้างฐานข้อมูล ซึ่งจะจัดการให้อัตโนมัติโดย Database Configuration Assistant และอีกลักษณะคือผู้ดูแลระบบเป็นผู้สร้างขึ้นเอง โดย Tablespace ที่สร้างโดย Database Configuration Assistant นั้นมีดังนี้

- SYSTEM เป็นพื้นที่สำหรับใช้สำหรับเก็บข้อมูลเกี่ยวกับพจนานุกรมข้อมูล (Data dictionary)
- SYSAUX เป็นองค์ประกอบของ Optional database
- USERS tablespace หรือ Permanent tablespaces ใช้สำหรับเก็บตารางต่างๆที่ผู้ใช้สร้างขึ้น
- Undo tablespace ใช้สำหรับเก็บ “Before image” ซึ่งเป็นพื้นที่ที่ใช้เก็บข้อมูลเก่าหลังจากปฏิบัติ Transaction
- Temporary tablespace ใช้สำหรับเก็บข้อมูลที่มีช่วงอายุ (Lifespan) ที่สั้น หรือเป็นข้อมูลที่ใช้ชั่วคราว

Tablespace นั้นสามารถสร้างด้วยคำสั่ง DDL ดังนี้

1) User tablespace

```
CREATE BIGFILE TABLESPACE TEST_FLASHBACK
DATAFILE ‘test_flashback.dbf’
SIZE 20M AUTOEXTEND ON;
```

2) Undo tablespace

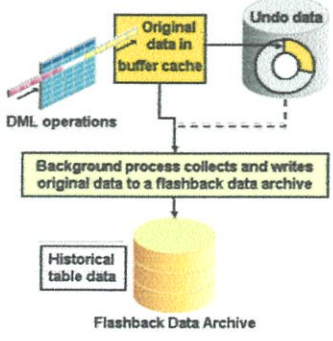
```
CREATE UNDO TABLESPACE TEST_UNDOTABLESPACE
DATAFILE 'test_undotablespace.dbf'
SIZE 10M AUTOEXTEND ON
```

3) Temporary tablespace

```
CREATE TEMPORARY TABLESPACE TEMP_TABLESPACE
TEMPFILE 'temp_tablespace.dbf'
SIZE 5M AUTOEXTEND ON;
RETENTION GUARANTEE;
```

2.4.3 Flashback Data Archive

Flashback Data Archive นั้นเป็นกลไกลที่ Oracle ใช้ในการติดตามการเปลี่ยนแปลงของตาราง เพื่อให้ผู้ดูและระบบสามารถสอบถามข้อมูลที่เป็นอดีตของตารางนั้นๆได้โดยหากตารางใดๆก็ตาม ได้มีการเปิดการทำงาน (Enable) ของ Flashback Data Archive เอาไว้ที่ตารางนั้นๆ เมื่อข้อมูลในตารางมีการเปลี่ยนแปลงไปไม่ว่าจะเป็นการจัดเก็บข้อมูล การปรับปรุงข้อมูลและการลบข้อมูล นอกจากการที่ระบบจัดการฐานข้อมูลจะนำค่าเก่าไปจัดเก็บไว้ใน Undo tablespace สำหรับการกู้คืนข้อมูลและการดูแลความสอดคล้องกันของข้อมูล (Concurrency control) แล้ว ระบบจัดการฐานข้อมูลจะบันทึกข้อมูลเก่านั้นไปยัง Flashback Data Archive ด้วย ดังรูป



รูป 2.8 การทำงานของ Flashback Data Archive

(ที่มา Oracle Total Recall with Oracle Database 11g Release 2)

จากรูปจะเห็นได้ว่า หากมีการปฏิบัติกับตาราง ระบบจัดการฐานข้อมูลจะนำข้อมูลที่เป็นค่าเก่า (Old value) ไปเก็บไว้เป็น Undo data และจะมีกระบวนการทำงานเบื้องหลัง (Background process) คอยดูแลเกี่ยวกับการนำข้อมูลที่เป็นค่าดั้งเดิมนั้นไปจัดเก็บไว้ยัง Flashback Data Archive อีกด้วย

หากใช้ Flashback Query ไปยังตารางใดๆที่มีการเปิดการทำงาน Flashback Data Archive ไว้คอยจัดการเกี่ยวกับข้อมูลในอดีต จะสามารถสอบถามข้อมูลย้อนหลังได้ตราบเท่าที่หน่วยความจำจะอำนวย เนื่องจากขนาดของ Flashback Data Archive นั้นเราสามารถกำหนดได้อิสระ แต่หากตารางนั้นๆไม่ได้เปิดการทำงานของ Flashback Data Archive จะสามารถใช้ Flashback Query ได้เช่นกัน แต่ระบบจัดการฐานข้อมูลจะไปค้นข้อมูลที่เป็นค่าเก่า ซึ่งจัดเก็บอยู่ใน Undo table แทนแต่ Undo table นั้นเป็นเป็นพื้นที่จัดเก็บข้อมูลเก่าจากทุกๆตาราง ทำให้หากเกิดกรณีที่ Undo table เต็มข้อมูลที่มาอยู่ก่อนจะถูกลบทิ้งไป แล้วนำข้อมูลใหม่เข้ามาจัดเก็บแทนหรือในกรณีที่ระบบจัดการฐานข้อมูลเห็นว่าไม่มี Transaction ไหนที่ต้องการค่าเก่าแล้ว ระบบจัดการฐานข้อมูลก็จะลบข้อมูลนั้นทิ้งไปเช่นกัน ดังนั้นทำให้เราสามารถสอบถามข้อมูลในอดีตได้เพียงระยะเวลาหนึ่งเท่านั้น ซึ่ง Flashback Data Archive นั้นจะรับประกันว่าข้อมูลที่ต้องการหาจะไม่หายไป หากเรายังเปิดการทำงานของ Flashback Data Archive กับตารางนั้นๆอยู่ หรือรับประกันว่าจะไม่มีทางเกิด error snapshot-too-old ขึ้น หากเราใช้ Flashback Data Archive ซึ่งรายละเอียดเกี่ยวกับ Flashback Query จะกล่าวถึงภายหลัง

ข้อกำหนดในการใช้ Flashback Data Archive มีอยู่ 2 ข้อด้วยกัน ดังนี้

1) Tablespace ที่จะใช้ Flashback Data Drchive ได้นั้นจะต้องบริหารจัดการ Data block ด้วย Automatic Segment Space Management (ASSM) เท่านั้น

2) ต้องเปิดการทำงานของ Automatic Undo Management ซึ่งมีการเปิดการทำงานเป็นค่าเริ่มต้นอยู่แล้ว หากผู้ดูแลระบบสร้างฐานข้อมูลด้วย Database Configuration Assistant (DBCA)

Flashback Data Archive นั้นสามารถทำงานอยู่บน User tablespace ได้เท่านั้น หมายความว่าจำเป็นต้องสร้าง User tablespace ขึ้นมาก่อนเพื่อทำการสร้าง Flashback Data Archive ไว้บนพื้นที่ Tablespace ส่วนนั้น โดยคำสั่งที่ใช้จะเป็นดังนี้

```
CREATE FLASHBACK ARCHIVE CDBFBA1 TABLESPACE test_flashback
RETENTION 1 MONTH NO OPTIMIZE DATA;
```

หลังจากจบคำสั่งนี้ Flashback Data Archive จะถูกสร้างขึ้นแล้วบน Tablespace test_flashback แต่ข้อควรระวังคือ Flashback Data Archive นั้นจะทำงานได้บน User tablespace เท่านั้น ดังนั้นจะไม่สามารถสร้าง Flashback Data Archive บน Tablespace ชนิดอื่นได้

หลังจากที่ได้สร้าง Flashback Data Archive เรียบร้อยแล้ว จะต้องทำการเปิดการทำงานให้กับ Flashback Data Archive ที่สร้างขึ้นนั้นกับตารางที่ต้องการโดยคำสั่งดังนี้

```
ALTER TABLE TABLE_NAME flashback archive
FLASHBACKDATA_NAME;
```

2.4.4 Flashback Query

Flashback Query เป็นการเขียนคำสั่งเอสคิวแอล โดยระบุคำสำคัญพิเศษลงไปเพื่อทำการสอบถามข้อมูล ณ จุดเวลาในอดีต ตัวอย่างเช่น ต้องการทราบว่าตาราง EMPLOYEE ในฐานข้อมูลเมื่อวันที่ 1 มกราคม 2554 เวลา 09:00น เป็นอย่างไรมีข้อมูลอะไรอยู่บ้าง สามารถทำได้โดยระบุคำสำคัญเพิ่มเติมลงไปหลังคำสั่งเอสคิวแอล ดังนี้

```
SELECT * FROM EMPLOYEE

AS OF TIMESTAMP TO_TIMESTAMP('01-01-2554 09:00:00',
'DD-MM-YYYY HH:MI:SS');
```

จากคำสั่งเอสคิวแอลนี้ ระบบการจัดการฐานข้อมูลจะตรวจสอบดูว่าตาราง EMPLOYEE นั้นได้ทำการเปิดการทำงาน Flashback Data Archive เอาไว้หรือไม่ หากยังไม่ได้สร้าง Flashback Data Archive แล้วมาเปิดการทำงานให้มัน ระบบการจัดการฐานข้อมูลก็จะไปตรวจสอบใน Undo Log File ว่ามีค่าเก่าของวันที่ 1 มกราคม 2554 อยู่หรือไม่ หากมีก็จะนำค่านี้ส่งเป็นผลลัพธ์กลับมาเป็นผลลัพธ์ หากไม่มีก็จะฟ้องข้อผิดพลาด (Error) มายังผู้ดูแลระบบ

สาเหตุที่ Undo log file ของ Oracle เก็บข้อมูลเก่าๆไว้หลายเวอร์ชัน แทนที่จะเก็บเพียงแค่เวอร์ชันเดียว ตามหลักของการใช้ Before image journal (BIJ) ที่กล่าวไว้ว่าจะมีการเก็บข้อมูลเก่าจนกระทั่งมีการยืนยันการเปลี่ยนแปลง (Commit) แล้วจะลบข้อมูลเก่านั้นทิ้งออกไปจากระบบ เนื่องจากไม่จำเป็นต้องใช้ข้อมูลเก่านั้นในการทำ Undo ของกระบวนการกู้คืนข้อมูลแล้ว เป็นเพราะว่าการออกแบบ Undo log file ของ Oracle นั้นไม่ได้เป็นไฟล์แยกต่างหากระหว่าง DB space และ Log file แต่ Undo log file นั้นได้ถูกนำมารวมเป็นส่วนหนึ่งของ DB space แล้วเรียกส่วนนั้นว่า

Undo tablespace ด้วยเหตุผลที่ว่า Oracle นั้นเลือกใช้อัลกอริทึม Multiversion scheme ในการจัดการกับปัญหาความสอดคล้องกันของข้อมูล กล่าวคือหากมีหลาย Transaction วิ่งร่วมกันในระบบ Transaction ที่มาก่อนไม่ควรเห็นข้อมูลใหม่ของ Transaction ที่มาทีหลังซึ่งได้ทำการปรับปรุงข้อมูลไว้ เป็นเหตุผลที่ว่าระบบการจัดการฐานข้อมูลต้องเก็บข้อมูลเก่าๆไว้หลายเวอร์ชัน เพื่อเป็นการรองรับการใช้งานข้อมูล ดังนั้นจุดประสงค์ของการใช้ Undo log file ของ Oracle ไม่ใช่มีเพียงแค่จุดประสงค์ในการทำการกู้คืนข้อมูลเพียงอย่างเดียว แต่เป็นการทำดูแลความสอดคล้องกันของข้อมูลร่วมด้วยและผลจากตรงนี้ ทำให้ Flashback Query สามารถไปค้นข้อมูลเก่าที่เก็บไว้ใน Undo log มาได้สำหรับตารางที่ไม่ได้เปิดการทำงาน Flashback Data Archive เพื่อไว้ติดตามการเปลี่ยนแปลงของตาราง

เหตุผลที่ว่าทำไมเมื่อ Flashback Query สามารถสอบถามข้อมูลเก่าๆในตารางจาก Undo log file ได้แล้ว แต่ยังต้องมี Flashback Data Archive ไว้คอยดูแลตารางนั้นอีกซึ่งดูเหมือนจะเป็นการกระทำที่ซ้ำซ้อนในการนำข้อมูลเก่าไปจัดเก็บไว้ถึง 2 แห่ง เป็นเพราะว่า Undo log file นั้นไม่ได้ดูแลเฉพาะตารางใดตารางหนึ่งแต่ Undo log file นั้นเป็น Log file รวมของตารางทุกตารางใน DB space ซึ่งหากให้ Undo log file เก็บทุกๆการเปลี่ยนแปลงของทุกตารางในระบบ จะทำให้ขนาดของ Undo log file นั้นมีขนาดใหญ่มากและไม่ใช่ทุกตารางที่ต้องการจะเก็บการเปลี่ยนแปลงของข้อมูลในระบบ ดังนั้นการใช้ Flashback Data Archive นั้นจะสามารถดูแลเฉพาะตารางที่ต้องการใช้ข้อมูลเหล่านี้จริงๆ จึงเป็นวิธีการที่เหมาะสมกว่า ถึงแม้จะต้องแลกด้วยมูลค่า (Cost) ในการจัดเก็บข้อมูลเพิ่มเติม

ตัวอย่างของการใช้ Flashback Query เป็นดังนี้

สมมติให้มีการสร้างตาราง EMP ซึ่งเป็นตารางเก็บข้อมูลเกี่ยวกับ Employee และมีการเปิดการทำงานให้กับ Flashback Data Archive เอาไว้เรียบร้อยแล้ว สามารถใช้คำสั่งฐานข้อมูลเอสคิวเอลในการสอบถามข้อมูลแล้วตามด้วยคำสำคัญคือ as of timestamp ดังนี้

```
SELECT * FROM EMP

AS OF TIMESTAMP TO_TIMESTAMP('11092014 00:42:23','ddmmyyyy hh24:mi:ss');
```

อธิบายคำสั่งได้ว่า ต้องการตารางของ EMP เวอร์ชัน วันที่ 11 เดือนกันยายน ปี 2014 เวลา 00:42:23 น. โดย 'ddmmyyyy hh24:mi:ss' นั้นเป็นพารามิเตอร์ (parameter) เพื่อบอกระบบการ

จัดการฐานข้อมูลว่ารูปแบบ (Format) ของเวลาที่เราระบุเป็นเช่นไร ซึ่งสามารถปรับเปลี่ยนได้แต่เวลาที่ระบุจะต้องสอดคล้องกันกับรูปแบบของเวลา ซึ่งผลที่ได้จะเป็นดังนี้

	ENO	ESTART	EEND	EDEPT
1	22217	01-JAN-10	12-NOV-11	3
2	22218	01-JAN-10	12-NOV-11	3
3	22219	01-JAN-10	12-NOV-11	3

รูป 2.9 ผลลัพธ์จากการใช้ Flashback Query

จากที่ได้ทดสอบเพิ่มเติมพบว่า Flashback Query นั้นสามารถสอบถามข้อมูลในฐานข้อมูล ณ เวลาที่ระบุได้ทั้งข้อมูลที่เป็นปัจจุบันและข้อมูลที่เคยมีอยู่ในอดีต ตัวอย่างเช่น หากตอน 09.00 น. มีข้อมูลอยู่ใน ฐานข้อมูลทั้งสิ้น 2 แถว หากเพิ่มแถวข้อมูลเข้าไปเมื่อเวลา 10.00 น. แต่ทำการใช้ Flashback Query ในการ สอบถามข้อมูล ณ เวลา 9.59 น. ผลลัพธ์จะถูกส่งกลับออกมาเพียง 2 แถว เท่าเดิม เนื่องจากข้อมูลที่ได้ทำการเพิ่มเมื่อเวลา 10.00 น. ไม่ถูกรวมเข้าไปด้วย และเช่นกันหากสอบถามข้อมูล ณ เวลา 10.01 จะพบว่าตอนนี้ในฐานข้อมูลมีข้อมูลทั้งสิ้น 3 แถวโดยรวมแถวที่เพิ่มเข้ามาด้วย ดังนั้น Flashback Query ก็จะให้ผลลัพธ์กลับมาทั้งสิ้น 3 แถว จะเห็นว่าไม่จำเป็นต้องเป็นข้อมูลในอดีตเท่านั้นถึงจะใช้ Flashback Query ได้ ข้อมูลปัจจุบันก็สามารถสอบถามได้ด้วย Flashback Query เช่นกัน อีกทั้งยังสามารถใช้ Build-in function ร่วมกับ Flashback Query และ Flashback Versions Query ได้ แต่ Flashback Query ไม่สามารถใช้ร่วมกันกับตัวดำเนินการเชิงเวลา (Temporal operator) ตามมาตรฐานของภาษา SQL 2011 ได้เลย

2.4.5 Flashback Versions Query

Flashback Versions query เป็นการสอบถามข้อมูลแบบพิเศษที่มีการทำงานคล้ายคลึงกันกับ Flashback Query แต่มีความแตกต่างกันตรงที่ Flashback Query นั้นจะเป็นการสอบถามข้อมูลไปยังจุดจุดหนึ่งในอดีต ซึ่งผลที่ได้มานั้นคือจะได้ฐานข้อมูลที่มีเวอร์ชัน ณ เวลาที่เราต้องการ แต่หากใช้ Flashback Versions Query แล้วสิ่งที่จะเป็นผลลัพธ์และถูกส่งกลับมาก็คือ การเปลี่ยนแปลงทั้งหมดของตารางนั้นๆ ไม่ว่าจะเป็นการจัดเก็บข้อมูล การปรับปรุงข้อมูลหรือการลบข้อมูล ซึ่งสามารถใช้การดำเนินการ Where ในการกรองเอาเฉพาะการดำเนินการที่เราต้องการจะเรียกดูเฉพาะก็ได้ เช่น ถ้าต้องการมองภาพรวมของการเพิ่มข้อมูลทั้งหมดของตารางหรือจะทำการกู้คืนข้อมูลจากการเรียกใช้ Flashback Versions Query ก็ได้เช่นกัน การใช้ Flashback Versions Query นั้นจะต้องระบุคำสำคัญเพิ่มเติมเพื่อบอกระบบการจัดการฐานข้อมูลว่าเป็นการสอบถามแบบ Flashback Versions

Query คือ VERSIONS BETWEEN {SCN|TIMESTAMP} start AND end หรือจะใช้ VERSIONS PERIOD FOR user_valid_time [BETWEEN TIMESTAMP start AND end] ก็ได้

ตัวอย่างของการใช้ Flashback Versions Query เป็นดังนี้

สมมติให้มีสภาพแวดล้อม (Environment) ของระบบเช่นเดียวกับตัวอย่างการใช้ Flashback Query ทดสอบการใช้คำสั่ง Flashback Versions Query ได้ดังนี้

```
SELECT ENO,ESTART,EEND,EDEPT,VERSION_OPERATION OPERATION
FROM EMP
VERSION BETWEEN SCN MINVALUE AND MAXVALUE
```

ผลที่ได้เป็นดังนี้

	ENO	ESTART	EEND	EDEPT	OPERATION
1	22219	01-JAN-10	12-NOV-11	3	D
2	22218	01-JAN-10	12-NOV-11	3	D
3	22217	01-JAN-10	12-NOV-11	3	D
4	22217	01-JAN-10	12-NOV-11	3	(null)
5	22218	01-JAN-10	12-NOV-11	3	(null)
6	22219	01-JAN-10	12-NOV-11	3	(null)
7	22217	01-JAN-10	12-NOV-11	3	U
8	22217	01-JAN-10	12-NOV-11	4	I
9	22217	01-JAN-10	12-NOV-11	3	D
10	22217	01-JAN-10	12-NOV-11	3	U
11	11112	01-JAN-10	12-NOV-11	3	I
12	11113	01-JAN-10	12-NOV-11	3	I
13	11114	01-JAN-10	12-NOV-11	3	I

รูป 2.10 ผลลัพธ์จากการใช้ Flashback Versions Query

จะเห็นได้ว่าประวัติการเพิ่มข้อมูล การปรับปรุงข้อมูลและการลบข้อมูลจะถูกจัดเก็บไว้ทั้งหมดและสามารถเรียกดูย้อนหลังได้ ซึ่งข้อจำกัดที่ว่าสามารถเรียกดูย้อนได้มากแค่ไหนนั้นขึ้นอยู่กับว่าพื้นที่ในหน่วยความจำมีมากน้อยเท่าใด หากมีพื้นที่มากก็สามารถสร้าง Flashback Data Archive ที่ใหญ่เพื่อจัดเก็บรายละเอียดการเปลี่ยนแปลงเหล่านี้ได้ และเช่นกันหากพื้นที่มีจำกัดไม่เพียงพอต่อการสร้าง Flashback Data Archive ขึ้นมาเก็บรายละเอียดการเปลี่ยนแปลงของเวอร์ชันต่างๆของตารางได้ ก็จะทำได้เพียงแค่เรียกดูเวอร์ชันเก่าๆที่ยังคงมีอยู่ใน Undo log file เท่านั้น โดยค่าการดำเนินการที่เป็น null นั้นหมายถึงข้อมูลที่มีอยู่ในตารางเวอร์ชันปัจจุบัน ซึ่งเป็นข้อมูลที่ยังไม่มีการระบุว่ามีการดำเนินการอะไรเกิดขึ้นกับแถวนั้นบ้าง

2.4.6 Undo Tablespace และ Flashback Data Archive

Undo Tablespace และ Flashback Data Archive นั้นใช้จัดเก็บค่าเก่าคู่ แต่จุดประสงค์การใช้งานนั้นเป็นคนละอย่างกันดังที่ได้กล่าวไปข้างต้น ซึ่งจากข้อสันนิษฐานคิดว่าทั้งสองส่วน น่าจะเก็บแยกกันไม่น่าจะเป็นส่วนเดียวกัน ซึ่งจากการค้นคว้าได้ข้อสรุปที่บ่งบอกว่าทั้งสองส่วนจัดเก็บแยกกันดังนี้

2.4.6.1 Undo tablespace

Undo tablesapce นั้นจะเป็นส่วนที่ใช้เก็บข้อมูลที่จำเป็นสำหรับการยกเลิกเพื่อในการย้อนการทำงานกลับไปยังจุดเริ่มต้นการประมวลผลรายการเปลี่ยนแปลง (Rollback) ของกระบวนการกู้คืนข้อมูล ซึ่งผู้ดูแลระบบไม่สามารถสร้างตารางปกติบน Tablespace นั้นได้ซึ่ง Undo tablespace นั้นสามารถมีได้หลาย Undo tablespace แต่จะมีเพียง Undo tablespace หนึ่งเท่านั้นที่จะถูกเลือกมาใช้งาน ณ เวลาหนึ่ง Undo data ใน tablespace นั้นมีอยู่ 3 สถานะ ดังนี้

ตาราง 2.24 รายละเอียดเกี่ยวกับ Undo Data ใน Tablespace

[ที่มา http://docs.oracle.com/database/121/ADMIN/undo.htm#ADMIN10180]

สถานะ	ความหมาย	เวลาที่ Undo data ถูกเขียนทับ
Uncommitted undo information	คำสั่งต่างๆ ของ Transaction ที่กำลังถูกประมวลผล (Tx ยังไม่ยืนยันการเปลี่ยนแปลง) ซึ่งจะนำข้อมูลส่วนนี้ไปใช้ในการย้อนการทำงานกลับไปจุดเริ่มต้น	ไม่ถูกเขียนทับ
Committed undo information (Unexpired)	Undo data ยังอยู่ในช่วงของ Undo_retention (Unexpired undo)	หลังจาก Undo_retention หรือพื้นที่ของ Undo tablespace เหลือน้อย
Expired undo information	Undo data ที่ไม่ต้องการแล้ว	ถูกเขียนทับเสมอ

จากตารางจะเห็นได้ว่า Undo table นั้นไม่ได้เก็บข้อมูลเก่าไว้ตลอดไป แต่จะมีช่วงเวลาและเงื่อนไขในการเก็บซึ่งเวลาที่เป็นตัวบ่งบอกการเก็บข้อมูลไว้นั้นคือ undo_retention เป็นเวลาอย่างน้อยที่ Oracle database จะยังคงเก็บรักษาข้อมูลเก่านั้นไว้ก่อนจะเขียนทับไป

เพื่อที่จะแสดงว่าไม่ได้ใช้ undo tablespace ในการสืบค้นข้อมูลเก่า ณ เวลาใดๆ จะทำโดยสร้าง Undo tablespace อันใหม่และเปิดการทำงาน Flashback Data Archive เข้ากับตารางที่ต้องการจะเก็บข้อมูลในอดีต ก่อนที่จะทำการปิดการใช้งาน (Disable) Undo tablespace ที่เป็นค่าเริ่มต้นทิ้ง จากนั้นทำการเพิ่มข้อมูลลงไปในตารางนั้น ก่อนจะทำการสอบถามข้อมูลเก่าในตารางด้วย Flashback Query (ข้อมูลเก่าไม่จำเป็นต้องเป็นข้อมูลที่ลบหรือปรับปรุง แต่เป็นข้อมูลที่ยังมีอยู่ในปัจจุบันแต่ถามถึงข้อมูลที่อยู่ในอดีตได้) ซึ่งหากข้อมูลเก่าในตารางหายไปด้วยนั้นหมายความว่าข้อมูลเก่านั้นได้จัดเก็บอยู่ใน Flashback Data Archive ที่เป็นส่วนเดียวกันกับ Undo tablespace แต่หากยังสามารถสอบถามข้อมูลขึ้นมาได้นั่นหมายความว่า Undo tablespace และ Flashback Data Archive เป็นคนละส่วนกัน ซึ่งมีรายละเอียดการทดลอง ดังนี้

```
SQL>
create
 flashback archive fl_arch
  2  tablespace tbs_arch retention 1 year;
Flashback archive created.
SQL>
```

รูป 2.11 การสร้าง Flashback Data Archive
(ที่มา Oracle Total Recall Tips)

```
SQL>
create
 table tbl_fl_archive (id number, name varchar2(20));
Table created.

SQL>
insert into
 tbl_fl_archive values(1,'Flashback Archive');
1 row created.

SQL>
commit;
Commit complete.

SQL>
select * from
 tbl_fl_archive;

        ID NAME
---------- --------------------
         1 Flashback Archive

SQL>
alter
 table tbl_fl_archive flashback archive fl_archive;
Table altered.
SQL>
```

รูป 2.12 การเปิดการใช้งาน Flashback Data Archive
เข้ากับตารางที่ต้องการจะเก็บข้อมูลในอดีต (ที่มา Oracle Total Recall Tips)

```
SQL>
select
 to_char(sysdate,'ddmmyyyy hh24:mi:ss') ddate
from
 dual;

DDATE
-----------------
13022010 12:46:49
```

รูป 2.13 การเรียกดูเวลาที่มีข้อมูลอยู่ในตารางแล้ว (ที่มา Oracle Total Recall Tips)

```
SQL>
conn / as sysdba
Connected.

SQL>
show
 parameter undo_tablespace;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------
undo_tablespace                      string      UNDOTBS1

SQL>
select
 a.name
from
 v$datafile a, v$tablespace b
where
 a.ts#=b.ts# and b.name='UNDOTBS1';

NAME
-------------------------------------------------
C:\APP\ADMINISTRATOR\ORADATA\DB2\UNDOTBS01.DBF
```

รูป2.14 การเรียกดู Undo tablespace อันเดิม ก่อนจะสร้างอันใหม่แทนที่ (ที่มา Oracle Total Recall Tips)

```
SQL>
create
 undo tablespace undotbs2 datafile
'c:\app\administrator\oradata\db2\undotbs02.dbf' size 10m;
Tablespace created.

SQL>
alter
 system set undo_tablespace='undotbs2';
System altered.

SQL>
startup
 force
ORACLE instance started.

Total System Global Area  431038464 bytes
Fixed Size                  1333676 bytes
Variable Size             251659860 bytes
Database Buffers          171966464 bytes
Redo Buffers                6078464 bytes
Database mounted.
Database opened.

SQL>
show
 parameter undo_tablespace;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------
undo_tablespace                      string      UNDOTBS2
```

รูป 2.15 การสร้าง Undo tablespace อันใหม่พร้อมทั้งเปิดการเริ่มใช้งาน Undo tablespace อันใหม่นี้ (ที่มา Oracle Total Recall Tips)

```
SQL>
conn
 us1/us1
Connected.
SQL>
select * from
 tbl_fl_archive as of timestamp to_timestamp('13022010
12:45:30','ddmmyyyy hh24:mi:ss');

        ID NAME
---------- --------------------
         1 Flashback Archive

SQL>
```

รูป 2.16 การใช้ Flashback Query (ที่มา Oracle Total Recall Tips)

จะเห็นว่าเราได้ทำเปิดการใช้งาน Undo tablespace อันใหม่ไว้แล้ว แต่ก็ยังสามารถใช้ Flashback Query ในการสอบถามข้อมูลในเวลาอดีตขึ้นมาได้ จึงสรุปได้ว่า Flashback Data Archive และ Undo tablespace นั้นเป็นคนละส่วนกันจริง

กรณีที่ Flashback Data Archive เต็มหรือใกล้จะเต็มแล้ว หากดำเนินงานคำสั่งสำหรับการจัดการข้อมูล (Data Manipulation Language: DML) ในตารางที่ใช้ Flashback Data Archive นั้นจะไม่สามารถทำได้ ซึ่งผู้ดูแลระบบจำเป็นต้องบริหารจัดการดังนี้

1) กำหนดให้ระบบการจัการฐานข้อมูลล้างข้อมูลในอดีตโดยอัตโนมัติเมื่อพ้นกำหนดระยะเวลาการเก็บที่ได้เคยระบุไว้ โดยการระบุระยะเวลาในการจัดเก็บข้อมูล (Retention) ใน Flashback Data Archive ทำได้ดังนี้

```
ALTER FLASHBACK ARCHIVE test_archive1
   MODIFY RETENTION 1 MONTH;
```

รูป 2.17 การกำหนดระยะเวลาในการจัดเก็บข้อมูลใน Flashback Data Archive

2) ใช้วิธีการล้างข้อมูลเฉพาะกิจโดยผู้ดูแลระบบ ด้วยคำสั่ง PURGE ดังนี้

- PURGE ALL: ลบข้อมูลทั้งหมดใน Flashback Data Archive ซึ่งสามารถเรียกดูข้อมูลในอดีตได้โดยใช้ Flashback Query หากข้อมูลเหล่านั้นยังคงอยู่ใน Undo log file
- PURGE BEFORE SCN: ลบข้อมูลทั้งหมดจาก Flashback Data Archive ที่อยู่มาก่อน System change number ที่ระบุ
- PURGE BEFORE TIMESTAMP: ลบข้อมูลทั้งหมดจาก Flashback Data Archive ก่อน TIMESTAMP ที่ระบุ

ดังนั้นการใช้ Flashback Data Archive จะต้องออกแบบให้มีขนาดที่เหมาะสมกับระบบและมีผู้ดูแลคอยบริหารจัดการในกรณีที่ Flashback Data Archive ใกล้เต็ม เนื่องจากหาก Flashback Data Archive เต็มนั้น จะไม่สามารถจัดเก็บข้อมูลเพิ่มเติมได้เลย และอาจจะเกิดเหตุการณ์ข้อมูลหาย (Data lost) ได้

2.4.7 Rollback Segment กับ Undo Tablespace สำหรับข้อความแจ้งข้อผิดพลาด (Error Message)

ในกรณีที่ไม่ได้เปิดการใช้งาน Flashback Data Archive เอาไว้ เมื่อใช้ Flashback Query ไปแล้ว ระบบการจัดการฐานข้อมูลก็จะตรวจพบว่าข้อมูลที่เราต้องการหานั้นไม่ได้มีอยู่ใน Undo log file แล้วจะฟ้องข้อผิดพลาดว่า rollback segment too old หมายความว่า Oracle ยังใช้ข้อความแจ้งข้อผิดพลาดแบบเก่าอยู่ เนื่องจาก Rollback segment นั้นคือชื่อเก่าที่ Oracle เคยใช้เรียก Undo tablespace นั่นเอง

2.5 ฐานข้อมูลเชิงเวลาใน Oracle Database 12c

จากรายละเอียดเกี่ยวกับฐานข้อมูลข้อมูลเชิงเวลาที่ได้กล่าวไว้ข้างต้นที่ว่าระบบจัดการฐานข้อมูลของ Oracle นั้นสามารถดำเนินการกับฐานข้อมูลเชิงเวลาได้เป็นเวลานานแล้วก่อนที่จะมีมาตรฐานภาษาเอสคิวแอลรุ่น 2011 นั่นหมายความว่าความสามารถดังกล่าวที่เสนอในรายละเอียดเกี่ยวกับฐานข้อมูลเชิงเวลาดังกล่าวไม่ใช่เรื่องยากอีกต่อไป หากใช้ระบบการจัดการฐานข้อมูลของ Oracle ทั้งการสอบถามข้อมูลต่างๆรวมไปถึงการบริหารจัดการข้อมูลเชิงเวลา เพียงแต่ว่าความสามารถตรงนี้ไม่เป็นไปตามมาตรฐานภาษาเอสคิวแอลรุ่น 2011 จึงไม่เป็นที่ยอมรับทางวิชาการ Oracle จึงได้เพิ่มเติมความสามารถมาตรฐานนี้ลงไปยังระบบการจัดการฐานข้อมูลเวอร์ชันล่าสุดของ Oracle นั่นคือ Oracle database 12c พร้อมทั้งรับรองว่าสามารถทำตามมาตรฐานภาษาเอสคิวแอลรุ่น 2011 ได้เป็นที่เรียบร้อยแล้ว แต่จากการทดลองตามเอกสารอ้างอิงของ Krishna Kulkarni และ Jan-Eike Michels เรื่อง Temporal features in SQL:2011 พบว่าระบบการจัดการฐานข้อมูลของ Oracle 12c นั้นสามารถใช้งานได้เพียงแค่ส่วนชนิดข้อมูล (Data type) ที่เป็นคาบเวลา, คำสั่งการเพิ่มข้อมูลและการสอบถามข้อมูลตามมาตรฐานภาษาเอสคิวแอลรุ่น 2011 แต่การปรับปรุงข้อมูลแบบ Sequence, การลบข้อมูลแบบ Sequence และการดำเนินการทั้งหมดเกี่ยวกับการ Temporal select ยังไม่สามารถทำได้

2.5.1 การเปรียบเทียบคาบเวลา (Period comparison)

ในระบบการจัดการฐานข้อมูลของ Oracle 12c นั้นมีชนิดข้อมูลที่เป็นไปตามมาตรฐานเอสคิวแอล 2011 นั่นคือคาบเวลา ซึ่งการเปรียบเทียบชนิดข้อมูลคาบเวลาสามารถอธิบายได้ดังนี้

การจัดเก็บข้อมูลมีช่วงเวลาดังนี้

รูป 2.18 ช่วงเวลาที่ชนิดข้อมูลเป็นคาบเวลาในฐานข้อมูล

จากนั้นทำการสอบถามด้วยคำสั่งเอสคิวแอล ดังนี้

```
SELECT * FROM EMP

WHERE ESTART <= DATE '2011-01-02' AND EEND > DATE '2011-01-02';
```

ผลที่ได้คือ

รูป 2.19 ผลลัพธ์ของการสอบถามด้วยคำสั่ง Select ข้างต้น

จะได้ว่าแถวข้อมูลไหนที่มีช่วงเวลาที่ระบุเป็นซับเซต (Subset) ของช่วงเวลาของตัวเองก็จะถูกส่งกลับเป็นผลลัพธ์กลับมา

2.5.2 ฟังก์ชันมาตรฐานใน Oracle 12c (Build-In Function in Oracle 12c)

ฟังก์ชันมาตรฐานของระบบการจัดการฐานข้อมูล Oracle 12c มีดังนี้

2.5.2.1 ฟังก์ชัน Single-Row

เป็นฟังก์ชันที่ให้ผลลัพธ์เพียงข้อมูลแถวเดียวจากทุกๆ แถวของตารางที่ถูกสอบถาม โดยฟังก์ชันเหล่านี้จะปรากฏใน Select list, การดำเนินการ WHERE , START WITH และการดำเนินการ CONNECT BY , การดำเนินการ HAVING ซึ่งมีฟังก์ชัน Single row ที่เกี่ยวข้องกับเวลาดังนี้

ฟังก์ชันเกี่ยวกับวันเวลา (Datetime function): กระทำกับ date, timestamp และช่วงเวลา (interval)

- **ADD_MONTHS:** ให้ผลลัพธ์เป็นเดือนที่ต้องการ เช่น ADD_MONTHS(LAST_DAY(hire_date), 5) จะส่งกลับผลลัพธ์เป็นอีกห้าเดือนข้างหน้านับจากวันจ้าง
- **CURRENT_DATE:** ให้ผลลัพธ์เป็นวันที่ปัจจุบันในแต่ละโซนเวลา เช่น 29-MAY-2000 13:14:03 ตาม ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MON-YYYY HH24:MI:SS';
- **CURRENT_TIMESTAMP:** ให้ผลลัพธ์เป็น TIMESTAMP พร้อมโซนเวลา เช่น 04-APR-00 01.27.18.999220 PM -05:00
- **DBTIMEZONE:** ให้ผลลัพธ์เป็นโซนเวลาของฐานข้อมูล เช่น +00:00
- **EXTRACT (datetime):** ดึงข้อมูลและส่งผลลัพธ์เป็นวันเวลาที่ระบุจาก datetime หรือ interval expression เช่น – EXTRACT(month FROM order_date)
- **FROM_TZ:** แปลง Timestamp และโซนเวลาเป็น TIMESTAMP WITH TIME ZONE เช่น FROM_TZ(TIMESTAMP '2000-03-28 08:00:00', '3:00') ให้ผลลัพธ์เป็น 28-MAR-00 08.00.000000000 AM +03:00
- **LAST_DAY:** ให้ผลลัพธ์เป็นวันสุดท้ายของเดือนเป็น เช่น 31-OCT-04, 31-AUG-08
- **LOCALTIMESTAMP:** ให้ผลลัพธ์เป็นวันและเวลาปัจจุบันในแต่ละโซนเวลา โดยชนิดข้อมูลเป็น Timestamp เช่น 04-APR-00 01.27.19 PM
- **MONTHS_BETWEEN:** ระยะห่างระหว่างเดือนคิดเป็น Integer
- **NEW_TIME:** NEW_TIME(TO_DATE('11-10-09 01:23:45', 'MM-DD-YY HH24:MI:SS'), 'timezone1', 'timezone2') ผลลัพธ์จะเป็นวันและเวลาใหม่ที่เป็น timezone2

- **NEXT_DAY:** ให้ผลลัพธ์เป็นวันตาม char ที่อยู่ถัดจากวันที่ที่ระบุของสัปดาห์ถัดไป เช่น NEXT_DAY('15-OCT-2009','TUESDAY') ผลลัพธ์คือ 20-OCT-2009 00:00:00 ที่เป็นวันอังคาร
- **NUMTODSINTERVAL:** แปลง n เป็นช่วงเวลา โดยช่วงเวลาเป็นได้ทั้ง วัน (DAY), ชั่วโมง(HOUR), นาที (MINUTE), วินาที (SECOND) เช่น NUMTODSINTERVAL(100, 'day') เป็นการแปลงวันเป็น 100 วันนับจากนี้
- **NUMTOYMINTERVAL:** แปลง n เป็นช่วงเวลา โดยช่วงเวลาเป็นได้ทั้ง ปี (YEAR), เดือน (MONTH) เช่น NUMTOYMINTERVAL(1,'year') คู 1 ปีนับจากนี้
- **ORA_DST_AFFECTED:** ฟังก์ชั่นจะใช้วันเวลา (datetime) ที่แก้ไข TIMESTAMP WITH TIME ZONE หรือ VARRAY ที่มี TIMESTAMP WITH TIME ZONE ฟังก์ชันให้ผลลัพธ์เป็น 1 ถ้าวันเวลาได้รับผลกระทบหรือจะทำให้เกิดข้อผิดพลาด "nonexisting time" หรือ "duplicate time" กับข้อมูลโซนเวลาใหม่ นอกนั้นให้ผลลัพธ์เป็น 0 (เปลี่ยนไฟล์ข้อมูลโซนเวลา)
- **ORA_DST_CONVERT:** ฟังก์ชั่นช่วยในการระบุการจัดการข้อผิดพลาดสำหรับการแสดงวันเวลา (เปลี่ยนไฟล์ข้อมูลโซนเวลา)
- **ORA_DST_ERROR:** ฟังก์ชั่นจะใช้วันเวลาที่แก้ไข TIMESTAMP WITH TIME ZONE หรือVARRAY ที่มี TIMESTAMP WITH TIME ZONE และแสดงให้เห็นว่าค่าวันที่และเวลาจะส่งผลให้เกิดข้อผิดพลาดกับข้อมูลโซนเวลาใหม่ โดยเป็น 0 หมายถึงค่าวันเวลาไม่แสดงข้อผิดพลาดในข้อมูลโซนเวลาใหม่, 1878 หมายถึงค่าวันเวลาเกิดข้อผืดพลาด "nonexisting time" , 1883 หมายถึงค่าวันเวลาเกิดข้อผิดพลาด "duplicate time"
- **ROUND (date):** ประมาณเวลา เช่น ROUND (TO_DATE ('27-OCT-00'),'YEAR') ผลลัพธ์เป็น 01-JAN-01
- **SESSIONTIMEZONE**
- **SYS_EXTRACT_UTC:** แยก UTC (Coordinated Universal Time—formerly Greenwich Mean Time) จากค่าวันเวลาของโซนเวลาชดเชยหรือชื่อภูมิภาคโซนเวลา เช่น SYS_EXTRACT_UTC(TIMESTAMP '2000-03-28 11:30:00.00 -08:00') ผลลัพธ์เป็น 28-MAR-00 07.30.00 PM
- **SYSDATE:** ให้ผลลัพธ์เป็นวันที่ปัจจุบันและเวลาของระบบปฏิบัติการ (Operating system) ที่เครื่องแม่ข่ายให้บริการจัดเก็บข้อมูล (Database server) ติดตั้งอยู่ เช่น 04-13-2001 09:45:51

- **SYSTIMESTAMP:** ให้ผลลัพธ์เป็นวันที่ของระบบรวมถึงวินาทีและโซนเวลาของระบบ ชนิดของข้อมูลเป็น TIMESTAMP WITH TIME ZONE
- **TO_CHAR (datetime):** แปลงวันเวลาหรือช่วงเวลาของ DATE, TIMESTAMP, TIMESTAMP WITH TIME ZONE, TIMESTAMP WITH LOCAL TIME ZONE, INTERVAL DAY TO SECOND, INTERVAL YEAR TO MONTH เป็น VARCHAR2
- **TO_DSINTERVAL:** แปลง CHAR, VARCHAR2, NCHAR, NVARCHAR2 เป็น INTERVAL DAY TO SECOND เช่น TO_CHAR(TIMESTAMP '2009-01-01 00:00:00' + TO_DSINTERVAL('P100DT05H'), 'YYYY-MM-DD HH24:MI:SS') ผลลัพธ์เป็น 2009-04-11 05:00:00
- **TO_TIMESTAMP:** แปลง char ของ CHAR, VARCHAR2, NCHAR, หรือ NVARCHAR2 เป็น TIMESTAMP
- **TO_TIMESTAMP_TZ:** แปลง char ของ CHAR, VARCHAR2, NCHAR, หรือ NVARCHAR2 เป็น TIMESTAMP WITH TIMEZONE
- **TO_YMINTERVAL:** แปลง CHAR, VARCHAR2, NCHAR, NVARCHAR2 เป็น INTERVAL YEAR TO MONTH เช่น hire_date + TO_YMINTERVAL('01-02') หมายถึง จากวันที่จ้างบวกไปอีก 1 ปี กับอีก 2 เดือน
- **TRUNC (date):** เป็นการประมาณเวลาโดยการตัดทอนให้ครบรอบเวลา เช่น TRUNC(TO_DATE('27-OCT-92','DD-MON-YY'), 'YEAR') ผลลัพธ์เป็น 01-JAN-92
- **TZ_OFFSET** [15]

2.5.2.2 ฟังก์ชันการรวม (Aggregate Function)

เป็นฟังก์ชันที่ให้ผลลัพธ์เพียงแถวเดียวซึ่งขึ้นอยู่กับกลุ่มของแถวมากกว่าแบบ single row โดยฟังก์ชั่นนี้ปรากฏได้ใน Select list และในการดำเนินการ ORDER BY และ HAVING โดยทั่วไปนิยมใช้ฟังก์ชันการรวมร่วมกับ GROUP BY ในคำส่ง SELECT ซึ่ง Oracle จะแบ่งแถวของตารางที่ถูกสอบถามออกเป็นกลุ่มๆ และทำฟังก์ชันการรวมกับแต่ละกลุ่มของแถวและให้ผลลัพธ์เป็นแถวเดียวของแต่ละกลุ่ม ซึ่งหากถ้าไม่ใช้ GROUP BY แล้วนั้น Oracle จะทำฟังก์ชันการรวมกับทุกแถวใดๆ ซึ่งสามารถใช้ฟังก์ชันการรวมได้ในการดำเนินการ HAVING โดยมีรายละเอียดของฟังก์ชันดังนี้

- DISTINCT และ UNIQUE ซึ่งมีความหมายเหมือนกัน ทำให้ฟังก์ชันการรวมจะต้องพิจารณาเพียงค่าที่แตกต่างกันนับเพียงหนึ่ง ตัดที่ซ้ำออกไป ซึ่งรูปแบบการใช้ฟังก์ชั่นนั้นใช้คำสำคัญคือ DISTINCT

- ALL ทำให้ฟังก์ชันการรวมจะต้องพิจารณาค่าทั้งหมด ซึ่งรวมการซ้ำทั้งหมด เช่น DISTINCT average of 1, 1, 1, and 3 ผลลัพธ์คือ 2 คิดโดย [(1+3)/2] และ ALL average ผลลัพธ์คือ 1.5 คำนวนโดย [(1+1+1+3)/4] ถ้าหากไม่กำหนดอันใดอันหนึ่ง ค่าเริ่มต้นคือ ALL
- เราสามารถรวมฟังก์ชั่นต่างๆเข้าด้วยกันได้ ตัวอย่างเช่น คำนวณค่าเฉลี่ย (Average) ของค่าจ้างที่มากที่สุด (Maximum) ของทุกแผนก (Department) ดังตัวอย่าง

```
SELECT AVG(MAX(salary))
  FROM employees
  GROUP BY department_id;

AVG(MAX(SALARY))
----------------
      10926.3333
```

รูป 2.20 ตัวอย่างการใช้ฟังก์ชันมาตรฐาน

ตัวอย่าง ฟังก์ชันการรวม มีดังนี้

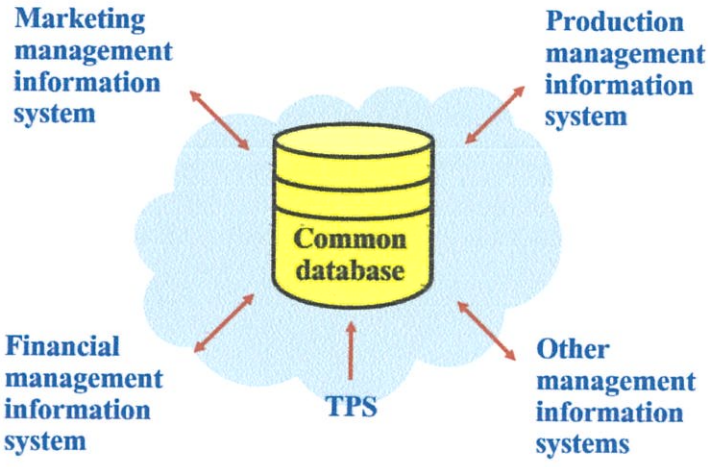
APPROX_COUNT_DISTINCT / AVG / COLLECT / CORR / CORR_* / COUNT / COVAR_POP / COVAR_SAMP / CUME_DIST / DENSE_RANK / FIRST / GROUP_ID / GROUPING / GROUPING_ID / LAST / LISTAGG / MAX / MEDIAN / MIN / PERCENT_RANK / PERCENTILE_CONT / PERCENTILE_DISC / RANK / REGR_ (Linear Regression) Functions / STATS_BINOMIAL_TEST / STATS_CROSSTAB / STATS_F_TEST / STATS_KS_TEST / STATS_MODE / STATS_MW_TEST / STATS_ONE_WAY_ANOVA / STATS_T_TEST_* / STATS_WSR_TEST / STDDEV / STDDEV_POP / STDDEV_SAMP / SUM / SYS_OP_ZONE_ID / SYS_XMLAGG / VAR_POP / VAR_SAMP / VARIANCE / XMLAGG [16]

2.6 ระบบฐานข้อมูลที่ใช้ในองค์กร

2.6.1 การรวมฐานข้อมูล

ในการออกแบบระบบฐานข้อมูลที่ใช้ในองค์กรนั้น สิ่งที่สำคัญที่อยากให้เกิดคือทำอย่างไรองค์กรถึงจะมีฐานข้อมูลที่ทุกๆโปรแกรมประยุกต์จะใช้งานฐานข้อมูลนั้นร่วมกันได้ ซึ่งสิ่งที่ควรจะคำนึงถึงในการออกแบบฐานข้อมูลคือ การที่ระบบมีโปรแกรมประยุกต์อยู่หลากหลายโปรแกรม จะทำอย่างไรให้หลากหลายโปรแกรมประยุกต์นั้นมีความเชื่อมโยงข้อมูลกันหรือหากจะกล่าวอีกนัยหนึ่งคือ แต่ละโปรแกรมประยุกต์นั้นมีการใช้ข้อมูลร่วมกัน ดังนั้นการจะออกแบบฐานข้อมูลที่มีหลายโปรแกรมใช้งานข้อมูลร่วมกันได้นั้น อันดับแรกจะต้องระบุให้ชัดเจนว่ามีโปรแกรมประยุกต์อะไรบ้างอยู่ในระบบขององค์กร และกระบวนการทำงานของแต่ละโปรแกรมมีขั้นตอนอย่างไร ซึ่งอาจจะอธิบายอยู่ในรูปแผนภาพกระแสข้อมูล (Data flow diagram) หรือแผนภาพแสดงลำดับการทำงานของระบบ (Sequence diagram) ก็ได้ หลังจากที่ทราบกระบวนการทำงานของแต่ละโปรแกรมประยุกต์แล้ว สิ่งที่จะได้ตามมาก็คือจะทราบแอตทริบิวต์จากกระบวนการทำงานนั้นๆ นั่นคือทราบว่าแต่ละขั้นตอนจะต้องใช้ข้อมูลอะไร จากแอตทริบิวต์ไหนบ้าง และแต่ละขั้นตอนให้ผลลัพธ์ข้อมูลอะไร ของแอตทริบิวต์ใดบ้าง แต่จะยังไม่ทราบว่า สุดท้ายแล้วข้อมูลเหล่านี้มีความเกี่ยวข้องกันอย่างไร หรือจะกลายเป็นตารางอะไร ดังนั้นการออกแบบฐานข้อมูล เราไม่ได้ออกแบบมารองรับฐานข้อมูลของทั้งหน่วยงาน แต่เราควรจะออกแบบฐานข้อมูลนั้นให้มารองรับแต่ละโปรแกรมประยุกต์มากกว่า ซึ่งจากที่เราทราบกระบวนกรทำงานของแต่ละโปรแกรมจนทำให้ทราบแอตทริบิวต์ของแต่ละกระบวนการทำงานที่เกี่ยวข้องแล้ว ก็นำเอาแอตทริบิวต์เหล่านั้นมาเขียนเป็นแบบจำลองเชิงความคิดแบบ ER-Diagram ได้ จะเห็นว่า ER-Diagram นั้นเป็น ER-Diagram ของแต่ละโปรแกรมประยุกต์ หมายความว่าแต่ละโปรแกรมก็จะได้ ER-Diagram คนละชุดและเมื่อสร้างเป็นตาราง แต่ละโปรแกรมก็จะมีตารางเป็นของตัวเอง ดังนั้นการจะรวมตารางเหล่านั้นเข้าเป็นฐานข้อมูลใหญ่ก้อนเดียวทำได้โดย อันดับแรกจะต้องทราบชัดเจนก่อนว่าชื่อแอตทริบิวต์, ชนิดของข้อมูลและค่าต่างๆต้องสอดคล้องกัน เช่น อะไรก็ตามที่เป็นสิ่งเดียวกันควรจะใช้ชื่อแอตทริบิวต์เดียวกัน เป็นข้อมูลชนิดเดียวกันและมีค่าภายในเหมือนกัน เป็นต้น จากนั้นจะใช้หลัก 5NF ในการรวมตารางต่างๆ เข้าด้วยกันโดยหากมี Common key รวมกันก็สามารถรวมเป็นตารางเดียวกันได้ เป็นต้น

2.6.2 ฐานข้อมูลกลาง (Common Database)



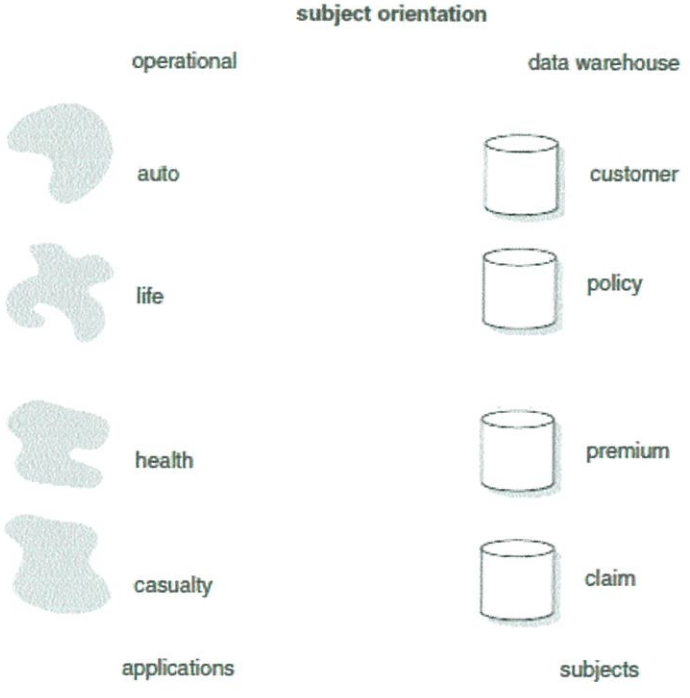
รูป 2.21 การทำฐานข้อมูลกลาง

ในปกติหากหน่วยงานกลางต้องการจะเรียกดูข้อมูลของแต่ละหน่วยงานย่อยที่สนใจเพื่อวิเคราะห์ข้อมูลเกี่ยวกับหน่วยงานเหล่านั้นโดยสามารถประมวลผลข้อมูลได้แบบทันทีทันใด สิ่งที่ดำเนินการทำคือจะต้องโหลดข้อมูลของหน่วยงานย่อยๆเหล่านั้นมาไว้ที่ฐานข้อมูลกลางเฉพาะหน่วยงานที่เราสนใจ จากนั้นเวลาจะตอบคำถามต่างๆก็จะประมวลผลที่ฐานข้อมูลกลางนั้น โดยข้อจำกัดคือข้อมูลที่โหลดมาจากหน่วยงานย่อยจะต้องมีรูปแบบข้อมูลเหมือนๆกันในแต่ละโปรแกรมประยุกต์ของหน่วยงานย่อยถึงจะทำการประมวลได้ ซึ่งแนวคิดที่ว่าต้องการให้ข้อมูลนั้นออนไลน์แบบเรียลไทม์ (Online-realtime) โดยการยุบศูนย์คอมพิวเตอร์ของแต่ละหน่วยงานย่อยแล้วให้หน่วยงานย่อยนั้นต่อจอมาใช้ศูนย์คอมพิวเตอร์เครื่องเดียวที่หน่วยงานกลางนั้นเป็นไปไม่ได้ เนื่องจากแต่ละหน่วยงานย่อยล้วนแล้วแต่มีศูนย์คอมพิวเตอร์ของตัวเอง การจะรวมศูนย์คอมพิวเตอร์ของทุกหน่วยงานย่อยจะทำได้ยากมาก ดังนั้นหลักการที่ควรใช้จึงเป็นการดึงข้อมูลมาจากแหล่งข้อมูลแล้วทำการย้ายข้อมูล (Transfer) โดยการปรับเปลี่ยนชนิดของข้อมูลให้เหมาะสมจากนั้นค่อยโหลดมาเก็บไว้ยังศูนย์คอมพิวเตอร์กลาง เรียกกระบวนการนี้ว่า Extract-Transfer-Load (ETL) ซึ่งแนวคิดในการทำ ETL นี้ทำให้เกิดการทำคลังข้อมูล

2.7 คลังข้อมูล

คลังข้อมูล คือกระบวนการนำข้อมูลที่มีการเก็บรวมรวมเป็นช่วงที่สม่ำเสมอจากหลายๆ แหล่งข้อมูล มารวบรวมและจัดเก็บเป็นแบบจำลองข้อมูลเชิงมิติ (Dimensional Model) หรือรูปแบบข้อมูลเชิงสัมพันธ์ (Relational Model) ก็ได้ เพื่อที่จะนำข้อมูลเหล่านั้นมาวิเคราะห์ในการตัดสินใจเชิงทำธุรกิจหรือทำการวิเคราะห์ข้อมูลในด้านอื่นๆ ซึ่งประโยชน์หลักที่ผู้ใช้งานต้องการจากคลังข้อมูลนั้นก็คือการวิเคราะห์ความเปลี่ยนแปลงที่เชื่อมโยงกันของข้อมูลเพื่อนำประโยชน์ส่วนนี้ไปวางแผนข้อมูลทางธุรกิจสำหรับผู้บริหาร โดยส่วนใหญ่การดึงข้อมูลจากฐานข้อมูลระบบงานประจำวัน จะทำเป็นชุดข้อมูลโดยมีคาบเวลา (period) ที่แน่นอน โดยจะไม่ทำทุกๆครั้งที่เกิดการเปลี่ยนแปลงที่เป็นผลมาจาก Transaction ที่แหล่งข้อมูล

2.7.1 สภาพแวดล้อมของคลังข้อมูล (Data warehouse environment)

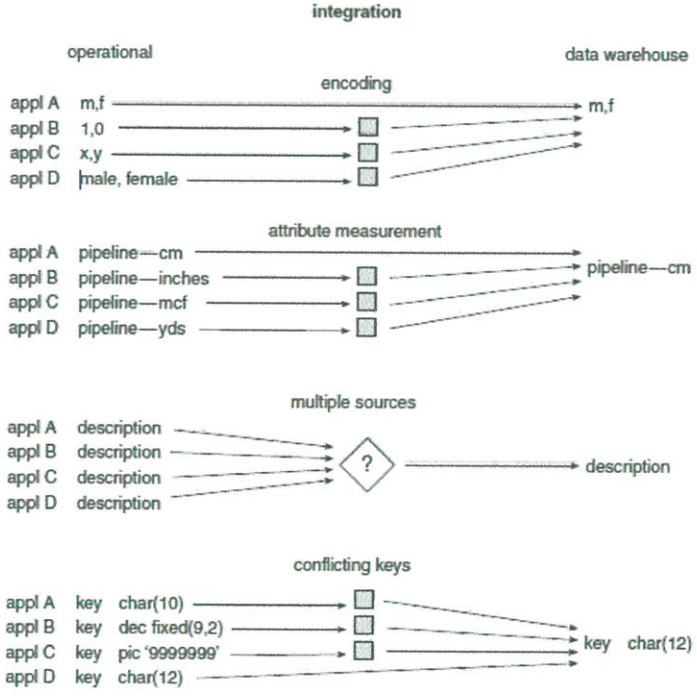


รูป 2.22 ตัวอย่างโปรแกรมประยุกต์และเรื่องที่ต้องการวิเคราะห์ในคลังข้อมูล
(ที่มา W.H Inmon Buiding the Data Warehouse Third Edition)

จากรูปจะแสดงตัวอย่างของฐานข้อมูลระบบงานประจำวันและคลังข้อมูล โดยที่ฐานข้อมูลระบบงานประจำวันนั้นคือโปรแกรมประยุกต์ ซึ่งจากตัวอย่างจะเห็นได้ว่ามีโปรแกรมประยุกต์ที่เกี่ยวกับประกันสุขภาพ, ประกันอุบัติเหตุเป็นต้น ส่วนทางด้านขวามือของรูป 2.26 จะเป็นเรื่องของ

คลังข้อมูล ซึ่งจะแสดงตามหัวข้อที่สนใจ (Subject) เนื่องจากในการวิเคราะห์ข้อมูลนั้นส่วนใหญ่มักจะไม่ถามตามโปรแกรมประยุกต์ แต่จะถามตามเรื่องที่สนใจมากกว่า ตัวอย่างเช่น ลูกค้า (รวมลูกค้าทุกประกัน), Claim, กรมธรรม์ต่างๆ เป็นต้น ซึ่งจะเห็นได้ว่า การออกแบบข้อมูลเพื่อใช้งานในโปรแกรมประยุกต์กับการแยกประเภทข้อมูลตามเรื่องที่สนใจนั้นมีความแตกต่างกัน กล่าวคือตาราง customer ของคลังข้อมูล จะต้องถูกโหลดมาจากทุกตารางที่มีข้อมูลเกี่ยวกับลูกค้าในโปรแกรม ดังนั้นข้อมูลที่จะโหลดมารวมกันจะต้องมีรูปแบบที่เหมือนกันถึงจะรวมกันได้

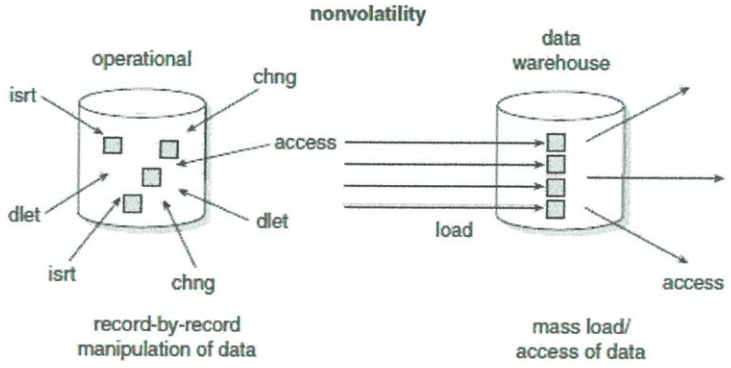
ปัญหาที่เกิดขึ้นคือแต่ละโปรแกรมประยุกต์อาจจะถูกพัฒนาโดยผู้พัฒนาโปรแกรมหลายๆคน ซึ่งเป็นไปได้ยากที่ข้อมูลจากโปรแกรมเหล่านั้นจะมีรูปแบบที่เหมือนกันทุกประการดังตัวอย่าง



รูป 2.23 การรวมข้อมูลจากแต่ละโปรแกรมประยุกต์ไปเป็นเรื่องที่สนใจในคลังข้อมูล
(ที่มา W.H Inmon Buiding the Data Warehouse Third Edition)

จะเห็นได้ว่าจากฐานข้อมูลระบบงานประจำวัน แต่ละแอพพลิเคชั่นนั้นมีรูปแบบของ ชื่อ, ค่าข้อมูล, ชนิดข้อมูลของแอตทริบิวต์ที่ไม่เหมือนกัน ดังนั้นเมื่อนำข้อมูลจากแอตทริบิวต์เหล่านั้นไปรวมกันในคลังข้อมูลจะต้องตกลงกันไว้ก่อนว่าเมื่อดึงข้อมูลไปเป็นข้อมูลในคลังข้อมูลแล้วจะให้เป็นแอตทริบิวต์ตัวกลางอย่างไร ซึ่งจะต้องชัดเจนเป็นอย่างมากในขั้นตอนนี้ ดังนั้นขั้นตอนแรกใน

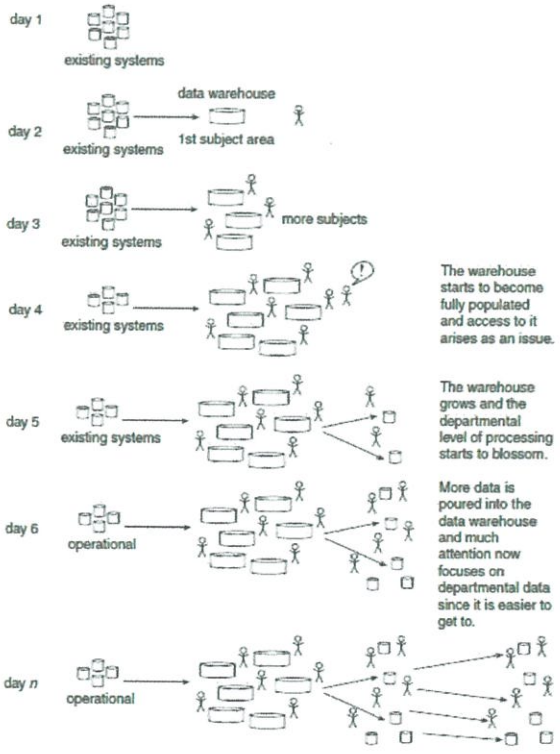
การออกแบบ คลังข้อมูล นั้นจะต้องกำหนดโค้ด (Code) ส่วนกลาง และปลายทางให้เรียบร้อยเสียก่อน ก่อนที่จะจัดซื้อฮาร์ดแวร์ (Hardware) หรือพัฒนาซอฟต์แวร์ (Software) ในส่วนอื่น



รูป 2.24 ความแตกต่างระหว่างฐานข้อมูลและคลังข้อมูล

(ที่มา W.H Inmon Buiding the Data Warehouse Third Edition)

ความแตกต่างระหว่างฐานข้อมูลและคลังข้อมูล คือ ฐานข้อมูลนั้นจะต้องมีการจัดเก็บข้อมูล, การปรับปรุข้อมูล, การลบข้อมูล แต่คลังข้อมูลนั้นไม่จำเป็นต้องมีการดำเนินการที่กล่าวข้างต้น มีแค่เพียงการโหลดและเข้าถึงข้อมูล (Access) ก็เพียงพอแล้ว



รูป 2.25 Day-1 to Day-n pheromenon

(ที่มา W.H Inmon Buiding the Data Warehouse Third Edition)

จากรูปเป็นการแสดง Day-1 to Day-n pheromenon นั่นคือ คลังข้อมูลที่ว่าจะเริ่มมาจาก day-1 คือมีระบบงานปัจจุบัน (Existing system) โดยไม่จำเป็นต้องมีหนึ่งฐานข้อมูลระบบงานประจำวัน อาจจะมีหลายฐานข้อมูลระบบงานประจำวัน ก็เป็นได้แล้วนำเอาการดำเนินการเหล่านั้นมาสร้างเป็นเรื่องที่สนใจครั้งที่ 1 (1st subject area) ใน Day-2 จากนั้นที่ Day-3 เมื่อมีคนเริ่มใช้คลังข้อมูล ก็จะเกิดเรื่องที่สนใจมากขึ้น ทำให้ผู้ใช้คลังข้อมูล เริ่มเกิดการสับสนว่าควรไปหยิบข้อมูลมาจากส่วนไหนกันแน่ที่ตัวเองต้องการ จึงต้องมีการดึงข้อมูลจากคลังข้อมูลไปเป็นตลาดข้อมูล (Data mart) ซึ่งจะมีขนาดเล็กกว่าคลังข้อมูล แต่บางครั้งตาดข้อมูลเพียงชั้นเดียวอาจจะยังใหญ่ไปสำหรับหน่วยงานที่มาใช้งานคลังข้อมูล ทำให้ต้องสร้างตลาดข้อมูลขึ้นมาหลายๆชั้น จนกระทั่งเกิดเป็น Day-n ดังรูป ถึงเวลาผู้ใช้งานก็จะไปหยิบตลาดข้อมูลระดับ (Level) ก่อนหน้าตนมาวิเคราะห์ จะเห็นได้ว่าแผนภาพนี้แสดงให้เห็นอย่างชัดเจนว่าคลังข้อมูลนั้นไม่ได้ออนไลน์กันแบบเรียลไทม์

แนวคิดในปัจจุบันคลังข้อมูลนั้นยังคงเป็นการดึงข้อมูลการดำเนินการจากหลายๆ แหล่งข้อมูลมารวมกัน แต่ไม่แน่ในอนาคตข้างหน้าทุกๆ การดำเนินการทั้งหมดจะอยู่บนเครื่องเดียวกัน และตลาดข้อมูลหรือคลังข้อมูล นั้นจะเป็นวิว (View) ที่เรียกการดำเนินการเหล่านั้นมาแสดง เนื่องจากแนวคิดเก่านั้น การดำเนินกาต่างๆ คือ ฐานข้อมูลเวอร์ชันปัจจุบัน แล้วเมื่อเวลาผ่านไปถึง ช่วงสิ้นสุดอายุขัยของข้อมูล ก็จะดึงข้อมูลเวอร์ชันปัจจุบันนั้นเก็บไว้ เมื่อต้องการจะทำคลังข้อมูลก็เพียงแต่โหลดเวอร์ชันที่เก่าๆ เหล่านั้นตามที่ต้องการมารวมไว้ในที่เดียวกันแล้วทำการวิเคราะห์ แต่หากทำการวิเคราะห์ขั้นตอนการทำคลังข้อมูลทั้งหมดจะพบว่า หากฐานข้อมูลที่ใช้เป็นฐานข้อมูลเชิงเวลาแล้ว การดำเนินการนั้นเป็นเวอร์ชันปัจจุบัน หากต้องการจะทำคลังข้อมูลก็เพียงแค่ค้นหา ย้อนหลังตราบเท่าที่ต้องการแล้วใช้ Machine ดูเฉพาะที่เราต้องการจะดู ก็จะสามารถทำคลังข้อมูลได้แล้ว อีกทั้งยังไม่ต้องทำการแปลงชนิดข้อมูลของแอตทริบิวต์อีกด้วย เนื่องจากทั้งหมดนั้นอยู่บนฐานข้อมูลเดียวกัน

2.7.2 คุณลักษณะของ คลังข้อมูล

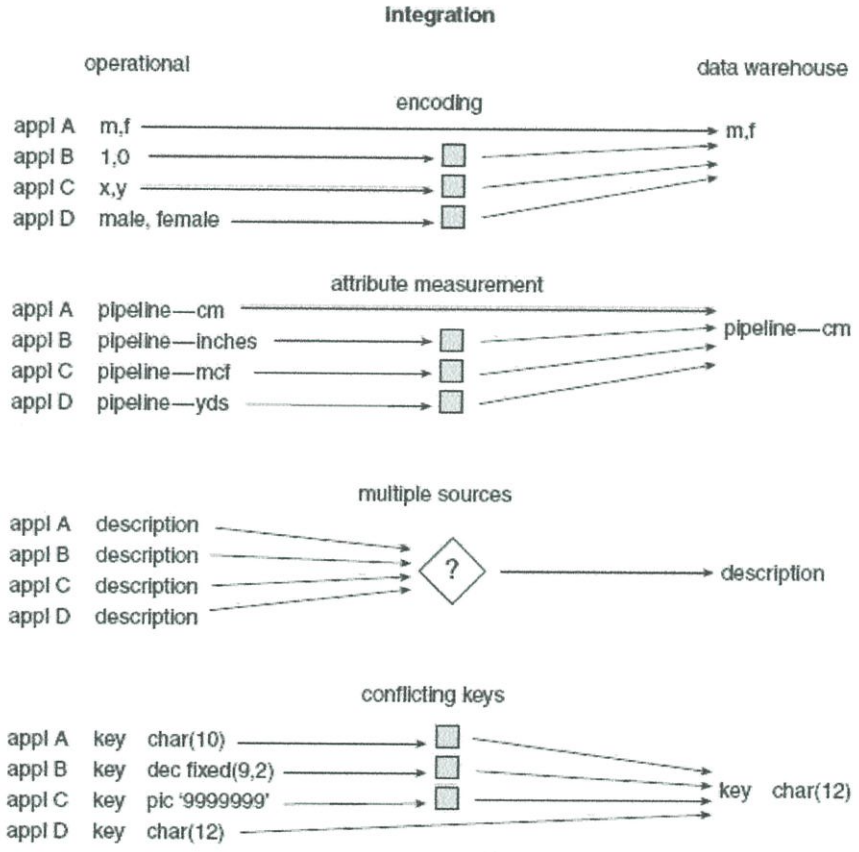
คลังข้อมูล มีคุณลักษณะที่สำคัญ 4 ประการดังนี้

2.7.2.1 Subject-Oriented

คือการจัดกลุ่มข้อมูลตามประเด็นหลักขององค์กรที่สนใจจะนำมาสร้างเป็น คลังข้อมูล ซึ่งข้อมูลจะถูกสร้างขึ้นมาจากหัวข้อ (Subject) ธุรกิจที่สนใจ เช่น ข้อมูลลูกค้า ข้อมูลสินค้า หรือข้อมูลยอดขาย ซึ่งข้อมูลที่สร้างขึ้นจะประกอบด้วยส่วนข้อมูลที่เกี่ยวข้องกับหัวข้อนั้นๆ เท่านั้น เพื่อที่จะประกอบกันเป็นข้อมูลที่ใช้ในการวิเคราะห์

2.7.2.2 Integrated

คือการจัดข้อมูลต่างรูปแบบให้อยู่ในรูปแบบเดียวกัน ซึ่งจะต้องมีความสอดคล้องไม่ขัดแย้งกันในคลังข้อมูลเดียวกัน เนื่องจากข้อมูลถูกรวบรวมมาจากแหล่งข้อมูลต่างๆ อีกทั้งยังมีรูปแบบข้อมูลที่หลากหลาย ดังนั้นก่อนจะทำการสร้างคลังข้อมูล สิ่งที่ต้องทำเป็นอันดับแรกคือการกำหนดค่าตัวแปรของฐานข้อมูลให้เหมือนกันเป็นหนึ่งเดียวก่อน

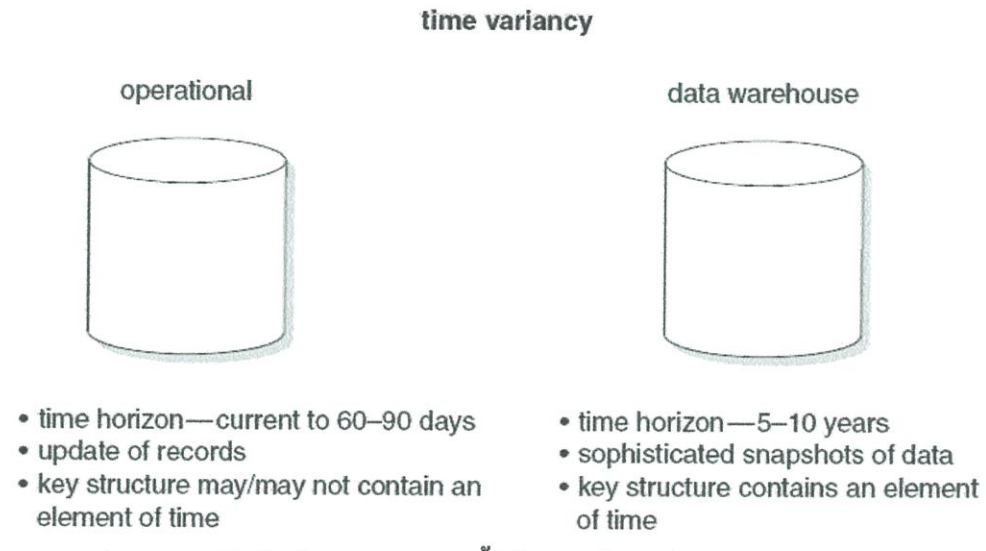


รูป 2.26 การรวบรวมข้อมูลก่อนที่จะนำเข้าสู่คลังข้อมูล

(ที่มา W.H Inmon Buiding the Data Warehouse Third Edition)

2.7.2.3 Time-Variant

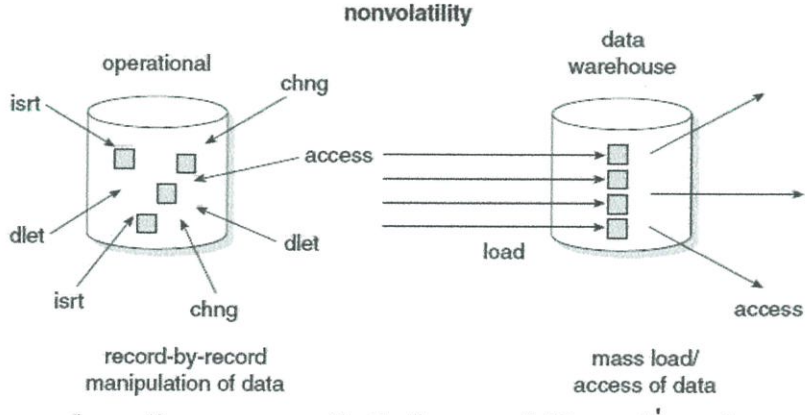
คลังข้อมูล เป็นการนำข้อมูลที่เก็บไว้เป็นระยะเวลานาน เช่น ข้อมูลในช่วง 5 ปีที่แล้ว เพื่อนำมาทำนายแนวโน้มความสัมพันธ์ของข้อมูลในแต่ละช่วงเวลา (Time granularity) ตามที่ผู้ใช้งานกำหนด ซึ่งตัวกำหนดช่วงเวลานั้นโดยส่วนมากในทางคลังข้อมูล มักจะใช้มุมมองของเวลาเป็นตัวระบุช่วงเวลา ซึ่งต่างจากการเก็บข้อมูลในฐานข้อมูลระบบงานประจำวัน คือ ฐานข้อมูลระบบงานประจำวัน จะเก็บแค่ข้อมูลที่เป็นจริงเฉพาะในปัจจุบันเท่านั้น (Current value) ในขณะที่โครงสร้างของคลังข้อมูลจะต้องมีองค์ประกอบของเวลาเข้ามารวมอยู่ด้วย



รูป 2.27 การจัดเก็บข้อมูลแบบระยะสั้นในฐานข้อมูลประจำวัน และระยะยาวในคลังข้อมูล (ที่มา W.H Inmon Buiding the Data Warehouse Third Edition)

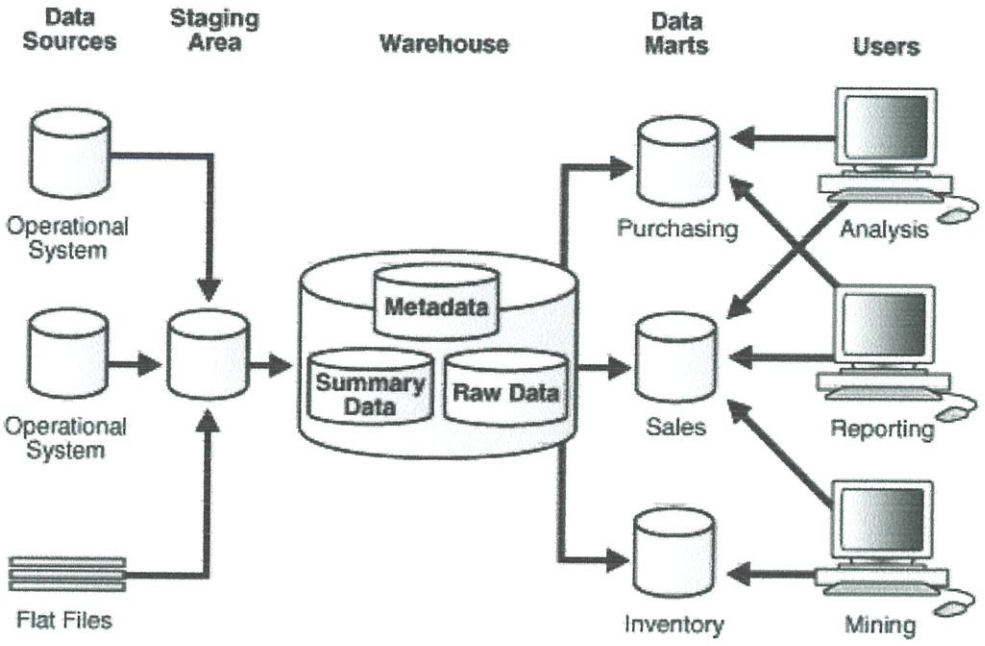
2.7.2.4 Nonvolatile

เมื่อมีการเปลี่ยนแปลงข้อมูลที่มีการโหลด (Load) ข้อมูลจากฐานข้อมูลระบบงานประจำวันไปยังคลังข้อมูล ข้อมูลในคลังข้อมูลนั้นจะไม่ได้มีการเปลี่ยนแปลงอีกหลังจากที่ถูกโหลด โดยกระบวนการโหลด (Load) ข้อมูลจากฐานข้อมูลระบบงานประจำวัน นั้นจะเรียกกระบวนการนี้ว่า การทำ Extract Transfer Load (ETL)



รูป 2.28 ลักษณะของการจัดเก็บข้อมูลแบบไม่มีการเปลี่ยนแปลง (ที่มา W.H Inmon Buiding the Data Warehouse Third Edition)

2.7.3 องค์ประกอบของคลังข้อมูล



รูป 2.29 องค์ประกอบของคลังข้อมูล

(ที่มา Oracle Database Data Warehousing Guide)

2.7.3.1 Date Staging Area

เป็นส่วนที่ให้ข้อมูลมาพักไว้ก่อนที่จะเข้าสู่คลังข้อมูล ซึ่งในส่วนนี้จะมีกระบวนการหลักคือการคัดเลือกข้อมูล, การรวบรวมข้อมูล, การทำข้อมูลให้เป็นมาตารฐาน ซึ่งส่วนพักข้อมูลนั้นจะเป็นทั้งส่วนที่เก็บข้อมูล และเป็นส่วนของการะบวนการ ETL

- Extract: การดึงข้อมูล คือการพยายามเข้าไปอ่านและพยายามเข้าถึงแหล่งข้อมูล
- Transformation: จะเป็นกระบวนการที่มีการคัดเลือก, จัดข้อมูลให้ถูกต้องและมีการรวมข้อมูลที่มาจากหลายแหล่งเข้าด้วยกัน
- Load: เป็นกระบวนการที่จะโหลดข้อมูลเข้าไปยังคลังข้อมูล ซึ่งจะโหลดเข้าไปจัดเก็บไว้ยัง Dimension table และ Fact table

2.7.3.2 Data Presentation Area

เป็นส่วนที่ต้องออกแบบว่าจะจัดเก็บข้อมูลอย่างไร โดยจะมีตลาดข้อมูลเป็นเสมือนวิวของคลังข้อมูลอีกทีหนึ่ง ซึ่งจุดประสงค์ของตลาดข้อมูลนั้นมีไว้เพื่อช่วยคัดกรองข้อมูลของแต่ละแผนกในองค์กรที่ไม่เกี่ยวข้องกัน อีกทั้งยังสามารถใช้ในการกำหนดสิทธ์ในการที่จะเข้าถึงข้อมูลของแต่ละแผนกในองค์กรได้อีกด้วย โดยที่ตลาดข้อมูลแต่ละตลาดนั้นอาจจะมีการใช้

Dimension Table ร่วมกันได้ โดยในส่วนการเก็บข้อมูลนั้นจะมีหลักในการออกแบบที่เกี่ยวข้องสองอย่างคือ Fact table และ Dimension table

2.7.3.3 Data Mart

ตลาดข้อมูลเป็นคลังข้อมูลขนาดเล็กที่คัดกรองเฉพาะข้อมูลที่เกี่ยวข้องกับสิ่งที่สนใจเท่านั้น โดยข้อมูลนั้นจะแยกย่อยจาก คลังข้อมูล อีกทีหนึ่ง เพื่อนำไปใช้สนับสนุนการทำงานของแต่ละแผนก ซึ่งเป็นข้อมูลสำหรับการดำเนินการทางธุรกิจเพียงธุรกิจเดียว (Single Business Process) เช่นดูเฉพาะเรื่องการขายของบริษัทเท่านั้น (โดยทั่วไปจะให้ 1 Mart ต่อ 1 Business Process) ซึ่งการทำตลาดข้อมูลนั้นยังช่วยในเรื่องของความรวดเร็วในการสืบค้นข้อมูล (Performance) อีกด้วย เนื่องจากในงาน คลังข้อมูล นั้น จะต้องใช้ข้อมูลปริมาณมากในการวิเคราะห์การเปลี่ยนแปลง ดังนั้นการแยกข้อมูลสำหรับแต่ละบิสิเนสโปรเซสจึงเป็นเรื่องที่สำคัญ

2.7.3.4 Meta Data

เนื่องจาก คลังข้อมูลนั้นเป็นพื้นที่สำหรับเก็บข้อมูลขนาดใหญ่ ดังนั้นย่อมมีความยุ่งยากกว่าการบริหารข้อมูลที่มีขนาดเล็กหรือปานกลาง ดังนั้นจึงจำเป็นต้องสร้าง Mete data ขึ้นมาเพื่ออธิบายข้อมูลที่อยู่ในคลังข้อมูลอีกทีหนึ่ง ซึ่งใน Mete data จะเก็บข้อเท็จจริงต่างๆ ที่เกี่ยวข้องกับข้อมูลในทุกแง่มุมเอาไว้ ทั้งนี้เพื่อสองวัตถุประสงค์คือ

1) เพื่ออธิบายความหมายหรือคำจำกัดความของข้อมูล

2) เพื่อใช้เป็นข้อมูลสำหรับการดำเนินงานต่างๆกับข้อมูลตามกระบวนการของคลังข้อมูล

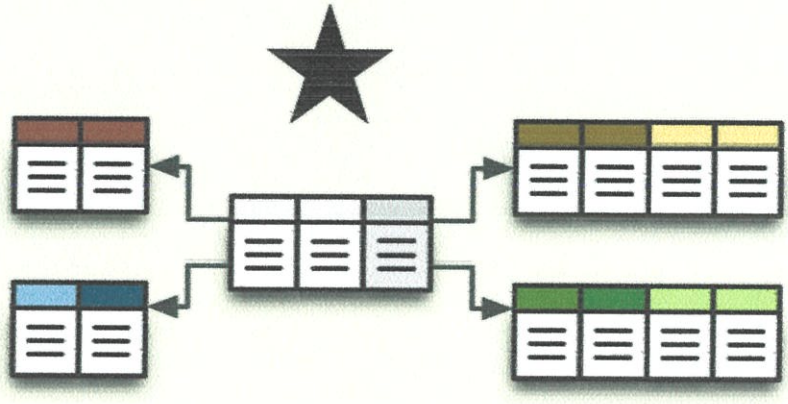
2.7.4 แบบจำลองข้อมูลสำหรับคลังข้อมูล (Data Model for Data Warehouse)

2.6.4.1 Dimensional Data Model

นิยาม

- **Measures** หมายถึงข้อมูลที่ต้องการวัด ทั้งในเชิงปริมาณ และเชิงคุณภาพของสิ่งใดสิ่งหนึ่ง เช่น ยอดขายรวม กำไร ค่าธรรมเนียม เป็นต้น โดยชนิดของข้อมูลเป็นตัวเลขเสมอ
- **Dimension** หมายถึง ข้อมูลที่เป็นมุมมองให้แก่ Measure เพื่อประโยชน์ในการวิเคราะห์ข้อมูล เช่น เวลา จังหวัด อำเภอ เป็นต้น
- **Facts** หมายถึงชุดของข้อมูลที่เกิดจากการจับคู่กันของ Dimension และ Measure ที่ทำให้เกิดค่าหนึ่งที่มีความหมายสามารถวัดได้ และบอกเล่าข้อเท็จจริงอย่างใดอย่างหนึ่ง

2.7.5 โครงสร้างแบบดวงดาว (Star Schema)

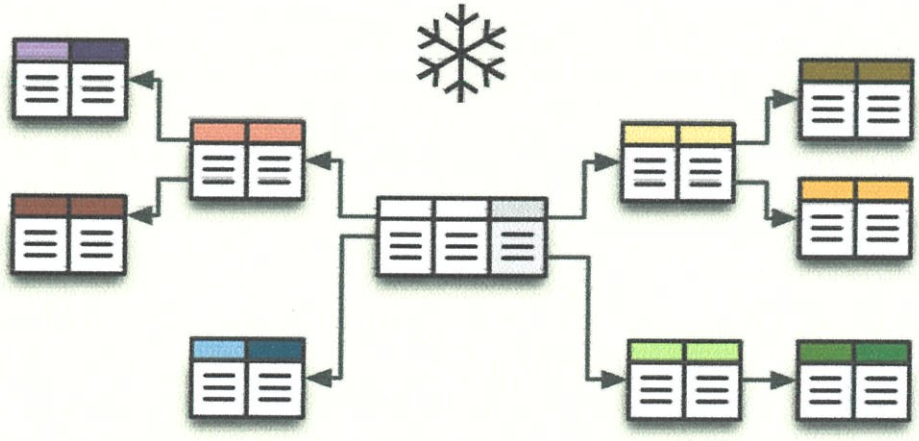


รูป 2.30 รูปแบบของโครงสร้างแบบดวงดาว

(ที่มา https://pythonhosted.org/cubes/backends/sql.html)

โครงสร้างแบบดวงดาว หมายถึง แบบจำลองข้อมูลเชิงมิติที่มี Fact table ขนาดใหญ่เพียงหนึ่งเดียวอยู่ตรงกลางและมี Dimension table จำนวนหนึ่งอยู่รอบข้างเพื่อกำหนดมุมมองที่จะมีต่อค่าที่ต้องการวัดใน Fact table นั้น โดยจำนวนมุมมองที่มองได้จะเท่ากับจำนวนของ Dimension table ที่รายรอบอยู่ และเท่ากับจำนวน Dimension ที่เชื่อมต่อโดยตรงจาก Fact table [20]

2.7.6 โครงสร้างแบบเกล็ดหิมะ (Snowflake Schema)



รูป 2.31 รูปแบบของโครงสร้างแบบเกล็ดหิมะ

(ที่มา https://pythonhosted.org/cubes/backends/sql.html)

โครงสร้างแบบเกล็ดหิมะ หมายถึง แบบจำลองข้อมูลเชิงมิติที่มี Fact table ขนาดใหญ่เพียงหนึ่งเดียวอยู่ตรงกลาง และมี Dimensional table จำนวนหนึ่งรายรอบอยู่เพื่อที่จะกำหนดมุมมองที่จะมีต่อค่าที่ต้องการจะวัดใน Fact table นั้นโดยจำนวนมุมมองที่มองได้จะเท่ากับจำนวน Dimension table ที่รายรอบอยู่ และจะมากกว่าจำนวน Dimension ที่เชื่อมต่อโดยตรงกับ Fact table โดยที่ Dimension ที่ไม่ได้เชื่อมต่อโดยตรงกับ Fact table จะมีความสัมพันธ์กับ Dimension ตัวอื่นๆจึง สามารถสรุปได้ว่า โครงสร้างแบบเกล็ดหิมะคือรูปแบบการเขียนโครงสร้างแบบดวงดาวที่จัด Dimension ให้อยู่ในรูป Normal form [20]

2.7.7 ประเภทของแหล่งข้อมูล (Data Source)

- Fact source คือ แหล่งข้อมูลที่จะโหลดเข้าคลังข้อมูล โดยนำมาทำเป็น Fact table ภายในคลังข้อมูล ข้อมูลภายในแหล่งข้อมูลสามารถเป็นได้ทั้งข้อมูลเชิงเวลา (Temporal data) หรือไม่ใช่ข้อมูลเชิงเวลา (Non-temporal data)
- Dimension source คือ แหล่งข้อมูลที่จะโหลดเข้าคลังข้อมูล โดยนำมาทำเป็น Dimension table ภายในคลังข้อมูล ข้อมูลภายในแหล่งข้อมูลสามารถเป็นได้ทั้งข้อมูลเชิงเวลา (Temporal data) หรือไม่ใช่ข้อมูลเชิงเวลา (Non-temporal data)

2.7.8 Slowly Changing Dimension

Slowly Changing Dimension นั้นเป็นปัญหาในการทำคลังข้อมูลที่เกิดขึ้นเมื่อ Dimension table มีการปรับปรุงข้อมูล ดังนั้นจึงมีรูปแบบในการแก้ไขปัญหาที่เกิดขึ้น 4 รูปแบบด้วยกัน ซึ่งแต่ละแบบนั้นมีข้อดีข้อเสียแตกต่างกันไป ขึ้นอยู่กับผู้ออกแบบระบบว่าจะแก้ไขกับปัญหานี้ได้อย่างไร

1) การเขียนทับข้อมูลเก่า (Overwriting the old value)

เป็นการจัดการกับข้อมูลเมื่อมีการปรับปรุงข้อมูลใน Dimension table จะทำการเขียนทับข้อมูลเวอร์ชันก่อนหน้า

Before the change:

Customer_ID	Customer_Name	Customer_Type
1	Cust_1	Corporate

After the change:

Customer_ID	Customer_Name	Customer_Type
1	Cust_1	Retail

รูป 2.32 การจัดการปัญหา Slowly Changing Dimension รูปแบบที่ 1

(ที่มา http://datawarehouse4u.info/SCD-Slowly-Changing-Dimensions.html)

หากเลือกวิธีนี้ในการจัดการกับข้อมูลที่มีการปรับปรุงจะไม่สามารถติดตามข้อมูลในอดีตได้ แต่มีข้อดีคือไม่เปลืองพื้นที่จัดเก็บ (Space) เหมาะสำหรับงานที่ไม่ต้องการความแม่นยำในการวิเคราะห์ข้อมูล และไม่ต้องการที่จะดูการเปลี่ยนแปลงของข้อมูลในอดีต

2) เพิ่มแถวใหม่เมื่อมีการปรับปรุงข้อมูล (Creating a new additional record)

วิธีนี้จะเพิ่มคอลัมน์มาใช้สำหรับเก็บข้อมูล 3 คอลัมน์คือ วันที่เริ่มต้น (Start date), วันที่สิ้นสุด (End date) และค่าบ่งบอกสถานะปัจจุบัน (Current flag) เพื่อบ่งบอกช่วงเวลาที่ Fact นั้นเป็นจริงและสถานะว่า Fact นั้นยังเป็นจริงอยู่อีกหรือไม่ ซึ่งหากข้อมูลมีการปรับปรุงจะทำการเพิ่มแถวใหม่ลงไปและตั้งค่าบ่งบอกสถานะปัจจุบันให้เป็นจริง (True) และค่าบ่งบอกสถานะปัจจุบันของแถวก่อนหน้าให้เป็นไม่จริง (False)

Before the change:

Customer_ID	Customer_Name	Customer_Type	Start_Date	End_Date	Current_Flag
1	Cust_1	Corporate	22-07-2010	31-12-9999	Y

After the change:

Customer_ID	Customer_Name	Customer_Type	Start_Date	End_Date	Current_Flag
1	Cust_1	Corporate	22-07-2010	17-05-2012	N
2	Cust_1	Retail	18-05-2012	31-12-9999	Y

รูป 2.33 การจัดการปัญหา Slowly Changing Dimension รูปแบบที่ 2

(ที่มา http://datawarehouse4u.info/SCD-Slowly-Changing-Dimensions.html)

ข้อดีคือสามารถติดตามการเปลี่ยนแปลงของข้อมูลได้ครบถ้วน ส่วนข้อเสียคือหากดูแลไม่ดีตารางจะโตไวมาก

3) เพิ่มคอลัมน์ใหม่ (Adding a new column)

เพิ่มคอลัมน์สำหรับเก็บค่าปัจจุบัน (Current) และค่าก่อนหน้า (Previous) ของข้อมูลก่อนที่จะมีการปรับปรุงเอาไว้เพื่อให้มีการติดตามเวอร์ชันก่อนที่จะมีการปรบปรุงได้ 1 ครั้ง

Before the change:

Customer_ID	Customer_Name	Current_Type	Previous_Type
1	Cust_1	Corporate	Corporate

After the change:

Customer_ID	Customer_Name	Current_Type	Previous_Type
1	Cust_1	Retail	Corporate

รูป 2.34 การจัดการปัญหา Slowly Changing Dimension รูปแบบที่ 3

(ที่มา http://datawarehouse4u.info/SCD-Slowly-Changing-Dimensions.html)

ข้อดีคือไม่เปลืองเนื้อที่ในการเก็บข้อมูลที่มีการเปลี่ยนแปลงและสามารถติดตามการเปลี่ยนแปลงของข้อมูลได้บ้างส่วน แต่ข้อเสียคือหากข้อมูลมีการเปลี่ยนแปลงมาก ก็ไม่สามารถติดตามข้อมูลได้ทุกเวอร์ชันของข้อมูลนั้นๆ

4) ใช้ตารางเก็บค่าในอดีต (Using historical table)

รูปแบบที่ 4 นั้นจะใช้ตารางที่เก็บค่าในอดีตของข้อมูลในการที่จะติดตามการเปลี่ยนแปลงเมื่อข้อมูลมีการปรับปรุงโดยที่ตาราง Dimension จะเก็บเฉพาะเวอร์ชันปัจจุบัน เท่านั้น

Current table:

Customer_ID	Customer_Name	Customer_Type
1	Cust_1	Corporate

Historical table:

Customer_ID	Customer_Name	Customer_Type	Start_Date	End_Date
1	Cust_1	Retail	01-01-2010	21-07-2010
1	Cust_1	Oher	22-07-2010	17-05-2012
1	Cust_1	Corporate	18-05-2012	31-12-9999

รูป 2.35 การจัดการปัญหา Slowly Changing Dimension รูปแบบที่ 4

(ที่มา http://datawarehouse4u.info/SCD-Slowly-Changing-Dimensions.html)

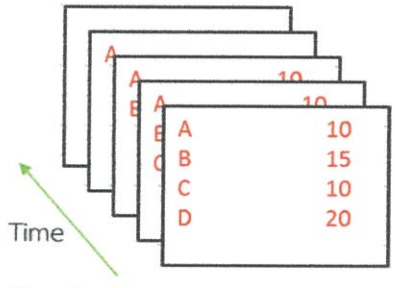
ข้อดีคือเราสามารถติดตามการเปลี่ยนแปลงของข้อมูลได้ทุกเวอร์ชันและจะไม่เปลืองเนื้อที่ในการจัดเก็บในฐานข้อมูลอีกด้วยเนื่องจากสามารถดึงข้อมูลตารางที่เก็บข้อมูลในอดีตไปเก็บไว้ในไฟล์ ได้

บทที่ 3
การออกแบบและพัฒนาซอฟต์แวร์

ในบทนี้จะกล่าวถึงขั้นตอนในการประยุกต์ใช้งาน Oracle Flashback Data Archive ในการสร้างคลังข้อมูลที่ทำการสรุปข้อมูลจากแหล่งข้อมูลชนิดต่างๆ ซึ่งจัดเก็บอยู่ใน ฐานข้อมูลระบบงานประจำวัน ในที่นี้ใช้เป็นระบบการจัดการฐานข้อมูล Oracle ประกอบไปด้วยแหล่งข้อมูลที่ไม่เป็นข้อมูลเชิงเวลา (Non-temporal data source) และแหล่งข้อมูลที่เป็นข้อมูลเชิงเวลา (Temporal data source) โดยแหล่งข้อมูลทั้งสองแบบนั้น มีวิธีสรุปข้อมูลมาเป็นคลังข้อมูลแตกต่างกันออกไป

3.1 Oracle Flashback Data Archive

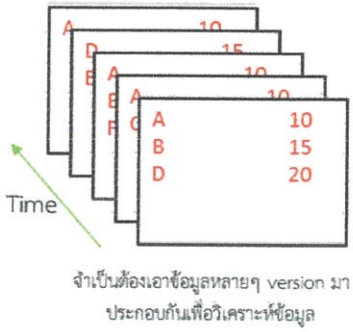
ลักษณะข้อมูลใน Oracle Flashback Data Archive ที่จะนำมาใช้ในคลังข้อมูลจะประกอบไปด้วยข้อมูลสองลักษณะดังนี้



รูป 3.1 ลักษณะการเก็บข้อมูลใน Oracle Flashback Data Archive รูปแบบที่ 1

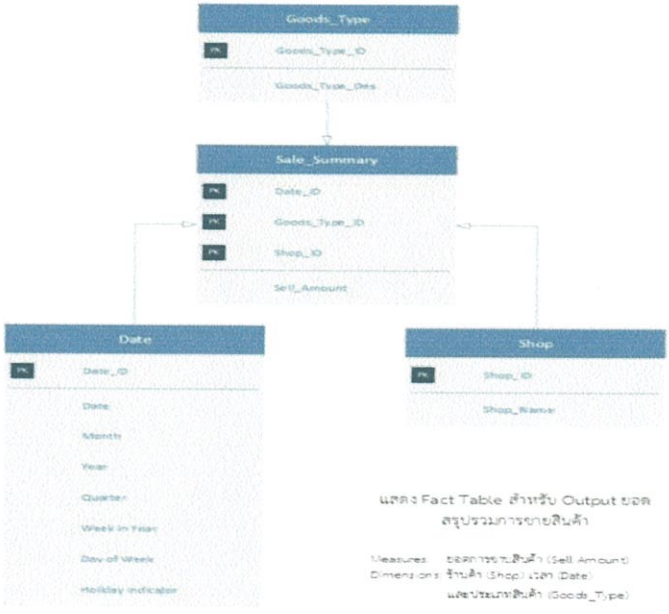
ลักษณะแรกนั้นจะเป็นข้อมูลที่มีการจัดเก็บเพียงอย่าเดียว ข้อมูลลักษณะเช่นนี้ หากจะนำมาใช้ในคลังข้อมูล เพียงแค่หยิบนำเอาเวอร์ชันล่าสุดมาใช้ เนื่องจากข้อมูลเวอร์ชันล่าสุดนั้นจะเป็นข้อมูลที่ครอบคลุมข้อมูลทุกๆเวอร์ชันทั้งหมดในอดีต เพราะว่าข้อมูลไม่มีการปรับปรุงเลย ซึ่งข้อมูลในลักษณะเช่นนี้มักจะปรากฏในคลังข้อมูลที่มี Fact source ไม่เป็นข้อมูลเชิงเวลาและมี Dimension source เป็นข้อมูลเชิงเวลาหรือไม่เป็นข้อมูลเชิงเวลาก็ได้ อีกทั้งความสามารถของ Oracle Flashback Data Archive ยังเป็นการเพิ่มขีดความสามารถให้แก่การทำคลังข้อมูลอีกด้วย โดยสามารถเลือกได้ว่าผู้ใช้ต้องการเวอร์ชันล่าสุดของข้อมูล ณ เวลาไหน หมายความหากเป็นฐานข้อมูลทั่วไป ถ้าผู้ใช้ต้องการทำคลังข้อมูล ณ ช่วงเวลา 1 เดือนก่อนหน้า เราจะต้องหยิบเอาเวอร์ชันที่มีอยู่ในฐานข้อมูลมาพักไว้ใน Data staging area จากนั้นทำการกรองแถวที่ไม่เกี่ยวข้องออก ซึ่งหากเป็นคลังข้อมูลที่สร้างขึ้นจาก Oracle Flashback Archive สามารถใช้ Oracle Flashback Query ไปหยิบข้อมูลเวอร์ชัน

1 เดือนก่อนหน้ามาได้เลย แล้วสามารถนำข้อมูลนั้นไปใช้ต่อได้ โดยที่ไม่จำเป็นต้องผ่าน Data staging area



รูป 3.2 ลักษณะการเก็บข้อมูลใน Oracle Flashback Data Archive รูปแบบที่ 2

ข้อมูลลักษณะที่สองนั้นจะเป็นข้อมูลที่มีการปรับปรุงอยู่เสมอ ดังนั้นผู้ใช้ไม่สามารถหยิบนำเอาเวอร์ชันปัจจุบันมาใช้งานได้เลย เนื่องจากข้อมูลนั้นอาจไม่สอดคล้องกับข้อมูลที่เคยเป็นอยู่ในอดีต ซึ่งข้อมูลในลักษณะนี้มักปรากฏกับ Dimension table เสียเป็นส่วนใหญ่ ตัวอย่างเช่น ผู้ใช้ต้องการทำคลังข้อมูลของการขายสินค้า โดยมีโครงสร้างแบบดวงดาว ดังนี้



รูป 3.3 โครงสร้างแบบดวงดาวของการขาย

จากโครงสร้างแบบดวงดาวข้างต้น ประกอบด้วยตาราง Sale_summary ไว้สำหรับเก็บรายการการซื้อขายที่เกิดขึ้น และมีตาราง Goods_type, Shop และ Date เป็น Dimension ของโครงสร้างแบบดวงดาวซึ่งหากข้อมูลใน Dimension table มีการเปลี่ยนแปลง ฐานข้อมูลทั่วไปจะไม่สามารถให้

คำตอบที่สอดคล้องได้ ตัวอย่างเช่นหาก ณ วันที่ 1 มกราคม ได้มีการซื้อขายและบันทึกข้อมูลอ้างอิงกับ Dimension table ของวันที่ 1 มกราคม ต่อมา วันที่ 31 มกราคมได้มีการปรับเปลี่ยนข้อมูลใน Dimension ตัวอย่างเช่น ลบแถวของ Shop เนื่องจากมีการปิดกิจการลง หรือเปลี่ยนความหมายของ goods type ดังนั้นในฐานข้อมูลทั่วไป หากทำการดึงข้อมูลของฐานข้อมูลทุกๆ สิ้นเดือน เราจะได้ sale_summary ที่มีข้อมูลไม่ถูกต้อง เนื่องจากเมื่อวันที่ 1 ถึงวันที่ 30 เราใช้ Dimension table เป็นเวอร์ชันก่อนที่จะมีการปรับปรุงข้อมูล ดังนั้นหากดึงข้อมูลวันที่ 31 เราจะได้ Dimension ของวันที่ 31 เท่านั้นและถ้าเราเอาข้อมูลตรงนี้ไปสร้างคลังข้อมูลก็จะได้คลังข้อมูลที่มีข้อมูลผิดเพี้ยนไป แต่หากคลังข้อมูลที่สร้างขึ้นนั้นเป็นคลังข้อมูลที่มีแหล่งข้อมูลมาจาก Oracle Flashback Data Archive เราสามารถแก้ไขปัญหาตรงนี้ได้โดยไม่จำเป็นต้องออกแบบฐานข้อมูลให้มารองรับปัญหาตรงนี้ กล่าวได้ว่าสามารถใช้ความสามารถที่ Oracle Flashback Data Archive มีเพื่อใช้แก้ปัญหานี้ได้เลย

3.2 Slowly Changing Dimensions ใน Oracle Flashback Data Archive

จากปัญหาที่ได้กล่าวไว้ข้างต้น การที่ Dimension table มีการเปลี่ยนแปลง เราสามารถใช้ Oracle Flashback Data Archive แก้ปัญหาได้ดังต่อไปนี้

ในขั้นแรกผู้ออกแบบระบบจะต้องกำหนดก่อนว่าข้อมูลที่จะหยิบมาทำคลังข้อมูลจาก Oracle Flashback Data Archive นั้นต้องการความละเอียดเท่าใด ทั้งนี้ความละเอียดของการหยิบขึ้นมาขึ้นอยู่กับว่าข้อมูลใน Dimension table มีการเปลี่ยนแปลงบ่อยเพียงใด ซึ่งตรงนี้ Oracle Flashback Data Archive สามารถดึงข้อมูลจากแหล่งข้อมูลที่มีการเปลี่ยนแปลงได้ถึงระดับวินาที

เมื่อกำหนดได้แล้วว่าจะต้องการความละเอียดในการติดตามการเปลี่ยนแปลงของ Dimension table เท่าใด สิ่งที่ต้องทำถัดมาคือใช้ Oracle Flashback Query เพื่อไปหยิบเอาเวอร์ชันต่างๆมาจาก Oracle Flashback Data Archive ตามที่ผู้ออกแบบกำหนด ดังตัวอย่างต่อไปนี้

EmpID	Name	Dept.	
101	A	1	01-Jun-15
102	B	4	

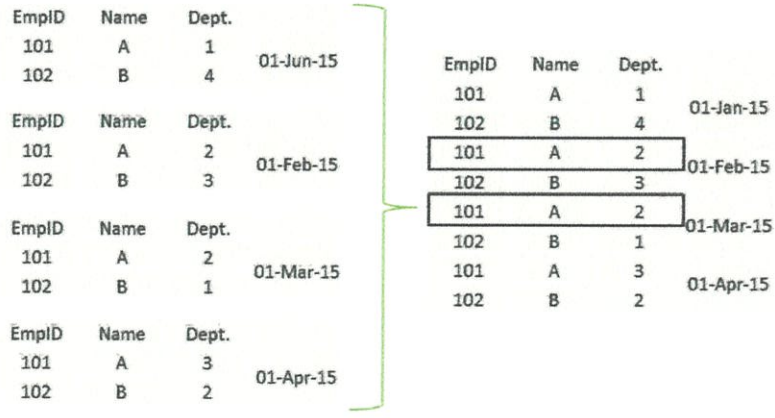
EmpID	Name	Dept.	
101	A	2	01-Feb-15
102	B	3	

EmpID	Name	Dept.	
101	A	2	01-Mar-15
102	B	1	

EmpID	Name	Dept.	
101	A	3	01-Apr-15
102	B	2	

รูป 3.4 ข้อมูลของ Dimension Table ที่มีการเปลี่ยนแปลง

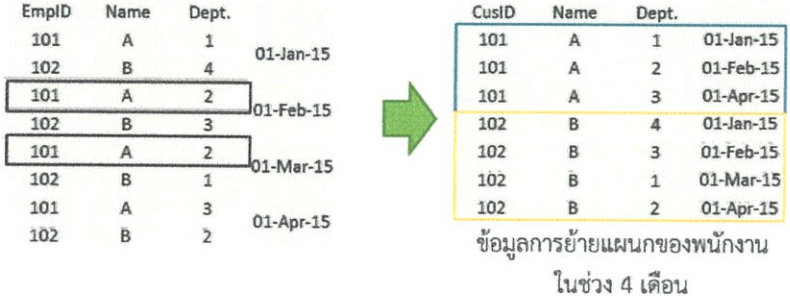
จากรูป จะเห็นได้ว่าตาราง Employee นั้นได้มีการปรับปรุงทุกๆเดือน ดังนั้นจึงจำเป็นต้องสร้าง Dimension table ที่เก็บการเปลี่ยนแปลงไว้ทั้งหมดเอาไว้ในตารางเดียวกัน พร้อมทั้งทำ Timestamp เอาไว้ว่าข้อมูลชุดนั้นเป็นของช่วงเวลาใด ดังตัวอย่างต่อไปนี้



รูป 3.5 การใช้ Oracle Flashback Query ในการหยิบเอาข้อมูลใน ตาราง Employee ของทุกๆเดือนมารวมกัน

จะเห็นได้ว่าเมื่อนำข้อมูลมารวมกันเป็น 1 ตารางแล้วจะเกิดความซ้ำซ้อนกันของข้อมูลเกิดขึ้น ดังนั้นจะต้องทำการตัดข้อมูลที่ซ้ำซ้อนออก ซึ่งกระบวนการนี้ไม่ได้มีความยุ่งยากแต่อย่างใด เนื่องจากระบบการจัดการฐานข้อมูล Oracle เป็นฐานข้อมูลที่ใช้แบบจำลองรีเลชัน (Relational model) ในการนำเสนอข้อมูลระดับโลจิเคิล (Logical) ดังนั้นระบบการจัดการฐานข้อมูล Oracle จึงใช้ภาษาเอสคิวแอลในการปฏิบัติกับฐานข้อมูล หนึ่งในคุณสมบัติของภาษาเอสคิวแอลคือ ตัวภาษา

จะคาดหวังให้ผลลัพธ์ที่ได้ออกมายังคงเป็นรีเลชัน (Relation) ซึ่งคุณสมบัติแถวซ้ำกัน ใน 1 ตาราง ไม่จัดเป็นคุณสมบัติของรีเลชันดังนั้นระบบการจัดการฐานข้อมูล Oracle จึงช่วยจัดการแถวที่ซ้ำกันที่ได้มาจากการใช้ Oracle Flashback Query



รูป 3.6 ผลลัพธ์จากการใช้ Oracle Flashback Query

จะเห็นได้ว่าการดูแลปัญหา Slowly changing dimension ของคลังข้อมูลที่มีแหล่งข้อมูลมาจาก Oracle Flashback Data Archive นั้นสามารถทำได้อย่างง่ายดาย

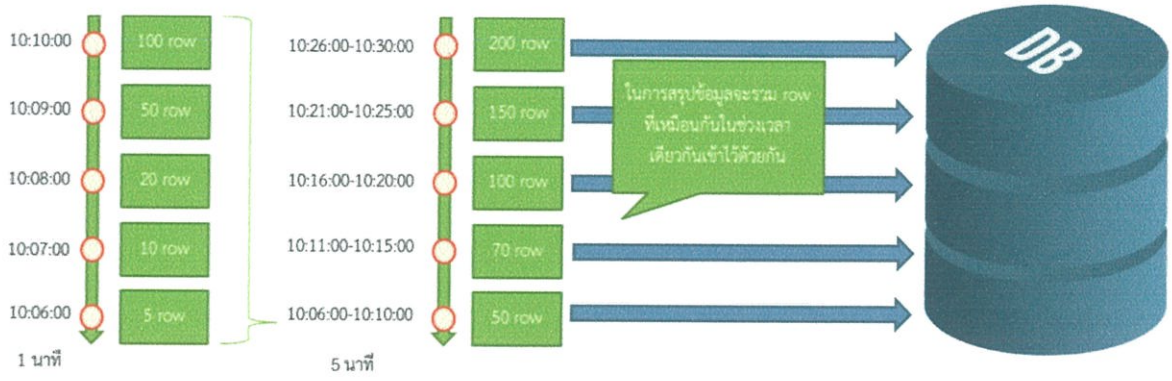
3.3 การสรุปข้อมูลที่เป็นแหล่งข้อมูลสำหรับการทำคลังข้อมูลเมื่อเวลาผ่านไป

เนื่องจาก Oracle Flashback Data Archive นั้นมีรูปแบบการเก็บข้อมูลแบบการเข้าก่อนออกก่อน (First In First Out: FIFO) ดังนั้นหากสร้างพื้นที่ไว้สำหรับ Oracle Flashback Data Archive ไม่เพียงพอก็อาจจะทำให้ข้อมูลเก่าสูญหายได้ ดังนั้นจึงเกิดแนวคิดที่ว่าข้อมูลเก่าที่ Oracle Flashback Data Archive เก็บไว้ให้เป็นหลักวินาทีนั้น จะเก็บไว้ถึงเมื่อไร และเมื่อไรควรจะสรุปข้อมูลเหล่านั้นเก็บแยกไว้ ก่อนที่ Oracle Flashback Data Archive จะลบข้อมูลส่วนนั้นเมื่อ Oracle Flashback Data Archive เต็ม

ในแง่ของการออกแบบคลังข้อมูล นั้นไม่สามารถกำหนดได้ตายตัวว่า เมื่อไหร่ควรจะสรุปข้อมูลที่ Oracle Flashback Data Archive เก็บไว้ในหลักวินาที มาเป็นระดับของเวลาในระดับที่ใหญ่กว่า, ควรจะสรุปกี่ช่วง และช่วงเวลาห่างกันเท่าใดของการสรุปแต่ละครั้ง ดังนั้นเพื่อความยืดหยุ่นในการสร้างโปรแกรมประยุกต์เพื่อที่จะสืบค้นข้อมูลจาก Oracle Flashback Data Archive มาสร้างเป็นคลังข้อมูลนั้น จึงมีอัลกอริทึมในการสรุปข้อมูลตามช่วงเวลาต่างๆ ดังนี้

3.3.1 การสรุปข้อมูลของแหล่งข้อมูลที่มีการจัดเก็บข้อมูลเพียงอย่างเดียว ไม่มีการปรับปรุงข้อมูล

ขั้นแรกผู้ออกแบบระบบคลังข้อมูลจะต้องกำหนดก่อนว่าจะให้มีการรวมข้อมูลทุกๆกี่วินาที/นาที/ชั่วโมง/วัน/เดือนหรือปี จากนั้นโปรแกรมประยุกต์จะต้องสร้างพื้นที่จัดเก็บสำหรับข้อมูลตรงนี้ ดังนี้



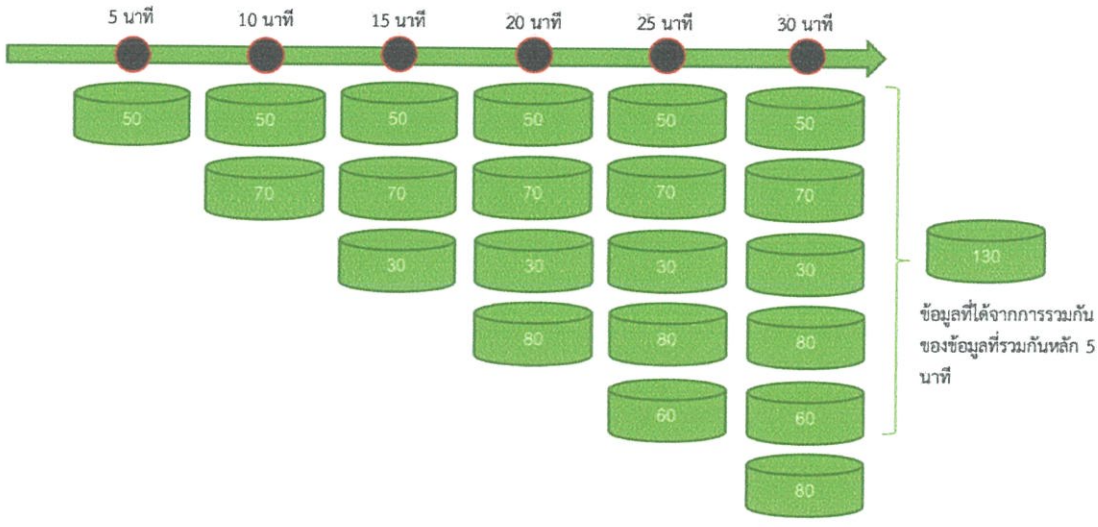
รูป 3.7 การสรุปข้อมูลตามช่วงเวลาตามที่กำหนดไปเก็บไว้ยังส่วนหนึ่งของฐานข้อมูลที่ไม่ใช่เนื้อที่ส่วนเดียวกับ Oracle Flashback Data Archive

จากรูปผู้ออกแบบ คลังข้อมูล กำหนดให้มีการรวมข้อมูลทุกๆ 5 นาที และ 30 นาทีตามลำดับ ดังนั้นเมื่อมีการจัดเก็บข้อมูลมาเรื่อยๆ จนเวลาถึง 5 นาที แล้วก็จะทำการรวมแถวที่เหมือนกัน โดยจะมีหลักในการพิจารณาการรวมดังนี้

1) หากแถวนั้นสามารถรวมกันได้โดยเป็นข้อมูลประเภทตัวเลข (Numeric) ผู้ออกแบบจะต้องออกแบบว่าจะทำการรวมกันอย่างไร โดยใช้หลักการทางคณิตศาสตร์สถิติ เช่น รวมกันโดยหาค่าเฉลี่ยของแถวทั้งหมด หรือรวมกันโดยหามัธยฐาน เป็นต้น

2) หากข้อมูลนั้นไม่สามารถรวมกันได้ กล่าวคือทั้งแถวมีเพียงข้อมูลประเภทไม่ใช่ตัวเลข (Non-numeric) จะคงข้อมูลนั้นไว้

ดังนั้นจากรูปจะเห็นว่า เมื่อเวลาผ่านไปครบ 5 นาที จากที่เคยจัดเก็บข้อมูลไปแล้ว 100 แถว ถูกรวมและคัดกรองแถวที่ซ้ำไปเหลือเพียง 50 แถว แล้วเพิ่มข้อมูลที่สรุปรวมกันแล้วไปยังพื้นที่ที่เตรียมไว้ จากนั้นเมื่อเวลาผ่านไปจนกระทั่งครบ 30 นาทีจะนำเอาข้อมูลที่รวมไว้ทุกๆ 5 นาทีมาทำการสรุปรวมกันอีกครั้งเพื่อสร้างเป็นชุดข้อมูลที่เกิดจากการรวมกันทุกๆ 30 นาที ดังนี้



รูป 3.8 การสรุปข้อมูลรวมกันทุก 5 และ 30 นาที ตามลำดับ

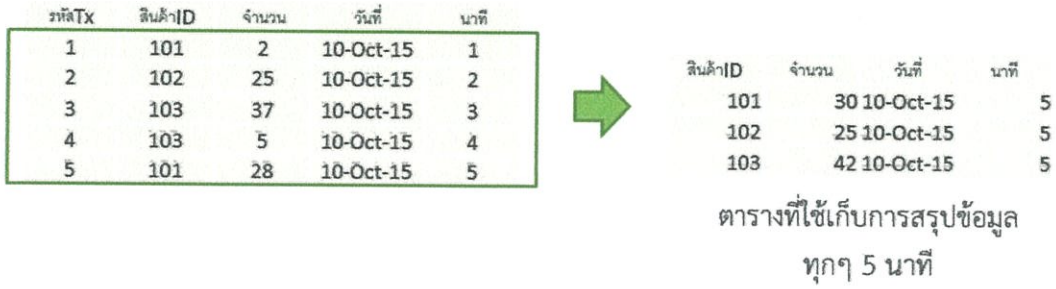
จากรูปเมื่อเวลาผ่านไป 5 นาทีจะเกิดการรวมข้อมูล 1 ครั้ง ซึ่งเป็นข้อมูลนาทีที่ 1 ถึงนาทีที่ 5 ต่อมาเมื่อเวลาผ่านไปจนถึงนาทีที่ 10 จะเกิดการรวมข้อมูลกันอีกครั้งหนึ่ง ซึ่งเป็นข้อมูลนาทีที่ 6 ถึง นาทีที่ 10 จึงเกิดเป็นชุดข้อมูลของการรวมกันทุกๆ 5 นาทีเก็บไว้สองชุด โดยจะเก็บไว้ที่เดียวกัน เป็นเช่นนี้เรื่อยๆ จนถึงนาทีที่ 30 จะเกิดชุดข้อมูลของการรวมกันทุกๆ 5 นาทีทั้งหมด 6 ชุด ดังนั้น เมื่อเวลาผ่านไป 30 นาที เพียงแค่หยิบนำเอาชุดข้อมูลที่เคยสรุปไว้ทุกๆ 5 นาทีมาสรุปรวมกันเป็น 1 ชุด ไม่จำเป็นต้องเริ่มสรุปจากข้อมูลนาทีที่ 1 ใหม่ จากนั้นเก็บแยกกันกับชุดข้อมูลที่รวมกันทุกๆ 5 นาที เพื่อป้องกันการซ้ำซ้อนข้องข้อมูลดังตัวอย่างต่อไปนี้

- เมื่อเวลาผ่านไป 5 นาที

  วิธีการหยิบข้อมูลสามารถทำได้ด้วย Oracle Flashback Query ดังนี้

```
Select * from table as of timestamp
to_timestamp (ณ เวลาที่ input ข้อมูล นาทีที่ 5)
```

ผลลัพธ์คือจะได้ตารางที่มีข้อมูลของการเพิ่มข้อมูล เมื่อเวลาผ่านไป 5 นาทีดังรูป



รูป 3.9 การรวมข้อมูลเมื่อเวลาครบ 5 นาที

จากรูปแสดงให้เห็นว่าเมื่อเวลาผ่านไปครบ 5 นาทีจะทำการหยิบข้อมูลขึ้นมาแล้วทำการคัดกรองข้อมูลที่มีความเหมือนกันออก จากนั้นจะนำข้อมูลไปลงตารางใหม่ซึ่งเป็นตารางที่มีโครงสร้างตาราง (Schema) เดียวกันกับตารางต้นฉบับ โดยได้เหลือแถวที่ไม่เหมือนกันเอาไว้เหลือเพียง 3 แถว

- เมื่อเวลาผ่านไป 10 นาที

จะทำการหยิบข้อมูล ณ นาทีที่ 10 ขึ้นมาจาก Oracle Flashback Data Archive ด้วยคำสั่งดังต่อไปนี้

```
Select * from table as of timestamp
to_timestamp (ณ เวลาที่ input ข้อมูล นาทีที่ 10)
```

จากนั้นทำการหยิบข้อมูลย้อนกลับ 5 นาที เนื่องจากเป็นเวลาที่ผู้ออกแบบกำหนดในการให้โปรแกรมประยุกต์ทำการสรุปข้อมูล เพื่อนำมาคัดกรองแถวที่ได้ทำการสรุปไปแล้วก่อนหน้านี้ ซึ่งเป็นข้อมูล ณ เวลา 5 นาทีด้วยคำสั่งดังนี้

```
Select * from table as of timestamp
to_timestamp (ณ เวลาที่ input ข้อมูล นาทีที่ 5)
```

จากนั้นนำข้อมูลทั้งสองส่วนที่ได้จากการสอบถามหักล้างกัน เราจะได้ข้อมูลของนาทีที่ 10 ที่ไม่มีข้อมูลที่สรุปไปแล้วตอนนาทีที่ 5 ปะปนอยู่ ดังแสดงในรูปต่อไปนี้

รหัสTx	สินค้าID	จำนวน	วันที่	นาที
1	101	2	10-Oct-15	1
2	102	25	10-Oct-15	2
3	103	37	10-Oct-15	3
4	103	5	10-Oct-15	4
5	101	28	10-Oct-15	5
6	105	6	10-Oct-15	6
7	101	15	10-Oct-15	7
8	101	37	10-Oct-15	8
9	102	28	10-Oct-15	9
10	104	23	10-Oct-15	10

รหัสTx	สินค้าID	จำนวน	วันที่	นาที
1	101	2	10-Oct-15	1
2	102	25	10-Oct-15	2
3	103	37	10-Oct-15	3
4	103	5	10-Oct-15	4
5	101	28	10-Oct-15	5

รูป 3.10 ข้อมูลของตารางเมื่อเวลาผ่านไป 10 นาที หักล้างกับข้อมูลของตารางเมื่อเวลาผ่านไป 5 นาที

เมื่อหักล้างแถวที่เคยสรุปไปออกเรียบร้อยแล้ว ก็จะทำการสรุปข้อมูลตามปกติ

รหัสTx	สินค้าID	จำนวน	วันที่	นาที
6	105	6	10-Oct-15	6
7	101	15	10-Oct-15	7
8	101	37	10-Oct-15	8
9	102	28	10-Oct-15	9
10	104	23	10-Oct-15	10

สินค้าID	จำนวน	วันที่	นาที
101	30	10-Oct-15	5
102	25	10-Oct-15	5
103	42	10-Oct-15	5
101	52	10-Oct-15	10
102	28	10-Oct-15	10
104	23	10-Oct-15	10
105	6	10-Oct-15	10

สรุปข้อมูลแล้วนำไปเก็บไว้ในตารางที่เก็บข้อมูลที่สรุปทุก 5 นาที

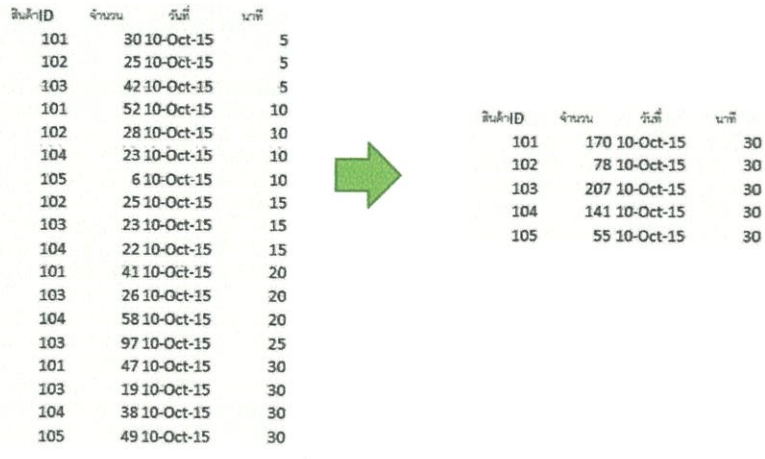
รูป 3.11 การสรุปข้อมูล ณ เวลา 10 นาที

เมื่อผ่านไป 30 นาทีจะได้ตารางที่มีข้อมูลดังนี้

สินค้าID	จำนวน	วันที่	นาที
101	30	10-Oct-15	5
102	25	10-Oct-15	5
103	42	10-Oct-15	5
101	52	10-Oct-15	10
102	28	10-Oct-15	10
104	23	10-Oct-15	10
105	6	10-Oct-15	10
102	25	10-Oct-15	15
103	23	10-Oct-15	15
104	22	10-Oct-15	15
101	41	10-Oct-15	20
103	26	10-Oct-15	20
104	58	10-Oct-15	20
103	97	10-Oct-15	25
101	47	10-Oct-15	30
103	19	10-Oct-15	30
104	38	10-Oct-15	30
105	49	10-Oct-15	30

รูป 3.12 การสรุปข้อมูล ณ เวลา 30 นาที

เมื่อเวลาครบ 30 นาทีแล้วจะตรงกับเงื่อนไขที่สองที่ต้องสรุปข้อมูลทุกๆ 30 นาที ดังนั้นจึงทำการสรุปข้อมูลที่มีอยู่ในตารางที่เก็บการสรุปทุก 5 นาทีให้กลายเป็น 1 ชุดข้อมูลในตาราง 30 นาทีดังนี้



สินค้าID	จำนวน	วันที่	นาที
101	30	10-Oct-15	5
102	25	10-Oct-15	5
103	42	10-Oct-15	5
101	52	10-Oct-15	10
102	28	10-Oct-15	10
104	23	10-Oct-15	10
105	6	10-Oct-15	10
102	25	10-Oct-15	15
103	23	10-Oct-15	15
104	22	10-Oct-15	15
101	41	10-Oct-15	20
103	26	10-Oct-15	20
104	58	10-Oct-15	20
103	97	10-Oct-15	25
101	47	10-Oct-15	30
103	19	10-Oct-15	30
104	38	10-Oct-15	30
105	49	10-Oct-15	30

สินค้าID	จำนวน	วันที่	นาที
101	170	10-Oct-15	30
102	78	10-Oct-15	30
103	207	10-Oct-15	30
104	141	10-Oct-15	30
105	55	10-Oct-15	30

รูป 3.13 การสรุปข้อมูลจากตารางที่เก็บการสรุปข้อมูลทุก 5 นาทีไปเป็น 1 ชุดข้อมูลในตารางที่สรุปข้อมูลทุกๆ 30 นาที

เมื่อทำการสรุปข้อมูลจากตารางที่เก็บข้อมูลทุก 5 นาทีเรียบร้อยแล้ว ควรทำการลบข้อมูลดังกล่าวออกเพื่อให้ประหยัดพื้นที่จัดเก็บและลดความซ้ำซ้อน โดยมีเงื่อนไขที่ว่า ถ้าข้อมูลผ่านการสรุปไปแล้วจะไม่สามารถเรียกย้อนดูข้อมูลก่อนมีการสรุปได้ ซึ่งหากเราใช้ฐานข้อมูลปกติจะเกิดความยุ่งยากตรงนี้คือจะต้องสอบถามข้อมูลขึ้นมาทั้งหมดแล้วมากรองแถวที่ไม่อยู่ในช่วงเวลาที่ต้องการออก หรือไม่ก็ต้องทำการลบ เมื่อทำการสรุปข้อมูลทิ้งไปแล้ว จึงส่งผลให้การพัฒนาโปรแกรมประยุกต์มีความซับซ้อนขึ้นไปอีก

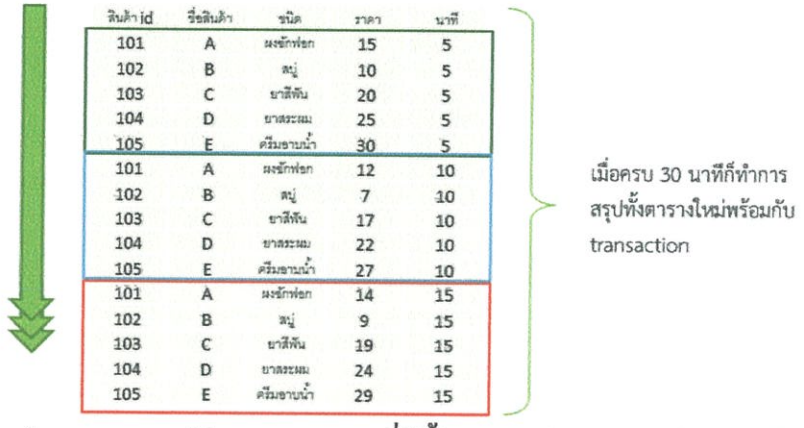
3.3.2 การสรุปข้อมูลของแหล่งข้อมูลที่มีทั้งการจัดเก็บข้อมูลและมีการปรับปรุงข้อมูล

หลังจากการที่เราสรุปข้อมูลที่มีการจัดเก็บข้อมูลเพียงอย่างเดียวไปแล้ว เราต้องทำการสรุปข้อมูลจากแหล่งข้อมูลที่มีทั้งการจัดเก็บและการปรับปรุงข้อมูลด้วย เพื่อที่เวลาสร้าง Fact table จะได้นำข้อมูลที่อยู่ในช่วงเวลาเดียวกันมาประกอบกัน ซึ่งการสรุปข้อมูลที่มีทั้งการจัดเก็บข้อมูลและมีการปรับปรุงข้อมูล มีวิธีการดังนี้

สินค้าid	ชื่อสินค้า	ชนิด	ราคา	นาที
101	A	ผงซักฟอก	15	1
102	B	สบู่	10	1
103	C	ยาสีฟัน	20	1
104	D	ยาสระผม	25	1
105	E	ครีมอาบน้ำ	30	1
101	A	ผงซักฟอก	10	8
102	B	สบู่	5	8
103	C	ยาสีฟัน	15	8
104	D	ยาสระผม	20	8
105	E	ครีมอาบน้ำ	25	8
101	A	ผงซักฟอก	15	12
102	B	สบู่	10	12
103	C	ยาสีฟัน	20	12
104	D	ยาสระผม	25	12
105	E	ครีมอาบน้ำ	30	12
101	A	ผงซักฟอก	20	19
102	B	สบู่	15	19
103	C	ยาสีฟัน	25	19
104	D	ยาสระผม	30	19
105	E	ครีมอาบน้ำ	35	19
101	A	ผงซักฟอก	40	30
102	B	สบู่	45	30
103	C	ยาสีฟัน	20	30
104	D	ยาสระผม	20	30
105	E	ครีมอาบน้ำ	20	30

รูป 3.14 ตารางข้อมูลที่มีทั้งการจัดเก็บข้อมูลและการปรับปรุงข้อมูลเมื่อเวลาผ่านไป

จากข้อกำหนดเดิมที่จะต้องทำการสรุปข้อมูลเมื่อเวลาผ่านไปทุก 5 นาทีและ 30 นาที จะเห็นว่าเมื่อเวลาผ่านไปถึงช่วงนาทีที่ 5 ถึงนาทีที่ 10 ข้อมูลมีการปรับปรุง ณ นาทีที่ 8 ดังนั้นขั้นตอนในการสรุปข้อมูลคือ ผู้ออกแบบระบบจะต้องกำหนดความละเอียดของการติดตามการเปลี่ยนแปลงเป็นเท่าใด จากในตัวอย่างกำหนดให้ความละเอียดในการติดตามการเปลี่ยนแปลงคือ 1 นาที ดังนั้นจึงต้องทำการสอบถามข้อมูลมาจาก Oracle Flashback Data Archive ขึ้นมาด้วยคาบเวลาคือ 1 นาที จากนั้นทำการสรุปรวมข้อมูลการสรุปข้อมูลในตารางที่มีการจัดเก็บเพียงอย่างเดียว โดยจากการสอบถามข้างต้นจะได้ข้อมูลทั้งหมด 5 ชุด เห็นได้ว่าการสรุปข้อมูลของ 5 นาทีแรกนั้นไม่มีความเปลี่ยนแปลง เนื่องจากข้อมูลมีการปรับปรุงครั้งแรก ณ นาทีที่ 8 ต่อมาทำการสรุปข้อมูล ณ นาทีที่ 6 ถึงนาทีที่ 10 จะเห็นความเปลี่ยนแปลงเกิดขึ้น ซึ่งเป็นผลจากการปรับปรุง ณ นาทีที่ 8



รูป 3.15 การสรุปข้อมูลของตารางที่มีทั้งการจัดเก็บและการปรับปรุงข้อมูล

ทำเช่นเดิมไปเรื่อยๆ จนครบ 30 นาทีแล้วจึงสรุปข้อมูลทั้งหมดไปเก็บไว้ในที่สำหรับเก็บข้อมูลที่สรุปทุกๆ 30 นาทีเช่นเดียวกับการสรุปข้อมูลที่มีการจัดเก็บเพียงอย่างเดียวจากนั้นจึงทำการลบข้อมูลที่สรุปเรียบร้อยแล้วทิ้งไป

จากที่กล่าวมาข้างต้นเป็นการสรุปข้อมูลที่มีอยู่ใน Oracle Flashback Data Archive เพื่อป้องกันไม่ให้เกิดเหตุการณ์ Oracle Flashback Data Archive เต็ม เนื่องจากว่าหาก Oracle Flashback Data Archive เต็มข้อมูลเก่าอาจจะถูกเขียนทับได้ ดังนั้นจึงต้องทำการสรุปข้อมูลไปเก็บไว้ที่ซึ่งแน่ใจว่าข้อมูลเหล่านั้นจะคงอยู่ถาวร

3.4 ขั้นตอนในการสรุปข้อมูลจาก Oracle Flashback Data Archive มาสร้างเป็นคลังข้อมูล

ในการสรุปข้อมูลจาก Oracle Flashback Data Archive นั้นจะแบ่งแหล่งข้อมูลเป็น 3 ลักษณะคือ

กรณีที่ 1 แหล่งข้อมูลที่มี Fact source ไม่ใช่ข้อมูลเชิงเวลา แต่ Dimension source เป็นข้อมูลเชิงเวลา

กรณีที่ 2 แหล่งข้อมูลที่มี Fact source เป็นข้อมูลเชิงเวลา แต่ Dimension source ไม่ใช้ข้อมูลเชิงเวลา

กรณีที่ 3 แหล่งข้อมูลที่มี Fact source เป็นข้อมูลเชิงเวลาและ Dimension source เป็นข้อมูลเชิงเวลาเช่นกัน

อัลกอริทึมในการสรุปแหล่งข้อมูลแต่ละแบบแตกต่างกันดังนี้

กรณีที่ 1 แหล่งข้อมูลที่มี Fact source ไม่ใช่ข้อมูลเชิงเวลา แต่ Dimension source เป็นข้อมูลเชิงเวลา

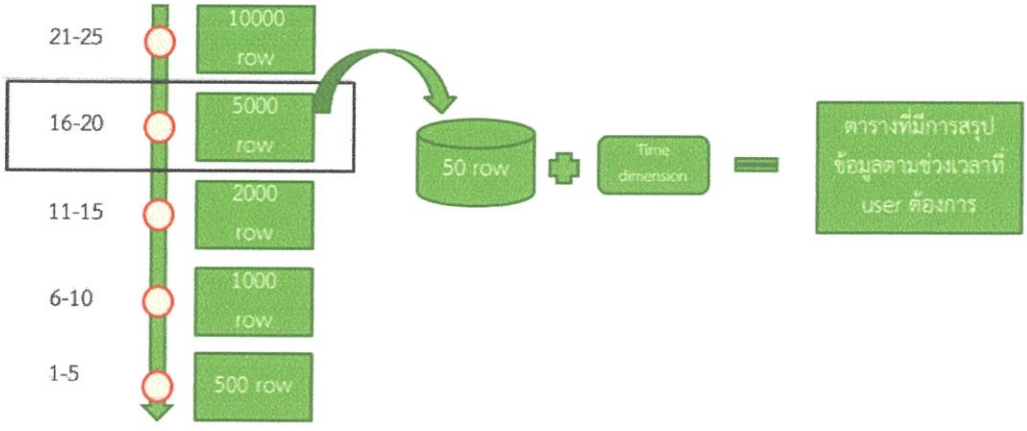
แหล่งข้อมูลลักษณะนี้ส่วนมากจะพบได้ในคลังข้อมูลทั่วไปที่มี Fact source เป็นรายการบันทึกที่มีเพียงการจัดเก็บข้อมูลลงตาราง โดยไม่มีการปรับปรุง ซึ่งแหล่งข้อมูลลักษณะนี้จะไม่มีการเปลี่ยนแปลงตามเวลา แต่ Dimension source เปลี่ยนแปลงตามเวลา ดังนั้นในคลังข้อมูลรูปแบบปกติจึงใช้ Slowly changing dimension เข้ามาจัดการกับปัญหาเหล่านี้ แต่หากเป็นคลังข้อมูลที่สร้างขึ้นจาก Oracle Flashback Data Archive สามารถสร้างคลังข้อมูลที่มีแหล่งข้อมูลลักษณะที่ 1 ได้ดังนี้

ในกรณีการสรุป Fact source ก่อนที่จะทำการสร้างคลังข้อมูลนั้นจะต้องตรวจสอบก่อนว่า ผู้สร้างต้องการจะทำคลังข้อมูลตั้งแต่เมื่อใด แล้วแหล่งข้อมูลของเวลาที่ต้องการดังกล่าวได้มีการสรุปไว้แล้วหรือยัง และสุดท้ายต้องพิจารณาว่าระดับเวลาที่ต้องการนั้น ระดับใหญ่กว่าหรือเท่ากับแหล่งข้อมูลเดิมที่เคยสรุปไว้แล้ว สามารถจำแนกได้ดังนี้

1) หากช่วงเวลาที่ผู้ออกแบบคลังข้อมูลต้องการอยู่ในช่วงเวลาที่ทำการสรุปข้อมูลไปแล้ว เราเพียงพิจารณาว่าระดับเวลาที่ต้องการอยู่ในระดับที่เท่ากันหรือใหญ่กว่าระดับเวลาที่เคยสรุปไปแล้วหากระดับเวลาที่ผู้ออกแบบคลังข้อมูลต้องการใหญ่กว่าที่เคยสรุปไว้แล้วก็เพียงแต่หยิบข้อมูลที่เคยสรุปไว้แล้วมาสรุปอีกครั้งหนึ่งตามระดับเวลาที่ผู้ออกแบบคลังข้อมูลต้องการโดยใช้อัลกอริทึมที่ได้กล่าวไว้แล้วข้างต้น แต่หากระดับเวลาที่ผู้ออกแบบคลังข้อมูลต้องการเท่ากับข้อมูลที่เคยสรุปไว้แล้วก็สามารถนำเอาข้อมูลนั้นมาตอบตามระดับเวลาที่ผู้ออกแบบคลังข้อมูลต้องการได้เลย ตัวอย่างเช่น มีตารางที่สรุปข้อมูลไว้แล้วที่เวลา 5 นาที, 30 นาที, 1 ชม, 1 วัน, 1 เดือน เป็นต้น หากผู้สร้างคลังข้อมูลต้องการดูข้อมูลการเปลี่ยนแปลงใน 1 วันดังนั้นตารางที่จะตอบคำถามนี้ได้ดีที่สุดคือตารางที่เก็บการสรุปข้อมูลทุกๆ 1 ชมและตารางนี้สามารถตอบคำถามนี้ได้เลย หากผู้ใช้ต้องการดูข้อมูลการเปลี่ยนแปลงทุกๆ 10 วัน จะต้องนำข้อมูลของการสรุปทุก 1 วันมา สรุปรวมกัน 10 วันโดยใช้มุมมองของเวลาเพื่อตอบคำถามแก่ผู้สร้างคลังข้อมูล

2) หากช่วงเวลาที่ผู้ออกแบบคลังข้อมูลต้องการนั้นไม่ได้มีการสรุปข้อมูลไว้ก่อนหน้ากล่าวคือข้อมูลเหล่านั้นยังคงอยู่ใน Oracle Flashback Data Archive ยังไม่ถึงเวลาที่ผู้ออกแบบคลังข้อมูลตั้งค่าเอาไว้ให้สรุปรวมข้อมูล เราจะต้องใช้ Oracle Flashback Query ดึงข้อมูลมาจาก Oracle Flashback Data Archive โดยที่ผู้ออกแบบคลังข้อมูลจะต้องระบุว่าต้องการความละเอียดเท่าใด (อัลกอริทึมเช่นเดียวกับการสรุปข้อมูลข้างต้น) จากนั้นแสดงข้อมูลที่สรุปได้ตามช่วงระดับเวลาที่กำหนด

ตัวอย่าง หากผู้สร้างคลังข้อมูลต้องการระบุ scope ในการทำคลังข้อมูล เช่น เก็บข้อมูลมาถึงวันที่ 25 แต่ผู้ใช้ไม่ได้ต้องการข้อมูลวันที่ 21-25 ดังนั้นสามารถสอบถามเวอร์ชัน ณ วันที่ 20 มาทำคลังข้อมูลได้เลยโดยไม่ต้องใช้คำสั่ง Select แล้วใช้ where กรองแถวของวันที่ 21-25 ออกดังรูป



รูป 3.16 การสรุป Fact source ไปเป็น Fact table ในคลังข้อมูล

ซึ่งการสรุป Fact source จาก Oracle Flashback Data Archive ไปเป็นคลังข้อมูลนั้นจะต้องทำควบคู่กับการสรุป Dimension source ด้วยเพื่อให้คลังข้อมูลมีข้อมูลที่ถูกต้องสอดคล้องกันทั้งหมด ซึ่งมีเงื่อนไขในการสรุปดังต่อไปนี้

1) หากเป็นข้อมูลของแอตทริบิวต์ที่ไม่สามารถรวมกันได้ ตัวอย่างเช่น ชื่อ, ที่อยู่ ให้คงข้อมูลนั้นไว้สำหรับการทำ Time travel

2) หากเป็นข้อมูลของแอตทริบิวต์ที่สามารถรวมกันได้ เช่น ข้อมูลที่เป็นตัวเลข ผู้ใช้ต้องกำหนดความต้องการว่าต้องการให้สรุปข้อมูลของแอตทริบิวต์นั้นอย่างไรในช่วงเวลาดังกล่าว

   - ค่าเฉลี่ย (Average), ค่าน้อยที่สุด (min), ค่ามากที่สุด (max) ตามระดับของเวลาที่ใช้กับตาราง Transaction

   - หาค่าเฉลี่ยการเปลี่ยนแปลงทั้งหมดในตาราง Valid time state เป็นค่าใหม่เป็นค่าเดียว

กรณีที่ 2 แหล่งข้อมูลที่มี Fact source เป็นข้อมูลเชิงเวลา แต่ Dimension source ไม่ใช้ข้อมูลเชิงเวลา

แหล่งข้อมูลที่มีลักษณะเช่นนี้ จะแบ่งแหล่งข้อมูลได้เป็นสองประเภทคือ แหล่งข้อมูลที่มีลักษณะเป็นตาราง Valid time state ที่จะเก็บข้อมูลที่เป็นจริง ณ เวลาหนึ่งๆ และแหล่งข้อมูลที่มีลักษณะเป็นตาราง Transaction time state ที่จะเก็บเวลาที่ Fact ถูกเพิ่มลงในฐานข้อมูลในกรณีที่แหล่งข้อมูลเป็นตาราง Valid time state จะมีขั้นตอนในการสรุปดังนี้

FDYD_ID	LOT_IDNUM	PED_ID	HD_CNT	FROM_DATE	TO_DATE
1	137	1	17	7-Feb-98	18-Feb-98
1	219	1	43	25-Feb-98	1-Mar-98
1	219	1	20	1-Mar-98	30-Mar-98
1	219	2	23	1-Mar-98	14-Mar-98
1	137	1	10	18-Feb-98	18-Apr-98
1	137	2	7	18-Feb-98	18-Apr-98
1	219	2	15	14-Mar-98	30-Mar-98
1	219	3	8	14-Mar-98	14-Jun-98
1	374	1	14	2-Feb-98	20-Feb-98
1	374	1	5	20-Feb-98	20-Apr-98
1	374	2	5	21-Feb-98	21-Apr-98
1	374	3	4	22-Feb-98	22-Apr-98
1	219	1	35	30-Mar-98	14-Jun-98
1	137	1	17	18-Apr-98	NOW
1	219	1	20	14-Jun-98	NOW
1	219	2	23	14-Jun-98	NOW
1	374	1	14	20-Apr-98	NOW

รูป 3.17 ข้อมูลที่มีในตาราง Valid time state

ตัวอย่างข้อมูลในตาราง Valid time state มีลักษณะดังข้างต้น ซึ่งขั้นตอนในการสรุปข้อมูลคือ ในขั้นแรกผู้ออกแบบคลังข้อมูลจะต้องระบุก่อนว่าต้องการความละเอียดของแหล่งข้อมูลเป็นเท่าใดเช่นเดียวกับการสรุปข้อมูลในกรณีที่ 1 จากตัวอย่างระบุความละเอียดของแหล่งข้อมูลเป็นระดับวันจะต้องทำการสอบถามข้อมูลจาก Oracle Flashback Data Archive ได้ข้อมูลดังนี้

Seq#	FDYD_ID	OT_IDNUM	PED_ID	HD_CNT	Date
1	1	137	1	17	2-Feb-98
2	1	219	1	43	2-Feb-98
3	1	374	1	14	2-Feb-98
4	1	137	1	17	3-Feb-98
5	1	219	1	43	3-Feb-98
6	1	374	1	14	3-Feb-98
7	1	137	1	17	4-Feb-98
8	1	219	1	43	4-Feb-98
9	1	374	1	14	4-Feb-98
10	1	137	1	10	5-Feb-98
11	1	137	2	7	5-Feb-98
12	1	219	1	43	5-Feb-98
13	1	374	1	14	5-Feb-98
14	1	137	1	10	6-Feb-98
15	1	137	2	7	6-Feb-98
16	1	219	1	20	6-Feb-98
17	1	219	2	23	6-Feb-98
18	1	374	1	14	6-Feb-98
19	1	137	1	10	7-Feb-98
20	1	137	2	7	7-Feb-98
21	1	219	1	20	7-Feb-98
22	1	219	2	23	7-Feb-98
23	1	374	1	5	7-Feb-98
24	1	374	2	5	7-Feb-98
25	1	374	3	4	7-Feb-98
26	1	137	1	10	8-Feb-98
27	1	137	2	7	8-Feb-98
28	1	219	1	20	8-Feb-98
29	1	219	2	23	8-Feb-98
30	1	374	1	5	8-Feb-98
31	1	374	2	5	8-Feb-98
32	1	374	3	4	8-Feb-98
33	1	137	1	10	9-Feb-98
34	1	137	2	7	9-Feb-98
35	1	219	2	15	9-Feb-98
36	1	219	3	8	9-Feb-98
37	1	219	1	20	9-Feb-98
38	1	374	1	5	9-Feb-98
39	1	374	2	5	9-Feb-98
40	1	374	3	4	9-Feb-98

รูป 3.18 ข้อมูลการสอบถามจาก Oracle Flashback Data Archive

วันที่ 2 Feb. 1998 ถึงวันที่ 9 Feb. 1998

41	1	137	1	17	10-Feb-98
42	1	219	2	15	10-Feb-98
43	1	219	3	8	10-Feb-98
44	1	219	1	20	10-Feb-98
45	1	374	1	5	10-Feb-98
46	1	374	2	5	10-Feb-98
47	1	374	3	4	10-Feb-98
48	1	137	1	17	11-Feb-98
49	1	219	3	8	11-Feb-98
50	1	219	1	35	11-Feb-98
51	1	374	1	5	11-Feb-98
52	1	374	2	5	11-Feb-98
53	1	374	3	4	11-Feb-98
54	1	137	1	17	12-Feb-98
55	1	219	3	8	12-Feb-98
56	1	219	1	35	12-Feb-98
57	1	374	1	14	12-Feb-98
58	1	137	1	17	13-Feb-98
59	1	219	3	8	13-Feb-98
60	1	219	1	35	13-Feb-98
61	1	374	1	14	13-Feb-98
62	1	137	1	17	14-Feb-98
63	1	219	3	8	14-Feb-98
64	1	219	1	35	14-Feb-98
65	1	374	1	14	14-Feb-98
66	1	137	1	17	15-Feb-98
67	1	219	1	20	15-Feb-98
68	1	219	2	23	15-Feb-98
69	1	374	1	14	15-Feb-98

รูป 3.19 ข้อมูลการสอบถามจาก Oracle Flashback Data Archive
วันที่ 10 Feb. 1998 ถึงวันที่ 15 Feb. 1998

และเมื่อสอบถามขึ้นมาทุกๆ เวอร์ชัน ด้วย Oracle Flashback Query สิ่งที่ได้คือผลลัพธ์ที่เป็นรีเลชัน ดังนั้นแถวที่ซ้ำจะไม่ปรากฏเป็นผลลัพธ์ ดังแสดงต่อไปนี้

Seq#	FDYD_ID	LOT_IDNUM	PED_ID	HD_CNT	Date
1	1	137	1	17	4-Feb-98
2	1	219	1	43	4-Feb-98
3	1	374	1	14	4-Feb-98
4	1	137	1	10	5-Feb-98
5	1	137	2	7	5-Feb-98
6	1	219	1	20	6-Feb-98
7	1	219	2	23	6-Feb-98
8	1	374	1	5	7-Feb-98
9	1	374	2	5	7-Feb-98
10	1	374	3	4	7-Feb-98
11	1	219	2	15	9-Feb-98
12	1	219	3	8	9-Feb-98
13	1	137	1	17	10-Feb-98
14	1	219	1	35	11-Feb-98
15	1	374	1	14	12-Feb-98
16	1	219	1	20	15-Feb-98
17	1	219	2	23	15-Feb-98

รูป 3.20 การใช้ Oracle Flashback Query ดึงข้อมูลจากช่วงเวลาต่างๆมาไว้ ณ ตารางเดียวกัน

จากนั้นจะแสดงตัวอย่างการตอบคำถามตามระดับเวลาต่างๆ โดยจะแสดงการตอบคำถามช่วงเวลา 3 วันและ 1 สัปดาห์ตามลำดับ

Seq#	FDYD_ID	LOT_IDNUM	PED_ID	HD_CNT	Date
1	1	137	1	17	4-Feb-98
2	1	137	1	10	5-Feb-98
3	1	137	2	7	5-Feb-98
4	1	137	1	17	10-Feb-98
5	1	219	1	43	4-Feb-98
6	1	219	1	20	6-Feb-98
7	1	219	2	23	6-Feb-98
8	1	219	2	15	9-Feb-98
9	1	219	3	8	9-Feb-98
10	1	219	1	35	11-Feb-98
11	1	219	1	20	15-Feb-98
12	1	219	2	23	15-Feb-98
13	1	374	1	14	4-Feb-98
14	1	374	1	5	7-Feb-98
15	1	374	2	5	7-Feb-98
16	1	374	3	4	7-Feb-98
17	1	374	1	14	12-Feb-98

3 day

Seq#	FDYD_ID	OT_IDNUM	PED_ID	HD_CNT	Date
1	1	137	1	17	4-Feb-98
4	1	137	1	17	10-Feb-98
5	1	219	1	43	4-Feb-98
8	1	219	2	15	9-Feb-98
9	1	219	3	8	9-Feb-98
10	1	219	1	35	11-Feb-98
11	1	219	1	20	15-Feb-98
12	1	219	2	23	15-Feb-98
13	1	374	1	14	4-Feb-98
14	1	374	1	5	7-Feb-98
15	1	374	2	5	7-Feb-98
16	1	374	3	4	7-Feb-98
17	1	374	1	14	12-Feb-98

รูป 3.21 การตอบคำถามด้วยระดับเวลา 3 วัน

จากรูปจะเห็นได้ว่าข้อมูลที่เหมือนกันในช่วงเวลา 3 วันจะถูกรวมเป็นแถวเดียวกัน แต่หากข้อมูลอยู่ห่างกันเกิน 3 วันข้อมูลเหล่านั้นก็จะไม่ถูกรวมกันแม้จะมีข้อมูลเหมือนกันก็ตาม

Seq#	FDYD_ID	LOT_IDNUM	PED_ID	HD_CNT	Date
1	1	137	1	17	4-Feb-98
2	1	137	1	10	5-Feb-98
3	1	137	2	7	5-Feb-98
4	1	137	1	17	10-Feb-98
5	1	219	1	43	4-Feb-98
6	1	219	1	20	6-Feb-98
7	1	219	2	23	6-Feb-98
8	1	219	2	15	9-Feb-98
9	1	219	3	8	9-Feb-98
10	1	219	1	35	11-Feb-98
11	1	219	1	20	15-Feb-98
12	1	219	2	23	15-Feb-98
13	1	374	1	14	4-Feb-98
14	1	374	1	5	7-Feb-98
15	1	374	2	5	7-Feb-98
16	1	374	3	4	7-Feb-98
17	1	374	1	14	12-Feb-98

Week

Seq#	FDYD_ID	OT_IDNUM	PED_ID	HD_CNT	Date
4	1	137	1	17	10-Feb-98
5	1	219	1	43	4-Feb-98
12	1	219	2	23	15-Feb-98
13	1	374	1	14	4-Feb-98
17	1	374	1	14	12-Feb-98

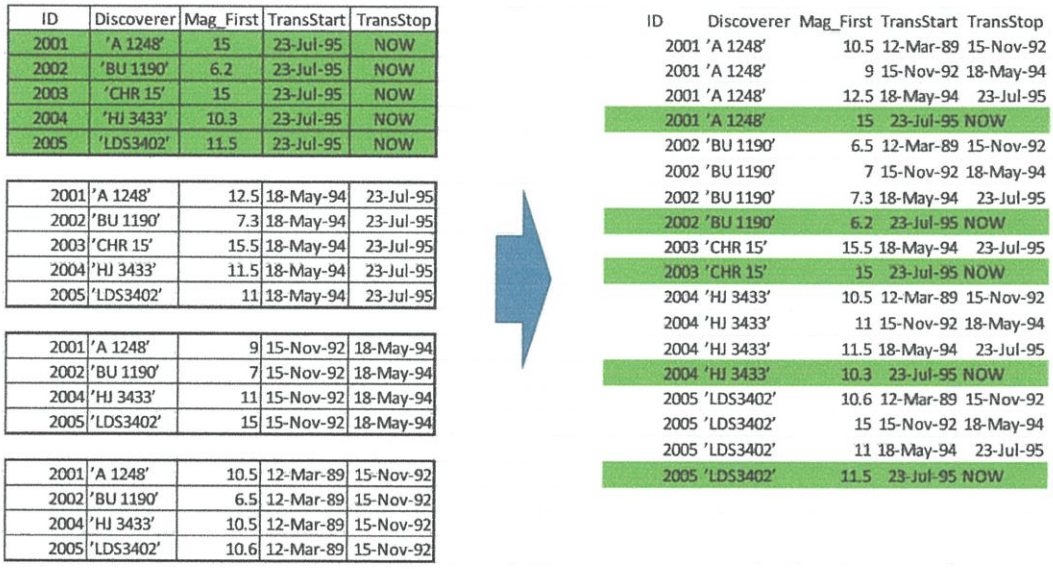
รูป 3.22 การตอบคำถามด้วยระดับเวลา 1 สัปดาห์

จากรูปเป็นการขยายระดับเวลาจากเดิม 3 วันเป็น 1 สัปดาห์ จะเห็นได้ว่าข้อมูลมีแถวลดลงเนื่องจากระดับเวลาที่หยาบขึ้นทำให้เกิดการรวมกันของแต่ละแถวมากขึ้น

ในกรณีที่แหล่งข้อมูลเป็นตาราง Transaction time state จะมีการสรุปข้อมูลเช่นเดียวกันกับการสรุปแหล่งข้อมูลที่เป็นตาราง Valid time state โดยจะแสดงการสรุปข้อมูลด้วยคำสั่ง Oracle Flashback Query ดังนี้

```
select * from star_table as of timestamp to_timestamp('23-Jul-95') union
select * from star_table as of timestamp to_timestamp('18-May-94') union
select * from star_table as of timestamp to_timestamp('12-Nov-92') union
select * from star_table as of timestamp to_timestamp('12-Mar-89')
```

ซึ่งจะได้ผลลัพธ์ดังรูป



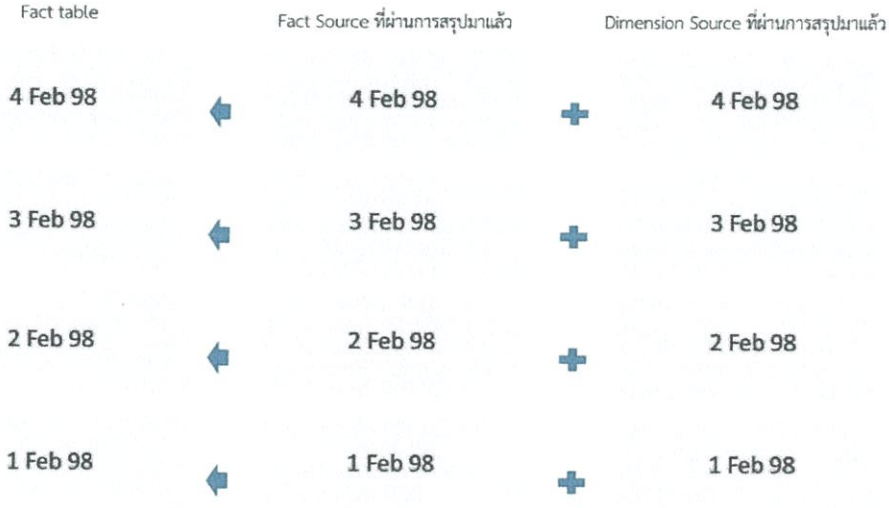
รูป 3.23 ผลลัพธ์จากการใช้ Oracle Flashback Query

สรุปข้อมูลจากแหล่งข้อมูลที่เป็นตาราง Transaction time state

จากรูปจะเห็นได้ว่า เราจะได้ข้อมูลการจัดเก็บของตารางตั้งแต่เวอร์ชันแรกจนถึงเวอร์ชันปัจจุบันและคงข้อมูลการปรับปรุงไว้อย่างครบถ้วน

กรณีที่ 3 แหล่งข้อมูลที่มี Fact source เป็นข้อมูลเชิงเวลาและ Dimension source เป็นข้อมูลเชิงเวลาเช่นกัน

จะมีขั้นตอนในการสรุปข้อมูลคล้ายคลึงกับรูปแบบในการสรุปแหล่งข้อมูลสองแบบข้างต้น แต่แตกต่างกันตรงที่จะต้องทำการสรุป Fact source ควบคู่กับ Dimension source ไปพร้อมกันเนื่องจากข้อมูลในทั้งสองตารางเป็นข้อมูลเชิงเวลา ทั้งคู่โดยหากจะสร้างคลังข้อมูลจะต้องสร้าง Fact table อ้างอิงกับเวลาของข้อมูลนั้นๆ เพื่อให้ได้ข้อมูลที่ถูกต้องสมบูรณ์



รูป 3.24 การสรุปสร้างคลังข้อมูลที่ข้อมูลทั้ง Fact source และ Dimension source เป็นข้อมูลเชิงเวลา

บทที่ 4

การทดลองและผลการทดลอง

4.1 การทดลองการสรุปข้อมูลในคลังข้อมูลที่ได้รวบรวมข้อมูลมาจากแหล่งข้อมูลประเภทต่างๆ

สิ่งที่จะเพิ่มขีดความสามารถให้กับคลังข้อมูล คือการพัฒนาตัวโปรแกรมประยุกต์ที่นำแหล่งข้อมูลในรูปแบบต่างๆที่เป็นไปได้ ซึ่งต่างก็อยู่บนฐานข้อมูลระบบงานประจำวันเดียวกัน มาสรุปเป็นคลังข้อมูลที่ประกอบไปด้วย Fact Table และ Dimension Table ตามหลักเกณฑ์ในการสร้างคลังข้อมูลทั่วไปพร้อมทั้งสามารถตอบคำถามเชิงการตัดสินใจทางธุรกิจได้และนำเสนอคำตอบนั้นในรูปแบบของกราฟ โดยแหล่งข้อมูลดังกล่าวนี้จะแบ่งย่อยได้ 4 กรณี ดังนี้

กรณีที่ 1 แหล่งข้อมูลที่มี Fact source ไม่ใช่ข้อมูลเชิงเวลาและ Dimension source ไม่ใช่ข้อมูลเชิงเวลาเช่นกัน

กรณีที่ 2 แหล่งข้อมูลที่มี Fact source ไม่ใช่ข้อมูลเชิงเวลา แต่ Dimension source เป็นข้อมูลเชิงเวลา

กรณีที่ 3 แหล่งข้อมูลที่มี Fact source เป็นข้อมูลเชิงเวลา แต่ Dimension source ไม่ใช้ข้อมูลเชิงเวลา

กรณีที่ 4 แหล่งข้อมูลที่มี Fact source เป็นข้อมูลเชิงเวลาและ Dimension source เป็นข้อมูลเชิงเวลาเช่นกัน

โดยได้กำหนดข้อมูลตัวอย่างสำหรับแหล่งข้อมูล ในกรณีต่างๆ เพื่อคิดค้นอัลกอลิทึมในการสรุปข้อมูลและทำการแสดงผลคำตอบผ่านโปรแกรมประยุกต์

4.1.1 การทดลองกรณีที่ 1 แหล่งข้อมูลที่มี Fact source ไม่ใช่ข้อมูลเชิงเวลาและ Dimension source ไม่ใช่ข้อมูลเชิงเวลาเช่นกัน

ข้อมูลตัวอย่างที่จัดเก็บอยู่ใน ฐานข้อมูลระบบงานประจำวัน เดียวกันนั้นเป็นข้อมูลเกี่ยวกับการจัดการแข่งขันโอลิมปิก โดยประกอบด้วยฐานข้อมูลที่มีโครงตาราง ดังนี้

1) Continent เป็นตารางที่บอกว่ารหัสประเทศดังกล่าว มีชื่อประเทศว่าอะไร และอยู่ในทวีปใด

CTN_ID	Country_Name	Continent_Name

2) Medal เป็นตารางที่บอกว่าเหรียญที่จะมอบเป็นของรางวัลมีอะไรบ้าง บอกเป็นรหัส และชื่อเหรียญ

Medal ID	Medal Name

3) Edition เป็นตารางที่บอกว่าการแข่งขันครั้งใดๆ อยู่ในปีอะไร และตรงกับซีซั่นไหน

Edition_ID	Year	Season

4) Disciplines เป็นตารางที่บอกว่าการแข่งขันกีฬาที่มีรหัสและชื่อรุ่นดังกล่าว จัดอยู่ในชื่อกีฬาประเภทใด เช่น การแข่ง D0050 Ski Jumping และ D0051 Snow board จัดอยู่ในกีฬา Skiing

Dis_ID	Dis_Name	Sport Name

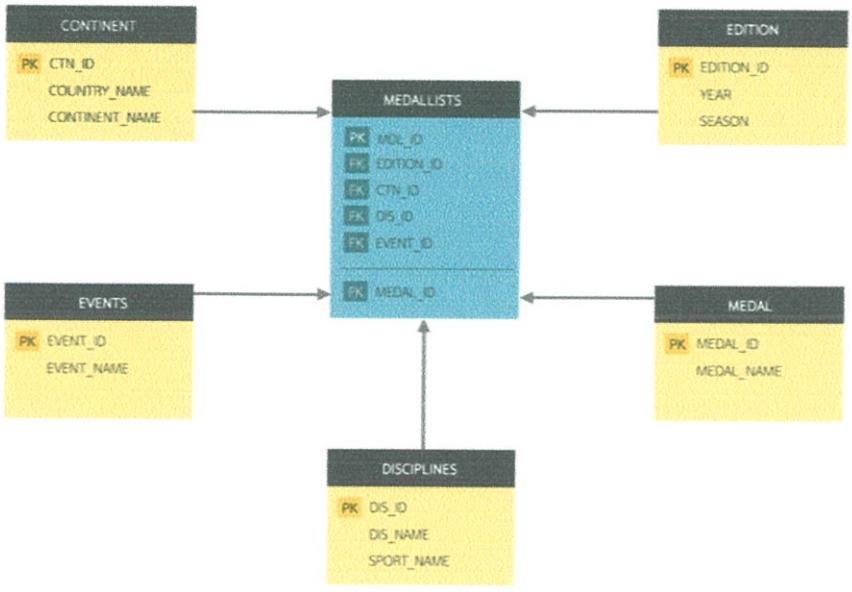
5) Events เป็นตารางที่บอกว่ามีการแข่งขันอะไรเกิดขึ้นในกีฬาโอลิมปิกบ้าง เป็นรหัสงานและชื่องาน

Event_ID	Event_Name

6) Medallists เป็นตารางที่บอกว่ามีผู้เข้าแข่งขันที่ได้รับรางวัลเป็นหญิงหรือชาย ได้รับรางวัลเหรียญอะไร ของการแข่งขันชื่องานอะไร เป็นกีฬารุ่นใด ซึ่งเป็นการได้รับรางวัลสำหรับประเทศนี้ ในปีใด ซึ่งตรงกับซีซันใด

MDL_ID	Edition_ID	CTN_ID	Gender	EventID	DisciplineID	Medal_ID	Season

โดยทั้ง 6 ตารางนี้ต่างอยู่ในฐานข้อมูลระบบงานประจำวันเดียวกัน และเป็นฐานข้อมูลแบบปกติทั้งหมด จึงโหลดตารางทั้งหมดนี้เข้ามาในคลังข้อมูลเพื่อใช้วิเคราะห์ข้อมูลเกี่ยวกับโอลิมปิกตามโครงสร้างแบบดวงดาวที่ได้ออกแบบไว้

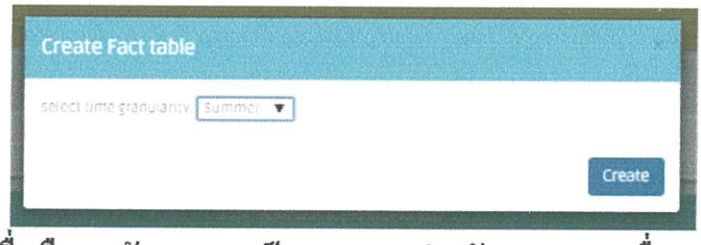


รูป 4.1 โครงสร้างแบบดวงดาวสำหรับคลังข้อมูลที่ Dimension table และ Fact table ไม่ได้มาจากข้อมูลเชิงเวลา

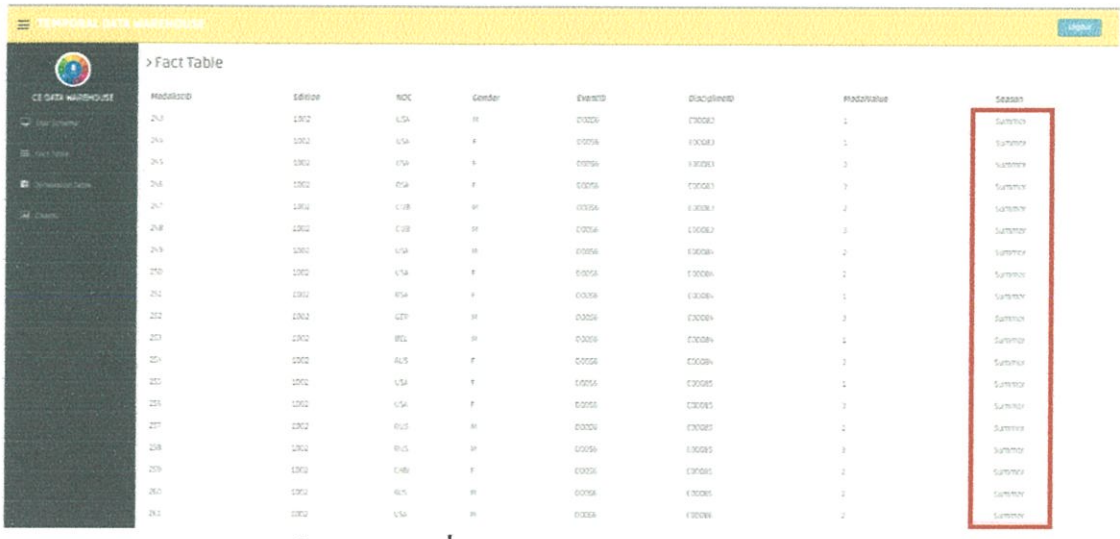
ผลการทำงานของโปรแกรมประยุกต์เป็นดังนี้

4.1.1.1 การสรุป Fact table ตามระดับของเวลาต่างๆ

ตาราง Medallists เป็นตารางที่บอกค่าที่ผู้ใช้ต้องการวัดคือ Medal_ID (ประเภทเหรียญที่ได้รับ) และมี Foreign key มาประกอบ ซึ่งเป็น Primary key ของ Dimension table โดยสิ่งที่ต้องสามารถแสดงผลให้ผู้ใช้เห็นคือการสรุปข้อมูลค่าที่ต้องการวัดด้วยการสอบถามข้อมูลขึ้นมาตามระดับของเวลา ในที่นี้ใช้ตาม Season ซึ่งแต่ละปีที่ผ่านมาก็จะตรงกับ Winter บ้าง Summer บ้าง ในขั้นตอนนี้แสดงผลเป็นตาราง Fact table ที่มีค่าอยู่ในช่วง Season ที่เลือกดู

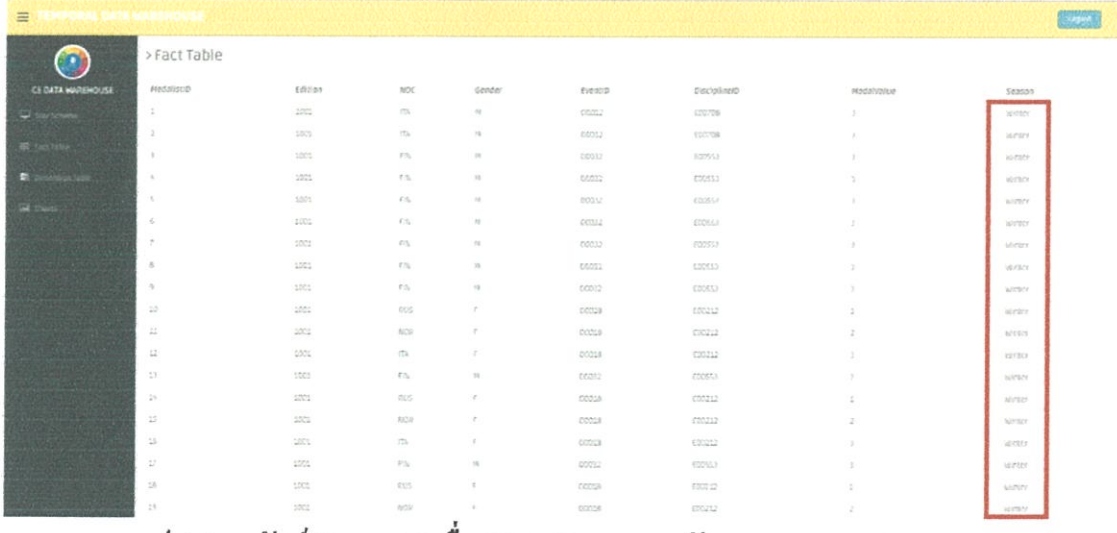


รูป 4.2 เมื่อเลือกระดับของเวลาเป็น Summer สำหรับ Fact table ชื่อ Medallists



รูป 4.3 ผลลัพธ์ Fact table ชื่อ Medallists ตามระดับของเวลาแบบ Summer

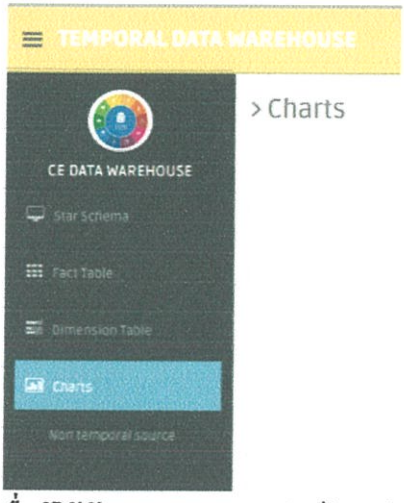
รูป 4.4 เมื่อเลือกระดับของเวลาเป็น Winter สำหรับ Fact table ชื่อ Medallists



รูป 4.5 ผลลัพธ์ Fact table ชื่อ Medallist ตามระดับของเวลาแบบ Summer

4.1.1.2 การตอบคำถามทางคลังข้อมูลที่เกี่ยวข้องกับระดับของเวลาต่างๆ

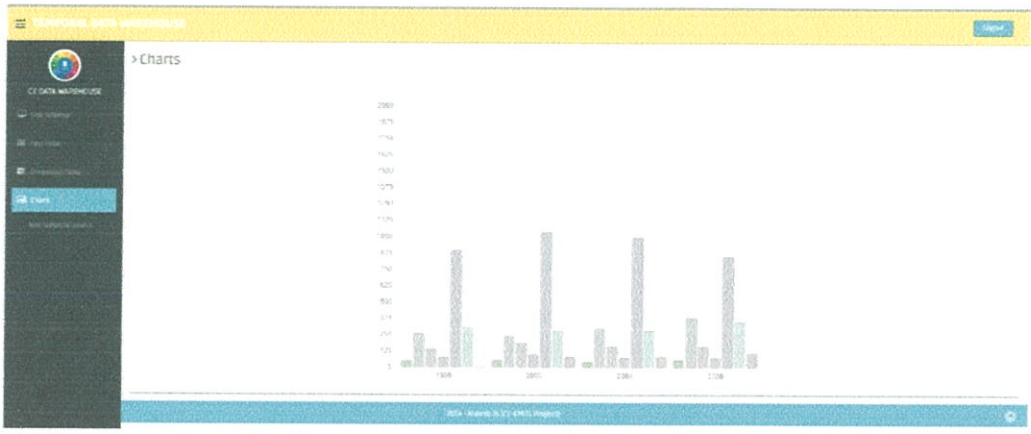
สิ่งที่ต้องแสดงผลให้ผู้ใช้เห็น คือการแสดงผลลัพธ์ที่ได้จากการสรุปข้อมูลตามระดับของเวลาและแสดงผลในรูปกราฟตามคำถามที่ผู้ใช้ต้องการทราบ ดังนี้



รูป 4.6 การเข้าใช้งาน เมื่อผู้ใช้ต้องการทราบผลลัพธ์ในรูปกราฟที่ตอบคำถามทางคลังข้อมูลตามระดับของเวลาที่เตรียมไว้

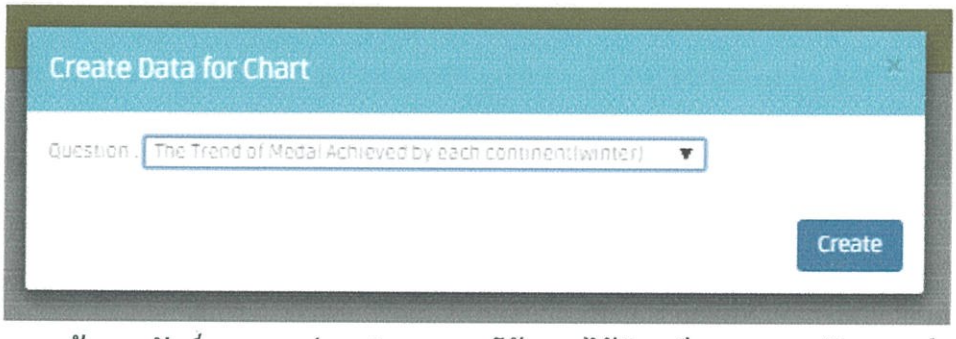
<u>คำถามที่ 1</u> แสดงแนวโน้มการได้รับเหรียญทุกๆรางวัลของแต่ละทวีปในปีที่จัดตรงกับ Summer

รูป 4.7 การเข้าดูผลลัพธ์แบบกราฟจากคำถามแนวโน้มการได้รับเหรียญทุกๆ รางวัลของแต่ละทวีปในปีที่จัดตรงกับ Summer เป็นอย่างไร

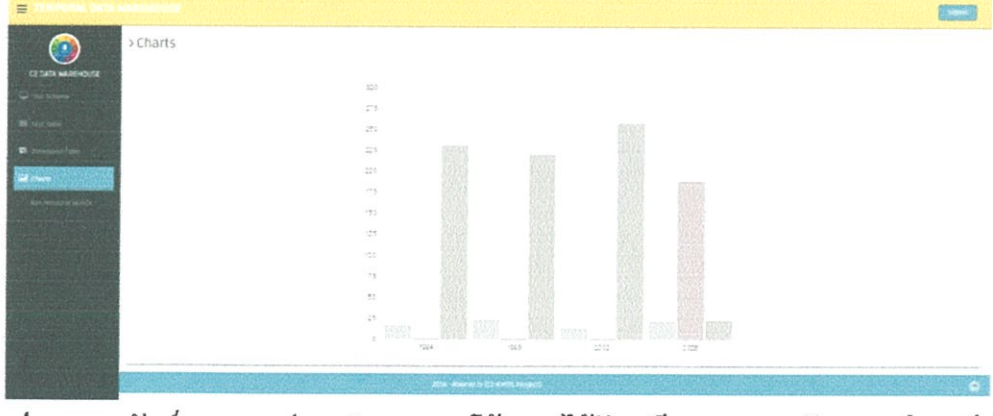


รูป 4.8 ผลลัพธ์แบบกราฟจากคำถามแนวโน้มการได้รับเหรียญทุกๆรางวัลของแต่ละทวีป ในปีที่จัดตรงกับ Summer เป็นอย่างไร

<u>คำถามที่ 2</u> แสดงแนวโน้มการได้รับเหรียญทุกๆ รางวัลของแต่ละทวีปในปีที่จัดตรงกับ Winter

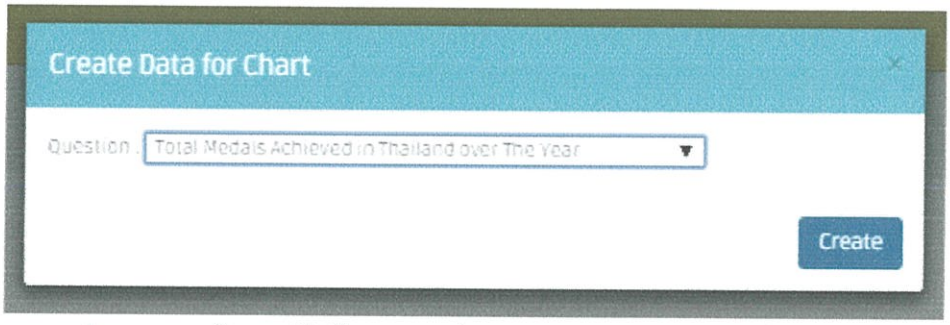


รูป 4.9 การเข้าดูผลลัพธ์แบบกราฟจากคำถามแนวโน้มการได้รับเหรียญทุกๆรางวัลของแต่ละทวีป ในปีที่จัดตรงกับ Winter เป็นอย่างไร

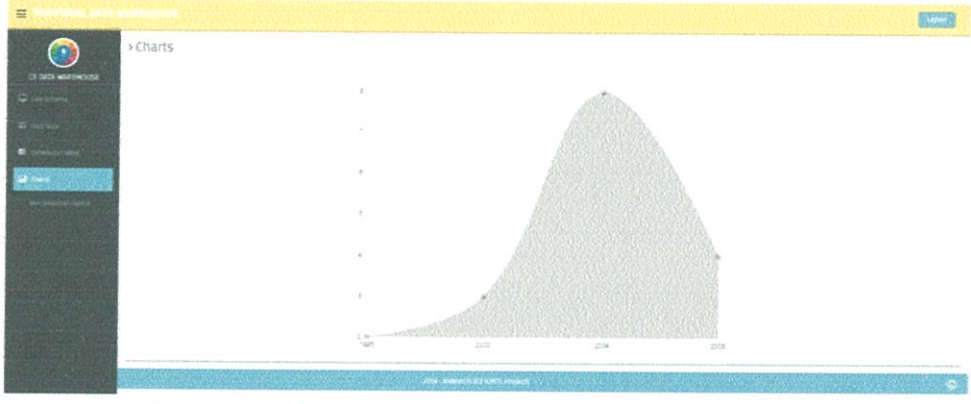


รูป 4.10 ผลลัพธ์แบบกราฟจากคำถามแนวโน้มการได้รับเหรียญทุกๆรางวัลของแต่ละทวีป ในปีที่จัดตรงกับ Winter เป็นอย่างไร

คำถามที่ 3 แสดงจำนวนเหรียญรางวัลที่ประเทศไทยได้รับทั้งหมดในแต่ละปีที่ผ่านมา

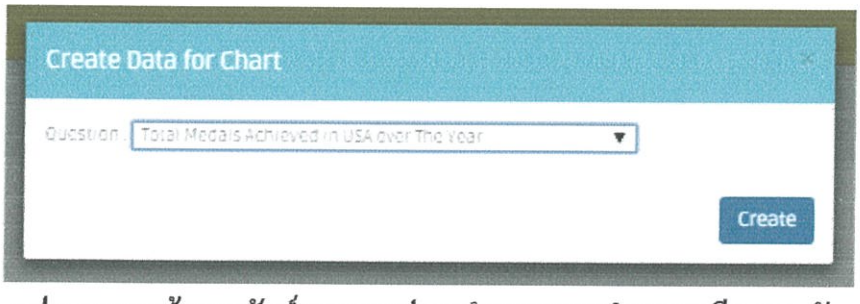


รูป 4.11 การเข้าดูผลลัพธ์แบบกราฟจากคำถามแสดงจำนวนเหรียญรางวัลที่ประเทศไทยได้รับทั้งหมดในแต่ละปีที่ผ่านมา

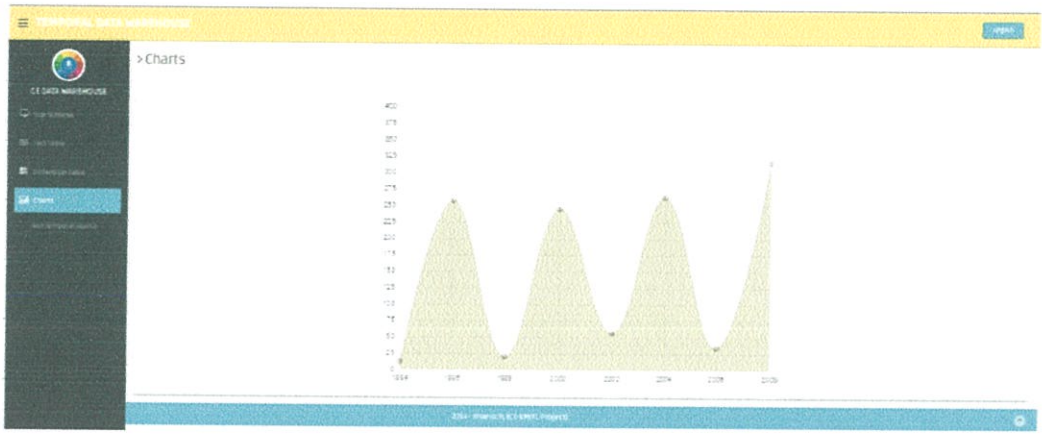


รูป 4.12 ผลลัพธ์แบบกราฟจากคำถามแสดงจำนวนเหรียญรางวัลที่ประเทศไทยได้รับทั้งหมดในแต่ละปีที่ผ่านมา

คำถามที่ 4 แสดงจำนวนเหรียญรางวัลที่สหรัฐอเมริกาได้รับทั้งหมดในแต่ละปีที่ผ่านมา

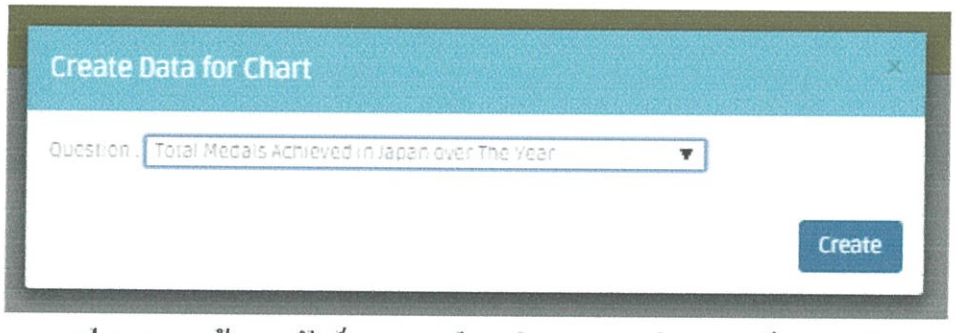


รูป 4.13 การเข้าดูผลลัพธ์แบบกราฟจากคำถามแสดงจำนวนเหรียญรางวัลที่สหรัฐอเมริกาได้รับทั้งหมดในแต่ละปีที่ผ่านมา

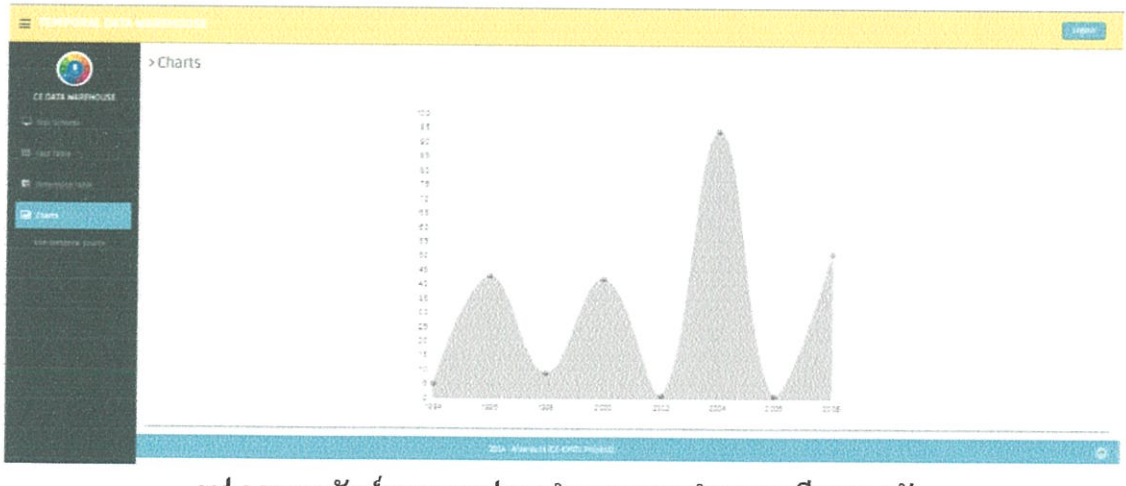


รูป 4.14 ผลลัพธ์แบบกราฟจากคำถามแสดงจำนวนเหรียญรางวัล ที่สหรัฐอเมริกาได้รับทั้งหมดในแต่ละปีที่ผ่านมา

คำถามที่ 5 แสดงจำนวนเหรียญรางวัลที่ญี่ปุ่นได้รับทั้งหมดในแต่ละปีที่ผ่านมา



รูป 4.15 การเข้าดูผลลัพธ์แบบกราฟจากคำถามแสดงจำนวนเหรียญรางวัล ที่ญี่ปุ่นได้รับทั้งหมดในแต่ละปีที่ผ่านมา



รูป 4.16 ผลลัพธ์แบบกราฟจากคำถามแสดงจำนวนเหรียญรางวัล ที่ญี่ปุ่นได้รับทั้งหมดในแต่ละปีที่ผ่านมา

4.1.3 การทดลองกรณีที่ 2 แหล่งข้อมูลที่มี Fact source ไม่ใช่ข้อมูลเชิงเวลา แต่ Dimension source เป็นข้อมูลเชิงเวลา

ข้อมูลตัวอย่างที่จัดเก็บอยู่ในฐานข้อมูลระบบงานประจำวั เดียวกันนั้นเป็นข้อมูลเกี่ยวกับการขายสินค้าขององค์กรต่างๆที่เป็นสาขาย่อยของบริษัทแห่งหน่ง โดยประกอบด้วยฐานข้อมูลที่มีโครงตาราง ดังนี้

1) CityLocation เป็นตารางที่บอกสถานที่ว่าแต่ละเมืองที่มีรหัสเมืองดังนี้ อยู่ในประเทศอะไร บอกเป็นรหัสประเทศ

Loc_ID	City_Name	Country_Code

2) ORG เป็นตารางที่บอกข้อมูลเกี่ยวกับองค์กรที่รับสินค้ามาขายซึ่งจัดตั้งสาขาย่อย องค์กรนี้มีรหัสองค์กรและชื่อว่าอย่างไร ตั้งอยู่ที่เมืองไหน ชื่อสาขาที่ตั้งชื่อว่าอะไร ใครเป็นผู้ดูแลการขายสินค้า

ORG_ID	ORG_Name	Loc_ID	Branch Name	Emp_Name

3) Promotion เป็นตารางที่บอกโปรโมชั่นที่จัดขึ้นสำหรับแต่ละสินค้า ประกอบด้วย รหัสโปรโมชั่น ชื่อโปรโมชั่น และประเภทของโปรโมชั่น เช่น ลดราคา หรือสะสมแต้ม เป็นต้น

Pro_ID	Pro_Name	Pro_Type

4) Product เป็นตารางที่บอกรายละเอียดเกี่ยวกับสินค้าที่ขาย ประกิบด้วย รหัสสินค้า ชื่อประเภทสินค้า ชื่อประเภทสินค้าย่อย ชื่อยี่ห้อสินค้า ชื่อรุ่นของแต่ละยี่ห้อ รวมทั้งราคาต่อหน่วย

Prod_ID	ProdC_Name	ProdSC_Name	Brand Name	ProdFeat	UnitPrice

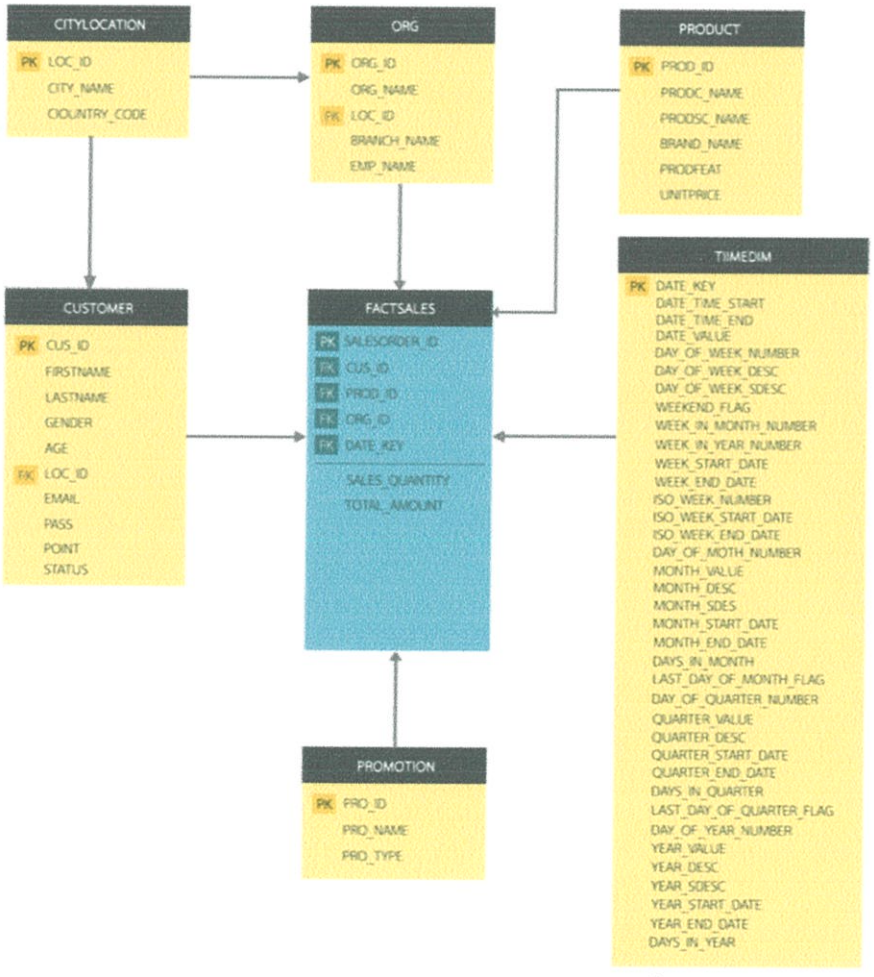
5) Customer เป็นตารางที่บอกรายละเอียดลูกค้าซึ่งจะต้องเป็นสมาชิกของบริษัท

Cus_ID	FirstName	LastName	Gender	Age	Loc_ID	Email	Pass	Point	Status

6) Sales เป็นตารางที่บอกรายละเอียดเกี่ยวกับการขายสินค้าว่าสินค้านี้ที่ขายโดยองค์กรใด ของเมืองอะไร ใครซื้อไป ตอนนั้นโปรโมชั่นของสินค้าคือโปรโมชั่นอะไร ซื้อไปกี่ชิ้น ราคาก่อนลด เป็นเท่าไร เมื่อลดราคาแล้ว ต้องจ่ายสินค้านี้ราคาเท่าไร ขายไปเมื่อใด

SaleOrder ID	Cus_ID	Prod_ID	Pro_ID	Org_ID	Payment type	Unit Price	Sales_ Quantity	Sales_ Amount	Discount	Total_ Amount	Date_Key

โดยทั้ง 6 ตารางนี้ต่างอยู่ในฐานข้อมูลระบบงานประจำวันเดียวกัน มี Sales เป็นฐานข้อมูลแบบปกติที่มีการจัดเก็บข้อมูลเพียงอย่างเดียว ส่วน CityLocation, Org, Promotion, Product, Customer เป็นฐานข้อมูลเชิงเวลาที่มีการเปลี่ยนแปลงข้อมูลเรื่อยๆ แล้วจึงโหลดตารางทั้งหมดนี้เข้ามาในคลังข้อมูล เพื่อใช้วิเคราะห์ข้อมูลเกี่ยวกับการขายสินค้าของบริษัท ตามโครงสร้างแบบดวงดาวที่ได้ออกแบบไว้



รูป 4.17 โครงสร้างแบบดวงดาวสำหรับคลังข้อมูล ที่ Fact table มาจากข้อมูลที่ไม่ขึ้งเวลาและ Dimension table มาจากข้อมูลเชิงเวลา

ผลการทำงานของโปรแกรมประยุกต์เป็นดังนี้

4.1.2.1 การสรุป Fact table ตามระดับของเวลาต่างๆ

ตาราง Sales เป็นตารางที่บอกค่าที่ผู้ใช้ต้องการวัดคือ Sales_Quantity (ปริมาณสินค้าที่ขายออกไป) กับ Total_Amount (ราคาสินค้าที่ต้องจ่าย) และมี Foreign key มาประกอบ ซึ่งเป็น Primary key ของ Dimension table โดยสิ่งที่ต้องสามารถแสดงผลให้ผู้ใช้เห็นคือการสรุปข้อมูลค่าที่ต้องการวัดด้วยการสอบถามข้อมูลขึ้นมาตามระดับของเวลา แต่ในโปรแกรมประยุกต์ข้างต้นนี้ ยังไม่ได้ทำในส่วนการสรุปข้อมูลตามระดับของเวลาเป็นตัวซอฟต์แวร์ ได้ทดลองแค่เพียงยกตัวอย่างการเปลี่ยนแปลงของ Dimension table ต่างๆ ให้มีการเปลี่ยนแปลง เช่น การปรับปรุงบางแถวข้อมูล, การลบข้อมูลบางแถว ในทุกๆวัน เป็นบางชั่วโมง แล้วคิดว่าจะต้องสรุปผลและแสดงเป็น Fact table อย่างไร มีหน้าตาแบบไหน เมื่อ Dimension table มีการเปลี่ยนแปลง จัดทำเป็นไฟล์ใน Microsoft Excel

4.1.2 การทดลองกรณีที่ 3 แหล่งข้อมูลที่มี Fact source เป็นข้อมูลเชิงเวลา แต่ Dimension source ไม่ใช้ข้อมูลเชิงเวลา

ข้อมูลตัวอย่างที่จัดเก็บอยู่ใน ฐานข้อมูลระบบงานประจำวัน เดียวกันนั้นเป็นข้อมูลเกี่ยวกับการดูแลคอกวัวโดยประกอบด้วยฐานข้อมูลที่มีโครงตาราง ดังนี้

1) FDYD เป็นตารางที่บอกว่าลานอาหารรหัสดังกล่าว มีชื่อว่าอะไร ตั้งอยู่ที่ไหน

FDYD_ID	FDYD_Name	FDYD_Loction

2) Lot เป็นตารางที่บอกว่ารหัส LOT ดังกล่าว เป็นสายพันธุ์อะไร พร้อมคำอธิบายสายพันธุ์

LOT_IDNUM	LOT_Breed	LOT_Des

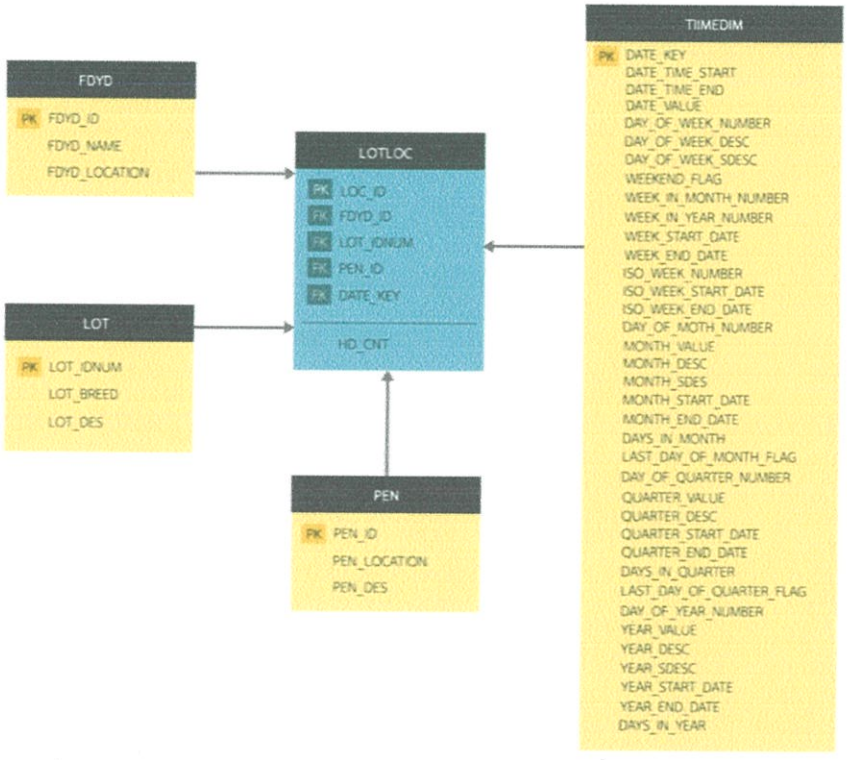
3) Pen เป็นตารางที่บอกเลขคอก เป็นที่ให้วัวแต่ละ LOT มาอาศัยอยู่ บอกที่ตั้งและคำอธิบายคอก

PEN_ID	PEN_Location	PEN_Des

4) LotLoc เป็นตารางที่บอกว่าดำเนินการดูแลจัดสรรให้วัวจำนวนเท่านี้ อยู่ในคอกไหน กินอาหารที่ลานอาหารไหน เมื่อวันที่เท่าใด

LOC_ID	FDYD_ID	LOT_IDNUM	PEN_ID	HD_CNT	DATE_KEY

โดยทั้ง 4 ตารางนี้ต่างอยู่ใน ฐานข้อมูลระบบงานประจำวัน เดียวกัน มี FDYD, Lot และ Pen เป็นฐานข้อมูลปกติ ส่วน LotLoc เป็นฐานข้อมูลเชิงเวลา จึงโหลดตารางทั้งหมดนี้เข้ามาในคลังข้อมูล เพื่อใช้วิเคราะห์ข้อมูลเกี่ยวกับฝูงวัว ตามโครงสร้างแบบดวงดาวที่ได้ออกแบบไว้

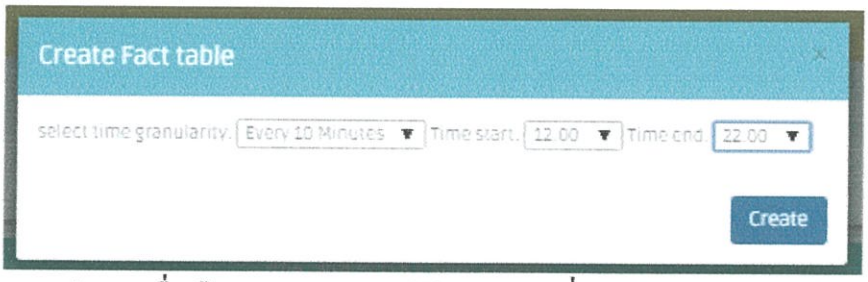


รูป 4.18 โครงสร้างแบบดวงดาวสำหรับคลังข้อมูลที่ Dimension table มาจากข้อมูลที่ไม่ใช่เวลาและ Fact table มาจากข้อมูลเชิงเวลา

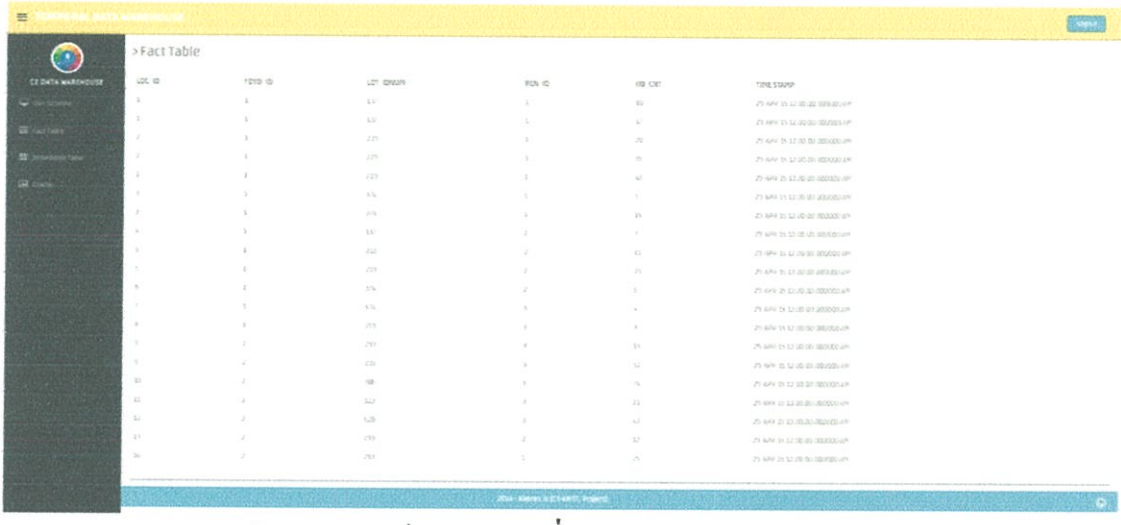
ผลการทำงานของโปรแกรมประยุกต์เป็นดังนี้

4.1.2.1 การสรุป Fact table ตามระดับของเวลาต่างๆ

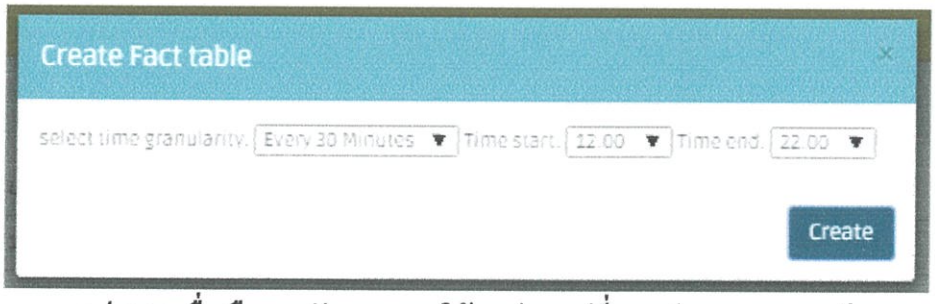
ตาราง LotLoc เป็นตารางที่บอกค่าที่ผู้ใช้ต้องการวัดคือ HD_CNT (จำนวนวัว) และมี Foreign key มาประกอบ ซึ่งเป็น Primary key ของ Dimension table โดยสิ่งที่ต้องสามารถแสดงผลให้ผู้ใช้เห็นคือการสรุปข้อมูลค่าที่ต้องการวัดด้วยการสอบถามข้อมูลขึ้นมาตามระดับของเวลาในที่นี้ผู้ใช้สามารถเลือกว่าจะต้องการให้สรุปการเปลี่ยนแปลงข้อมูลทุกช่วงเวลาเท่าใด ตั้งแต่เวลาไหนถึงเวลาไหน ในขั้นตอนนี้แสดงผลเป็นตาราง Fact table ที่ทำการสรุปข้อมูลตามระดับของเวลาที่ผู้ใช้ต้องการ ซึ่งสามารถเลือกช่วงเวลาตามเวลาที่ข้อมูลมีอยู่ในฐานข้อมูลได้ตั้งแต่ 12:00 น. ถึง 22:00 น. ด้วยการเลือกเวลาเริ่มต้นและเวลาสิ้นสุด จากนั้นตามช่วงเวลาดังกล่าวเลือกได้ว่าจะให้สรุปการเปลี่ยนแปลงข้อมูลทุกๆ 10 นาที 30 นาที หรือ 1 ชั่วโมง



รูป 4.19 เมื่อเลือกระดับของเวลาให้สรุปการเปลี่ยนแปลงทุกๆ 10 นาที ตั้งแต่ 12:00 น. ถึง 22:00 น. สำหรับ Fact table ชื่อ LotLoc

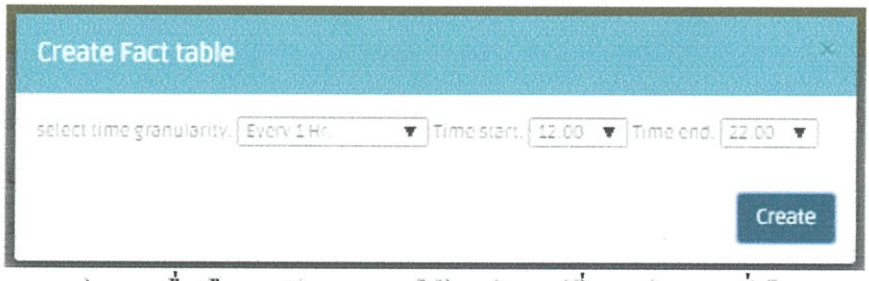


รูป 4.20 ผลลัพธ์ Fact table ชื่อ LotLoc ตามระดับของเวลาให้สรุปการเปลี่ยนแปลงทุกๆ 10 นาที ตั้งแต่ 12:00 น. ถึง 22:00 น.

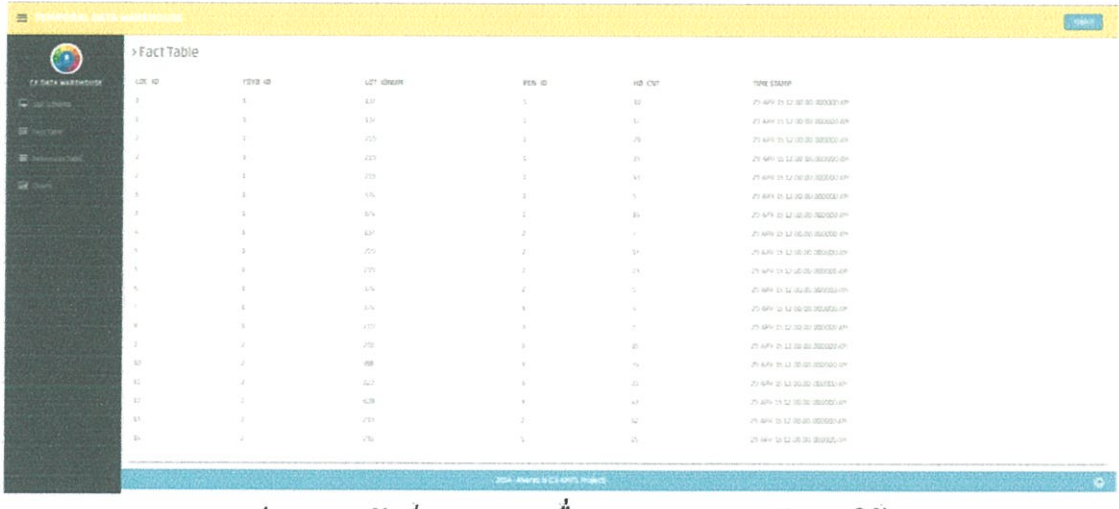


รูป 4.21 เมื่อเลือกระดับของเวลาให้สรุปการเปลี่ยนแปลงทุกๆ 30 นาที ตั้งแต่ 12:00 น. ถึง 22:00 น. สำหรับ Fact table ชื่อ LotLoc

รูป 4.22 ผลลัพธ์ Fact table ชื่อ LotLoc ตามระดับของเวลา
ให้สรุปการเปลี่ยนแปลงทุกๆ 30 นาที ตั้งแต่ 12:00 น. ถึง 22:00 น.



รูป 4.23 เมื่อเลือกระดับของเวลาให้สรุปการเปลี่ยนแปลงทุกๆชั่วโมง
ตั้งแต่ 12:00 น. ถึง 22:00 น. สำหรับ Fact table ชื่อ LotLoc



รูป 4.24 ผลลัพธ์ Fact table ชื่อ LotLoc ตามระดับเวลาให้
สรุปการเปลี่ยนแปลงทุกๆชั่วโมง ตั้งแต่ 12:00 น. ถึง 22:00 น.

4.1.4 การทดลองกรณีที่ 4 แหล่งข้อมูลที่มี Fact source เป็นข้อมูลเชิงเวลาและ Dimension source เป็นข้อมูลเชิงเวลาเช่นกัน

ข้อมูลตัวอย่างที่จัดเก็บอยู่ในฐานข้อมูลระบบงานประจำวันเดียวกันนั้นเป็นข้อมูลเกี่ยวกับการดูแลคอกวัวโดยประกอบด้วยฐานข้อมูลที่มีโครงตาราง ดังนี้

1) FeedYard เป็นตารางที่บอกว่าลานอาหารรหัสดังกล่าว ตั้งอยู่ที่ไหน บรรจุวัวได้กี่ตัว

FDYD_ID	FDYD_Location	FDYD_Size

2) Lot เป็นตารางที่บอกว่ารหัส LOT ดังกล่าว เป็นสายพันธุ์อะไร

LOT_ID_Num	LOT_Breed

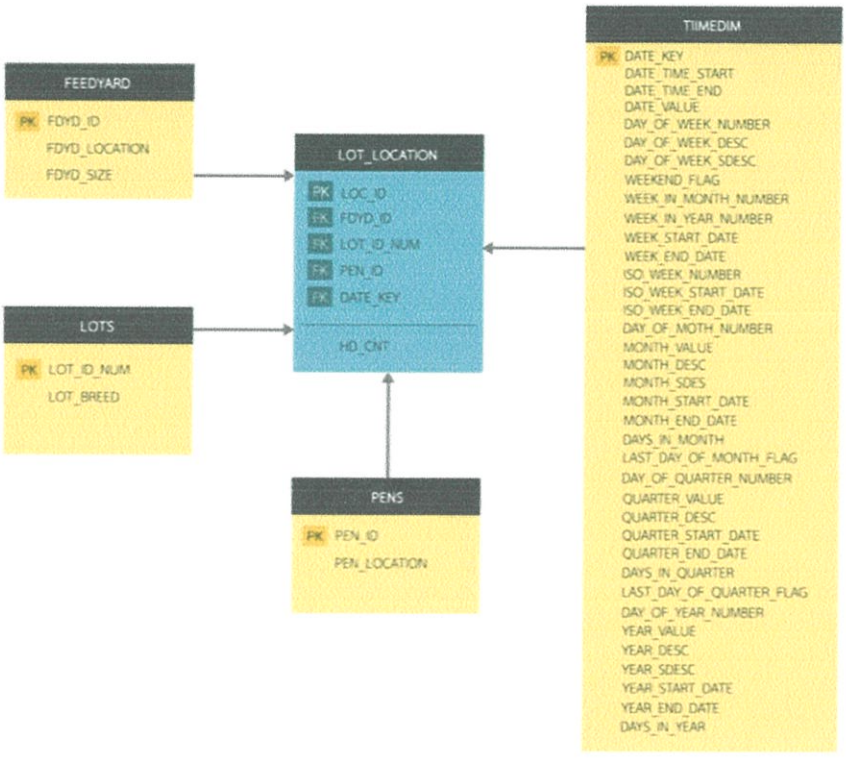
3) Pen เป็นตารางที่บอกเลขคอก เป็นที่ให้วัวแต่ละ LOT มาอาศัยอยู่ บอกที่ตั้ง

PEN_ID	PEN_Location

4) Lot_Location เป็นตารางที่บอกว่าดำเนินการดูแลจัดสรรให้วัวจำนวนเท่านี้ อยู่ในคอกไหน กินอาหารที่ลานอาหารไหน เมื่อวันที่เท่าใด

LOC_ID	FDYD_ID	LOT_ID_NUM	PEN_ID	HD_CNT	DATE_KEY

โดยทั้ง 4 ตารางนี้ต่างอยู่ใน ฐานข้อมูลระบบงานประจำวันเดียวกัน มี FeedYard, Lot, Pen และ Lot_Location เป็นฐานข้อมูลแล้วจึงโหลดตารางทั้งหมดนี้เข้ามาในคลังข้อมูล เพื่อใช้วิเคราะห์ข้อมูลเกี่ยวกับฝูงวัว ตามโครงสร้างแบบดวงดาวที่ได้ออกแบบไว้



รูป 4.25 โครงสร้างแบบดวงดาวสำหรับคลังข้อมูลที่

Dimension table และ Fact table มาจากข้อมูลเชิงเวลา

ผลการทำงานของโปรแกรมประยุกต์ เป็นดังนี้

4.1.2.1 การสรุป Fact table ตามระดับของเวลาต่างๆ

ตาราง Lot_Location เป็นตารางที่บอกค่าที่ผู้ใช้ต้องการวัดคือ HD_CNT (จำนวนวัว) กับและมี Foreign key มาประกอบ ซึ่งเป็น Primary key ของ Dimension table โดยสิ่งที่ต้องสามารถแสดงผลให้ผู้ใช้เห็นคือการสรุปข้อมูลค่าที่ต้องการวัดด้วยการสอบถามข้อมูลขึ้นมาตามระดับของเวลา แต่ในโปรแกรมประยุกต์ข้างต้นนี้ ยังไม่ได้ทำในส่วนการสรุปข้อมูลตามระดับของเวลาเป็นตัวซอฟต์แวร์ ได้ทดลองแค่เพียงยกตัวอย่างการเปลี่ยนแปลงของ Dimension table ต่างๆ และ Fact table ให้มีการเปลี่ยนแปลง เช่น ปรับปรุงบางแถว, ลบข้อมูลบางแถว ในทุกๆวัน เป็นบางชั่วโมง แล้วคิดว่าจะต้องสรุปผลและแสดงเป็น Fact table อย่างไร มีหน้าตาแบบไหน เมื่อ Dimension table และ Fact table มีการเปลี่ยนแปลง คือจะต้องมีการดูข้อมูลในช่วงเวลาที่ตรงกันในตารางทั้งสองแบบ จากนั้นจึงได้เป็นข้อมูลที่สรุปตามการเปลี่ยนแปลง จัดทำเป็นไฟล์ใน Microsoft Excel

บทที่ 5
บทสรุปและข้อเสนอแนะ

5.1 บทสรุปของโครงงาน

ในการศึกษาและพัฒนาโครงงานการประยุกต์ภาษาเอสคิวแอลกับข้อมูลเชิงเวลาโดยใช้ Oracle Flashback Data Archive ได้ข้อสรุปดังต่อไปนี้

Oracle Flashback Data Archive นั้นได้ถูกออกแบบมาเพื่อแก้ไขปัญหาสำหรับความผิดพลาดที่เกิดจากผู้ใช้งาน เช่น การบังเอิญลบข้อมูล หรือการบังเอิญลบทั้งตารางทิ้งโดยไม่ได้ตั้งใจ ให้สามารถทำการกู้คืนข้อมูลที่หายไปนั้นกลับมาโดยง่าย ซึ่ง Oracle Flashback Data Archive นั้นจะมีขนาดใหญ่ได้ ตามเท่าผู้ใช้งานกำหนด หากมีขนาดใหญ่มาก ก็จะสามารถเก็บข้อมูลได้นานยิ่งขึ้น

Oracle Flashback Data Archive นั้นจะมองฐานข้อมูลในอดีตเป็นจุดของเวลาว่า ณ เวลานั้นฐานข้อมูลมีข้อมูลเป็นอย่างไรดังนั้นการกระทำทุกอย่างไม่ว่าจะเป็นการจัดเก็บข้อมูล, การลบข้อมูลหรือการปรับปรุงข้อมูลจะส่งผลทำให้ฐานข้อมูล ณ เวลาต่างกันมีข้อมูลที่แตกต่างกันทั้งสิ้นซึ่งการสอบถามข้อมูลที่มีอยู่ใน Oracle Flashback Data Archive จะใช้สิ่งที่เรียกว่า Oracle Flashback Query ซึ่งเป็นคำสั่งพิเศษที่ไว้ใช้ควบคู่กับ Oracle Flashback Data Archive เท่านั้น

Oracle Flashback Query นั้นมีสามารถในการค้นหาข้อมูลที่เป็นเวอร์ชันในอดีตได้ถึงระดับวินาทีซึ่งสามารถนำความสามารถตรงนี้มาพัฒนาในการใช้กับคลังข้อมูลได้โดยง่ายแต่ในการใช้งานความสามารถตรงนี้ในการสร้างคลังข้อมูลนั้นยังคงต้องใช้โปรแกรมประยุกต์ในการควบคุมการทำงานของ Oracle Flashback Query อยู่ เนื่องจาก Oracle Flashback Query นั้นมีลักษณะในการทำการสอบถามเฉพาะกิจ (Ad-hoc query) เช่นเดียวกับภาษาเอสคิวแอล ดังนั้นความสามารถในการทำซ้ำ (Loop) ควรให้เป็นหน้าที่ของโปรแกรมประยุกต์ในการจัดการ

การสร้างคลังข้อมูลโดยมีแหล่งข้อมูลมาจาก Oracle Flashback Data Archive อาจจะมีความซ้ำซ้อนของข้อมูลเกิดขึ้น อันเนื่องมาจากข้อมูลสองช่วงเวลาไม่มีการเปลี่ยนแปลงใดๆ ซึ่งปัญหาตรงนี้ระบบการจัดการฐานข้อมูล Oracle ได้จัดการให้เนื่องจากระบบการจัดการฐานข้อมูล Oracle ใช้แบบจำลองข้อมูลเชิงสัมพันธ์เป็นแบบจำลองเพื่อนำเสนอข้อมูลระดับ Logical ดังนั้นหนึ่งในคุณสมบัติของแบบจำลองเชิงสัมพันธ์คือเนื้อข้อมูลจะต้องมีแถวไม่ซ้ำกันและเมื่อใช้ภาษาเอสคิวแอลในการค้นหาข้อมูลแล้ว ผลลัพธ์ที่ได้จากการสืบค้นนั้น จะต้องยังคงคุณสมบัติเป็นแบบจำลองเชิงสัมพันธ์อยู่

Oracle Flashback Data Archive นั้นมีการเก็บข้อมูลแบบ First In First Out (FIFO) ดังนั้นไม่ได้หมายความว่าข้อมูลจะคงอยู่ถาวรตลอดไป ทั้งนี้ขึ้นอยู่กับว่ามีการสร้างขนาดพื้นที่จัดเก็บใน Oracle Flashback Data Archive ไว้เท่าใด ซึ่งหากพื้นที่จัดเก็บมีขนาดใหญ่ ก็จะสามารถเก็บข้อมูลชุดเก่าๆ ไว้ได้มากและนานยิ่งขึ้น ซึ่งตรงนี้เราจะต้องดูแลโดยการสรุปข้อมูลที่มีอยู่ใน Oracle Flashback Data Archive ตามช่วงเวลาต่างๆที่ผู้ออกแบบคลังข้อมูลกำหนดเพื่อที่จะรักษาข้อมูลตรงนี้ไว้ไม่ให้เกิดการสูญหาย

มุมมองของเวลาในคลังข้อมูลยังคงมีความจำเป็นอยู่ ถึงแม้ Oracle Flashback Data Archive จะมีมุมมองของเวลาผูกติดอยู่กับแหล่งข้อมูลคือเราสามารถนำแหล่งข้อมูลตามช่วงเวลาต่างๆมารวมกันได้เพื่อที่จะสร้างเป็นคลังข้อมูล แต่ในคลังข้อมูลนั้นควรจะมีมุมมองของเวลาไว้เพื่อที่จะนำเสนอข้อมูลในช่วงเวลาต่างๆ ซึ่งการนำเสนอนั้นจะเกิดจากการสรุปข้อมูลจากแหล่งข้อมูลที่ได้รวมกันจาก Oracle Flashback Data Archive เรียบร้อยแล้ว

5.2 ปัญหาอุปสรรคและแนวทางแก้ไข

1) หากจะต้องใช้ Oracle Flashback Data Archive สร้างคลังข้อมูลนั้น จะต้องสร้างข้อมูลที่จัดเก็บอยู่ใน Oracle Flashback Data Archive ก่อนซึ่งการเพิ่มข้อมูลลงไปยัง Oracle Flashback Data Archive นั้นไม่สามารถตรงๆได้เลย ระบบการจัดการฐานข้อมูลจะเป็นผู้จัดการตรงนี้ให้ ดังนั้นเราจะไม่สามารถกำหนดเวลาของข้อมูลเองได้ ข้อมูลที่ใช้ใน Oracle Flashback Data Archive จะต้องเป็นเวลาจริงๆ ซึ่งวิธีแก้ปัญหาในการทำโครงงานนี้คือต้องทำการออกแบบไว้ล่วงหน้าว่าต้องเพิ่มข้อมูลลงตารางวันไหน เวลาเท่าไรบ้าง เพื่อให้ได้ชุดข้อมูลสำหรับการทดลองในโครงงานนี้

2) คลังข้อมูลมีคำถามที่ใช้ถามอย่างหลากหลายและโดยส่วนใหญ่มักจะเป็นคำถามทางด้านธุรกิจ ดังนั้นทำให้การพัฒนาโปรแกรมประยุกต์ที่ใช้สร้างคลังข้อมูลในโครงงานนี้ทำได้ค่อนข้างยาก เพราะไม่ค่อยมีความรู้และความเข้าใจในทางธุรกิจพอสมควร จึงแก้ไขโดยการคัดเฉพาะคำถามที่สำคัญและพบบ่อยให้ผู้ใช้งานเลือกใช้ โดยกำหนดคำถามโดยผู้พัฒนาโปรแกรมประยุกต์

3) การสร้างสภาพแวดล้อมของฐานข้อมูลในการทดลองและศึกษาโครงงานนี้ค่อนข้างยากและจำเป็นต้องใช้ข้อมูลที่เป็นข้อมูลตามสภาพเวลาจริง จึงมีแนวทางป้องกันข้อผิดพลาดที่อาจจะเกิดขึ้นโดยจำลองสภาพแวดล้อมทั้งหมดไว้ในโปรแกรม Virtual Box

5.3 แนวทางในการพัฒนาต่อ

โครงงานการประยุกต์ภาษาเอสคิวแอลกับข้อมูลเชิงเวลาโดยใช้ Oracle Flashback Data Archive มาแนวทางในการศึกษาและพัฒนาต่อดังนี้

1) ศึกษา Flex และ Bison ของหลักการสร้าง Compiler ในการสร้างภาษาสำหรับการสร้างและถามคำถามเกี่ยวกับคลังข้อมูลเพื่อให้โปรแกรมประยุกต์มีความยืดหยุ่นในการสร้างและตอบคำถามเกี่ยวกับคลังข้อมูลมากยิ่งขึ้น

2) ศึกษาฐานข้อมูลที่มีการจัดการเกี่ยวกับ Temporal data เพื่อสร้างแหล่งข้อมูลที่สามารถบริหารจัดการเกี่ยวกับ Temporal data ได้

3) ศึกษาทฤษฎีการทำเหมืองข้อมูลและข้อมูลการตัดสินใจทางธุรกิจเพื่อเพิ่มขีดความสามารถให้แก่ระบบคลังข้อมูล

บรรณานุกรม

[1] Richard T. Snodgrass. September 3, 1998. **Managing Temporal Data. A Five-Part Series.** TR-28. A TIMECENTER Technical Report

[2] Kulkarni, G. K. and Michels, J. E. 2012. **Temporal features in SQL: 2011.** ACM SIGMOD Record. 41(3): 34-43 (2012)

[3] Abello´, A. and Marti´n, C. **A Bitemporal Storage Structure for a Corporate Data Warehouse.**In: Proc. ICEIS. (2003) 177–183

[4] Martin, C. and Abello, A. (2003) **A Temporal Study of Data sources to Load a Corporate Data Warehouse.** In proceedings of the 5th International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2003), Prague, Czech Republic, September 3-5, 2003, pp. 119-118.

[5] Abello´, A. and Marti´n, C. **The Data Warehouse: An Object-Oriented Temporal database.** In: Proc. JISBD 2003, Alicante, Spain (2003) 675–684 [5]

[6] Devlin, B. **Managing time in the Data Warehouse.** InfoDB 11(1) (1997) 7–12

[7] Iftikhar, N. and Pedersen, T. B. 2010. **Schema Design Alternatives for Multi-granular Data Warehousing.** In Pablo Garcia Bringas; Abdelkader Hameurlain & Gerald Quirchmayr, ed., 'DEXA (2)', Springer, pp. 111-125.

[8] Stat4u. What are Slowly Changing Dimensions?. [Online]. Available : http://datawarehouse4u.info/SCD-Slowly-Changing-Dimensions.html

[9] Rizzi, S. and Golfarelli, M. **What Time is it in the Data Warehouse?.** [Online]. Available : http://bias.csr.unibo.it/golfarelli/Papers/DAWAK06.pdf

[10] Oracle Help Center. 2014. **Oracle Database Online Documentation 12c Release 1 (12.1). Database Administration. 12 Using Oracle Flashback Technology.** [Online]. Available : http://docs.oracle.com/cd/E11882_01/appdev.112/e41502/adfns_flashback.htm#ADFNS598

[11] Oracle Help Center. 2014. **Oracle Database Online Documentation 12c Release 1 (12.1). Database Administration. 16 Managing Undo.** [Online]. Available : http://docs.oracle.com/database/121/ADMIN/undo.htm#ADMIN10180

[12] Oracle Help Center. 2014. **Oracle Database Online Documentation 12c Release 1 (12.1). Database Administration. 11 Managing the Redo Log.** [Online]. Available : http://docs.oracle.com/database/121/ADMIN/onlineredo.htm#ADMIN007

[13] Oracle Help Center. 2014. **Oracle Database Online Documentation 12c Release 1 (12.1). Database Administration. CREATE FLASHBACK ARCHIVE.** [Online]. Available : https://docs.oracle.com/database/121/SQLRF/statements_5011.htm #SQLRF20008

[14] Oracle Help Center. 2014. **Oracle Database Online Documentation 12c Release 1 (12.1). Database Administration. ALTER FLASHBACK ARCHIVE.** [Online]. Available : https://docs.oracle.com/database/121/SQLRF/statements_1010.htm #SQLRF20009

[15] Oracle Help Center. 2014. **Oracle Database Online Documentation 12c Release 1 (12.1). Database Administration.** Oracle Help Center. 2014. **Single-Row Function.** [Online]. Available : https://docs.oracle.com/database/121/SQLRF/functions002.htm #SQLRF51181

[16] Oracle Help Center. 2014. **Oracle Database Online Documentation 12c Release 1 (12.1). Database Administration.** Oracle Help Center. 2014. **Aggregate Function.** [Online]. Available : https://docs.oracle.com/database/121/SQLRF/functions003.htm #SQLRF20035

[17] Oracle Help Center. 2014. **Oracle Database Online Documentation 12c Release 1 (12.1). Data Warehousing and Business Intelligence. 1 What Is Data Mining?.** [Online]. Available : https://docs.oracle.com/database/121/DMCON/process.htm#DMCON002

[18] Oracle Help Center. 2014. **Oracle Database Online Documentation 12c Release 1 (12.1). Data Warehousing and Business Intelligence. 1 Data Warehousing Concepts.** [Online].Available : https://docs.oracle.com/cd/E11882_01/server.112/e25554/concept.htm#DWHSG8063

[19] Oracle White Paper. **Automatic Data Optimization with Oracle Database 12c.** [Online]. Available : http://www.oracle.com/technetwork/database/automatic-data-optimization-wp-12c-1896120.pdf

[20] กิตติพงษ์ กลมกล่อม. 2552. **การออกแบบและพัฒนาคลังข้อมูล (คลังข้อมูล).** กรุงเทพฯ: เคทีพี คอมพ์ แอนด์ คอนซัลท์

[21] ชนวัฒน์ ศรีสอ้าน. 2551. **ฐานข้อมูล คลังข้อมูล และเหมืองข้อมูล.** กรุงเทพฯ: สำนักพิมพ์มหาวิทยาลัยรังสิต

[22] Stewashton. 2014. **SQL for date ranges, gaps and overlaps.** [Online]. Available : http://stewashton.wordpress.com/2014/03/11/sql-for-date-ranges-gaps-and-overlaps

[23] Hall, T. **Temporal Validity in Oracle Database 12c Release 1 (12.1).** [Online]. Available : http://www.oracle-base.com/articles/12c/temporal-validity-12cr1.php

[24] DataDisk. 2014. **Undo Data.** [Online]. Available : http://www.datadisk.co.uk/html_docs/oracle/undo.htm

[25] Burleson, D. 2012. **Oracle Total Recall Tips.** [Online]. Available : http://www.dba-oracle.com/t_total_recall.htm

[26] Hesse, U. 2014. **"Total Recall": Brief introduction into Flashback Data Archive.** [Online]. Available : http://uhesse.com/2009/10/23/total-recall-brief-introduction-into-flashback-data-archive/

[27] Jernigan, K. September 2009. **Oracle Total Recall with Oracle Database 11g Release 2.** [Online]. Available : http://www.oracle.com/technetwork/database/focus-areas/storage/total-recall-whitepaper-171749.pdf

[28] Morgan, D. 2014. **Flashback Archive.** [Online]. Available : http://www.morganslibrary.org/reference/flash_archive.html

[29] Sandro, R. Elena, M. and Nenad, A. 2014. **Performance Evaluation of Temporal Feature Defined in Oracle 12c Database.** [Online]. Available : http://books.google.co.th/books?id=7Y3fAwAAQBAJ&pg=PA858&lpg=PA858&dq=performance+evaluation+of+temporal+feature+defined+in+oracle&แหล่งข้อมูล=bl&ots=XwXWpPqiq2&sig=Pt6udYtc_SuUAZMtkGDPQnDudaA&hl=th&sa=X&ei=KiRGVKa_BeG_mwXa94GQCQ&ved=0CCEQ6AEwAA#v=onepage&q=performance%20evaluation%20of%20temporal%20feature%20defined%20in%20oracle&f=false