การสร้างกรณีทดสอบการออกแบบคลังข้อมูล

นางสาวปิยภรณ์ สามสุวรรณ์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2558

GENERATION OF DATA WAREHOUSE DESIGN TEST CASES

Miss Piyaporn Samsuwan

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Computer Science
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2015

หัวข้อวิทยานิพนธ์	การสร้างกรณีทดสอบการออกแบบคลังข้อมูล
โดย	นางสาวปิยภรณ์ สามสุวรรณ์
สาขาวิชา	วิทยาศาสตร์คอมพิวเตอร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	รองศาสตราจารย์ ดร. ญาใจ ลิ่มปิยะกรณ์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

.....คณบดีคณะวิศวกรรมศาสตร์

(ศาสตราจารย์ ดร. บัณฑิต เอื้ออาภรณ์)

คณะกรรมการสอบวิทยานิพนธ์

.....ประธานกรรมการ

(ศาสตราจารย์ ดร. บุญเสริม กิจศิริกุล)

.....อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

(รองศาสตราจารย์ ดร. ญาใจ ลิ่มปิยะกรณ์)

.....กรรมการภายนอกมหาวิทยาลัย

(อาจารย์ ดร. ภาสกร อภิรักษ์วรพินิต)

ปิยภรณ์ สามสุวรรณ์ : การสร้างกรณีทดสอบการออกแบบคลังข้อมูล (GENERATION OF DATA WAREHOUSE DESIGN TEST CASES) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: รศ. ดร. ญาใจ ลิ่มปิยะกรณ์, 109 หน้า.

คลังข้อมูลสำหรับฐานข้อมูลเชิงสัมพันธ์ประกอบด้วยหลากหลายระยะวงจรชีวิต โดยทั่วไปการจำลองแบบในระยะการออกแบบมักให้ความสำคัญกับโครงสร้างทางความคิด โครงสร้างทางตรรกะ และโครงสร้างทางกายภาพ งานวิจัยนี้ได้นำเสนอแนวทางการสร้างกรณีทดสอบโครงสร้างแต่ละระดับข้างต้นด้วยวิธีการดึงข้อมูลจากโครงสร้างที่ได้จากการออกแบบซึ่งแสดงอยู่ในเอกสารการออกแบบ และสร้างคำสั่งในการทดสอบด้วยภาษาเอสคิวแอลที่สามารถแก้ไขคำผิดบิดเบือนด้วยวิธีระยะการแก้ไขน้อยสุด ระบบที่พัฒนาขึ้นคาดว่าจะสามารถทำให้คุณภาพข้อมูลเป้าหมายดีขึ้นและลดภาระการแก้งาน กล่าวคือ แนวทางที่นำเสนอจะลดจำนวนข้อบกพร่องที่เกิดขึ้นในระยะการออกแบบก่อนที่จะเข้าสู่ระยะอีทีแอล ส่งผลให้ภาระการแก้งานสำหรับการประมวลผลก่อนของข้อมูลเป้าหมายที่ระยะอีทีแอลลดน้อยลง

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....

สาขาวิชา วิทยาศาสตร์คอมพิวเตอร์ ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

ปีการศึกษา 2558

5671040721 : MAJOR COMPUTER SCIENCE
KEYWORDS: DATA WAREHOUSE / TEST CASE GENERATION / DESIGN TEST / SQL / EDIT DISTANCE

PIYAPORN SAMSUWAN: GENERATION OF DATA WAREHOUSE DESIGN TEST CASES. ADVISOR: ASSOC. PROF. YACHAI LIMPIYAKORN, Ph.D., 109 pp.

There are several data warehouse lifecycle for Relational databases. Typically, most design modeling concentrates on Conceptual schema, Logical schema, and Physical schema. This research presents an approach to generating test cases for data warehouse design tests, including the level of conceptual, logical, and physical. The test cases contain SQL statements for verifying some certain predefined aspects. The minimum edit distance component is also available for correcting the garbled terms found in the SQL. The implemented system would enhance the quality of target data as well as lessen the rework. In particular, the presented approach would reduce the number of defects injected in detailed designed prior to entering the ETL phase. This leads to less rework for preprocessing target data at the ETL phase.

Department: Computer Engineering Student's Signature .....
Field of Study: Computer Science Advisor's Signature .....
Academic Year: 2015

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยความอนุเคราะห์อย่างดียิ่งของ รองศาสตราจารย์ ดร.ญาใจ ลิ่มปิยะกรณ์ อาจารย์ที่ปรึกษาวิทยานิพนธ์ ซึ่งได้ให้ความรู้ คำแนะนำ ข้อคิดเห็นต่าง ๆ และช่วยเหลือในการแก้ไขข้อบกพร่องต่าง ๆ ด้วยความเอาใจใส่เพื่อให้การทำวิทยานิพนธ์ฉบับนี้สมบูรณ์ ผู้วิจัยมีความซาบซึ้งในความกรุณาอันดีจากอาจารย์ และกราบขอบพระคุณเป็นอย่างสูงไว้ ณ โอกาสนี้

ขอบพระคุณศาสตราจารย์ ดร.บุญเสริม กิจศิริกุล และอาจารย์ ดร.ภาสกร อภิรักษ์วรพินิต กรรมการสอบวิทยานิพนธ์ ที่กรุณาเสียสละเวลา ให้คำแนะนำ ช่วยตรวจสอบเพื่อแก้ไขวิทยานิพนธ์ฉบับนี้จนสมบูรณ์

ขอกราบขอบพระคุณ คุณพ่อ คุณแม่ และครอบครัว ที่ให้กำลังใจและการสนับสนุนช่วยเหลือในด้านต่าง ๆ กราบขอบพระคุณอาจารย์ทุกท่านที่ได้อบรมสั่งสอนวิชาความรู้ให้ผู้วิจัยจนสามารถทำวิทยานิพนธ์ได้สำเร็จลุล่วง

สุดท้ายนี้ผู้วิจัยขอขอบคุณเพื่อนร่วมงานและเพื่อน ๆ ทุกคนที่คอยให้กำลังใจและความเชื่อเหลือต่าง ๆ ตลอดจนผู้มีพระคุณทุกท่านที่มิได้กล่าวถึง คุณประโยชน์และคุณค่าอันเกิดจากการศึกษาวิจัยครั้งนี้ ผู้วิจัยขอมอบแด่บิดา มารดา ครู อาจารย์และผู้มีพระคุณทุกท่านด้วยความซาบซึ้งใจเป็นอย่างยิ่ง

สารบัญ

หน้า

บทคัดย่อภาษาไทย ..... ง

บทคัดย่อภาษาอังกฤษ ..... จ

กิตติกรรมประกาศ ..... ฉ

สารบัญ ..... ช

สารบัญตาราง ..... ฎ

สารบัญรูป ..... ฑ

บทที่ 1 บทนำ 1

1.1 ที่มาและความสำคัญของปัญหา 1

1.2 วัตถุประสงค์ของการวิจัย 2

1.3 ขอบเขตของการวิจัย 2

1.4 ขั้นตอนและวิธีดำเนินการวิจัย 2

1.5 ประโยชน์ที่คาดว่าจะได้รับ 2

1.6 ผลงานตีพิมพ์จากวิทยานิพนธ์ 3

1.7 ลำดับการจัดเรียงเนื้อหาในวิทยานิพนธ์ 3

บทที่ 2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง 4

2.1 ทฤษฎีที่เกี่ยวข้อง 4

2.1.1 แบบจำลองและการออกแบบคลังข้อมูล (Data Warehouse Modeling and Design)[1] 4

2.1.2 การวัดระยะการแก้ไข (Edit Distance)[3] 6

2.1.3 เอสคิวแอล (SQL: Structured Query Language)[4] 8

2.2 งานวิจัยที่เกี่ยวข้อง 9

2.2.1 Automated SQL query generation for systematic testing of database engines[5] 9

หน้า

2.2.2 A Comprehensive Approach to Data Warehouse Testing[6] 11
บทที่ 3 แนวคิดวิธีดำเนินการวิจัย 14
3.1 ภาพรวมของแนวทางการสร้างกรณีทดสอบการออกแบบคลังข้อมูล 14
3.1.1 Design Document 15
3.1.2 Extract Data 17
3.1.3 Data for test case generation 17
3.1.4 Generate test case 20
3.1.5 Test case with SQL 26
3.1.6 Format SQL statement 28
3.1.7 Test case with corrected form at SQL 29
3.1.8 Execute test case 29
3.1.9 Edit SQL 30
3.1.10 Compare result with expected result 32
3.1.11 Update actual result / test result 33
3.1.12 Report test result summary 34
บทที่ 4 การออกแบบและพัฒนาระบบ 36
4.1 ข้อกำหนดเบื้องต้นของระบบ 36
4.1.1 ผู้ใช้งาน (User) 36
4.1.2 ข้อมูลนำเข้า (Input) 36
4.1.3 ข้อมูลนำออก (Output) 36
4.1.4 ข้อจำกัดของระบบ (Constraint) 36
4.2 ความต้องการเชิงหน้าที่ (Functional Requirements) 37
4.3 การออกแบบระบบ 39

หน้า

4.3.1 แผนภาพยูสเคส 39
4.3.2 คำอธิบายยูสเคส 39
4.4 การพัฒนาระบบ 45
4.4.1 สภาพแวดล้อมและเครื่องมือที่ใช้ในการพัฒนาระบบ 45
4.4.2 การติดตั้งซอฟต์แวร์ที่ใช้สำหรับการพัฒนาระบบ 46
4.4.3 การพัฒนาส่วนต่อประสาน (User interface) 46
4.4.3.1 ส่วนของการนำเข้าไฟล์และแสดงผล 47
4.4.3.2 ส่วนของกรณีทดสอบ 47
4.4.3.3 ส่วนของรายการคำสั่งเอสคิวแอลทดสอบ 48
บทที่ 5 การทดสอบและประเมินผลระบบ 55
5.1 การทดสอบระบบ 55
5.1.1 การทดสอบการนำเข้าไฟล์เอกสารการออกแบบในรูปแบบเอกซ์เซล 55
5.1.2 ทดสอบการค้นหาเอกสารการออกแบบ 56
5.1.3 ทดสอบการสร้างกรณีทดสอบ 56
5.1.4 ทดสอบการส่งคำสั่งไปดำเนินการที่ฐานข้อมูล 56
5.1.5 ทดสอบการแก้ไขคำสั่งทดสอบ 56
5.1.6 ทดสอบการปรับปรุงค่า Actual result 56
5.1.7 ทดสอบเปรียบเทียบค่า Actual result และ Expected result ของกรณีทดสอบ 57
5.1.8 ทดสอบการแสดงผลการทดสอบกรณีทดสอบ 57
5.1.9 ทดสอบการแสดงรายละเอียดผลการทดสอบกรณีทดสอบ 57
5.2 การประเมินผลระบบ 71
5.2.1 ข้อมูลนำเข้า 71
5.2.2 ข้อมูลนำออก 81

5.2.3 สรุปการประเมินผลระบบ 84
บทที่ 6 สรุปผลการวิจัยและข้อเสนอแนะ 87
6.1 สรุปผลการวิจัย 87
6.2 ข้อจำกัด 87
6.3 แนวทางการวิจัยต่อ 87
รายการอ้างอิง 88
ภาคผนวก 89
ภาคผนวก ก การติดตั้งฐานข้อมูล 90
ภาคผนวก ข การติดตั้งโปรแกรมไพทอน 100
ภาคผนวก ค การติดตั้งกรอบงาน Django 106
ประวัติผู้เขียนวิทยานิพนธ์ 109

สารบัญตาราง

ตารางที่ 1 การคำนวณหาค่าระยะการแก้ไขระหว่างคำว่า INTENTION และ EXECUTION 7
ตารางที่ 2 รูปแบบการเขียนเอกสารการออกแบบส่วนข้อมูลตารางต้นทางในงานวิจัย 16
ตารางที่ 3 รูปแบบการเขียนเอกสารการออกแบบส่วนข้อมูลความสัมพันธ์ของตารางต้นทาง 16
ตารางที่ 4 รูปแบบการเขียนเอกสารการออกแบบส่วนข้อมูลความสัมพันธ์ของตารางต้นทาง 16
ตารางที่ 5 การสร้างกรณีทดสอบการออกแบบแต่ละระดับจากข้อมูลเอกสารการออกแบบ 20
ตารางที่ 6 การสร้างกรณีทดสอบจากเอกสารการออกแบบในระดับ conceptual schema 21
ตารางที่ 7 การสร้างกรณีทดสอบจากเอกสารการออกแบบในระดับ logical schema 22
ตารางที่ 8 การสร้างกรณีทดสอบจากเอกสารการออกแบบในระดับ physical schema 23
ตารางที่ 9 โครงสร้างข้อมูลสำหรับจัดเก็บกรณีทดสอบและคำสั่งทดสอบ 26
ตารางที่ 10 ความต้องการเชิงหน้าที่ 37
ตารางที่ 11 ค่ากำหนดในเอกสารการออกแบบส่วนของตารางต้นทาง (Source table relationship) 38
ตารางที่ 12 ค่ากำหนดในเอกสารการออกแบบส่วนของข้อมูลความสัมพันธ์ของตารางต้นทาง (Source table relationship) 38
ตารางที่ 13 ค่ากำหนดในเอกสารการออกแบบส่วนของข้อมูลตารางปลายทางที่สัมพันธ์กับตารางข้อมูลต้นทาง (Target table detail) 38
ตารางที่ 14 คำอธิบายยูสเคส Import Excel 40
ตารางที่ 15 คำอธิบายยูสเคส Search Excel 40
ตารางที่ 16 คำอธิบายยูสเคส Extract Excel design document 41
ตารางที่ 17 คำอธิบายยูสเคส Generate test cases และ SQL script 41
ตารางที่ 18 คำอธิบายยูสเคส Display test case และ test script 42
ตารางที่ 19 คำอธิบายยูสเคส Execute SQL script 42
ตารางที่ 20 คำอธิบายยูสเคส Edit SQL script 43
ตารางที่ 21 คำอธิบายยูสเคส Execute SQL script 44

ตารางที่ 22 คำอธิบายยูสเคส Display result script test on user interface 44
ตารางที่ 23 คำอธิบายยูสเคส Display test case result detail on user interface 45
ตารางที่ 24 ทดสอบการนำเข้าเอกสารการออกแบบ 57
ตารางที่ 25 ทดสอบตรวจสอบการนำเข้าเอกสารที่มีค่าว่างเปล่า 57
ตารางที่ 26 ทดสอบการนำเข้าจำนวนแผ่นเอกสารในเอกสารการออกแบบ 58
ตารางที่ 27 ทดสอบการนำเข้าแผ่นเอกสารส่วนข้อมูลตารางต้นทางเป็นค่าว่าง 58
ตารางที่ 28 ทดสอบการนำเข้าแผ่นเอกสารส่วนข้อมูลความสัมพันธ์ของตารางต้นทางเป็นค่าว่าง ... 59
ตารางที่ 29 ทดสอบการนำเข้าแผ่นเอกสารข้อมูลตารางปลายทางที่สัมพันธ์กับตารางข้อมูลต้นทางเป็นค่าว่าง 59
ตารางที่ 30 ทดสอบค่า table type ส่วนข้อมูลตารางต้นทางไม่เป็นไปตามกำหนด 60
ตารางที่ 31 ทดสอบค่า select method ส่วนข้อมูลตารางต้นทางไม่เป็นไปตามกำหนด 60
ตารางที่ 32 ทดสอบค่า join type ส่วนข้อมูลความสัมพันธ์ของตารางต้นทางไม่เป็นไปตามกำหนด 61
ตารางที่ 33 ทดสอบค่า data type ส่วนข้อมูลตารางปลายทางที่สัมพันธ์กับตารางข้อมูลต้นทางไม่เป็นไปตามกำหนด 61
ตารางที่ 34 ทดสอบการใช้คำค้นที่เป็นค่าว่าง 62
ตารางที่ 35 ทดสอบการใช้คำค้นที่เป็นตัวพิมพ์ใหญ่หรือตัวพิมพ์เล็กทั้งหมด 62
ตารางที่ 36 ทดสอบการใช้คำค้นที่ตรงกับชื่อเอกสารใด ๆ 63
ตารางที่ 37 ทดสอบการใช้คำค้นที่ไม่ตรงกับชื่อเอกสารใด ๆ 63
ตารางที่ 38 ทดสอบการสร้างคำสั่งทดสอบเพื่อทดสอบชนิดข้อมูล 63
ตารางที่ 39 ทดสอบการสร้างคำสั่งทดสอบแบบ Aggregate function 64
ตารางที่ 40 ทดสอบการสร้างคำสั่งทดสอบเมื่อตารางตั้งต้นมีความสัมพันธ์แบบ one to one 64
ตารางที่ 41 ทดสอบการสร้างคำสั่งทดสอบเมื่อตารางตั้งต้นมีความสัมพันธ์แบบ one to many 64
ตารางที่ 42 ทดสอบการสร้างกรณีทดสอบประเภท transform test แบบคำสั่ง case when 65
ตารางที่ 43 ทดสอบการสร้างกรณีทดสอบประเภท transform test แบบคำสั่ง if-else 65

ตารางที่ 44 ทดสอบการส่งคำสั่งทดสอบที่ไม่สามารถดำเนินการได้ไปยังฐานข้อมูล 66
ตารางที่ 45 ทดสอบการส่งคำสั่งทดสอบที่สามารถดำเนินการได้ไปยังฐานข้อมูล 66
ตารางที่ 46 ทดสอบการปรับปรุงค่า test script result เมื่อคำสั่งทดสอบไม่สามารถใช้ดำเนินการได้ 67
ตารางที่ 47 ทดสอบการปรับปรุงค่า test script result เมื่อคำสั่งทดสอบสามารถดำเนินการได้ 67
ตารางที่ 48 ทดสอบการแยกประเภทข้อความแสดงข้อผิดพลาดของชื่อตาราง 67
ตารางที่ 49 ทดสอบการแยกประเภทข้อความแสดงข้อผิดพลาดของชื่อคอลัมน์ 68
ตารางที่ 50 ทดสอบการแก้ไขคำสั่งทดสอบส่วนชื่อตาราง 68
ตารางที่ 51 ทดสอบการแก้ไขคำสั่งทดสอบส่วนชื่อคอลัมน์ 68
ตารางที่ 52 ทดสอบการปรับปรุงค่า คำสั่งทดสอบหลังการแก้ไขในตารางกรณีทดสอบ 68
ตารางที่ 53 ทดสอบการปรับปรุงค่า Actual result เมื่อมีข้อความแสดงข้อผิดพลาดของคำสั่งทดสอบ 69
ตารางที่ 54 ทดสอบการปรับปรุงค่า Actual result เมื่อเมื่อคำสั่งทดสอบสามารถดำเนินการได้ 69
ตารางที่ 55 ทดสอบกรณีค่า Actual result เป็นค่าว่างเปล่า 70
ตารางที่ 56 ทดสอบกรณีค่า Actual result เท่ากับค่า Expected result 70
ตารางที่ 57 ทดสอบกรณีค่า Actual result ไม่เท่ากับค่า Expected result 70
ตารางที่ 58 ทดสอบการแสดงผลการทดสอบกรณีทดสอบ 71
ตารางที่ 59 ทดสอบการแสดงรายละเอียดผลการทดสอบกรณีทดสอบ 71
ตารางที่ 60 ข้อมูลเอกสารการออกแบบส่วนตารางตั้งต้น 71
ตารางที่ 61 ข้อมูลเอกสารการออกแบบส่วนความสัมพันธ์ของตารางตั้งต้น 72
ตารางที่ 62 ส่วนความสัมพันธ์ของตารางปลายทางกับตารางตั้งต้น(ตารางปลายทาง) 73
ตารางที่ 63 ส่วนความสัมพันธ์ของตารางปลายทางกับตารางตั้งต้น(ตารางตั้งต้น) 75

สารบัญรูป

รูปที่ 1ขั้นตอนหลักของการออกแบบคลังข้อมูล[1] 4
รูปที่ 2 ตัวอย่างโครงสร้างทางความรู้เชิงแนวคิด[2] 5
รูปที่ 3 ตัวอย่างโครงสร้างทางความรู้เชิงตรรกะ[2] 5
รูปที่ 4 ตัวอย่างโครงสร้างทางความรู้เชิงกายภาพ[2] 6
รูปที่ 5 อัลกอริทึมการวัดระยะการแก้ไขเลเวนชเตยน์ 7
รูปที่ 6 ตัวอย่างคำสั่งเอสคิวแอลที่สร้างจากแนวทางที่นำเสนอ 10
รูปที่ 7 ข้อมูลสนับสนุนไวยากรณ์ในภาษาเอสคิวแอล 10
รูปที่ 8 ความสัมพันธ์ระหว่างข้อมูลส่วนทดสอบและวิธีการทดสอบ[6] 12
รูปที่ 9 ขั้นตอนการทำงานของระบบ 14
รูปที่ 10 ตัวอย่างของเอกสารการออกแบบที่เก็บใน Class model TEDWCF_SRCTABLE 18
รูปที่ 11 ตัวอย่างเอกสารการออกแบบที่ถูกนำมาจัดเก็บที่ Class model TEDWCF_RELATION .. 18
รูปที่ 12 ตัวอย่างเอกสารการออกแบบที่ถูกนำมาจัดเก็บที่ Class model TEDWCF_TARGET 20
รูปที่ 13หน้าจอแสดงผลกรณีทดสอบและคำสั่งทดสอบ 28
รูปที่ 14 อธิบายลักษณะการทำงานของ sqlparse python package 29
รูปที่ 15 หน้าจอแสดงกรณีทดสอบหลังคำสั่งทดสอบได้รับการจัดรูปแบบ 29
รูปที่ 16 หน้าจอแสดงผล test script result 30
รูปที่ 17 ลักษณะการทำงานของ python Levenshtein package 31
รูปที่ 18 หน้าจอแสดงคำสั่งทดสอบที่ได้รับการแก้ไข 31
รูปที่ 19 ตัวอย่างหน้าจอแสดงผล Actual result และ Test result 34
รูปที่ 20 หน้าจอแสดงผลสรุปการทดสอบกรณีทดสอบ 35
รูปที่ 21 แผนภาพยูสเคสของระบบสร้างกรณีทดสอบการออกแบบคลังข้อมูล 39
รูปที่ 22 ส่วนต่อประสานสำหรับนำเข้าเอกสารการออกแบบ 47
รูปที่ 23 ส่วนจัดเก็บเอกสารการออกแบบที่เว็บเซิร์ฟเวอร์ 47

รูปที่ 24 คำสั่งส่วนการจัดเก็บกรณีทดสอบ 48
รูปที่ 25 ตัวอย่างกรณีทดสอบและคำสั่งทดสอบในฐานข้อมูล 48
รูปที่ 26 ส่วนต่อประสานแสดงกรณีทดสอบและคำสั่งทดสอบ 49
รูปที่ 27 ตัวอย่างคำสั่งสร้างกรณีทดสอบในระดับ conceptual 51
รูปที่ 28 ตัวอย่างคำสั่งสร้างกรณีทดสอบในระดับ logical 53
รูปที่ 29 ตัวอย่างคำสั่งสร้างกรณีทดสอบในระดับ physical 54
รูปที่ 30 รูปแบบการทดสอบแบบกล่องดำ 55
รูปที่ 31 กรณีทดสอบ Fact test ในระดับ Conceptual design 81
รูปที่ 32 กรณีทดสอบ Conform test ในระดับ Conceptual design 82
รูปที่ 33 กรณีทดสอบ Join test ในระดับ Logical design 82
รูปที่ 34 กรณีทดสอบ Star test ในระดับ Logical design 82
รูปที่ 35 กรณีทดสอบ Constraints test ในระดับ Physical design 82
รูปที่ 36 กรณีทดสอบ Transform test ในระดับ Physical design 83
รูปที่ 37 คำสั่งทดสอบที่ได้รับการแก้ไข 83
รูปที่ 38 ผลการดำเนินการกรณีทดสอบ 84
รูปที่ 39 รายละเอียดผลการดำเนินการทดสอบ 84
รูปที่ 40 การดาวน์โหลดโปรแกรมติดตั้งฐานข้อมูล oracle 90
รูปที่ 41 โปรแกรมติดตั้งฐานข้อมูล Oracle 91
รูปที่ 42 หน้าจอ Dos เพื่อแจ้งว่าพร้อมสำหรับการติดตั้ง 91
รูปที่ 43 Welcome Screen 91
รูปที่ 44 หน้าจอแสดงการเตรียมพร้อมสำหรับการติดตั้งฐานข้อมูล 92
รูปที่ 45 หน้าจอแสดงผลสรุปของการติดตั้งฐานข้อมูล 93
รูปที่ 46 หน้าจอแสดงการความคืบหน้าติดตั้ง 93
รูปที่ 47 หน้าจอ configurations assistants 94

รูปที่ 48 หน้าจอแสดงความคืบหน้า Configuration assistants..... 94
รูปที่ 49 ข้อความแสดงการติดตั้งเสร็จสมบูรณ์..... 95
รูปที่ 50 หน้าจอ Password management..... 95
รูปที่ 51 หน้าจอสิ้นสุดการดำเนินการติดตั้งฐานข้อมูล..... 96
รูปที่ 52 ตัวอย่างการเขียน class model..... 96
รูปที่ 53 ผลจากการรันคำสั่ง Python manage.py migrate..... 97
รูปที่ 54 ผลจากการรันคำสั่ง Python manage.py makemigrations..... 97
รูปที่ 55 ผลจากการรันคำสั่ง Python manage.py migrate ครั้งที่ 2..... 98
รูปที่ 56 ผลการสร้างตารางที่ฐานข้อมูลผ่านทาง class model..... 98
รูปที่ 57 ตารางที่สร้างผ่าน class model..... 99
รูปที่ 58 รายละเอียดภายในตารางที่สร้างผ่าน class model..... 99
รูปที่ 59 เว็บไซต์สำหรับการดาวน์โหลดโปรแกรมติดตั้งไพทอน..... 100
รูปที่ 60 การ install ไพทอน..... 100
รูปที่ 61 หน้าจอเริ่มต้นการติดตั้งไพทอน..... 101
รูปที่ 62 หน้าจอสำหรับเลือก python directory path..... 101
รูปที่ 63 หน้าจอแสดงความคืบหน้าการติดตั้ง..... 102
รูปที่ 64 หน้าจอแสดงการติดตั้งโปรแกรมเสร็จสมบูรณ์..... 102
รูปที่ 65 หน้าจอแสดงผลการรัน Command Prompt..... 103
รูปที่ 66 หน้าจอแสดง My computer properties..... 103
รูปที่ 67 หน้าจอสำหรับตั้งค่า Advanced system setting..... 104
รูปที่ 68 หน้าจอสำหรับเพิ่ม Environment Variable..... 104
รูปที่ 69 หน้าจอ Command Prompt สำหรับการ install python package..... 105
รูปที่ 70 ตัวอย่างการติดตั้ง python Levenshtein package ด้วย pip..... 105
รูปที่ 71 หน้าจอ Command Prompt แสดง Django directory path..... 106

รูปที่ 72 หน้าจอ Command Prompt แสดงคำสั่งการติดตั้ง Django framework..... 106
รูปที่ 73 หน้าจอ Command Prompt แสดงผลจากคำสั่ง python..... 107
รูปที่ 74 หน้าจอ Command Prompt แสดงเวอร์ชันของ Django framework..... 107
รูปที่ 75 project mysite directory..... 108
รูปที่ 76 mysite website directory..... 108

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของปัญหา

ระหว่างขั้นตอนการสร้างข้อมูลเพื่อตอบสนองความต้องการทางธุรกิจสามารถทำได้ตามสถาปัตยกรรมคลังข้อมูลที่มีอยู่หลายรูปแบบซึ่งทุกรูปแบบประกอบด้วยชุดข้อมูลต้นทาง ขั้นตอนการทำอีทีแอล (ETL:Extract Transform load) , และขั้นตอนการแสดงข้อมูลที่ได้ด้วยแอปพลิเคชันแสดงผล โดยทั่วไปงานควบคุมคุณภาพของข้อมูลมักถูกกระทำไปพร้อมกับการสร้างข้อมูลเสมอ การออกแบบข้อมูลเป็นอีกส่วนหนึ่งที่สำคัญ ในการเขียนรายละเอียดของการแปลงข้อมูลจากต้นทางไปเป็นชุดข้อมูลที่ต้องการ ก่อนรายละเอียดนั้นจะถูกส่งต่อไปในช่วงอีทีแอล สำหรับการออกแบบเพื่อนำไปสู่การได้ข้อมูลที่ต้องการนั้นสามารถแบ่งได้ตามลักษณะความซับซ้อนของตัวแบบการออกแบบในระดับต่าง ๆ เริ่มจาก 1) ตัวแบบเชิงแนวคิด หรือ Conceptual Model เพื่อบอกข้อมูลความสัมพันธ์อย่างคร่าว ๆของชุดข้อมูลจากต้นทางว่าจะถูกแปลงมาเป็นข้อมูลที่ต้องการได้อย่างไร 2) ตัวแบบเชิงตรรกะ หรือ Logical Model บอกความสัมพันธ์ของตารางต้นทางที่มีความสัมพันธ์ เช่น กฎการรวมตาราง 3) ตัวแบบเชิงกายภาพ หรือ Physical Model เป็นตัวแบบที่มีความซับซ้อนมากที่สุดซึ่งบอกถึงชื่อตาราง ชื่อคอลัมน์ และชนิดของข้อมูลที่ถูกเก็บในคอลัมน์ต่าง ๆ ของข้อมูลที่ต้องการหลังจากพัฒนาเสร็จสิ้น จึงทำให้เห็นได้ว่าการออกแบบข้อมูลและเอกสารที่จัดส่งไปยังช่วงอีทีแอลเป็นสิ่งสำคัญอย่างยิ่งในการให้ได้มาซึ่งข้อมูลที่ถูกต้อง

อย่างไรก็ตาม ในขั้นตอนของการออกแบบข้อมูลเป้าหมายจะมีการทบทวนให้การออกแบบตรงตามความต้องการของผู้ใช้งานแล้วก็ตาม รายละเอียดการออกแบบต้องถูกนำไปเขียนลงเอกสารเพื่อส่งเอกสารให้อีทีแอลทำการพัฒนาข้อมูล ซึ่งในแต่ละองค์กรจะเลือกใช้เครื่องมือในการเขียนแตกต่างกันออกไป ในงานวิจัยนี้ผู้วิจัยสนใจการเขียนรายละเอียดการออกแบบในเอกสารประเภทไฟล์เอกซ์เซล(Excel) ซึ่งไม่สามารถช่วยตรวจสอบความถูกต้องในการเขียนรายละเอียดของการออกแบบข้อมูลได้ ทำให้อาจเกิดข้อบกพร่องและนำไปสู่ความเข้าในผิดของผู้พัฒนาข้อมูลในช่วงอีทีแอล

งานวิจัยนี้ได้ศึกษาความซับซ้อนของการออกแบบแต่ละระดับ เพื่อนำมาเชื่อมโยงกับข้อมูลในเอกสารการออกแบบเอกซ์เซล และเพื่อหารูปแบบการทดสอบความถูกต้องของการออกแบบทั้ง 3 ระดับ โดยคำสั่งในการทดสอบเป็นชุดคำสั่งภาษาเอสคิวแอล และนำการวัดระยะการแก้ไข (Edit Distance) มาประยุกต์ใช้เพื่อการเสนอค่าข้อมูลที่มีความเป็นไปได้มากที่สุดในกรณีที่พบความผิดพลาดในเอกสารการออกแบบหลังจากทดสอบและมีค่าความผิดพลาด

1.2 วัตถุประสงค์ของการวิจัย

เพื่อเสนอแนวทางการสร้างกรณีทดสอบเอกสารการออกแบบคลังข้อมูล และพัฒนาระบบต้นแบบเพื่อสนับสนุนแนวทางที่นำเสนอ

1.3 ขอบเขตของการวิจัย

1. งานวิจัยนี้ใช้ไฟล์เอกซ์เซลเป็นเครื่องมือในการเก็บรายละเอียดการออกแบบ
2. งานวิจัยนี้สร้างคำสั่งทดสอบด้วยภาษาเอสคิวแอล
3. งานวิจัยนี้ใช้แพ็กเกจในการสกัดไฟล์เอกซ์เซล, จัดรูปแบบคำสั่งเอสคิวแอลและการวัดระยะการแก้ไขในภาษาไพทอน
4. งานวิจัยนี้ทดลองโดยใช้ข้อมูลการออกแบบข้อมูลเป้าหมายซึ่งเป็นความต้องการจริงจากผู้ใช้งานในหน่วยงานคลังข้อมูล
5. กรณีทดสอบและผลการทดสอบจะแสดงในรูปแบบที่ผู้วิจัยกำหนดเท่านั้น
6. การประเมินผลการสร้างกรณีทดสอบเอกสารการออกแบบอย่างน้อย 100 กรณีทดสอบ

1.4 ขั้นตอนและวิธีดำเนินการวิจัย

1. ศึกษาและทำความเข้าใจปัญหาในขั้นตอนการออกแบบข้อมูลเป้าหมาย
2. หาแนวทางเพื่อแก้ไขปัญหาการทดสอบด้วยข้อมูลจากเอกสารการออกแบบ
3. ศึกษาและทำความเข้าใจทฤษฎีและงานวิจัยที่เกี่ยวข้อง
4. ศึกษาการการเขียนแอปพลิเคชันด้วยไพทอน
5. วิเคราะห์และกำหนดระเบียบวิธีวิจัย
6. ออกแบบ ตั้งสมมติฐาน ที่เกี่ยวข้องกับงานวิจัย
7. พัฒนาระบบ
8. ทดสอบและประเมินผลระบบที่พัฒนา
9. สรุปผลงานวิจัย และนำผลที่ได้ไปปรับปรุงระบบเพื่อให้ได้วัตถุประสงค์ที่กำหนด
10. ตีพิมพ์ผลงานทางวิชาการ
11. จัดทำวิทยานิพนธ์

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. ได้วิธีการและเครื่องมือสร้างกรณีทดสอบแบบอัตโนมัติจากเอกสารการออกแบบ ในการพัฒนาการสร้างคลังข้อมูล
2. ได้เครื่องมือในการบูรณาการกับงานด้านการสร้างคลังข้อมูลให้สะดวกรวดเร็วขึ้น

3. ได้วิธีการที่ช่วยลดความผิดพลาดอันจะเกิดจากช่วงการออกแบบที่จะถูกส่งต่อไปยังช่วงอีทีแอล

1.6 ผลงานตีพิมพ์จากวิทยานิพนธ์

ส่วนหนึ่งของวิทยานิพนธ์นี้ได้รับการตีพิมพ์ในรายงานสืบเนื่องจากการประชุมวิชาการระดับนานาชาติเรื่อง “Generation of Data warehouse test cases”, Piyaporn Samsuwan and Yachai Limpiyakorn, in Proceedings of 5th International Conference on IT Convergence and Security (ICITCS 2015), August 24-27, 2015 in Kuala Lumpur, Malaysia.

1.7 ลำดับการจัดเรียงเนื้อหาในวิทยานิพนธ์

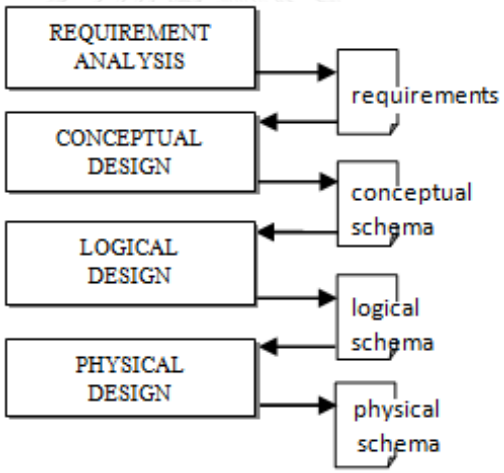
วิทยานิพนธ์นี้แบ่งเนื้อหาออกเป็น 6 บท ดังต่อไปนี้ บทที่ 1 เป็นบทนำกล่าวถึงที่มาและความสำคัญของปัญหา วัตถุประสงค์ ขอบเขตและประโยชน์ที่คาดว่าจะได้รับจากงานวิจัย บทที่ 2 กล่าวถึงทฤษฎีพื้นฐานและงานวิจัยที่เกี่ยวข้องกับงานวิจัย บทที่ 3 อธิบายถึงแนวทางในการสร้างกรณีทดสอบแบบบูรณาการจากแผนภาพส่วนประกอบ บทที่ 4 อธิบายถึงรายละเอียดการออกแบบของระบบต้นแบบ บทที่ 5 อธิบายถึงการประเมินผลระบบต้นแบบ และบทที่ 6 กล่าวถึงข้อสรุปของงานวิจัยและข้อเสนอแนะ

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

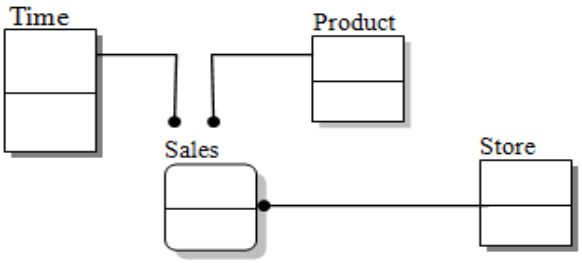
2.1.1 แบบจำลองและการออกแบบคลังข้อมูล (Data Warehouse Modeling and Design)[1]

การออกแบบคลังข้อมูลถูกนำเสนอด้วยหลายวิธีการโดยเริ่มตั้งแต่การวิเคราะห์ความต้องการเพื่อนำไปสู่ข้อสรุปความต้องการของผู้ใช้ การออกแบบเริ่มตั้งแต่การออกแบบในขั้นตอนการออกแบบเชิงแนวคิด (Conceptual Design) เพื่อให้ทราบว่าข้อมูลเป้าหมายจะสามารถตอบสนองความต้องการได้จริง ความซับซ้อนของการออกแบบ เช่นความสัมพันธ์ระหว่างตารางที่มาจากต้นทางเป็นอย่างไรอยู่ในขั้นตอนการออกแบบเชิงตรรกะ (Logical Design) และรายละเอียดของข้อมูลเป้าหมายว่านำมาจากตารางต้นทางไหนบ้างจะอยู่ในขั้นตอนการออกแบบทางกายภาพ (Physical Design) ซึ่งสามารถสรุปได้ตามรูปที่ 1



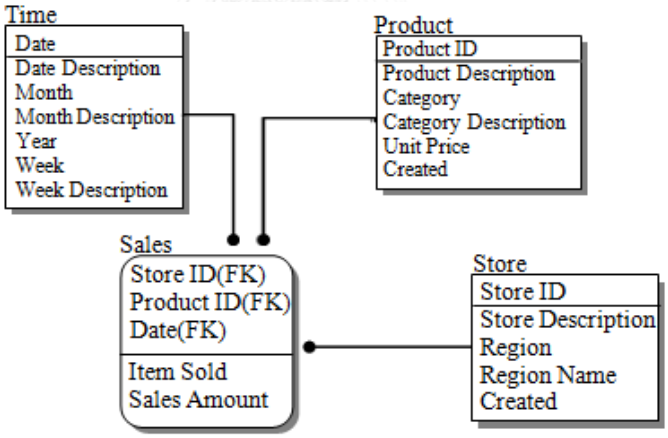
รูปที่ 1ขั้นตอนหลักของการออกแบบคลังข้อมูล[1]

การจำลองแบบเชิงแนวคิด (Conceptual Modeling)[2] การจำลองแบบเชิงแนวคิดเป็นการกำหนดสาระสำคัญระดับสูงเพื่อการอธิบายทุกลักษณะกระบวนการและโครงสร้างของคลังข้อมูลอันนำไปสู่การบรรลุผลตามที่ผู้ใช้งานต้องการ การออกแบบเชิงแนวคิดมีความจำเป็นในแง่เป็นการออกแบบขั้นพื้นฐานในการสร้างฐานข้อมูลซึ่งเป็นเอกสารข้อมูลที่เต็มไปด้วยความต้องการอันมีนัยสำคัญจากผู้ใช้ข้อมูลซึ่งโดยปกติการจะเป็นสัญกรณ์ (notation) กราฟิกซึ่งสะดวกต่อการเขียน การทำความเข้าใจและการจัดการโครงสร้างทางความรู้เชิงแนวคิด (conceptual schema) ของทั้งผู้ใช้ข้อมูลและผู้ออกแบบ ตัวอย่างโครงสร้างทางความรู้เชิงแนวคิดตามรูปที่ 2 แสดงให้เห็นถึงข้อมูลต้นทางที่ต้องนำเข้ามาเพื่อตอบสนองความต้องการของผู้ใช้ข้อมูลและแสดงให้เห็นว่าข้อมูลต้นทางใดสัมพันธ์ต่อกันบ้าง



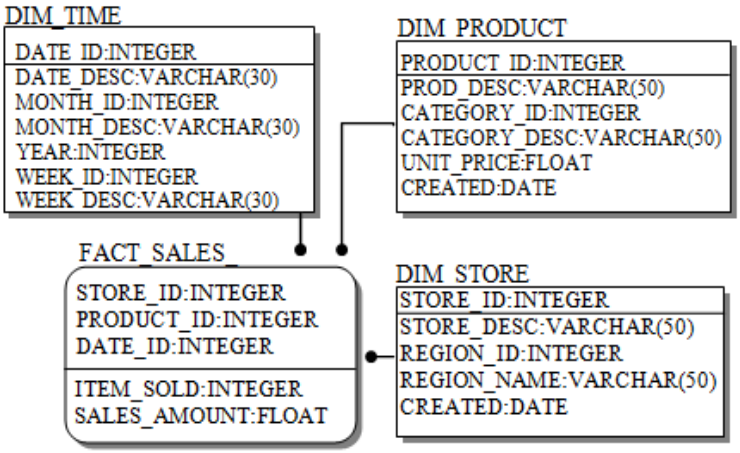
รูปที่ 2 ตัวอย่างโครงสร้างทางความรู้เชิงแนวคิด[2]

การจำลองแบบเชิงตรรกะ (Logical Modeling)[2] การจำลองแบบเชิงตรรกะมักถูกสร้างขึ้นหลังจากการจำลองแบบเชิงแนวคิดเสร็จสิ้น โดยงานทั้งหมดของการจำลองแบบเชิงตรรกะคือการแปลงจากโครงสร้างทางความรู้เชิงแนวคิดไปเป็นโครงสร้างทางความรู้เชิงตรรกะ (logical schema) สามารถทำให้ระบบเป้าหมายสร้างได้สำเร็จและใช้ได้อย่างเหมาะสม โครงสร้างทางความรู้เชิงตรรกะแสดงตามรูปที่ 3 ซึ่งนอกจากจะแสดงข้อมูลต้นทางยังแสดงถึงความสัมพันธ์ของข้อมูลต้นทางแต่ละที่ว่ามีความสัมพันธ์กันด้วยคอลัมน์อะไร ทำให้ทราบถึงรายละเอียดของข้อมูลต้นทางแต่ยังไม่บอกให้ทราบว่าสามารถดำเนินการให้บรรลุผลจริงๆ ได้อย่างไร



รูปที่ 3 ตัวอย่างโครงสร้างทางความรู้เชิงตรรกะ[2]

การจำลองแบบเชิงกายภาพ (Physical Modeling)[2] การจำลองแบบเชิงกายภาพเพื่อแสดงความซับซ้อนที่เพิ่มมากขึ้นของการออกแบบข้อมูลเป้าหมายมาแสดงผ่านโครงสร้างทางความรู้เชิงกายภาพ (physical schema) ซึ่งทำให้ทราบว่าจะทำให้บรรลุผลสำเร็จจริงได้อย่างไร เนื่องจากจะแสดงให้ทราบถึง ชื่อตารางจริง แม้แต่ที่อยู่ของข้อมูลที่จะนำเข้ามาแปลงเป็นคอลัมน์เป้าหมายว่าคอลัมน์ใดจากข้อมูลต้นทางจะมาเป็นคอลัมน์ใดในข้อมูลเป้าหมายโดยตัวอย่างแสดงตามรูปที่ 4



รูปที่ 4 ตัวอย่างโครงสร้างทางความรู้เชิงกายภาพ[2]

2.1.2 การวัดระยะการแก้ไข (Edit Distance)[3]

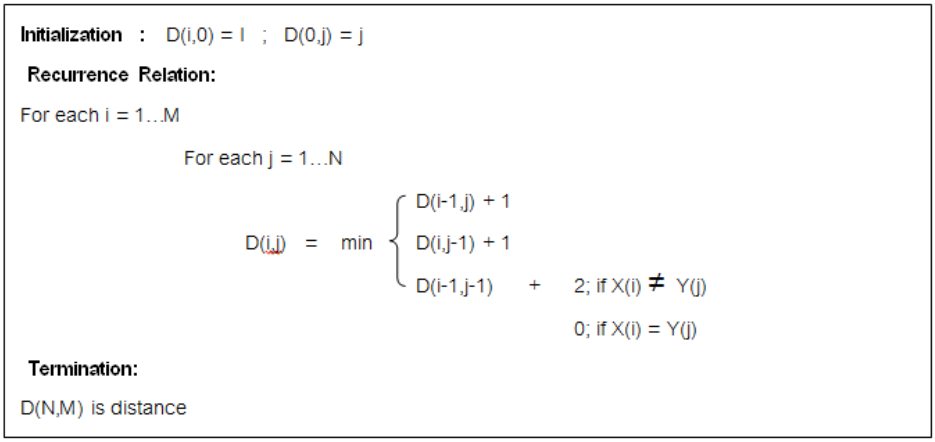
การวัดระยะการแก้ไขเป็นวิธีการในการวัดหาค่าความต่างกันระหว่างชุดตัวอักษรสองชุด ระหว่างชุดแรกที่เป็นชุดต้นแบบ และชุดที่สองที่เป็นชุดเปรียบเทียบ โดยค่าความแตกต่างจะวัดจากจำนวนครั้งของการที่จะต้องทำการแทรก ตัดออก และแทนที่ตัวอักษรในชุดตัวอักษรที่นำมาเปรียบเทียบจนกระทั่งชุดตัวอักษรที่นำมาเปรียบเทียบ (ชุดที่สอง) มีลักษณะเหมือนกับตัวอักษรชุดต้นแบบ (ชุดแรก) ทุกประการ ซึ่งโดยทั่วไปการวัดระยะการแก้ไขจะอ้างอิงถึงระยะทางเลเวนชเตยน์ (Levenshtein Distance) ซึ่งเป็นทฤษฎีที่เกี่ยวข้องกับการวัดระยะการแก้ไขที่ถูกตั้งชื่อตาม Vladimir Levenshtein ผู้ที่ปรับปรุงชุดอัลกอริทึมชุดนี้

ระยะทางเลเวนชเตยน์ ขั้นตอนวิธีการ Levenshtein Distance จะเป็นการนำชุดตัวอักษร 2 ชุด มาเปรียบเทียบจำนวนความแตกต่าง โดยจะพิจารณาจาก 3 รูปแบบ คือ

การแทรก : เป็นการนำเอาตัวอักษรตัวใดๆ มา เพื่อให้ชุดตัวอักษรชุดนั้นเหมือนกับอีกชุดตัวอักษรหนึ่งในภายหลัง

การตัดออก : เป็นการตัดตัวอักษรออกครั้งละ 1 ตัวจากชุดตัวอักษรชุดหนึ่งเพื่อให้ชุดตัวอักษรชุดนั้นเหมือนกับอีกชุดตัวอักษรหนึ่งในภายหลัง

การแทนที่ : เป็นการนำตัวอักษรของชุดตัวอักษรหนึ่งไปแทนตัวอักษรของอีกชุดตัวอักษรหนึ่งเพื่อให้ชุดตัวอักษรชุดนั้นเหมือนกับอีกชุดตัวอักษรหนึ่งในภายหลัง



รูปที่ 5 อัลกอริทึมการวัดระยะการแก้ไขเลเวนชเตยน์

รูปที่ 5 แสดงถึงขั้นตอนวิธี (Algorithm) ของการวัดระยะการแก้ไขเลเวนชเตยน์ เพื่อหาค่าการดำเนินการที่น้อยที่สุดของการเปลี่ยนคำหนึ่งคำไปเป็นคำอื่น โดยหากกำหนดว่าความยาวของอักษรคำแรกเป็น N และความยาวของอักษรอีกคำเป็น M จะคำนวณหาระยะการแก้ไขเลเวนชเตยน์ D(N,M) ได้โดยแนวทางการคำนวณเป็นแบบกำหนดการพลวัต (Dynamic Programming) ที่ช่วยหาค่าย่อย D(i,j) ซึ่งเป็นค่าการดำเนินการที่น้อยที่สุดที่คำนวณมาจากแต่ละตำแหน่งของคำเพื่อนำมารวมกันเป็นค่าระยะการแก้ไขD (N,M) ตัวอย่างการคำนวณค่า D(N,M) จาก 2 คำคือคำว่า INTENTION และคำว่า EXECUTION แสดงในตารางที่ 1

ตารางที่ 1 การคำนวณหาค่าระยะการแก้ไขระหว่างคำว่า INTENTION และ EXECUTION

N	9	8	9	10	11	12	11	10	9	8
O	8	7	8	9	10	11	10	9	8	9
I	7	6	7	8	9	10	9	8	9	10
T	6	5	6	7	8	9	8	9	10	11
N	5	4	5	6	7	8	9	10	11	10
E	4	3	4	5	6	7	8	9	10	9
T	3	4	5	6	7	8	7	8	9	8
N	2	3	4	5	6	7	8	7	8	7
I	1	2	3	4	5	6	7	6	7	8
#	0	1	2	3	4	5	7	7	8	9
	#	**E**	**X**	**E**	**C**	**U**	**T**	**I**	**O**	**N**

2.1.3 เอสคิวแอล (SQL: Structured Query Language)[4]

เอสคิวแอลเป็นภาษาสำหรับการเขียนโปรแกรมเพื่อจัดการกับข้อมูล หรือสำหรับการดำเนินการเกี่ยวกับกระแสข้อมูลภายในที่อยู่ภายระบบจัดการฐานข้อมูลเชิงสัมพันธ์ (Relational Database Management System : RDBMS) โดยเอสคิวแอลเริ่มมีการพัฒนาที่ IBM โดย Donald D. Chamberlin และ Raymond F. Boyce ตั้งแต่ต้นปี ค.ศ. 1970

การสืบค้นข้อมูลด้วยเอสคิวแอล คิวรี (SQL Queries) การสืบค้นประกอบด้วยรายชื่อคอลัมน์ (column) ซึ่งจะไปเป็นผลลัพธ์สุดท้าย เริ่มต้นด้วยคำสำคัญ ‘SELECT’ และเครื่องหมายดอกจัน (*) สามารถใช้เพื่อให้คืนผลลัพธ์จากทุกคอลัมน์ในตารางที่สืบค้น ‘SELECT’ จะกลายเป็นคำสั่งที่ซับซ้อนมากขึ้นเมื่อมีคำสำคัญอื่น ๆ เพิ่มเข้ามาประกอบด้วย

ข้อความ ‘FROM’ (FROM clause) ซึ่งเป็นตัวชี้ไปยังตารางต่าง ๆ ที่รับข้อมูลมา ‘FROM’ ยังสามารถมีข้อความย่อย ‘JOIN’ (JOIN subs clauses) เพื่อระบุกฎการรวมตารางต่าง ๆเข้าด้วยกัน

ข้อความ ‘FROM’ (FROM clause) ซึ่งเป็นตัวชี้ไปยังตารางต่าง ๆ ที่รับข้อมูลมา ‘FROM’ ยังสามารถมีข้อความย่อย ‘JOIN’ (JOIN subs clauses) เพื่อระบุกฎการรวมตารางต่าง ๆเข้าด้วยกัน

ข้อความ ‘WHERE’ (WHERE clause) แสดงการเปรียบเทียบ ซึ่งช่วยจำกัดแถวที่แสดงเป็นผลลัพธ์ ข้อความ ‘WHERE ’ จะทำให้ผลลัพธ์ที่กลับคืนมาเป็นจำนวนแถวทั้งหมดหากเงื่อนไขที่กำหนดไม่เป็นจริง

ข้อความ ‘GROUP BY’ (GROUP BY clause) ใช้สำหรับจัดกลุ่มของแต่ละแถวที่เหมือนกัน ‘GROUP BY’ มักใช้เมื่อต้องการผลลัพธ์ในลักษณะฟังก์ชันการรวมกลุ่ม (aggregation function)

ข้อความ ‘HAVEING’ (HAVING clause) ใช้สำหรับกรองผลลัพธ์จากการใช้ข้อความ ‘GROUP BY’

ข้อความ ‘ORDER BY’ (ORDER BY clause) เพื่อเรียงลำดับผลลัพธ์ของข้อมูลจากคอลัมน์ที่กำหนด

ฟังก์ชันการรวม (Aggregate Function)

นอกจากภาษาเอสคิวแอลจะสามารถเขียนเพื่อดึงข้อมูลแบบแสดงทุกแถวข้อมูลได้แล้วยังสามารถเลือกแสดงแบบผลรวมของบางคอลัมน์ที่ผู้ใช้งานสนใจ โดยใช้คำสั่งจากฟังก์ชันการรวมที่มีให้เลือกใช้ตามมาตรฐานดังนี้

ข้อความ COUNT ฟังก์ชันการนับจำนวนแถวของข้อมูลภายใต้เงื่อนไข

ข้อความ SUM ฟังก์ชันหาผลรวมจากคอลัมน์ที่เลือกหาผลรวม

ข้อความ AVG ฟังก์ชันหาค่าเฉลี่ยจากคอลัมน์ที่เลือกหาผลรวม

ข้อความ MIN เป็นฟังก์ชันหาค่าต่ำสุดจากคอลัมน์ที่เลือกหาผลรวม

ข้อความ MAX เป็นฟังก์ชันหาค่าสูงสุดจากคอลัมน์ที่เลือกหาผลรวม

โดยมีข้อความ GROUP BY ใช้ในการจัดกลุ่มเพื่อหาผลรวมของคอลัมน์ในกลุ่มที่ทำการ GROUP BY หากไม่ใช้คำสั่ง GROUP BY จะเป็นการหาผลรวมของข้อมูลทั้งหมดที่มีในฐานข้อมูล

2.2 งานวิจัยที่เกี่ยวข้อง

2.2.1 Automated SQL query generation for systematic testing of database engines[5]

ในงานวิจัยนี้นำเสนอการสร้างประโยคที่ให้ความหมายถูกต้อง เป็นคำสั่งเอสคิวแอลในการใช้ทดสอบระบบฐานข้อมูลเชิงสัมพันธ์ เพื่อตรวจหาความผิดพลาดในฐานข้อมูล เนื่องจากในการทดสอบซอฟแวร์ย่อมต้องมีต้นทุน และยังมีการทดสอบข้อมูลขาเข้าที่มาจากฐานข้อมูล ดังนั้นการทดสอบแบบอัตโนมัติ จะช่วยลดต้นทุนในการทดสอบซอฟแวร์ได้ ในงานวิจัยนี้ประกอบด้วย 3 ขั้นตอนหลักในการทดสอบระบบจัดการฐานข้อมูล (DBMS) คือ 1) การสร้างคำสั่งที่ชี้ไปยังโครงสร้างทางความรู้ฐานข้อมูล (Database schema) 2) สร้างเซตของฐานข้อมูลทดสอบซึ่งก็คือตารางต่าง ๆ และ 3) ทำการดำเนินการกับคำสั่งเพื่อพิสูจน์ผลลัพธ์จากการดำเนินการจากข้อมูลจากข้อมูลเข้าที่ใช้กับระบบการจัดการฐานข้อมูลที่ใช้กับแอปพลิเคชัน การสร้างคำสั่งเอสคิวแอลนั่นเริ่มต้นจากการที่มีตารางที่มีความสัมพันธ์กันอยู่ในระบบการจัดการฐานข้อมูล แล้วใช้ข้อมูลดังกล่าวมาใช้เพื่อสร้างเป็นคำสั่งเอสคิวแอลในรูปแบบต่าง ๆ ที่สามารถเป็นไปได้เพื่อใช้ในการทดสอบ โดยในรูปที่ 6 เป็นการแสดงตัวอย่างขอคำสั่งเอสคิวแอลที่ถูกสร้างขึ้นตามแนวทางที่นำเสนอเพื่อใช้ในการทดสอบแอปพลิเคชัน คำสั่งเอสคิวแอลที่ถูกสร้างขึ้นมาจะพิจารณาจากข้อมูลที่ได้จากโครงสร้างเชิงความรู้ฐานข้อมูล เช่น รายละเอียดต่าง ๆ ดังนี้

```
QUERY ::= SELECT FROM
SELECT ::= 'SELECT' selectTerm+
FROM ::= 'FROM' (table | table JOIN table)
selectTerm ::= term | agg(term)
table ::= 'students' | 'grades'
term ::= 'id'|'name'|'studentID'|'courseID'|'grade'
agg ::= 'MAX' | 'MIN'
```

```
SELECT courseID, studetnID FROM GRADES, STUDENT;
SELECT MAX (courseID), MAX (NAME) FROM GRADES, STUDENT;
SELECT MIN (courseID), MIN (NAME) FROM GRADES, STUDENT;
SELECT courseID FROM GRADES, STUDENT;
SELECT MAX (courseID), MIN (NAME) FROM GRADES, STUDENT;
SELECT MAX (NAME), MIN (courseID) FROM GRADES, STUDENT;
SELECT courseID, MIN (NAME) FROM GRADES, STUDENT;
SELECT courseID, MAX (NAME) FROM GRADES, STUDENT;
SELECT NAME, MAX (courseID) FROM GRADES, STUDENT;
SELECT NAME FROM STUDENT;
SELECT MIN (NAME) FROM STUDENT;
SELECT id FROM STUDENT;
SELECT MAX (NAME), MIN (id) FROM STUDENT;
SELECT MAX (id), MIN (NAME) FROM STUDENT;
SELECT id, MAX (NAME) FROM STUDENT;
...
```

รูปที่ 6 ตัวอย่างคำสั่งเอสคิวแอลที่สร้างจากแนวทางที่นำเสนอ

จากการสร้างคำสั่งเอสคิวแอลในงานวิจัยนี้ได้นำแบบจำลองอัลลอย (Alloy Model) มาปรับใช้เพื่อช่วยแก้ไขเรื่องไวยากรณ์ของคำสั่งเพื่อแปลงไปเป็นคำสั่งเอสคิวแอลที่สมบูรณ์ โดยจากรูปที่ 6 จะมีการสร้างคำสั่งที่เป็นไปได้จากข้อมูลที่ได้รับจากโครงสร้างเชิงความรู้ฐานข้อมูลออกมาถึง 186 กรณี จากรูปที่ 7 เป็นข้อมูลที่สนับสนุนด้านไวยากรณ์ของภาษาเอสคิวแอลที่ใช้ในงานวิจัย

```
QUERY ::= SELECT FROM WHERE GROUP_BY HAVING
SELECT ::= 'SELECT' selectTerm+
selectTerm :: term | aggregate(term)
FROM ::= 'FROM' (table | table JOIN table)
WHERE ::= 'WHERE' term operator (term | value)
GROUP_BY ::= 'GROUP BY' term
HAVING ::= 'HAVING' term operator value
aggregate ::= 'MAX' | 'MIN' | 'AVG' | 'COUNT'
operator ::= '<' | '<=' | '>' | '>=' | '='
```

รูปที่ 7 ข้อมูลสนับสนุนไวยากรณ์ในภาษาเอสคิวแอล

ในงานวิจัยนี้อ่านข้อมูลจากโครงสร้างเชิงความรู้ฐานข้อมูลและสร้างเงื่อนไขข้อบังคับสำหรับตารางและคอลัมน์โดยอัตโนมัติขึ้นเป็นอันดับแรก ผู้วิจัยสร้างชื่อคอลัมน์และชื่อตารางด้วยส่วนประกอบที่ปรากฏอยู่ทั้งหมดในโครงสร้างเชิงความรู้ฐานข้อมูล และใช้คำสำคัญ (Keyword) เพื่อเติมเต็มจนเกิดเป็นคำสั่งเอสคิวแอล

ข้อสรุปของแนวทางที่นำเสนอสำหรับการสร้างคำสั่งเอสคิวแอลเพื่อทดสอบระบบการจัดการฐานข้อมูลแบบอัตโนมัติ ในงานวิจัยได้สร้างคำสั่งเอสคิวแอลผูกเป็นประโยคที่มีความหมายถูกต้องสมบูรณ์ เมื่อรวมกับงานก่อนหน้าที่นำเสนอคือ ADUSA ทำให้สามารถ 1) ทดสอบคำสั่งเอสคิวแอล 2) ล่วงรู้ข้อมูลเข้าของการสืบค้นในฐานข้อมูล และ 3) ทดสอบออราเคิลเพื่อยืนยันพิสูจน์ผลลัพธ์ที่ได้จากการดำเนินการ งานวิจัยนี้ใช้ชุดเครื่องมืออัลลอย (Alloy tools set) และเพิ่มส่วนของการทำให้แน่ใจเรื่องความผิดพลาดเชิงความหมายของคำสั่งและใช้ตัววิเคราะห์อัลลอย (Alloy analyzer) เพื่อสร้างคำสั่งเอสคิวแอลทดสอบที่สามารถเป็นไปได้ และได้เปรียบเทียบผลลัพธ์จากงานวิจัยซึ่งใช้เซตย่อยที่แตกต่างกันของไวยากรณ์ในภาษาเอสคิวแอลรวมกับงานที่ได้วิจัยก่อนหน้า กรอบงาน (framework) ในงานวิจัยช่วยให้ตรวจพบข้อผิดพลาดใหม่ ๆและช่วยลดข้อผิดพลาดในระบบฐานข้อมูลที่แตกต่างออกไป

2.2.2 A Comprehensive Approach to Data Warehouse Testing[6]

ในงานวิจัย A Comprehensive Approach to Data Warehouse Testing พูดถึงกิจกรรมการทดสอบความถูกต้องในกระบวนการการสร้างคลังข้อมูลโดยสรุปข้อมูลซึ่งเกี่ยวข้องกับระยะของการทดสอบในระดับต่าง ๆ และกล่าวถึงวิธีการทดสอบว่าสามารถทำด้วยวิธีการอย่างไร โดยกล่าวถึงการทดสอบที่สัมพันธ์กับแต่ละช่วงของการออกแบบคลังข้อมูล ซึ่งสามารถแบ่งออกได้เป็น 8 ช่วงดังนี้

- ช่วงการวิเคราะห์ความต้องการ (Requirements analysis)
- ช่วงวิเคราะห์และทำให้ลงรอย (Analysis and reconciliation)
- ช่วงการออกแบบเชิงแนวคิด (Conceptual design)
- ช่วงการแบ่งภาระงานเบื้องต้น (workload refinement)
- ช่วงการออกแบบเชิงตรรกะ (Logical design)
- ช่วงการออกแบบข้อมูลรอพัก (Data Staging design)
- ช่วงการออกแบบเชิงกายภาพ (Physical design)
- ช่วงการนำไปปฏิบัติ (Implementation)

การทดสอบคุณภาพของการออกแบบข้อมูลสามารถทำได้โดยการตรวจสอบความต้องการของผู้ใช้งานข้อมูลซึ่งแทนด้วยโครงสร้างทางความรู้เชิงแนวคิดของคลังข้อมูล แนวคิดในการออกแบบและส่วนของโครงสร้างทางความรู้เชิงตรรกะ โดยรวมส่วนที่สามารถนำมาทำการทดสอบได้มีดังนี้

- Conceptual schema

- Logical schema
- ETL procedures
- Database
- Front-end

จากส่วนทดสอบที่สามารถนำมาใช้ในการทดสอบข้างต้น และประสบการณ์ของผู้วิจัยสรุปได้ว่าคุณลักษณะการทดสอบที่เหมาะสมกับส่วนของข้อมูลทั้ง 5 คือ

- Functional test คือการทดสอบส่วนทดสอบว่าตรงความต้องการทางธุรกิจ
- Usability test คือการประเมินผลคลังข้อมูลโดยยินยอมให้ผู้ใช้งานประเมินผลในรูปแบบที่สามารถโต้ตอบกับส่วนทดสอบ
- Performance test คือการตรวจสอบสมรรถนะที่สำคัญภายใต้เงื่อนไขตามภาระงาน
- Stress test คือการทดสอบสมรรถนะโดยนำเข้าข้อมูลและภาระงานจำนวนมาก
- Recovery test คือการตรวจสอบส่วนทดสอบที่ผิดหลังจากได้รับการแก้ไข
- Security test คือการตรวจสอบการป้องกันข้อมูล
- Regression test คือการตรวจสอบว่าส่วนทดสอบยังคงถูกต้องหลังจากแก้ไขความผิดพลาดของส่วนทดสอบอื่น

	Conceptual schema	Logical schema	ETL procedures	Database	Front-end
Functional	✓	✓	✓		✓
Usability	✓	✓			✓
Performance		✓	✓	✓	✓
Stress			✓	✓	✓
Recovery			✓	✓	
Security			✓	✓	✓
Regression	✓	✓	✓	✓	✓
	Analysis & design		Implementation		

รูปที่ 8 ความสัมพันธ์ระหว่างข้อมูลส่วนทดสอบและวิธีการทดสอบ[6]

ซึ่งส่วนทดสอบทั้ง 5 ส่วนและการทดสอบแต่ละแบบสามารถแสดงดังรูปที่ 8 ซึ่งอธิบายถึงการทดสอบ Conceptual schema สามารถกระทำได้โดย Functional test, Usability test และ Regression test และการทดสอบ Logical schema สามารถทำได้โดย Functional test, Usability test, Performance test และ Regression test การทดสอบ ETL procedures

สามารถใช้การทดสอบแบบ Functional test, Performance test, Stress test, Recovery test, Security test และ Regression test ในส่วนของ Database สามารถใช้รูปแบบการทดสอบแบบ Performance test, Stress test, Recovery test, Security test และ Regression test และในส่วนของ Front-end ก็สามารถทดสอบด้วยวิธีการทดสอบแบบทุกรูปแบบยกเว้น Recovery test

ในการทดสอบ Conceptual schema ส่วนของ Functional test วิธีการย่อยในการทดสอบเรียกว่า Fact test ซึ่งสามารถใช้ในการตรวจสอบภาระงานที่เกิดขึ้นจากการวิเคราะห์ความต้องการของผู้ใช้งานที่สามารถแทนด้วย Conceptual schema โดยสามารถตรวจสอบโดยง่ายสำหรับแต่ละภาระงานโดยการใช้ ส่วนของ measurement ที่อยู่ใน fact table เพื่อทำ aggregation การทดสอบส่วนที่ 2 เรียกว่า Conformity test ซึ่งสามารถอธิบายโดย Bus matrix ซึ่งเป็นความสัมพันธ์ของ dimension ของ fact table ค่าความหนาแน่นที่แสดงผ่านทาง Bus matrix สามารถแสดงให้เห็นถึงความสอดคล้องของการใช้งานข้อมูลต้นทางอย่างสมเหตุสมผล โดยหาก Bus matrix มีความหนาแน่นมากแปลได้ว่าผู้ออกแบบ Conceptual schema ลืมพิจารณา dimension ที่ซ้ำซ้อน ทำให้ต้องใช้ fact table ที่ไม่จำเป็นในการนำมาสร้างคลังข้อมูล

ในการทดสอบ Logical schema ในส่วนของ Functional test การใช้คำสั่งสืบค้นข้อมูลอย่างง่ายในการทำ preliminary workload ด้วยภาษาเอสคิวแอลบน logical schema เรียกการทดสอบแบบนี้ว่า Star test การสร้างคำสั่งเช่น การสร้างคำสั่งแบบ Aggregation ขึ้นโดยผูกความสัมพันธ์ระหว่าง dimension attribute

ผู้วิจัยได้สรุปบทเรียนหลักที่ผู้วิจัยได้เรียนรู้เกี่ยวกับการทดสอบดังนี้

1.) การเปลี่ยนแปลงเพื่อสร้างประสิทธิภาพของงานทดสอบขึ้นอยู่กับความสมบูรณ์ของการเตรียมเอกสาร และความแม่นยำในส่วนของการรับความต้องการที่ถูกต้องและคำอธิบายเกี่ยวกับงานที่จะต้องทำการทดสอบ หากเราไม่ทราบว่าความต้องการคืออะไร เราก็ไม่สามารถที่จะคาดหวังความถูกต้องได้

2.) ช่วงของการทดสอบเป็นส่วนหนึ่งของกระบวนการสร้างคลังข้อมูลถือเป็นส่วนเสริมกันกับส่วนของการออกแบบ ดังนั้นส่วนของการทดสอบควรมีการวางแผนตั้งแต่ช่วงเริ่มต้นของการทำโครงการ ซึ่งสามารถตั้งเป้าหมายของการทดสอบ รูปแบบการทดสอบที่จะนำมาใช้ หรือ ข้อมูลอะไรที่ต้องการนำมาทดสอบ รวมทั้งค่าความคาดหวังของการทดสอบ

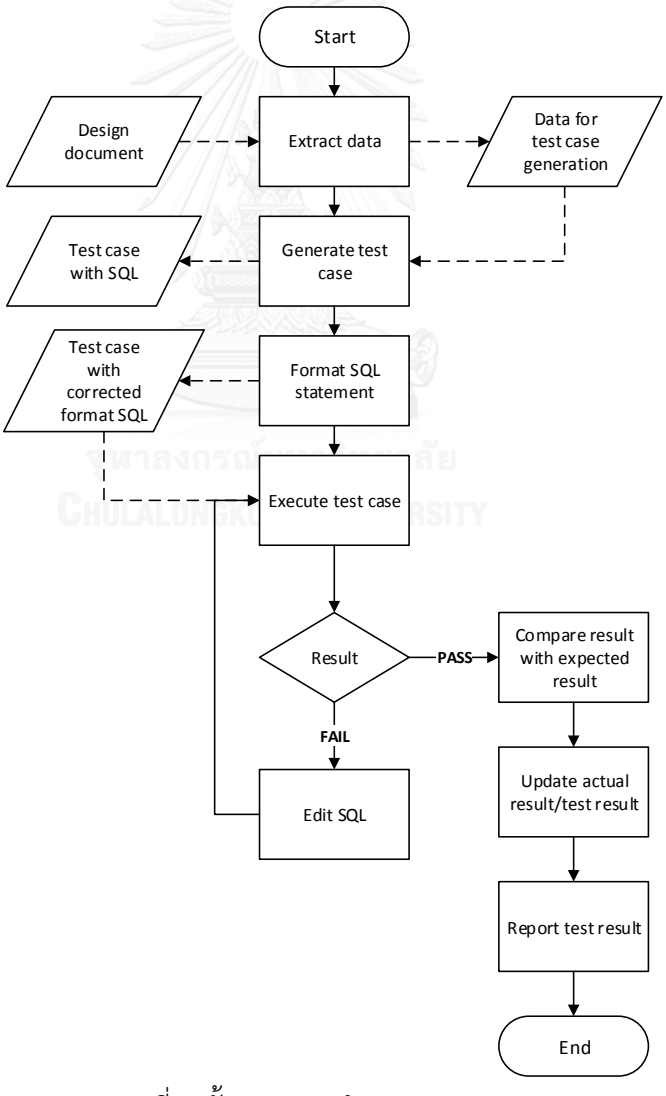
3.) การทดสอบไม่ใช่การทำงานของคนเพียงคนเดียว แต่การทดสอบควรทำร่วมกันเป็นทีมทั้ง ผู้ทดสอบ ผู้พัฒนา ผู้ออกแบบ หรือแม้แต่ผู้ดูแลฐานข้อมูล

4.) การทดสอบของระบบคลังข้อมูลอยู่บนพื้นฐานของข้อมูลที่มีขนาดใหญ่ ดังนั้นการประสบความสำเร็จในการทดสอบจึงขึ้นอยู่กับข้อมูลที่นำมาทดสอบเป็นข้อมูลจริงมากน้อยเพียงใดด้วย

บทที่ 3
แนวคิดวิธีดำเนินการวิจัย

3.1 ภาพรวมของแนวทางการสร้างกรณีทดสอบการออกแบบคลังข้อมูล

งานวิจัยนี้นำเสนอแนวทางการสร้างกรณีทดสอบการออกแบบคลังข้อมูลซึ่งมีเอกสารการออกแบบเป็นข้อมูลนำเข้าที่ใช้ในการทดสอบ การพัฒนาระบบเพื่อให้สามารถทดสอบเอกสารการออกแบบดังกล่าวต้องประกอบด้วยขั้นตอนหลักต่าง ๆ ดังแสดงในรูปที่ 9 สำหรับการสร้างคลังข้อมูลใหม่ในหน่วยงาน Data warehouse ในช่วงของการออกแบบจะมีเอกสารที่ต้องเขียนเพื่อแสดงรายละเอียดของการออกแบบ ส่งต่อให้ช่วงอีทีแอลเพื่อทำความเข้าใจและพัฒนางาน งานวิจัยนี้จึงนำเอกสารดังกล่าวเพื่อมาสร้างกรณีทดสอบการออกแบบก่อนจะถูกส่งต่อไปยังช่วงอีทีแอลเพื่อลดข้อผิดพลาดและลดระยะเวลาในการแก้ไขข้อผิดพลาดในช่วงอีทีแอล



รูปที่ 9 ขั้นตอนการทำงานของระบบ

3.1.1 Design Document

Design document คือ เอกสารการออกแบบที่เป็นผลจากการออกแบบของนักออกแบบคลังข้อมูลซึ่งถูกเขียนให้อยู่ในรูปแบบของเอกสารประเภทเอกซ์เซล ซึ่งมีส่วนประกอบหลัก 3 ส่วนคือ ข้อมูลตารางต้นทาง (Source table summary) ข้อมูลความสัมพันธ์ของตารางต้นทาง (Source table relationship) และข้อมูลตารางปลายทางที่สัมพันธ์กับตารางข้อมูลต้นทาง (Target table detail) ข้อมูลดังกล่าวทั้ง 3 ส่วนจะถูกเขียนขึ้นและถูกส่งไปยังนักพัฒนาในช่วงอีทีแอลต่อไป รายละเอียดข้อมูลแต่ละส่วนประกอบด้วย

- ส่วนข้อมูลตารางต้นทาง

System	ชื่อฐานข้อมูลเซิร์ฟเวอร์
Schema Name	ชื่อ User ที่สร้างตาราง
Table Name	ชื่อตาราง
Table Type	ชนิดของตารางมี 2 แบบคือ Master และ Supplementary
Selection Method	ชนิดการนำเข้าข้อมูลจากตารางมี 2 แบบคือ delta และ full
Delta Criteria	เงื่อนไขการนำข้อมูลเข้าหลังจาก implement สู่ระบบใช้งานจริง
Filtering Criteria	เงื่อนไขการกรองข้อมูลจากตารางต้นทาง

- ข้อมูลความสัมพันธ์ของตารางต้นทาง

Table A	ชื่อตารางหลักที่ใช้ผูกความสัมพันธ์
Table B	ชื่อตารางเสริมที่ใช้ผูกความสัมพันธ์
Join Type	รูปแบบความสัมพันธ์มี 2 ประกอบด้วย inner join และ left join
Join Condition	เงื่อนไขในการผูกความสัมพันธ์

- ส่วนข้อมูลตารางปลายทางที่สัมพันธ์กับตารางข้อมูลต้นทาง

Target Column Name	ชื่อคอลัมน์ในตารางที่ออกแบบ
Target Data type	ชนิดข้อมูลของคอลัมน์ในตารางที่ออกแบบ
Target PK	ตัวชี้ความเป็นค่าคีย์หลัก
Description	คำอธิบายชื่อคอลัมน์
Source Schema Name	Username ที่สร้างตารางต้นทาง
Source Table Name	ชื่อตารางต้นทาง
Source Column Name	ชื่อคอลัมน์ในตารางต้นทาง

Transformation Rule คำสั่งในการแปลงค่าจากคอลัมน์ต้นทางไปยังคอลัมน์ปลายทาง

ตารางที่ 2 รูปแบบการเขียนเอกสารการออกแบบส่วนข้อมูลตารางต้นทางในงานวิจัย

System	SERVERDEV1
Schema Name	SYSMAN
Table Name	CDR_DATA
Table Type	Master
Selection Method	Delta
Delta Criteria	CALL_DATE< START_DATE + 1
Filtering Criteria	CALL_TYPE in (1,2,4,5,13,12) and Duration > 0

ตารางที่ 3 รูปแบบการเขียนเอกสารการออกแบบส่วนข้อมูลความสัมพันธ์ของตารางต้นทาง

Table A	CDR_DATA
Table B	DIM_DATE
Join Type	Inner join
Join Condition	CDR_DATA.Call_date = DIM_DATE.Date_key

ตารางที่ 4 รูปแบบการเขียนเอกสารการออกแบบส่วนข้อมูลความสัมพันธ์ของตารางต้นทาง

Target Column Name	MSISDN
Target Data type	VARCHAR2(255)
Target PK	Null
Description	Mobile subscriber number
Source Schema Name	SYSMAN
Source Table Name	CDR_DATA
Source Column Name	Null
Transformation Rule	CASE WHEN CALL_TYPE IN (1,5,12,13) THEN A_MSISDN ELSE B_MSISDN END

3.1.2 Extract Data

Extract Data คือขั้นตอนของการสกัดข้อมูลออกจากเอกซ์เซลไฟล์ ตามข้อมูล 3 ส่วนที่อธิบายไว้ในเรื่องเอกสารการออกแบบในหัวข้อ 3.1.1 โดย python xlrd package นำมาใช้ในการสกัดข้อมูล โดยสรุปขั้นตอนดังนี้

1. ทำการ upload ไฟล์เข้าระบบ และบันทึกชื่อไฟล์รวมทั้งรหัส ID ไฟล์ไว้ในฐานข้อมูล
2. เปิดไฟล์ด้วยคำสั่ง xlrd.open_workbook(file_location) ของ python
3. ทำการอ่านข้อมูลในแต่ละ sheet ของเอกซ์เซลไฟล์ด้วยคำสั่ง open_doc.sheet_by_Inde x(sheet_no)
4. ทำการอ่านข้อมูลในแต่ละ cell ด้วยคำสั่ง sheet.cell_value(r,c) เมื่อ r คือเลขแถว และ c คือเลขหลักของข้อมูลภายใน sheet ของเอกซ์เซลไฟล์

3.1.3 Data for test case generation

ข้อมูลต่าง ๆ ทั้งชื่อไฟล์และข้อมูลที่สกัดออกมาจากไฟล์จะถูกนำไปจัดเก็บไว้ในฐานข้อมูลผ่านทาง model ของ Django framework เพื่อสร้างเป็นคำสั่งทดสอบเอกสารการออกแบบโดยตารางที่เก็บข้อมูลดังกล่าวมีดังนี้

- Class model TEDWCF_DESIGN_DOC ประกอบด้วย

doc_name	คอลัมน์เก็บ ชื่อของ user change request หรือ project number ที่ผู้ใช้งานกรอกลงไปขณะทำการ upload ไฟล์
doc_file	คอลัมน์เก็บชื่อของเอกซ์เซลไฟล์ที่ทำการ upload ไปที่ระบบ
doc_key	คอลัมน์เก็บค่า ID ของไฟล์ที่ทำการ upload ซึ่งเป็นค่าที่เกิดจากการนำค่าdoc_name มาต่อกับ ppn_tm
ppn_tm	เวลาที่ทำการ upload ไฟล์ไปยังระบบ

- Class model TEDWCF_SRCTABLE ประกอบด้วย

doc_src	คอลัมน์เก็บ ID ของไฟล์
system	คอลัมน์เก็บชื่อฐานข้อมูล
schema_nm	คอลัมน์เก็บชื่อ Username ที่เป็นเจ้าของตาราง
table_nm	คอลัมน์เก็บชื่อ source table
tabletype	คอลัมน์เก็บชนิดของ source table ประกอบด้วยประเภท master และประเภทsupplementary

select_method	คอลัมน์เก็บประเภทการดึงข้อมูลจากตารางว่าเป็นแบบ delta หรือ full
delta_criteria	คอลัมน์เก็บประเภทการนำข้อมูลเข้าหลัง implement บนระบบจริง
filter	คอลัมน์เก็บเงื่อนไขในการกรองข้อมูลจากตาราง
remark	คอลัมน์เก็บหมายเหตุ
ppn_tm	คอลัมน์เก็บเวลาที่ทำการบันทึกข้อมูล

ซึ่งตัวอย่างของเอกสารการออกแบบที่เก็บใน Class model TEDWCF_SRCTABLE แสดงดังรูปที่ 10

Data Source Summary

No.	System	Schema Name	Table Name	Table Type	Selection Method	Delta Criteria	Filtering Criteria
1	TEDW	CDR	CDRDB	Master	Delta	DATESTARTOFCHARGING >= START_DTE AND DATESTARTOFCHARGING < START_DTE + 1	(RECORDTYPE in (1,4) and CAUSE NOT IN (1,2)) or RECORDTYPE IN (5,7)
2	TEDW	CDR	LKP_DATE	Supplementary	Full		
3	TEDW	CDR	LKP_DOW_TOD	Supplementary	Full		

รูปที่ 10 ตัวอย่างของเอกสารการออกแบบที่เก็บใน Class model TEDWCF_SRCTABLE

- Class model TEDWCF_RELATION ประกอบด้วย

doc_ral	คอลัมน์เก็บ ID ของไฟล์
tablea	คอลัมน์เก็บชื่อตารางหลักที่ทำการผูกความสัมพันธ์
tableb	คอลัมน์เก็บชื่อตารางเสริมที่ทำการผูกความสัมพันธ์
jointype	คอลัมน์เก็บข้อมูลประเภทการผูกความสัมพันธ์
joincond	คอลัมน์เก็บข้อมูลเงื่อนไขที่ใช้ผูกความสัมพันธ์
ppn_tm	คอลัมน์เก็บเวลาที่ทำการบันทึกข้อมูล

ตัวอย่างเอกสารการออกแบบที่ถูกนำมาจัดเก็บที่ Class model TEDWCF_RELATION แสดงดังรูปที่ 11

Tables Relationship

No.	Table A	Table B	Join Type	Join Condition
1	CDRDB	LKP_DATE	Inner Join	TRUNC(CDRDB.DATESTARTOFCHARGING) = LKP_DATE.DATE_KEY
2	CDRDB	LKP_DOW_HOD	Inner Join	CASE WHEN LKP_DATE.IS_HOLIDAY = 'Y' THEN 'HOLIDAY' ELSE TRIM(TO_CHAR(CDRDB.DATESTARTOFCHARGING,'DAY')) END = LKP_DOW_HOD.DAY_OF_WEEK AND TRIM(TO_CHAR(CDRDB.DATESTARTOFCHARGING,'HH24')) = LKP_DOW_HOD.HOUR24

รูปที่ 11 ตัวอย่างเอกสารการออกแบบที่ถูกนำมาจัดเก็บที่ Class model TEDWCF_RELATION

- Class model TEDWCF_TARGET ประกอบด้วย

doc_tar	คอลัมน์เก็บ ID ของไฟล์
column_nm	คอลัมน์เก็บชื่อคอลัมน์ของตารางที่ออกแบบ
datatype	คอลัมน์เก็บชนิดข้อมูลของคอลัมน์ของตารางที่ออกแบบ
pk_key	คอลัมน์เก็บข้อมูลการเป็นค่าคีย์ของคอลัมน์ในตารางที่ออกแบบ
desc	คอลัมน์เก็บคำอธิบายคอลัมน์ของตารางที่ออกแบบ
s1_schema	คอลัมน์เก็บชื่อ Username ที่เป็นเจ้าของตารางต้นทางแหล่งที่ 1
s1_table	คอลัมน์เก็บชื่อตารางต้นทางจากแหล่งที่ 1
s1_column	คอลัมน์เก็บชื่อคอลัมน์ในตารางต้นทางแหล่งที่ 2
s1_rule	คอลัมน์เก็บการแปลงค่าจากคอลัมน์ในตารางต้นทางแหล่งที่ 1
s2_schema	คอลัมน์เก็บชื่อ Username ที่เป็นเจ้าของตารางต้นทางแหล่งที่ 2
s2_table	คอลัมน์เก็บชื่อตารางต้นทางจากแหล่งที่ 2
s2_column	คอลัมน์เก็บชื่อคอลัมน์ในตารางต้นทางแหล่งที่ 2
s2_rule	คอลัมน์เก็บการแปลงค่าจากคอลัมน์ในตารางต้นทางแหล่งที่ 2
s3_schema	คอลัมน์เก็บชื่อ Username ที่เป็นเจ้าของตารางต้นทางแหล่งที่ 3
s3_table	คอลัมน์เก็บชื่อตารางต้นทางจากแหล่งที่ 3
s3_column	คอลัมน์เก็บชื่อคอลัมน์ในตารางต้นทางแหล่งที่ 3
s3_rule	คอลัมน์เก็บการแปลงค่าจากคอลัมน์ในตารางต้นทางแหล่งที่ 3
s4_schema	คอลัมน์เก็บชื่อ Username ที่เป็นเจ้าของตารางต้นทางแหล่งที่ 4
s4_table	คอลัมน์เก็บชื่อตารางต้นทางจากแหล่งที่ 4
s4_column	คอลัมน์เก็บชื่อคอลัมน์ในตารางต้นทางแหล่งที่ 4
s4_rule	คอลัมน์เก็บการแปลงค่าจากคอลัมน์ในตารางต้นทางแหล่งที่ 4
s5_schema	คอลัมน์เก็บชื่อ Username ที่เป็นเจ้าของตารางต้นทางแหล่งที่ 5
s5_table	คอลัมน์เก็บชื่อตารางต้นทางจากแหล่งที่ 5
s5_column	คอลัมน์เก็บชื่อคอลัมน์ในตารางต้นทางแหล่งที่ 5
s5_rule	คอลัมน์เก็บการแปลงค่าจากคอลัมน์ในตารางต้นทางแหล่งที่ 5
ppn_tm	คอลัมน์เก็บเวลาที่ทำการบันทึกข้อมูล

ตัวอย่างเอกสารการออกแบบที่ถูกนำมาจัดเก็บที่ Class model TEDWCF_TARGET แสดงดังรูปที่ 12

Schema.Table Name : CDR.DAILY_EXTRACT

Target Table Information					Source Table Information			
No.	Column Name	Datatype	PK	Description	Schema Name	Table Name	Column Name	Transformation Rule
1	MSISDN	VARCHAR2(255)		Served MSISDN.	CDR	CDRDB	B_MSISDN	CASE WHEN RECORDTYPE IN (1,5,12,13) THEN A_MSISDN ELSE B_MSISDN END
2	IMEI	VARCHAR2(16)		International Mobile Equipment Identification	CDR	CDRDB	IMEI	IMEI
3	IMSI	VARCHAR2(16)		International Mobile Subscriber Identity	CDR	CDRDB	IMSI	IMSI
4	DAY	DATE		DATE of calllink to LKP_DATE	CDR	CDRDB	DATESTARTOFCHARGING	DATESTARTOFCHARGING
5	A_PARTY	VARCHAR2(255)		Calling _party	CDR	CDRDB	A_MSISDN	A_MSISDN

รูปที่ 12 ตัวอย่างเอกสารการออกแบบที่ถูกนำมาจัดเก็บที่ Class model TEDWCF_TARGET

3.1.4 Generate test case

การสร้างกรณีทดสอบเริ่มโดยการออกแบบแนวทางการสร้างกรณีทดสอบในขั้นของการออกแบบคลังข้อมูล เนื่องจากยังไม่เคยมีนำเอกสารการออกแบบคลังข้อมูลมาเป็นข้อมูลตั้งต้นในการสร้างกรณีทดสอบ ผู้วิจัยจึงประยุกต์ความรู้เกี่ยวกับการทดสอบในขั้นการออกแบบของคลังข้อมูลโดยการทดสอบจะแบ่งตามระดับของการออกแบบคือ

- การทดสอบในระดับ Conceptual schema
- การทดสอบในระดับ Logical schema
- การทดสอบในระดับ Physical schema

ข้อมูลที่ถูกเขียนเป็นเอกสารการออกแบบสามารถนำมาแทนในส่วนของ schema ทั้ง 3 ระดับข้างต้นเพื่อสร้างกรณีทดสอบในแต่ละระดับตามตารางที่ 5

ตารางที่ 5 การสร้างกรณีทดสอบการออกแบบแต่ละระดับจากข้อมูลเอกสารการออกแบบ

ส่วนข้อมูลออกแบบ	ระดับการทดสอบ		
	Conceptual	Logical	Physical
Source Table	✓	✓	
Join Tables		✓	
Join Type		✓	
Join Condition		✓	
Attribute Name			✓
Attribute Data Type			✓

Source Table Mapping	✓		✓
Source Attribute Mapping	✓		✓
Transformation Rules			✓

จากตารางที่ 5 อธิบายการสร้างกรณีทดสอบในระดับ conceptual โดยใช้ข้อมูลจากเอกสารการออกแบบในส่วนของ source table summary และใช้ข้อมูลส่วนของ target table detail การสร้างกรณีทดสอบในระดับ logical ใช้ข้อมูลในส่วนของ source table relationship และการสร้างกรณีทดสอบในระดับ physical ใช้ข้อมูลจากเอกสารการออกแบบในส่วนของ target table detail

การสร้างกรณีทดสอบในแต่ละระดับมีรายละเอียดดังต่อไปนี้

- การสร้างกรณีทดสอบในระดับ conceptual

การทดสอบในระดับ conceptual ที่ผู้วิจัยนำมาใช้ในการสร้างกรณีทดสอบมีวัตถุประสงค์เพื่อทดสอบภาระงานของการดำเนินการกับข้อมูลจากต้นทางโดยการใช้ Aggregate function และเพื่อทดสอบความสอดคล้องของการเขียนเอกสารการออกแบบจากระดับ conceptual ว่าสามารถสนับสนุนในส่วนของเอกสารการออกแบบในระดับ physical ได้จริงกรณีทดสอบแสดงดังตารางที่ 6

ตารางที่ 6 การสร้างกรณีทดสอบจากเอกสารการออกแบบในระดับ conceptual schema

ระดับการทดสอบ	ประเภทการทดสอบ	คำอธิบายประเภทการทดสอบ
Conceptual	Fact test	การทดสอบโดยใช้ Aggregate function ของภาษาเอสคิวแอล เพื่อวัดภาระงาน โดย Aggregate function ที่ใช้ทดสอบตามมาตรฐานประกอบด้วย 1)AVG หาค่าเฉลี่ยของ measure column 2)MIN หาค่าต่ำสุดของ measure column 3)SUM หาผลรวมของ measure column 4)COUNT นับจำนวนแถวทั้งหมด 5)MAX หาค่าสูงสุดของ measure column
Conceptual	Conform test	สร้างกรณีทดสอบโดยสร้างคำสั่งทดสอบจากส่วนของ source table summary และ target table detail เพื่อพิสูจน์การมีอยู่จริง

		ของ column name และ table name ที่สอดคล้องกัน

- การสร้างกรณีทดสอบในระดับ logical

การทดสอบในระดับ logical ที่ผู้วิจัยนำมาใช้ในการสร้างกรณีทดสอบมีวัตถุประสงค์เพื่อทดสอบความเป็นจริงของการผูกความสัมพันธ์ที่ถูกเขียนขึ้นในเอกสารการออกแบบและเพื่อทดสอบความสอดคล้องของข้อมูลในส่วนของ source table relationship และ source table summary ว่ามีความถูกต้องตรงกันหรือไม่ กรณีทดสอบแสดงดังตารางที่ 7

ตารางที่ 7 การสร้างกรณีทดสอบจากเอกสารการออกแบบในระดับ logical schema

ระดับการทดสอบ	ประเภทการทดสอบ	คำอธิบายประเภทการทดสอบ
Logical	Join test	การสร้างกรณีทดสอบโดยการสร้างคำสั่งตามข้อมูลในส่วนของ source table relationship เพื่อทดสอบความถูกต้องของการผูกความสัมพันธ์ตามเงื่อนไขที่ระบุ
Logical	Star test	การทดสอบโดยใช้ Aggregate function ของภาษาเอสคิวแอลที่ถูกสร้างขึ้นหลังจากการสร้างความสัมพันธ์ตามเงื่อนไข โดย Aggregate function ที่ใช้ทดสอบตามมาตรฐานประกอบด้วย 1)AVG หาค่าเฉลี่ยของ measure column 2)MIN หาค่าต่ำสุดของ measure column 3)SUM หาผลรวมของ measure column 4)COUNT นับจำนวนแถวทั้งหมด 5)MAX หาค่าสูงสุดของ measure column

- การสร้างกรณีทดสอบในระดับ physical

การทดสอบในระดับ physical ที่ผู้วิจัยนำมาใช้ในการสร้างกรณีทดสอบมีวัตถุประสงค์เพื่อทดสอบค่าความจริงระหว่างข้อมูลที่เขียนขึ้นในเอกสารการออกแบบกับข้อมูลจริงจากตารางต้นทางจากแหล่งต่าง ๆ ที่สามารถระบุได้ 5 แหล่ง รวมถึงการทดสอบข้อกำหนดต่าง ๆ เกี่ยวกับค่าที่จะบันทึกเป็นคอลัมน์ในตารางที่ออกแบบได้แก่ การทดสอบค่าชนิดของข้อมูลว่าชนิดของข้อมูลที่เขียนไว้

ในเอกสารการออกแบบสอดคล้องชนิดข้อมูลจริงของคอลัมน์ในตารางต้นทางหรือไม่ การทดสอบคอลัมน์ในเอกสารการออกแบบที่ระบุคีย์หลักว่าสามารถใช้ค่าคอลัมน์ดังกล่าวเป็นคีย์หลักได้หรือไม่ การทดสอบการแปลงค่าตามเงื่อนไขและการทดสอบการมีอยู่จริงของชื่อคอลัมน์ที่ระบุในเอกสารการออกแบบว่ามีอยู่จริงในตารางต้นทางที่ระบุในเอกสารการออกแบบดังกล่าวโดยกรณีทดสอบในระดับ physical แสดงดังตารางที่ 8

ตารางที่ 8 การสร้างกรณีทดสอบจากเอกสารการออกแบบในระดับ physical schema

ระดับการทดสอบ	ประเภทการทดสอบ	คำอธิบายประเภทการทดสอบ
Physical	Constraints test	การสร้างกรณีทดสอบโดยการสร้างคำสั่งในภาษาเอสคิวแอลเพื่อทดสอบดังต่อไปนี้ 1) Data type โดยคำสั่งเอสคิวแอลจะเขียนขึ้นเพื่อดำเนินการหาชนิดข้อมูลของ source column name ใน source table name ที่ได้ระบุไว้ในส่วนของ target table detail เพื่อเปรียบเทียบว่าตรงตามชนิดข้อมูลที่ระบุในเอกสารการออกแบบของ target column name หรือไม่ 2) Primary key โดยคำสั่งเอสคิวแอลที่เขียนขึ้นเพื่อดำเนินการเช็คว่า target column name ที่ถูกระบุให้เป็นคีย์หลักของตารางที่ออกแบบ สามารถเป็นคีย์ได้จริงหรือไม่หากค่าที่ได้จาก source column name มีค่าซ้ำกันหรือมีค่าว่าง ก็ไม่สามารถเป็นคีย์หลัก

Physical	Transform test	การทดสอบโดยการเขียนคำสั่งในภาษาเอสคิวแอลเพื่อทดสอบเงื่อนไขในการแปลงค่าจาก source column name และ source table name ว่าสามารถแปลงค่าได้จริงตามที่ระบุในเอกสารการออกแบบหรือไม่ และค่าที่เป็นไปได้ตรงกับที่ระบุในเอกสารการออกแบบหรือไม่ โดย เงื่อนไขการแปลงค่า(transformation rules) ในงานวิจัยกำหนดให้ใช้งานได้ 2 ลักษณะ คือ 1) คำสั่ง CASE WHEN 2) คำสั่ง IF ELSE

รูปแบบคำสั่งทดสอบในแต่ละกรณีทดสอบมีดังนี้

- Conceptual Fact test

```
SELECT AGGREGATE_FUNCTION(MEASURE_COLUMN) AS AGG
FROM OWNER.TABLE_NAME
```

- Conceptual Conform test

```
SELECT SUM(TOTAL_ROW) AS TOTAL_ROW FROM
(
SELECT COUNT(*) AS TOTAL_ROW FROM ALL_ALL_TABLES WHERE TO_CHAR(OWNER||TABLE_NAME) = 'OWNER&TABLE_NAME'
UNION
SELECT COUNT(*) AS TOTAL_ROW FROM ALL_VIEWS WHERE TO_CHAR(OWNER ||VIEW_NAME) = 'OWNER&TABLE_NAME'
)
```

- Logical Join test

```
SELECT COUNT(*) AS TRANSACTIONS
FROM OWNER.TABLE_NAME AS TABLE_NAME
```

```
INNER JOIN OWNER.TABLE_NAME AS TABLE_NAME
ON JOIN CONDITION
```

- Logical Star test

```
SELECT AGGREGATE_FUNCTION(COLUMN_NAME) AS AGG
FROM OWNER.TABLE_NAME AS TABLE_NAME
INNER JOIN OWNER.TABLE_NAME AS TABLE_NAME
ON JOIN CONDITION
```

- Physical Constraints test Data type

```
SELECT DATA_TYPE
FROM ALL_TAB_COLUMNS
WHERE OWNER ='OWNER' AND TABLE_NAME ='TABLE_NAME'
AND COLUMN_NAME ='COLUMN_NAME'
```

- Physical Constraints test Primary key

```
SELECT COUNT(RW) AS MANY_KEY
FROM  (
        SELECT ROW_NUMBER() OVER(PARTITION BY KEY_COLUMN_NAME ORDER BY
        KEY_COLUMN_NAME  )RW FROM OWNER.TABLE_NAME
        )
WHERE RW =2
```

- Physical Transform test CASE WHEN

```
SELECT * FROM
( SELECT DISTINCT
CASE WHEN CONDITION THEN
        CASE WHEN CONDITION THEN RESULT
        WHEN CONDITION      THEN  RESULT
        WHEN CONDITION      THEN  RESULT
        END
ELSE RESULT
```

```
END AS R FROM OWNER.TABLE_NAME
)
```

- Physical Transform test IF-ELSE ระบบทำการแก้ไขเงื่อนไขจาก If-else เป็น case-when แทน

```
SELECT * FROM
(
    SELECT DISTINCT
    CASE WHEN CONDITION THEN RESULT
    ELSE RESULT
    END AS R
    FROM OWNER.TABLE_NAME
)
```

3.1.5 Test case with SQL

หลังจากผู้ใช้งานเลือกคำสั่งสร้างกรณีทดสอบ คำสั่งทดสอบจะถูกเขียนขึ้นจากภาษาเอสคิวแอล ซึ่งโครงสร้างของการเขียนคำสั่งอธิบายไว้ในหัวข้อ 3.1.4 คำสั่งทดสอบจะถูกจัดเก็บไว้ที่ฐานข้อมูลที่ในตารางเดียวกันกับกรณีทดสอบเพื่อให้ง่ายแก่การเรียกใช้งานซึ่งกรณีทดสอบและคำสั่งทดสอบที่ถูกจัดเก็บที่ฐานข้อมูลดังกล่าวมีการออกแบบตามตารางที่ 9 ซึ่งการสร้างคำสั่งทดสอบสามารถมีได้มากสุด 5 ชุดคำสั่ง เนื่องจากผู้วิจัยมีใช้ประสบการณ์จากการทำงานมาวิเคราะห์ถึงความเป็นไปได้ที่คอลัมน์ 1 คอลัมน์อาจสามารถมีที่มาได้จากหลาย ๆ แหล่งซึ่งเป็นแนวคิดหลักของคลังข้อมูลที่ต้องการนำข้อมูลจากหลายแหล่ง ที่มีชนิดข้อมูลต่างกัน และอาจใช้ชื่อคอลัมน์ต่างกันมารวมเข้าไว้ในคอลัมน์เดียวกันเพื่อความสะดวกแก่การนำไปใช้งาน

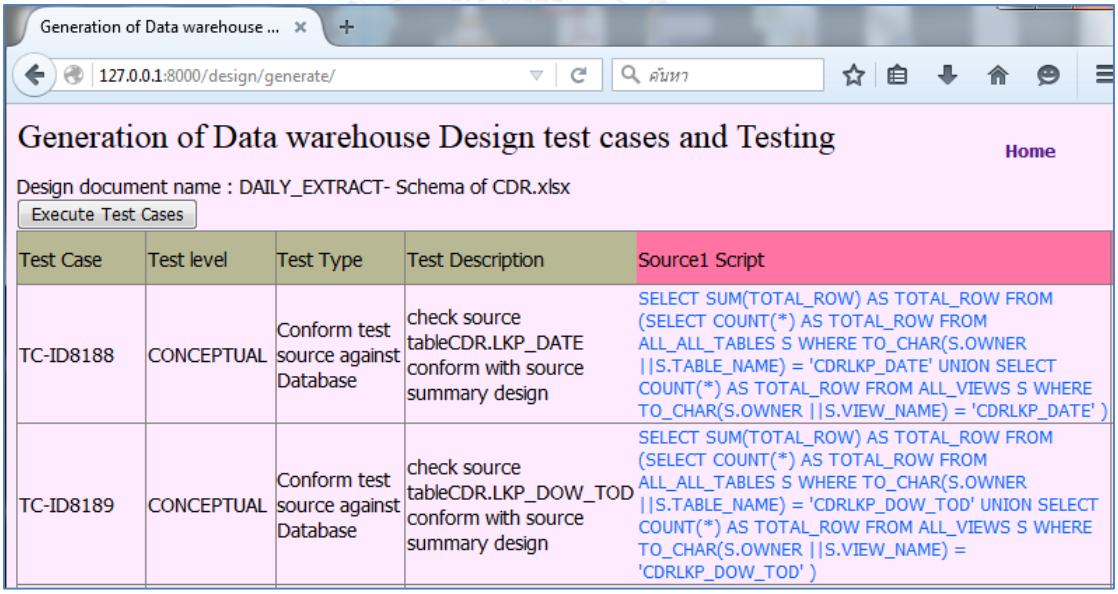
ตารางที่ 9 โครงสร้างข้อมูลสำหรับจัดเก็บกรณีทดสอบและคำสั่งทดสอบ

ชื่อคอลัมน์	**คำอธิบาย**
doc_test	บันทึกรหัสเอกสารการออกแบบที่นำมาทดสอบ
test_case_id	บันทึกรหัสกรณีทดสอบ
test_level	บันทึกระดับของการออกแบบที่ทดสอบ
test_type	บันทึกรูปแบบของการทดสอบ
test_desc	บันทึกคำอธิบายการทดสอบ

source_scr1	บันทึกคำสั่งทดสอบที่สร้างจากข้อมูลต้นทางแหล่งที่ 1
esource_scr1	บันทึกคำสั่งทดสอบจากข้อมูลต้นทางแหล่งที่ 1 ที่ถูกแก้ไขด้วยการวัดระยะการแก้ไข
exp_result1	บันทึกค่าการดำเนินการที่คาดหวังจากข้อมูลต้นทางแหล่งที่ 1
actual_result1	บันทึกค่าการดำเนินการจริงจากการดำเนินการกับคำสั่งทดสอบที่ 1
test_script_rs1	ผลการดำเนินการกับคำสั่งทดสอบที่สร้างจากข้อมูลแหล่งที่ 1
test_rs1	ผลการเปรียบเทียบผลลัพธ์จากการดำเนินการกับคำสั่งทดสอบจากแหล่งที่ 1 กับค่าที่คาดหวังจากเอกสารการออกแบบ
source_scr2	บันทึกคำสั่งทดสอบที่สร้างจากข้อมูลต้นทางแหล่งที่ 2
esource_scr2	บันทึกคำสั่งทดสอบจากข้อมูลต้นทางแหล่งที่ 2 ที่ถูกแก้ไขด้วยการวัดระยะการแก้ไข
exp_result2	บันทึกค่าการดำเนินการที่คาดหวังจากข้อมูลต้นทางแหล่งที่ 2
actual_result2	บันทึกค่าการดำเนินการจริงจากการดำเนินการกับคำสั่งทดสอบที่ 2
test_script_rs2	ผลการดำเนินการกับคำสั่งทดสอบที่สร้างจากข้อมูลแหล่งที่ 2
test_rs2	ผลการเปรียบเทียบผลลัพธ์จากการดำเนินการกับคำสั่งทดสอบจากแหล่งที่ 2 กับค่าที่คาดหวังจากเอกสารการออกแบบ
exp_result3	บันทึกค่าการดำเนินการที่คาดหวังจากข้อมูลต้นทางแหล่งที่ 3
actual_result3	บันทึกค่าการดำเนินการจริงจากการดำเนินการกับคำสั่งทดสอบที่ 3
source_scr3	บันทึกคำสั่งทดสอบที่สร้างจากข้อมูลต้นทางแหล่งที่ 3
esource_scr3	บันทึกคำสั่งทดสอบจากข้อมูลต้นทางแหล่งที่ 3 ที่ถูกแก้ไขด้วยการวัดระยะการแก้ไข
test_script_rs3	ผลการดำเนินการกับคำสั่งทดสอบที่สร้างจากข้อมูลแหล่งที่ 3
test_rs3	ผลการเปรียบเทียบผลลัพธ์จากการดำเนินการกับคำสั่งทดสอบจากแหล่งที่ 3 กับค่าที่คาดหวังจากเอกสารการออกแบบ
exp_result4	บันทึกค่าการดำเนินการที่คาดหวังจากข้อมูลต้นทางแหล่งที่ 4
actual_result4	บันทึกค่าการดำเนินการจริงจากการดำเนินการกับคำสั่งทดสอบที่ 4
source_scr4	บันทึกคำสั่งทดสอบที่สร้างจากข้อมูลต้นทางแหล่งที่ 4
esource_scr4	บันทึกคำสั่งทดสอบจากข้อมูลต้นทางแหล่งที่ 4 ที่ถูกแก้ไขด้วยการวัด

	ระยะการแก้ไข
test_script_rs4	ผลการดำเนินการกับคำสั่งทดสอบที่สร้างจากข้อมูลแหล่งที่ 4
test_rs4	ผลการเปรียบเทียบผลลัพธ์จากการดำเนินการกับคำสั่งทดสอบจากแหล่งที่ 4 กับค่าที่คาดหวังจากเอกสารการออกแบบ
source_scr5	บันทึกคำสั่งทดสอบที่สร้างจากข้อมูลต้นทางแหล่งที่ 5
esource_scr5	บันทึกคำสั่งทดสอบจากข้อมูลต้นทางแหล่งที่ 5 ที่ถูกแก้ไขด้วยการวัดระยะการแก้ไข
exp_result5	บันทึกค่าการดำเนินการที่คาดหวังจากข้อมูลต้นทางแหล่งที่ 5
actual_result5	บันทึกค่าการดำเนินการจริงจากการดำเนินการกับคำสั่งทดสอบที่ 5
test_script_rs5	ผลการดำเนินการกับคำสั่งทดสอบที่สร้างจากข้อมูลแหล่งที่ 5
test_rs5	ผลการเปรียบเทียบผลลัพธ์จากการดำเนินการกับคำสั่งทดสอบจากแหล่งที่ 5 กับค่าที่คาดหวังจากเอกสารการออกแบบ
ppn_tm	บันทึกเวลาที่ระบบสร้างกรณีทดสอบ

จากโครงสร้างข้อมูลที่ใช้ในการจัดเก็บกรณีทดสอบและคำสั่งทดสอบ ในส่วนที่แสดงผ่าน user interface ได้แสดงดังรูปที่ 13

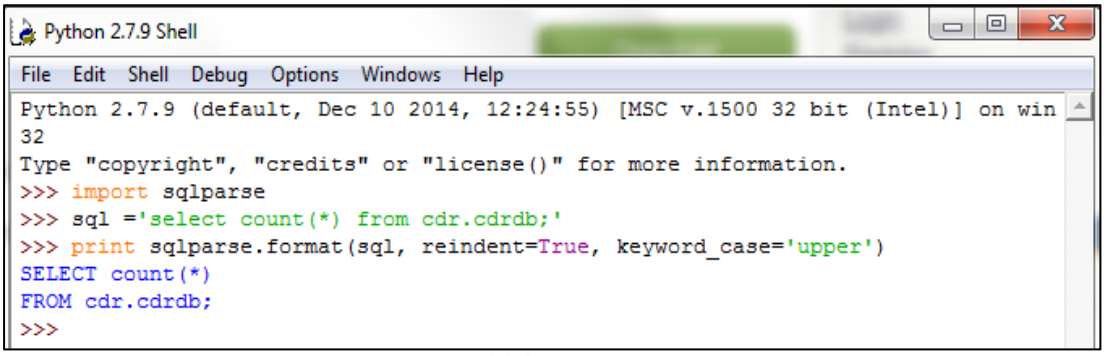


รูปที่ 13หน้าจอแสดงผลกรณีทดสอบและคำสั่งทดสอบ

3.1.6 Format SQL statement

หลังจากที่กรณีทดสอบและคำสั่งทดสอบถูกสร้างขึ้น ผู้วิจัยได้ทำการนำ package ของไพทอนมาประยุกต์ใช้เพิ่มเติมเพื่อนำมาจะระเบียบให้กับคำสั่งทดสอบในภาษาเอสคิวแอล โดย package ที่ใช้มี

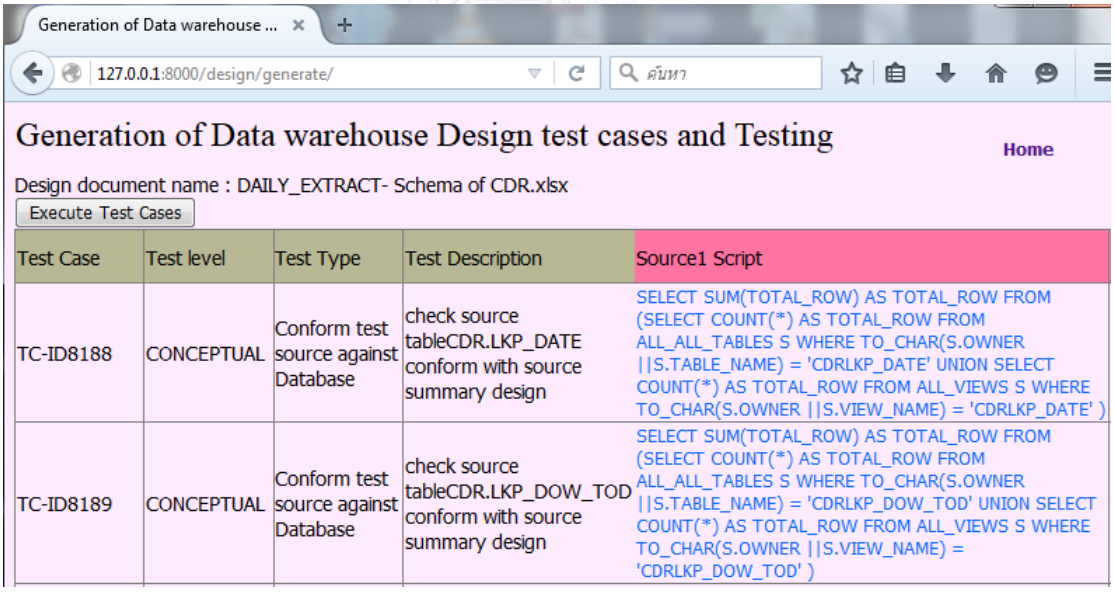
ชื่อว่า sqlparse ซึ่งหลังจากทำการจัดรูปแบบคำสั่งเอสคิวแอลที่ได้จะนำมาปรับค่าให้คำสั่งทดสอบเดิมก่อนนำไปดำเนินการ query ที่ฐานข้อมูลและนำมาแสดงที่หน้าจอ user interface ลักษณะการทำงานของ package sqlparse แสดงดังรูปที่ 14



รูปที่ 14 อธิบายลักษณะการทำงานของ sqlparse python package

3.1.7 Test case with corrected form at SQL

หลังจากที่ใช้ package sqlparse ของไพทอนเพื่อช่วยจัดรูปแบบคำสั่งทดสอบแล้วระบบจะนำคำสั่งทดสอบที่ได้ผ่านการจัดรูปแบบดังกล่าวไปปรับปรุงค่าในตารางกรณีทดสอบที่ฐานข้อมูลและนำมาแสดงผลให้ผู้ใช้ผ่านทาง user interface ดังรูปที่ 15

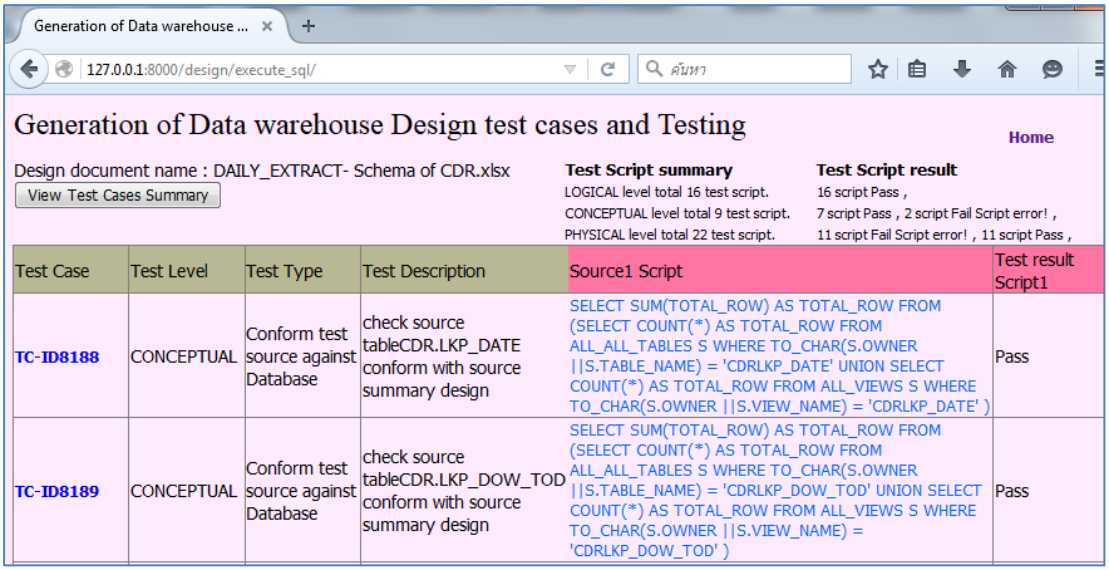


รูปที่ 15 หน้าจอแสดงกรณีทดสอบหลังคำสั่งทดสอบได้รับการจัดรูปแบบ

3.1.8 Execute test case

จากหัวข้อ 3.1.5 ระบบได้ทำการสร้างกรณีทดสอบและคำสั่งทดสอบเรียบร้อยแล้ว จากนั้นระบบแสดงกรณีทดสอบและคำสั่งทดสอบผ่านทางหน้าจอ user interface ให้ผู้ใช้งาน จากนั้นรอรับคำสั่งทดสอบกรณีทดสอบจากผู้ใช้งาน เพื่อส่งคำสั่งทดสอบไป query ข้อมูลจากฐานข้อมูลซึ่งฐานข้อมูลที่ใช้อ้างอิงในงานวิจัยนี้เป็นฐานข้อมูลออราเคิล เวอร์ชัน 10 เมื่อคำสั่งทดสอบถูกดำเนินการ query ที่

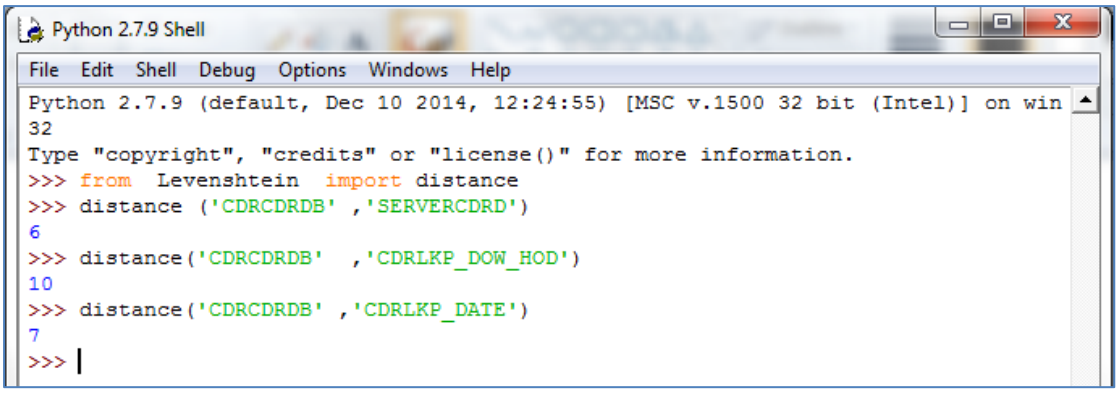
ฐานข้อมูลจะมี 2 กรณีที่เป็นไปได้ คือ 1.)สามารถ query ข้อมูลจากคำสั่งทดสอบได้และได้ผลลัพธ์จากการ query จากคำสั่งทดสอบหรือ 2.) ไม่สามารถ query ข้อมูลผลลัพธ์จากคำสั่งทดสอบที่ส่งไปยังฐานข้อมูลได้ หากเป็นตามกรณีที่ 1 ก็จะได้ผลลัพธ์จากการทดสอบกรณีทดสอบและนำผลลัพธ์ดังกล่าวไปปรับค่าให้ actual result ในตารางกรณีทดสอบที่ฐานข้อมูลและปรับค่า test script result ให้เป็นผ่าน (Pass) แต่หากเป็นตามกรณีที่ 2 ฐานข้อมูลจะไม่สามารถส่งผลลัพธ์จากคำสั่งทดสอบได้ แต่จะส่งข้อความที่ระบุข้อผิดพลาดของคำสั่งทดสอบกลับมายังระบบ จากนั้นระบบจะทำการปรับค่า test script result เป็นไม่ผ่าน (Fail Script error!) โดยหน้าจอ User interface จะแสดงข้อความว่าคำสั่งทดสอบของกรณีทดสอบมีข้อผิดพลาดดังแสดงในรูที่ 16 และทำการส่งคำสั่งทดสอบดังกล่าวไปแก้ไขด้วยการวัดระยะการแก้ไขในหัวข้อ 3.1.9



รูปที่ 16 หน้าจอแสดงผล test script result

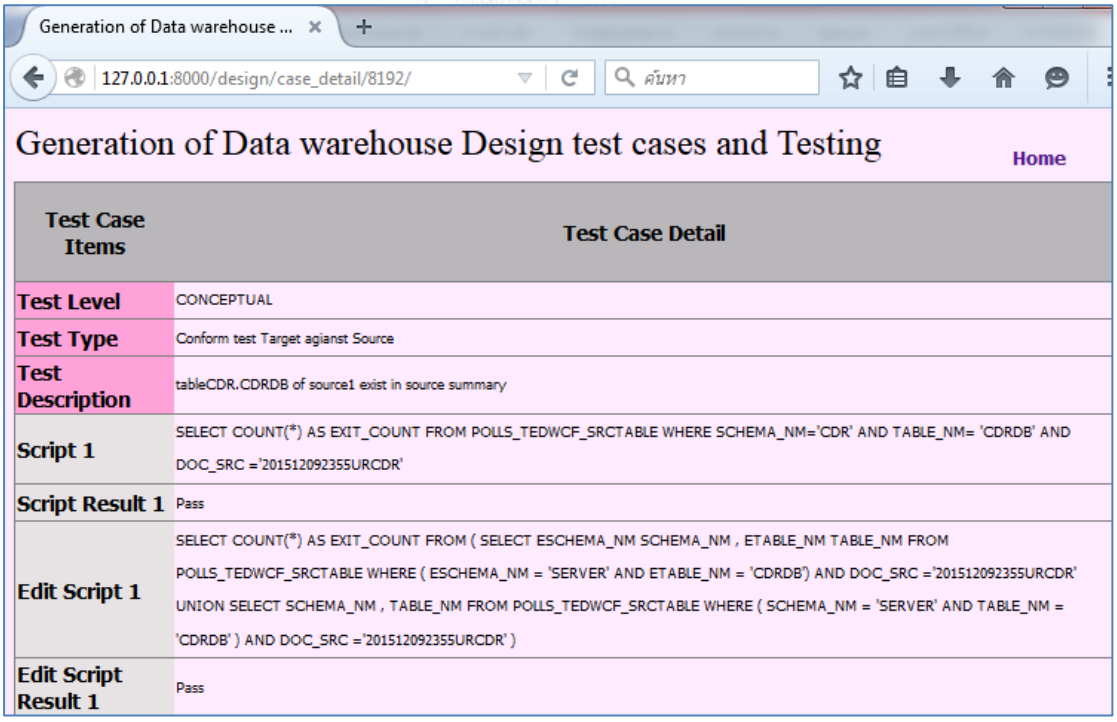
3.1.9 Edit SQL

จากหัวข้อที่ 3.1.8 ได้ทำการแก้ไขคำสั่งทดสอบที่มีได้รับข้อความระบุความผิดปกติจากฐานข้อมูลเพื่อเป็นการช่วยแนะนำผู้ออกแบบฐานข้อมูลให้ทราบว่าข้อมูลส่วนที่ระบุผิดในเอกสารการออกแบบควรปรับปรุงค่าเป็นค่าใด ผู้วิจัยจงได้นำองค์ความรู้เรื่องการวัดระยะการแก้ไขเพื่อคำนวณค่าชื่อตารางหรือชื่อคอลัมน์ที่มีลักษณะใกล้เคียงกับชื่อตารางหรือชื่อคอลัมน์ในคำสั่งทดสอบที่มีค่าผิดพลาดโดยการที่ระบบจะพิจารณาว่า ชื่อตารางหรือชื่อคอลัมน์ที่มีความผิดพลาดในคำสั่งทดสอบคือชื่อตารางหรือคอลัมน์ใด ระบบจะทำการค้นหาชื่อตารางหรือชื่อคอลัมน์ที่มีอยู่ในฐานข้อมูลส่วนของข้อมูลต้นทางเพื่อนำมาคำนวณหาชื่อตารางหรือชื่อคอลัมน์ที่ใกล้เคียงที่สุด โดยการคำนวณผู้วิจัยเลือกใช้ package ของภาษาไพทอนที่มีชื่อว่า python Levenshtein โดยลักษณะการทำงานของ python Levenshtein package มีลักษณะดังรูปที่ 17



รูปที่ 17 ลักษณะการทำงานของ python Levenshtein package

ระบบจะนำค่าคำสั่งทดสอบที่ได้รับการแก้ไขไปปรับปรุงค่า esource scr ในตารางกรณีทดสอบ และนำมาแสดงที่หน้าจอ user interface อีกครั้งดังรูปที่ 18 พร้อมกับระบบจะส่งคำสั่งทดสอบกรณีทดสอบดังกล่าวไปดำเนินการเพื่อหาผลลัพธ์ที่ฐานข้อมูลอีกครั้งแต่หากไม่สามารถดำเนินการได้และมีข้อความระบุความผิดพลาดส่งกลับมายังระบบ ระบบจะทำการกลับไปแก้ไขคำสั่งทดสอบด้วยการวัดระยะการแก้ไขอีกครั้ง แต่หากสามารถดำเนินการได้ และได้ผลลัพธ์จากการดำเนินการก็จะนำผลลัพธ์ดังกล่าวไปปรับปรุงค่า actual result ในตารางกรณีทดสอบเพื่อทำการเปรียบเทียบผลจากการดำเนินการกับผลที่คาดหวัง



รูปที่ 18 หน้าจอแสดงคำสั่งทดสอบที่ได้รับการแก้ไข

3.1.10 Compare result with expected result

หลังจากระบบได้ทำการดำเนินการกับคำสั่งทดสอบซึ่งบางคำสั่งอาจไม่สามารถดำเนินการเพื่อหาผลลัพธ์จากฐานข้อมูลได้ แต่คำสั่งทดสอบโดยส่วนใหญ่ที่สามารถหาค่าการดำเนินการได้ และค่าดำเนินการดังกล่าวถูกนำมาเก็บไว้ที่ actual result ในตารางกรณีทดสอบ จากนั้นในขั้นตอนนี้จะทำการเปรียบเทียบผลที่ได้จากการดำเนินการและผลที่คาดหวังที่ได้สกัดมาจากเอกสารการออกแบบ ซึ่งการเปรียบเทียบผลการทดสอบในแต่ละกรณีทดสอบมีดังนี้

- ผลจากการดำเนินการกับคำสั่งทดสอบในกรณีทดสอบประเภท Conceptual aggregate test ค่าที่คาดหวังเป็นค่าตัวเลขใด ๆ ซึ่งได้จากผลของ aggregate function ผลของกรณีทดสอบจะเป็น Pass เมื่อค่าที่ได้จากการดำเนินการกับคำสั่งทดสอบหรือคำสั่งทดสอบที่ได้รับการแก้ไขเป็นตัวเลขใด ๆ และไม่เป็นค่า Null
- ผลจากการดำเนินการกับคำสั่งทดสอบในกรณีทดสอบประเภท Conceptual Conform test source against Database ค่าที่คาดหวังของกรณีทดสอบเป็นค่าตัวเลข 1 หมายความว่าชื่อตารางดังกล่าวมีอยู่จริงในฐานข้อมูล ดังนั้นผลการทดสอบกรณีทดสอบจะเป็น Pass เมื่อค่าที่ได้จากการรันคำสั่งทดสอบหรือคำสั่งทดสอบที่ได้รับการแก้ไขเป็นเลข 1
- ผลจากการดำเนินการกับคำสั่งทดสอบในกรณีทดสอบประเภท Conceptual Conform test Target against Source ค่าที่คาดหวังของกรณีทดสอบเป็นค่าตัวเลข 1 หมายความว่าชื่อตารางดังกล่าวมีอยู่จริงในฐานข้อมูล ดังนั้นผลการทดสอบกรณีทดสอบจะเป็น Pass เมื่อค่าที่ได้จากการรันคำสั่งทดสอบหรือคำสั่งทดสอบที่ได้รับการแก้ไขเป็นเลข 1
- ผลจากการดำเนินการกับคำสั่งทดสอบในกรณีทดสอบประเภท Logical Join condition test ค่าที่คาดหวังจากการทดสอบคำสั่งทดสอบคือตัวเลขใด ๆ ที่มีค่ามากกว่า 0 ซึ่งหมายความว่า การผูกความสัมพันธ์ดังกล่าวให้ค่าผลลัพธ์ของข้อมูลออกมามากกว่า 0 แถวดังนั้นผลของกรณีทดสอบจะเป็น Pass เมื่อผลการดำเนินการกับคำสั่งทดสอบหรือคำสั่งทดสอบที่ได้รับการแก้ไขมีค่ามากกว่า 0
- ผลจากการดำเนินการกับคำสั่งทดสอบในกรณีทดสอบประเภท Logical Relation star test ค่าที่คาดหวังเป็นค่าตัวเลขใด ๆ ซึ่งได้จากผลของ aggregate function ผลของกรณีทดสอบจะเป็น Pass เมื่อค่าที่ได้จากการดำเนินการกับคำสั่งทดสอบหรือคำสั่งทดสอบที่ได้รับการแก้ไขเป็นตัวเลขใด ๆ และไม่เป็นค่า Null
- ผลจากการดำเนินการกับคำสั่งทดสอบในกรณีทดสอบประเภท Physical Constraints test data type ค่าที่คาดหวังจากการดำเนินการกับคำสั่งทดสอบหรือคำสั่งทดสอบที่ได้รับการ

แก้ไขคือ ค่าชนิดขอมูลที่ผู้ออกแบบเขียนไว้ในเอกสารการออกแบบส่วน เช่น VARCHAR(20) ดังนั้นหากทดสอบคำสั่งทดสอบแล้วได้ค่าออกมาเป็น VARCHAR ผลของกรณีทดสอบคือ Pass

- ผลจากการดำเนินการกับคำสั่งทดสอบในกรณีทดสอบประเภท Physical Constraints test primary key ค่าที่คาดหวังจากการดำเนินการกับคำสั่งทดสอบหรือคำสั่งทดสอบที่ได้รับการแก้ไขคือ เลข 0 แปลว่าไม่มีข้อมูลจากคอลัมน์ที่ถูกระบุว่าเป็น primary key ที่มีมากกว่า 1 แถว ดังนั้นผลกรณีทดสอบจะเป็น Pass เมื่อผลจากคำสั่งทดสอบเป็นเลข 0
- ผลจากการดำเนินการกับคำสั่งทดสอบในกรณีทดสอบประเภท Physical Transform test Case When ค่าที่คาดหวังจากการดำเนินการกับคำสั่งทดสอบหรือคำสั่งทดสอบที่ได้รับการแก้ไขคือค่าที่ได้จากการสกัด when condition ที่ระบุในเอกสารการออกแบบเช่น ‘D’, ‘C’ ดังนั้นหากค่าที่ได้จากการดำเนินการกับกรณีทดสอบเป็น ‘D’, ‘C’ ผลของกรณีทดสอบจะเป็น Pass
- ผลจากการดำเนินการกับคำสั่งทดสอบในกรณีทดสอบประเภท Physical Transform test If Else ค่าที่คาดหวังจากการดำเนินการกับคำสั่งทดสอบหรือคำสั่งทดสอบที่ได้รับการแก้ไขคือค่าที่ได้จากการสกัด if-else condition ที่ระบุในเอกสารการออกแบบเช่น ‘D’, ‘C’ ดังนั้นหากค่าที่ได้จากการดำเนินการกับกรณีทดสอบเป็น ‘D’, ‘C’ ผลของกรณีทดสอบจะเป็น Pass

3.1.11 Update actual result / test result

ผลที่ได้จากขั้นตอนที่ 3.1.10 คือ 1.)หากค่าที่ได้จากการดำเนินการตรงกับค่าที่คาดหวังระบบจะทำการปรับปรุงค่า TEST RESULT ให้เป็นผ่าน (Pass) 2.).ในกรณีที่ผลลัพธ์จากการดำเนินการไม่ตรงกับค่าที่คาดหวังระบบจะปรับปรุงค่า TEST RESULT ในตารางกรณีทดสอบให้เป็นไม่ผ่าน (Fail) ซึ่งหมายความว่าการออกแบบของผู้ออกแบบตั้งสมมติฐานค่าที่ควรให้แก่คอลัมน์ในคลังข้อมูลไม่ตรงกับค่าจริงในฐานข้อมูลต้นทาง หลังจากทำการปรับค่า actual result และ test result แล้วระบบจะแสดงค่าต่าง ๆ ผ่านทาง user interface อีกครั้งดังแสดงในรูปที่ 19 ซึ่งประกอบด้วย

- Test case ID แสดงลำดับตัวเลขกรณีทดสอบ
- Test level ระดับของกรณีทดสอบแบ่งตามข้อมูลการออกแบบซึ่งมี 3 ค่า คือ Conceptual, Logical, และ Physical
- Test Description แสดงคำอธิบายของกรณีทดสอบ

- Test script แสดงคำสั่งทดสอบที่ถูกสร้างขึ้นจากข้อมูลในเอกสารการออกแบบตั้งแต่แหล่งที่ 1 จนถึงแหล่งที่ 5
- Edit test script แสดงคำสั่งทดสอบที่ถูกแก้ไขโดยระบบ เมื่อเกิดกรณีที่ไม่สามารถดำเนินการที่ฐานข้อมูลได้ตั้งแต่แหล่งที่ 1 จนถึงแหล่งที่ 5
- Actual result ผลจากการดำเนินการของคำสั่งทดสอบที่สามารถดำเนินการผ่านตั้งแต่ 1 ถึง 5
- Test result แสดงผลของการเปรียบเทียบค่าที่คาดหวังที่ได้จากเอกสารการออกแบบกับค่าที่ได้รับจากการดำเนินการกับคำสั่งทดสอบตั้งแต่ 1 ถึง 5

โดยตัวอย่างหน้าจอแสดงผล Actual result และ test result แสดงดังรูปที่ 19

Generation of Data warehouse Design test cases and Testing Home

Test Case Items	Test Case Detail
Test Level	CONCEPTUAL
Test Type	Conform test source against Database
Test Description	check source tableCDR.LKP_DOW_TOD conform with source summary design
Script 1	SELECT SUM(TOTAL_ROW) AS TOTAL_ROW FROM (SELECT COUNT(*) AS TOTAL_ROW FROM ALL_ALL_TABLES S WHERE TO_CHAR(S.OWNER ||S.TABLE_NAME) = 'CDRLKP_DOW_TOD' UNION SELECT COUNT(*) AS TOTAL_ROW FROM ALL_VIEWS S WHERE TO_CHAR(S.OWNER ||S.VIEW_NAME) = 'CDRLKP_DOW_TOD')
Script Result 1	Pass
Edit Script 1	SELECT SUM(TOTAL_ROW) AS TOTAL_ROW FROM (SELECT COUNT(*) AS TOTAL_ROW FROM ALL_ALL_TABLES S WHERE TO_CHAR(S.OWNER ||S.TABLE_NAME) = 'CDRLKP_DOW_HOD' UNION SELECT COUNT(*) AS TOTAL_ROW FROM ALL_VIEWS S WHERE TO_CHAR(S.OWNER ||S.VIEW_NAME) = 'CDRLKP_DOW_HOD')
Edit Script Result 1 Expected Result 1	Pass table name CDR.LKP_DOW_TOD exist in database
Actual Result 1	[{'TOTAL_ROW': 0}]
Edit Script Actual Result 1	[{'TOTAL_ROW': 1}]
Test Result 1	Pass

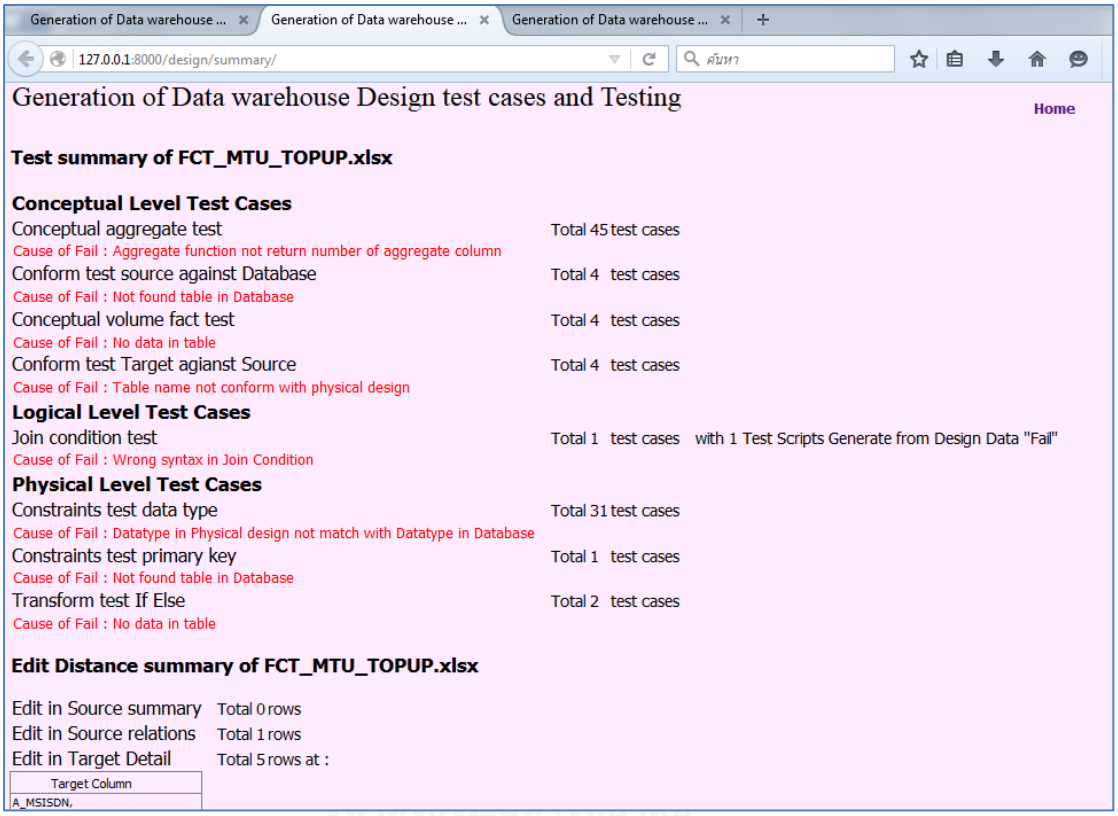
รูปที่ 19 ตัวอย่างหน้าจอแสดงผล Actual result และ Test result

3.1.12 Report test result summary

หน้าจอแสดงผลสรุปการทดสอบกรณีทดสอบแสดงดังรูปที่ 20 ซึ่งรูปแบบที่แสดงส่วนที่สำคัญจะประกอบด้วย

- ส่วนของ Test type เพื่อบอกประเภทของการกรณีทดสอบ
- ส่วนของจำนวนกรณีทดสอบในแต่ละประเภท

- ส่วนของจำนวนคำสั่งทดสอบที่ได้สร้างจากข้อมูลในเอกสารการออกแบบและยังไม่ได้รับการแก้ไขและมีค่าการดำเนินการเป็นไม่ผ่าน
- ส่วนของรายละเอียดการแก้ไขข้อมูลในส่วนของเอกสารการออกแบบ แบ่งตามระดับของการออกแบบ ซึ่งจะแสดงให้ผู้ใช้งานเห็นถึงความผิดปกติของข้อมูลในเอกสารการออกแบบได้



รูปที่ 20 หน้าจอแสดงผลสรุปการทดสอบกรณีทดสอบ

บทที่ 4
การออกแบบและพัฒนาระบบ

รายละเอียดบทนี้จะอธิบายการออกแบบและพัฒนาระบบต้นแบบเพื่อสนับสนุนแนวทางในการสร้างกรณีทดสอบการออกแบบคลังข้อมูลที่ได้นำเสนอในบทที่ 3 โดยเนื้อหาประกอบด้วย ข้อกำหนดเบื้องต้นของระบบ ความต้องการเชิงหน้าที่ แผนภาพยูสเคส คำอธิบายยูสเคส การออกแบบ และการพัฒนาระบบ

4.1 ข้อกำหนดเบื้องต้นของระบบ

4.1.1 ผู้ใช้งาน (User)

ผู้ใช้งานเครื่องมือต้นแบบควรเป็นผู้ออกแบบคลังข้อมูล ผู้ที่มีส่วนเกี่ยวข้องในการพัฒนาคลังข้อมูล นักทดสอบ หรือผู้ที่มีความรู้เกี่ยวกับภาพรวมของฐานข้อมูลซึ่งจะสามารถช่วยในการตรวจสอบและช่วยให้คำแนะนำเมื่อได้ผลการทดสอบกรณีไม่ผ่าน

4.1.2 ข้อมูลนำเข้า (Input)

เอกสารการออกแบบคลังข้อมูลได้มาจากช่วงการออกแบบซึ่งเขียนโดยผู้ออกแบบคลังข้อมูลมีรูปแบบของการเขียนเป็นเอกสารประเภท Excel โดยใช้เครื่องมือ เอกซ์เซล รวมถึงต้องมีรายละเอียดที่เพียงพอต่อการทดสอบและมีค่าที่ระบบสามารถรองรับได้ ภายในเอกสารเอกซ์เซลต้องประกอบด้วยส่วน ข้อมูลตารางต้นทาง (Source table summary) ข้อมูลความสัมพันธ์ของตารางต้นทาง (Source table relationship) และข้อมูลตารางปลายทางที่สัมพันธ์กับตารางข้อมูลต้นทาง (Target table detail)

4.1.3 ข้อมูลนำออก (Output)

- รายการกรณีทดสอบแสดงออกในรูปแบบตารางบนหน้าจอ user interface
- คำสั่งทดสอบเป็นรูปแบบภาษาเอสคิวแอล

4.1.4 ข้อจำกัดของระบบ (Constraint)

ผลลัพธ์ที่ได้จากการทดสอบแสดงเพียง PASS หรือ FAILED โดยไม่มีรายละเอียดอธิบายความผิดพลาดในกรณีทดสอบแล้ว FAILED เนื่องจากข้อมูลที่หลากหลายทำให้ในส่วนของการดักจับข้อผิดพลาดของการทดสอบที่ยังซับซ้อนอยู่มาก จึงยังไม่สามารถระบุข้อผิดพลาดออกมาได้อย่างชัดเจน ในกรณีของการหาค่าผลลัพธ์ที่คาดหวังสำหรับการทดสอบ Transform test ในระดับ Physical ยังรองรับการเขียนเงื่อนไขด้วย CASE WHEN และ IF ELSE เท่านั้น

4.2 ความต้องการเชิงหน้าที่ (Functional Requirements)

ระบบสร้างกรณีทดสอบการออกแบบคลังข้อมูลมีความต้องการเชิงหน้าที่แสดงดังตารางที่ 10

ตารางที่ 10 ความต้องการเชิงหน้าที่

รหัสอ้างอิง	ชื่อ	คำอธิบาย
FR-01	นำเอกสารการออกแบบเข้าในรูปแบบเอกสารเอกซ์เซล	ระบบสามารถนำเข้าเอกสารการออกแบบในรูปแบบเอกสารเอกซ์เซลได้
FR-02	ค้นหาเอกสารการออกแบบ	ระบบสามารถค้นหาเอกสารการออกแบบจากคำค้น ผ่าน user interface เพื่อทำการสร้างกรณีทดสอบ
FR-03	สกัดข้อมูลการออกแบบ	ระบบสามารถสกัดข้อมูลภายในเอกสารการออกแบบได้โดยมีค่ากำหนดตามตารางที่11-13 ได้
FR-04	สร้างกรณีทดสอบและคำสั่งทดสอบด้วยภาษาเอสคิวแอล	ระบบสามารถนำข้อมูลที่ได้จากการสกัดจาก FR-03 เพื่อสร้างกรณีทดสอบตามตารางที่ 11-13 และสร้างคำสั่งทดสอบได้
FR-05	ส่งคำสั่งทดสอบไปยังฐานข้อมูลเพื่อหาผลลัพธ์	ระบบสามารถส่งคำสั่งทดสอบไปยังฐานข้อมูลเพื่อดำเนินการหาผลลัพธ์หรือข้อผิดพลาดจากคำสั่งที่สร้างขึ้นมาได้
FR-06	แสดงผลลัพธ์จากคำสั่งทดสอบ	ระบบสามารถแสดงผลลัพธ์จากการทดสอบ
FR-07	แก้ไขคำสั่งทดสอบบางกรณีด้วยการวัดระยะการแก้ไข	ระบบสามารถช่วยแก้ไขคำสั่งที่ส่งไปยังฐานข้อมูลและไม่สามารถดำเนินการรันคำสั่งดังกล่าวได้
FR-08	ส่งคำสั่งทดสอบที่แก้ไขแล้วไปยังฐานข้อมูลเพื่อหาผลลัพธ์	ระบบสามารถนำคำสั่งที่ได้จาก FR-07 ส่งไปยังฐานข้อมูลเพื่อหาผลลัพธ์ใหม่อีกครั้งได้
FR-09	แสดงผลลัพธ์จากคำสั่งทดสอบที่แก้ไข	ระบบสามารถแสดงผลลัพธ์จากคำสั่งในขั้น FR-07 ได้
FR-10	แสดงรายละเอียดหลังการทดสอบกรณีทดสอบ	ระบบสามารถแสดงรายละเอียดหลังการทดสอบกรณีทดสอบแต่ละกรณีได้ โดยที่ผู้ใช้เลือกดูผลกรณีทดสอบที่แสดงบน user interface

ตารางที่ 11 ค่ากำหนดในเอกสารการออกแบบส่วนของตารางต้นทาง (Source table relationship)

ข้อมูล	รายละเอียด
System	ชื่อฐานข้อมูลเซิร์ฟเวอร์ สามารถเป็นค่าว่างได้
Schema Name	ชื่อ User ที่สร้างตาราง ไม่สามารถเป็นค่าว่างได้
Table Name	ชื่อตาราง ไม่สามารถเป็นค่าว่างได้
Table Type	ชนิดของตารางมี 2 แบบคือ Master และ Supplementary
Selection Method	ชนิดการนำเข้าข้อมูลจากตารางมี 2 แบบคือ delta และ full
Delta Criteria	สามารถเป็นค่าว่างได้
Filtering Criteria	สามารถเป็นค่าว่างได้

ตารางที่ 12 ค่ากำหนดในเอกสารการออกแบบส่วนของข้อมูลความสัมพันธ์ของตารางต้นทาง (Source table relationship)

ข้อมูล	รายละเอียด
Table A	ชื่อตารางหลักที่ใช้ผูกความสัมพันธ์ ห้ามเป็นค่าว่าง
Table B	ชื่อตารางเสริมที่ใช้ผูกความสัมพันธ์ ห้ามเป็นค่าว่าง
Join Type	มี 2 แบบคือ inner join และ left join
Join Condition	เงื่อนไขในการผูกความสัมพันธ์ ห้ามเป็นค่าว่าง

ตารางที่ 13 ค่ากำหนดในเอกสารการออกแบบส่วนของข้อมูลตารางปลายทางที่สัมพันธ์กับตารางข้อมูลต้นทาง (Target table detail)

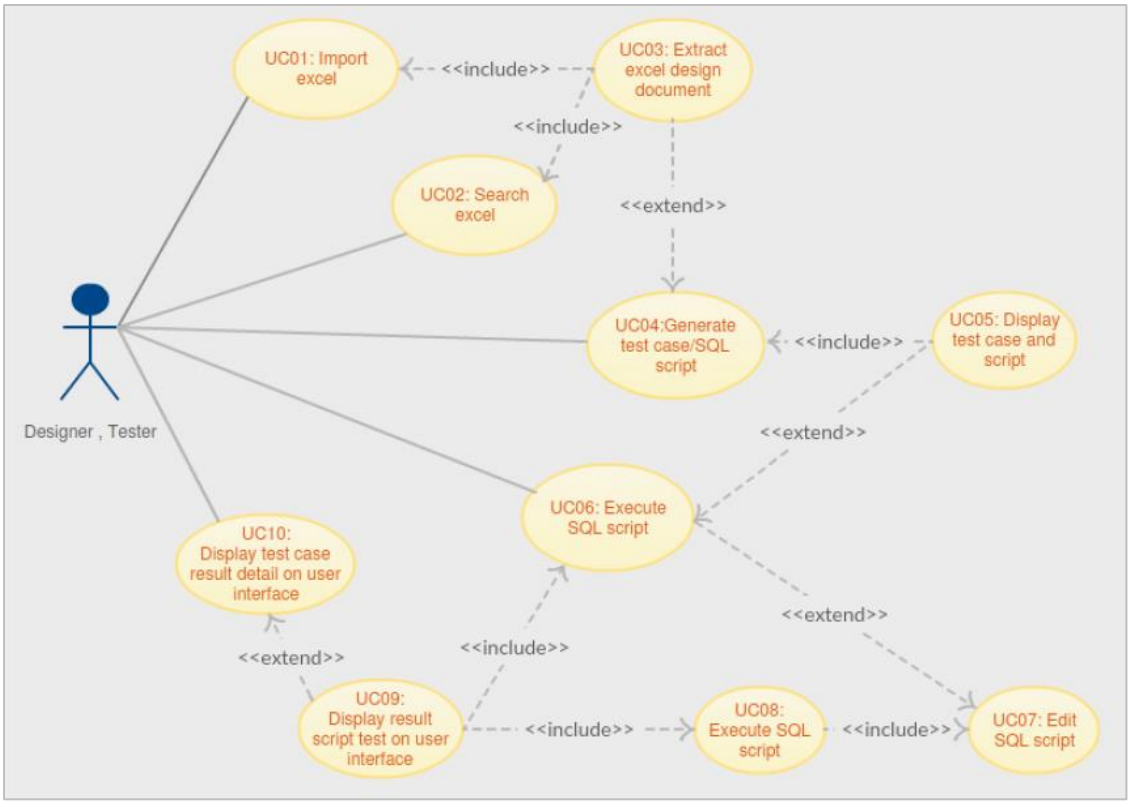
ข้อมูล	รายละเอียด
Target Column Name	ชื่อคอลัมน์ในตารางที่ออกแบบ ห้ามเป็นค่าว่าง
Target Data type	ชนิดข้อมูลของคอลัมน์ในตารางที่ออกแบบ ห้ามเป็นค่าว่าง
Target PK	ตัวชี้ความเป็นค่าคีย์หลัก สามารถเป็นค่าว่างได้
Description	คำอธิบายชื่อคอลัมน์ สามารถเป็นค่าว่างได้
Source Schema Name	User ที่สร้างตารางต้นทาง ห้ามเป็นค่าว่าง
Source Table Name	ชื่อตารางต้นทาง ห้ามเป็นค่าว่าง
Source Column Name	ชื่อคอลัมน์ในตารางต้นทางที่ต้องการนำมาเป็นค่าในคอลัมน์ในตาราง

	ที่ออกแบบ ห้ามเป็นค่าว่าง
Transformation Rule	คำสั่งในการแปลงค่าจากคอลัมน์ต้นทางไปยังคอลัมน์ปลายทาง สามารถเป็นค่าว่างได้

4.3 การออกแบบระบบ

4.3.1 แผนภาพยูสเคส

จากความต้องการเชิงหน้าที่สามารถแสดงเป็นแผนภาพยูสเคส (Use Case) ของระบบการสร้างกรณีทดสอบการออกแบบคลังข้อมูลได้ดังรูปที่ 21



รูปที่ 21 แผนภาพยูสเคสของระบบสร้างกรณีทดสอบการออกแบบคลังข้อมูล

4.3.2 คำอธิบายยูสเคส

อธิบายถึงฟังก์ชันความต้องการเชิงหน้าที่ ว่าผู้ใช้งานมีความเกี่ยวข้องอย่างไรกับระบบ ระบบต้องตอบสนองผู้ใช้งานอย่างไร และขั้นตอนการทำงานของระบบเป็นอย่างไร โดยคำอธิบายยูสเคสแสดงดังตารางที่ 14 ถึงตารางที่ 23

ตารางที่ 14 คำอธิบายยูสเคส Import Excel

หมายเลขยูสเคส:	UC01	เวอร์ชัน:	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (High)		
ผู้เกี่ยวข้อง:	ผู้ออกแบบคลังข้อมูลหรือนักทดสอบ		
รายละเอียดโดยย่อ:	อธิบายการนำเข้าเอกสารการออกแบบในรูปแบบเอกซ์เซล		
เงื่อนไขขั้นต้น:	เอกสารการออกแบบต้องอยู่ในรูปแบบเอกสารเอกซ์เซลและมีรูปแบบการเขียนที่ถูกต้องเหมาะสมแก่การดำเนินการ		
การทำงานโดยปกติ:	1. ผู้ใช้งานกดปุ่มเพิ่มเอกสารการออกแบบฉบับใหม่ 2. ผู้ใช้งานเลือกปุ่มเลือกไฟล์ 3. ระบบตรวจสอบรูปแบบไฟล์ 4. ระบบจัดเก็บไฟล์และบันทึกข้อมูลไฟล์ลงฐานข้อมูล		
ทางเลือกเพิ่มเติมในการทำงาน:	1. นำเข้าไฟล์ที่มีรูปแบบไม่ถูกต้อง 2. ระบบไม่ทำการจัดเก็บไฟล์ 3. กลับไปทำขั้นตอนที่ 1 อีกครั้ง		

ตารางที่ 15 คำอธิบายยูสเคส Search Excel

หมายเลขยูสเคส:	UC02	เวอร์ชัน:	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (Moderate)		
ผู้เกี่ยวข้อง:	-		
รายละเอียดโดยย่อ:	อธิบายการค้นหาเอกสารการออกแบบจากคำค้นชื่อเอกสาร		
เงื่อนไขขั้นต้น:	มีข้อมูลเอกสารการออกแบบจัดเก็บในระบบ		
การทำงานโดยปกติ:	1. ผู้ใช้ทำการป้อนคำค้นที่เป็นชื่อเอกสารการออกแบบ 2. ผู้ใช้งานกดปุ่มค้นหา 3. เอกสารที่ชื่อมีส่วนประกอบของคำค้นแสดงบน user interface		
ทางเลือกเพิ่มเติมในการทำงาน:	1. ไม่มีชื่อเอกสารใด ๆ ที่มีส่วนประกอบของคำค้น 2. ระบบกลับไปแสดงหน้าเอกสารการออกแบบทั้งหมด		

ตารางที่ 16 คำอธิบายยูสเคส Extract Excel design document

หมายเลขยูสเคส:	UC03	เวอร์ชัน:	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (High)		
ผู้เกี่ยวข้อง:	-		
รายละเอียดโดยย่อ:	อธิบายการสกัดคำจากเอกสารการออกแบบ		
เงื่อนไขขั้นต้น:	มีการนำเข้าเอกสารการออกแบบอย่างสมบูรณ์		
การทำงานโดยปกติ:	1. ระบบทำการเปิดไฟล์และอ่านข้อมูลภายในไฟล์ 2. ระบบทำการบันทึกข้อมูลที่ต้องการทดสอบในไฟล์ลงฐานข้อมูลที่ระบุตามเงื่อนไข		
ทางเลือกเพิ่มเติมในการทำงาน:	1. ข้อมูลในไฟล์มีรูปแบบที่ไม่สามารถอ่านได้ตามเงื่อนไข 2. ระบบแจ้งเตือนรูปแบบไฟล์ที่ผิดปกติ 3. ระบบเปลี่ยนการทำงานกลับไปขั้นตอนใน UC01 และ UC02 ซึ่งอยู่บน user interface เดียวกัน		

ตารางที่ 17 คำอธิบายยูสเคส Generate test cases และ SQL script

หมายเลขยูสเคส:	UR04	เวอร์ชัน:	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (Moderate)		
ผู้เกี่ยวข้อง:	ผู้ออกแบบคลังข้อมูลหรือนักทดสอบ		
รายละเอียดโดยย่อ:	อธิบายการสั่งงานของผู้ออกแบบให้ระบบสร้างกรณีทดสอบพร้อมกับสร้างคำสั่งทดสอบด้วยภาษาเอสคิวแอล		
เงื่อนไขขั้นต้น:	ข้อมูลที่นำมาสร้างกรณีทดสอบสามารถสกัดออกมาได้จากขั้นตอนใน UC03		
การทำงานโดยปกติ:	1. ระบบทำการอ่านข้อมูลเพื่อสร้างกรณีทดสอบจากฐานข้อมูล 2. ระบบนำข้อมูลที่อ่านจากฐานข้อมูลมาสร้างคำสั่งตามข้อกำหนดการสร้างกรณีทดสอบ 3. ระบบจัดเก็บคำสั่งทดสอบและกรณีทดสอบลงฐานข้อมูล		
ทางเลือกเพิ่มเติมในการทำงาน:	1. ระบบไม่สามารถบันทึกกรณีทดสอบเนื่องจากความ		

	ผิดปกติบางประการ 2. ระบบแจ้งเตือนให้ผู้ใช้ระบบตรวจสอบแก้ไขไฟล์และทำการทดสอบใหม่

ตารางที่ 18 คำอธิบายยูสเคส Display test case และ test script

หมายเลขยูสเคส:	UC05	เวอร์ชัน:	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (High)		
ผู้เกี่ยวข้อง:	-		
รายละเอียดโดยย่อ:	อธิบายการแสดงผลส่วนของกรณีทดสอบที่ได้จาก UC04		
เงื่อนไขขั้นต้น:	กรณีทดสอบและคำสั่งทดสอบสามารถบันทึกลงฐานข้อมูลอย่างถูกต้อง		
การทำงานโดยปกติ:	1. ระบบแสดงคำสั่งกรณีทดสอบและคำสั่งทดสอบเรียงตามลดับที่ระบบกำหนดไว้		
ทางเลือกเพิ่มเติมในการทำงาน:	-		

ตารางที่ 19 คำอธิบายยูสเคส Execute SQL script

หมายเลขยูสเคส:	UC06	เวอร์ชัน:	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (High)		
ผู้เกี่ยวข้อง:	ผู้ออกแบบคลังข้อมูลหรือนักทดสอบ		
รายละเอียดโดยย่อ:	อธิบายการสั่งงานของผู้ใช้สั่งให้ระบบนำคำสั่งทดสอบที่เป็นภาษาเอสคิวแอลเพื่อส่งไปดำเนินการหาค่าจากการรันคำสั่งดังกล่าวจากฐานข้อมูลที่เก็บตารางข้อมูลต้นทางที่นำมาสร้างคลังข้อมูล		
เงื่อนไขขั้นต้น:	คำสั่งในการทดสอบถูกจัดเก็บในฐานข้อมูลอย่างถูกต้องตามเงื่อนไข		
การทำงานโดยปกติ:	1. ผู้ใช้งานเลือกคำสั่งดำเนินการทดสอบ 2. ระบบส่งคำสั่งทดสอบเพื่อดำเนินการที่ฐานข้อมูล 3. ฐานข้อมูลดำเนินการส่งผลลัพธ์กลับมาที่ระบบ		

	4. ระบบจัดการเก็บผลการดำเนินการบันทึกไว้ในฐานข้อมูลเพื่อเปรียบเทียบกับผลการดำเนินการที่คาดหวังที่ได้จาก UC04
ทางเลือกเพิ่มเติมในการทำงาน:	1. ฐานข้อมูลไม่สามารถดำเนินการกับคำสั่งทดสอบเนื่องจากมีความผิดปกติเกิดขึ้นกับคำสั่งทดสอบ 2. ระบบส่งคำสั่งทดสอบไปยังกิจกรรมใน UC07 เพื่อทำการแก้ไขคำสั่ง

ตารางที่ 20 คำอธิบายยูสเคส Edit SQL script

หมายเลขยูสเคส:	UC07	เวอร์ชัน:	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (Moderate)		
ผู้เกี่ยวข้อง:	-		
รายละเอียดโดยย่อ:	อธิบายการรับคำสั่งทดสอบที่เกิดความผิดพลาดในการดำเนินการที่ฐานข้อมูลจาก UC06 เพื่อทำการช่วยแก้ไขและนำส่งไปดำเนินการที่ฐานข้อมูลอีกครั้ง		
เงื่อนไขขั้นต้น:	คำสั่งทดสอบที่ส่งไปดำเนินการที่ฐานข้อมูลและได้ข้อความระบุความผิดพลาดที่ไม่สามารถดำเนินการกับคำสั่งได้ถูกส่งกลับมาที่ระบบ		
การทำงานโดยปกติ:	1. ระบบทำการตรวจสอบข้อความระบุความผิดพลาด 2. ระบบทำการใช้การวัดระยะการแก้ไขเพื่อช่วยแก้ไขคำสั่งทดสอบในส่วนของชื่อตารางและชื่อคอลัมน์ 3. ระบบส่งคำสั่งหลังการแก้ไขไปดำเนินการที่ฐานข้อมูลอีกครั้ง		
ทางเลือกเพิ่มเติมในการทำงาน:	1. ระบบตรวจสอบข้อความระบุความผิดพลาดและไม่สามารถแยกประเภทของความผิดพลาดได้ 2. ระบบไม่ทำการแก้ไขคำสั่งทดสอบและไม่ส่งคำสั่งทดสอบไปดำเนินการที่ฐานข้อมูล 3. ระบบให้ผลไม่ผ่านสำหรับกรณีทดสอบของคำสั่งทดสอบดังกล่าว		

ตารางที่ 21 คำอธิบายยูสเคส Execute SQL script

หมายเลขยูสเคส:	UC08	เวอร์ชัน:	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (Moderate)		
ผู้เกี่ยวข้อง:	-		
รายละเอียดโดยย่อ:	อธิบายการนำคำสั่งทดสอบที่ผ่านการแก้ไขจาก UC07 นำไปดำเนินการที่ฐานข้อมูลอีกครั้งเพื่อหาค่าการดำเนินการจากคำสั่งดังกล่าวและนำไปเปรียบเทียบกับค่าดำเนินการที่คาดหวังจาก UC04		
เงื่อนไขขั้นต้น:	คำสั่งทดสอบจาก UC06 ได้รับการแก้ไขใน UC07		
การทำงานโดยปกติ:	1. ระบบส่งคำสั่งทดสอบที่ได้แก้ไขไปดำเนินการที่ฐานข้อมูล 2. ฐานข้อมูลดำเนินการกับคำสั่งทดสอบ 3. ฐานข้อมูลส่งผลจากการดำเนินการกลับมาที่ระบบ 4. ระบบเปรียบเทียบค่าจากการดำเนินการคำสั่งทดสอบ 5. ระบบให้ผลกรณีทดสอบและบันทึกผลกรณีทดสอบลงฐานข้อมูล		
ทางเลือกเพิ่มเติมในการทำงาน:	1. ระบบได้รับข้อความระบุความผิดพลาดจากการดำเนินการกับคำสั่งทดสอบ 2. ระบบให้ผลกรณีทดสอบเป็นไม่ผ่านและบันทึกผลกรณีทดสอบที่ฐานข้อมูล		

ตารางที่ 22 คำอธิบายยูสเคส Display result script test on user interface

หมายเลขยูสเคส:	UC09	เวอร์ชัน:	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (High)		
ผู้เกี่ยวข้อง:	-		
รายละเอียดโดยย่อ:	อธิบายการแสดงผลการทดสอบจากคำสั่งในกรณีทดสอบ		
เงื่อนไขขั้นต้น:	กรณีทดสอบได้ดำเนินการทดสอบและได้ให้ผลการทดสอบจาก UC08		

การทำงานโดยปกติ:	ระบบอ่านข้อมูลการทดสอบด้วยภาษาเอสคิวแอลจากฐานข้อมูลและแสดงผลการทดสอบผ่าน User interface
ทางเลือกเพิ่มเติมในการทำงาน:	-

ตารางที่ 23 คำอธิบายยูสเคส Display test case result detail on user interface

หมายเลขยูสเคส:	UC010	เวอร์ชัน:	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (Moderate)		
ผู้เกี่ยวข้อง:	ผู้ออกแบบคลังข้อมูลหรือนักทดสอบ		
รายละเอียดโดยย่อ:	อธิบายการแสดงรายละเอียดผลการทดสอบกรณีทดสอบแต่ละกรณีที่ผู้ใช้งานเรียกให้แสดง		
เงื่อนไขขั้นต้น:	กรณีทดสอบได้ดำเนินการทดสอบและได้ให้ผลการทดสอบจาก UC08		
การทำงานโดยปกติ:	1. ระบบได้รับคำสั่งให้แสดงรายละเอียดผลจากการทดสอบกรณีทดสอบใด ๆ 2. ระบบสืบค้นข้อมูลจากตารางกรณีทดสอบเพื่อนำมาแสดงผ่าน user interface		
ทางเลือกเพิ่มเติมในการทำงาน:	-		

4.4 การพัฒนาระบบ

4.4.1 สภาพแวดล้อมและเครื่องมือที่ใช้ในการพัฒนาระบบ

สภาพแวดล้อมที่ใช้ในการพัฒนาระบบมีสภาพแวดล้อมทางด้านฮาร์ดแวร์และซอฟแวร์ดังต่อไปนี้

ฮาร์ดแวร์

- หน่วยประมวลผลกลาง (CPU) อินเทล คอร์ไอห้า 2.4 กิกะเฮิรตซ์ (CPU intel Core i5 2.4 GHz)
- หน่วยความจำ (RAM) 4.00 Gb (2.93 Gb useable)
- จานบันทึกข้อมูล (Hard Disk) 500 กิกะไบต์ (Hard disk 500 GB)

ซอฟต์แวร์

- ไพทอน เวอร์ชัน 2.7.9 (Python version 2.7.9)
- โปรแกรมสำเร็จไพทอน (Python software package)
- โปรแกรมอีดิทพลัส เวอร์ชัน 3 (Editplus version 3)
- กรอบงานดีจังโก้ เวอร์ชัน 1.5.1 (Django framework version 1.5.1)
- ฐานข้อมูลออราเคิลเซิร์ฟเวอร์ เวอร์ชัน 10จี (Oracle database erver version 10g)
- โปรแกรมพีแอล เอสคิวแอล ดีเวลล็อปเปอร์ (PL SQL developer)
- ระบบปฏิบัติการ ไมโครซอฟต์วินโดว์เซเว่น โปรเฟสชั่นแนล 32 บิต (Microsoft Windows 7 Professional 32 bit)
- ไมโครซอฟต์ เอกซ์เซล 2010 (Microsoft Excel 2010)
- มอซิลลา ไฟร์ฟอกซ์ เว็บบราวเซอร์ (Mozilla Firefox web browser)

4.4.2 การติดตั้งซอฟต์แวร์ที่ใช้สำหรับการพัฒนาระบบ

ทำการติดตั้งเครื่องมือในการพัฒนาระบบทั้งหมดลงในเครื่องคอมพิวเตอร์ที่ใช้พัฒนาระบบโดยเริ่มลำดับการติดตั้งตามขั้นตอนดังต่อไปนี้

1. ติดตั้งระบบปฏิบัติการ ไมโครซอฟต์วินโดว์เซเว่น โปรเฟสชั่นแนล 32 บิต
2. ติดตั้งไมโครซอฟต์ เอกซ์เซล 2010
3. ติดตั้งไพทอน เวอร์ชัน 2.7.9
4. ติดตั้งโปรแกรมสำเร็จไพทอน
5. ติดตั้งโปรแกรมอีดิทพลัส
6. ติดตั้งกรอบงานดีจังโก้ เวอร์ชัน 1.5.1
7. ติดตั้งฐานข้อมูลออราเคิลเซิร์ฟเวอร์ เวอร์ชัน 10จี
8. ติดตั้งโปรแกรมพีแอล เอสคิวแอล ดีเวลล็อปเปอร์
9. ติดตั้งมอซิลลา ไฟร์ฟอกซ์ เว็บบราวเซอร์

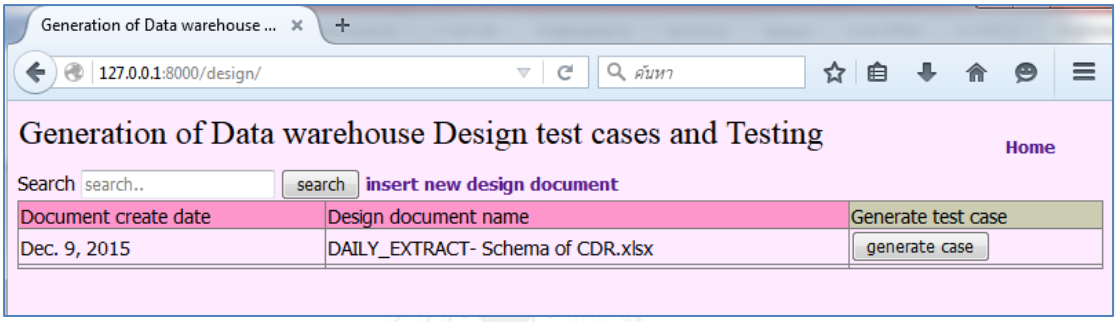
4.4.3 การพัฒนาส่วนต่อประสาน (User interface)

การพัฒนาส่วนต่อประสานเพื่อให้สอดคล้องกับข้อจำกัดในบทที่ 1 และมีการทำงานตรงกับความต้องการด้านหน้าที่ที่กำหนดไว้ในบทที่ 4 ในการพัฒนาส่วนต่อประสานพัฒนาด้วยโปรแกรมอีดิทพลัส (Editplus version 3) โดยใช้ภาษาในการพัฒนาเป็นภาษาไพทอน (Python programming

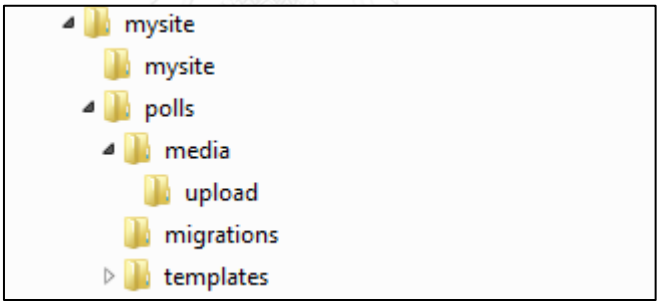
version 2.7)ร่วมกับกรอบงานดีจังโก้ (Django framework version 1.5.1) ในการจัดระเบียบการพัฒนาระบบและเป็นส่วนของเว็บเซิร์ฟเวอร์ที่สามารถแสดงผลบนเว็บบราวเซอร์

4.4.3.1 ส่วนของการนำเข้าไฟล์และแสดงผล

เมื่อทำการเปิดระบบสร้างกรณีทดสอบการออกแบบคลังข้อมูลด้วย URL http://127.0.0.1:8080/design/ ระบบจะแสดงหน้าจอหลักดังรูปที่22 ซึ่งเป็นส่วนหน้าจอหลักสำหรับนำเข้า เอกสารการออกแบบในรูปแบบเอกซ์เซลและแสดงคำสั่งสร้างกรณีทดสอบ สำหรับเอกสารนำเข้าจะแสดงชื่อเอกสารในแฟ้มเอกสารของเว็บเซิร์ฟเวอร์ตามรูปที่ 23 แสดงที่ D:\mysite\polls\media\upload



รูปที่ 22 ส่วนต่อประสานสำหรับนำเข้าเอกสารการออกแบบ



รูปที่ 23 ส่วนจัดเก็บเอกสารการออกแบบที่เว็บเซิร์ฟเวอร์

4.4.3.2 ส่วนของกรณีทดสอบ

เป็นส่วนของการจัดเก็บกรณีการทดสอบจะถูกจัดเก็บไว้ที่ฐานข้อมูลโดยทำการจัดเก็บผ่าน class model ของ Django framework ซึ่ง class ที่ใช้ในการจัดเก็บกรณีทดสอบแสดงดังรูปที่ 24

```
class TEDWCF_TESTCASE(models.Model) :
    doc_test  = models.CharField(max_length=30,default='', blank=True ,null=True)
    test_case_id =models.CharField(max_length=32, blank=True ,null=True)
    test_level = models.CharField(max_length=32, blank=True ,null=True)
    test_type  = models.CharField(max_length=32, blank=True ,null=True)
    test_desc =  models.CharField(max_length=500, blank=True ,null=True)
```

```
source_scr1 = models.CharField(max_length=2000, blank=True ,null=True)
esource_scr1 = models.CharField(max_length=2000, blank=True ,null=True)
source_scr2 = models.CharField(max_length=2000, blank=True ,null=True)
esource_scr2 = models.CharField(max_length=2000, blank=True ,null=True)
source_scr3 = models.CharField(max_length=2000, blank=True ,null=True)
esource_scr3 = models.CharField(max_length=2000, blank=True ,null=True)
source_scr4 = models.CharField(max_length=2000, blank=True ,null=True)
esource_scr4 = models.CharField(max_length=2000, blank=True ,null=True)
source_scr5 = models.CharField(max_length=2000, blank=True ,null=True)
esource_scr5 = models.CharField(max_length=2000, blank=True ,null=True)
target_scr  = models.CharField(max_length=2000, blank=True ,null=True)
exp_result = models.CharField(max_length=300, blank=True ,null=True)
actual_result = models.CharField(max_length=300, blank=True ,null=True)
ppn_tm = models.DateTimeField(auto_now =True)
```

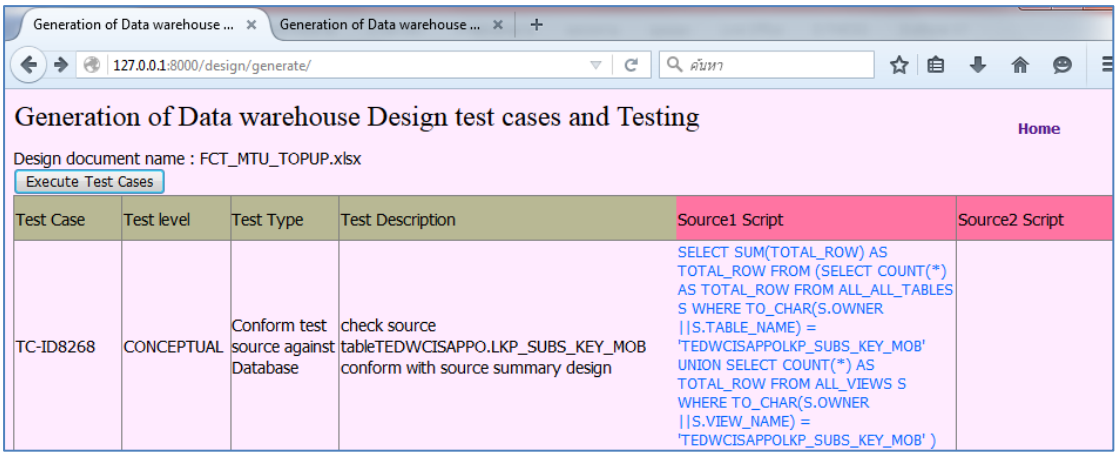
รูปที่ 24 คำสั่งส่วนการจัดเก็บกรณีทดสอบ

4.4.3.3 ส่วนของรายการคำสั่งเอสคิวแอลทดสอบ

เป็นส่วนที่แสดงรายการของคำสั่งเอสคิวแอลที่ใช้ทดสอบ โดยคำสั่งในการทดสอบจะถูกจัดเก็บไว้ที่ class model เดียวกับกรณีทดสอบ ซึ่งกรณีทดสอบและคำสั่งทดสอบที่ถูกจัดเก็บในฐานข้อมูลออราเคิลมีลักษณะดังรูปที่ 25 การดึงออกมาใช้จะใช้คำสั่งเอสคิวแอลในการดึงกรณีทดสอบและคำสั่งทดสอบแสดงดังรูปที่ 26

	ID	DOC_TEST	TEST_LEVEL	TEST_TYPE	SOURCE_SCR1	EXP_RESULT	ACTUAL_RESULT
1	3584	2015110122226ASEDFR	Logical	Join test	SELECT count(*) as transactions FROM cdr.cdrdb cdrdb inner jo	transactions > 0	
2	3585	2015110122226ASEDFR	Physical	Constraints test data type	SELECT datestartofcharging FROM cdr.cdrdb	date	
3	3586	2015110122226ASEDFR	Physical	Constraints test data type		varchar2(255)	
4	3581	2015110122226ASEDFR	Conceptual	Fact Test	SELECT COUNT(*) AS TOTAL_TRANSACTIONS FROM cdr.lkp	Data type : int	
5	3582	2015110122226ASEDFR	Conceptual	Fact Test	SELECT COUNT(*) AS TOTAL_TRANSACTIONS FROM cdr.lkp	Data type : int	
6	3583	2015110122226ASEDFR	Conceptual	Fact Test	SELECT COUNT(*) AS TOTAL_TRANSACTIONS FROM cdr.cd	Data type : int	
7	3587	2015110122226ASEDFR	Physical	Constraints test data type	SELECT imei FROM cdr.cdrdb	varchar2(16)	
8	3588	2015110122226ASEDFR	Physical	Constraints test data type	SELECT imsi FROM cdr.cdrdb	varchar2(16)	
9	3589	2015110122226ASEDFR	Physical	Constraints test data type	SELECT a_msisdn FROM cdr.cdrdb	varchar2(255)	
10	3590	2015110122226ASEDFR	Physical	Constraints test data type		number	
11	3591	2015110122226ASEDFR	Physical	Constraints test data type	SELECT b_msisdn FROM cdr.cdrdb	varchar2(255)	
12	3592	2015110122226ASEDFR	Physical	Constraints test data type	SELECT tod_key FROM cdr.lkp_dow_hod	number	
13	3593	2015110122226ASEDFR	Physical	Constraints test data type		number	
14	3597	2015110122226ASEDFR	Physical	Constraints test data type		number	
15	3598	2015110122226ASEDFR	Physical	Constraints test data type	SELECT chargeableduration FROM cdr.cdrdb	number	

รูปที่ 25 ตัวอย่างกรณีทดสอบและคำสั่งทดสอบในฐานข้อมูล



รูปที่ 26 ส่วนต่อประสานแสดงกรณีทดสอบและคำสั่งทดสอบ

สำหรับกรณีทดสอบที่ถูกสร้างขึ้นโดยมีการระบุระดับการทดสอบ (Test level) และ รูปแบบการทดสอบ คำอธิบายการทดสอบ คำสั่งทดสอบ ซึ่งผู้วิจัยได้เขียนคำสั่งในภาษาไพทอนเพื่อสร้างกรณีทดสอบในระดับ conceptual ตามตัวอย่างดังรูปที่ 27 คำสั่งสร้างกรณีทดสอบในระดับ logical ดังรูปที่ 28 และคำสั่งสร้างกรณีทดสอบในระดับ physical ดังรูปที่ 29

```
def conform_test(request) :
   doc_key ='201511012226ASEDFR'
   test_level  ='Conceptual'
   if doc_key :
      tar_list1 = my_custom_sql("select distinct S1_SCHEMA,S1_TABLE from
polls_TEDWCF_TARGET  s  where doc_tar = '"+doc_key+"' )
      tar_list2 = my_custom_sql("select distinct S2_SCHEMA,S2_TABLE from
polls_TEDWCF_TARGET  s  where doc_tar = '"+doc_key+"')
      tar_list3 = my_custom_sql("select distinct S3_SCHEMA,S3_TABLE from
polls_TEDWCF_TARGET  s  where doc_tar = '"+doc_key+"' )
      tar_list4 = my_custom_sql("select distinct S4_SCHEMA,S4_TABLE from
polls_TEDWCF_TARGET  s  where doc_tar = '"+doc_key+"' )
      tar_list5 = my_custom_sql("select distinct S5_SCHEMA,S5_TABLE from
polls_TEDWCF_TARGET  s  where doc_tar = '"+doc_key+"' )
      if tar_list1 :
         for d in tar_list1 :
            if d['S1_SCHEMA'] and d['S1_TABLE']:
               SQL_COMMAND ="SELECT SCHEMA_NM,TABLE_NM FROM
```

```
POLLS_TEDWCF_SRCTABLE WHERE SCHEMA_NM='"+d['S1_SCHEMA']+"' AND TABLE_NM=
'"+d['S1_TABLE']+"'"
            TEDWCF_TESTCASE( doc_test = doc_key
            ,test_level = test_level
            ,test_type  ="Conform test"
            ,test_desc  ="check source table conform with target table design from
"+d['S1_SCHEMA']+"."+d['S1_TABLE']+" table"
            ,source_scr1 = SQL_COMMAND
            ,exp_result1 = d['S1_SCHEMA']+"."+d['S1_TABLE']+" exist in source table design "
            ).save()
    if tar_list2 :
      for d in tar_list2 :
        if d['S2_SCHEMA'] and d['S2_TABLE']:
            SQL_COMMAND ="SELECT SCHEMA_NM,TABLE_NM FROM
POLLS_TEDWCF_SRCTABLE WHERE SCHEMA_NM='"+d['S2_SCHEMA']+"' AND TABLE_NM=
'"+d['S2_TABLE']+"'"
            TEDWCF_TESTCASE( doc_test = doc_key
            ,test_level = test_level
            ,test_type  ="Conform test"
            ,test_desc  ="check source table conform with target table design from
"+d['S2_SCHEMA']+"."+d['S2_TABLE']+" table"
            ,source_scr1 = SQL_COMMAND
            ,exp_result1 = d['S2_SCHEMA']+"."+d['S2_TABLE']+" exist in source table design "
            ).save()
    if tar_list3 :
      for d in tar_list3 :
        if d['S3_SCHEMA'] and d['S3_TABLE']:
            SQL_COMMAND ="SELECT SCHEMA_NM,TABLE_NM FROM
POLLS_TEDWCF_SRCTABLE WHERE SCHEMA_NM='"+d['S3_SCHEMA']+"' AND TABLE_NM=
'"+d['S3_TABLE']+"'"
            TEDWCF_TESTCASE( doc_test = doc_key
            ,test_level = test_level
            ,test_type  ="Conform test"
            ,test_desc  ="check source table conform with target table design from
"+d['S3_SCHEMA']+"."+d['S3_TABLE']+" table"
```

```
                ,source_scr1 = SQL_COMMAND
                ,exp_result1 = d['S3_SCHEMA']+"."+d['S3_TABLE']+" exist in source table design "
                ).save()
        if tar_list4 :
            for d in tar_list4 :
                if d['S4_SCHEMA'] and d['S4_TABLE']:
                    SQL_COMMAND ="SELECT SCHEMA_NM,TABLE_NM FROM
POLLS_TEDWCF_SRCTABLE WHERE SCHEMA_NM='"+d['S4_SCHEMA']+"' AND TABLE_NM=
'"+d['S4_TABLE']+"'"
                    TEDWCF_TESTCASE( doc_test = doc_key
                    ,test_level = test_level
                    ,test_type  ="Conform test"
                    ,test_desc  ="check source table conform with target table design from
"+d['S4_SCHEMA']+"."+d['S4_TABLE']+" table"
                    ,source_scr1 = SQL_COMMAND
                    ,exp_result1 = d['S4_SCHEMA']+"."+d['S4_TABLE']+" exist in source table design "
                    ).save()
        if tar_list5 :
            for d in tar_list5 :
                if d['S5_SCHEMA'] and d['S5_TABLE']:
                    SQL_COMMAND ="SELECT SCHEMA_NM,TABLE_NM FROM
POLLS_TEDWCF_SRCTABLE WHERE SCHEMA_NM='"+d['S5_SCHEMA']+"' AND TABLE_NM=
'"+d['S5_TABLE']+"'"
                    TEDWCF_TESTCASE( doc_test = doc_key
                    ,test_level = test_level
                    ,test_type  ="Conform test"
                    ,test_desc  ="check source table conform with target table design from
"+d['S5_SCHEMA']+"."+d['S5_TABLE']+" table"
                    ,source_scr1 = SQL_COMMAND
                    ,exp_result1 = d['S6_SCHEMA']+"."+d['S6_TABLE']+" exist in source table design "
                    ).save()
            context ={'testcase' : SQL_COMMAND}
```

รูปที่ 27 ตัวอย่างคำสั่งสร้างกรณีทดสอบในระดับ conceptual

```
def join_test(doc_key, test_level) :
  if doc_key  and test_level :
```

```
    test_type = 'Join test'
    exp_result ='transactions > 0'
    logical_data =
TEDWCF_RELATION.objects.filter(doc_ral__contains=doc_key).order_by().values('tablea').distinct()
    for detail in logical_data :
      SchemaA =''
      TBA = detail.get('tablea')
      Schema= TEDWCF_SRCTABLE.objects.filter(doc_src=doc_key ,
table_nm=TBA).values('schema_nm')[:1]
      if Schema :
        for  A in Schema :
          SchemaA = A.get('schema_nm')
      TBA_data = TEDWCF_RELATION.objects.filter(doc_ral=doc_key ,
tablea=TBA).order_by('tableb').values()
      source_scr1 = 'SELECT count(*) as transactions FROM '+SchemaA+'.'+detail.get('tablea')+'
'+ detail.get('tablea')+ ' '
      test_desc = 'Join test of table '+ detail.get('tablea')+' '
      for TBA_detail in TBA_data :
        TBB = TBA_detail.get('tableb')
        SchemaB =''
        Schema= TEDWCF_SRCTABLE.objects.filter(doc_src=doc_key ,
table_nm=TBB).values('schema_nm')[:1]
        if Schema :
          for  B in Schema :
            SchemaB = B.get('schema_nm')
        source_scr1= source_scr1 + TBA_detail.get('jointype')+' '+SchemaB+'.'+
TBA_detail.get('tableb')+' '+ TBA_detail.get('tableb')+' on '+ TBA_detail.get('joincond')+ ' '
        test_desc =  test_desc + ','+ TBA_detail.get('tableb')+ ' '
      TEDWCF_TESTCASE(doc_test = doc_key
                      ,test_level = test_level
                      ,test_type  = test_type
                      ,test_desc = test_desc
                      ,source_scr1 = source_scr1
                      ,exp_result = exp_result
                     ).save()
```

```
    context = {'source_scr1':TBA_data}
  return  context
```

รูปที่ 28 ตัวอย่างคำสั่งสร้างกรณีทดสอบในระดับ logical

```
def test_pk(data_detail,doc_key, test_level):
  suss_ind ='0'
  if data_detail :
    for detail  in data_detail :
            doc_test = doc_key
            test_type =  'Constraints test primary key '
            test_desc =  'Test primary key of target column: ' +  detail.get('column_nm')
            target_scr  = ''
            if  detail.get('pk_key') !='' :
                source_scr1 =  "SELECT "+ detail.get('s1_column')+" FROM "  +
detail.get('s1_schema')+"."+ detail.get('s1_table')
                source_scr2 =  "SELECT "+ detail.get('s2_column')+" FROM "  +
detail.get('s2_schema')+"."+ detail.get('s2_table')
                source_scr3 =  "SELECT "+ detail.get('s3_column')+" FROM "  +
detail.get('s3_schema')+"."+ detail.get('s3_table')
                source_scr4 =  "SELECT "+ detail.get('s4_column')+" FROM "  +
detail.get('s4_schema')+"."+ detail.get('s4_table')
                source_scr5 =  "SELECT "+ detail.get('s5_column')+" FROM "  +
detail.get('s5_schema')+"."+ detail.get('s5_table')
            else :
                source_scr1 =  ""
                source_scr2 =  ""
                source_scr3 =  ""
                source_scr4 =  ""
                source_scr5 =  ""
            exp_result  = 'not return null data'
            if ( source_scr1 != ''  or source_scr2 != '' or   source_scr3 != '' or  source_scr4 != '' or
source_scr5 != '' ) :
              TEDWCF_TESTCASE(doc_test =doc_test
                          ,test_level =test_level
                          ,test_type  =test_type
                          ,test_desc=test_desc
```

```
                        ,source_scr1 =source_scr1
                        ,source_scr2 =source_scr2
                        ,source_scr3 =source_scr3
                        ,source_scr4 =source_scr4
                        ,source_scr5 =source_scr5
                        ,target_scr =target_scr
                        ,exp_result =exp_result
                         ).save()


            suss_ind ='1'
return  suss_ind
```

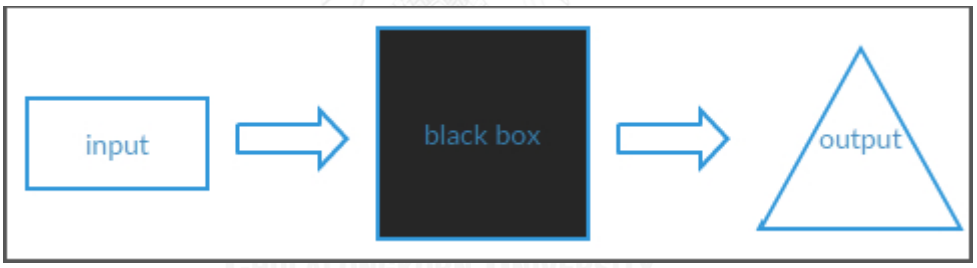
รูปที่ 29 ตัวอย่างคำสั่งสร้างกรณีทดสอบในระดับ physical

บทที่ 5
การทดสอบและประเมินผลระบบ

รายละเอียดในบทนี้จะอธิบายเกี่ยวกับการทดสอบและการประเมินระบบต้นแบบเพื่อสนับสนุนแนวทางในการสร้างกรณีทดสอบเอกสารการออกแบบตามที่ได้ออกแบบและพัฒนาในบทที่ 4 โดยเนื้อหาในบทนี้จะประกอบด้วย การทดสอบระบบ การประเมินผลระบบ ตั้งแต่การเข้าเข้าเอกสารจนถึงการแสดงผลรายละเอียดของกรณีทดสอบหลังจากที่ทำการทดสอบเสร็จ โดยการประเมินผลจะใช้เอกสารการออกแบบที่ให้กรณีทดสอบอย่างน้อย 100 กรณีเพื่อยืนยันความถูกต้องของการทำงานของระบบ

5.1 การทดสอบระบบ

การทดสอบระบบสร้างกรณีทดสอบการออกแบบคลังข้อมูลได้ออกแบบการทดสอบแบบมองว่าระบบเป็นกล่องดำ (Black Box Testing) โดยที่ไม่สนใจว่าระบบจะทำอย่างไรกับข้อมูลนำเข้าเพื่อให้ได้ข้อมูลนำออกที่ถูกต้องเท่านั้นว่าถูกต้องสอดคล้องกับความต้องการเชิงหน้าที่ที่ได้ระบุไว้ในเอกสารบทที่ 4 ดังรูปที่ 30 ซึ่งมีกรณีทดสอบและผลการทดสอบดังต่อไปนี้



รูปที่ 30 รูปแบบการทดสอบแบบกล่องดำ

5.1.1 การทดสอบการนำเข้าไฟล์เอกสารการออกแบบในรูปแบบเอกซ์เซล

1. ทดสอบการนำเข้าเอกสารการออกแบบ
2. ทดสอบตรวจสอบการนำเข้าเอกสารที่มีค่าว่างเปล่า
3. ทดสอบการนำเข้าเอกสารที่มีแผ่นงานที่ไม่ตรงตามความต้องการของระบบ
4. ทดสอบการนำเข้าเอกสารที่มีแผ่นงานที่ตรงตามความต้องการของระบบ
5. ทดสอบการนำเข้าแผ่นเอกสารส่วนข้อมูลตารางต้นทางเป็นค่าว่าง
6. ทดสอบการนำเข้าแผ่นเอกสารส่วนข้อมูลความสัมพันธ์ของตารางต้นทางเป็นค่าว่าง
7. ทดสอบการนำเข้าแผ่นเอกสารข้อมูลตารางปลายทางที่สัมพันธ์กับตารางข้อมูลต้นทางเป็นค่าว่าง
8. ทดสอบค่า tabletype ส่วนข้อมูลตารางต้นทางไม่เป็นไปตามกำหนด

9. ทดสอบค่า select_method ส่วนข้อมูลตารางต้นทางไม่เป็นไปตามกำหนด
10. ทดสอบค่า jointype ส่วนข้อมูลความสัมพันธ์ของตารางต้นทางไม่เป็นไปตามกำหนด
11. ทดสอบค่า datatype ส่วนข้อมูลตารางปลายทางที่สัมพันธ์กับตารางข้อมูลต้นทางไม่เป็นไปตามกำหนด

5.1.2 ทดสอบการค้นหาเอกสารการออกแบบ

1. ทดสอบการใช้คำค้นที่เป็นค่าว่าง
2. ทดสอบการใช้คำค้นที่เป็นตัวพิมพ์ใหญ่หรือตัวพิมพ์เล็กทั้งหมด
3. ทดสอบการใช้คำค้นที่ตรงกับชื่อเอกสารใด ๆ
4. ทดสอบการใช้คำค้นที่ไม่ตรงกับชื่อเอกสารใด ๆ

5.1.3 ทดสอบการสร้างกรณีทดสอบ

1. ทดสอบการสร้างคำสั่งทดสอบเพื่อทดสอบชนิดข้อมูล
2. ทดสอบการสร้างคำสั่งทดสอบแบบ Aggregate function
3. ทดสอบการสร้างคำสั่งทดสอบเมื่อตารางตั้งต้นมีความสัมพันธ์แบบ one to one
4. ทดสอบการสร้างคำสั่งทดสอบเมื่อตารางตั้งต้นมีความสัมพันธ์แบบ one to many
5. ทดสอบการสร้างกรณีทดสอบประเภท transform test แบบคำสั่ง case when
6. ทดสอบการสร้างกรณีทดสอบประเภท transform test แบบคำสั่ง if-else

5.1.4 ทดสอบการส่งคำสั่งไปดำเนินการที่ฐานข้อมูล

1. ทดสอบการส่งคำสั่งทดสอบที่ไม่สามารถดำเนินการได้
2. ทดสอบการส่งคำสั่งทดสอบที่ไม่สามารถดำเนินการได้
3. ทดสอบการปรับปรุงค่า test script result เมื่อคำสั่งทดสอบไม่สามารถใช้ดำเนินการได้
4. ทดสอบการปรับปรุงค่า test script result เมื่อคำสั่งทดสอบสามารถดำเนินการได้

5.1.5 ทดสอบการแก้ไขคำสั่งทดสอบ

1. ทดสอบการแยกประเภทข้อความแสดงข้อผิดพลาดของชื่อตาราง
2. ทดสอบการแยกประเภทข้อความแสดงข้อผิดพลาดของชื่อคอลัมน์
3. ทดสอบการแก้ไขคำสั่งทดสอบส่วนชื่อตาราง
4. ทดสอบการแก้ไขคำสั่งทดสอบส่วนชื่อคอลัมน์
5. ทดสอบการปรับปรุงค่า คำสั่งทดสอบหลังการแก้ไขในตารางกรณีทดสอบ

5.1.6 ทดสอบการปรับปรุงค่า Actual result

1. ทดสอบการปรับปรุงค่า Actual เมื่อมีข้อความแสดงข้อผิดพลาดของคำสั่งทดสอบ

2. ทดสอบการปรับปรุงค่า Actual เมื่อเมื่อคำสั่งทดสอบสามารถดำเนินการได้

5.1.7 ทดสอบเปรียบเทียบค่า Actual result และ Expected result ของกรณีทดสอบ

3. ทดสอบกรณีค่า Actual result เป็นค่าว่างเปล่า
4. ทดสอบกรณีค่า Actual result เท่ากับค่า Expected result
5. ทดสอบกรณีค่า Actual result ไม่เท่ากับค่า Expected result

5.1.8 ทดสอบการแสดงผลการทดสอบกรณีทดสอบ

5.1.9 ทดสอบการแสดงรายละเอียดผลการทดสอบกรณีทดสอบ

ตารางที่ 24 ทดสอบการนำเข้าเอกสารการออกแบบ

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC01	การนำเข้าเอกสารที่มีค่าว่างเปล่า	นำเข้าเอกสารการออกแบบคลังข้อมูลที่ไม่มีข้อมูลอยู่ภายใน	ระบบแสดงข้อความเตือน ‘ขออภัย!กรุณาเลือกเอกสารที่มีข้อมูลการออกแบบ’	ถูกต้อง
			นำชื่อไฟล์เอกสารออกมาแสดงที่หน้าจอ user interface ได้ถูกต้อง	ถูกต้อง

ตารางที่ 25 ทดสอบตรวจสอบการนำเข้าเอกสารที่มีค่าว่างเปล่า

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC02	การนำเข้าเอกสารที่มีค่าว่างเปล่า	นำเข้าเอกสารการออกแบบคลังข้อมูลที่ไม่มีข้อมูลอยู่ภายใน	การอ่านเอกสารการออกแบบได้ตรงตามที่ผู้ใช้สั่งงาน	ถูกต้อง
			ระบบแสดงข้อความเตือน ‘ขออภัย! กรุณาเลือกเอกสารที่มีข้อมูลการออกแบบ’	ถูกต้อง
			ระบบสามารถทำงานต่อได้	ถูกต้อง

ตารางที่ 26 ทดสอบการนำเข้าจำนวนแผ่นเอกสารในเอกสารการออกแบบ

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC03	ทดสอบการนำเข้าเอกสารที่มีแผ่นงานที่ไม่ตรงตามความต้องการของระบบ	นำเข้าเอกสารที่มีแผ่นงานไม่ตรงกับความต้องการของระบบ	ระบบแจ้งเตือน ‘แผ่นงานภายในเอกสารมีรูปแบบไม่เหมาะสมต่อการทดสอบ’	ถูกต้อง
			ระบบสามารถงานต่อได้	ถูกต้อง
TC04	ทดสอบการนำเข้าเอกสารที่มีแผ่นงานตรงตามความต้องการของระบบ	นำเข้าเอกสารที่มีแผ่นงานตรงตามความต้องการของระบบ	ระบบนำเข้าเอกสารและจัดเก็บข้อมูลเอกสารในฐานข้อมูล	ถูกต้อง
			ระบบแสดงผลการนำเข้าเอกสารได้ถูกต้อง	ถูกต้อง

ตารางที่ 27 ทดสอบการนำเข้าแผ่นเอกสารส่วนข้อมูลตารางต้นทางเป็นค่าว่าง

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC05	ทดสอบการนำเข้าแผ่นเอกสารส่วนข้อมูลตารางต้นทางเป็นค่าว่าง	นำเข้าเอกสารที่แผ่นเอกสารส่วนของข้อมูลตารางต้นทางเป็นค่าว่าง	ระบบมีข้อความแจ้งเตือน ‘กรุณานำเข้าเอกสารที่มีข้อมูลตารางต้น’	ถูกต้อง
			ระบบสามารถทำงานต่อได้	ถูกต้อง

ตารางที่ 28 ทดสอบการนำเข้าแผ่นเอกสารส่วนข้อมูลความสัมพันธ์ของตารางต้นทางเป็นค่าว่าง

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC06	ทดสอบการนำเข้าแผ่นเอกสารส่วนข้อมูลความสัมพันธ์ของตารางต้นทางเป็นค่าว่าง	นำเข้าแผ่นเอกสารส่วนข้อมูลความสัมพันธ์ของตารางต้นทางเป็นค่าว่าง	ระบบมีข้อความแจ้งเตือน ‘กรุณานำเข้าเอกสารที่มีข้อมูลความสัมพันธ์ของตาราง’	ถูกต้อง
			ระบบสามารถทำงานต่อได้	ถูกต้อง

ตารางที่ 29 ทดสอบการนำเข้าแผ่นเอกสารข้อมูลตารางปลายทางที่สัมพันธ์กับตารางข้อมูลต้นทางเป็นค่าว่าง

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC07	ทดสอบการนำเข้าแผ่นเอกสารข้อมูลตารางปลายทางที่สัมพันธ์กับตารางข้อมูลต้นทางเป็นค่าว่าง	นำเข้าแผ่นเอกสารข้อมูลตารางปลายทางที่สัมพันธ์กับตารางข้อมูลต้นทางเป็นค่าว่าง	ระบบมีข้อความแจ้งเตือน ‘กรุณานำเข้าเอกสารที่มีข้อมูลตารางปลายทาง’	ถูกต้อง
			ระบบสามารถทำงานต่อได้	ถูกต้อง

ตารางที่ 30 ทดสอบค่า table type ส่วนข้อมูลตารางต้นทางไม่เป็นไปตามกำหนด

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC08	ทดสอบค่า table type ส่วนข้อมูลตารางต้นทางไม่เป็นไปตามรูปแบบที่กำหนด	นำเข้าเอกสารข้อมูลที่ค่า table type ในส่วนข้อมูลตารางต้นทางไม่เป็นไปตามรูปแบบที่กำหนด	ระบบแสดงข้อความแจ้งเตือน ‘มีผิดพลาดของข้อมูล table type ในส่วนข้อมูลตารางต้นทาง’	ถูกต้อง
			ระบบสามารถทำงานต่อได้	ถูกต้อง

ตารางที่ 31 ทดสอบค่า select method ส่วนข้อมูลตารางต้นทางไม่เป็นไปตามกำหนด

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC09	ทดสอบการนำเข้าเอกสารการออกแบบที่มีค่าselect method ส่วนข้อมูลตารางต้นทางไม่เป็นไปตามกำหนด	นำเข้าเอกสารการออกแบบที่มีค่าselect method ส่วนข้อมูลตารางต้นทางไม่เป็นไปตามกำหนด	ระบบแสดงข้อความแจ้งเตือน ‘มีผิดพลาดของข้อมูล select method ในส่วนข้อมูลตารางต้นทาง’	ถูกต้อง
			ระบบสามารถทำงานต่อได้	ถูกต้อง

ตารางที่ 32 ทดสอบค่า join type ส่วนข้อมูลความสัมพันธ์ของตารางต้นทางไม่เป็นไปตามกำหนด

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC10	ทดสอบการนำเข้าเอกสารการออกแบบที่มีค่า join type ส่วนข้อมูลความสัมพันธ์ของตารางต้นทางไม่เป็นไปตามกำหนด	นำเข้าเอกสารการออกแบบที่มีค่า join type ส่วนข้อมูลความสัมพันธ์ของตารางต้นทางไม่เป็นไปตามกำหนด	ระบบแสดงข้อความแจ้งเตือน ‘มีผิดพลาดของข้อมูล join type ในส่วนข้อมูลความสัมพันธ์ตาราง’	ถูกต้อง
			ระบบสามารถทำงานต่อได้	ถูกต้อง

ตารางที่ 33 ทดสอบค่า data type ส่วนข้อมูลตารางปลายทางที่สัมพันธ์กับตารางข้อมูลต้นทางไม่เป็นไปตามกำหนด

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC11	ทดสอบการนำเข้าเอกสารการออกแบบที่มีค่า data type ส่วนข้อมูลตารางปลายทางที่สัมพันธ์กับตารางข้อมูลต้นทางไม่เป็นไปตามกำหนด	นำเข้าเอกสารการออกแบบที่มีค่า data type ส่วนข้อมูลตารางปลายทางที่สัมพันธ์กับตารางข้อมูลต้นทางไม่เป็นไปตามกำหนด	ระบบแสดงข้อความแจ้งเตือน ‘มีผิดพลาดของข้อมูล data type ในส่วนข้อมูลความสัมพันธ์ตาราง’	ถูกต้อง
			ระบบสามารถทำงานต่อได้	ถูกต้อง

ตารางที่ 34 ทดสอบการใช้คำค้นที่เป็นค่าว่าง

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC12	ทดสอบการค้นหาเอกสารการออกแบบโดยการใช้คำค้นที่เป็นค่าว่าง	ค้นหาเอกสารการออกแบบโดยการใช้คำค้นที่เป็นค่าว่าง	หน้าจอแสดงผลเอกสารทั้งหมดโดยไม่มีการเลือกเอกสารใด ๆ	ถูกต้อง

ตารางที่ 35 ทดสอบการใช้คำค้นที่เป็นตัวพิมพ์ใหญ่หรือตัวพิมพ์เล็กทั้งหมด

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC13	ทดสอบการค้นหาเอกสารการออกแบบโดยการใช้คำค้นที่เป็นตัวพิมพ์ใหญ่ทั้งหมด	ค้นหาเอกสารการออกแบบโดยการใช้คำค้นที่เป็นตัวพิมพ์ใหญ่ทั้งหมด	ระบบแสดงเอกสารที่มีคำค้นเป็นส่วนประกอบของชื่อแม้ว่าชื่อเอกสารจะเป็นตัวพิมพ์เล็ก	ถูกต้อง
	ทดสอบการค้นหาเอกสารการออกแบบโดยการใช้คำค้นที่เป็นตัวพิมพ์เล็กทั้งหมด	ค้นหาเอกสารการออกแบบโดยการใช้คำค้นที่เป็นตัวพิมพ์เล็กทั้งหมด	ระบบแสดงเอกสารที่มีคำค้นเป็นส่วนประกอบของชื่อแม้ว่าชื่อเอกสารจะเป็นตัวพิมพ์เล็ก	ถูกต้อง

ตารางที่ 36 ทดสอบการใช้คำค้นที่ตรงกับชื่อเอกสารใด ๆ

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC14	ทดสอบการค้นหาเอกสารโดยใช้คำค้นที่ตรงกับชื่อเอกสารใด ๆ	ค้นหาเอกสารโดยใช้คำค้นที่ตรงกับชื่อเอกสารใด ๆ	ระบบแสดงเอกสารที่มีคำค้นเป็นส่วนประกอบของชื่อได้ถูกต้อง	ถูกต้อง

ตารางที่ 37 ทดสอบการใช้คำค้นที่ไม่ตรงกับชื่อเอกสารใด ๆ

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC15	ทดสอบการค้นหาเอกสารโดยใช้คำค้นที่ไม่ตรงกับชื่อเอกสารใด ๆ	ค้นหาเอกสารโดยใช้คำค้นที่ไม่ตรงกับชื่อเอกสารใด ๆที่จัดเก็บในระบบ	ระบบมีข้อความแจ้งเตือนไม่พบเอกสารที่ตรงกับคำค้น	ถูกต้อง
			ระบบสามารถทำงานต่อได้	ถูกต้อง

ตารางที่ 38 ทดสอบการสร้างคำสั่งทดสอบเพื่อทดสอบชนิดข้อมูล

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC16	ทดสอบการสร้างคำสั่งทดสอบเพื่อทดสอบชนิดข้อมูล	ระบบสร้างคำสั่งทดสอบเพื่อทดสอบชนิดข้อมูลในส่วนของข้อมูลตารางปลายทางที่สัมพันธ์กับตารางข้อมูลต้นทาง	คำสั่งทดสอบมีความถูกต้องตามที่ระบบกำหนด	ถูกต้อง

ตารางที่ 39 ทดสอบการสร้างคำสั่งทดสอบแบบ Aggregate function

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC17	ทดสอบการสร้างคำสั่งทดสอบแบบ Aggregate function สำหรับข้อมูลในส่วนของข้อมูลตารางต้นทาง	ระบบสร้างคำสั่งทดสอบแบบ Aggregate function สำหรับข้อมูลในส่วนของข้อมูลตารางต้นทาง	คำสั่งทดสอบมีความถูกต้องตามที่ระบบกำหนด	ถูกต้อง
TC18	ทดสอบการสร้างคำสั่งทดสอบแบบ Aggregate function สำหรับข้อมูลความสัมพันธ์ของตารางต้นทาง	ระบบสร้างคำสั่งทดสอบแบบ Aggregate function สำหรับข้อมูลความสัมพันธ์ของตารางต้นทาง	คำสั่งทดสอบมีความถูกต้องตามที่ระบบกำหนด	ถูกต้อง

ตารางที่ 40 ทดสอบการสร้างคำสั่งทดสอบเมื่อตารางตั้งต้นมีความสัมพันธ์แบบ one to one

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC19	ทดสอบการสร้างคำสั่งทดสอบเมื่อตารางตั้งต้นมีความสัมพันธ์แบบ one to one	ระบบสร้างคำสั่งทดสอบแบบ join test เมื่อตารางตั้งต้นและตารางเสริมมีความสัมพันธ์แบบ one to one	คำสั่งทดสอบมีลักษณะ join กัน จำนวน 2 ตาราง	ถูกต้อง

ตารางที่ 41 ทดสอบการสร้างคำสั่งทดสอบเมื่อตารางตั้งต้นมีความสัมพันธ์แบบ one to many

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC20	ทดสอบการสร้างคำสั่ง	ระบบสร้างคำสั่ง	คำสั่งทดสอบมี	ถูกต้อง

	ทดสอบเมื่อตารางตั้งต้นมีความสัมพันธ์แบบ one to many	ทดสอบแบบ join test เมื่อตารางตั้งต้นและตาราวเสริมมีความสัมพันธ์แบบ one to many	ลักษณะ join กัน เท่ากับจำนวนตารางเสริมบวกจำนวนตารางหลักตาราง	

ตารางที่ 42 ทดสอบการสร้างกรณีทดสอบประเภท transform test แบบคำสั่ง case when

หมายเลขกรณีทดสอบ	**การทดสอบ**	**คำอธิบาย**	**ผลการทดสอบที่คาดหวัง**	**ผลการทดสอบ**
TC21	ทดสอบการสร้างกรณีทดสอบประเภท transform test แบบคำสั่ง case when	ระบบสร้างคำสั่งทดสอบในส่วนของ transform test จากข้อมูลในเอกสารการออกแบบมีการใช้คำสั่ง case when ใน transformation rule	ระบบสามารถสร้างคำสั่งได้ถูกต้องตามที่ระบบกำหนด	ถูกต้อง
			ระบบสามารถสกัดค่าที่คาดหวังของการทดสอบจากคำสั่ง case when ได้ถูกต้อง	ถูกต้อง

ตารางที่ 43 ทดสอบการสร้างกรณีทดสอบประเภท transform test แบบคำสั่ง if-else

หมายเลขกรณีทดสอบ	**การทดสอบ**	**คำอธิบาย**	**ผลการทดสอบที่คาดหวัง**	**ผลการทดสอบ**
TC22	ทดสอบการสร้างกรณีทดสอบประเภท transform test แบบคำสั่ง if-else	ระบบสร้างคำสั่งทดสอบในส่วนของ transform test จากข้อมูลในเอกสารการออกแบบมีการใช้คำสั่ง if-else ใน transformation	ระบบสามารถสร้างคำสั่งได้ถูกต้องตามที่ระบบกำหนด	ถูกต้อง
			ระบบสามารถสกัดค่าที่คาดหวังของการทดสอบจาก	ถูกต้อง

		rule	คำสั่ง if-else ได้ถูกต้อง	

ตารางที่ 44 ทดสอบการส่งคำสั่งทดสอบที่ไม่สามารถดำเนินการได้ไปยังฐานข้อมูล

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC23	ทดสอบการส่งคำสั่งทดสอบที่ไม่สามารถดำเนินการได้ไปดำเนินการที่ฐานข้อมูล	ส่งคำสั่งทดสอบที่ไม่สามารถดำเนินการได้ไปดำเนินการที่ฐานข้อมูล	ระบบได้รับข้อความแสดงความผิดพลาดเกี่ยวกับคำสั่งทดสอบกลับมาจากฐานข้อมูล	ถูกต้อง
			ระบบระบุความผิดปกติของคำสั่งทดสอบว่าเป็นความผิดปกติจากชื่อตารางหรือชื่อคอลัมน์ได้ถูกต้อง	ถูกต้อง
			ระบบสามารถทำงานต่อได้	ถูกต้อง

ตารางที่ 45 ทดสอบการส่งคำสั่งทดสอบที่สามารถดำเนินการได้ไปยังฐานข้อมูล

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC24	ทดสอบการส่งคำสั่งทดสอบที่สามารถดำเนินการได้เพื่อดำเนินการหาผลลัพธ์ที่ฐานข้อมูล	ส่งคำสั่งทดสอบที่สามารถดำเนินการได้เพื่อดำเนินการหาผลลัพธ์ที่ฐานข้อมูล	ระบบได้รับผลลัพธ์จากการดำเนินการที่ฐานข้อมูลถูกต้อง	ถูกต้อง
			ระบบทำการ update ค่า actual result ได้ถูกต้องตามกรณีทดสอบ	ถูกต้อง
			ระบบสามารถทำงานต่อได้	ถูกต้อง

ตารางที่ 46 ทดสอบการปรับปรุงค่า test script result เมื่อคำสั่งทดสอบไม่สามารถใช้ดำเนินการได้

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC25	ทดสอบการปรับปรุงค่า test script result เมื่อคำสั่งทดสอบไม่สามารถใช้ดำเนินการที่ฐานข้อมูลได้	ระบบทำการปรับปรุง test script result เมื่อคำสั่งทดสอบไม่สามารถใช้ดำเนินการที่ฐานข้อมูลได้	ระบบทำการ update ค่า test script result เป็น fail	ถูกต้อง

ตารางที่ 47 ทดสอบการปรับปรุงค่า test script result เมื่อคำสั่งทดสอบสามารถดำเนินการได้

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC26	ทดสอบการปรับปรุงค่า test script result เมื่อคำสั่งทดสอบสามารถดำเนินการได้	ระบบปรับปรุงค่า test script result เมื่อคำสั่งทดสอบสามารถดำเนินการได้	ระบบทำการ update ค่า test script result เป็น pass	ถูกต้อง

ตารางที่ 48 ทดสอบการแยกประเภทข้อความแสดงข้อผิดพลาดของชื่อตาราง

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC27	ทดสอบการแยกประเภทข้อความแสดงข้อผิดพลาดของชื่อตาราง	ระบบส่งคำสั่งทดสอบที่มีการเขียนชื่อตารางผิด	ระบบได้รับข้อความแสดงความผิดพลาดของคำสั่งทดสอบจากฐานข้อมูล	ถูกต้อง
			ระบบสามารถระบุความผิดปกติว่าเกิดจากชื่อตาราง	ถูกต้อง

ตารางที่ 49 ทดสอบการแยกประเภทข้อความแสดงข้อผิดพลาดของชื่อคอลัมน์

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC28	ทดสอบการแยกประเภทข้อความแสดงข้อผิดพลาดของชื่อคอลัมน์	ระบบส่งคำสั่งทดสอบที่มีการเขียนชื่อคอลัมน์ผิด	ระบบได้รับข้อความแสดงความผิดพลาดของคำสั่งทดสอบจากฐานข้อมูล	ถูกต้อง
			ระบบสามารถระบุความผิดปกติว่าเกิดจากชื่อคอลัมน์	ถูกต้อง

ตารางที่ 50 ทดสอบการแก้ไขคำสั่งทดสอบส่วนชื่อตาราง

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC29	ทดสอบการแก้ไขคำสั่งทดสอบส่วนชื่อตาราง	ระบบทำการแก้ไขคำสั่งทดสอบในส่วนของชื่อตาราง	ระบบแก้ไขคำสั่งทดสอบในส่วนชื่อตารางโดยเลือกชื่อตารางที่มีค่าระยะการแก้ไขน้อยที่สุด	ถูกต้อง

ตารางที่ 51 ทดสอบการแก้ไขคำสั่งทดสอบส่วนชื่อคอลัมน์

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC30	ทดสอบการแก้ไขคำสั่งทดสอบส่วนชื่อคอลัมน์	ระบบทำการแก้ไขคำสั่งทดสอบในส่วนของชื่อคอลัมน์	ระบบแก้ไขคำสั่งทดสอบในส่วนชื่อตารางโดยเลือกชื่อคอลัมน์ที่มีค่าระยะการแก้ไขน้อยที่สุด	ถูกต้อง

ตารางที่ 52 ทดสอบการปรับปรุงค่า คำสั่งทดสอบหลังการแก้ไขในตารางกรณีทดสอบ

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC31	ทดสอบการ	ระบบทำการ	ระบบ update คำสั่ง	ถูกต้อง

	ปรับปรุงค่า คำสั่งทดสอบหลังการแก้ไขในตารางกรณีทดสอบ	update ค่าคำสั่งทดสอบที่ได้รับการแก้ไข	ทดสอบที่ทำการแก้ไขถูกต้องตรงกับกรณีทดสอบ	

ตารางที่ 53 ทดสอบการปรับปรุงค่า Actual result เมื่อมีข้อความแสดงข้อผิดพลาดของคำสั่งทดสอบ

หมายเลขกรณีทดสอบ	**การทดสอบ**	**คำอธิบาย**	**ผลการทดสอบที่คาดหวัง**	**ผลการทดสอบ**
TC32	ทดสอบการปรับปรุงค่า Actual result เมื่อมีข้อความแสดงข้อผิดพลาดของคำสั่งทดสอบ	ระบบทำการการปรับปรุงค่า Actual result เมื่อมีข้อความแสดงข้อผิดพลาดของคำสั่งทดสอบ	ค่า Actual result ไม่มีการ update ใดๆ	ถูกต้อง

ตารางที่ 54 ทดสอบการปรับปรุงค่า Actual result เมื่อเมื่อคำสั่งทดสอบสามารถดำเนินการได้

หมายเลขกรณีทดสอบ	**การทดสอบ**	**คำอธิบาย**	**ผลการทดสอบที่คาดหวัง**	**ผลการทดสอบ**
TC33	ทดสอบการปรับปรุงค่า Actual result เมื่อเมื่อคำสั่งทดสอบสามารถดำเนินการได้	ระบบทำการปรับปรุงค่า Actual result เมื่อเมื่อคำสั่งทดสอบสามารถดำเนินการได้	ค่า Actual result ถูก update ด้วยค่าที่ได้รับมาจากฐานข้อมูลหลังคำสั่งทดสอบถูกดำเนินการ	ถูกต้อง

ตารางที่ 55 ทดสอบกรณีค่า Actual result เป็นค่าว่างเปล่า

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC34	ทดสอบกรณีค่า Actual result เป็นค่าว่างเปล่า	ระบบทำการ update ค่า test result เมื่อค่า Actual result ว่างเปล่า	ค่า test result เป็น fail เนื่องจากคำสั่งทดสอบที่สร้างจากเอกสารไม่สามารถดำเนินการได้	ถูกต้อง

ตารางที่ 56 ทดสอบกรณีค่า Actual result เท่ากับค่า Expected result

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC35	ทดสอบกรณีค่า Actual result เท่ากับค่า Expected result	ระบบทำการ update ค่า test result เมื่อค่า Actual result เท่ากับค่า Expected result	ระบบ update ค่า test result เป็น ‘Pass’	ถูกต้อง

ตารางที่ 57 ทดสอบกรณีค่า Actual result ไม่เท่ากับค่า Expected result

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC36	ทดสอบกรณีค่า Actual result ไม่เท่ากับค่า Expected result	ระบบทำการ update ค่า test result เมื่อค่า Actual result ไม่เท่ากับค่า Expected result	ระบบ update ค่า test result เป็น ‘Fail’	ถูกต้อง

ตารางที่ 58 ทดสอบการแสดงผลการทดสอบกรณีทดสอบ

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC37	ทดสอบการแสดงผลการทดสอบกรณีทดสอบ	ระบบแสดงผลกรณีทดสอบหลังจากทดสอบเสร็จ	หน้าจอ user interface สามารถแสดงผลการทดสอบได้ถูกต้องตรงตามกรณีทดสอบ	ถูกต้อง

ตารางที่ 59 ทดสอบการแสดงรายละเอียดผลการทดสอบกรณีทดสอบ

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบ
TC38	ทดสอบการแสดงรายละเอียดผลการทดสอบกรณีทดสอบ	ระบบแสดงรายละเอียดผลการทดสอบกรณีทดสอบ	ระบบแสดงรายละเอียดการทดสอบตรงตามกรณีทดสอบที่ผู้ใช้งานเลือก	ถูกต้อง

5.2 การประเมินผลระบบ

การประเมินผลระบบจะประเมินโดยทดสอบระบบการสร้างกรณีทดสอบการออกแบบคลังข้อมูล จากการนำเข้าเอกสารการออกแบบจำนวน 2 ฉบับ

5.2.1 ข้อมูลนำเข้า

เอกสารการออกแบบประกอบด้วยข้อมูล 3 ส่วนคือ ส่วนตารางตั้งต้น, ส่วนความสัมพันธ์ของตารางตั้งต้น, และส่วนความสัมพันธ์ของตารางปลายทางกับตารางตั้งต้น ซึ่งตัวอย่างการประเมินผลระบบจากเอกสารการออกแบบฉบับที่ 1 ดังตารางที่ 60 ถึงตารางที่ 64

ตารางที่ 60 ข้อมูลเอกสารการออกแบบส่วนตารางตั้งต้น

	ข้อมูลแถวที 1	ข้อมูลแถวที 2	ข้อมูลแถวที 3
System	TEDW	TEDW	TEDW
Schema Name	CDR	CDR	CDR
Table Name	CDRDB	LKP_DATE	LKP_DOW_TOD
Table Type	Master	Supplementary	Supplementary

Selection Method	Delta	Full	Full
Delta Criteria	DATESTARTOFCHARGING >= START_DTE AND DATESTARTOFCHARGING < START_DTE + 1		
Filtering Criteria	(RECORDTYPE in (1,4) and CAUSE NOT IN (1,2)) Or RECORDTYPE IN (5,7) Or RECORDTYPE in (1,4,5,7) Or RECORDTYPE in (1,4,5,7) Or (RECORDTYPE = 12 AND SUBSTR(IMSI,1,3) <> '520') or (RECORDTYPE = 12 AND SUBSTR(IMSI,1,3) <> '520') or (RECORDTYPE = 13 AND SUBSTR(MISC38,1,3) = '520')		

ตารางที่ 61 ข้อมูลเอกสารการออกแบบส่วนความสัมพันธ์ของตารางตั้งต้น

	ข้อมูลแถวที่ 1	ข้อมูลแถวที่ 2
Table A	CDRDB	CDRDB
Table B	LKP_DATE	LKP_DOW_HOD
Join Type	Inner Join	Inner Join
Join Condition	TRUNC(CDRDB.DATESTARTOFCHARGING) = LKP_DATE.DATE_KEY	CASE WHEN LKP_DATE.IS_HOLIDAY = 'Y' THEN 'HOLIDAY' ELSE TRIM(TO_CHAR(CDRDB.DATESTARTOFCHARGING,'DAY')) END = LKP_DOW_HOD.DAY_OF_WEEK

		AND TRIM(TO_CHAR(CDRDB.DATESTARTOFCHARGING,'HH24')) = LKP_DOW_HOD.HOUR24

ตารางที่ 62 ส่วนความสัมพันธ์ของตารางปลายทางกับตารางตั้งต้น(ตารางปลายทาง)

Target Column Name	Target Data type	Target PK	Description
MSISDN	VARCHAR2(255)		Served MSISDN.
IMEI	VARCHAR2(16)		International Mobile Equipment Identification
IMSI	VARCHAR2(16)		International Mobile Subscriber Identity
DAY	DATE		DATE of calllink to LKP_DATE
A_PARTY	VARCHAR2(255)		Calling _party
A_PARTY_KEY	NUMBER		Description: The DIM_SUBSCRIBER table and CDRNumberanalysis is used to categorise the destinations. Possible Values: 1:TRUE 2:COMPETITOR 3:IDD 4:DOMESTIC 5:APN 6:UNKNOWN 7:RMV 8:RFT
B_PARTY	VARCHAR2(255)		Description: Called Party
B_PARTY_KEY	NUMBER		Description:The DIM_SUBSCRIBER table and CDRNumberanalysis is used to categorise the

			destinations. Possible Values: 1:TRUE 2:COMPETITOR 3:IDD 4:DOMESTIC 5:APN 6:UNKNOWN 7:RMV 8:RFT
FIRSTCGI	VARCHAR2(255)		Description: Cellid where call/sms/data was initiated.
CALL_TYPE_KEY	NUMBER		Description:?Unique Call Type id. Source:This links to LKP_CALL_TYPE
TOD_KEY	NUMBER		Description: Time of Day and Day of Week Mapping. Source:This links to LKP_DOW_HOD.
FAULT_KEY	VARCHAR2(51)		Description: Primary key for Faults (Platform and Cause value concatenated)
SUBSCRIBER_KEY	NUMBER		Description: Indication of Subscriber Network e.g Home Subscriber,Domestic roamer, International Roamer. Links to LKP_SUBSCRIBER
DURATION	NUMBER		Description: Call Duration (seconds)
CALLS	NUMBER		Description:Number of Calls
UPLOAD_VOLUME	NUMBER		Description: Data Volume Uploaded
DOWNLOAD_VOLUME	NUMBER		Description: Data Volume Downloaded

LATELANDINGS	VARCHAR2(10)		Description: Any record of a day which gets loaded in CDRDB after completing Daily processing (procedure execution) of that day is considered as Late landing record. Late landing records are checked from start of the month until month end summaries are generated.
LOADDATETIME	DATE		Description: Datetime (system datetime) when data was loaded into the table
ONNET_ID	NUMBER		The field will be populated with a 1 to indicate that a call was ONNET
SYSTEM_TYPE	VARCHAR2(10)		To identify 2G/3G/4G
A_OPERATOR	VARCHAR2(15)		To identify Calling Operator
B_OPERATOR	VARCHAR2(15)		To identify Called Operator

ตารางที่ 63 ส่วนความสัมพันธ์ของตารางปลายทางกับตารางตั้งต้น(ตารางตั้งต้น)

Source Schema Name	**Source Table Name**	**Source Column Name**	**Transformation Rule**
CDR	CDRDB	B_MSISDN	CASE WHEN RECORDTYPE IN (1,5,12,13) THEN A_MSISDN ELSE B_MSISDN END
CDR	CDRDB	IMEI	IMEI
CDR	CDRDB	IMSI	IMSI
CDR	CDRDB	DATESTARTOFCHARGING	DATESTARTOFCHARGING
CDR	CDRDB	A_MSISDN	A_MSISDN

CDR	CDRDB		CASE WHEN (A_MSISDN IS NULL) THEN 6 WHEN (A_MSISDN LIKE '001%') THEN 3 WHEN (SUBSTR(A_MSISDN,1,2) in ('08','09','06')) AND LENGTH(A_MSISDN) = 10 AND NVL(A1,'UNKN') NOT IN ('TMV','RMV','CATCDMA','RFT') AND (SUBSTR(IMSI,1,5) NOT IN ('52004','52099') or SUBSTR(IMSI,1,7) <> '5200020') THEN 2 WHEN (SUBSTR(A_MSISDN,1,2) in ('08','09','06')) AND LENGTH(A_MSISDN) = 10 AND (A1 in ('RMV' , 'CATCDMA') OR SUBSTR(IMSI,1,7) = '520020') THEN 7 WHEN (SUBSTR(A_MSISDN,1,2) in ('08','09','06)) AND LENGTH(A_MSISDN) = 10 AND (A1 = 'TMV' OR SUBSTR(IMSI,1,5) = '52099') THEN 1 WHEN (SUBSTR(A_MSISDN,1,2) in ('08','09','06')) AND LENGTH(A_MSISDN) = 10 AND (A1 = 'RFT' OR SUBSTR(IMSI,1,5) = '52004') THEN 8 ELSE 4 END CASE WHEN A_OPERATOR = 'TMV'

			THEN 1 WHEN A_OPERATOR = 'RMV' THEN 7 WHEN A_OPERATOR = 'RFT' THEN 8 ELSE A_PARTY_KEY from 1st Step END
CDR	CDRDB	B_MSISDN	B_MSISDN
CDR	CDRDB		CASE WHEN (RECORDTYPE in (12,13)) THEN 5 WHEN RECORDTYPE NOT IN (12,13) AND (B_MSISDN IS NULL) THEN 6 WHEN RECORDTYPE NOT IN (12,13) AND (B_MSISDN LIKE '001%') THEN 3 WHEN RECORDTYPE NOT IN (12,13) AND (SUBSTR(B_MSISDN,1,2) in ('08','09','06')) AND LENGTH(B_MSISDN) = 10 AND NVL(B1,'UNKN') NOT IN ('TMV','RMV','CATCDMA','RFT') AND (SUBSTR(IMSI,1,5) NOT IN ('52004','52099') OR SUBSCR(IMSI,1,7) <> '5200020') THEN 2 WHEN RECORDTYPE NOT IN (12,13) AND (SUBSTR(B_MSISDN,1,2) in ('08','09','06')) AND LENGTH(B_MSISDN) = 10 AND (B1

			in ('RMV','CATCDMA') OR SUBSTR(IMSI,1,7) = '5200020') THEN 7 WHEN RECORDTYPE NOT IN (12,13) AND (SUBSTR(B_MSISDN,1,2) in ('08','09','06')) AND LENGTH(B_MSISDN) = 10 AND (B1 ='TMV' OR SUBSTR(IMSI,1,5) = '52099') THEN 1 WHEN RECORDTYPE NOT IN (12,13) AND (SUBSTR(B_MSISDN,1,2) in ('08','09','06')) AND LENGTH(B_MSISDN) = 10 AND (B1 ='RFT' OR SUBSTR(IMSI,1,5) = '52004') THEN 8 ELSE 4 END CASE WHEN B_OPERATOR = 'TMV' THEN 1 WHEN B_OPERATOR = 'RMV' THEN 7 WHEN B_OPERATOR = 'RFT' THEN 8 ELSE B_OPERATOR_KEY FROM 1st Step END
CDR	CDRDB	FIRSTCGI	If PLATFORM = 'AGP' AND RECORDTYPE = '12' CASE WHEN MISC81 IS NULL AND MISC62 IS NOT NULL AND

			INSTR(MISC62,'RMV') > 0 THEN '00-'||SUBSTR(FIRSTCGI,4) WHEN MISC81 IS NULL AND MISC62 IS NOT NULL AND INSTR(MISC62,'RMV') = 0 THEN '99-'||SUBSTR(FIRSTCGI,4) WHEN MISC81 IS NOT NULL AND INSTR(MISC81,'R8') > 0 THEN '04-'||SUBSTR(FIRSTCGI,4) END Else FIRSTCGI End
CDR	CDRDB	RECORDTYPE	CASE WHEN (RECORDTYPE = 1) AND NVL(TSC,17) = 17 THEN 1 WHEN (RECORDTYPE = 1) AND NVL(TSC,17) = 18 THEN 2 WHEN (RECORDTYPE = 4) AND NVL(TSC,17) = 17 THEN 3 WHEN (RECORDTYPE = 5) THEN 4 WHEN (RECORDTYPE = 7) THEN 5 WHEN (RECORDTYPE = 12) AND PLATFORM = 'AGP' THEN 6 WHEN (RECORDTYPE = 12) AND PLATFORM = 'HGP' THEN 9 WHEN (RECORDTYPE = 13) THEN 10 WHEN (RECORDTYPE = 1) AND

			NVL(TSC,17) IN (96,97,98,99) THEN 7 WHEN (RECORDTYPE = 4) AND NVL(TSC,17) IN (96,97,98,99) THEN 8 ELSE 100 END
CDR	LKP_DOW_HOD	TOD_KEY	TOD_KEY
CDR	CDRDB		PLATFORM||'_'|| CAUSE
CDR	CDRDB		CASE WHEN SUBSTR(IMSI,1,5) in ('52099','52000','52004') THEN 1 WHEN SUBSTR(IMSI,1,3) = '520' AND SUBSTR(IMSI,1,5) NOT IN ('52099','52000','52004') THEN 2 ELSE 3 END
CDR	CDRDB	CHARGEABLEDURATION	SUM(CHARGEABLEDURATION)
CDR	CDRDB		COUNT(*)
CDR	CDRDB	UPLOAD_VOLUME	SUM(UPLOAD_VOLUME)
CDR	CDRDB	DOWNLOAD_VOLUME	SUM(DOWNLOAD_VOLUME)
CDR			
CDR		SYSTEM DATE	SYSTEM DATETIME
CDR	CDRDB	ONNET_ID	

CDR	CDRDB		CASE WHEN RECORDTYPE in (12,13) THEN CASE WHEN (MISC43 is null OR MISC43 = '2') THEN '2G' WHEN MISC43 = '1' THEN '3G' WHEN MISC43 = '6' THEN '4G' END ELSE NULL END

5.2.2 ข้อมูลนำออก

ข้อมูลนำออกเพื่อดำเนินการเป็นคำสั่งทดสอบที่ถูกสร้างจากกรณีทดสอบแต่ละรูปแบบดังนี้

- กรณีทดสอบ Fact test ในระดับ Conceptual design ดังรูปที่ 31
- กรณีทดสอบ Conform test ในระดับ Conceptual design ดังรูปที่ 32
- กรณีทดสอบ Join test ในระดับ Logical design ดังรูปที่ 33
- กรณีทดสอบ Star test ในระดับ Logical design ดังรูปที่ 34
- กรณีทดสอบ Constraints test ในระดับ Physical design ดังรูปที่ 35
- กรณีทดสอบ Transform test ในระดับ Physical design ดังรูปที่ 36

DOC_TEST	201512092355URCDR
TEST_LEVEL	CONCEPTUAL
TEST_TYPE	Conceptual volume fact test
SOURCE_SCR1	SELECT COUNT(*) AS TOTAL_TRANSACTIONS FROM CDR.CDRDB
ACTUAL_RESULT1	
TEST_SCRIPT_RS1	Fail Script error!
EXP_RESULT1	Data type : int , count transaction > 0 row
TEST_DESC	count data volumn of table : CDR.CDRDB
ESOURCE_SCR1	SELECT COUNT(*) AS TOTAL_TRANSACTIONS FROM SERVER.CDRDB
TEST_ESCRIPT_RS1	Pass
EACTUAL_RESULT1	[{'TOTAL_TRANSACTIONS': 2202}]
TEST_RS1	Pass

รูปที่ 31 กรณีทดสอบ Fact test ในระดับ Conceptual design

DOC_TEST	201512092355URCDR
TEST_LEVEL	CONCEPTUAL
TEST_TYPE	Conform test Target agianst Source
SOURCE_SCR1	SELECT COUNT(*) AS EXIT_COUNT FROM POLLS_TEDWCF_SRCTABLE WHERE SCHEMA_NM='CDR' AND TABLE_NM= 'LKP_DOW_HOD' AND DOC_SRC ='201512092355URCDR'
EXP_RESULT1	CDR.LKP_DOW_HOD exist in source table design
ACTUAL_RESULT1	[{'EXIT_COUNT': 0}]
PPN_TM	09 ธ.ค. 2015 16.55.58.514000
TEST_DESC	tableCDR.LKP_DOW_HOD of source1 exist in source summary
TEST_SCRIPT_RS1	Pass

รูปที่ 32 กรณีทดสอบ Conform test ในระดับ Conceptual design

DOC_TEST	201512092355URCDR
TEST_LEVEL	LOGICAL
TEST_TYPE	Join condition test
TEST_DESC	Join test of table SERVER.CDRDB
SOURCE_SCR1	SELECT COUNT(*) AS TRANSACTIONS FROM SERVER.CDRDB CDRDB INNER JOIN CDR.LKP_DOW_HOD LKP_DOW_HOD ON
TEST_RS1	Pass
EXP_RESULT1	observation not equal to zero.
ACTUAL_RESULT1	[{'TRANSACTIONS': 17616}]
PPN_TM	09 ธ.ค. 2015 16.56.00.225000
ESOURCE_SCR1	SELECT COUNT(*) AS TRANSACTIONS FROM SERVER.CDRDB CDRDB INNER JOIN CDR.LKP_DOW_HOD LKP_DOW_HOD ON
TEST_SCRIPT_RS1	Pass
EACTUAL_RESULT1	[{'TRANSACTIONS': 17616}]
TEST_ESCRIPT_RS1	Pass

รูปที่ 33 กรณีทดสอบ Join test ในระดับ Logical design

DOC_TEST	201512092355URCDR
TEST_LEVEL	LOGICAL
TEST_TYPE	Relation star test
TEST_DESC	Aggeergate test of column : UPLOAD_VOLUME
SOURCE_SCR1	SELECT AVG(DATABYTES) AS AGG FROM SERVER.CDRDB CDRDB INNER JOIN CDR.LKP_DOW_HOD LKP_DOW_HOD ON TRIM(TO_CHAR(CDRDB.DATESTARTOFCHARGING,'HH24')) = LKP_DOW_HOD.HOUR24 INNER JOIN CDR.LKP_DATE LKP_DATE ON TRUNC(CDRDB.DATESTARTOFCHARGING) = LKP_DATE.DATE_KEY
TEST_SCRIPT_RS1	Pass
EXP_RESULT1	THE FUNCTION RETURNS THE SAME DATA TYPE AS THE NUMERIC DATA TYPE OF THE
ACTUAL_RESULT1	[{'AGG': Decimal('626818.68513119533527696793002915451895 04')}]
PPN_TM	09 ธ.ค. 2015 16.56.01.137000
TEST_RS1	Pass

รูปที่ 34 กรณีทดสอบ Star test ในระดับ Logical design

DOC_TEST	201512092355URCDR
TEST_LEVEL	PHYSICAL
TEST_TYPE	Constraints test data type
SOURCE_SCR1	SELECT DATA_TYPE FROM ALL_TAB_COLUMNS WHERE OWNER ='CDR' AND TABLE_NAME ='CDRDB' AND COLUMN_NAME ='B_MSISDN'
EXP_RESULT1	VARCHAR2(255)
ACTUAL_RESULT1	[]
TEST_SCRIPT_RS1	Pass
PPN_TM	09 ธ.ค. 2015 16.56.02.905000
TEST_DESC	Test datatype of target column : B_PARTY
ESOURCE_SCR1	SELECT DATA_TYPE FROM ALL_TAB_COLUMNS WHERE OWNER ='SERVER' AND TABLE_NAME ='CDRDB' AND COLUMN_NAME ='B_MSISDN'
TEST_ESCRIPT_RS1	Pass
EACTUAL_RESULT1	[{'DATA_TYPE': u'VARCHAR2'}]
TEST_RS1	Pass

รูปที่ 35 กรณีทดสอบ Constraints test ในระดับ Physical design

DOC_TEST	2015120923555URCDR
TEST_LEVEL	PHYSICAL
TEST_TYPE	Transform test Case When
TEST_DESC	Test transformation rule of target column : SYSTEM_TYPE
EXP_RESULT1	,'2G','3G','4G',NULL,
SOURCE_SCR1	SELECT * FROM(SELECT DISTINCT CASE WHEN RECORDTYPE IN (12,13) THEN CASE WHEN (MISC43 IS NULL OR MISC43 = '2') THEN '2G'
TEST_SCRIPT_RS1	Fail Script error!
ACTUAL_RESULT1	
PPN_TM	09 ธ.ค. 2015 16.56.04.657000
ESOURCE_SCR1	SELECT * FROM (SELECT DISTINCT CASE WHEN RECORDTYPE IN (12,13) THEN CASE WHEN (MISC43 IS NULL OR MISC43 = '2') THEN '2G'
TEST_ESCRIPT_RS1	Pass
EACTUAL_RESULT1	[{'R': u'2G'}, {'R': u'3G'}, {'R': u'4G'}, {'R': None}]
TEST_RS1	Pass

รูปที่ 36 กรณีทดสอบ Transform test ในระดับ Physical design

ข้อมูลนำออกส่วนคำสั่งทดสอบที่ได้รับการแก้ไข ซึ่งสามารถแบ่งได้เป็น 2 ขั้นของการสร้างคำสั่ง คือ ขั้นแรกสร้างคำสั่งจากข้อมูลภายในเอกสารการออกแบบโดยไม่ทำการแก้ไขใด ๆแสดงในช่อง SOURCE_SRC1 และ หลังจากทำการดำเนินการกับคำสั่งดังกล่าวแล้วหากผลออกมาเป็น Fail ระบบจะทำการแก้ไขคำสั่งโดยใช้การวัดระยะการแก้ไข จากนั้นจะได้คำสั่งในขั้นที่สองเป็นคำสั่งที่ผ่านการแก้ไข แสดงในช่อง ESOURCE_SCR1 หากข้อมูลในเอกสารการออกแบบมีมากกว่า 1 แหล่งก็จะมี SOURCE_SRC2 และ ESOURCE_SCR2 เพิ่มเข้ามาได้จนถึง แหล่งที่ 5 ดังรูปที่ 37

DOC_TEST	2015120923555URCDR
TEST_LEVEL	CONCEPTUAL
TEST_TYPE	Conform test source against Database
TEST_DESC	check source tableCDR.LKP_DOW_TOD conform with source summary design
SOURCE_SCR1	SELECT SUM(TOTAL_ROW) AS TOTAL_ROW FROM (SELECT COUNT(*) AS TOTAL_ROW FROM ALL_ALL_TABLES S WHERE TO_CHAR(S.OWNER ||S.TABLE_NAME) = 'CDRLKP_DOW_TOD' UNION SELECT COUNT(*) AS TOTAL_ROW FROM ALL_VIEWS S WHERE TO_CHAR(S.OWNER ||S.VIEW_NAME) = 'CDRLKP_DOW_TOD')
ESOURCE_SCR1	SELECT SUM(TOTAL_ROW) AS TOTAL_ROW FROM (SELECT COUNT(*) AS TOTAL_ROW FROM ALL_ALL_TABLES S WHERE TO_CHAR(S.OWNER ||S.TABLE_NAME) = 'CDRLKP_DOW_HOD' UNION SELECT COUNT(*) AS TOTAL_ROW FROM ALL_VIEWS S WHERE TO_CHAR(S.OWNER ||S.VIEW_NAME) = 'CDRLKP_DOW_HOD')
EXP_RESULT1	table name CDR.LKP_DOW_TOD exist in database
ACTUAL_RESULT1	[{'TOTAL_ROW': 0}]

รูปที่ 37 คำสั่งทดสอบที่ได้รับการแก้ไข

ข้อมูลนำออกส่วนผลการดำเนินการกับคำสั่งทดสอบและผลการทดสอบกรณีทดสอบแสดงดังรูปที่ 38 สำหรับผลข้อมูลนำออกส่วนของผลการดำเนินการ จะแสดง 2 ส่วนคือส่วนของ Test Script result ซึ่งเป็นผลจากการใช้ script ที่สร้างจากข้อมูลในเอกสารการออกแบบ เพื่อเป็นการยืนยันความถูกต้องของข้อมูลที่เขียนโดยผู้ออกแบบได้ส่วนหนึ่ง ส่วนที่ 2 คือ Test result ของกรณีทดสอบเกิดจากการเปรียบเทียบค่าคาดหวังของกรณีทดสอบกับค่าที่ได้จากการดำเนินการที่เกิดจากคำสั่งทดสอบหรือคำสั่งทดสอบที่ได้รับการแก้ไขแล้ว

TEST_LEVEL	TEST_DESC	TEST_TYPE	TEST_RS1	TEST_SCRIPT_RS1
LOGICAL	Aggeergate test of column : DURATION	Relation star test	Pass	Pass
LOGICAL	Aggeergate test of column : DURATION	Relation star test	Pass	Pass
PHYSICAL	Test datatype of target column : A_PARTY	Constraints test data type	Pass	Pass
PHYSICAL	Test datatype of target column : B_PARTY	Constraints test data type	Pass	Pass
PHYSICAL	Test datatype of target column : TOD_KEY	Constraints test data type	Pass	Pass
PHYSICAL	Test datatype of target column : FAULT_KEY	Constraints test data type	Fail	Fail Script error!
PHYSICAL	Test datatype of target column : UPLOAD_VOLUME	Constraints test data type	Pass	Pass
PHYSICAL	Test datatype of target column : DOWNLOAD_VOLUME	Constraints test data type	Pass	Pass
PHYSICAL	Test datatype of target column : LOADDATETIME	Constraints test data type	Fail	Pass
PHYSICAL	Test datatype of target column : ONNET_ID	Constraints test data type	Pass	Pass
PHYSICAL	Test datatype of target column : DURATION	Constraints test data type	Pass	Pass
PHYSICAL	Test transformation rule of target column : FIRSTCGI	Transform test If Else	Fail	Fail Script error!
PHYSICAL	Test transformation rule of target column : A_OPERATOR	Transform test If Else	Pass	Fail Script error!
PHYSICAL	Test transformation rule of target column : B_OPERATOR	Transform test If Else	Pass	Fail Script error!
PHYSICAL	Test transformation rule of target column : A_PARTY_KEY	Transform test Case When	Fail	Fail Script error!
PHYSICAL	Test transformation rule of target column : B_PARTY_KEY	Transform test Case When	Fail	Fail Script error!

รูปที่ 38 ผลการดำเนินการกรณีทดสอบ

ข้อมูลนำออกส่วนรายละเอียดกรณีทดสอบแสดงดังรูปที่ 39 แสดงรายละเอียดของกรณีทดสอบ คำสั่งทดสอบ คำสั่งทดสอบที่ผ่านการแก้ไข ผลการดำเนินการจากคำสั่งทดสอบ และผลการทดสอบกรณีทดสอบ

DOC_TEST	201512092355URCDR
TEST_LEVEL	PHYSICAL
TEST_TYPE	Constraints test data type
SOURCE_SCR1	SELECT DATA_TYPE FROM ALL_TAB_COLUMNS WHERE OWNER ='CDR' AND TABLE_NAME ='CDRDB' AND COLUMN_NAME ='PLATFORM||'_'|| CAUSE'
TEST_SCRIPT_RS1	Fail Script error!
EXP_RESULT1	VARCHAR2(51)
ACTUAL_RESULT1	
TEST_ESCRIPT_RS1	Fail Script error!
TEST_RS1	Fail
TEST_DESC	Test datatype of target column : FAULT_KEY
ESOURCE_SCR1	SELECT DATA_TYPE FROM ALL_TAB_COLUMNS WHERE OWNER ='SERVER' AND TABLE_NAME ='CDRDB' AND COLUMN_NAME ='PLATFORM||'_'|| CAUSE'
PPN_TM	09 ธ.ค. 2015 16.56.03.074000

รูปที่ 39 รายละเอียดผลการดำเนินการทดสอบ

5.2.3 สรุปการประเมินผลระบบ

ผลการทดสอบระบบโดยการสร้างกรณีทดสอบจากข้อมูลนำเข้าเป็นเอกสารการออกแบบจำนวน 2 ฉบับให้กรณีทดสอบจำนวน 154 กรณีทดสอบ

เอกสารการออกแบบฉบับที่ 1 การทดสอบกับเอกสารการออกแบบชื่อ DAILY_EXTRACT-Schema of CDR.xlsx ได้กรณีทดสอบจำนวน 62 กรณีทดสอบ

กรณีทดสอบในระดับ Conceptual จำนวน 24 กรณีทดสอบ

- สร้างคำสั่งทดสอบ 9 คำสั่ง คำสั่งทดสอบที่ดำเนินการได้จำนวน 7 คำสั่ง
- สร้างคำสั่งทดสอบที่ได้รับการแก้ไข 21 คำสั่ง คำสั่งทดสอบที่ดำเนินการได้จำนวน 21 คำสั่ง
- การวัดระยะการแก้ไขส่งผลให้ได้ Result ในกรณีทดสอบทั้งหมดจาก 7 results เพิ่มเป็น 24 results
- ผลการเปรียบเทียบ result และ expected result เป็น pass 24 กรณีทดสอบ pass ตามจริง 23 กรณีทดสอบ

กรณีทดสอบในระดับ Logical จำนวน 16 กรณีทดสอบ

- สร้างคำสั่งทดสอบ 16 คำสั่ง คำสั่งทดสอบที่ดำเนินการได้จำนวน 16 คำสั่ง
- สร้างคำสั่งทดสอบที่ได้รับการแก้ไข 0 คำสั่ง
- ผลการเปรียบเทียบ result และ expected result เป็น pass 16 กรณีทดสอบ pass ตามจริง 16 กรณีทดสอบ

กรณีทดสอบในระดับ Physical จำนวน 22 กรณีทดสอบ

- สร้างคำสั่งทดสอบ 21 คำสั่ง คำสั่งทดสอบที่ดำเนินการได้จำนวน 11 คำสั่ง
- สร้างคำสั่งทดสอบที่ได้รับการแก้ไข 19 คำสั่ง คำสั่งทดสอบที่ดำเนินการได้จำนวน 14 คำสั่ง
- การวัดระยะการแก้ไขส่งผลให้ได้ Result ในกรณีทดสอบทั้งหมดจาก 11 results เพิ่มเป็น 16 results
- ผลการเปรียบเทียบ result และ expected result เป็น pass 15 กรณีทดสอบ pass ตามจริง 15 กรณีทดสอบ

เอกสารการออกแบบฉบับที่ 1 การทดสอบกับเอกสารการออกแบบชื่อ FCT_MTU_TOPUP.xlsx ได้กรณีทดสอบจำนวน 92 กรณีทดสอบ

กรณีทดสอบในระดับ Conceptual จำนวน 57 กรณีทดสอบ

- สร้างคำสั่งทดสอบ 47 คำสั่ง คำสั่งทดสอบที่ดำเนินการได้จำนวน 47 คำสั่ง
- สร้างคำสั่งทดสอบที่ได้รับการแก้ไข 10 คำสั่ง คำสั่งทดสอบที่ดำเนินการได้จำนวน 10 คำสั่ง
- การวัดระยะการแก้ไขส่งผลให้ได้ Result ในกรณีทดสอบทั้งหมดจาก 47 results เพิ่มเป็น 57 results
- ผลการเปรียบเทียบ result และ expected result เป็น pass 56 กรณีทดสอบ pass ตามจริง 56 กรณีทดสอบ

กรณีทดสอบในระดับ Logical จำนวน 1 กรณีทดสอบ

- สร้างคำสั่งทดสอบ 1 คำสั่ง คำสั่งทดสอบที่ดำเนินการได้จำนวน 0 คำสั่ง
- สร้างคำสั่งทดสอบที่ได้รับการแก้ไข 1 คำสั่ง คำสั่งทดสอบที่ดำเนินการได้จำนวน 0 คำสั่ง
- ผลการเปรียบเทียบ result และ expected result เป็น pass 0 กรณีทดสอบ pass ตามจริง 0 กรณีทดสอบ

กรณีทดสอบในระดับ Physical จำนวน 34 กรณีทดสอบ

- สร้างคำสั่งทดสอบ 34 คำสั่ง คำสั่งทดสอบที่ดำเนินการได้จำนวน 31 คำสั่ง
- สร้างคำสั่งทดสอบที่ได้รับการแก้ไข 3 คำสั่ง คำสั่งทดสอบที่ดำเนินการได้จำนวน 3 คำสั่ง
- การวัดระยะการแก้ไขส่งผลให้ได้ Result ในกรณีทดสอบทั้งหมดจาก 31 results เพิ่มเป็น 34 results
- ผลการเปรียบเทียบ result และ expected result เป็น pass 34 กรณีทดสอบ pass ตามจริง 34 กรณีทดสอบ

ผลการประเมินผลระบบได้ค่าความแม่นยำของการให้ผลกรณีทดสอบ 99.3%

ผลการประเมินผลระบบได้ค่าความแม่นยำของการใช้การวัดระยะการแก้ไข 88.8%

บทที่ 6
สรุปผลการวิจัยและข้อเสนอแนะ

6.1 สรุปผลการวิจัย

ผลลัพธ์โดยสรุปที่ได้จากงานวิจัยนี้มีดังนี้

1. ได้แนวทางสำหรับการสร้างกรณีทดสอบการออกแบบคลังข้อมูล
2. ได้ระบบต้นแบบสำหรับการสร้างกรณีทดสอบการออกแบบคลังข้อมูล
3. ได้แนวทางและเครื่องมือที่ใช้ในการสร้างกรณีทดสอบและลำดับการทดสอบ
4. ได้เครื่องมือในการสร้างคำสั่งทดสอบ
5. ได้เครื่องมือที่สามารถใช้ในการสร้างกรณีทดสอบการออกแบบคลังข้อมูลที่นำไปปรับใช้กับฐานข้อมูลของงานด้านคลังข้อมูลได้จริง
6. ได้เครื่องมือที่ช่วยสนับสนุนและลดข้อผิดพลาดการสร้างคลังข้อมูล

6.2 ข้อจำกัด

1. เอกสารการออกแบบที่นำเข้าเครื่องมือในงานวิจัยนี้รอบรับเฉพาะไมโครซอฟท์ เอกซ์เซล รวมทั้งรูปแบบการเขียนเอกสารการออกแบบต้องเขียนตามรูปแบบที่กำหนดไว้ในงานวิจัย
2. เครื่องมือที่ใช้ในการทดสอบสามารถติดต่อกับฐานข้อมูลเซิร์ฟเวอร์เดียวและเป็นระบบฐานข้อมูลแบบเชิงสัมพันธ์
3. ผลการทดสอบยังมีค่า 2 ค่า คือ Pass และ Fail โดยไม่ได้อธิบายรายละเอียดของผลการทดสอบ

6.3 แนวทางการวิจัยต่อ

งานวิจัยนี้ได้เสนอแนวทางการสร้างกรณีทดสอบเพื่อทดสอบเอกสารการออกแบบคลังข้อมูล โดยผู้วิจัยเห็นว่าในการออกแบบคลังข้อมูลมีลักษณะความหลากหลาย การหาแนวทางทดสอบทางด้านคลังข้อมูลถือเป็นเรื่องท้าทายในการตรวจสอบการออกแบบในลักษณะนี้อีกมาก เช่นการสร้างกรณีทดสอบกับกรณีข้อมูลต้นทางมีลักษณะเป็นไฟล์ต่าง ๆ การสร้างกรณีทดสอบที่ไม่ใช้ข้อมูลต้นทางจาก Staging area แต่สามารถเชื่อมต่อไปหลาย ๆ ฐานข้อมูลเซิร์ฟเวอร์ หรือฐานข้อมูลที่มีขนาดใหญ่ (Big data) ซึ่งต้องหาวิธีเพื่อการสร้างกรณีทดสอบสำหรับการออกแบบที่แตกต่างกันและทำให้แนวทางที่ออกแบบสามารถนำไปใช้ได้ในการทำงานจริงต่อไป

รายการอ้างอิง

1. Stefano Rizzi, A.A., Jens Lechtenbörger,Juan Trujillo, *Research in data warehouse modeling and design: dead or alive?*, in *DOLAP '06 Proceedings of the 9th ACM international workshop on Data warehousing and OLAP*. 2006, ACM New York, NY, USA ©2006. p. 3-10
2. *Data Modeling - Conceptual, Logical, And Physical Data Models*. Available from: http://www.1keydata.com.
3. *Minimun Edit Distance*. Available from: https://web.stanford.edu.
4. *SQL*. Available from: http://en.wikipedia.org/wiki/SQL.
5. Shadi Abdul Khalek, S.K., *Automated SQL query generation for systematic testing of database engines*, in *ACM international conference on Automated software engineering*. 2010, ACM New York, NY, USA ©2010. p. 329-332
6. Matteo Golfarelli , S.R., *A comprehensive approach to data warehouse testing*, in *Proceedings of the ACM twelfth international workshop on Data warehousing and OLAP*. 2009, ACM New York, NY, USA ©2009. p. 17-24

ภาคผนวก

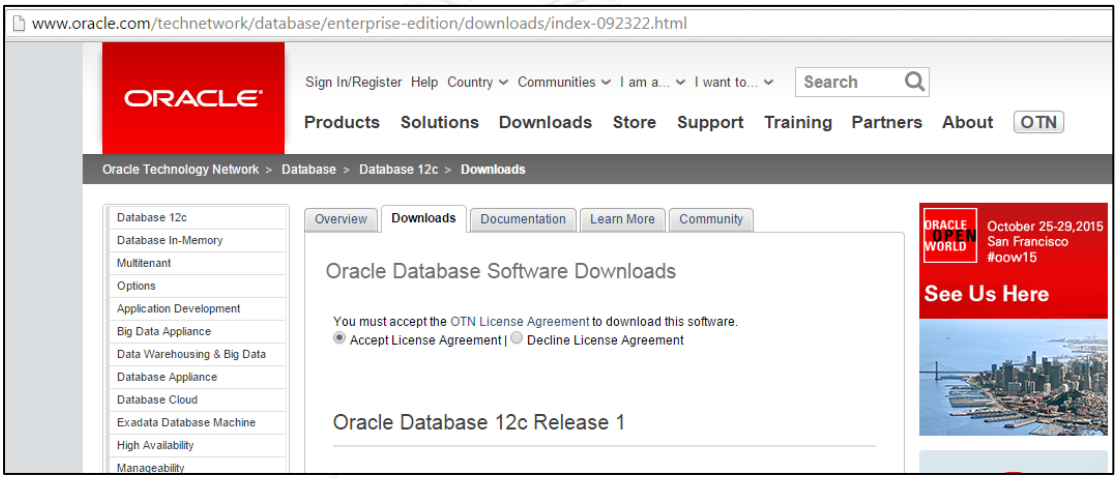
ภาคผนวก ก

การติดตั้งฐานข้อมูล

1. การติดตั้ง 10g Release 2 (10.2) สำหรับ Microsoft Windows (32-Bit)

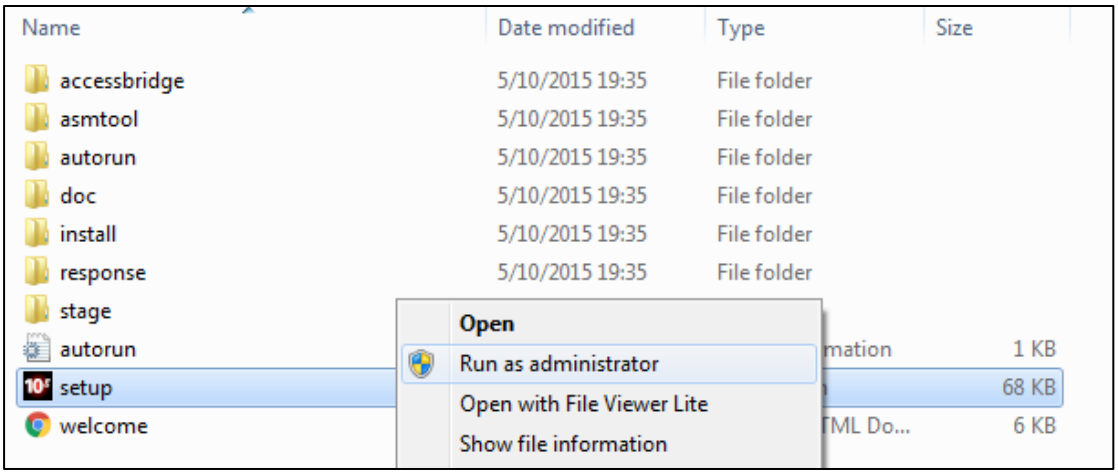
ฐานข้อมูล Oracle ใช้สำหรับจำลองฐานข้อมูลที่ใช้สำหรับการทดสอบคำสั่งเอสคิวแอลในงานวิจัยโดยเลือก Oracle version 10g สำหรับ Windows 32 bits มีขั้นตอนการติดตั้งดังนี้

ทำการดาวน์โหลดโปรแกรมติดตั้งฐานข้อมูล oracle ได้จาก http://www.oracle.com /technetwork/database/enterprise-edition/downloads/index-092322.html เลือก accept License Agreement แล้วเลือกเวอร์ชันของโปรแกรมติดตั้งที่ต้องการ โดยในงานวิจัยนี้ผู้วิจัยเลือก Oracle Database 10g Release 2 สำหรับ Microsoft Windows (32-bit) ทำการดาวน์โหลดและบันทึกไว้ใน directory ใดๆ



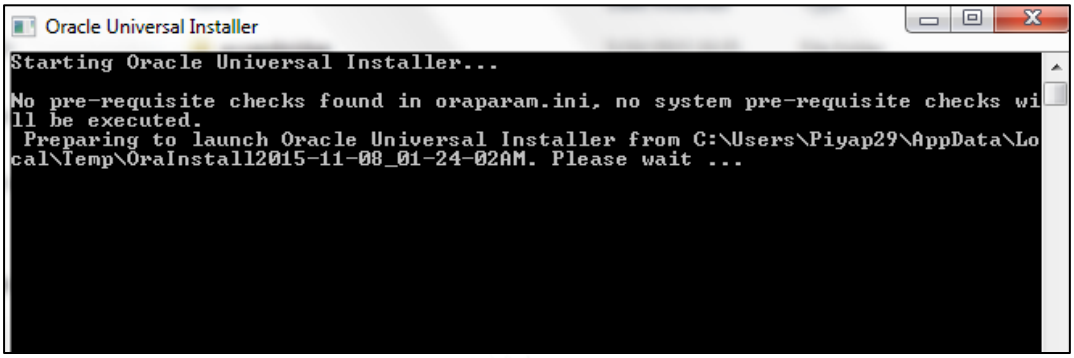
รูปที่ 40 การดาวน์โหลดโปรแกรมติดตั้งฐานข้อมูล oracle

เริ่มต้นการติดตั้งจากการเปิดไปที่ folder directory path ของโปรแกรมติดตั้งโดยให้ทำการคลิกขวาที่ Setup แล้วเลือก Run as administrator ดังรูปที่ 41



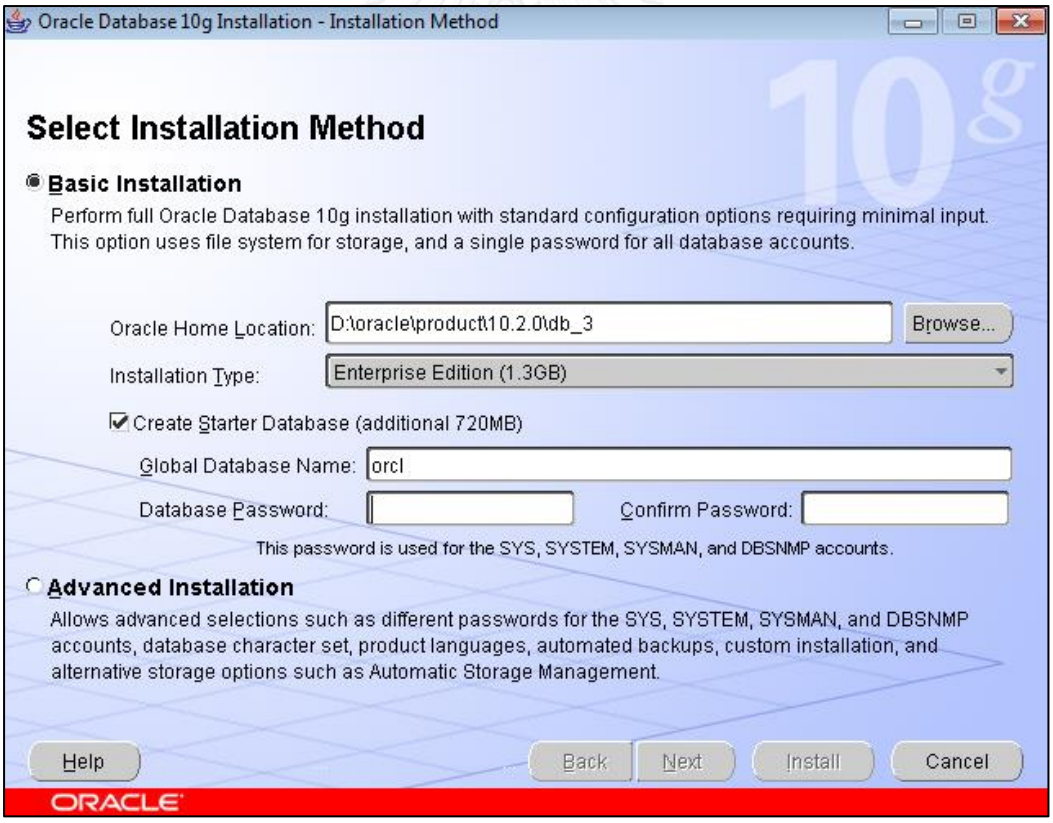
รูปที่ 41 โปรแกรมติดตั้งฐานข้อมูล Oracle

เมื่อสั่ง Run แล้วจะปรากฏหน้าจอ Dos เพื่อบอกว่าเครื่องพร้อมสำหรับการติดตั้งฐานข้อมูลหรือไม่ หากไม่พร้อมจะปรากฏข้อความ error



รูปที่ 42 หน้าจอ Dos เพื่อแจ้งว่าพร้อมสำหรับการติดตั้ง

เริ่มติดตั้งฐานข้อมูลโดยจะปรากฏหน้า Welcome Screen



รูปที่ 43 Welcome Screen

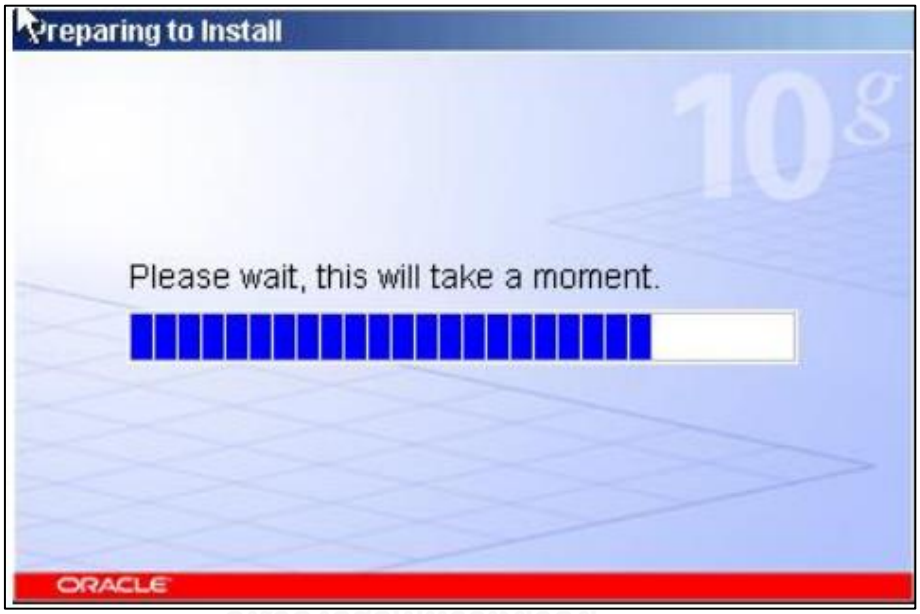
โดยผู้ใช้งานต้องเลือกตั้งค่าดังนี้

ส่วนของ Oracle Home Location ตำแหน่งของ Directory บนเครื่องที่ต้องการติดตั้ง Oracle Installation Type มี 3 แบบ โดยเลือกแบบ Enterprise Edition

ส่วนของ Create Starter Database กำหนดให้ป้อนข้อมูลดังนี้

- Global Name ชื่อเรียกของฐานข้อมูล
- Database Password / Confirm Password เป็น Password ที่ใช้สำหรับ User ต่าง ๆ ที่จะถูกสร้างขึ้นจากการติดตั้งฐานข้อมูล

เมื่อเลือกที่ปุ่ม Next จะเข้าสู่ขั้นตอนการเตรียมพร้อมสำหรับติดตั้ง

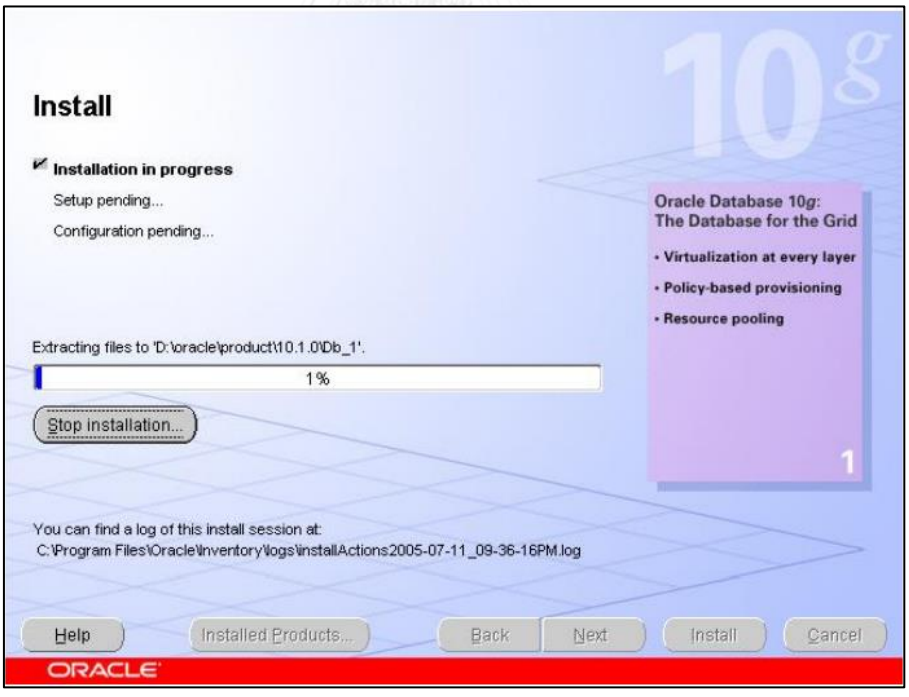


รูปที่ 44 หน้าจอแสดงการเตรียมพร้อมสำหรับการติดตั้งฐานข้อมูล

จากนั้นทำการจะปรากฏหน้าจอที่แสดงผลสรุปของการติดตั้งฐานข้อมูลว่าจะมีการติดตั้งอะไรบ้างดังรูปที่ 45 จากนั้นเลือกที่ปุ่ม install เพื่อติดตั้ง

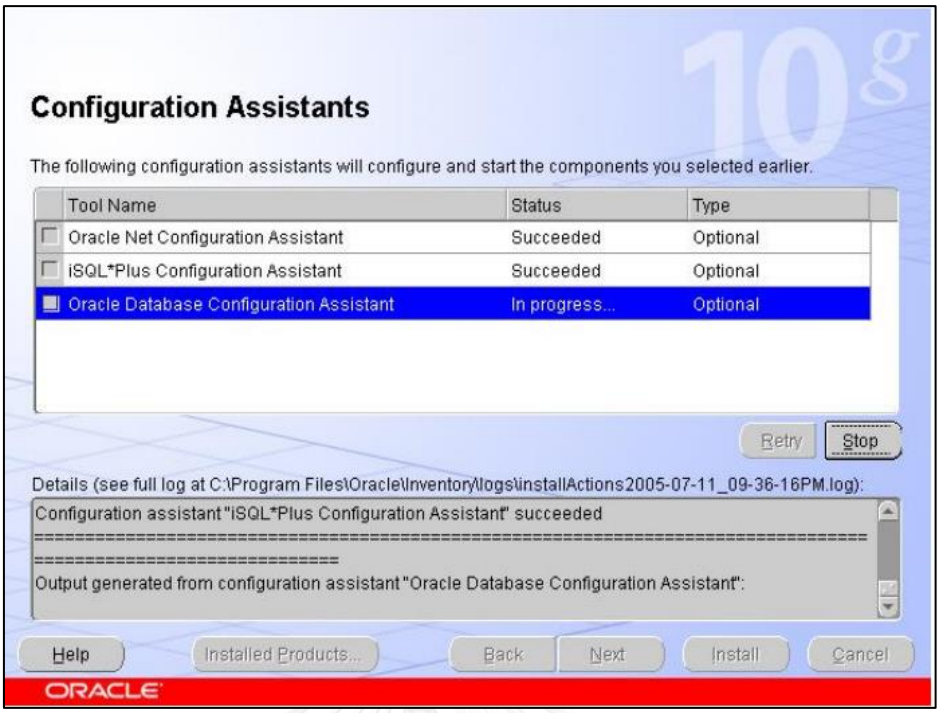
รูปที่ 45 หน้าจอแสดงผลสรุปของการติดตั้งฐานข้อมูล

จากนั้นจะปรากฏหน้าจอที่แสดงความคืบหน้าของการติดตั้ง

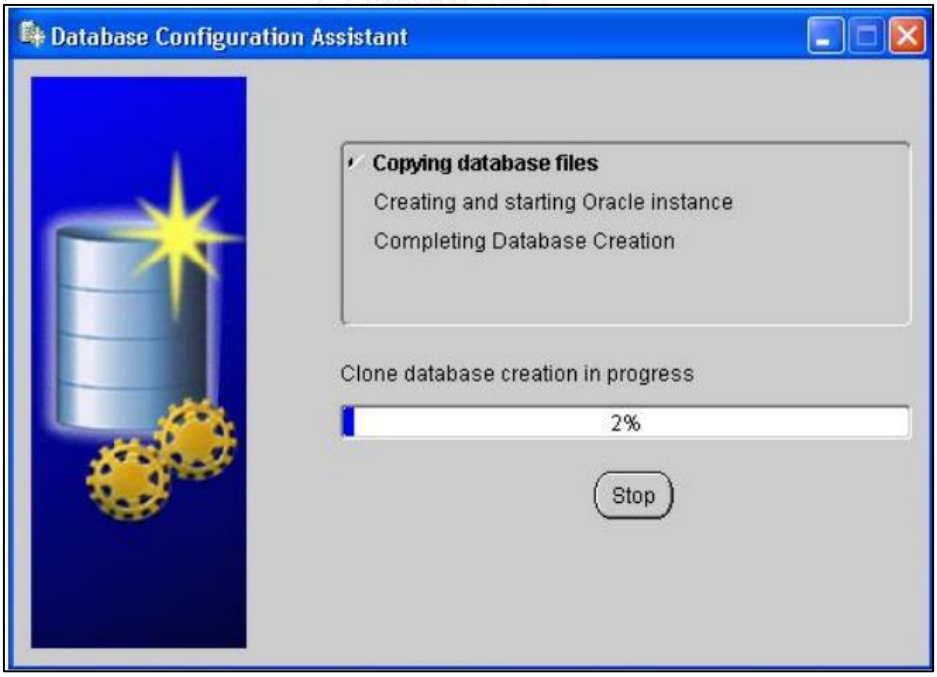


รูปที่ 46 หน้าจอแสดงการความคืบหน้าติดตั้ง

เมื่อทำการติดตั้งครบ 100% แล้วจะปรากฏหน้าจอ Configuration Assistants ให้รอจนกว่า configurations assistants จะทำงานครบ 100%



รูปที่ 47 หน้าจอ configurations assistants

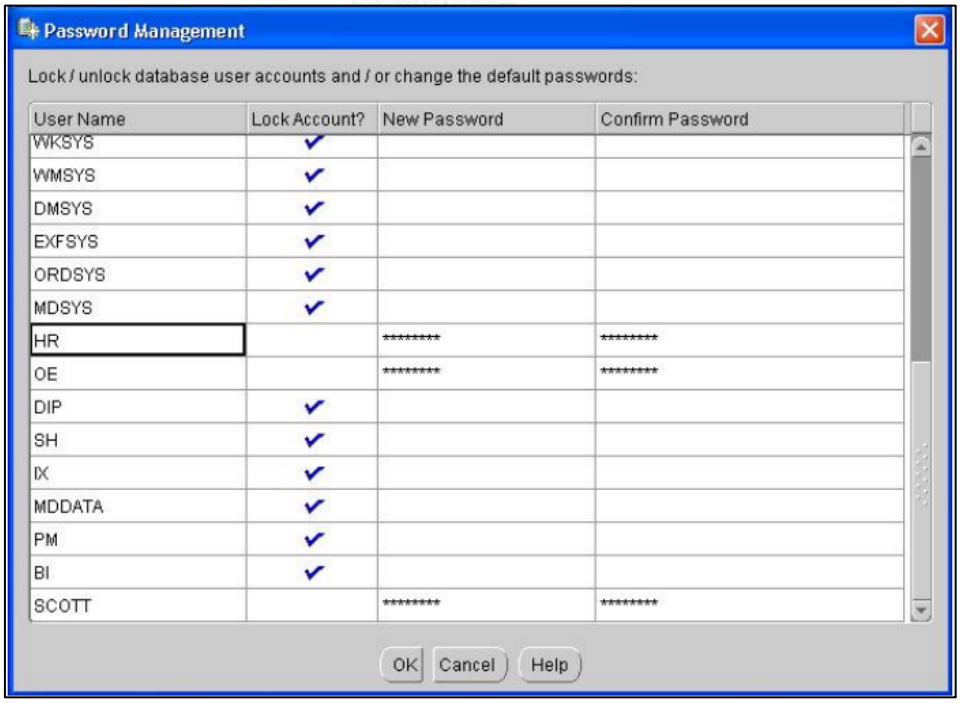


รูปที่ 48 หน้าจอแสดงความคืบหน้า Configuration assistants

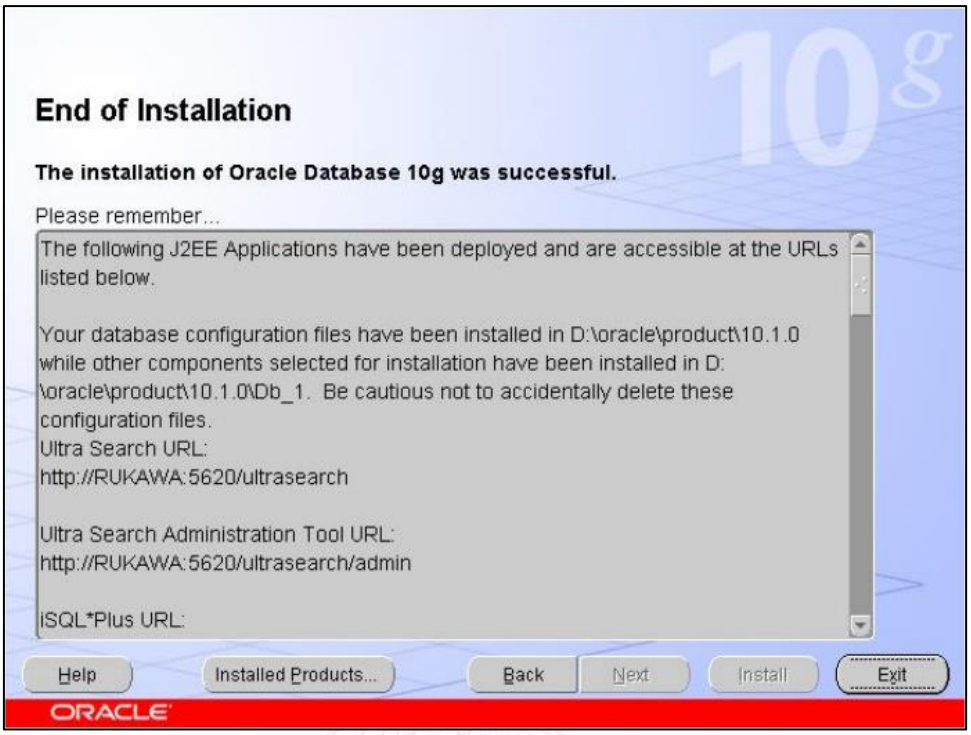
หากการติดตั้งเสร็จสมบูรณ์จะปรากฏข้อความดังรูปที่ 49 หากทำการเลือกที่ปุ่ม Password Management จะปรากฏหน้าจอสำหรับการจัดการ user name และ password ที่ใช้ในการจัดการฐานข้อมูล

รูปที่ 49 ข้อความแสดงการติดตั้งเสร็จสมบูรณ์

จากหน้าจอ Pass word Management ดังรูปที่ 50 จะเห็นว่ามี user name ที่ถูกสร้างขึ้นโดยอัตโนมัติซึ่งหมายถึง user ในระดับผู้ดูแลระบบ การ uncheck ที่ Lock Account เพื่อให้ user name สามารถใช้งานได้ จากนั้นให้เลือกที่ปุ่ม OK เป็นการเสร็จสิ้นการติดตั้งฐานข้อมูล



รูปที่ 50 หน้าจอ Password management



รูปที่ 51 หน้าจอสิ้นสุดการดำเนินการติดตั้งฐานข้อมูล

2. การสร้าง Class model เพื่อสร้างตารางที่ฐานข้อมูล

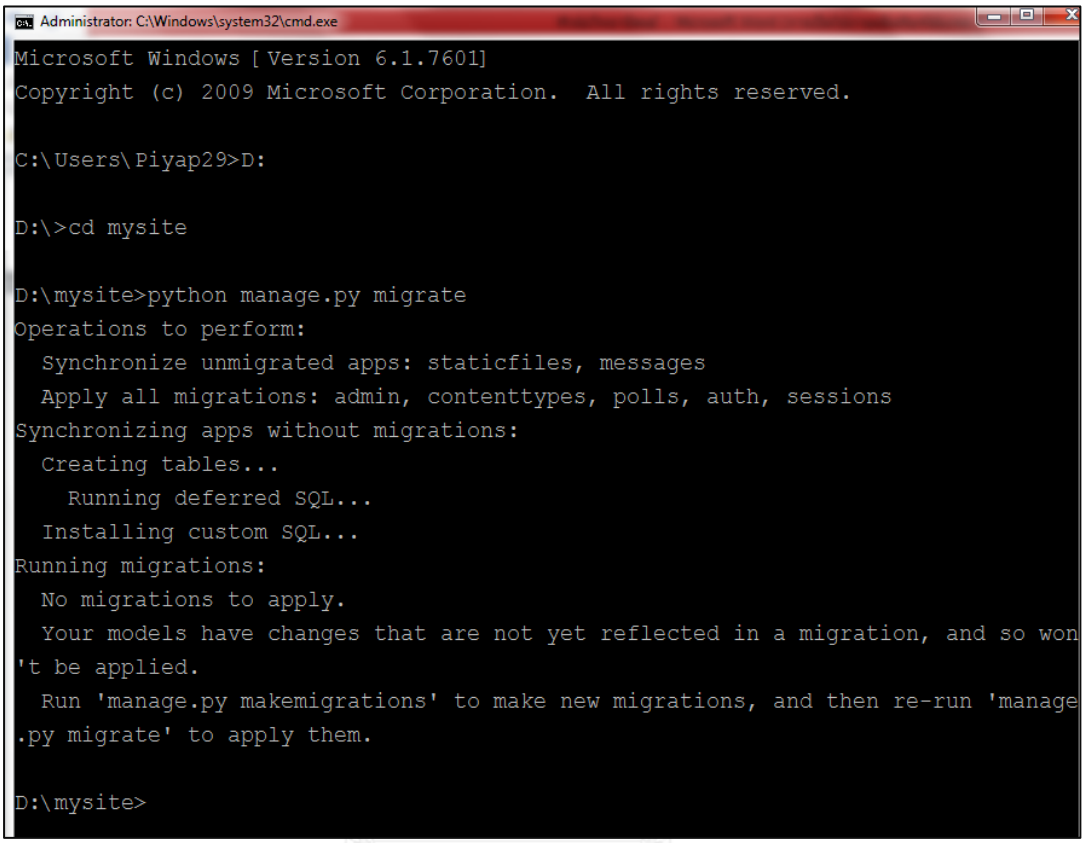
สำหรับการสร้างตารางในฐานข้อมูลเพื่อบันทึกค่าต่าง ๆ ในส่วนของ Django framework กำหนดให้สามารถทำผ่านการเขียนคำสั่งที่ file model.py ซึ่งมีลักษณะคล้ายกับการเขียนคำสั่งเพื่อสร้างตารางในฐานข้อมูลด้วยคำสั่งเอสคิวแอลทั่วไป ตัวอย่างการเขียน class model ดังรูปที่ 52 โดย TEST_CREATE_TABLE คือชื่อของตารางที่จะถูกสร้างในฐานข้อมูล และ COLUMN_A ถึง COLUMN_D เป็นคอลัมน์ที่อยู่ในตาราง ที่มีการเขียนเพิ่มในส่วนของชนิดข้อมูล และข้อกำหนดเพิ่มเติมอื่น ๆ

```
1  from django.db import models
2
3  # Create your models here.
4
5
6  class TEST_CREATE_TABLE(models.Model) :
7      COLUMN_A  = models.CharField(max_length=100,blank=True)
8      COLUMN_B  = models.DateField(auto_now =True)
9      COLUMN_C  = models.CharField(max_length=100,default='',blank=True, null=True)
10     COLUMN_D  = models.CharField(max_length=30 ,default='' )
11
12
```

รูปที่ 52 ตัวอย่างการเขียน class model

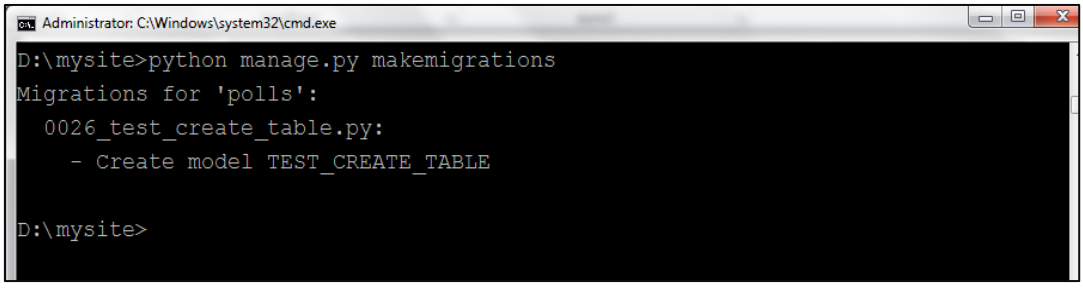
เมื่อเขียนคำสั่งสร้าง Class model แล้วจึงจะต้องทำให้ เว็บเซิร์ฟเวอร์ทำการสร้างตารางลงในฐานข้อมูลจริง ๆ ด้วยการรันคำสั่งผ่านหน้าจอ Command Prompt ซึ่งแสดงการรันคำสั่งผ่าน Command Prompt โดยเริ่มจากคำส่งชี้ไปยัง directory path ของข้อมูลเว็บเซิร์ฟเวอร์ จากนั้นทำ

การรันคำสั่ง ‘Python manage.py migrate’ หาไม่มีข้อผิดพลาดระบบจะสั่งให้รันคำสั่งถัดไปดังรูปที่ 53



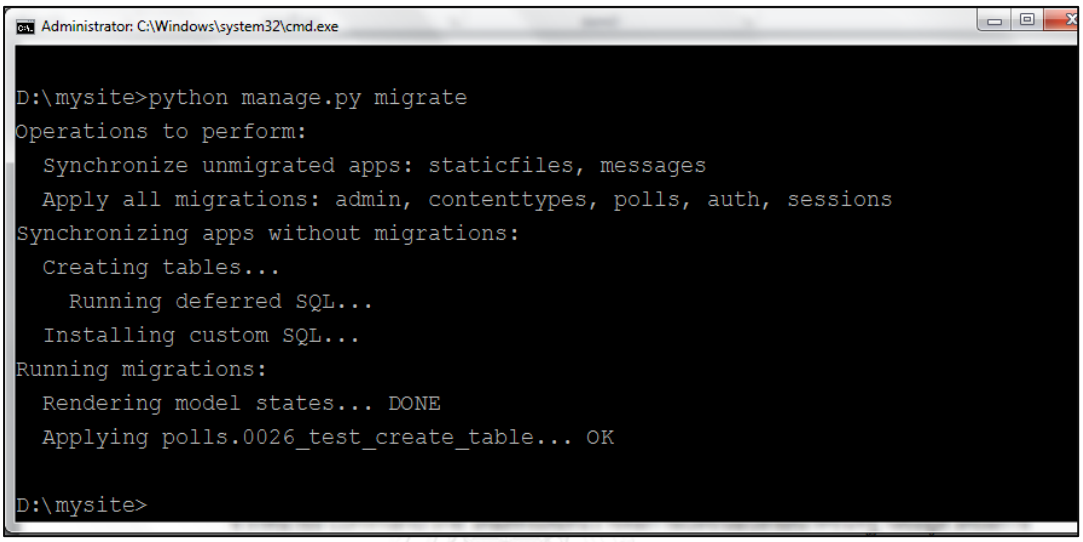
รูปที่ 53 ผลจากการรันคำสั่ง Python manage.py migrate

ในขั้นตอนการรันคำสั่ง ‘Python manage.py migrate’ เป็นพียงการตรวจสอบข้อผิดพลาดของระบบว่าสามารถสร้างตารางได้หรือไม่หากไม่มีข้อผิดพลาดที่หน้าจอ Command Prompt จะสั่งให้รันคำสั่งต้อมาคือคำสั่ง ‘Python manage.py makemigrations’ ผลจากการรันคำสั่งนี้คือ การตรวจสอบ class model ที่ผู้ใช้งานเขียนเอาไว้ว่าสั่งให้สร้างตารางหรือเพิ่มคอลัมน์ใดบ้าง หากไม่มีข้อผิดพลาดในการรันคำสั่งดังกล่าวหน้าจอ Command Prompt จะแสดงข้อสรุปในการสร้างตารางดังรูปที่ 54



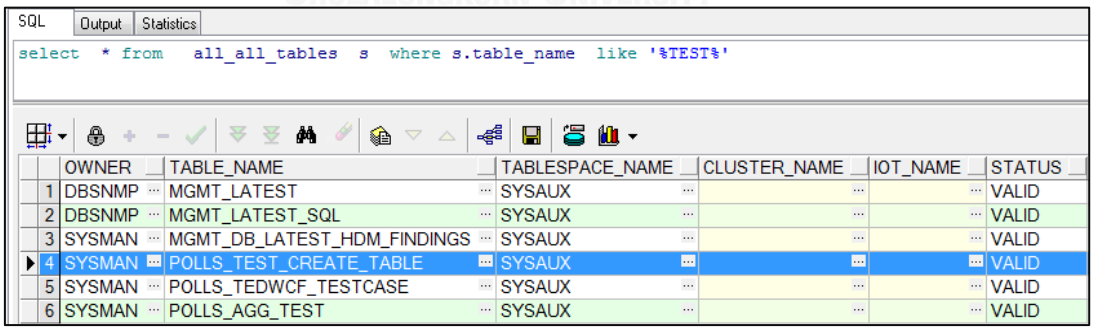
รูปที่ 54 ผลจากการรันคำสั่ง Python manage.py makemigrations

จากหน้าจอ Command Prompt ได้แสดงให้เห็นว่าจะมีการเปลี่ยนแปลงใดเกิดขึ้นที่ฐานข้อมูล โดยมีการเพิ่ม ตารางใหม่ชื่อ TEST_CREATE_TABLE ที่ผู้ใช้งานเขียนคำสั่งไว้ใน class model จากนั้นให้รันคำสั่ง 'Python manage.py migrate' อีกครั้งเพื่อดำเนินการสร้างตารางหรือเพิ่มคอลัมน์ใหม่ จากที่หน้าจอ Command Prompt ได้สรุปเอาไว้ หากไม่มีข้อผิดพลาดหน้าจอ Command Prompt จะแสดงผลลัพธ์ดังรูปที่ 55



รูปที่ 55 ผลจากการรันคำสั่ง Python manage.py migrate ครั้งที่ 2

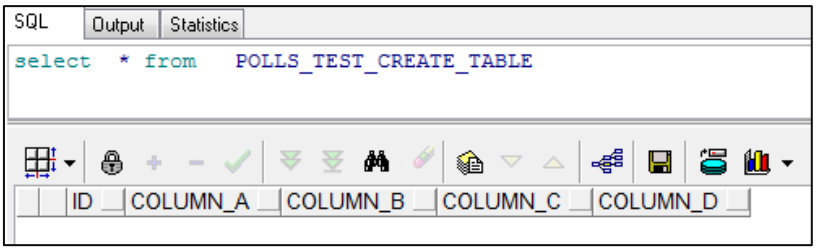
หลังจากนั้นผู้ใช้สามารถไปตรวจสอบตารางที่ฐานข้อมูลว่ามีการสร้างไว้จริงหรือไม่ โดยผู้วิจัยใช้วิธีติดต่อกับฐานข้อมูลผ่านทางโปรแกรม PL SQL developer เมื่อทำการใช้คำสั่งสืบค้นข้อมูลจากชื่อตาราง ก็จะปรากฏผลลัพธ์ดังรูปที่ 56 จะเห็นได้ว่าชื่อตารางจะมีการเติมคำนำหน้าซึ่งมาจากชื่อ application ที่เขียนไว้ในเว็บเซิร์ฟเวอร์



	OWNER	TABLE_NAME	TABLESPACE_NAME	CLUSTER_NAME	IOT_NAME	STATUS
1	DBSNMP	MGMT_LATEST	SYSAUX			VALID
2	DBSNMP	MGMT_LATEST_SQL	SYSAUX			VALID
3	SYSMAN	MGMT_DB_LATEST_HDM_FINDINGS	SYSAUX			VALID
4	SYSMAN	POLLS_TEST_CREATE_TABLE	SYSAUX			VALID
5	SYSMAN	POLLS_TEDWCF_TESTCASE	SYSAUX			VALID
6	SYSMAN	POLLS_AGG_TEST	SYSAUX			VALID

รูปที่ 56 ผลการสร้างตารางที่ฐานข้อมูลผ่านทาง class model

และเมื่อใช้คำสั่งสืบค้นข้อมูลเพื่อดูรายละเอียดภายในตารางจะได้ผลลัพธ์ตามที่เขียนคำสั่งไว้ที่ class model ดังรูปที่ 58



รูปที่ 57 ตารางที่สร้างผ่าน class model

```
-- Create table
create table POLLS_TEST_CREATE_TABLE
(
  ID       NUMBER(11) not null,
  COLUMN_A NVARCHAR2(100),
  COLUMN_B DATE not null,
  COLUMN_C NVARCHAR2(100),
  COLUMN_D NVARCHAR2(30)
)
```

รูปที่ 58 รายละเอียดภายในตารางที่สร้างผ่าน class model

จากรายละเอียดการสร้างตารางจะเห็นว่าคอลัมน์ต่าง ๆ ที่ถูกสร้างขึ้นมีค่าตามที่คำสั่งใน class model กำหนดไว้ คือ column_a, column_c, column_d ถูกกำหนดชนิดข้อมูลให้เป็น varchar ความยาวตามที่กำหนด และ column_c ถูกกำหนดให้เป็น date และห้ามเป็นค่า null ถือเป็นอันเสร็จสิ้นการสร้างตารางที่ฐานข้อมูลผ่านทาง class model ของ Django

ภาคผนวก ข

การติดตั้งโปรแกรมไพทอน

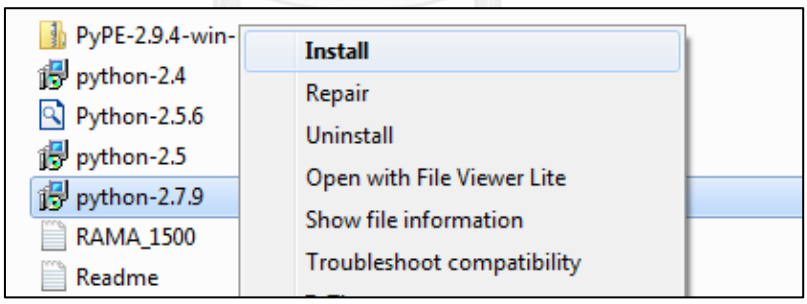
การติดตั้ง python version 2.7 และการติดตั้ง python package ด้วย pip โดยมีขั้นตอนดังต่อไปนี้

1. การติดตั้งไพทอน

ทำการดาวน์โหลดโปรแกรมจากURL : https://www.python.org/downloads/ โดยเลือก version 2.7

รูปที่ 59 เว็บไซต์สำหรับการดาวน์โหลดโปรแกรมติดตั้งไพทอน

เมื่อทำการดาวน์โหลดโปรแกรมมาแล้วให้ทำการติดตั้ง

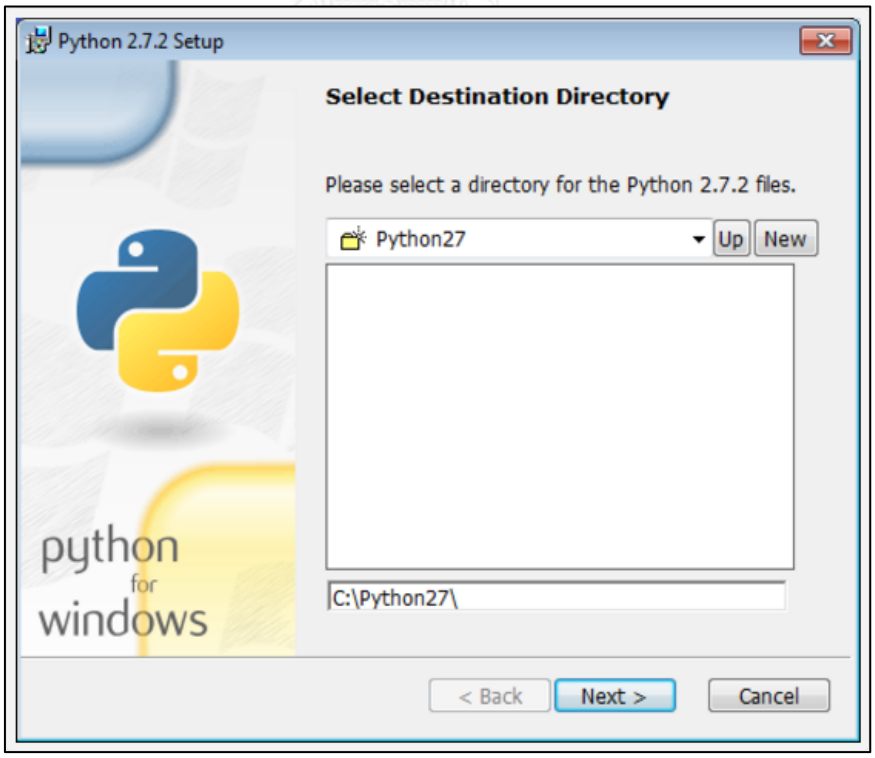


รูปที่ 60 การ install ไพทอน

เมื่อเลือก install จะปรากฏหน้าจอเริ่มต้นการติดตั้งดังรูปที่ 60 ให้เลือก install for all user จากนั้นเลือกปุ่ม Next เพื่อเข้าสู่ขั้นตอนถัดไป

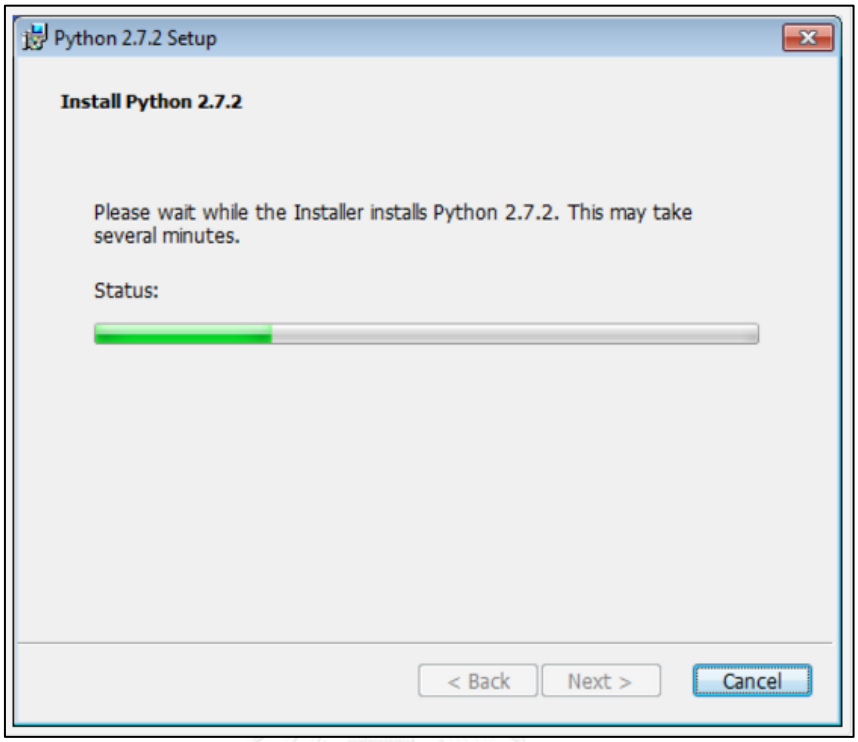
รูปที่ 61 หน้าจอเริ่มต้นการติดตั้งไพทอน

ทำการตั้งค่า directory path ที่ต้องการให้โปรแกรมติดตั้ง ตามรูปที่ 61 จากนั้นให้เลือกปุ่ม Next



รูปที่ 62 หน้าจอสำหรับเลือก python directory path

หลังจากเลือกปุ่ม next จะเข้าสู่การติดตั้งโปรแกรมดังรูปที่ 63

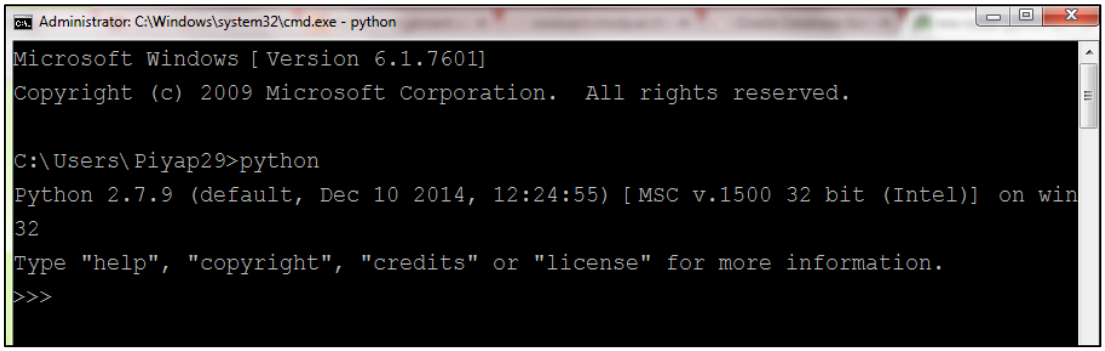


รูปที่ 63 หน้าจอแสดงความคืบหน้าการติดตั้ง

เมื่อทำการติดตั้งเสร็จสิ้นจะปรากฏหน้าจอแสดงการติดตั้งเสร็จสมบูรณ์ดังรูปที่ 64

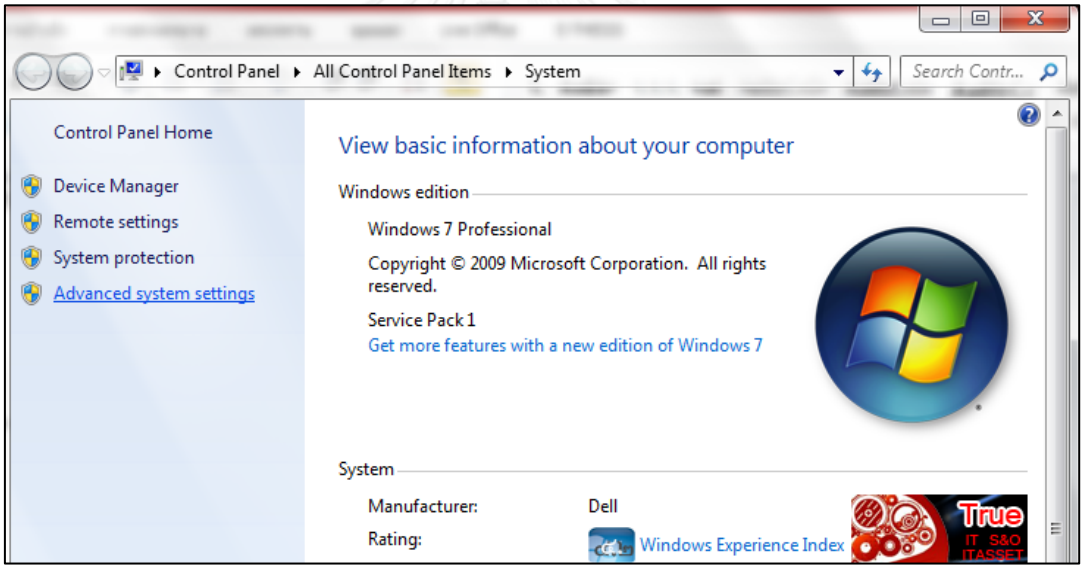
รูปที่ 64 หน้าจอแสดงการติดตั้งโปรแกรมเสร็จสมบูรณ์

การเช็คว่าโปรแกรมพร้อมใช้งานหรือไม่สามารถทำได้โดยการเปิด Command Prompt แล้วไปที่ python directory path ที่ตั้งค่าไว้เมื่อตอนเริ่มต้นการติดตั้งโปรแกรมจากนั้น พิมพ์คำสั่ง ‘python’ หากโปรแกรมพร้อมใช้งานจะปรากฏข้อความดังรูปที่ 65



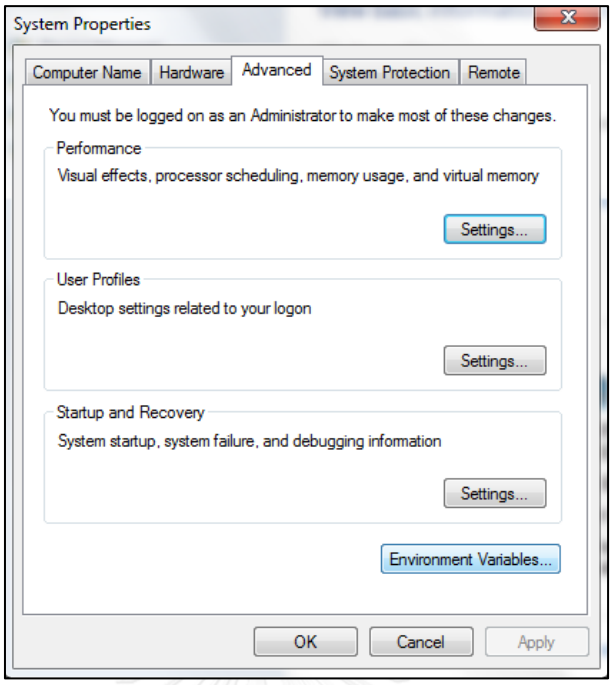
รูปที่ 65 หน้าจอแสดงผลการรัน Command Prompt

ทำการตั้งค่า path ที่ environment ของเครื่องโดยคลิกขวาที่ My computer แล้วเลือก Properties แล้วเลือกที่ Advanced system setting



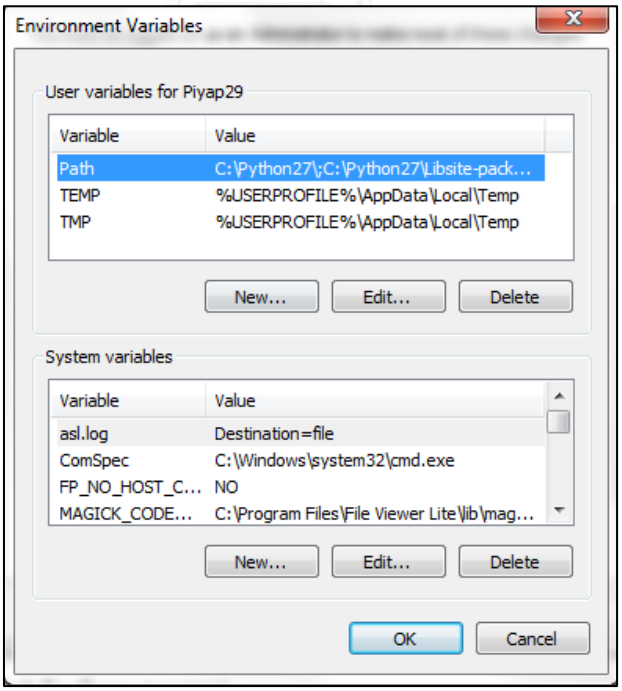
รูปที่ 66 หน้าจอแสดง My computer properties

เมื่อเลือก Advanced system setting จะปรากฏหน้าจอสำหรับการตั้งค่าต่าง ๆ ดังรูปที่ 66 ให้เลือกที่แถบเมนู Advanced และเลือก Environment Variable



รูปที่ 67 หน้าจอสำหรับตั้งค่า Advanced system setting

เมื่อเลือกที่ปุ่ม Environment Variable แล้วจะปรากฏหน้าจอสำหรับให้ทำการเพิ่ม Environment Variable ดังรูปที่ 67



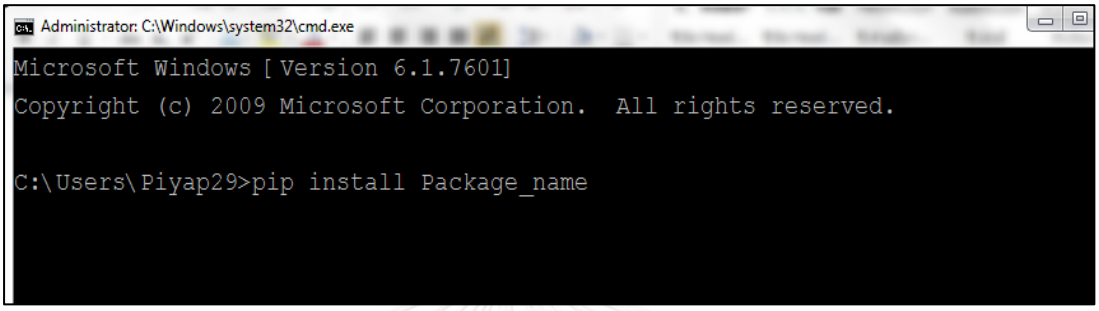
รูปที่ 68 หน้าจอสำหรับเพิ่ม Environment Variable

ให้ทำการแก้ไขตัวแปร path โดยให้เพิ่มค่า ‘python directory path\Python27’ เข้าไปต่อท้ายจากค่าที่มีอยู่เดิมเป็นการเสร็จสิ้นการติดตั้งโปรแกรมไพทอน

2. การติดตั้ง python package ด้วย pip

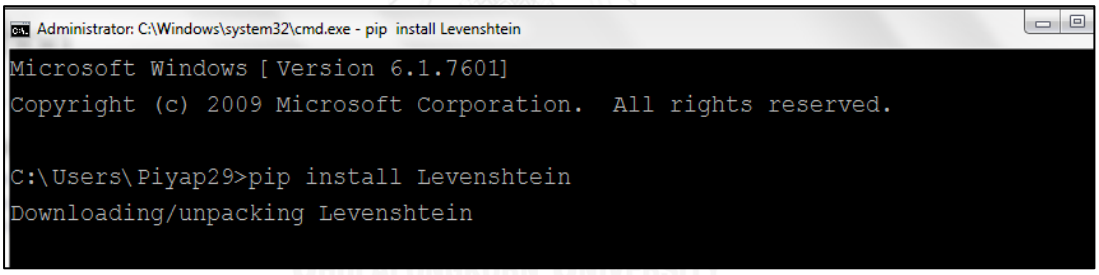
เริ่มต้นด้วยการตั้งค่า path ใน Environment Variable โดยเพิ่ม ‘python directory path \Python27\Libsite-packages; python directory path \Python27\Scripts;’ ต่อท้ายจากค่าเดิม

ทำการติดตั้ง package ด้วย pip โดยเปิด Command Prompt ขึ้นมา จากนั้นทำการพิมพ์คำสั่ง ‘pip install’ ตามด้วยชื่อ package ที่ต้องการ install ดังรูปที่ 69



รูปที่ 69 หน้าจอ Command Prompt สำหรับการ install python package

โดย package ที่ต้องทำการติดตั้งคือ 1.xlrd เพื่อใช้ในการจัดการ Excel และ 2. Levenshtein เพื่อใช้ในการหาระยะการแก้ไข ตัวอย่างการใช้ pip เพื่อ ติดตั้ง package เป็นดังรูปที่ 70

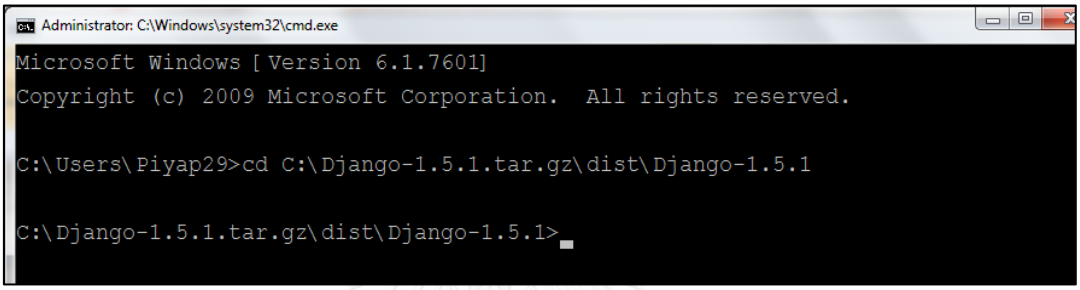


รูปที่ 70 ตัวอย่างการติดตั้ง python Levenshtein package ด้วย pip

ภาคผนวก ค
การติดตั้งกรอบงาน Django

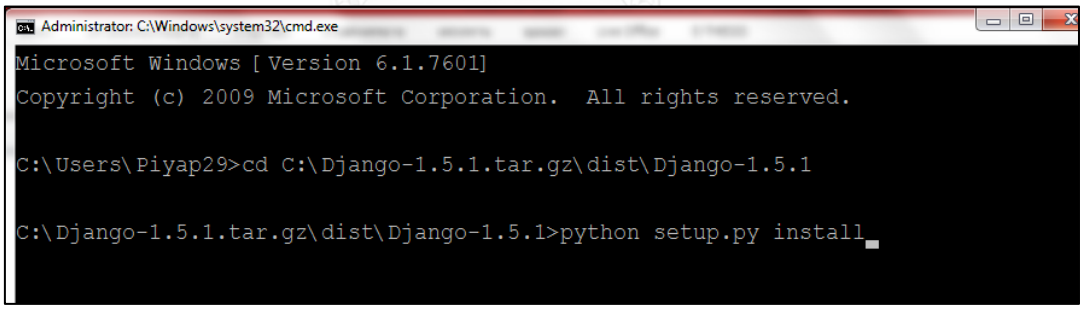
1. การติดตั้ง Django framework

ทำการดาวน์โหลดโปรแกรมติดตั้ง Django framework จาก url https://www.djangoproject .com/download/1.5.1/tarball/ ซึ่งไฟล์ที่ได้จากการดาวน์โหลดจะมีนามสกุลเป็น .tar.gz ให้นำไฟล์ที่ได้บันทึกไปที่ directory path ที่ต้องการซึ่งในงานวิจัยนี้ผู้วิจัยได้วางไฟล์ดังกล่าวไว้ที่ directory C:\ จากนั้นเปิด Command Prompt ขึ้นมาแล้วชี้ไปยัง directory C:\Django-1.5.1.tar.gz\dist\Django-1.5.1 ดังรูปที่ 71



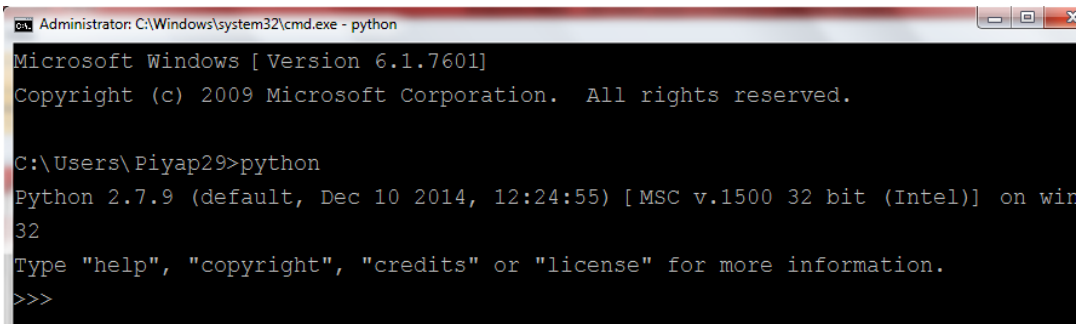
รูปที่ 71 หน้าจอ Command Prompt แสดง Django directory path

ทำการติดตั้ง Django framework ด้วยการพิมพ์คำสั่ง ‘python setup.py install’ ดังรูปที่ 72 จากนั้นรอให้การติดตั้งเสร็จ



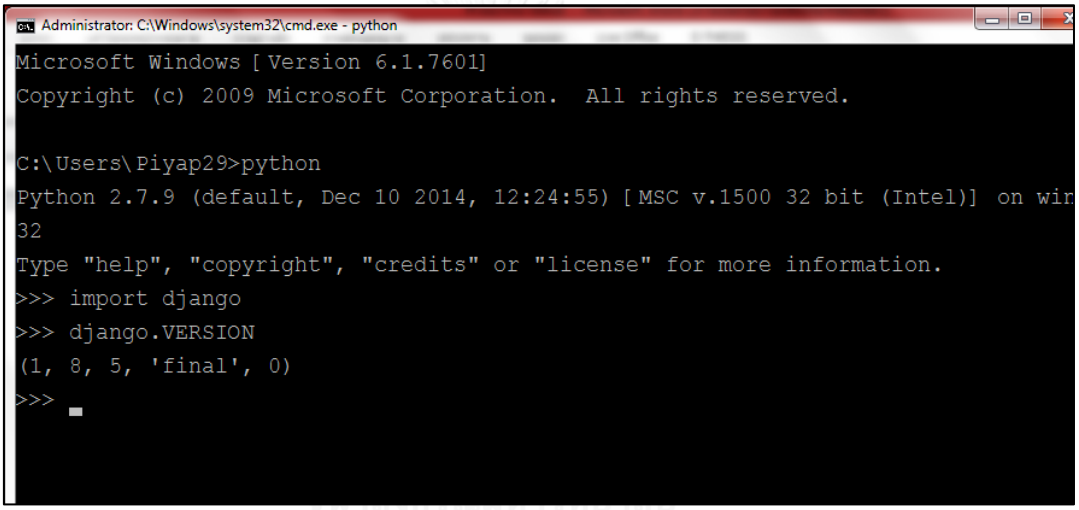
รูปที่ 72 หน้าจอ Command Prompt แสดงคำสั่งการติดตั้ง Django framework

เมื่อทำการติดตั้ง Django framework เสร็จแล้วให้ทำการตรวจสอบการติดตั้งอีกครั้งโดยการพิมพ์คำสั่ง ‘python’ ดังรูปที่ 73 จะปรากฏรายละเอียดเวอร์ชันของไพทอนที่ทำการติดตั้งไว้



รูปที่ 73 หน้าจอ Command Prompt แสดงผลจากคำสั่ง python

ทำการตรวจสอบ Django framework ด้วยการพิมพ์คำสั่ง ‘import django’ แล้วกดปุ่ม enter 1 ครั้ง จากนั้นพิมพ์คำสั่ง ‘django.VERSION’ แล้วกดปุ่ม enter 1 ครั้ง จะปรากฏรายละเอียดเวอร์ชันของ Django framework ที่ทำการติดตั้งดังรูปที่ 74



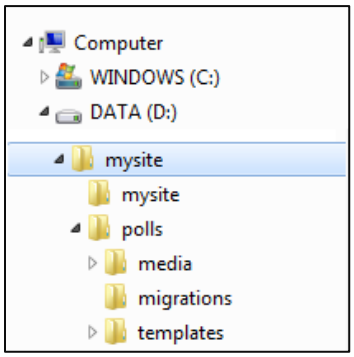
รูปที่ 74 หน้าจอ Command Prompt แสดงเวอร์ชันของ Django framework

ในกรณีที่ไม่แสดง version หรือพบ error message หรือ error code ให้ทบทวนวิธีการติดตั้งใหม่อีกครั้ง แต่สำหรับใครก็ตามที่แสดง version นั้นหมายถึงการติดตั้ง Django framework สำเร็จเรียบร้อยแล้ว พร้อมสำหรับการสร้าง project และการรัน server เพื่อเขียนเว็บไซต์ หรือทำเว็บไซต์ด้วยภาษาไพทอน

2. การเริ่มต้นสร้าง Project ด้วย Django framework

2.1 สร้างชื่อ Folder ที่ผู้ใช้งานต้องการใน directory path ใดๆ โดยไม่จำเป็นจะต้องอยู่ใน Django Folder โดยชื่อ Folder ที่จะทำการสร้างเปรียบเสมือนว่าเป็นชื่อ Project ในงานวิจัยนี้ทำการสร้าง Folder Project ชื่อ mysite ใน Drive D:

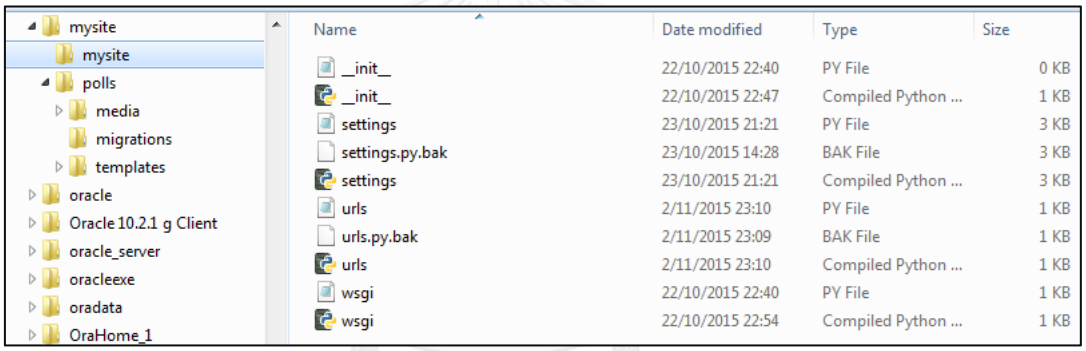
2.2 เปิด Command Prompt แล้วรันคำสั่งย้ายการทำงานไปที่ directory D:\mysite จากนั้นรันคำสั่ง ‘django-admin.py startproject mysite’ จะได้ project directory แสดงดังรูปที่ 75



รูปที่ 75 project mysite directory

โดย mysite คือชื่อของเว็บไซต์ภายใน Project mysite สามารถเปลี่ยนชื่อเป็นชื่อเว็บไซต์ใด ๆ ก็ได้ตามที่ต้องการ

2.3 หลังจากรันคำสั่งแล้วเมื่อเข้าไปใน Folder mysite> mysite จะพบโครงสร้างของ Directory ดังรูปที่ 76



รูปที่ 76 mysite website directory

2.4 ตรวจสอบความถูกต้องของ directory และไฟล์ต่าง ๆ เป็นอันเสร็จสิ้นการสร้าง project ด้วย Django Framework

ประวัติผู้เขียนวิทยานิพนธ์

นางสาวปิยภรณ์ สามสุวรรณ์ เกิดเมื่อวันที่ 15 เมษายน พ.ศ. 2531 ที่จังหวัดนครศรีธรรมราช สำเร็จการศึกษาปริญญาตรีหลักสูตรวิศวกรรมศาสตรบัณฑิต (วศ.บ) สาขาวิชาวิศวกรรมสารสนเทศ ภาควิชาวิศวกรรมสารสนเทศ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังในปีการศึกษา 2553 และ เข้าศึกษาต่อในหลักสูตรวิทยาศาสตรมหาบัณฑิต (วท.บ) ที่ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2556