# เอเจนต์หลายแพลตฟอร์มแบบติดตั้งเพิ่มเติมเพื่อให้เป็นฐานข้อมูลรีเลชันนอลแบบแอกทิฟ

# Multiplatform Add-On Agent for Active Relational Database Systems

ภัทรกิติ ไชยสิงห์ (Pattarakitti Chaiyasing)* วิโรจน์ ทวีปวรเดช (Wiroj Taweepworadej)[1]**

**บทคัดย่อ**

บทความนี้นำเสนอโมเดลเอเจนต์หลายแพลตฟอร์มแบบติดตั้งเพิ่มเติม เพื่อให้ฐานข้อมูลรีเลชันนอลที่มีการทำงานแบบแพซซิฟ เปลี่ยนเป็นฐานข้อมูลที่มีการทำงานแบบแอกทิฟ ที่สามารถตรวจจับเหตุการณ์ที่สนใจและตอบสนองต่อผู้ใช้ได้โดยอัตโนมัติ โมเดลที่นำเสนอถูกทดสอบกับฐานข้อมูล 3 แพลตฟอร์ม ได้แก่ MySQL, Oracle และ PostgreSQL ผู้วิจัยได้พัฒนาชุดคำสั่งเพื่อทดสอบการตรวจจับข้อมูลชนิดตัวเลข ที่มีเงื่อนไขในการทดสอบ คือ มากกว่า น้อยกว่า และอยู่ในช่วง ข้อมูลที่ใช้ทดสอบได้จากการสุ่มตัวเลขให้มีค่าที่เป็นและไม่เป็นไปตามเงื่อนไขที่ต้องการทดสอบ การทดสอบใช้รูปแบบการตรวจจับทั้งก่อนและหลังการปรับปรุงข้อมูล โดยใช้จำนวนข้อมูล 100 รายการในแต่ละกรณี ซึ่งพบว่าการทำงานของเอเจนต์ที่พัฒนามีผลการทำงานถูกต้อง 100 เปอร์เซนต์ในทุกกรณี

**ABSTRACT**

This paper has proposed the multiplatform add-on agent model for changing the passive RDBS to be active. The proposed agent can detect a predefined event and automatically perform a proper response. The model was designed to work on 3 platforms including MySQL, Oracle and PostgreSQL. The testing script was implemented in order to detect events in both before and after the data update within 3 criteria such as greater than, less than and in between. The testing data was randomly set up for 100 dataset each to fit and unfit the desired criteria. The result showed that the proposed agent has a 100 percent corrects in all cases.

**คำสำคัญ:** โมเดลแบบหลายแพลตฟอร์ม เอเจนต์แบบติดตั้งเพิ่มเติม ระบบฐานข้อมูลรีเลชันนอลแบบแอกทิฟ

**Key Words:** Multiplatform Model, Add-On Agent, Active Relational Database Systems

[1] *Correspondent author: wirtaw@kku.ac.th*

** นักศึกษา หลักสูตรวิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยขอนแก่น*

*** รองศาสตราจารย์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยขอนแก่น*

**บทนำ**

ระบบฐานข้อมูลส่วนใหญ่ที่มีการใช้งานเป็นฐานข้อมูลรีเลชันนอล ที่มีลักษณะการทำงานเป็นแบบแพซซิฟ (passive) กล่าวคือ ฐานข้อมูลจะมีการกระทำก็ต่อเมื่อมีคำสั่งต่าง ๆ จากผู้ใช้งาน หรือจากคำสั่งของโปรแกรม (Dayal, Hanson and Widom, 1994) เช่น การเพิ่มข้อมูล การสืบค้น การปรับปรุง และการลบข้อมูล เป็นต้น ทำให้การใช้งานระบบฐานข้อมูลขาดความยืดหยุ่น นักวิจัยได้พัฒนาระบบฐานข้อมูลแบบรีเลชันนอลให้สามารถตอบสนองต่อเหตุการณ์ได้ทันทีตามเงื่อนไขที่กำหนดไว้ล่วงหน้า เรียกว่าระบบฐานข้อมูลรีเลชันนอลแบบแอกทิฟ (Active Relational Database System) (Dittrich, Gatziu and Geppert, 1995) กล่าวคือ เมื่อมีเหตุการณ์ ๆ หนึ่งเกิดขึ้นในระบบฐานข้อมูล ซึ่งเป็นเหตุการณ์ที่ตรงตามเงื่อนไขที่กำหนดไว้ ระบบฐานข้อมูลก็จะทำตามกระบวนการที่ได้กำหนดไว้ เช่น ระบบคลังสารเคมีที่มีส่วนตรวจสอบการเบิกสารเคมี ที่สามารถทำการระงับการเบิกสารเคมี ถ้าพบว่า หลังการเบิกสารเคมี ปริมาณสารเคมีมีจำนวนน้อยกว่าที่กำหนด (Sasithorn, Witcha and Wiroj, 2010) หรือในด้านตลาดหุ้นที่ราคาหุ้นมีการเปลี่ยนแปลงอยู่ตลอดเวลา จะมีระบบซื้อขายหุ้นที่จะดำเนินการซื้อขายในทันทีที่พบว่าหุ้นมีราคาต่ำสุดหรือสูงสุดตามราคาที่เสนอซื้อหรือขายโดยอัตโนมัติ เป็นต้น ส่งผลให้ระบบการจัดการฐานข้อมูลจำเป็นต้องพัฒนาให้มีศักยภาพในการทำงานแบบแอกทิฟ

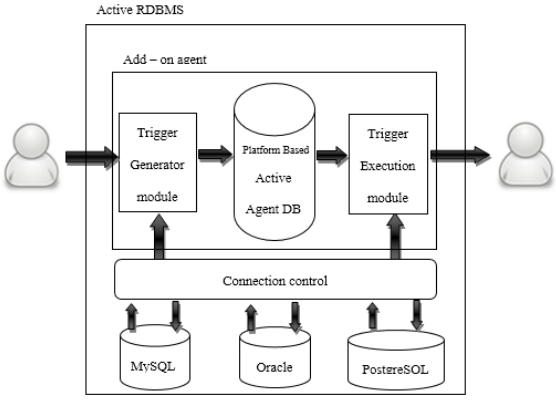
มีหลายงานวิจัยที่ศึกษาเรื่องฐานข้อมูลแบบแอกทิฟในด้านต่าง ๆ เช่น ความพยายามที่จะจัดการกับการเปลี่ยนแปลงกฎที่มีในฐานข้อมูลแบบแอกทิฟให้มีประสิทธิภาพมากขึ้น เพื่อไม่ให้ส่งผลกระทบต่อโปรแกรมที่ใช้อยู่ในปัจจุบัน (Viana, Pav´on and Rady, 2006) และการพัฒนาขยายความสามารถของกฎของฐานข้อมูลและวิธีการตอบสนองต่อเหตุการณ์ ให้ทริกเกอร์สามารถสร้างเงื่อนไขในการตรวจจับได้ซับซ้อนมากยิ่งขึ้นและสามารถตอบสนองต่อเหตุการณ์ได้หลากหลายรูปแบบมากขึ้น (Nellainayagam, 2002)

ในขณะที่งานวิจัยส่วนใหญ่ศึกษาการพัฒนาขีดความสามารถของกฎการเรียนรู้และทริกเกอร์ Sasithorn *et al.* (2010) ศึกษาและพัฒนาโมเดลการทำงานของแอกทิฟโมดูลในรูปของเอเจนต์ที่สามารถติดตั้งเพิ่มเติม (Add-on) ในฐานข้อมูลแบบรีเลชันนอล เพื่อทำให้ระบบฐานข้อมูลแบบรีเลชันนอลมีลักษณะการทำงานเป็นแบบแอกทิฟ โดยออกแบบระบบการจัดการทริกเกอร์ ในรูปของ GUI ที่สะดวกต่อการใช้งาน อย่างไรก็ตาม แอกทิฟเอเจนต์ในงานวิจัยนี้ มีการทำงานเป็นแบบทริกเกอร์เดียว (Single Trigger) ที่สามารถทำงานได้บนแพลตฟอร์ม MySQL 5.0 เท่านั้น พรศักดิ์ ขวากุพันธ์ (2554) ได้พัฒนาอัลกอริทึมให้ระบบฐานข้อมูลรีเลชันนอลแบบแอกทิฟสามารถรองรับการทำงานของทริกเกอร์ หลายตัวบนตารางข้อมูลเดียวกันได้ โดยอาศัยแนวคิดการเปิด-ปิดการทำงานของทริกเกอร์ตามลำดับความสำคัญของทริกเกอร์แต่ละตัว แต่ในงานวิจัยนี้ได้พัฒนาโดยใช้โปรแกรมเข้ามาควบคุมการเปิด-ปิดทริกเกอร์ ทำให้ไม่เป็นอิสระต่อการใช้งานและไม่สามารถนำไปติดตั้งกับระบบอื่น ๆ ได้

งานวิจัยนี้มีแนวคิดในการศึกษาพัฒนาเอเจนต์ที่สามารถติดตั้งเพิ่มเติมในฐานข้อมูลรีเลชันนอลแบบแพซซิฟหลายแพลตฟอร์ม เพื่อทำให้ระบบฐานข้อมูลรีเลชันนอลดังกล่าวมีลักษณะการทำงานเป็นแบบแอกทิฟ โดยแอกทิฟเอเจนต์ที่พัฒนาสามารถเรียนรู้คำสั่งในการติดต่อระหว่างโปรแกรมกับฐานข้อมูลสกีมา และคำสั่งที่ใช้ในการสร้างทริกเกอร์ของแต่ละแพลตฟอร์มที่แตกต่าง ให้สามารถเชื่อมต่อระหว่างโปรแกรมกับระบบการจัดการฐานข้อมูลของแต่ละแพลตฟอร์มให้อยู่ในรูปแบบเดียวกัน เพื่อรองรับการทำงานบนระบบการจัดการฐานข้อมูลในแพลตฟอร์มที่ต่างกัน

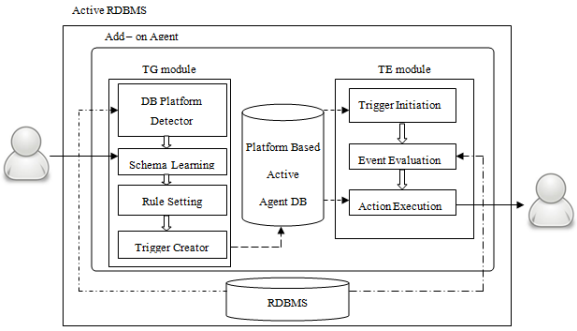
## โมเดลเอเจนต์หลายแพลตฟอร์มแบบติดตั้งเพิ่มเติม

เพื่อให้ฐานข้อมูลแบบรีเลชันนอลที่มีการทำงานแบบแพซซิฟต่างแพลตฟอร์ม มีการเปลี่ยนแปลงการทำงานเป็นแบบแอกทิฟ ผู้วิจัยได้ออกแบบโมเดลแอกทิฟเอเจนต์ที่ประกอบด้วยโมดูลการสร้างทริกเกอร์ (Trigger Generator module) โมดูลการประมวลผลทริกเกอร์ (Trigger Execution) และ ส่วนการควบคุมการติดต่อ (Connection Control) แสดงดังรูปที่ 1 โมดูลการสร้างทริกเกอร์จะทำการเตรียมข้อมูลพื้นฐานและสร้างทริกเกอร์ ตามเงื่อนไขการตรวจจับและวิธีการตอบสนองของทริกเกอร์ที่ผู้ใช้เป็นผู้กำหนด ในขณะที่โมดูลการประมวลผลทริกเกอร์จะเริ่มการทำงานของทริกเกอร์ เพื่อตรวจจับเหตุการณ์ เช่น การเพิ่มหรือการปรับปรุงข้อมูล และเมื่อพบเหตุการณ์ที่เป็นไปตามเงื่อนไขที่กำหนด โมดูลนี้จะทำการบันทึกผลการตรวจจับลงในฐานข้อมูลแอกทิฟเอเจนต์ เพื่อแสดงผลการตรวจจับต่อผู้ใช้งาน และส่วนควบคุมการติดต่อจะทำหน้าที่ตรวจสอบแพลตฟอร์มของระบบจัดการฐานข้อมูล แล้วเอเจนต์จะเรียนรู้คำสั่งในการติดต่อระหว่างโปรแกรมกับฐานข้อมูล สกีมา และคำสั่งที่ใช้ในการสร้างทริกเกอร์ตามแพลตฟอร์มของระบบจัดการฐานข้อมูลที่ใช้




**รูปที่ 1** โมเดลเอเจนต์หลายแพลตฟอร์มแบบติดตั้งเพิ่มเติม

รูปที่ 2 แสดงส่วนประกอบภายในโมเดลแอกทิฟเอเจนต์ โดยโมดูลการสร้างทริกเกอร์ประกอบด้วยส่วนสำคัญ 4 ส่วน ได้แก่ ส่วนตรวจจับแพลตฟอร์มฐานข้อมูล (Database Platform Detector) ทำการเรียนรู้แพลตฟอร์มของระบบจัดการฐานข้อมูล เพื่อให้เอเจนต์มีการเรียนรู้คำสั่งในการติดต่อระหว่างโปรแกรมกับฐานข้อมูล สกีมา และคำสั่งที่ใช้ในการสร้างทริกเกอร์ตามแพลตฟอร์มของฐานข้อมูลที่ตรวจพบ ส่วนการเรียนรู้สกีมา (Schema Learning) ทำหน้าที่เรียนรู้โครงสร้างข้อมูลของฐานข้อมูล เพื่อใช้ในการกำหนดค่าในการตรวจจับของทริกเกอร์ ส่วนกำหนดค่าในการตรวจจับ (Rule Setting) เป็นส่วนที่ผู้ใช้กำหนดเงื่อนไขการตรวจจับและวิธีการตอบสนองของทริกเกอร์ และส่วนการสร้างทริกเกอร์ (Trigger Creator) จะนำค่าการตรวจจับที่ผู้ใช้กำหนดมาสร้าง ทริกเกอร์และบันทึกข้อมูลของทริกเกอร์ลงในฐานข้อมูลแอกทิฟเอเจนต์ตามแพลตฟอร์มที่ติดตั้ง




**รูปที่ 2** โครงสร้างภายในโมเดลเอเจนต์หลายแพลตฟอร์มแบบติดตั้งเพิ่มเติม

โมดูลการประมวลผลทริกเกอร์ประกอบด้วยส่วนสำคัญ 3 ส่วน ได้แก่ ส่วนเริ่มการทำงานของทริกเกอร์ (Trigger Initiation) เป็นส่วนที่เอเจนต์จะเริ่มการทำงานของทริกเกอร์ โดยดึงข้อมูลการตรวจจับของ ทริกเกอร์จากฐานข้อมูลแอกทิฟเอเจนต์ และเมื่อส่วนประเมินเหตุการณ์ (Event Evaluation) ตรวจพบเหตุการณ์ที่เกิดขึ้นกับฐานข้อมูลที่เป็นไปตามเงื่อนไขที่กำหนด จะติดต่อกับส่วนประมวลผลการกระทำ (Action

Execution) ให้ดึงข้อมูลวิธีการตอบสนองของทริกเกอร์จากฐานข้อมูลแอกทิฟเอเจนต์ เพื่อแสดงผลไปยังผู้ใช้

## สถาปัตยกรรมระบบ

เอเจนต์หลายแพลตฟอร์มแบบติดตั้งเพิ่มเติมถูกออกแบบและพัฒนาเป็น web application โดยใช้ภาษา php เวอร์ชัน 5.2.6 และใช้ระบบจัดการฐานข้อมูล 3 แพลตฟอร์มในการทดสอบ ได้แก่ MySQL 5.0.51b, Oracle 11g Express Edition และ PostgreSQL 9.2 ผู้วิจัยได้ออกโมเดลสถาปัตยกรรมระบบโดยมีส่วนควบคุมการเชื่อมต่อเพื่อทำหน้าที่จัดการการเชื่อมต่อระหว่าง web application และระบบจัดการฐานข้อมูลแต่ละแพลตฟอร์ม โดยเอเจนต์ที่พัฒนานี้สามารถติดตั้งกับระบบจัดการฐานข้อมูลบน 3 แพลตฟอร์ม ได้แก่ MySQL, Oracle และ PostgreSQL

## ขั้นตอนการทดลอง

ผู้วิจัยได้ออกแบบและสร้างฐานข้อมูลเพื่อใช้ในการทดสอบโมเดลที่พัฒนาขึ้น ประกอบด้วยตารางข้อมูลที่ใช้ทดสอบจำนวน 7 ตาราง โดยที่แต่ละตารางประกอบไปด้วยฟิลด์ข้อมูลจำนวนอย่างน้อย 10 ฟิลด์ ที่เป็นฟิลด์ข้อมูลชนิดข้อความ ตัวอักษร วันที่ และตัวเลข ในการทดลองนี้ได้สร้างทริกเกอร์เพื่อตรวจจับฟิลด์ที่มีชนิดข้อมูลแบบตัวเลข และทดสอบการทำงานของ ทริกเกอร์ โดยแบ่งการตรวจจับเป็น 2 แบบ คือ การตรวจจับที่ 1 ตรวจจับก่อนการดำเนินการ (before) และการตรวจจับที่ 2 ตรวจจับหลังการดำเนินการ (after) โดยที่แต่ละแบบสามารถแยกได้เป็น 2 กรณีย่อย คือ กรณีที่ 1 กำหนดทริกเกอร์ตรวจจับทั้งตาราง และกรณีที่ 2 คือให้ทริกเกอร์ตรวจจับเฉพาะเรคคอร์ดในตาราง โดยที่เงื่อนไขในการตรวจจับทริกเกอร์ที่ทดสอบมีทั้งหมด 3 เงื่อนไข ได้แก่

1) เงื่อนไขที่ 1 ฟิลด์ที่ตรวจจับมีค่าน้อยกว่า 500
2) เงื่อนไขที่ 2 ฟิลด์ที่ตรวจจับมีค่ามากกว่า 1000
3) เงื่อนไขที่ 3 ฟิลด์ที่ตรวจจับมีค่าอยู่ในช่วง 500 – 1000

ในการทดสอบแต่ละเงื่อนไข ผู้วิจัยได้พัฒนาชุดคำสั่งให้ทำการสุ่มตัวเลข 50 ชุด ให้มีค่าอยู่ในช่วงที่ต้องการทดสอบ แล้วทำการปรับปรุงข้อมูลในตารางข้อมูลที่ทดสอบโดยใช้ตัวเลขที่ทำการสุ่มได้ ซึ่งในการทดสอบแต่ละเงื่อนไขจะมีรายการทดสอบปรับปรุงข้อมูล 100 ชุด ประกอบด้วย

1) รายการ A คือ รายการที่ไม่ตรงตามเงื่อนไข จำนวน 50 ชุด
2) รายการ B คือ รายการที่เป็นไปตามเงื่อนไข จำนวน 50 ชุด

## ผลการทดลองและอภิปรายผล

ผลการทำงานของทริกเกอร์ที่สร้างขึ้นจากเอเจนต์หลายแพลตฟอร์มแบบติดตั้งเพิ่มเติม ในการตรวจจับฟิลด์ของข้อมูลทั้งตาราง และตรวจจับฟิลด์เฉพาะเรคคอร์ดที่กำหนดในตาราง แสดงดังตารางที่ 1 และ 2 ตามลำดับ ซึ่งพบว่ามีความถูกต้อง 100 เปอร์เซนต์ในทุกกรณี ไม่ว่าจะเป็นการตรวจจับก่อนการดำเนินการหรือการตรวจจับหลังการดำเนินการ ทั้งในกรณีที่มีเงื่อนไขการตรวจจับเป็น มากกว่า น้อยกว่า หรืออยู่ในช่วง ซึ่งการตรวจจับก่อนดำเนินการนั้น เมื่อทริกเกอร์ตรวจพบเหตุการณ์และเป็นไปตามเงื่อนไขที่กำหนด ทริกเกอร์จะทำการบันทึกผลการตรวจจับลงฐานข้อมูลเอเจนต์และจบการทำงานโดยไม่มีการปรับปรุงข้อมูลตามที่ผู้ใช้ร้องขอ แต่ในกรณีที่เป็นการตรวจจับหลังดำเนินการ ฐานข้อมูลจะทำการปรับปรุงข้อมูลตามที่ผู้ใช้ร้องขอก่อน ทริกเกอร์จึงจะทำการตรวจสอบเหตุการณ์ และหากเป็นไปตามเงื่อนไขที่กำหนด ทริกเกอร์จะทำการบันทึกผลการตรวจจับลงฐานข้อมูลเอเจนต์และจบการทำงาน

## สรุปและข้อเสนอแนะ

ในงานวิจัยนี้ผู้วิจัยได้นำเสนอโมเดลเอเจนต์เพื่อพัฒนาโปรแกรมเอเจนต์ที่สามารถติดตั้งเพิ่มเติม ในฐานข้อมูลรีเลชันนอลแบบแพซซิฟต่างแพลตฟอร์ม เพื่อทำให้ระบบฐานข้อมูลรีเลชันนอลมีลักษณะการทำงานเป็นแบบแอกทิฟ โดยที่ไม่มีการเปลี่ยนแปลงโครงสร้างฐานข้อมูลและระบบงานเดิม โปรแกรมเอเจนต์ที่พัฒนาสามารถติดตั้งได้กับระบบจัดการฐานข้อมูล 3 แพลตฟอร์ม ได้แก่ MySQL, Oracle และ PostgreSQL โดยรองรับการเปิดการทำงานของทริกเกอร์ได้หลายตัวพร้อมกัน ภายใต้ข้อจำกัดที่ว่า ทริกเกอร์ที่เปิดนั้นจะต้องไม่ตรวจจับข้อมูลในตารางเดียวกัน และจะเน้นตรวจจับฟิลด์ที่เป็นฟิลด์ตัวเลข โดยเงื่อนไขที่นำมาใช้ทดสอบคือ เงื่อนไข มากกว่า น้อยกว่า และ อยู่ในช่วง โดยใช้เพียงเงื่อนไขเดียวในการทดสอบการทำงานของระบบ อย่างไรก็ตาม โมเดลเอเจนต์หลายแพลตฟอร์มแบบติดตั้งเพิ่มเติมที่เสนอแนะจำเป็นต้องได้รับการพัฒนาให้เป็นโมเดลของเอเจนต์แบบหลายทริกเกอร์ (Multitrigger Agent) ที่สามารถควบคุมการทำงานของทริกเกอร์ได้หลายตัวพร้อมกันสำหรับหนึ่งตาราง โดยการสร้างส่วนควบคุมให้สามารถเปิด-ปิดการทำงานของทริกเกอร์ โดยเลือกเปิดการทำงานทริกเกอร์เฉพาะเมื่อตรวจสอบพบว่าตรงตามเงื่อนไขที่กำหนดเท่านั้น

## เอกสารอ้างอิง

พรศักดิ์ ขวากุพันธ์. 2554. การพัฒนาระบบฐานข้อมูลเชิงสัมพันธ์แบบแอกทิฟ. วิทยานิพนธ์ปริญญาวิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์ บัณฑิตวิทยาลัย มหาวิทยาลัยขอนแก่น.

Abbas, R., Rohollah, A., and Ahmad, A. 2006. A new approach for event triggering probability estimation in Active Database Systems to rule scheduling improvement. IEEE Information and Communication Technologies, 2006. ICTTA '06. 2nd. (pp. 2920 – 2925). Damascus: IEEE.

Alesheykh, R., and Abdollahzadeh, A. 2005. Evaluation of Shortest Job First Approach for Rule Scheduling in Active Database Systems. Proceedings of the 9th IASTED International Conference on Artificial Intelligence & Soft Computing (ASC'05). Spain: Benidorm.

Dai, M., and Huang, LY. 2007. Data Mining used in rule design for Active Database Systems. Paper presented at Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), Haikou, Chaina.

Dayal, U., Hanson, EN., and Widom, j. 1994. Active Database Systems. Modern Database Systems: The Object Model, Interoperability, and Beyond, Addison-Wesley, Reading, Massachusetts.

Dittrich, KR., Gatziu, S., and Geppert, A. 1995. The Active Database Management System Manifesto: A Rulebase of ADBMS Features. Sellis (ed.), Proc. 2nd Workshop on Rules in Databases (RIDS), Athens, Greece,. Lecture Notes in Computer Science, Springer 1995.

Moraes, AC., Salgado, AC., and Tedesco, PA. 2009. AutonomousDB: A Tool for Autonomic Propagation of Schema Updates in Heterogeneous Multi-database Environments. This paper appears in: Autonomic and Autonomous Systems, 2009. ICAS '09. Fifth International Conference.

Nellainayagam, C. 2002. A Mediator Based Approach to Support ECA Rules in DB2. University of Texas at Arlington, 2002.

Sasithorn, S., Witcha F., and Wiroj, T. 2010. Add-On Agent for Active Relational Database Case Study: Faculty's Chemical Storage Database. Proceedings of the 25th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC 2010). (pp.1049-1052), July 2010.

Viana S., Pav´on J., and Rady de Almeida Junior, J. 2006. Rule Mangement in Active Database Systems. Proceedings of the 15th International Conference on Computing (CIC'06). (pp.315-322), Mexico: Mexico City.

**ตารางที่ 1** ผลการทำงานของทริกเกอร์ แบบตรวจจับฟิลด์ทุกเรคคอร์ดในตาราง

| ชนิดฐานข้อมูล | ลำดับตรวจจับ | เงื่อนไข 1 | | | | เงื่อนไข 2 | | | | เงื่อนไข 3 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | รายการ | ผลการตรวจจับ | การแก้ไขข้อมูล | ความถูกต้อง | รายการ | ผลการตรวจจับ | การแก้ไขข้อมูล | ความถูกต้อง | รายการ | ผลการตรวจจับ | การแก้ไขข้อมูล | ความถูกต้อง |
| MySQL | before | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% |
| | | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% |
| | after | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% |
| | | B | ตรวจพบ 50 | แก้ไข 50 | 100% | B | ตรวจพบ 50 | แก้ไข 50 | 100% | B | ตรวจพบ 50 | แก้ไข 50 | 100% |
| Oracle | before | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% |
| | | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% |
| | after | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% |
| | | B | ตรวจพบ 50 | แก้ไข 50 | 100% | B | ตรวจพบ 50 | แก้ไข 50 | 100% | B | ตรวจพบ 50 | แก้ไข 50 | 100% |
| PostgreSQL | before | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% |
| | | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% |
| | after | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% |
| | | B | ตรวจพบ 50 | แก้ไข 50 | 100% | B | ตรวจพบ 50 | แก้ไข 50 | 100% | B | ตรวจพบ 50 | แก้ไข 50 | 100% |

**ตารางที่ 2** ผลการทำงานของทริกเกอร์ แบบตรวจจับฟิลด์เฉพาะเรคคอร์ดในตาราง

| ชนิดฐานข้อมูล | ลำดับตรวจจับ | เงื่อนไข 1 | | | | เงื่อนไข 2 | | | | เงื่อนไข 3 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | รายการ | ผลการตรวจจับ | การแก้ไขข้อมูล | ความถูกต้อง | รายการ | ผลการตรวจจับ | การแก้ไขข้อมูล | ความถูกต้อง | รายการ | ผลการตรวจจับ | การแก้ไขข้อมูล | ความถูกต้อง |
| MySQL | before | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% |
| | | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% |
| | after | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% |
| | | B | ตรวจพบ 50 | แก้ไข 50 | 100% | B | ตรวจพบ 50 | แก้ไข 50 | 100% | B | ตรวจพบ 50 | แก้ไข 50 | 100% |
| Oracle | before | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% |
| | | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% |
| | after | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% |
| | | B | ตรวจพบ 50 | แก้ไข 50 | 100% | B | ตรวจพบ 50 | แก้ไข 50 | 100% | B | ตรวจพบ 50 | แก้ไข 50 | 100% |
| PostgreSQL | before | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% |
| | | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% | B | ตรวจพบ 50 | ไม่แก้ไข 50 | 100% |
| | after | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% | A | ไม่พบ 50 | แก้ไข 50 | 100% |
| | | B | ตรวจพบ 50 | แก้ไข 50 | 100% | B | ตรวจพบ 50 | แก้ไข 50 | 100% | B | ตรวจพบ 50 | แก้ไข 50 | 100% |