|  |
| --- |
| **แนะนำระบบจัดการฐานข้อมูล** |

  เมื่อกล่าวถึงระบบฐานข้อมูล คำว่า “ ข้อมูล ” คือสิ่งที่จะต้องพิจารณาก่อนสิ่งอื่นใด ความหมายของคำว่า ข้อมูล (data) คือ ข้อเท็จจริงต่างๆ ซึ่งต่อไปก็จะถูกนำมาประมวลผลนั่นเอง หรือ อาจ จะ เรียกว่าเป็นวัตถุดิบ เมื่อข้อมูลถูกนำมาประมวล เช่น การจัด เรียง การ แยก กลุ่มจัดกลุ่ม การ เชื่อมโยง หรือการคิด คำนวณ เพื่อให้ได้ผลลัพธ์ และ ต่อมาก็ จัดให้อยู่ในรูปแบบที่ใช้ประโยชน์ได้ ง่าย เรียกว่าเป็น “ สารสนเทศ ” กล่าวโดยสรุป สารสนเทศ ( information) ก็คือข้อมูลที่ผ่านการประมวลผลแล้ว ตัวอย่างเช่น ข้อมูลของคะแนนสอบของรายวิชาต่าง ๆ ซึ่งจะถูกนำมาประมวลเป็นเกรดของนิสิตแต่ละคนในแต่ละภาคการศึกษา แล้วนำเกรดที่ได้มาคำนวณเกรดเฉลี่ย เพื่อนำไปตรวจสอบสถานภาพเพื่อแจ้งสภาพของนิสิตต่อไป

ลักษณะของสารสนเทศที่ดี ควรจะมีองค์ประกอบใน 4 ลักษณะ ต่อไปนี้

**1. ต้อง มีค่า ถูกต้อง เที่ยงตรง (Correctness and relevant)** สารสนเทศ ที่ ถูกต้องและไม่ผิดพลาด จะทำให้ ผู้ ใ ช้ได้ข้อมูลสมบูรณ์ และสามารถ ทำงานในส่วนของตน รับผิดชอบได้อย่างเต็มความสามารถไม่ต้องกังวลว่าได้รับข้อมูลที่ผิดพลาด ระบบสารสนเทศ ที่มีความถูกต้องเที่ยงตรงเหล่า นั้นก็จะถูกจัดว่าเป็นระบบที่มีประสิทธิภาพ

**2. สารสนเทศ เป็นปัจจุบัน (current)** ข้อมูล ที่จะนำมาเปลี่ยนสภาพเป็นสารสนเทศ อาจ จะ มีการเปลี่ยน แปลง ไป อยู่เสมอ ตามกาลเวลา ตามความต้องการ เช่น ข้อมูลเกี่ยวกับเกรดเฉลี่ยของ นิสิต/ นักศึกษาในแฟ้มประวัติของนักศึกษา ซึ่ง จะต้องเปลี่ยน แปลง ไปในแต่ละภาค การศึกษา ระบบสารสนเทศที่ดีจะต้องยืดหยุ่น โดย ให้มีการเปลี่ยน แปลง ค่าให้เป็นปัจจุบัน อยู่เสมอ หรือ ยัง คง เก็บ ค่าเก่าไว้เพื่อประโยชน์ ใน การใช้งาน ในกรณีอื่น ๆ ที่ แตกต่างกัน ไป

**3. สารสนเทศที่ ทันเวลา (timely)** สารสนเทศ มีคุณค่าทางเวลาเข้ามาเกี่ยวข้อง ถ้าไม่ได้สารสนเทศในเวลาที่ต้องการ อาจจะเกิดการสูญเสียโอกาสที่ไม่อาจจะได้กลับมาใหม่ ถ้า มหาวิทยาลัย ไม่สามารถหาข้อมูลสารสนเทศได้ทันเวลาประมูล มหาวิทยาลัย ก็อาจจะเสียโอกาสนั้นไป ระบบสารสนเทศที่มีประสิทธิภาพคือ ระบบที่จะต้องจัดสรรให้ได้สารสนเทศเมื่อผู้ใช้ต้องการ ในเวลาที่ต้องการ

**4. ไม่มีความขัดแย้งกัน (consistant)** ใน หลาย ๆ กรณี สารสนเทศ ทำให้ เกิดความขัดแย้ง กันของ ข้อมูลที่จัดเก็บในหลาย ๆ ที่ หลาย ๆ แหล่งข้อมูล ซึ่งจะทำให้ข้อมูลหรือสารสนเทศเหล่านั้น ไม่ตรงกัน ได้ หรือแม้แต่ วิธีการประมวลผลที่ แตก ต่างกัน ก็ อาจทำให้เกิดความคลาดเคลื่อนขึ้น ได้ ดังนั้น ข้อมูล ที่ มีความคงที่มากที่สุด จะทำให้ข้อมูลมีคุณค่าควรแก่การใช้งาน

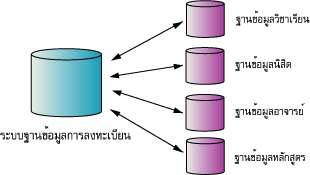
           นอก จากนี้แล้วการนำเสนอข้อมูลหรือการนำเสนอสารสนเทศในรูปแบบที่เหมาะสม ย่อมทำให้สารสนเทศนั้นมีคุณค่ามากยิ่งขึ้นตามไปด้วย ตัวอย่างเช่น ถ้าอาจารย์สอนนิสิต/นักศึกษาหลายร้อยคนและต้องการจะดูคะแนนรวมของนาย ก. แต่ในระบบข้อมูลมีวิธีการจัดเรียงข้อมูลตามรหัสนักศึกษา ดังนั้นอาจารย์จะต้องค้นหาชื่อนักศึกษาตั้งแต่ต้นจนกว่าจะพบชื่อที่ต้องการ โดยที่อาจารย์ไม่ทราบว่านักศึกษาผู้นั้นมีรหัสเท่าใด ดังนั้นระบบสารสนเทศที่มีประสิทธิภาพ คือระบบที่มีความยืดหยุ่นในการนำเสนอสารสนเทศให้กับผู้ที่ต้องการใช้ สารสนเทศนั้น ๆ

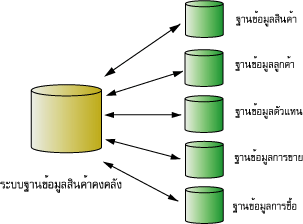
**แนวความคิดพื้นฐานเกี่ยวกับระบบฐานข้อมูล**

         สารสนเทศ เป็นผลลัพธ์ที่ได้จากการประมวลข้อมูลจากตารางข้อมูลที่มีความสัมพันธ์กันการ วางรูปแบบของโครงสร้างของข้อมูลที่มีความสัมพันธ์กัน อาจวาดเป็นภาพของบัตรรายการโดยที่แต่ละบัตรจะเรียกว่าเป็น **เรคอร์ด** (record หรือ row) เนื้อหาภายในบัตรรายการก็จะเป็นรายการ ข้อมูลแต่ละหน่วยหรือเรียกว่า **ฟิลด์** (field หรือ column) และบัตรรายการในเรื่องเดียวกัน เช่น ประวัตินิสิต/นักศึกษา หลาย ๆ ใบรวมกันก็จะถูกเก็บไว้เป็นกลุ่ม เรียกว่า **แฟ้มข้อมูล** (file หรือ table หรือเรียกต่อไปว่า relational table)

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | [http://tsl.tsu.ac.th/file.php/1/courseware/database/lesson1/images/p1.gif](javascript:openWindow5('less_11.htm')) | | **ภาพ ที่ 1.1** ตัวอย่างข้อมูลเป็นเรคอร์ด  (ในรูปของบัตรรายการ ) | | **ฟิลด์ :** หน่วยพื้นฐานของข้อมูล ( เช่น ชื่อนิสิต นามสกุลนิสิต รหัสนิสิต เพศ ที่อยู่ )  **เรคอร์ด :** ชุดของข้อมูลที่สัมพันธ์กัน เรื่องราวเดียวกัน จากตัวอย่างคือประวัตินิสิต 1 คน (1 บัตรรายการ )  **ไฟล์ :** จำนวนบัตรรายการทั้งหมดรวมกัน (บัตรรายการเรื่องเดียวกัน) (1 แฟ้มข้อมูล )          ฐานข้อมูล ( Database)  หมายถึง  ชุดของข้อมูลที่รวมเอาข้อมูลที่เกี่ยวข้องกันเป็นเรื่องราวเดียวกันรวมกัน เป็นกลุ่มหรือเป็นชุดข้อมูล เช่น ฐานข้อมูลนิสิต ฐานข้อมูลค้า และ ฐานข้อมูลวิชาเรียน เป็นต้น ซึ่งข้อมูลเหล่านี้ได้มาจากการบันทึกข้อมูลโดยผู้ใช้ หรือบางข้อมูลอาจจะได้มาจากการประมวลผลข้อมูลแล้วบันทึกข้อมูลกลับไปเก็บที่ ตำแหน่งที่ต้องการ |

      ระบบฐานข้อมูล ( Database System) ความ หมายของระบบฐานข้อมูลก็คือ  ที่รวมของฐานข้อมูลต่าง ๆ หรือที่รวมของข้อมูลทั้งหมด ซึ่งอาจจะได้จากการคำนวณ หรือประมวลผลต่าง ๆ หรืออาจจจะได้จากการบันทึกข้อมูลโดยผู้ใช้ เช่น ระบบฐานข้อมูลงานทะเบียนนิสิตมหาวิทยาลัยทักษิณ ก็จะรวมเอาฐานข้อมูลต่าง ๆ เช่น ฐานข้อมูลวิชาเรียน ฐานข้อมูลนิสิต ฐานข้อมูลอาจารย์ผู้สอน และ ฐานข้อมูลหลักสูตร เป็นต้น ซึ่งรวมกันเป็นระบบฐานข้อมูลของงานทะเบียนนิสิต หรือฐานข้อมูลห้างร้านต่าง ๆ ก็จะประกอบด้วย ฐานข้อมูลสินค้า ฐานข้อมูลลูกค้า ฐานข้อมูลระบบบัญชี ฐานข้อมูลลูกหนี้ และฐานข้อมูลตัวแทนจำหน่าย เป็นต้น ดังภาพประกอบต่อไปนี้





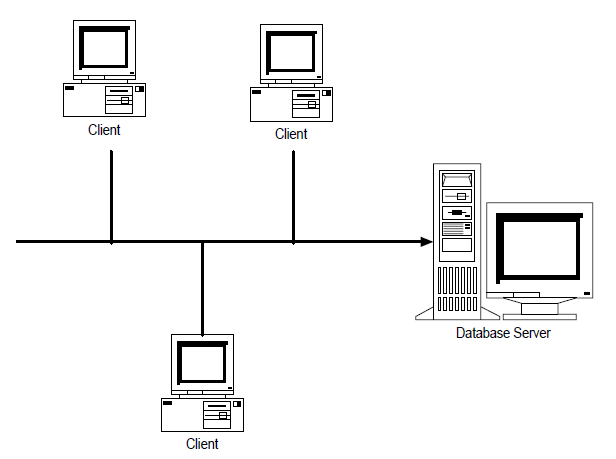
**ภาพ ที่ 1.2** ตัวอย่างระบบฐานข้อมูล และฐานข้อมูลย่อย

          จากที่กล่าวมาข้างต้น ระบบฐานข้อมูลต่าง ๆ ตามตัวอย่างนั้น ถ้าเป็นระบบฐานข้อมูลที่มีขนาดใหญ่มาก ระบบฐานข้อมูลขององค์กรก็ควรจะประกอบด้วยระบบฐานข้อมูลย่อย ๆ หลาย ๆ ระบบฐานข้อมูล เพื่อความสะดวกในการปฏิบัติงานและการจัดการ

|  |  |
| --- | --- |
| [http://tsl.tsu.ac.th/file.php/1/courseware/database/lesson1/images/database6.gif](javascript:openWindow6('lesson11.html')) | |
| **ภาพ ที่ 1.3** ระบบฐานข้อมูลงานลงทะเบียน  ที่ประกอบด้วยระบบฐานข้อมูลย่อย ๆ | **ภาพ ที่ 1.3** ระบบฐานข้อมูลงานลงทะเบียน ที่ประกอบด้วยระบบฐานข้อมูลย่อย ๆ |

**ระบบฐานข้อมูลแบบ Client/Server**

ระบบ Client/Server นี้เป็นการแยกการทำ งานของ Font-end กับ Back-end ออกจากกันโดยผู้ใช้สามารถที่จะทำการทำงานกับฐานข้อมูลได้โดยไม่จำ เป็นต้องทำ งานอยู่ที่เครื่องที่ทำ หน้าที่เก็บฐานข้อมูลอยู่นั้นเราจะเรียกว่า Back-end ในการนำ เสนอระบบนี้อย่างไรนั้นขึ้นกับ Platform ที่ Font-end กับ Back-end ทำ งานอยู่ และระดับการจัดการที่ถูกแบ่งออกเป็น 2 ส่วน

****

**โครงสร้างของระบบ Client/Server Database**

***ข้อดี ของระบบฐานข้อมูลแบบ Client/Server***

* การแบ่งแยกการจัดการระหว่างระบบของ Client และ Database Server ออกจากกันโดยการจัดการฐานข้อมูลจะถูกทำ ที่ส่วนของ Back-end ส่วน DBMS จะถูกจัดการอยู่ที่ Server ทำให้สามารถทำ การขยายการใช้งานของเครื่องคอมพิวเตอร์ที่มีขนาดเล็กหรือช้าเกินกว่าที่จะสามารถทำ การ Run DBMS ที่ซับซ้อนบนเครื่องนั้น
* ช่วยลดโหลด(Load)ให้กับระบบ Nerwork ที่เชื่อมต่อได้ด้วยแทนที่จะต้องทำ การส่งข้อมูลทั้งหมดไปและกลับผ่านทางสายแลน(LAN) ไปยังเครื่องที่ทำ การติดต่อเข้ามาทำ ให้การจราจรบนสายส่งลดน้อยลงเหลือเพียงแค่การส่ง Query มาจาก Client มายังตัวที่เป็นฐานข้อมูล เพื่อทำ งานอย่างใดอย่างหนึ่ง เมื่อ Server ได้รับ Query ที่ส่งมาก็จะทำ การคำ นวณและส่งผลลัพธ์กลับไปยัง Client เพียงเท่านั้น
* การทำ งานของโปรแกรมไม่ขึ้นกับเครื่องที่ทำ งาน ผู้ใช้จะไม่ถูกจำ กัดบนเครื่องระบบใดระบบหนึ่งเท่านั้น สามารถนำ เอาโปรแกรมไปทำ งานบนเครื่องคอมพิวเตอร์เครื่องไหนก็ได้และสามารถทำ งานได้กับระบบปฎิบัติการได้หลายตัวไม่ว่าจะเป็น MS-Windows,IBM OS/2,MS/PC-Dos เป็นต้นนอกจากนี้ Client และ Server ก็ไม่จำ เป็นต้องใช้ฐานข้อมูลเดียวกัน
* การรกั ษาความถูกต้องของข้อมูล ในปัจจุบันระบบ Database Server สว่ นมากทำ งานบน DBMS ที่ใช้การจัดการแบบ Relation DBMS เมอื่ ผู้ใช้ต้องการที่จะทำ การแก้ไขค่าของข้อมูลจะต้องทำ การ Log In เข้ามาใช้งานที่ Server กอ่ นทำ ให้ข้อมูลไม่กระจัดกระจาย และมีความถูกต้องอยู่เสมอ
* การปกป้องข้อมูล ที่เครื่องที่ทำ หน้าที่ในการเก็บข้อมูล (Server) บางครั้งอาจมีการเข้ารหัสที่เก็บ ไฟล ดยขอ้ มลู ถกู เข้ารหัสเพื่อป้องกันการดูจากภายนอก DBMS ได้

***ข้อเสียของระบบ Client/Server***

* การเพิ่มค่าใช้จ่ายในการจัดการและเตรียมบุคลากรที่จะทำ หน้าที่บำ รุงรักษา Database Serverเมื่อจำนวนผู้ใช้มีมากขึ้นหรือเมื่อฐานข้อมูลมีขนาดใหญ่มากขึ้น นอกจากนี้การฝึกฝนผู้ควบคุมระบบยงั เปน็ ค่าใช้จ่ายที่เพิ่มขึ้นในตอนเริ่มต้นด้วย เนื่องจากคนที่เข้ามาทำ งานอาจจะไม่คุ้นเคยกับระบบที่ทำอยู่
* การเพิ่มค่าใช้จ่ายทางด้าน Hardware ก็จะเพิ่มขึ้น

|  |
| --- |
| **มาทำความรู้จักกับ MySQL** |

**MySQL** (อ่านว่า “มาย-เอส-คิว-แอล”) จัดเป็นระบบจัดการฐานข้อมูลเชิงสัมพันธ์ (RDBMS: Relational Database Management System) ตัวหนึ่ง ซึ่งเป็นที่นิยมกันมากในปัจจุบัณ โดยเฉพาะอย่างยิ่งในโลกของอินเตอร์เน็ต สาเหตุเพราะว่า MySQL เป็นฟรีแวร์ทางด้านฐานข้อมูลที่มีประสิทธิภาพสูง เป็นทางเลือกใหม่จากผลิตภัณฑ์ระบบจัดการฐานข้อมูลในปัจจุบัน ที่มักจะเป็นการผูกขาดของผลิตภัณฑ์เพียงไม่กี่ตัว นักพัฒนาระบบฐานข้อมูลที่เคยใช้ MySQL ต่างยอมรับในความสามารถความรวดเร็ว การรองรับจำนวนผู้ใช้ และขนาดของข้อมูลจำนวนมหาศาล ทั้งยังสนับสนุนการใช้งานบนระบบปฏิบัติการมากมาย ไม่ว่าจะเป็น Unix, OS/2, Mac OS หรือ Windows ก็ตาม นอกจากนี้ MySQL ยังสามารถใช้งานร่วมกับ Web Development Platform ทั้งหลาย ไม่ว่าจะเป็น C, C++, Java, Perl, PHP, Python, Tcl หรือ ASP ก็ตามที ดังนั้นจึงไม่เป็นที่น่าแปลกใจเลยว่า ทำไม MySQL จึงได้รับความนิยมอย่างมากในปัจจุบัน และมีแนวโน้มสูงยิ่งขึ้นต่อๆ ไปในอนาคต MySQL จัดเป็นซอฟต์แวร์ประเภท Open Source Software สามารถดาวน์โหลด Source Code ต้นฉบับ ได้จากอินเตอร์เน็ต โดยไม่เสียค่าใช้จ่ายใดๆ การแก้ใขก็สามารถกระทำได้ตามความต้องการ MySQL ยึดถือสิทธิบัตรตาม GPL (GNU General Public License) ซึ่งเป็นข้อกำหนดของซอฟต์แวร์ประเภทนี้ส่วนใหญ่ โดยจะเป็นการชี้แจงว่า สิ่งใดทำได้ หรือทำไม่ได้สำหรับการใช้งานในกรณีต่างๆ ทั้งนี้ถ้าต้องการข้อมูลเพิ่มเติม หรือ รายละเอียดของ GPL สามารถหาข้อมูลได้จากเว็บไซต์ [**http://www.gnu.org/**](http://www.gnu.org/)

MySQL ได้รับการยอมรับและทดสอบเรื่องของความรวดเร็วในการใช้งาน โดยจะมีการทดสอบและเปรียบเทียบกับผลิตภัณฑ์ทางด้านฐานข้อมูลอื่นอยู่เสมอ มีการพัฒนาอย่างต่อเนื่อง โดยเริ่มตั้งแต่เวอร์ชันแรกๆ ที่ยังไม่ค่อยมีความสามารถมากนัก มาจนถึงทุกวันนี้ MySQL ได้รับการพัฒนาให้มีความสามารถมากยิ่งขึ้น รองรับข้อมูลจำนวนมหาศาล สามารถใช้งานหลายผู้ใช้ได้พร้อมๆ กัน (Multi-user) มีการออกแบบให้สามารถแตกงานออก เพื่อช่วยการทำงานให้เร็วยิ่งขึ้น รองรับข้อมูลจำนวนมหาศาล เพื่อช่วยการทำงานเร็วยิ่งขึ้น (Multi-threaded)วิธีและการเชื่อมต่อที่ดีขึ้น การกำหนดสิทธิและการรักษาความปลอดภัยของข้อมูลมีความรัดกุมน่าเชื่อถือยิ่งขึ้นเครื่องมือหรือโปรแกรมสนับสนุนทั้งของตัวเองและของผู้พัฒนาอื่นๆ มีมากยิ่งขึ้น

นอกจากนี้สิ่งหนึ่งที่สำคัญคือ MySQL ได้รับการพัฒนาไปในแนวทางตามข้อกำหนดมาตรฐาน SQL ดังนั้น เราสามารถใช้คำสั่ง SQL ในการทำงาน MySQL ได้” นักพัฒนาที่ใช้ SQL มาตรฐานอยู่แล้ว ไม่ต้องศึกษาคำสั่งเพิ่มเติม แต่อาจจะต้องเรียนรู้ถึงรูปแบบและข้อจำกัดบางอย่างโดยเฉพาะ ทั้งนี้ทั้งนั้น ทางทีมงานผู้พัฒนา MySQL มีเป้าหมายอย่างชัดเจนที่จะพัฒนาให้ MySQL มีความสามารถสนับสนุนตามข้อกำหนด SQL92 มากที่สุด และจะพัฒนาให้เป็นไปตามข้อกำหนด SQL99 ต่อไป

(Real-world Application) ก็มักจะแยก Client และ Server ออกเป็นคนละเครื่องกัน และ

สามารถรองรับงานได้ดีมากกว่า ดังนั้น ผู้บริหารระบบ หรือผู้กำหนดนโยบายสำหรับการทำงานเครือข่าย จะต้องคำนึงถึงเรื่องที่เกี่ยวข้องเหล่านี้ให้ดี เพื่อที่จะทำให้ระบบมีการทำงานรับการให้บริการแก่ผู้ใช้อย่ามีประสิทธิภาพและข้อมูลมีความปลอดภัยมากที่สุด

**MySQL กับมาตรฐานภาษา SQL**

จุดมุ่งหมายหนึ่งของทีมผู้พัฒนา MySQL ต้องการทำให้ MySQL เป็น DB Server ตามข้อกำหนด ANSI SOL โดยในขั้นแรกนี้อ้างอิงตาม ANSI SQL92 เป็นหลัก ความสามารถต่างๆ โดยส่วนใหญ่ครอบคลุมการใช้งาน อาจจะมีบางสิ่งที่ยังไม่สามารถทำได้ในปัจจุบัน หรือบางสิ่งก็มีแผนจะเพิ่มเติมเข้าไป ซึ่งได้กล่าวถึงรายละเอียดไปแล้วข้างต้น แต่ในส่วนนี้เราจะมาดูสำหรับความสามารถบางอย่างที่ MySQL เพิ่มเติมเข้ามาสำหรับการทำงาน นอกเหนือจากข้อกำหนด ANSI SQL ดังนี้

* ฟิลด์ประเภท MEDIUMINT, SET, ENUM และ BLOB รวมทั้ง TEXT ต่างๆ
* การกำหนดคุณลักษณะของฟิลด์ เช่น AUTO\_INCREMENT, BINARY, NULL, UNSIGNED และ ZEROFILL
* การเปรียบเทียบสตริงเป็นแบบ Case Insensitive การจัดเรียงข้อมูลใช้รูปแบบภาษา ISO-8859-1 (Latin1) เป็นค่า default
* ชื่อ database และชื่อตารางข้อมูลจะ Case Sensitive (โดยเฉพาะ Unix Platform)
* สามารถใช้คำสั่งของระบบปฏิบัติในการสำรอง เปลี่ยนชื่อ ย้าย ลบ หรือคัดลอกตารางข้อมูลซึ่งเกี่ยวข้องกับไฟล์ .MYD, .MYI และ .frm ได้โดยตรง
* MySQL ไม่สนับสนุน Tablespaces
* โอเปอเรเตอร์ LIKE สามารถใช้กับ Numeric Columns ได้
* การใช้ INTO OUTFILE และ STRAIGHT\_JOIN ในคำสั่ง SELECT
* สามารถใช้ SQL\_SMALL\_RESULT option ในคำสั่ง SELECT ได้
* คำสั่ง EXPLAIN SELECT เพื่อให้แสดงรายละเอียดของการเชื่อมกันของตารางข้อมูล
* สามารถใช้ TEMPORARY หรือ IF NOT EXISTS สำหรับการ CREATE TABLE ได้
* สามารถทำการ CHANGE ชื่อคอลัมน์, DROP ชื่อคอลัมน์ หรือ DROP INDEX, IGNORE หรือRENAME ในการใช้คำสั่ง ALTER TABLE ได้
* สามารถทำการ RENAME TABLE ได้
* สามารถ ADD, ALTER, DROP หรือ CHANGE clauses หลายครั้ง ในคำสั่ง ALTER TABLE ได้
* สามารถทำการ DROP TABLE ด้วยคีย์เวิร์ด IF EXISTS
* สามารถทำการดรอป (Drop) ตารางข้อมูลหลายๆ ตารางได้ด้วยคำสั่ง DROP TABLE เพียงครั้งเดียว
* การใช้ LIMIT clause ในคำสั่ง DELETE
* การใช้ DELAYED clause ในคำสั่ง INSERT และ REPLACE
* การใช้ LOW\_PRIORITY clause ในคำสั่ง INSERT, REPLACE, DELETE และ UPDATE
* การใช้งาน LOAD DATA INFILE ซึ่งไวยากรณ์ใช้ได้กับรูปแบบการโหลดดาต้า (Load Data) ของ Oracle
* คำสั่ง ANALYZE TABLE, CHECK RABLE, OPTIMIZE TABLE และ REPAIR TABLE
* คำสั่ง SHOW
* คำสั่ง SET OPTION
* ฟังก์ชันหลายฟังก์ชันที่จำเป็น เช่น ฟังก์ชันสำหรับทำงานสตริง ซึ่งสามารถใช้ได้ทันที เนื่องจากมี ชื่อและการทำงานที่เหมือนกันของ ANSI SQL และ ODBC
* สามารถใช้ || แทน OR และ && แทน AND ซึ่งเหมือนกับในภาษา C แต่ไม่สามารถใช้ในการเชื่อมสตริงได้ ซึ่ง MySQL ให้ใช้ฟังก์ชัน CONCAT() แทน
* คำสั่งในการ CREATE DATABASE หรือ DROP DATABASE
* การใช้โอเปอเรเตอร์ % โดยจะให้ผลลัพธ์เดียวกันการใช้ฟังก์ชัน MOD() ตัวอย่าง เช่น M % N = MOD(M,N) เป็นต้น
* โอเปอเรเตอร์ =, < >, <=, <, >=>, <<, >>, <=>, AND, OR หรือ LIKE สามารถใช้ในการเปรียบเทียบคอลัมน์ในการใช้คำสั่ง SELECT ได้ ทั้งนี้จะอยู่ทางด้านหน้า FROM ดังตัวอย่างเช่น SELECT id = 12345 AND name=’somchai’ FROM person;
* ฟังก์ชัน LAST\_INSERT\_ID()
* ฟังก์ชัน BIT\_COUNT(), CASE, ELT(), FROM\_DATS(), FORMAT(), IF(), PASSWORD(),ENCRYPT(), md5(), ENCODE(), DECODE(), PERIOD\_ADD(), PERIOD\_DIFF(),TO\_DAYS(), หรือ WEEKDAY()
* ฟังก์ชัน TRIM() เพื่อการตัดส่วนของสตริง ซึ่งใน ANSI SQL จะตัดได้เพียงหนึ่งตัวอักษรเท่านั้น
* GROUP BY function STD(), BIT\_OR() และ BIT\_AND()
* การใช้คำสั่ง REPLACE แทนการใช้คำสั่ง DELETE ตามด้วยคำสั่ง INSERT
* คำสั่ง FLUSH flush\_option
* การใช้ตัวแปร sql statement ด้วย := ตัวอย่างเช่น SELECT @asum:=SUM (total), @cnt:=COUNT(\*), @asum/@cnt AS avg FROM person;

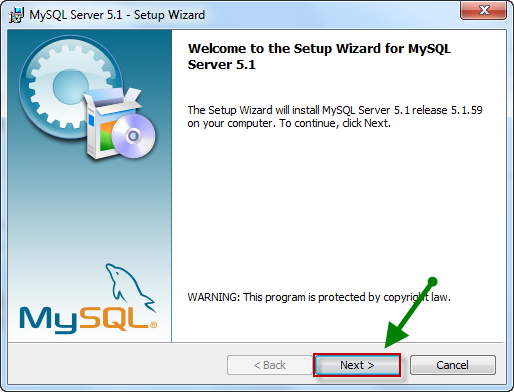
|  |
| --- |
| **การติดตั้ง MySQL** |

เว็บไซต์ www.mysql.com เป็นเว็บไซต์อย่างเป็นทางการของ MySQL ซึ่งเนื้อหาภายในเว๊บไซต์หลักๆจะเป็นการให้ข้อมูลและรายละเอียดเกี่ยวกับ MySQL ไม่ว่าจะเป็น Server, Client หรือ เครื่องมืออื่นๆ ที่เกี่ยวข้องจุดประสงค์ที่สำคัญอันหนึ่ง ก็คือไว้สำหรับการเผยแพร่ MySQL ที่ได้รับการพัฒนา มีให้ดาวน์โหลดตั้งแต่เวอร์ชันที่สมบูรณ์ (Stable) หรือที่กำลังพัฒนา (Beta) มีหลากหลายระบบปฏิบัติการ ผู้ใช้ที่จะนำ MySQLไปรันบนระบบปฏิบัติการตัวใดก็สามารถเลือกได้จากที่เว๊บไซต์นี้ <http://dev.mysql.com/downloads/> นอกเหนือจากนี้ก็จะเป็นลิงก์ต่างๆ ที่มีการนำMySQL ไปพัฒนาเพิ่มเติม โดยมีทั้งแบบ Free และ Commercial ให้เลือกตามอัธยาศัย

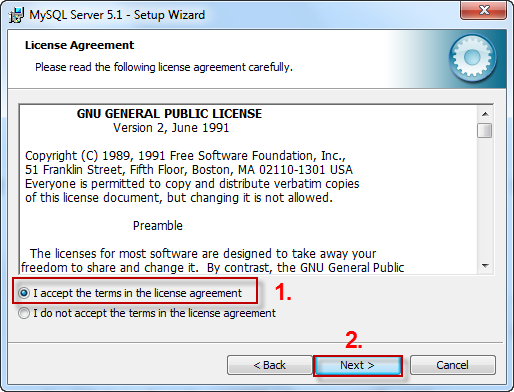
**ติดตั้ง MySQL สำหรับ Windows Platform**

สำหรับ Windows Platform เมื่อดาวน์โหลดไฟล์ติดตั้งมาเรียบร้อยแล้ว ให้ทำตามขั้นตอนดังนี้

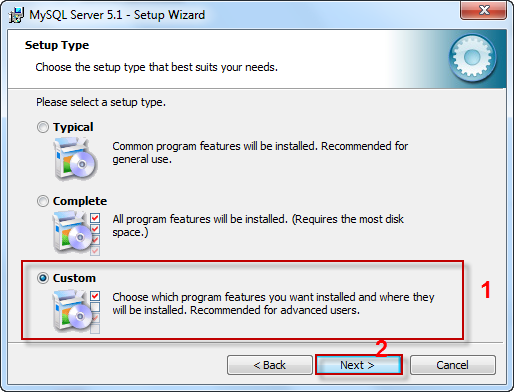
1. ดับเบิ้ลคลิกที่ ไฟล์ mysql-essential-5.1.59-win32.msi
2. เริ่มเข้าสู่ขั้นตอนการติดตั้ง กดปุ่ม Next



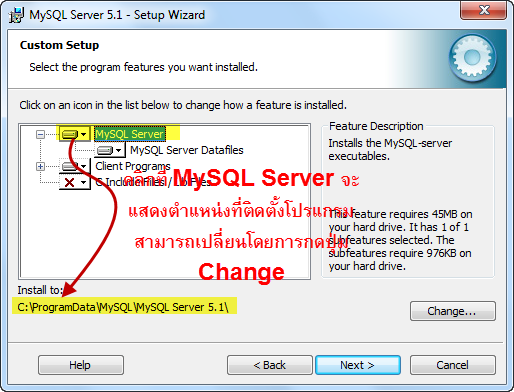
1. การยอมรับ License Agreement ให้เลือก Accept ตามรูป (1) แล้วกดปุ่ม Next(2)

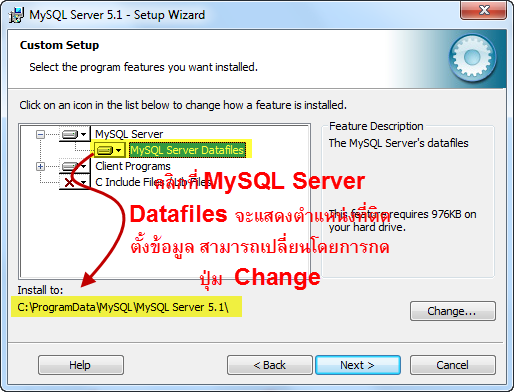


1. เลือกประเภทของการติดตั้ง ให้เลือก Custom (1) แล้วกดปุ่ม Next(2)

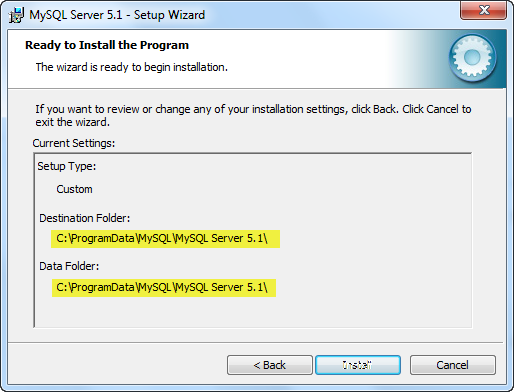


1. กำหนดตำแหน่งที่ต้องการติดตั้ง โดยปกติจะติดตั้งที่ C:\ProgramData\MySQL\MySQL Server 5.1\ สามารถกำหนดไดรฟ์ และโฟล์เดอร์โดยการกดปุ่ม Change หลังจากนั้นให้กดปุ่ม Next

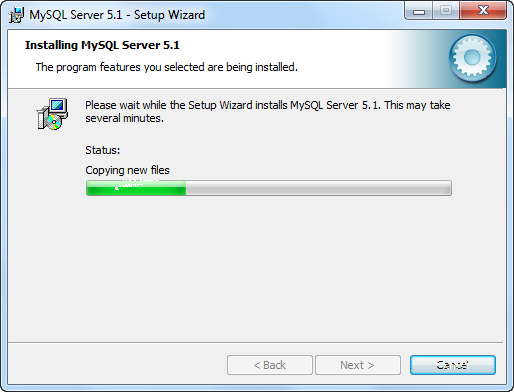




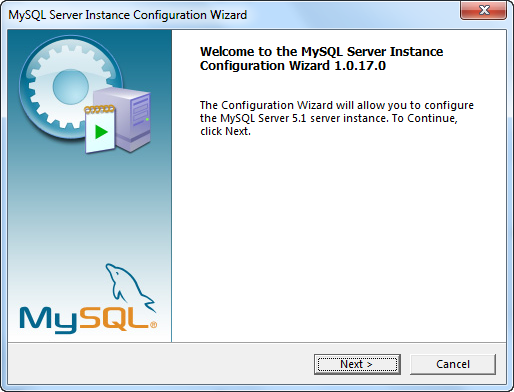
1. เข้าสู่หน้าจอเริ่มการติดตั้งถ้าต้องการเปลี่ยนแปลง ให้กดปุ่ม Back แต่ถ้าต้องการติดตั้งต่อไปให้กดปุ่ม Install



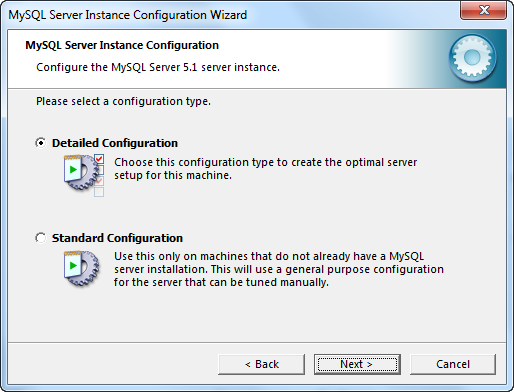
1. แสดงหน้าจอสถานะการติดตั้ง



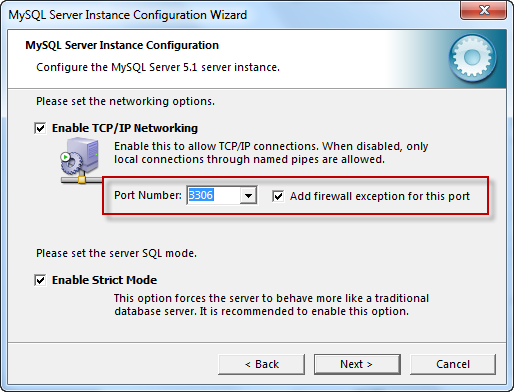
1. เข้าสู่หน้าจอการกำหนดค่าเริ่มต้นของ MySQL Server แล้วกดปุ่ม Next



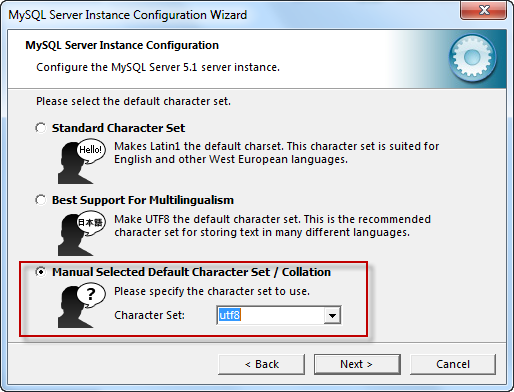
1. เลือก Detail configuration แล้วกดปุ่ม Next



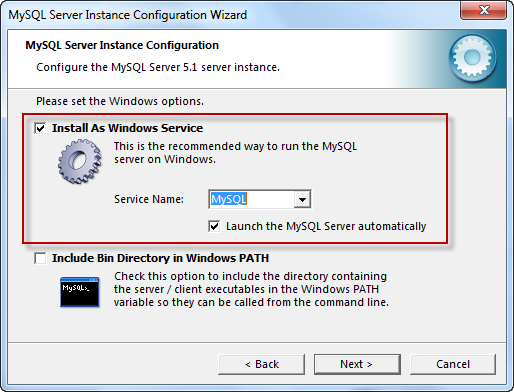
1. กำหนด Port Number ที่ MySQL ใช้งานโดยปกติคือ 3306 แต่สามารถกำหนดเลข port ใหม่ได้โดยกำหนดในช่อง Port Number คลิกเลือก Add firewall exception for this port แล้วกดปุ่ม Next



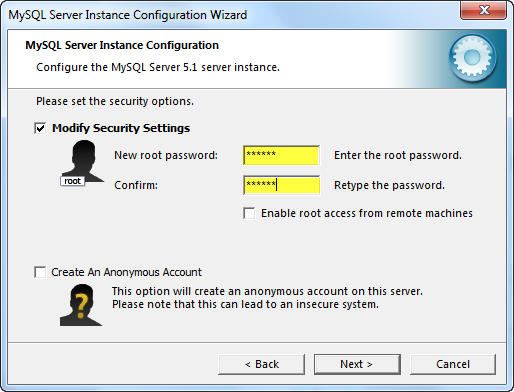
1. กำหนด Character Set เป็น UTF8 แล้วกดปุ่ม Next



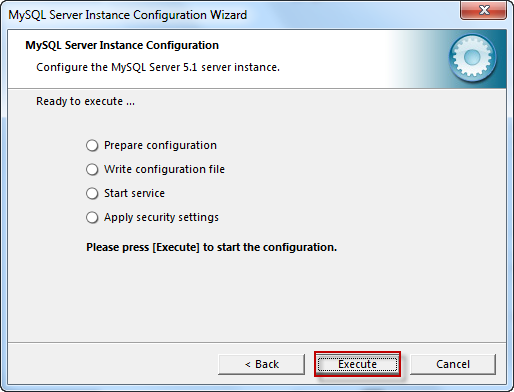
1. กำหนดการติดตั้ง MySQL เป็น Service หนึ่งของ Windows โดยกำหนดชื่อ Service “MySQL” แล้วกดปุ่ม Next



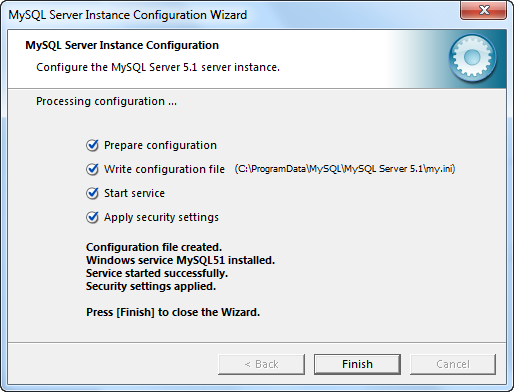
1. กำหนดรหัสผ่านของ root โดยพิมพ์รหัสในช่อง New Root Password และช่อง Confirm รหัสผ่านดังกล่าว คลิกเลือกที่ Enable root access from remote machines ในกรณีที่ต้องการใช้งาน MySQL จากเครื่องภายนอก จากนั้นกดปุ่ม Next



1. เข้าสู่หน้าจอการปรับแต่งคอนฟิก ถ้าต้องการแก้ไขให้กดปุ่ม Back ถ้าต้องการติดตั้งต่อไปให้กดปุ่ม Execute



1. เมื่อโปรแกรมทำการปรับแต่งคอนฟิกเสร็จแล้ว ให้กดปุ่ม Finish เป็นอันจบการติดตั้ง MySQL



|  |
| --- |
| **ประภทข้อมูล MySQL** |

ในส่วนนี้จะกล่าวถึงรายละเอียดของประเภทข้อมูลที่สามารถกำหนดให้แต่ล่ะฟิลด์ ซึ่งสำหรับระบบฐานข้อมูล MySQL จะมีส่วนที่สนับสนุนกับมาตรฐาน และส่วนที่แตกต่าง เพิ่มเติมจากมาตรฐาน และประเภทตารางข้อมูลที่สามารถเลือกใช้ได้ใน MySQL การศึกษาเรียนรู้ถึงประเภทข้อมูลใน MySQL จะทำให้เราเข้าใจ และสามารถทำการกำหนดประเภทของ ข้อมูลให้ถูกต้องเหมาะสมตามจุดประสงค์ หรือตามนโยบายการทำงานของเรา มิฉะนั้นแล้วอาจก่อให้เกิดความเสียหาย หรือทำให้เกิดข้อจำกัดในเรื่องข้อมูลของการทำงานได้ การศึกษาในเรื่องของประเภทข้อมูลใน MySQL ก็มีจุดประสงค์เช่นเดียวกัน ซึ่งจะได้ศึกษาในรายละเอียดต่อไป

**หลักการพิจารณาเลือกใช้ประเภทข้อมูล ดังนี้**

1. **ประเภทข้อมูลหลัก** ได้แก่ ตัวเลข, ตัวอักษร, วันที่และเวลา, ข้อมูลไบนารี และอื่นๆ
2. **ในแต่ละประเภทข้อมูล จะมีประเภทย่อยๆ ลงไปอีก** เพื่อให้ผู้ใช้เลือกใช้ประเภทข้อมูลที่เหมาะสม เพราะนอกจากจะทำงานได้ถูกต้องแล้ว ยังเป็นการช่วยประหยัดเนื้อที่การเก็บข้อมูลอีกด้วย ซึ่งต่อไปเราจะได้ ศึกษาเพิ่มเติมว่าในแต่ละประเภทข้อมูลเป็นอย่างไร มีขอบเขตข้อมูลอะไรบ้าง ใช้เนื้อที่ในการเก็บข้อมูลเท่าไร
3. **การเลือกใช้ประเภทข้อมูล** หากมีความต้องการที่จะสนับสนุนหรือใช้มาตรฐาน เช่น ODBC หรือ ต้องการ port ข้อมูลไปใช้ในระบบอื่น ควรเลือกใช้ประเภทข้อมูลที่สนับสนุนกับมาตรฐานเท่านั้น เพื่อป้องกัน ความผิดพลาดที่อาจจะเกิดขึ้นต่อไปในอนาคต
4. **ฟิลด์ใดที่ไม่มีการคำนวณ หรือไม่มีแนวโน้มจะเกี่ยวข้องกับการคำนวณเลย** ควรเลือกใช้ข้อมูลประเภทตัวอักษร เพราะจะประหยัดเนื้อที่ในการจัดเก็บจริง
5. **การเลือกใช้ประเภทของข้อมูลในแต่ละฟิลด์** นอกจากจะดูความเป็นไปได้ในปัจจุบันแล้ว จะต้องคำนึงและดูแนวโน้มความเป็นไปของข้อมูล หรือปริมาณของข้อมูลต่อไปในอนาคตด้วย เพื่อป้องกันข้อจำกัดของข้อมูลเมื่อมีการขยายตัวเพิ่มขึ้น
6. **การเลือกใช้ข้อมูลประเภทวันที่และเวลาใน MySQL จะต้องเลือกใช้อย่างระมัดระวัง** เนื่องจากมีข้อจำกัดและรายละเอียดที่แตกต่างไปจากมาตรฐาน เช่น จะสามารถใช้ข้อมูลเริ่มต้นได้ที่ปี ค.ศ. 1000 เท่านั้น เป็นต้น

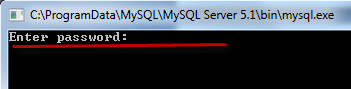
**ประเภทข้อมูลใน MySQL แบ่งออกเป็นกลุ่มได้ ดังนี้**

1. **ประเภทข้อมูลสำหรับตัวเลข** ไว้สำหรับเก็บข้อมูลตัวเลข ซึ่งอาจจะใช้ในการคำนวณ หรือการจัดเรียงข้อมูลเปรียบเทียบกันในฟิลด์นั้นๆ ประกอบด้วยประเภทข้อมูลย่อยๆ ได้แก่ จำนวนเต็ม, จำนวนทศนิยม, จำนวนจริง
   * 1. **TINYINT[(M)] [UNSIGNED] [ZEROFILL]** ข้อมูลชนิดตัวเลขแบบคิดเครื่องหมาย จะใช้ได้ตั้งแต่ –128 ถึง 127 แต่ถ้าแบบไม่คิดเครื่องหมาย จะใช้ได้ ตั้งแต่ 0 ถึง 255 ข้อมูลชนิดที่ใช้เนื้อที่เก็บข้อมูลขนาด 1 ไบต์
     2. **SMALLINT[(M)] [UNSIGNED] [ZEROFILL]** ข้อมูลชนิดตัวเลขแบบคิดเครื่องหมาย จะใช้ได้ตั้งแต่ –32768 ถึง 32767 แต่ถ้าแบบไม่คิดเครื่องหมาย จะใช้ได้ตั้งแต่ 0 ถึง 65535 ข้อมูลชนิดนี้ใช้เนื้อที่เก็บข้อมูลขนาด 2 ไบต์
     3. **MEDIUMINT[(M)] [UNSIGNED] [ZEROFILL]** ข้อมูลชนิดตัวเลขแบบคิดเครื่องหมาย จะใช้ได้ตั้งแต่ –8388608 ถึง 8388607 แต่ถ้าแบบไม่คิด เครื่องหมาย จะใช้ได้ตั้งแต่ 0 ถึง 16777215 ข้อมูลชนิดนี้ใช้เนื้อที่เก็บข้อมูลขนาด 3 ไบต์
     4. **INT[(M)] [UNSIGNED] [ZEROFILL] หรือ INTEGER[(M)] [UNSIGNED] [ZEROFILL]** ข้อมูลชนิดตัวเลขแบบคิดเครื่องหมาย จะใช้ได้ตั้งแต่ –2147483648 ถึง 2147483647 แต่ถ้าแบบไม่คิดเครื่องหมาย จะใช้ได้ตั้งแต่ 0 ถึง 4294967295 ข้อมูลชนิดนี้ใช้เนื้อที่เก็บข้อมูลขนาด 4 ไบต์
     5. **BIGINT[(M)] [UNSIGNED] [ZEROFILL]** ข้อมูลชนิดตัวเลขแบบคิดเครื่องหมาย จะใช้ได้ตั้งแต่ –9223372036854775808 ถึง 9223372036854775807 แต่ถ้าแบบไม่คิดเครื่องหมาย จะใช้ได้ตั้งแต่ 0 ถึง 18446744073709551615 ข้อมูลชนิดนี้ ใช้เนื้อที่เก็บข้อมูลขนาด 8 ไบต์
     6. **FLOAT(precision) [ZEROFILL]** ข้อมูลชนิดตัวเลขแบบคิดเครื่องหมาย precision เป็นค่าความละเอียดทศนิยม ซึ่งแบ่งเป็นชนิด single (มี ค่าตั้งแต่ 0-24) และแบบ double (ตั้งแต่ 25-53) ข้อมูลชนิดนี้ใช้เนื้อที่เก็บข้อมูลขนาด 4 หรือ 8 ไบต์ โดยแบบsingle จะใช้ 4 ไบต์ และแบบ double จะใช้ 8 ไบต์ ชนิดข้อมูลประเภทนี้สำหรับกรณีใช้กับ ODBC มาตรฐาน
     7. **FLOAT[(M,D)] [ZEROFILL]** ข้อมูลชนิดตัวเลขแบบคิดเครื่องหมาย จะใช้ได้ตั้งแต่ -3.402823466E+38 ถึง -1.175494351E -38,0 และ 1.175494351E-39 ถึง 3.402823466E+38 ค่า M เป็นจำนวนหลักที่ต้องการแสดงผล และ D เป็นจำนวนจุดทศนิยม ข้อมูลชนิดนี้ใช้เนื้อที่เก็บข้อมูลขนาด 4 ไบต์
     8. **DOUBLE[(M,D)] [ZEROFILL] หรือ DOUBLE PRECISION[(M,D)] [ZEROFILL] หรือ REAL[(M,D)] [ZEROFILL]** ข้อมูลชนิดตัวเลขแบบคิดเครื่องหมาย จะใช้ได้ตั้งแต่ –1.7976931348623157E+308 ถึง -2.2250738585072014E-308, 0 และ 2.2250738585072014E-308 ถึง 1.7976931348623157E+308 ค่า M เป็นจำนวนหลักที่ต้องการแสดงผล และ D เป็นจำนวนจุดทศนิยม ข้อมูลชนิดนี้ใช้เนื้อที่เก็บข้อมูล ขนาด 8 ไบต์
     9. **DECIMAL[(M[,D])] [ZEROFILL] หรือ NUMERIC[M,D] [ZEROFILL]** ข้อมูลชนิดตัวเลขชนิด unpacked คืออนุญาตให้สามารถเก็บข้อมูลตัวอักษรเข้าไปด้วย โดยตัวอักษรหนึ่ง ตัวแทนแต่ละหลัก สามารถใช้ตัวเลขได้เท่ากับแบบ DOUBLE ใช้เนื้อที่เก็บข้อมูลเท่ากับ M ไบต์ (D+2, ถ้า M < D)
   1. **ประเภทข้อมูลสำหรับวันที่และเวลา**
      1. **DATE** ข้อมูลชนิดวันที่ ตั้งแต่วันที่ 1 มกราคม ค.ศ. 1000 ถึง 31 ธันวาคม ค.ศ. 9999 การแสดงผลวันที่อยู่ใน รูปแบบ ‘YYYY-MM-DD’ ข้อมูลชนิดนี้ใช้เนื้อที่เก็บข้อมูลขนาด 3 ไบต์
      2. **DATETIME** ข้อมูลชนิดวันที่และเวลา ตั้งแต่วันที่ 1 มกราคม ค.ศ. 1000 เวลา 00:00:00 ถึง 31 ธันวาคม ค.ศ. 9999 เวลา 23:59:59 การแสดงผลวันที่และเวลาอยู่ในรูปแบบ ‘YYYY-MM-DD HH:MM:SS’ ข้อมูลชนิดนี้ใช้เนื้อที่เก็บ ข้อมูลขนาด 8 ไบต์
      3. **TIMESTAMP[(M)]** ค่า timestamp เป็นค่าตัวเลขที่นับจำนวนวินาทีตั้งแต่วันที่ 1 มกราคม ค.ศ. 1000 เวลา 00:00:00 เป็นต้นมา โดยมีจุดสิ้นสุดประมาณ ค.ศ. 2037 การแสดงผลค่า timestamp อยู่ในรูปแบบ YYYYMMDDHHMMSS, YYMMDDHHMMSS, YYYYMMDD หรือ YYMMDD ค่า M คือจำนวนตัวเลขที่บรรจุ ซึ่งอาจจะเป็น 14, 12, 8 หรือ 6 ข้อมูลชนิดนี้ใช้เนื้อที่เก็บข้อมูลขนาด 4 ไบต์
      4. **TIME** ข้อมูลประเภทเวลา สามารถเป็นได้ตั้งแต่ ‘-838:59:59’ ถึง ‘838:59:59’ แสดงผลในรูปแบบ HH:MM:SS ข้อมูลชนิดนี้ใช้เนื้อที่เก็บข้อมูลขนาด 3 ไบต์
      5. **YEAR[(2/4)]** ข้อมูลประเภทปี ค.ศ. โดยสามารถเลือกว่าจะใช้แบบ 2 หรือ 4 หลัก (ค่าโดยปริยายจะเป็น 4 หลัก) ถ้า เลือกใช้แบบ 4 หลัก จะใช้ได้ตั้งแต่ ปี ค.ศ. 1901 ถึง 2155 ถ้าเลือกใช้แบบ 2 หลัก จะใช้ได้ตั้งแต่ ปี ค.ศ. 1970 ถึง 2069 แสดงในรูปแบบ 70 ถึง 69 ข้อมูลชนิดนี้ใช้เนื้อที่เก็บข้อมูลขนาด 1 ไบต์
   2. **ประเภทข้อมูลสำหรับตัวอักษร** ข้อมูลประเภทตัวอักษรถือว่าเป็นสิ่งที่ใช้กันมากที่สุด เพราะแทบจะครอบคลุมข้อมูลที่เป็นไปได้ทั้งหมด การจัดเรียงข้อมูลจะเป็นในลักษณะไม่คำนึงตัวใหญ่ตัวเล็ก (case-insensitive) ซึ่งเมื่อเปรียบเทียบกับข้อมูล ประเภทไบนารี (BLOB) แล้วจะพบว่าคล้ายกันมาก จะแตกต่างกันที่ BLOB จะมีการจัดเรียงข้อมูลเป็น caseinsensitive เท่านั้น แต่ข้อมูลประเภทตัวอักษรก็มีรายละเอียดเพิ่มเติมไปอีก โดยมีรายละเอียดดังนี้
      1. **[NATIONAL] CHAR(M) [BINARY]** ข้อมูลประเภทสตริงที่จำกัดขนาดความกว้าง (ไม่สามารถปรับขนาดได้) โดยขนาดความกว้างเป็นได้ ตั้งแต่ 1 ถึง 255 ตัวอักษร ตามปกติเมื่อมีการเรียงข้อมูล จะเป็นลักษณะ case-sensitive คือคำนึงถึงตัวเล็กตัวใหญ่ การระบุชนิดข้อมูลย่อยว่าเป็น BINARY จะเป็นลักษณะไม่คำนึงตัวเล็กตัวใหญ่ ดังนั้นเมื่อมีการเรียงข้อมูลก็จะให้ผลต่างจาก CHAR ธรรมดา การเก็บข้อมูลก็ใช้ไบต์ตามจำนวนตัวอักษรที่ระบุ
      2. **[NATIONAL] VARCHAR(M) [BINARY]** ข้อมูลประเภทนี้ก็คล้ายกับแบบ CHAR ต่างกันตรงที่ VARCHAR จะสามารถปรับขนาดตามข้อมูลที่เก็บในฟิลด์ ขนาดความกว้างอยู่ตั้งแต่ 1 ถึง 255 การเก็บข้อมูลจะเท่ากับของข้อมูลจริงในฟิลด์ๆ + 1 ไบต์
      3. **TINYTEXT** ข้อมูลประเภทนี้สามารถใช้ความกว้างข้อมูลได้สูงสุด 255 ตัวอักษร และใช้เนื้อที่เก็บข้อมูลเท่ากับ จำนวนข้อมูลจริง + 1 ไบต์
      4. **TEXT** ข้อมูลประเภทนี้สามารถใช้ความกว้างข้อมูลได้สูงสุด 65,535 ตัวอักษร และใช้เนื้อที่เก็บข้อมูลเท่ากับ จำนวนข้อมูลจริง + 2 ไบต์
      5. **MEDIUMTEXT** ข้อมูลประเภทนี้สามารถใช้ความกว้างข้อมูลได้สูงสุด 16,777,215 ตัวอักษร และใช้เนื้อที่เก็บข้อมูล เท่ากับจำนวนข้อมูลจริง + 3 ไบต์
      6. **LONGTEXT** ข้อมูลประเภทนี้สามารถใช้ความกว้างข้อมูลได้สูงสุด 4,294,967,295 ตัวอักษร และใช้เนื้อที่เก็บข้อมูล เท่ากับจำนวนข้อมูลจริง + 4 ไบต์
      7. **ENUM(‘value1’,’value2’,…)** ข้อมูลประเภทระบุเฉพาะค่าที่ต้องการ หรือถ้าไม่มีจะให้ค่า NULL สามารถกำหนดค่าได้ถึง 65,535 ค่าและใช้เนื้อที่เก็บข้อมูลตามจำนวนค่าที่ระบุ
      8. **SET(‘value1’,’value2’,…)** ข้อมูลประเภทเซต ประกอบด้วยข้อมูลตั้งแต่ไม่มีค่า หรือมีค่าตามสมาชิกที่กำหนด สามารถมีจำนวนสมาชิกในเซตได้ทั้งสิ้น 64 ตัว และใช้เนื้อที่เก็บข้อมูลตามจำนวนสมาชิกที่ระบุ
   3. **ประเภทข้อมูลสำหรับไบนารี (BLOB:Binary Large Object)** จากที่กล่าวไปแล้วถึงความคล้ายกันของแต่ล่ะข้อมูลประเภทตัวอักษรและข้อมูลประเภทไบนารี ดังนั้น ประเภทข้อมูลย่อยและการเก็บข้อมูลจึงไม่แตกต่างกัน แต่ตามปกติเรามักใช้ BLOB สำหรับเก็บข้อมูลที่ไม่เป็น ศูนย์เทคโนโลยีสารสนเทศและการสื่อสาร สป. การใช้งานฐานข้อมูล MySQL 53 ภาษามนุษย์ ข้อมูลไบนารี ได้แก่ รูปภาพ, ไฟล์ข้อมูล หรือข้อมูลที่ประกอบด้วยตัวอักษรพิเศษ ข้อมูลที่กำหนด ด้วยรหัสควบคุม เป็นต้น
      1. **TINYBLOB** สำหรับข้อมูลไบนารี สามารถใช้ความกว้างข้อมูลได้สูงสุด 255 ตัวอักษร และใช้เนื้อที่เก็บข้อมูลเท่ากับ จำนวนข้อมูลจริง + 1 ไบต์
      2. **BLOB** สำหรับข้อมูลไบนารี สามารถใช้ความกว้างข้อมูลได้สูงสุด 65,535 ตัวอักษร และใช้เนื้อที่เก็บข้อมูล เท่ากับจำนวนข้อมูลจริง + 2 ไบต์
      3. **MEDIUMBLOB** สำหรับข้อมูลไบนารี สามารถใช้ความกว้างข้อมูลได้สูงสุด 16,777,215 ตัวอักษร และใช้เนื้อที่เก็บข้อมูลเท่ากับจำนวนข้อมูลจริง + 3 ไบต์
      4. **LONGBLOB** สำหรับข้อมูลไบนารี สามารถใช้ความกว้างข้อมูลได้สูงสุด 4,294,967,295 ตัวอักษร และใช้เนื้อที่เก็บ ข้อมูลเท่ากับจำนวนข้อมูลจริง + 4 ไบต์

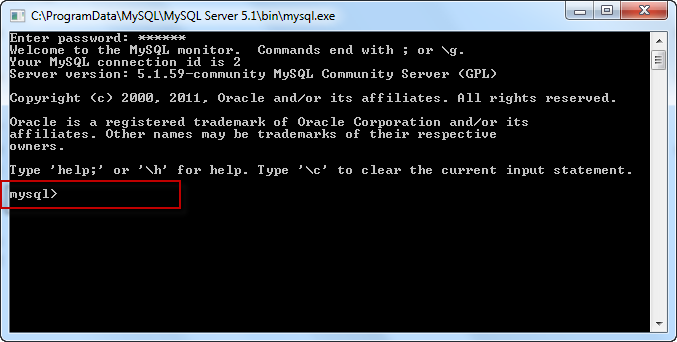
|  |
| --- |
| **การใช้งาน MySQL** |

หลังจากติดตั้งโปรแกรมเรียบร้อยแล้ว สามารถเข้าใช้งาน MySQL ได้ดังนี้

* 1. คลิกเลือก Start >> MySQL >> MySQL Server 5.0 >> MySQL Command line Client
  2. จะปรากฎหน้าจอของ MySQL Command Line Client แล้วให้พิมพ์รหัสผ่านที่ใส่ในขั้นตอนการติดตั้ง แล้วกด Enter



* 1. จะเข้าสู่หน้าจอการใช้งาน MySQL โดยสังเกตุที่ mysql> ดังรูป



* 1. ในกรณีที่ต้องการกำหนดรหัสผ่านใหม่ ให้ใช้คำสั่งดังนี้
* UPDATE mysql.user SET password=password(‘1234’) WHERE user = ‘root’; [กด Enter]
* Flush privileges; [กด Enter]

|  |
| --- |
| MySQL Command Line |

**คำสั่งเพื่อเข้าใช้งาน MySQL**

|  |
| --- |
| mysql [db\_name] -h hostname -u root –p |

|  |  |
| --- | --- |
| -h [ชื่อ server] | ชื่อหรือ IP Address ของ Database Server เช่น localhost หรือ 127.0.0.1 |
| -u [user name] | ชื่อผู้ใช้งานฐานข้อมูล |
| -P [Port Number] | เลข Port ของ MySQL เช่น 3333 |
| -p [password] | รหัสผ่าน |
| [db\_name] | ชื่อฐานข้อมูลที่ต้องการใช้งาน |

เช่น mysql –h localhost –u root

**เรียกใช้งานฐานข้อมูล**

|  |
| --- |
| **use** [db name]; |

**คำสั่ง SHOW**

|  |  |
| --- | --- |
| SHOW DATABASES | เป็นคำสั่งสำหรับแสดงรายละเอียด (Information) ของ databases, tables, columns หรือ สถานะต่างๆ  ของ Server |
| SHOW TABLES | แสดงรายชื่อ databases ทั้งหมดในระบบฐานข้อมูลแสดงรายชื่อ tables ทั้งหมดของ current database |
| SHOW OPEN TABLES | แสดงรายชื่อ tables ที่ถูกเปิดอยู่ในปัจจุบัน |
| SHOW COLUMNS | แสดงรายละเอียดฟิลด์ทั้งหมดใน table นั้น |
| SHOW FIELDS | ให้ผลลัพธ์เช่นเดียวกับ SHOW COLUMNS  (ให้ผลลัพธ์เช่นเดียวกับคำสั่ง DESCRIBE) |
| SHOW INDEX | แสดงรายละเอียดว่ามีดัชนีอะไรบ้างสำหรับ table นั้นๆ |
| SHOW KEYS | ให้ผลลัพธ์เช่นเดียวกับ SHOW INDEX |

**ขอความช่วยเหลือ**

|  |
| --- |
| help หรือ \h |

**เลิกใช้งาน MySQL**

|  |
| --- |
| exit หรือ \q |

|  |
| --- |
| Data Definition Language |

**คำสั่งสร้างฐานข้อมูล**

|  |
| --- |
| **CREATE DATABASE** db\_name |

เป็นคำสั่งสร้างฐานข้อมู(database) ชื่อ db\_name ขึ้นมาใหม่ ในกรณีที่มี database นั้นอยู่แล้ว ก็จะปรากฏข้อความแสดงความผิดพลาด (Error Message) Can't create database ' db\_name '. Database exists

**คำสั่งลบฐานข้อมูล**

|  |
| --- |
| **DROP DATABASE** db\_name |

เป็นคำสั่งลบ database ชื่อ db\_name ออกไปจากระบบ

**คำสั่งสร้างตารางข้อมูล**

|  |
| --- |
| **CREATE TABLE** phonebook (  id INT (4) NOT NULL A UTO\_INCREMENT,  fname VARCHAR (35) ,  lname VARCHAR (35) ,  email VARCHAR (50) ,  office VARCHAR (200) ,  province CHAR (2) ,  PRIMARY KEY(id),  UNIQUE(id),  INDEX(id)**)** |

โดยทั่วไปขนาดของตารางข้อมูลหนึ่งๆ จะมีขนาดไม่เกิน 2G หรือ 4G ขึ้นอยู่กับระบบปฏิบัติการ เราสามารถทำให้ประหยัดเนื้อที่ได้อีกโดยการบีบอัด (PACK) ข้อมูลดัชนี ซึ่งสามารถให้บีบอัดโดยกำหนดค่าPACK\_KEYS=1 การบีบอัดนี้จะทำให้การทำงานช้าลง แต่ก็ประหยัดเนื้อที่ขึ้น และตามปกติถ้าผู้ใช้ไม่ได้ กำหนดให้บีบอัด ข้อมูลที่เป็นสตริงหรือตัวอักษรก็จะได้รับการบีบอัดข้อมูลอยู่แล้ว สำหรับข้อมูลตัวเลข ค่าการทำงานปกติจะไม่รับการบีบอัด เว้นแต่กำหนดการบีบอัดด้วยการใช้ UNION จะใช้กรณีที่ต้องการรวมตารางข้อมูลหลายๆ ตารางเข้าด้วยกันเป็นอันเดียว ซึ่งจะทำงานกับตารางประเภท MERGE เท่านั้น

**คำสั่งแก้ไขตารางข้อมูล**

|  |
| --- |
| **ALTER TABLE** phonebook **ADD tel VARCHAR(30) ;**  **ALTER TABLE** phonebook **MODIFY**  fname  **VARCHAR(50) ;** |

**ALTER TABLE** เป็นคำสั่งสำหรับการปรับเปลี่ยนแก้ไขโครงสร้างของตารางข้อมูล เช่น การเพิ่ม-ลบฟิลด์, การสร้าง-ลบดัชนี, การเปลี่ยนประเภทข้อมูลของฟิลด์, การเปลี่ยนชื่อฟิลด์ เป็นต้น

**เพิ่มคอลัมภ์ในตารางข้อมูล**

|  |
| --- |
| **ALTER TABLE** phonebook ADD notetext |

**ลบคอลัมภ์ตารางข้อมูล**

|  |
| --- |
| **ALTER TABLE** phonebook **DROP** note |

**คำสั่งเปลี่ยนชื่อตารางข้อมูล**

|  |
| --- |
| **RENAME TABLE** phonebook **to** phone |

เป็นคำสั่งเพื่อเปลี่ยนชื่อตารางข้อมูลเป็นชื่อใหม่

**คำสั่งลบตารางข้อมูล**

|  |
| --- |
| **DROP TABLE** phonebook |

เป็นคำสั่งเพื่อให้ลบตารางข้อมูลออกจาก สำหรับการใช้คำสั่ง DROP TABLE จะต้องให้ความระมัดระวังในการใช้งานเป็นอย่างยิ่ง โดยเฉพาะการทำงานกับตารางข้อมูลประเภท nontransaction เพราะจะทำการ commit การทำงานโดยอัตโนมัติ

**คำสั่ง OPTIMIZE TABLE**

|  |
| --- |
| **OPTIMIZE TABLE** phonebook |

เป็นคำสั่งสำหรับจัดระเบียบข้อมูลให้เป็นระเบียบ หรือการทำดีแฟร๊กเมนต์นั่นเองนั่นเอง เพราะถ้าตารางข้อมูลมีการใช้งานมานาน มีการใช้งานที่ทำให้ข้อมูลมีการเปลี่ยนแปลงความยาว หรือเปลี่ยนขนาดของเนื้อที่เก็บข้อมูล โดยเฉพาะกับข้อมูลประเภทที่ยืดหยุ่นไปตามข้อมูล เช่น VARCHAR, BLOB หรือ TEXT การใช้คำสั่ง OPTIMIZE TABLE ก็จะช่วยให้มีการจัดระเบียบข้อมูลให้ดียิ่งขึ้น หากปล่อยระยะเวลานานๆ จะเกิดช่องว่างระหว่างข้อมูล การอ่านข้อมูลของสื่อจะไม่ราบรื่น ช้า เนื่องจากเสียเวลาในการกระโดดข้ามช่องว่างนั้นนั่นเอง

|  |
| --- |
| Data Manipulation Language |

**คำสั่งเพิ่มข้อมูล INSERT**

|  |
| --- |
| **INSERT INTO phonebook ( fname, lname, email, office, province)**  **VALUES ( 'sineenat', 'phradmali', 'sineenat@health.moph.go.th', 'ศทส.', '12')** |

เป็นคำสั่งสำหรับการเพิ่มแถวข้อมูล หรือระเบียนใหม่เข้าตารางข้อมูล สำหรับฟิลที่เป็น Auto Increment ไม่ต้องใส่ค่า ระบบจะทำการเพิ่มค่าให้โดยอัตโนมัติ

**คำสั่ง UPDATE**

|  |
| --- |
| **UPDATE phonebook**  **SET office= 'ศูนย์เทคโนโลยีสารสนเทศและการสื่อสาร'**  **WHERE fname=’sineenat’** |

เป็นคำสั่งเพื่อทำการเปลี่ยนแปลงข้อมูลในตารางให้เป็นตามที่ต้องการ

**คำสั่ง DELETE**

|  |
| --- |
| **DELETE FROM phonebook WHERE id=4** |

เป็นคำสั่งสำหรับลบข้อมูลจากตาราง โดยสามารถกำหนดเงื่อนไขสำหรับการลบได้ ซึ่งถ้าไม่ได้ระบุเงื่อนไข จะเป็นการลบข้อมูลทั้งหมดออกจากตาราง และถ้าอยู่ใน AUTOCOMMIT mode (กรณีปกติทั่วไป) จะไม่สามารถทำการยกเลิกการลบครั้งนี้ได้

**คำสั่ง TRUNCATE**

|  |
| --- |
| **TRUNCATE phonebook** |

เป็นคำสั่งที่ให้ผลเช่นเดียวกับ DELETE FROM table\_name แต่มีข้อแตกต่างดังนี้

- จะทำการดรอปตารางข้อมูลก่อน แล้วทำการสร้างใหม่ ซึ่งจะให้ผลที่เร็วกว่าการลบ

ตารางทั้งหมด

- การทำงานเป็น non-transaction ไม่สามารถทำการยกเลิกได้

**คำสั่ง FLUSH**

|  |
| --- |
| **FLUSH TABLES phonebook** |

เป็นคำสั่งที่ใช้สำหรับการลบหน่วยความจำภายใน (Internal Cache) ที่ใช้งาน ทั้งนี้การใช้คำสั่ง FLUSH จะอนุญาตเฉพาะ user ที่มีสิทธิ์ (Privilege) เท่านั้น

**Flush\_option สามารถระบุได้ดังนี้**

|  |  |
| --- | --- |
| **HOSTS** | ทำการล้าง hose cache table เพื่อปรับปรุงข้อมูลของ host หรือ client ที่  คอนเน็กต์เข้าใช้งาน ซึ่งอาจมีการเปลี่ยนแปลง IP address หรืออื่นๆ  โดยเมื่อทำการ flush เรียบร้อยแล้ว ก็จะอนุญาตให้ host นั้นๆ ทำการคอนเน็กต์เข้ามาใหม่อีกครั้งหนึ่ง |
| **LOGS** | ปิดและเปิด log files ใหม่ |
| **PRIVILEGES** | ทำการ reload สิทธิ์ต่างๆ ของ user จาก grant tables ของ MySQL Server |
| **TABLES** | ปิด tables ทั้งหมด หากตารางใดยังถูกเปิดอยู่ ก็จะถูก force ให้ปิดลง |

|  |
| --- |
| Data Retrieveal Language |

**คำสั่งแสดงข้อมูลอย่างง่าย**

|  |
| --- |
| **SELECT \* FROM phonebook ;** |

แสดงข้อมูลทุกคอลัมภ์จากตาราง phonebook

**คำสั่งแสดงข้อมูลเฉพาะคอลัมภ์ที่ต้องการ**

|  |
| --- |
| **SELECT fname ,lname,email FROM phonebook ;** |

แสดงข้อมูลเฉพาะคอลัมภ์ที่ต้องการแสดงจากตาราง phonebook โดยคั่นระหว่างคอลัมภ์ด้วยคอมมา(,)

**คำสั่งแสดงข้อมูลโดยระบุจำนวนรายการที่ต้องการ**

|  |
| --- |
| **SELECT fname ,lname,email FROM phonebook limit 5 ;**  **SELECT fname ,lname,email FROM phonebook limit 3,5 ;** |

แสดงข้อมูลโดยระบุจำนวนรายการที่ต้องการ

* Limit 5 แสดงข้อมูลตั้งแต่รายการแรก จนถึง 5 รายการ
* Limit 3,5 แสดงข้อมูลตั้งแต่รายการที่ 3(เริ่มรายการแรกเป็นลำดับที่ 0 ) ไปอีกจนครบ 5 รายการ

**คำสั่งแสดงข้อมูลโดยเรียงลำดับตามที่กำหนด**

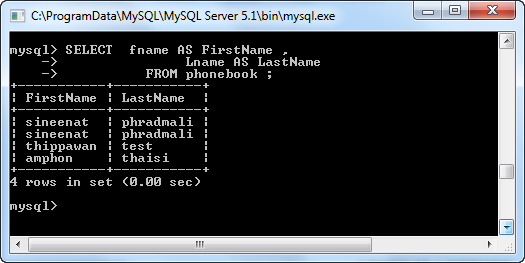
|  |
| --- |
| **SELECT fname ,lname,email FROM phonebook ORDER BY fname ;**  **SELECT fname ,lname,email**  **FROM phonebook**  **ORDER BY fname DESC ;** |

ตามปกติหากไม่ได้ระบุวิธีการจัดเรียงข้อมูล ก็จะเป็นการเรียงจากน้อยไปมาก (Ascending) หรือ คีย์เวิร์ด ASC แต่ถ้าต้องการเรียงจากมากไปน้อย (Descending) ให้ใช้คีย์เวิร์ด DESC

**การตั้งชื่อฟิลด์(Alias)**

|  |
| --- |
| **SELECT fname AS FirstName ,**  **Lname AS LastName**  **FROM phonebook ;** |

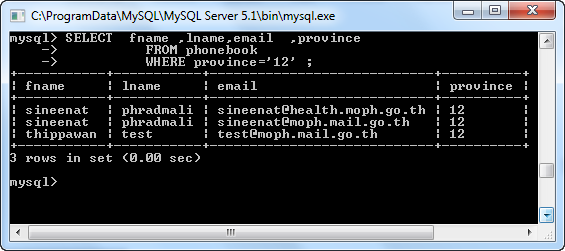
ปกติหากไม่ได้ระบุชื่อฟิลด์จะถือเอาชื่อฟิลด์ตามตารางข้อมูลเป็นหลัก หากต้องการตั้งชื่อใหม่ตามที่ต้องการ สามารถใช้ AS ตามหลังชื่อฟิลด์ แล้วใส่ชื่อที่ต้องการโดยชื่อที่ตั้งขึ้นใหม่นี้ห้ามมีช่องว่าง แต่สามารถใช้เครื่องหมาย Underscore(\_) ได้



**การระบุเงื่อนไข**

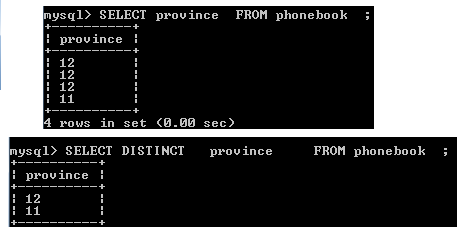
|  |
| --- |
| **SELECT fname ,lname,email ,province**  **FROM phonebook**  **WHERE province=’12’ ;** |

การกำหนดเงื่อนไขใช้ WHERE โดยหลังคำสั่ง WHERE จะเป็นเงื่อนไข เช่น province=’12’ หมายถึง แสดงเฉพาะข้อมูลในฟิลด์ province มีค่าเท่ากับ 12 ค่าในฟิลด์ province เป็นชนิด Character ต้องใส่ Quote (‘) ครอบ



**การข้อมูลที่ซ้ำซ้อน**

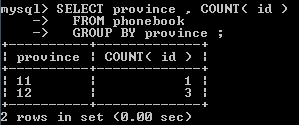
|  |
| --- |
| **SELECT DISTINCT province FROM phonebook ;**  **SELECT COUNT( province) FROM phonebook ;**  **SELECT COUNT( DISTINCT province ) FROM phonebook ;** |



**การจัดกลุ่มข้อมูล**

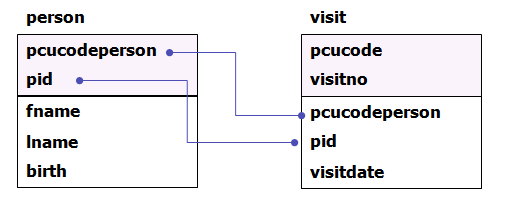
จัดกลุ่มข้อมูลที่หมือนกัน ไว้เป็นกลุ่มเดียวกันใช้ GROUP BY แล้วระบุชื่อฟิลด์ที่ต้องการจัดกลุ่ม

|  |
| --- |
| **SELECT province , COUNT( id )**  **FROM phonebook**  **GROUP BY province ;** |



|  |
| --- |
| การเรียกดูข้อมูลขั้นสูง |

การเรียกดูข้อมูลจาก 2 ตารางขึ้นไป เช่น ประชากร( Person ) และข้อมูลการรับบริการ(visit)



|  |
| --- |
| SELECT person.pid,prename,fname,lname,visitdate  FROM person,visit  WHERE person.pid=visit.pid  AND person.pcucodeperson=visit.pcucodeperson  Limit 10 ; |

การเรียกใช้ข้อมูลจาก 2 ตาราง หลัง FROM จะตามด้วยชื่อตารางคั่นระหว่างชื่อตารางด้วย Comma(,) เช่น FROM person,visit

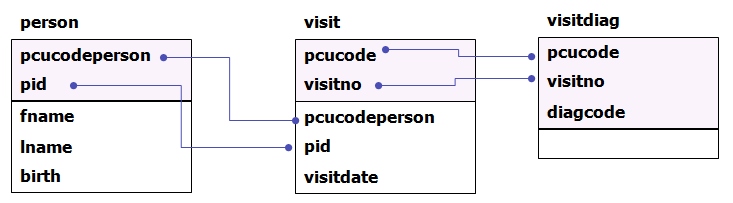
ในส่วนของความสัมพันธ์ของทั้งสองตารางใส่หลัง where โดยถ้าทั้งสองตารางมีชื่อเหมือนกันจะต้องจะต้องใส่ชื่อตารางนำหน้าตามด้วยจุด(.)แล้วจึงใส่ชื่อฟิลด์ โปรแกรมจะแสดงผลที่มีค่าตรงกันเท่านั้น เช่น WHERE person.pid=visit.pid AND person.pcucodeperson=visit.pcucodeperson

หากไม่ระบุค่าความสัมพันธ์ จะแสดงผลัพธ์เป็นผลคูณคาร์ทีเซียน

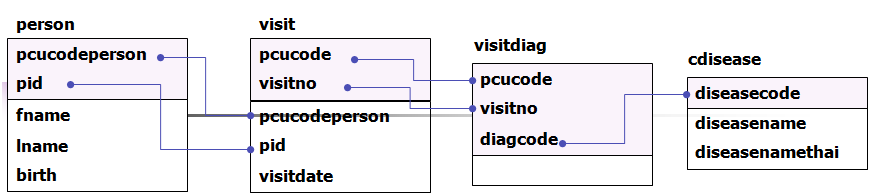
**การกำหนดชื่อย่อของตารางในการเชื่อมตาราง**

|  |
| --- |
| SELECT **p**.pid,prename,fname,lname,visitdate  FROM person **p** ,visit **v**  WHERE **p**.pid = **v**.pid  AND **p**.pcucodeperson = **v**.pcucodeperson  Limit 10 ; |

ประชากรแต่ละคนสามารถมารับบริการได้มากกว่า 1 ครั้ง แต่ละครั้ง Diagได้มากว่า 1โรค



|  |
| --- |
| SELECT person.pid,prename,fname,lname,visitdate ,diagcode  FROM person,visit  WHERE person.pid=visit.pid  AND person.pcucodeperson=visit.pcucodeperson  AND visit.pcucod=visitdiag.pcucode  AND visit.visitno=visitdiag.visitno  AND visitdate = ‘2010-02-01’ |

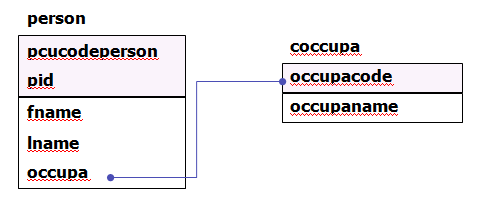


|  |
| --- |
| SELECT person.pid,prename,fname,lname,visitdate,diagcode,  diseasename  FROM person,visit,visitdiag,cdisease  WHERE person.pid=visit.pid  AND person.pcucodeperson=visit.pcucodeperson  AND visit.pcucod=visitdiag.pcucode  AND visit.visitno=visitdiag.visitno  AND visit.diagcode= cdisease.diseasecode  AND visitdate = ‘2010-02-01’ |

**การเชื่อมตารางทางขวาและทางซ้าย**

SQL Left Join เป็นคำสั่งที่ใช้สำหรับการระบุเงื่อนไขการเลือกใช้ข้อมูลในตาราง โดยเงื่อนไขการทำ Left join จะทำการเลือกข้อมูลหลัก โดยใช้ตารางที่อยู่ด้านซ้ายของคำสั่ง Join เป็นตัวตั้ง และเลือกตารางด้านขวาของคำสั่งที่มีค่าตรงกัน โดยที่ถ้ามีข้อมูลจากตารางด้านซ้าย ไม่ตรงกับด้านขวา ด้านขวาจะแสดงข้อมูลเป็น NULL

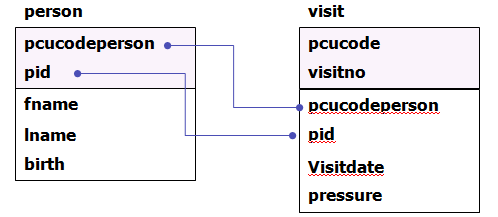
SQL Right Join เป็นคำสั่งที่ใช้สำหรับการระบุเงื่อนไขการเลือกใช้ข้อมูลในตาราง โดยเงื่อนไขการทำ Right join จะทำการเลือกข้อมูลหลัก โดยใช้ตารางที่อยู่ด้านขวาของคำสั่ง Join เป็นตัวตั้ง และเลือกตารางด้านซ้ายของคำสั่งที่มีค่าตรงกัน โดยที่ถ้ามีข้อมูลจากตารางด้านขวา ไม่ตรงกับด้านซ้าย ด้านซ้ายจะแสดงข้อมูลเป็น NULL



|  |
| --- |
| SELECT fname ,lname , occupa , occupaname  FROM person  **LEFT JOIN** coccupa  **ON** person.occupa = coccupa.occupacode ; |

|  |
| --- |
| การเรียกใช้ Subqueries |

เป็นการเรียกดูข้อมูลชุดหนึ่ง ภายในข้อมูลอีกชุดหนึ่ง



**แสดงรายชื่อประชากรที่ได้รับการวัด BP ในเดือนที่กำหนด ครั้งล่าสุด**

|  |
| --- |
| SELECT fname ,lname , occupa , occupaname  FROM person  **WHERE**  ( SELECT pcucodeperson , pid  FROM visit  WHERE month(visitdate) = '02' and pressure is not null  GROUP BY pcucodeperson , pid  Having max(visitdate)  ) |

|  |
| --- |
| **แสดงรายชื่อประชากรที่อยู่ในเขตรับผิดชอบที่ยังไม่เสียชีวิต** |

|  |
| --- |
| SELECT person.pid,prename,fname,lname,visitdate  FROM person  WHERE CONCAT(person.pid,person.pcucodeperson) NOT IN (  **SELECT CONCAT(persondeath.pid,persondeath.pcucodeperson)**  **FROM persondeath**  **WHERE persondeath.deaddate IS NULL**  **OR persondeath.deaddate<= ‘2011-10-30’**  ) |

|  |
| --- |
| การใช้งาน Function |

## Arithmetic operator

|  |  |
| --- | --- |
| + | บวก |
| - | ลบ |
| \* | คูณ |
| / | หาร |
| % | หารเอาเศษ |

## Comparison operator

|  |  |
| --- | --- |
| = | เท่ากับ |
| != หรือ <> | ไม่เท่ากับ |
| < | น้อยกว่า |
| <= | น้อยกว่าหรือเท่ากับ |
| > | มากกว่า |
| >= | มากกว่าหรือเท่ากับ |
| Exp BETWEEN Min AND Max | มีค่าระหว่างค่าต่ำสุด(Min) และค่าสูงสุด (MAX) |
| Exp IS NULL | เป็นค่าว่าง |
| Exp IS NOT NULL | ไม่เป็นค่าว่าง |
| Exp in(X1 ,X2,….,Xn) | มีค่าอยู่ในชุดข้อมูลที่เปรียบเทียบ |

## Logical operator

|  |  |
| --- | --- |
| AND หรือ && | และ |
| OR หรือ || | หรือ |
| NOT หรือ ! | ไม่ |
| XOR | Excusive OR |

## Mathematical Function

|  |  |
| --- | --- |
| TRUNCATE ( expression, decimal ) | ใช้ตัดเศษทศนิยมของ expression ทิ้ง ให้เหลือจุดทศนิยม จำนวนตำแหน่ง decimal ที่กำหนด |
| ROUND(n , d) | ใช้ปัดตัวเลขทศนิยมของ n ให้มีจำนวนจุดทศนิยมเท่ากับ d ถ้าไม่กำหนด d จะปัดเป็นจำนวนเต็ม |
| FLOOR(n) | ใช้ปัดเศษของ expression ให้มีค่าน้อยลง |
| CEILING(n) | ใช้ปัดเศษของ expression ให้มีค่าขึ้น |
|  |  |

## String Function

|  |  |
| --- | --- |
| CONCAT ( char1, char2, ..., charN ) | ใช้นำสตริงของ char1, char2, ..., charN นำมาต่อกัน |
| LENGTH ( char ) | ใช้หาค่าความยาวของ char ที่กำหนด |
| LEFT ( sentence, length ) | ใช้คืนค่า sentence โดยจะตัดสตริงให้เหลือเพียงจำนวนตัวอักษร length โดยจะเริ่มนับจากทางซ้าย |
| RIGHT ( sentence, length ) | ใช้คืนค่า sentence โดยจะตัดสตริงให้เหลือเพียงจำนวนตัวอักษร length โดยจะเริ่มนับจากทางขวา |
| SUBSTRING ( sentence, position, length ) | ใช้ตัดสตริงของ sentence โดยจะเริ่มตัดจากตำแหน่ง position และตัดไปจำนวน length ตัวอักษร |
| SUBSTRING ( sentence FROM position FOR length ) | ใช้ตัดสตริงของ sentence โดยจะเริ่มตัดจากตำแหน่ง position และตัดไปจำนวน length ตัวอักษร |
| MID ( sentence, position, length ) | ใช้ตัดสตริงของ sentence โดยจะเริ่มตัดจากตำแหน่ง position และตัดไปจำนวน length ตัวอักษร |
| SUBSTRING ( sentence, position ) | ใช้ตัดสตริงของ sentence โดยจะเริ่มตัดจากตำแหน่ง position และตัดจนจบประโยค |
| SUBSTRING ( sentence FROM position ) | ใช้ตัดสตริงของ sentence โดยจะเริ่มตัดจากตำแหน่ง position และตัดจนจบประโยค |
| SUBSTRING\_INDEX ( sentence, delimeter, count ) | ใช้ตัดสตริงของ sentence โดยมีรูปแบบของการตัด |
| LTRIM ( sentence ) | ใช้ต้ดช่องว่างทางซ้ายของ sentence ออกไป |
| RTRIM ( sentence ) | ใช้ต้ดช่องว่างทางขวาของ sentence ออกไป |
| TRIM ( sentence ) | ใช้ต้ดช่องว่างทางซ้ายและทางขวาของ sentence ออกไป |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| LCASE ( sentence ) | ใช้แปลงค่าตัวอักษรภายใน sentence ให้เป็น ตัวพิมพ์เล็กทั้งหมด |
| LOWER ( sentence ) | ใช้แปลงค่าตัวอักษรภายใน sentence ให้เป็น ตัวพิมพ์เล็กทั้งหมด |
| UCASE ( sentence ) | ใช้แปลงค่าตัวอักษรภายใน sentence ให้เป็น ตัวพิมพ์ใหญ่ทั้งหมด |
| UPPER ( sentence ) | ใช้แปลงค่าตัวอักษรภายใน sentence ให้เป็น ตัวพิมพ์ใหญ่ทั้งหมด |

## Date and Time Function

|  |  |
| --- | --- |
| DAYOFWEEK ( date ) | ใช้คืนค่า วันที่ในสัปดาห์ โดยค่าที่เป็นไปได้ได้แก่ 1 ถึง 7 คือเรียงจาก 1=อาทิตย์ จนถึง 7=เสาร์ |
| WEEKDAY ( date ) | ใช้คืนค่า วันที่ในสัปดาห์ โดยค่าที่เป็นไปได้ได้แก่ 0 ถึง 6 คือเรียงจาก 0=จันทร์ จนถึง 6=อาทิตย์ |
| DAYOFMONTH ( date ) | ใช้คืนค่าวันที่ของเดือน โดยค่าที่เป็นไปได้ได้แก่ 1 ถึง 31 |
| DAYOFYEAR ( date ) | ใช้คืนค่าวันที่ของปี โดยค่าที่เป็นไปได้ได้แก่ 1 ถึง 366 |
| MONTH ( date ) | ใช้คืนค่าเดือนที่ของปี โดยค่าที่เป็นไปได้ได้แก่ 1 ถึง 12 |
| YEAR ( date ) | ใช้คืนค่าปีคริสต์ศักราช โดยค่าที่เป็นไปได้ได้แก่ 1000 ถึง 9999 |
| YEARWEEK ( date ) | ใช้คืนค่าปีคริสต์ศักราช และลำดับที่ของสัปดาห์ |
|  | ใช้คืนค่าชั่วโมงที่ โดยค่าที่เป็นไปได้ได้แก่ 0 ถึง 23 |
|  | ใช้คืนค่านาทีที่ โดยค่าที่เป็นไปได้ได้แก่ 0 ถึง 59 |
|  | ใช้คืนค่าวินาทีที่ โดยค่าที่เป็นไปได้ได้แก่ 0 ถึง 59 |
| DATE\_ADD ( date, INTERVAL num datepart ) | ใช้เพิ่มวันที่หรือเวลาตามส่วน datepart ที่กำหนด โดยจะเพิ่มเข้าไปจำนวน num หน่วย |
| DATE\_SUB ( date, INTERVAL num datepart ) | ใช้หักลบวันที่หรือเวลาตามส่วน datepart ที่กำหนด โดยจะเพิ่มเข้าไปจำนวน num หน่วย |
| ADDDATE ( date, INTERVAL num datepart ) | ใช้เพิ่มวันที่หรือเวลาตามส่วน datepart ที่กำหนด โดยจะเพิ่มเข้าไปจำนวน num หน่วย |
| SUBDATE ( date, INTERVAL num datepart ) | ใช้หักลบวันที่หรือเวลาตามส่วน datepart ที่กำหนด โดยจะเพิ่มเข้าไปจำนวน num หน่วย |
|  |  |
| DATEDIFF(date1,date2) | หาจำนวนวันระหว่าง date1 และ date2 |
| DATE\_FORMAT ( date, format ) | ใช้จัดรูปแบบวันที่ให้เป็นไปตาม format ที่กำหนด |
| CURDATE ( ) | ใช้คืนค่าวันที่ปัจจุบัน โดยมีรูปแบบคือ YYYY-MM-DD หรือ YYYYMMDD |
| CURRENT\_DATE ( ) | ใช้คืนค่าวันที่ปัจจุบัน โดยมีรูปแบบคือ YYYY-MM-DD หรือ YYYYMMDD |
| NOW ( ) | ใช้คืนค่าวันที่และเวลาปัจจุบัน โดยมีรูปแบบคือ YYYY-MM-DD HH:MM:SS หรือ YYYYMMDDHHMMSS |
| SYSDATE ( ) | ใช้คืนค่าวันที่และเวลาปัจจุบัน โดยมีรูปแบบคือ YYYY-MM-DD HH:MM:SS หรือ YYYYMMDDHHMMSS |
| CURRENT\_TIMESTAMP ( ) | ใช้คืนค่าวันที่และเวลาปัจจุบัน โดยมีรูปแบบคือ YYYY-MM-DD HH:MM:SS หรือ YYYYMMDDHHMMSS |

## Comparison Function

|  |  |
| --- | --- |
| ISNULL ( expression ) | ใช้ตรวจสอบว่า expression ที่กำหนดเป็นค่า NULL ใช่หรือไม่ |
| COALESCE ( ) | ใช้คืนค่า ค่าข้อมูลของ expression ตัวแรก ที่ไม่ใช่ค่า NULL หรือคืนค่า NULL ถ้า expression ทั้งหมดเป็นค่า NULL |
| INTERVAL ( ) | ใช้คืนค่า 0 ถ้า expression1 < expression2 คืนค่า 1 ถ้า expression2 < expression3 และจะเป็นอย่างนี้ไปเรื่อยๆจนกว่าจะถึงค่าสุดท้าย |
| STRCMP ( exp1, exp2 ) | ใช้เปรียบเทียบสตริง |

## Control Flow Function

|  |  |
| --- | --- |
| IFNULL ( expression1, expression2 ) | ใช้ตรวจสอบว่า expression1 เป็นค่า NULL ใช่หรือไม่ |
| NULLIF ( expression1, expression2 ) | ใช้คืนค่า NULL ถ้าหากว่า expression1 และ expression2 มีค่าเท่ากัน |
| IF (exp , true ,false ) | เป็นเงื่อนไข โดยถ้า expression boolean คืนค่า true กลับมา |
| CASE WHEN THEN ELSE END | ใช้กำหนดเงื่อนไขของ expression ให้มีหลายๆเงื่อนไข |

## Miscellaneous Function

|  |  |
| --- | --- |
| SESSION\_USER ( ) | ใช้คืนค่า username ที่ติดต่ออยู่ |
| PASSWORD ( char ) | ใช้เข้ารหัส ด้วย function password |
| ENCRYPT ( char ) | ใช้เข้ารหัสด้วย function encrypt |
| ENCODE ( char, code ) | ใช้เข้ารหัสด้วย function encode โดยมีการกำหนด code ด้วย |
| DECODE ( char, code ) | ใช้ถอดรหัสด้วย function decode ซึ่งเป็นการถอดรหัสของค่าที่ได้จากการเข้ารหัส encode โดยมีการกำหนด code ด้วย |
| MD5 ( char ) | ให้เข้ารหัสด้วย function md5 โดยค่าที่จะได้คือ เลขฐานสิบหก ความยาว 32 ตัวอักษร |
| FORMAT ( num, decimal ) | ใช้จัดรูปแบบการแสดงผลตัวเลข แบบมี comma ทุกๆ 3 หลักของตัวเลข โดยมีการกำหนดจุดทศนิยมไว้ด้วย |
| VERSION ( ) | ใช้คืนค่า version ของ mysql |
| CONNECTION\_ID ( ) | ใช้คืนค่า id ของการเชื่อมต่อฐานข้อมูล |

## Aggreation Function

|  |  |
| --- | --- |
| COUNT ( column\_name ) | ใช้นับจำนวนแถวทั้งหมด แต่จะไม่นับค่าข้อมูลที่เป็น NULL โดยค่าที่คืนกลับมาจะเป็นชนิด int |
| SUM ( column\_name ) | ใช้หาผลรวมของค่าข้อมูลในทุกแถว |
| AVG ( column\_name ) | ใช้หาค่าเฉลี่ยของค่าข้อมูลในทุกแถว |
| MAX ( column\_name ) | ใช้หาค่าสูงสุด เมื่อเทียบกับค่าข้อมูลในทุกแถว ถ้าใช้กับข้อมูลที่เป็นตัวอักษร จะแสดงผลแถวแรกสุด |
| MIN ( column\_name ) | ใช้หาค่าต่ำสุด เมื่อเทียบกับค่าข้อมูลในทุกแถว ถ้าใช้กับข้อมูลที่เป็นตัวอักษร จะแสดงผลแถวท้ายสุด |

﻿concat

|  |
| --- |
| SELECT concat(fname ,’ ’,lname) AS name FROM person |

case

|  |
| --- |
| SELECT concat(fname ,' ',lname) AS name ,  CASE WHEN sex='1' THEN 'ชาย'  WHEN sex='2' THEN 'หญิง'  ELSE '-' END AS sex  FROM person |

อายุ(ปี)

|  |
| --- |
| SELECT getAgeYearNum(birth,current\_date) AS age  FROM person |

sum

|  |
| --- |
| SELECT SUM(CASE WHEN sex='1' THEN 1 ELSE null END) AS m,  SUM(CASE WHEN sex='2' THEN 1 ELSE null END) AS w ,  COUNT(sex ) AS total  FROM person |

DATE\_FORMAT

|  |
| --- |
| SELECT DATE\_FORMAT( birth ,"%y-%m-%d") as birthdate  FROM person |

max

|  |
| --- |
| SELECT DATE\_FORMAT( birth ,"%y-%m-%d") as birthdate  FROM person |

convert

|  |
| --- |
| SELECT convert(ifnull(visit.`money2`,0.0) using utf8) AS visit\_money2  FROM visit |

|  |
| --- |
| การสร้าง View |

เมื่อมีการใช้คำสั่ง SQL ที่ซับซ้อน ผู้ใช้จะมีความยากลำบากในการเขียนคำสั่งที่มีการ Join ตารางหลายๆตาราง มีการใช้ Function ในการใช้งานมากมาย หากมีการเรียกใช้งานคำสั้งนั้นบ่อยๆ แล้ว วิธีปัญหานั้นคือการสร้าง View

View คือมุมมองของข้อมูลที่ผู้ใช้สร้างขึ้นใหม่โดย View จะถูกสร้างมาจากตาราง 1 ตาราง หรือหลายๆตารางก็ได้

ประโยชน์ของ View

1. สามารถสร้าง View สำหรับกลุ่มผู้ใช้ต่างๆด้วยเงื่อนไขต่างๆกัน
2. สามารถสร้างจาก Query ที่ซับซ้อน ให้งาสยต่อการใช้งาน
3. จำกัดการเข้าถึงข้อมูล เนื่องจากการสร้าง View จะเป็นการเลือกเฉพาะฟิลด์ที่ต้องการแสดง

**Create View Syntax**

|  |
| --- |
| CREATE  [OR REPLACE]  [ALGORITHM = {UNDEFINED | MERGE | TEMPTABLE}]  [DEFINER = { *user* | CURRENT\_USER }]  [SQL SECURITY { DEFINER | INVOKER }]  VIEW *view\_name* [(*column\_list*)]  AS *select\_statement*  [WITH [CASCADED | LOCAL] CHECK OPTION] |

เช่น การนับจำนวนประชากร จำแนกตามเพศ โดยไม่นับคนที่เสียชีวิตแล้ว

|  |
| --- |
| **CREATE OR REPLACE VIEW v\_count\_person AS (**  SELECT SUM(CASE WHEN sex='1' THEN 1 ELSE null END) AS m,  SUM(CASE WHEN sex='2' THEN 1 ELSE null END) AS w ,  COUNT(sex ) AS total  FROM person  WHERE CONCAT(person.pid,person.pcucodeperson) NOT IN (  SELECT CONCAT(persondeath.pid,persondeath.pcucodeperson)  FROM persondeath  )  **)** |

วิธีเรียกใช้งาน View

|  |
| --- |
| select \* from v\_count\_person ; |

|  |
| --- |
| การกำหนดสิทธิการใช้งานฐานข้อมูล |

คำสั่ง GRANT และ REVOKE

|  |
| --- |
| GRANT ALL PRIVILEGES ON demo TO user1@localhost WITH GRANT OPTION  REVOKE ALL PRIVILEGES ON demo FROM user1@localhost |

* คำสั่ง GRANT เป็นคำสั่งสำหรับการกำหนดสิทธิ์หรือความสามารถใดๆ ให้กับ user
* คำสั่ง REVOKE หรือคำสั่งที่ตรงกันข้ามกับ GRANT คือเป็นการถอนหรือยกเลิกสิทธิ์ใดๆจากUser
* การใช้คำสั่ง GRANT หรือ REVOKE จะอนุญาตเฉพาะผู้บริหารระบบในการกำหนดสิทธิ์ต่างๆ สำหรับผู้ใช้

**MySQL จะแบ่งสิทธิ์ (Privilege) เป็น 4 ระดับ คือ**

1. **Global level** ระดับบนสุด ครอบคลุมทุก database รายละเอียดสำหรับสิทธิ์ของระดับนี้จะถูกเก็บไว้ในตาราง mysql.user
2. **Database level** ระดับ database ครอบคลุมเฉพาะใน database อันใดอันหนึ่ง เช่น ความสามารถในการกระทำใดๆ ภายใน database นั้นๆ รายละเอียดสำหรับสิทธิ์ของระดับนี้ จะถูกเก็บไว้ในตาราง mysql.db และ mysql.host
3. **Table level** ระดับ table ครอบคลุมเฉพาะภายในตารางนั้นๆ เช่น ความสามารถในการกระทำกับ table นั้นๆ สามารถทำการ select, insert, delete, update ได้หรือไม่ เป็นต้น รายละเอียดสำหรับสิทธิ์ของระดับนี้ จะถูกเก็บอยู่ในตาราง mysql.tables\_priv
4. **Column level** ระดับ column ครอบคลุมเฉพาะฟิลด์ใดฟิลด์หนึ่งภายในตารางข้อมูล รายละเอียดสำหรับสิทธิ์ของระดับนี้ จะถูกเก็บอยู่ในตาราง mysql.columns\_priv รายละเอียด priv\_type มีดังต่อไปนี้

|  |  |  |
| --- | --- | --- |
| ALL PRIVILEGES | FILE | RELOAD |
| ALTER | INDEX | SELECT |
| CREATE | INSERT | SHUTDOWN |
| DELETE | PROCESS | UPDATE |
| DROP | USAGE |  |

* เราสามารถใช้ ALL หรือ ALL PRIVILEGES ได้อย่างใดอย่างหนึ่ง
* USAGE มีความหมายเท่ากับ no privileges โดยใช้กรณีที่เราต้องการ create user และกำหนดให้มีสิทธิ์เป็น no privileges
* การถอนสิทธิ์การ GRANT ของ user ใดๆ ให้ใช้ priv\_type เป็น GRANT OPTION ตัวอย่าง: REVOKE GRANT OPTION ON … FROM …;
* การใช้ priv\_type สำหรับ table: SELECT, INSERT, UPDATE, DELETE, CREATE, DROP,GRANT, INDEX และ ALTER
* การใช้ priv\_type สำหรับฟิลด์ SELECT, INSERT, UPDATE\
* การกำหนดสิทธิ์ระดับ database ใช้ ON \*.\*
* สำหรับ user\_name สามารถระบุได้ถึงโฮสต์ที่ใช้งาน เช่น mysql@localhost หรือ root@’%’

**การกำหนดสิทธิของผู้ใช้งาน**

หลักการที่ MySQL ใช้ เรียกว่า Access Control Lists (ACLs) จะมีการตรวจสอบสิทธิผู้ใช้งาน ว่าแต่ละคนมีสิทธิจะทำอะไรได้บ้าง กับฐานข้อมูลใดบ้าง ซึ่งสิทธิเหล่านี้จะถูกบันทึกเอาไว้ในตารางชื่อ mysqlข้อควรกระทำและไม่ควรกระทำในการป้องกันหรือรักษาความปลอดภัยของข้อมูล

1. ห้ามอนุญาตให้ใครก็ตาม ที่ไม่ใช้ผู้ดูแลระบบ มีสิทธิเข้าไปเรียกดูหรือใช้งานตารางชื่อ mysqlเพราะ mysql เป็นแหล่งที่เก็บข้อมูลเกี่ยวกับสิทธิต่าง ๆ
2. อย่างกำหนดสิทธิให้กับผู้ใช้งานใด ๆ มากเกินความจำเป็นผู้ใช้แต่ละคนควรจะได้รับสิทธิแตกต่างกัน และควรมีสิทธิจำกัดเป็นกรณี ๆ ไป เช่น หากเป็นผู้ใช้ทั่ว ๆ ไปควรจะมีสิทธิใช้งานแค่ฐานข้อมูลใดฐานข้อมูลหนึ่ง หรือ อาจจะมีสิทธิแค่ตารางใดตารางหนึ่งในแต่ละฐานข้อมูลระดับสิทธิการเรียกใช้ข้อมูลมี 4 ระดับคือ

2.1 ระดับโฮสต์ (host level)

2.2 ระดับฐานข้อมูล (db level)

2.3 ระดับตารางข้อมูล (table level)

* 1. ระดับคอลัมน์ข้อมูล (column level)

1. ระวังผู้ใช้ที่ใช้ชื่อว่า root root คือ ผู้สร้างฐานข้อมูล มีสิทธิต่าง ๆ มาก ข้อแนะนำว่าอย่าให้มีผู้ใช้ชื่อ root อยู่ในสารบบของ MySQL โดยเราสามารถกำหนดผู้ใช้ขึ้นมาให้มีสิทธิเทียบเท่า root และอีกประการคือ รายชื่อผู้ใช้ของ MySQL กับของ Server ที่ใช้งาน จะเป็นคนละชุดกัน
2. ระวังการตั้งรหัสผ่านของผู้ใช้ การตั้งรหัสผ่านควรหลีกเลี่ยงคำศัพท์ทั่วไปที่สามารถค้นหาได้จากดิกชินนารี, หลีกเลี่ยงชื่อเล่น หรือ ชื่อ สถานที่ที่ผู้บุกรุกอาจจะคาดเดาได้
3. หากไม่จำเป็นต้องใช้ Telnet ก็ปิดไปดีกว่า ในกรณีที่ Server ของเราสามารถรองรับการทำงานของโปรแกรม Telnet แต่ในทางปฏิบัติไม่มีความจำเป็นที่ต้องอนุญาตให้ใครมาใช้โปรแกรมนี้ ก็ควรยกเลิกการทำงาน
4. อย่าเชื่อมือผู้ใช้มากนัก เพราะหากผู้ใช้เผลอไปใช้คำสั่ง Drop database mysql ถ้าเราพัฒนาโปรแกรมขึ้นมาใช้ ควรจะมีระบบตรวจสอบคำสั่งนี้ โดยห้ามใช้คำสั่งนี้อย่างเด็ดขาด

**ระดับสิทธิของผู้ใช้**

1. **ระดับโฮสต์ (host level)** คือ ระดับนี้จะจำกัดสิทธิว่า ผู้ใช้สามารถติดต่อเข้ามายัง MySQL ได้จากที่ไหนบ้าง หรือจาก โฮสต์อะไรทั้งนี้เพราะถ้า MySQL ของเราอยู่ในเครื่องคอมพิวเตอร์ที่เชื่อมโยงกับเครื่องอื่น ๆหรือเชื่อมโยงกับเครือข่าย Internet การติดต่อสื่อสารก็ย่อมกระทำจากที่ไหนก็ได้ในโลก ผ่านทางโพรโตคอล TCP/IP ลักษณะการตรวจสอบสิทธิการใช้งาน เมื่อเครื่องลูกข่ายที่ร้องขอติดต่อเข้ามา จะต้องแจ้งให้ทราบว่ามาจากไหน ชื่อโฮสต์อะไร หมายเลข IP Address เท่าไร เมื่อ MySQL รับทราบการร้องขอ ก็จะนำข้อมูลที่ได้รับแจ้งไป ตรวจสอบว่าโฮสต์นั้นมีสิทธิหรือไม่หรือมีสิทธิอะไรบ้าง

ลักษณะชื่อโฮสต์ที่สามารถกำหนดได้

|  |  |
| --- | --- |
| ลักษณะชื่อโฮสต์ที่สามารถกำหนดได้ | หมายความว่าติดต่อมาจาก… |
| Localhost | เครื่องเดียวกันกับ MySQL server |
| % | เครื่องไหนก็ได้ |
| %.cc-moph | เครื่องใดก็ตามที่อยู่ในโดเมน cc-moph |
| 203.157% | เครื่องใดก็ตามที่มี IP address เริ่มจาก 203.157 |

ตัวอย่าง กำหนดสิทธิให้กับผู้ใช้ที่ชื่อว่า train ก็ระบุในลักษณะ train@host เช่น อนุญาตให้ train สามารถติดต่อมายัง MySQL ได้จากเครื่องหมายเลข 203.157.4.27 ก็ระบุเป็น train@203.157.4.27 ดังนั้นถ้าติดต่อ MySQL โดยใช้สคริปต์ PHP ก็ต้องเขียนคำสั่งในรูปแบบดังนี้

|  |
| --- |
| mysql\_connect(“MySQLHost”,”train”,”รหัสผ่าน”); |

คำสั่งกำหนดสิทธิการใช้งาน GRANT มีรูปแบบดังนี้

|  |
| --- |
| GRANT สิทธิต่าง ๆ ON ชื่อฐานข้อมูล.ชื่อตาราง TO ชื่อผู้ใช้งาน@โฮสต์ [IDENTIFILE BY ‘รหัสผ่าน’]  GRANT ALL ON \*.\* TO train@203.157.4.27 IDENTIFILE BY ‘aBcXyz’; |

สิทธิต่าง คือ สิทธิ (privilege) ที่จะสามารถทำได้เช่น การเรียกดูข้อมูล (SELECT), การลดข้อมูล(DELECT) หรือจะระบุสิทธิต่าง ๆ เป็น ALL ก็ได้ซึ่งหมายความว่าสามารถทำได้ทุกอย่าง

1. **ระดับฐานข้อมูล (db level)** ทั้งนี้ขึ้นอยู่กับว่ามีการระบุสิทธิต่าง ๆ ตามหลังคำสั่ง GRANT ถ้าระบุคำว่า ALL ก็หมายความว่าให้สิทธิในกรเรียกใช้ทุกคำสั่ง เช่น

|  |
| --- |
| GRANT สิทธิต่าง ๆ ON ชื่อฐานข้อมูล.ชื่อตาราง TO ชื่อผู้ใช้งาน@โฮสต์[IDENTIFILE BY ‘รหัสผ่าน’] |

GRANT ALL ON \*.\* TO train@203.157.4.27 IDENTIFILE BY ‘aBcXyz’;

GRANT SELECT ON \*.\* TO User1@localhost IDENTIFILE BY ‘Password’;

หมายความว่า อนุญาตให้ user1 สามารถใช้คำสั่ง SELECT สำหรับเรียกดูข้อมูลทุกตารางทุกฐานข้อมูล

1. **ระดับตารางข้อมูล (table level)** สิทธิระดับนี้จะสัมพันธ์กับสิทธิระดับฐานข้อมูล คือ ถ้าจะให้สิทธิในการใช้คำสั่งกับทุก ๆ ตารางใน ฐานข้อมูลก็ระบุด้วยเครื่องหมาย \* เช่น

|  |
| --- |
| GIS.\* หมายถึงอนุญาตให้มีสิทธิใช้คำสั่งกับทุกตารางที่มีอยู่ในฐานข้อมูลชื่อ GIS |

|  |
| --- |
| GRANT SELECT ON db1.table1 TO User1@localhost IDENTIFILE BY ‘aBcXyz’;  หมายความว่าอนุญาตให้ user1 เรียกใช้เฉพาะคำสั่ง SELECT เพื่อดึงข้อมูลจากตางรางชื่อ table1 ซึ่งอยู่ในฐานข้อมูล db1 เท่านั้น |

1. **ระดับคอลัมน์ข้อมูล (Column level)** เป็นสิทธิระดับที่ลึกลงไปจากสิทธิระดับอื่น ๆ อีก โดยมีความสัมพันธ์กับสิทธิระดบฐานข้อมูลและระดับตาราด้วย เพียงแต่วิธีการกำหนดสิทธิต้องระบุในตางรางที่ชื่อ column\_priv ซึ่งตารางนี้อยู่ในฐานข้อมูลmysql อีกที ฉะนั้นจึงต้องใช้คำสั่ง QL สั่งไปตรง ๆ ที่ตาราง column\_priv มิใช่กำหนดจาการใช้คำสั่ง GRANT เหมือนกับสิทธิระดับอื่น ๆ

**สิทธิต่าง ๆ หรือ privilege** หมายความว่า เมื่อเราอนุญาตให้ผู้ใช้เข้ามายังฐานข้อมูล, ตารางข้อมูล, หรือเข้ามายังMySQL ได้แล้ว สามารถใช้คำสั่ง SQL ทำอะไรได้บ้าง

|  |  |  |
| --- | --- | --- |
| สิทธิต่าง ๆ | การทำงาน | ตัวอย่างการใช้งาน |
| SELECT | เรียกดูรายการข้อมูล | SELECT \* FROM table1 |
| INSERT | เพิ่มรายการข้อมูล | INSERT INTO table1 … |
| UPDATE | ปรับปรุงแก้ไขรายการข้อมูล | UPDATE table1 SET … |
| DELETE | ลบรายการข้อมูล | DELETE FROM table1 |
| ALTER | แก้ไขโครงสร้างตารางข้อมูล | ALTER TABLE table1 ADD INDEX index1(name) |
| CREATE | สร้างตารางหรือฐานข้อมูลขึ้นมาใหม่ | CREATE DATABASE database1  CREATE TABLE table1 |
| DROP | ลบตารางหรือฐานข้อมูล | DROP DATABASE database1  DROP TABLE table1 |
| RELOAD | สั่งให้ server เซ็นค่าต่าง ๆ ใหม่ | FLUSH HOST (ปลดล็อคในกรณีที่ server มีการล็อค  ป้องกันการใช้งานบางอย่าง เช่น ป้องกันมิให้มีผู้ใช้งาน  พร้อมกันมาเกิดกำหนด) |
| SHUTDOWN | อนุญาตให้สั่งปิดการทำงานของ server | Mysqladmin shutdown |
| PROCESS | ใช้คำสั่งพิเศษกับ server เช่น SHOW | SHOW PROCESSLIST (ขอให้แสดงรายการ โพรเซสที่  กำลังทำงานอยู่) |
| FILE | อ่าน – เขียน ไฟล์ที่ server ได้ | LOAD DATA INFILE ‘data.txt’ INTO TABLE table1 (  โหลดข้อมูลจากไฟล์ชื่อ data.txt ไปบันทึกลงใน  ตารางข้อมูล tabel1) |

เมื่อคุณ install MySQL เสร็จ สิ่งแรกที่คุณควรทำ คือการกำ หนด password สำ หรับ root user โดยวิธีการ ดังนี้

|  |
| --- |
| shell> mysql -u root  mysql>use mysql;  mysql> UPDATE user SET Password=PASSWORD('new\_password') WHERE user='root';  mysql> FLUSH PRIVILEGES; |

**วิธีการกำหนด password**

จากตัวอย่างในหัวข้อที่ผ่านมาจะเห็นได้ว่าเมื่อคุณ INSERT หรือ UPDATE password คุณต้องใช้ function PASSWORD() เพื่อเข้ารหัส password เนื่องจากการกระทำ เช.นนี้ ทำ ให้ password ถูกเก็บลง table ในแบบที่ถูกเข้ารหัส เพื่อความปลอดภัยของข้อมูล หากคุณลืมใช้ function PASSWORD() , password ก็จะถูกเก็บในรูปแบบของตัวหนังสือ (plaintext)

|  |
| --- |
| การนำเข้าข้อมูล |

รูปแบบการใช้ LOAD DATA INFILE มีดังนี้

|  |
| --- |
| LOAD DATA [LOW\_PRIORITY] [LOCAL] INFILE 'file\_name.txt'  [REPLACE | IGNORE]  INTO TABLE tbl\_name  [FIELDS  [TERMINATED BY '\t']  [OPTIONALLY] ENCLOSED BY '']  [ESCAPED BY '\\' ]]  [LINES TERMINATED BY '\n']  [IGNORE number LINES]  [(col\_name,...)] |

LOCAL หมายถึง file ที่ถูกอ่านถูกเก็บไว้ที่ client ซึ่งจะทำ ให้เครื่องทำ งานช้าลง เนื่องจาก server จะต้องไปอ่าน file จากเครื่อง client ถ้าไม่มีคำ ว่า LOCAL , file จะต้องถูกเก็บไว้ที่ server และ การอ่านtext files ที่อยู่บน server file จะต้องอยู่ใน database directory หรือเป็น file ที่ user ทุกคนสามารถเปิดอ่านได้ นั่นคือถ้าคุณต้องการอ่าน file จาก server คุณจะต้องกำ หนด file privilege บน server host.

ถ้า LOW\_PRIORITY ถูกกำ หนด หมายถึงการทำ คำ สั่ง LOAD DATA จะถูกเลื่อนออกไปจนกระทั่งไม่มี clients เครื่องใดอ่านข้อมูลจากตาราง การหาตำแหน่งของ file บน server นั้น server จะต้องทราบ pathname ของ file เพื่อหาตำ แหน่งของ file โดยใช้กฎดังนี้ :

* ถ้า pathname เป็นแบบ absolute path, server จะใช้ pathname ในการค้นหา file
* ถ้า pathname เป็นแบบ relative pathname, server จะค้นหา file นั้นโดยอ้างอิง จาก directory ของ data ใน server
* ถ้า file name ไม่ได้กำ หนด path มาด้วย, server จะค้นหา file จาก database directory ของdatabase ที่ใช้ปัจจุบัน
* IGNORE number LINES ใช้สำ หรับทำ ให้ MySQL ไม่สนใจข้อมูล number บรรทัดของส่วนหัวข้อ file เช่น ขอ้ มลู บรรทัดแรกเป็นชื่อ ของ column เป็นต้น

|  |
| --- |
| LOAD DATA INFILE "/tmp/file\_name" into table test IGNORE 1 |

|  |
| --- |
| การ Export ข้อมูล |

สามารถใช้คำสั่งเพื่อส่งออกข้อมูลที่ต้องการออกเป็น Text File ได้ตามตัวอย่าง

|  |
| --- |
| SELECT idcard , fname ,lname  FROM person  INTO OUTFILE '/tmp/exdata.txt'; |

|  |
| --- |
| SELECT idcard , fname ,lname  INTO OUTFILE '/tmp/exdata.csv'  FIELDS TERMINATED BY ','  ENCLOSED BY '"'  ESCAPED BY '\\'  LINES TERMINATED BY '\n'  FROM person |

|  |  |
| --- | --- |
| SELECT idcard , fname ,lname | เลือกฟิลด์ข้อมูลที่ต้องการส่งออก |
| INTO OUTFILE '/tmp/exdata.csv' | ตำแหน่งและชื่อไฟล์ที่ต้องการส่งออก |
| FIELDS TERMINATED BY ',' | คั้นข้อมูลแต่ละฟิลด์ด้วยคอมมา |
| ENCLOSED BY '"' | ครอบข้อมูลด้วยเครื่องหมายคำพูด(“”) |
| ESCAPED BY '\\' | ใส่ Escape Character ด้วย Back Slash(\) |
| LINES TERMINATED BY '\n' | ขึ้นบรรทัดใหม้ของแต่ละรายการด้วย(\n) |
| FROM person | ชื่อตารางข้อมูลที่ต้องการส่งออก |

|  |
| --- |
| การสำรองข้อมูล |

ข้อมูลเป็นสิ่งที่สำคัญ ควรหาวิธีการป้องกันการเสียหายของข้อมูล โดยการสำรองข้อมูลให้มากเท่าที่สามารถจะทำได้ โดยการสำรองข้อมูลเพื่อให้ข้อมูลมีความครบถ้วนมากที่สุดนั้นมักจะบันทึกเป็น 3 ระดับคือ

* Full System Backup
* Full Data Backup
* Incremental Backup

       สำหรับการสำรองข้อมูลสามารถแบ่งออกเป็น 2 วิธีหลัก ๆ ดังนี้

**1. Offline Backup :** เป็นวิธีเบื้องต้นที่หลาย ๆ ท่านที่เคยทำการสำรองข้อมูลคงคุ้นเคย นั่นคือการหยุดให้บริการต่าง ๆ บนเครื่องเซิร์ฟเวอร์ทั้งหมด หรือการ Stop Service ต่าง ๆ ที่มีอยู่ทั้งหมดก่อนทำการสำรองข้อมูล (Backup) ซึ่งวิธีนี้อาจจะไม่เหมาะสำหรับองค์กรที่จำเป็นต้องทำการให้บริการผู้ใช้งาน ตลอด 24 ชม. ทั้ง 7 วัน

**2. Online Backup :** ความไม่สะดวกเกิดขึ้นเมื่อจำเป็นต้องทำการสำรองข้อมูล แต่ไม่สามารถหยุดการให้บริการได้ จึงได้มีการพัฒนาวิธีการทำการสำรองข้อมูลขึ้นมาอีกวีธีหนึ่งคือ สามารถทำการสำรองข้อมูลไปพร้อม ๆ กับการให้บริการต่าง ๆ แก่ผู้ใช้งานได้ ซึ่งเรียกวิธีนี้ว่า Online Backup หลายผลิตภัณฑ์ในท้องตลาดมีการทำงานลักษณะนี้

      ส่วนกระบวนการในการสำรองข้อมูล สามารถแบ่งออกตามลักษณะของทิศทางการไหลของข้อมูลได้ 2 ประเภท

**1. LAN Backup :** เป็นกระบวนการแบ็คอัพที่มีทิศ ทางการไหลของข้อมูลที่ต้องการแบ็คอัพไปตามเน็ตเวิร์ก เครื่องที่เป็น Backup Server คือเครื่องที่มี Tape Drive ต่ออยู่ ข้อมูลจาก Server อื่น ๆ จะถูกดึงออกมาผ่านทางเน็ตเวิร์กส่งไปยัง Backup Server

**2. LAN Free Backup :** กระบวนการแบ็คอัพสำหรับองค์กรที่ใช้งานตู้ Storage หรือที่เรียกกันว่า SAN ( Storage Area Network ) ที่เครื่องเซิร์ฟเวอร์ทำการเชื่อมต่อเพื่อเข้ามาใช้เนื้อที่ใน SAN ซึ่งข้อมูลสำคัญที่จำเป็นต้องการทำการ Backup

สำหรับ database ขนาดใหญ่แล้ว การใช้คำสั่ง mysqldump สำหรับสำรองข้อมูล นั้นมีประสิทธิภาพมากที่สุดแล้ว

ในการใช้คำสั่ง mysqldump เพื่อทำการ backup และ mysql เพื่อ restore ให้ใช้คำสั่งทั้งสองโดยที่ไม่ต้อง Login เข้า console ของ database

|  |
| --- |
| mysqldump [*options*] *db\_name* [*tbl\_name* ...]  mysqldump [*options*] --databases *db\_name* ...  mysqldump [*options*] --all-databases |

-u : username  
-p : password (ห้ามกรอกในบรรทัด ให้เว้นไว้ เมื่อ กด enter ระบบจะให้ใส่เอง)  
-h : host หรือ ip เครื่อง server (ใส่หรือไม่ก็ได้ หากใช้ localhost)

-P : Port Number ในกรณีที่ใช้ Default Port(3306) ไม่ต้องใส่ option นี้

สำหรับ option อื่นในคำสั่ง mysqldump สามารถดูได้จาก

|  |
| --- |
| mysqldump –h |

สำรองข้อมูลทั้งฐานข้อมูล

|  |
| --- |
| mysqldump -uroot –p123456 jhcisdb --lock-all-table  > /tmp/jhcisBackup\_$(date +%Y%m%d\_%H%M%S).sql |

สำรองข้อมูลเฉพาะตารางที่ต้องการ

|  |
| --- |
| mysqldump -u root -p jhcisdb -tables person > jhcis\_person.sql |

|  |
| --- |
| การนำข้อมุลสำรองมาใช้งาน |

เมื่อต้องการนำข้อมูลที่ทำการสำรองไว้มาใช้งาน สามารถนำข้อมูลจากไฟล์มานำเข้าฐานข้อมูลด้วยคำสั่งดังนี้

|  |
| --- |
| mysql -uUsername  -pPassword  DatabaseName  -PportNumber  -hServer < DatabaseName.sql |

เช่น mysql -hlocalhost -uroot –p123456 –P3333 jhcisdb < jhcisdb\_backup.sql

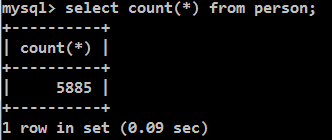
|  |
| --- |
| การเพิ่มประสิทธิภาพ และการซ่อมแซมตารางข้อมูล |

MySQL Optimize Table สิ่งที่ควรทำสำหรับตารางที่มี ลบ แก้ไขบ่อยๆ

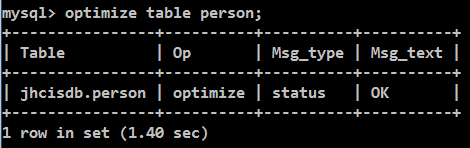
|  |
| --- |
| OPTIMIZE [NO\_WRITE\_TO\_BINLOG | LOCAL] TABLE  *tbl\_name* [, *tbl\_name*] ... |

ตัวอย่่าง

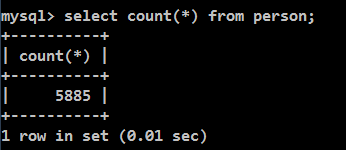
* ก่อนทำ Optimize



* คำสั่ง Optimize

****

* หลังจาก Optimize

****

คำสั่งสำหรับการตรวจสอบและซ่อมแซมฐานข้อมูล MySQL

|  |
| --- |
| mysqlcheck [options] db\_name [tables]  mysqlcheck [options] --databases DB1 [DB2 DB3...]  mysqlcheck [options] --all-databases |

mysqlcheck –h localhost -u root –P3333 -p --auto-repair –databases jhcisdb

ในการใช้คำสั่งนี้สามารถใช้ คำสั่ง หรือ option ต่างๆได้ดังนี้

|  |  |
| --- | --- |
| mysqlrepair | --repair |
| mysqlanalyze | --analyze |
| mysqloptimize | --optimize |
| --auto-repair | ถ้าตรวจสอบตารางแล้วพบความเสียหาย จะทำการซิอมให้โดยอัตโนมัติ |

ถ้าฐานข้อมูลเป็นแบบ myisam สามารถใช้คำสั่งนี้ โดยตรงกับไฟล์ฐานข้อมูลได้

|  |
| --- |
| myisamchk -r \*.MYI |

หรือถ้าหากต้องการ Repair เฉพาะ Table

|  |
| --- |
| mysql –h localhost –u root –P3333 -p  use jhcisdb; |

ตรวจสอบว่า Table นี้มีปัญหาหรือไม่

|  |
| --- |
| check table table\_name; |

และสามารถซ่อม Table ที่เสียหาย

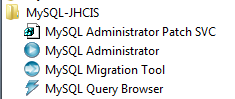
|  |
| --- |
| repair table table\_name; |

|  |
| --- |
| การใช้งาน MySQL Administrator |

โปรแกรม MySQL Administrator เป็นโปรแกรม ที่ใช้งานเพื่อช่วยในการบริหารจัดการฐานข้อมูล MySQL

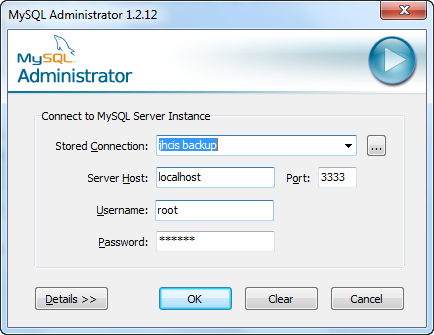
เริ่มใช้งานโปรแกรม

1. คลิก Start >> MySQL-JHCIS >> เลือก MySQL Administrator



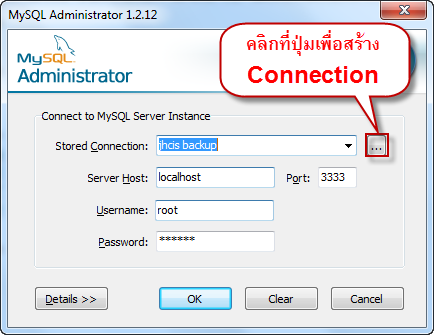
1. Connect to MySQL

กำหนดค่าเพื่อติดต่อกับฐานข้อมูล MySQL ดังรูป แล้วกดปุ่ม OK

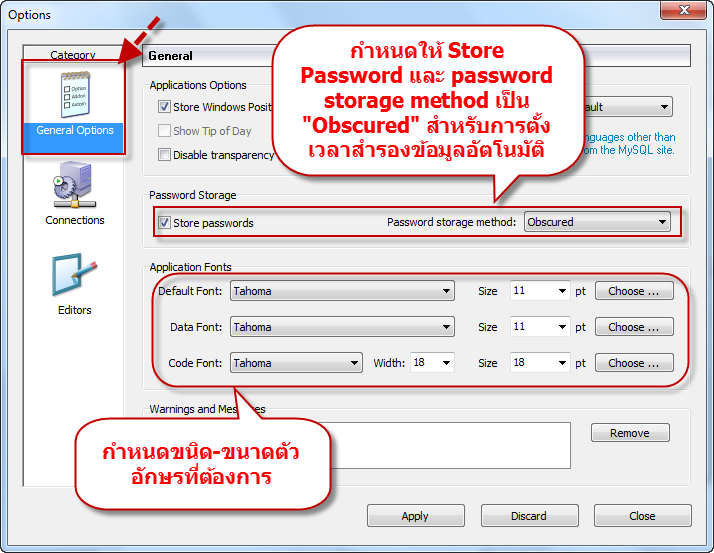


|  |  |
| --- | --- |
| Store Connection | สามารถสร้าง Connection ไว้ใช้งานได้โดย กดปุ่ม |
| Server Host | ชื่อ หรือ IP Address ของ MySQL Server ถ้าเป็นเครื่องเดียวกับ Server ใช้ชื่อ Localhost หรือ 127.0.0.1 |
| Port | Port Number ที่ใช้ สำหรับ JHCIS ใช้ port 3333 โดยปกติ MySQL ใช้ port 3306 |
| Username | ชื่อที่เข้าใช้งาน MySQL |
| Password | รหัสผ่านของ Username ที่เข้าใช้งาน MySQL |

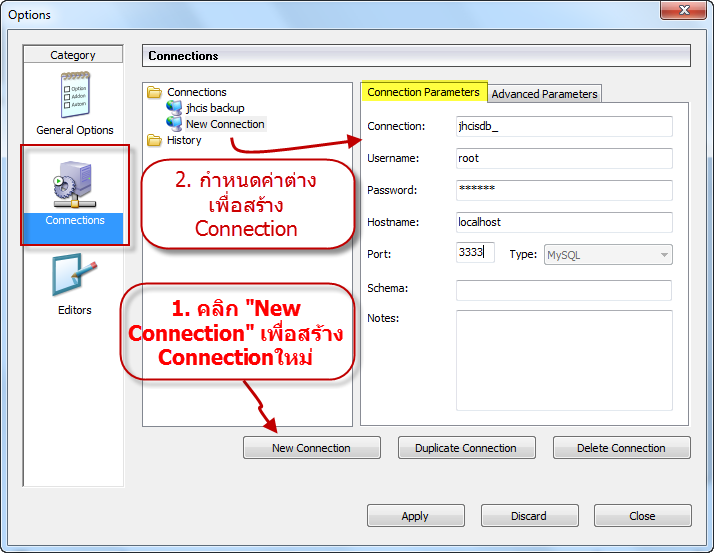
1. การสร้าง Connection



1. กำหนดค่าใน Category “General Options” แล้วกดปุ่ม Apply



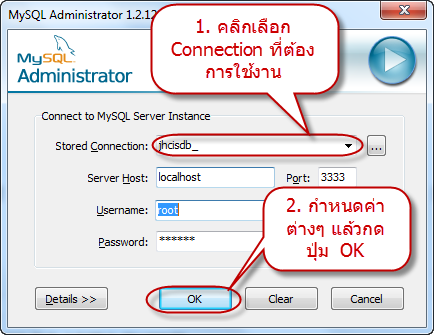
1. กำหนดค่าใน Category “Connections” แล้วกดปุ่ม Apply



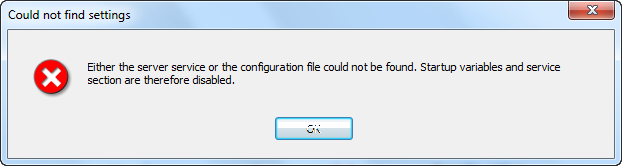
|  |  |
| --- | --- |
| **Connection Parameters** | |
| Store Connection | สามารถสร้าง Connection ไว้ใช้งานได้โดย |
| Server Host | ชื่อ หรือ IP Address ของ MySQL Server ถ้าเป็นเครื่องเดียวกับ Server ใช้ชื่อ Localhost หรือ 127.0.0.1 |
| Port | Port Number ที่ใช้ สำหรับ JHCIS ใช้ port 3333 โดยปกติ MySQL ใช้ port 3306 |
| Username | ชื่อที่เข้าใช้งาน MySQL |
| Password | รหัสผ่านของ Username ที่เข้าใช้งาน MySQL |

หากต้องการลบ Connection ให้คลิกที่ชื่อ Connections ที่ต้องการ แล้วกดปุ่ม Delete Connection แล้วกด Apply เพื่อบันทึกการลบ

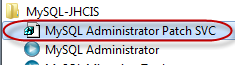
1. เมื่อกำหนดค่า Connection เสร็จแล้ว กดปุ่ม Apply เพื่อบันทึก แล้วกดปุ่ม Close เพื่อปิดหน้าต่าง
2. เลือก Connection ที่ต้องการ แล้วกดปุ่ม OK

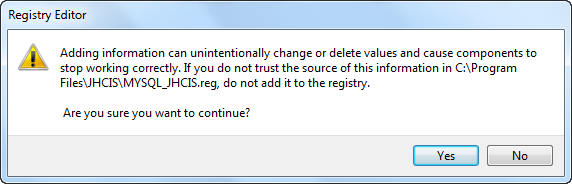


1. หากมีข้อความแสดงดังรูป ให้กดปุ่ม OK จากนั้นสามารถใช้งานได้ตามปกติ

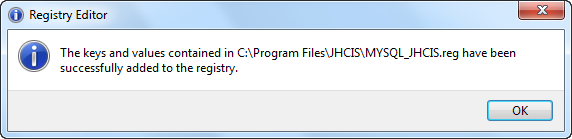


หากต้องการแก้ไขให้ไม่แสดงข้อความดังรูป ให้คลิก Start >> MySQL-JHCIS >> MySQL Adminstrator Patch SVC



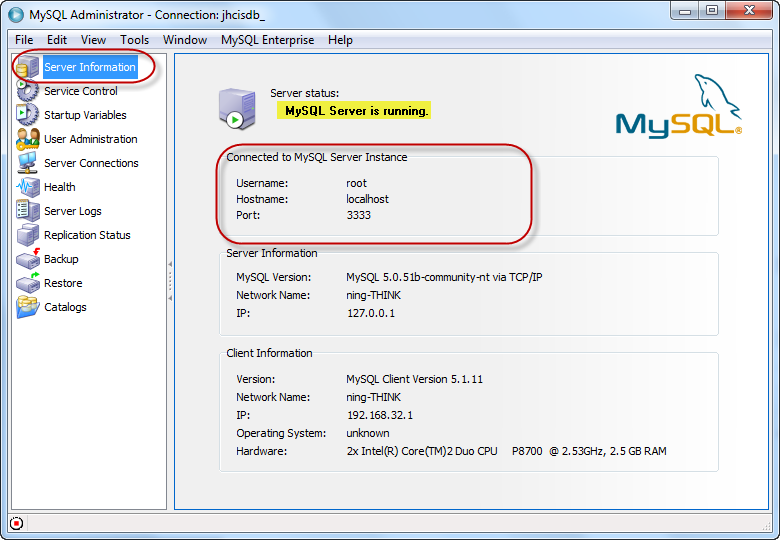


แสดงข้อความ Register Editor >> กดปุ่ม Yes



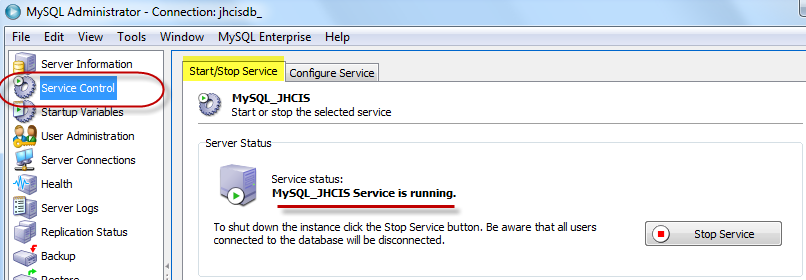
แสดงข้อความการแก้ไข Registry >> กดปุ่ม OK

1. แสดงข้อมูล Server Information



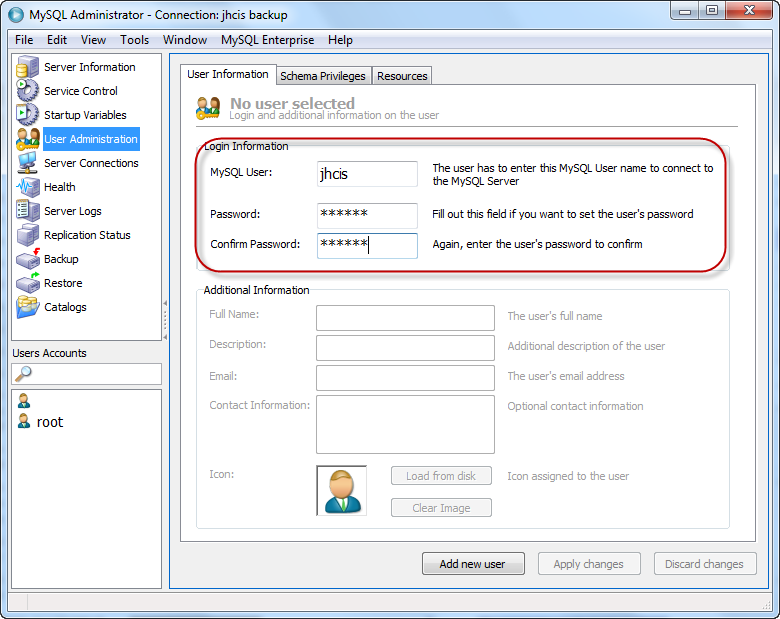
1. Service Control

Tab :: Start/Stop service



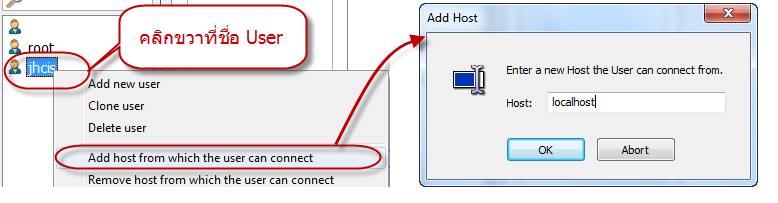
แสดงสถานะและชื่อ Service ที่ใช้งาน นั้นคือ Service ชื่อ “MySQL\_JHCIS” กำลังทหงานอยู่ หากต้องการหยุดการทำงาน ใหก้กดปุ่ม Stop Service

1. User Administration

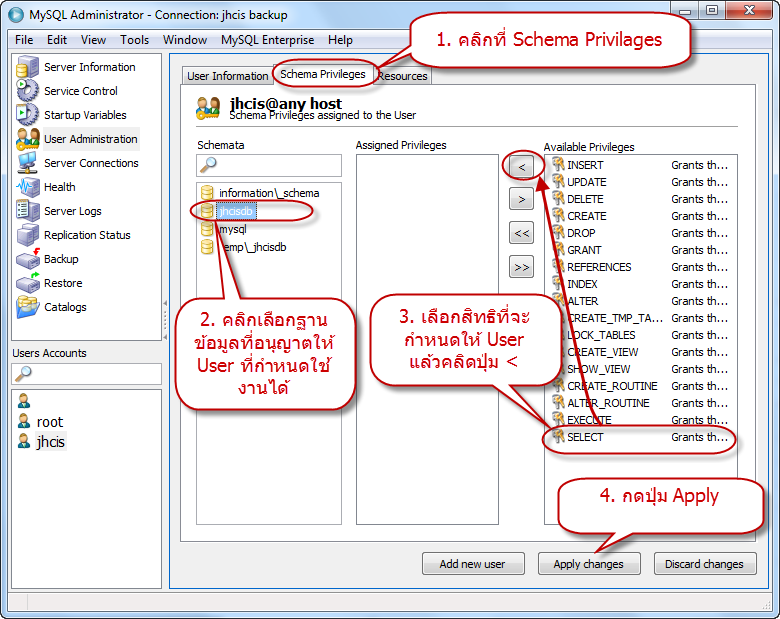


การเพิ่มผุ้ใช้ใน MySQL

* + - เลือก User Administrator แล้วคลิก ปุ่ม Add New User
    - หน้าจอ User Information จะแสดงขึ้นมาให้บันทึกข้อมูล
    - ใส่ User และ Password แล้วกดปุ่ม Apply Change
    - กำหนดการอนุญาตการให้ Connect จากเครื่องที่กำหนด

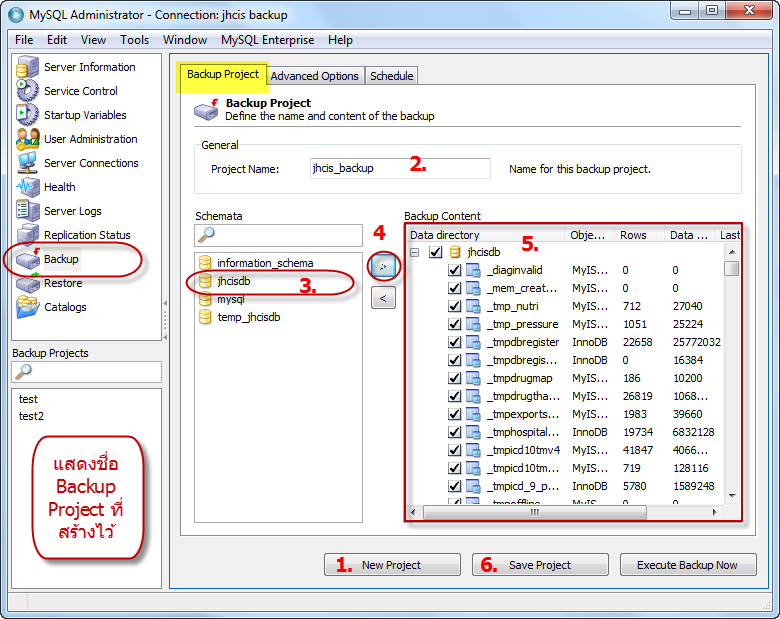


* + - คลิกที่ Tab :: Schema Privilage เพื่อกำหนดสิทธิในการใช้งาน



1. การสำรองข้อมูล

Backup คือ การสำรองข้อมูลไว้ เพื่อกู้ข้อมูลคืนในกรณีที่ เซิร์ฟเวอร์เกิดความเสียหายทั้งซอฟท์แวร์และฮาร์ดแวร์ การย้าย/เปลี่ยนเซิร์ฟเวอร์



1. กดปุ่ม New Project

2. กำหนดชื่อ Project เช่น jhcis\_backup

3. เลือกฐานข้อมูลที่ต้องการ Backup

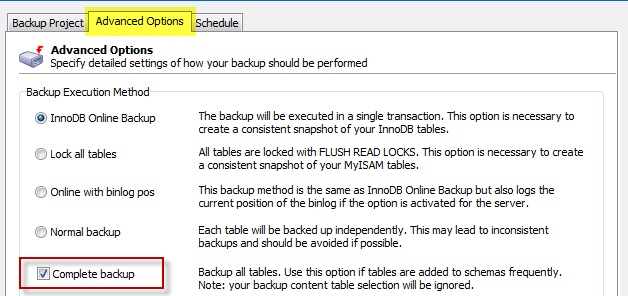
4. กดปุ่ม  เพื่อเลือกฐานข้อมูลที่ต้องการ Backup

5. ฐานข้อมูลที่ถูกเลือกจะแสดงใน Backup Content โดยส่วนที่ถูก Backup

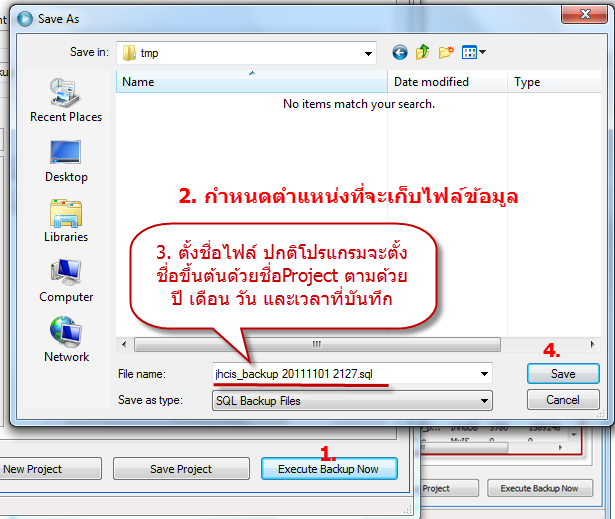
จะมีเครื่องหมาย 

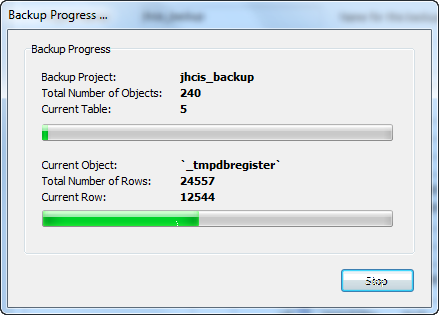
6. กดปุ่ม Save Project

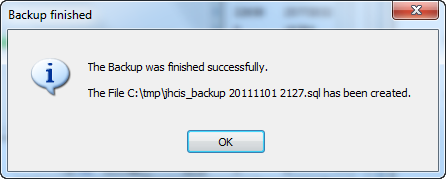
7. Tab:: Advance options เลือก Complete Backup



หากต้องการสำรองข้อมูลเองให้คลิกปุ่ม “Execute Backup Now”

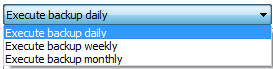




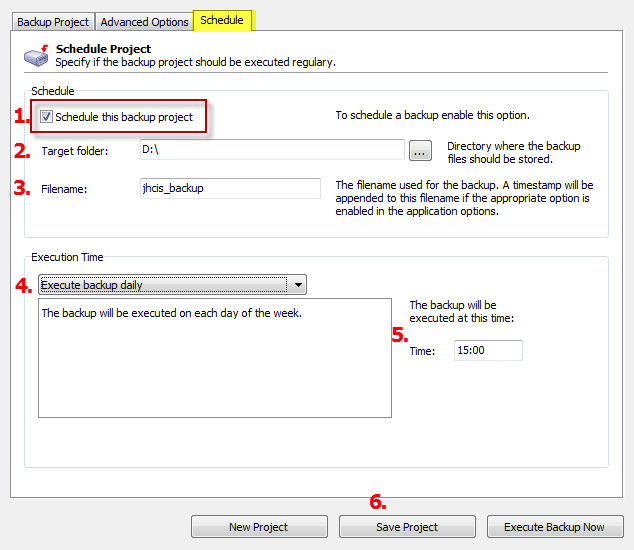


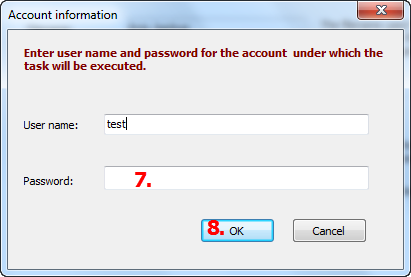
การตั้งเวลาสำรองข้อมูล

* + - เลือก Tab::Schedule
    - เลือกที่ Schedule this backup project (1)
    - กำหนดตำแหน่งที่ต้องการบันทึกข้อมูล (2)
    - กำหนดชื่อไฟล์โดยปกติจะใช้ชื่อ project เป็นชื่อไฟล์(3)
    - เลือกรูปแบบการสำรองข้อมูล (4)



* + - กำหนดเวลาที่ต้องการ โดยมีรูปแบบแป็น HH:MM:SS (5)
    - บันทึกโดยกดปุ่ม Save Project (6)
    - ใส่รหัสผ่านของชื่อผู้ใช้งาน windows ที่แสดงขึ้นมาในกล่องข้อความ (7)
    - กดปุ่ม Save (8)





1. Restore

Restore คือการนำไฟล์ข้อมูลที่เคยสำรอง(Backup)ไว้ กลับมาใช้งาน

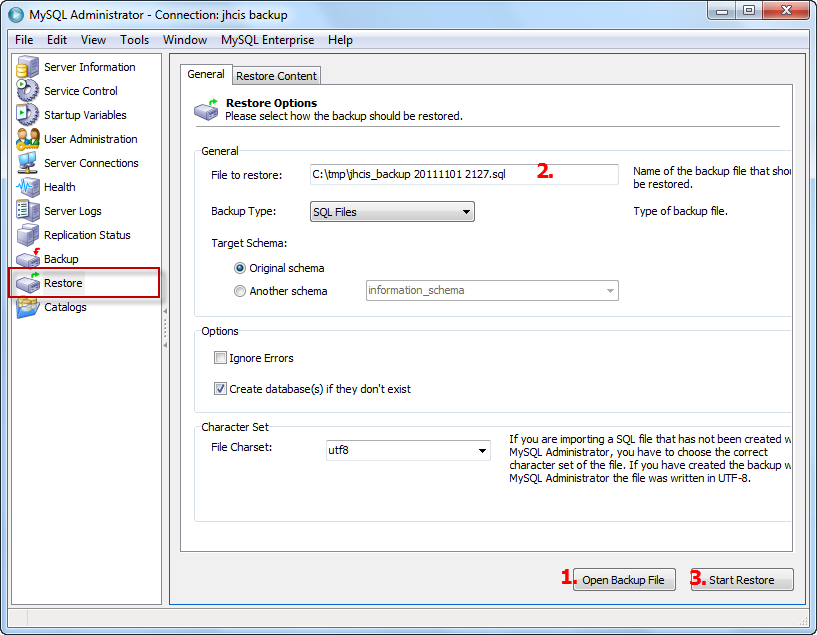
หมายเหตุ :: ไฟล์ SQL จากการใช้คำสั่ง mysqldump ไม่สามารถนำมา restore ด้วย MySQL Administrator

13.1 กดปุ่ม Open Backup File

13.2 เลือกไฟล์ที่ต้องการ Restoreแล้วกดปุ่ม Open

13.3 ชื่อไฟล์ที่เลือกจะแสดงในช่อง File to restore (2)

13.4 กดปุ่ม Start Restore

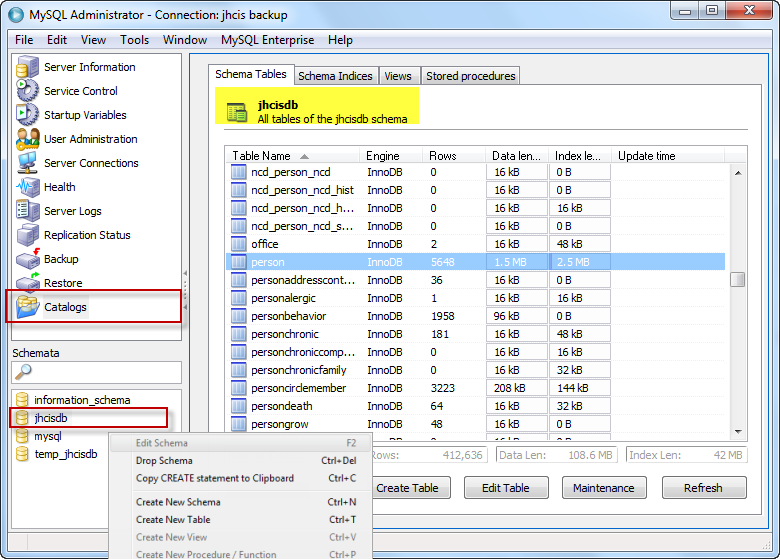


1. Catalog

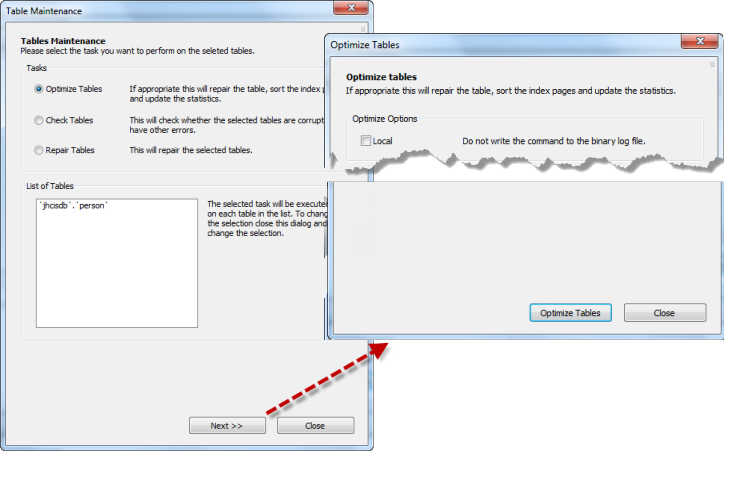
Catalog จะแสดงรายชื่อฐานข้อมูล(Database) ถ้าคลิกเลือกฐานข้อมูลที่ต้องการแล้ว โปรแกรมจะแสดงรายชื่อตารางข้อมูล ประเภทตารางและจำนวนข้อมุลใน Tab :: Schema Tables

ถ้าต้องการลบฐานข้อมูล ให้คลิกเลือกฐานข้อมูลที่ต้องการลบ >> คลิกขวา >> Drop Schema

กรณีที่ต้องการสร้างฐานข้อมูลใหม่ ให้คลิกเลือกฐานข้อมูล >> คลิกขวา >> Create New Schema



การตรวจสอบและดูแลตารางข้อมูล คลิกตารางข้อมูลที่ต้องการแล้วกดปุ่ม Maintainance >> Optimize

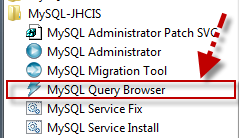


|  |
| --- |
| การใช้งาน MySQL Query Browser |

ในการใช้งาน MySQL ที่ในบทผ่านมาเป็นการใช้งานในรูปแบบ command line ซึ่งผู้ที่เริ่มใช้งาน MySQL อาจรู้สึกว่าไม่สะดวกในการใช้งาน ปัจจุบันมีโปรแกรมที่ช่วยให่การใช้งาน MySQL สะดวกขึ้นหลายโปรแกรม เช่น MySQL Front , Navicat เป็นต้น แต่ในบทนี้จะแนะนำเครื่องมือดีๆ ที่จะทำให้การใช้งาน mysql ง่ายขึ้นโดยโปรแกรมนี้จะถูกติดตั้งมาพร้อมกับ JHCIS MySQL query Browser โดยเป็นโปรแกรมที่สามารถใช้ได้ฟรี ดาวน์โหลดได้ที่ <http://dev.mysql.com/downloads/gui-tools/5.0.html>

1. เริ่มใช้งาน MySQL query Browser

คลิกที่ Start >> MySQL-JHCIS >> MySQL query Browser



1. กำหนดการติดต่อ MySQL Server

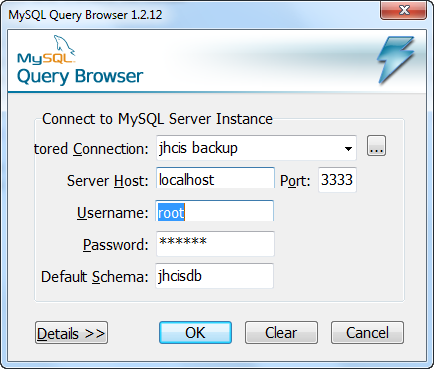
2.1 เลือกใช้ Stored Connection (หากยังไม่ดัสร้างไว้สามารถดูวิธีการสร้าง

Connection ได้จากเรื่องการใช้งาน MySQL Administrator )

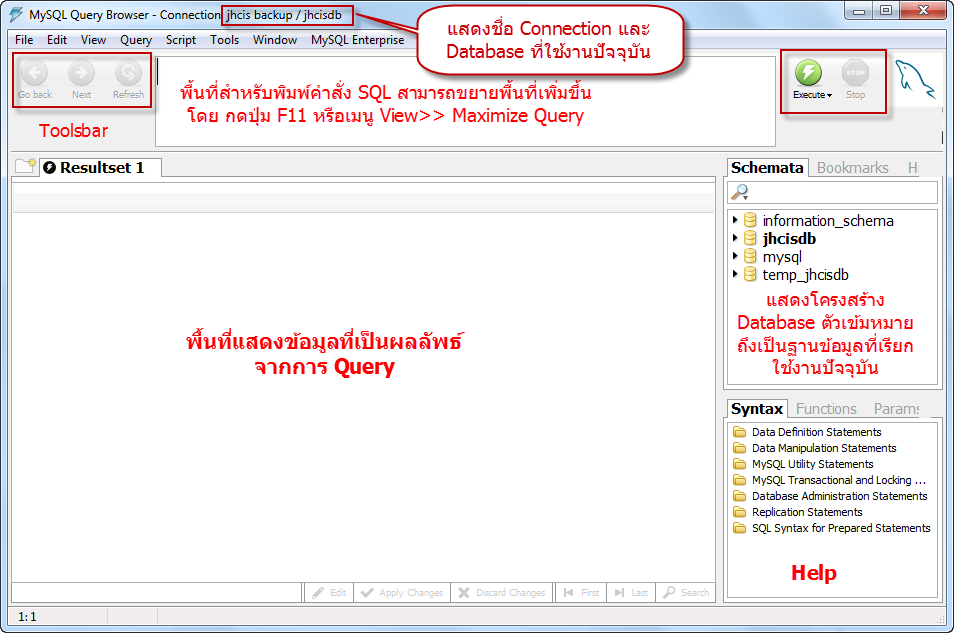
2.2 กำหนดค่าการติดต่อ MySQL Server

|  |  |
| --- | --- |
| Server Host | ชื่อ หรือ IP Address ของ MySQL Server ถ้าเป็นเครื่องเดียวกับ Server ใช้ชื่อ Localhost หรือ 127.0.0.1 |
| Port | Port Number ที่ใช้ สำหรับ JHCIS ใช้ port 3333 โดยปกติ MySQL ใช้ port 3306 |
| Username | ชื่อที่เข้าใช้งาน MySQL |
| Password | รหัสผ่านของ Username ที่เข้าใช้งาน MySQL |
| Default Schema | ชื่อฐานข้อมูลที่ต้องการใช้งาน เช่น 3333 |

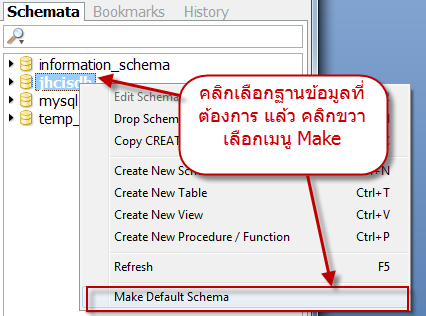
2.3 กดปุ่ม OK เพื่อเข้าใช้งาน



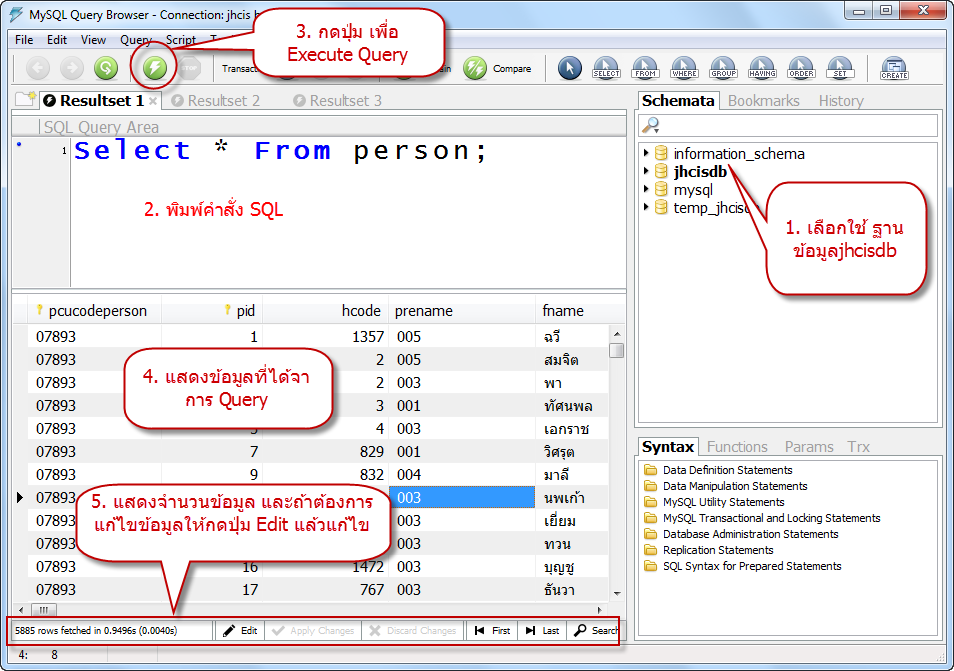
1. แสดงหน้าจอสำหรับใช้งานโปรแกรม มีส่วนประกอบต่างๆ ดังนี้



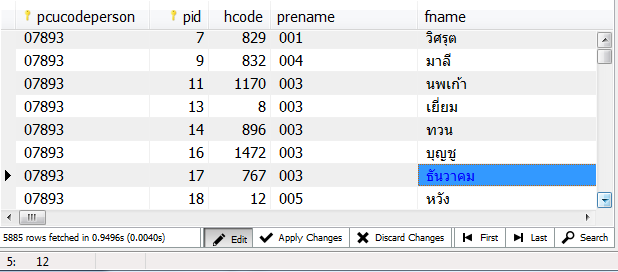
1. วิธีการเปลี่ยนฐานข้อมูล



1. การเขียนคำสั่ง SQL



1. แก้ไขข้อมูลในตาราง



|  |  |
| --- | --- |
|  | แก้ไขข้อมูลในตาราง |
|  | บันทึกการแก้ไข |
|  | ยกเลิกการแก้ไข |
| C:\Users\ning\AppData\Local\Temp\SNAGHTML4b202e.PNG | เลื่อนรายการข้อมูล |
|  | ค้นหาข้อมูลในตาราง |

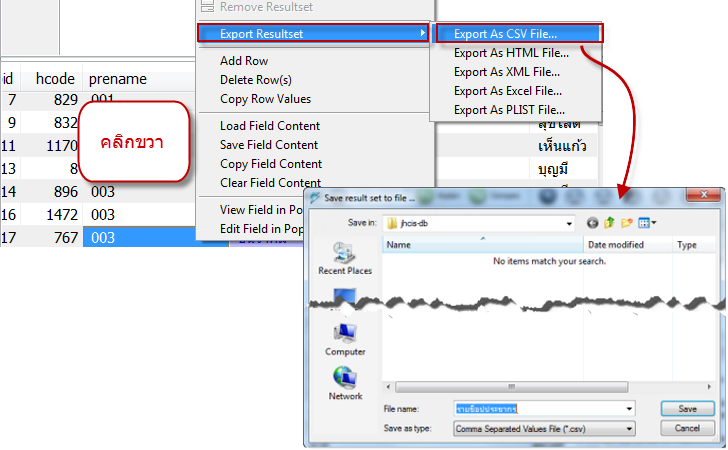
1. การส่งออกข้อมูลที่ได้จากการ Query ออกเป็นรูปแบบต่างๆโดย

7.1 คลิกขวาที่ตารางข้อมูล

7.2 เลือก Export Resultset

7.3 เลือกชนิดข้อมุลที่ต้องการส่งออก

7.4 กำหนดตำแหน่งที่จะบันทึก และชื่อไฟล์ แล้วกดปุ่ม Save



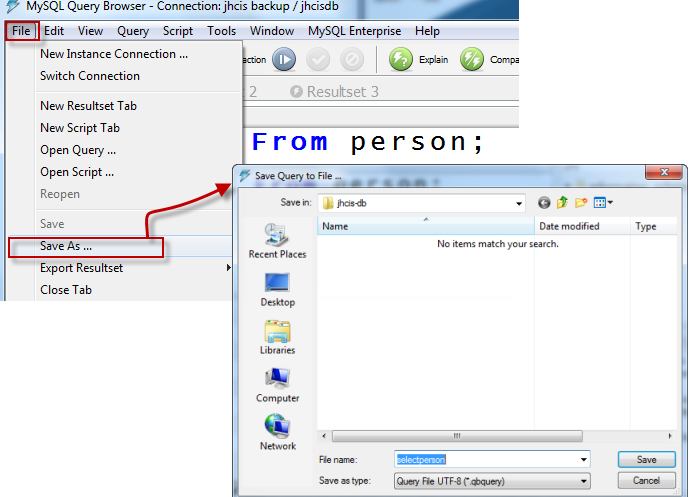
1. การบันทึกคำสั่ง SQL

8.1 เลือกเมนู File >> Save As…

8.2 กำหนดตำแหน่งที่จะบันทึก และชื่อไฟล์ แล้วกดปุ่ม Save

8.3 ชนิดของไฟล์ที่บันทึกโดยปกติจะเป็น qbquery สามารถกำหนดให้เป็นประเภทอื่น

ได้ เช่น .sql



1. การเรียกใช้งานคำสั่งที่มีอยู่

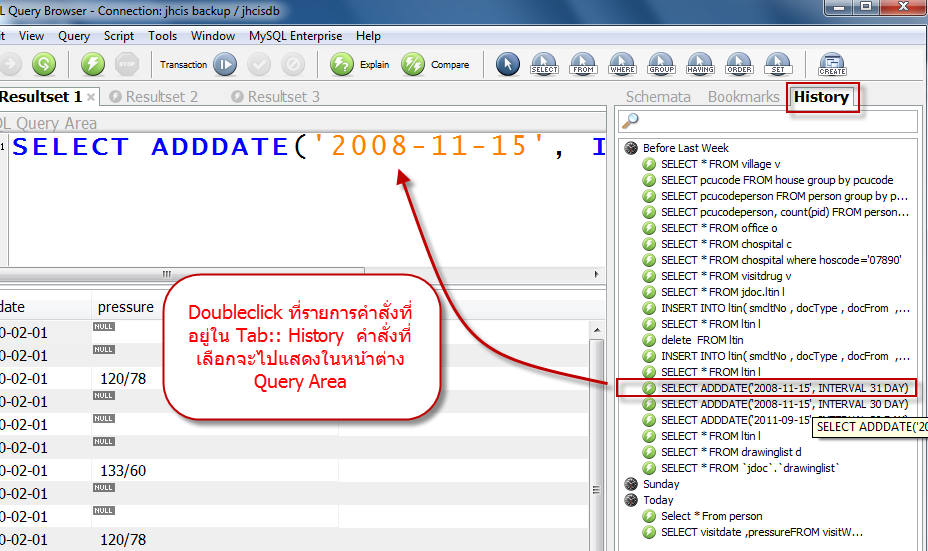
9.1 เลือกเมนู File >> Open Query สำหรับไฟล์ที่เป็น .qbquery

หรือ เลือกเมนู File >> Open Script สำหรับไฟล์ที่เป็น .sql

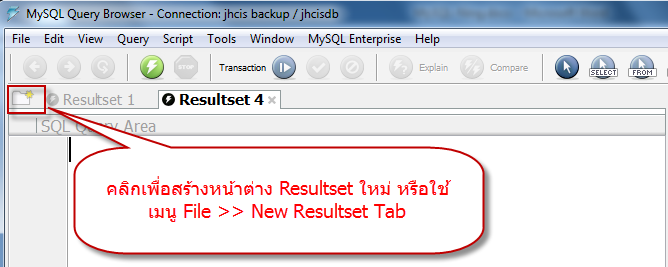
9.2 เลือกไฟล์จากตำแหน่งที่ต้องการ แล้วกดปุ่ม Open

9.3 สามารถแก้ไขสั่ง แล้วใช้งานได้

1. การเรียกใช้งานคำสั่งที่เคยใช้งาน ซึ่งโปรแกรมจะเก็บทุกคำสั่งที่เคยใช้งานไว้ที่ Tab History เราสามารถเรียกใช้งานคำสั่งที่อยู่ใน Tab History ได้



1. การสร้าง Resultset ใหม่เพื่อให้ใช้งานหลายๆหน้าต่าง



สารบัญ

|  |  |
| --- | --- |
| แนะนำระบบจัดการฐานข้อมูล | 1 |
| การติดตั้งและใช้งาน MySQL |  |
| ประเภทข้อมูลใน MySQL |  |
| Start and Stop MySQL Service |  |
| MySQL Command Line |  |
| Data Definition Language |  |
| Data Manipulation Language |  |
| Data Retrieval Language |  |
| การเรียกดูข้อมูลชั้นสูง |  |
| SQL Function |  |
| การสร้าง View |  |
| การกำหนดสิทธิในการใช้งานฐานข้อมูล |  |
| การสำรองข้อมูล |  |
| การนำข้อมูลสำรองใช้งาน |  |
| การเพิ่มประสิทธิภาพ และการซ่อมแซมตารางข้อมูล |  |
| การใช้งาน MySQL Administrator |  |
| การใช้งาน MySQL Query Browser |  |
| โครงสร้างฐานช้อมูล JHCIS |  |