ชนิดข้อมูลใน MySQL (Datatype)

VARCHAR : สำหรับเก็บข้อมูลประเภทตัวอักษร ทุกครั้งที่เลือกชนิดของฟิลด์เป็นประเภทนี้ จะต้องมี การกำหนดความยาวของข้อมูลลงไปด้วย ซึ่งสามารถกำหนดค่าได้ตั้งแต่ 1 - 255 ฟิลด์ชนิดนี้ เหมาะ
สำหรับการเก็บข้อมูลสั้นๆ เช่น ชื่อ นามสกุล หรือหัวข้อต่างๆ เป็นต้น... ในส่วนฟิลด์ประเภทนี้ จะ สามารถเลือก "แอตทริบิวต์" เป็น BINARY ได้ โดยปกติแล้วการจัดเรียงข้อมูลเวลาสืบค้น (query) สำหรับ VARCHAR จะเป็นแบบ case-sensitive (ตัวอักษรใหญ่ และเล็กมีความหมายแตกต่างกัน) แต่ หากระบุ "แอตทริบิวต์" เป็น BINARY ปุ๊บ การสืบค้นจะไม่คำนึงตัวอักษรว่าจะเป็นตัวใหญ่ หรือตัวเล็ก

CHAR : สำหรับเก็บข้อมูลประเภทตัวอักษร แบบที่ถูกจำกัดความกว้างเอาไว้คือ 255 ตัวอักษร ไม่ สามารถปรับเปลี่ยนได้ เหมือนกับ VARCHAR หากทำการสืบค้นโดยเรียงตามลำดับ ก็จะเรียงข้อมูล
แบบ case-sensitive เว้นแต่จะกำหนดแอตทริบิวต์เป็น BINARY ที่จะทำให้การเรียงข้อมูลเป็นแบบ non case-sensitive เช่นเดียวกับ VARCHAR

TINYTEXT : ในกรณีที่ข้อความยาวๆ หรือต้องการที่จะค้นหาข้อความ โดยอาศัยฟีเจอร์ FULL TEXT SEARCH ของ MySQL เราอาจจะเลือกที่จะไม่เก็บข้อมูลลงในฟิลด์ประเภท VARCHAR ที่มีข้อจำกัด
แค่ 256 ตัวอักษร แต่เราจะเก็บลงฟิลด์ประเภท TEXT แทน โดย TINYTEXT นี้ จะสามารถเก็บข้อมูล ได้ 256 ตัวอักษร ซึ่งมองเผินๆ ก็ไม่ต่างกับเก็บลงฟิลด์ประเภท CHAR หรือ VARCHAR(255) เลย แต่
จริงๆ มันต่างกันตรงที่ มันทำFULL TEXT SEARCH ได้

TEXT : สำหรับเก็บข้อมูลประเภทตัวอักษร เช่นเดียวกับ TINYTEXT แต่สามารถเก็บได้มากขึ้น โดย สูงสุดคือ 65,535 ตัวอักษร หรือ 64KB เหมาะสำหรับเก็บข้อมูลพวกเนื้อหาต่างๆ ที่ยาวๆ

MEDIUMTEXT : เก็บข้อมูลประเภทตัวอักษร เช่นเดียวกับ TINYTEXT แต่เก็บข้อมูลได้ 16,777,215 ตัวอักษร

LONGTEXT : เก็บข้อมูลประเภทตัวอักษร เช่นเดียวกับ TINYTEXT แต่เก็บข้อมูลได้ 4,294,967,295 ตัวอักษร

TINYINT : สำหรับเก็บข้อมูลชนิดตัวเลขที่มีขนาด 8 บิต ข้อมูลประเภทนี้เราสามารถกำหนดเพิ่มเติม ในส่วนของ "แอตทริบิวต์" ได้ว่าจะเลือกเป็น UNSIGNED หรือ UNSIGNED ZEROFILL โดยจะมี ความแตกต่างดังนี้

- UNSIGNED : จะหมายถึงเก็บค่าตัวเลขแบบไม่มีเครื่องหมาย แบบนี้จะทำให้สามารถเก็บค่าได้ ตั้งแต่ 0 - 255
- UNSIGNED ZEROFILL : เหมือนข้างต้น แต่ว่าหากข้อมูลที่กรอกเข้ามาไม่ครบตามจำนวน หลักที่เรากำหนด ตัว MySQL จะทำการเติม 0 ให้ครบหลักเอง เช่น ถ้ากำหนดให้ใส่ได้ 3 หลัก แล้วทำการเก็บข้อมูล 25 เข้าไป เวลาที่สืบค้นดู เราจะได้ค่าออกมาเป็น 025 หากไม่เลือก "แอ ตทริบิวต์" สิ่งที่เราจะได้ก็คือ SIGNED นั่นก็คือต้องเสียบิตนึงไปเก็บเครื่องหมายบวก/ลบ ทำ ให้สามารถเก็บข้อมูลได้อยู่ในช่วง -128 ถึง 127 เท่านั้น

SMALLINT : สำหรับเก็บข้อมูลประเภทตัวเลขที่มีขนาด 16 บิต จึงสามารถเก็บค่าได้ตั้งแต่ -32768 ถึง 32767 (ในกรณีแบบคิดเครื่องหมาย) หรือ 0 ถึง 65535 (ในกรณี UNSIGNED หรือไม่คิดเครื่องหมาย)
ซึ่งสามารถเลือก Attribute เป็น UNSIGNED และ UNSIGNED ZEROFILL ได้เช่นเดียวกับ TINYINT

MEDIUMINT : สำหรับเก็บข้อมูลประเภทตัวเลขที่มีขนาด 24 บิต นั่นก็หมายความว่าสามารถเก็บ ข้อมูลตัวเลขได้ตั้งแต่ -8388608 ไปจนถึง 8388607 (ในกรณีแบบคิดเครื่องหมาย) หรือ 0 ถึง 16777215
(ในกรณี ที่เป็น UNSIGNED หรือไม่คิดเครื่องหมาย) ซึ่งสามารถเลือก Attribute เป็น UNSIGNED และ UNSIGNED ZEROFILL ได้เช่นเดียวกับ TINYINT

INT : สำหรับเก็บข้อมูลประเภทตัวเลขที่มีขนาด 32 บิต หรือสามารถเก็บข้อมูลได้ตั้งแต่ -2147483648 ไปจนถึง 2147483647 ครับ (ในกรณีแบบคิดเครื่องหมาย) หรือ 0 ถึง 4294967295 (ในกรณีที่เป็น
UNSIGNED หรือไม่คิดเครื่องหมาย) ซึ่งสามารถเลือก Attribute เป็น UNSIGNED และ UNSIGNED ZEROFILL ได้เช่นเดียวกับ TINYINT

BIGINT : สำหรับ เก็บข้อมูลประเภทตัวเลขที่มีขนาด 64 บิต สามารถเก็บข้อมูลได้ตั้งแต่ - 9223372036854775808 ไปจนถึง 9223372036854775807 เลยทีเดียว (แบบคิดเครื่องหมาย) หรือ 0 ถึง 18446744073709551615 (ในกรณีที่เป็น UNSIGNED หรือไม่คิดเครื่องหมาย) ซึ่งสามารถเลือก Attribute เป็น UNSIGNED และ UNSIGNED ZEROFILL ได้เช่นเดียวกับ TINYINT

FLOAT[(M,D)] : ที่กล่าวถึงไปทั้งหมด ในตระกูล INT นั้นจะเป็นเลขจำนวนเต็ม หากเราบันทึกข้อมูล ที่มีเศษทศนิยม มันจะถูกปัดทันที ดังนั้นหากต้องการจะเก็บค่าที่เป็นเลขทศนิยม ต้องเลือกชนิดขอฟิลด์ เป็น FLOAT โดยจะเก็บข้อมูลแบบ 32 บิต คือมีค่าตั้งแต่ -3.402823466E+38 ไปจนถึง -1.175494351E- 38, 0 และ 1.175494351E-38 ถึง 3.402823466E+38

DOUBLE[(M,D)] : สำหรับเก็บข้อมูลประเภทตัวเลขทศนิยม เช่นเดียวกับ FLOAT แต่มีขนาดเป็น 64 บิต สามารถเก็บได้ตั้งแต่ -1.7976931348623157E+308 ถึง -2.2250738585072014E-308, 0 และ
2.2250738585072014E-308 ถึง 1.7976931348623157E+308

DECIMAL[(M,D)] : สำหรับเก็บข้อมูลประเภทตัวเลขทศนิยม เช่นเดียวกับ FLOAT แต่ใช้กับข้อมูลที่ ต้องการความละเอียดและถูกต้องของข้อมูลสูง ข้อสังเกต เกี่ยวกับข้อมูลประเภท FLOAT, DOUBLE และ DECIMAL ก็คือ เวลากำหนดความ ยาวของข้อมูลในฟิลด์ จะถูกกำหนดอยู่ในรูปแบบ (M,D) ซึ่งหมายความว่า ต้องมีการระบุว่า จะให้มี ตัวเลขส่วนที่เป็นจำนวนเต็มกี่หลัก และมีเลขทศนิยมกี่หลัก เช่น ถ้าเรากำหนดว่า FLOAT(5,2) จะ หมายความว่า เราจะเก็บข้อมูลเป็นตัวเลขจำนวนเต็ม 5 หลัก และทศนิยม 2 หลัก ดังนั้นหากทำการใส่ ข้อมูล 12345.6789 เข้าไป สิ่งที่จะเข้าไปอยู่ในข้อมูลจริงๆ ก็คือ 12345.68 (ปัดเศษให้มีจำนวนหลัก ตามที่กำหนดไว้)

DATE : สำหรับเก็บข้อมูลประเภทวันที่ โดยเก็บได้จาก 1 มกราคม ค.ศ. 1000 ถึง 31 ธันวาคม ค.ศ. 9999 โดยจะแสดงผลในรูปแบบ YYYY-MM-DD

DATETIME : สำหรับเก็บข้อมูลประเภทวันที่ และเวลา โดยจะเก็บได้ตั้งแต่ 1 มกราคม ค.ศ. 1000 เวลา 00:00:00 ไปจนถึง 31 ธันวาคม ค.ศ. 9999 เวลา 23:59:59 โดยรูปแบบการแสดงผล เวลาที่ทำการสืบค้น
(query) ออกมา จะเป็น YYYY-MM-DD HH:MM:SS

TIMESTAMP[(M)] : สำหรับเก็บข้อมูลประเภทวันที่ และเวลาเช่นกัน แต่จะเก็บในรูปแบบของ YYYYMMDDHHMMSS หรือ YMMDDHHMMSS หรือ YYYYMMDD หรือ YYMMDD แล้วแต่ ว่าจะระบุค่า M เป็น 14, 12, 8 หรือ 6 ตามลำดับ สามารถเก็บได้ตั้งแต่วันที่ 1 มกราคม ค.ศ. 1000 ไป จนถึงประมาณปี ค.ศ. 2037

TIME : สำหรับเก็บข้อมูลประเภทเวลา มีค่าได้ตั้งแต่ -838:59:59 ไปจนถึง 838:59:59 โดยจะแสดงผล ออกมาในรูปแบบ HH:MM:SS YEAR[(2/4)] : สำหรับเก็บข้อมูลประเภทปี ในรูปแบบ YYYY หรือ YY แล้วแต่ว่าจะเลือก 2 หรือ 4 (หากไม่ระบุ จะถือว่าเป็น 4 หลัก) โดยหากเลือกเป็น 4 หลัก จะเก็บค่าได้ตั้งแต่ ค.ศ. 1901 ถึง 2155 แต่ หากเป็น 2 หลัก จะเก็บตั้งแต่ ค.ศ. 1970 ถึง 2069 ข้อสังเกต ค่าที่เก็บในข้อมูลประเภท TIMESTAMP และ YEAR นั้นจะมีความสามารถพอๆ กับ การเก็บข้อมูลวันเดือนปี และเวลา ด้วยฟิลด์ชนิด VARCHAR แต่ต่างกันตรงที่ จะใช้เนื้อที่เก็บข้อมูล น้อยกว่า... ทว่า ฟิลด์ประเภท TIMESTAMP นั้นจะมีข้อจำกัดในเรื่องของเวลาที่สามารถเก็บได้ คือ จะต้องอยู่ในระหว่าง 1 มกราคม ค.ศ. 1000 ไปจนถึงแถวๆ ค.ศ. 2037 อย่างที่บอก แต่หากเก็บเป็น VARCHAR นั้นจะไม่ติดข้อจำกัดนี้ ฟิลด์ชนิด YEAR ก็เช่นกันครับ... ใช้เนื้อที่แค่ 1 ไบต์เท่านั้นในการ เก็บข้อมูล แต่ข้อจำกัดจะอยู่ที่ ปี ค.ศ. 1901 ถึง 2155 เท่านั้น (หรือ ค.ศ. 1970 ถึง 2069 ในกรณี 2 หลัก) แต่หากเก็บเป็น VARCHAR จะได้ตั้งแต่ 0000 ถึง 9999 เลย อันนี้เลยอยู่ที่ความจำเป็นมากกว่าครับ (แต่ ด้วยความที่ว่าปัจจุบันฮาร์ดดิสก์ราคาถูกมากๆ ผมเลยไม่ติดใจอะไรที่จะใช้ VARCHAR แทน เพื่อ ความสบายใจ อิอิ เพราะสมมติว่ากินเนื้อที่ต่างกัน 3 ไบต์ ต่อ 1 ระเบียน มีข้อมูล 4 ล้านระเบียน ก็เพิ่ง ต่างกัน 12 ล้านไบต์ หรือ 12 เมกะไบต์เท่านั้นเอง ซึ่งหากเทียบกับปริมาณข้อมูลทั้งหมดของข้อมูล 4 ล้านระเบียน ผมว่ามันต้องมีอย่างน้อยเป็นกิกะไบต์ ดังนั้นความแตกต่างที่ไม่กี่เมกะไบต์จึงไม่มากมาย)

TINYBLOB : สำหรับเก็บข้อมูลประเภทไบนารี ได้แก่ ไฟล์ข้อมูลต่างๆ, ไฟล์รูปภาพ, ไฟล์มัลติมีเดีย เป็นต้น คือไฟล์อะไรก็ตามที่อัพโหลดผ่านฟอร์มอัพโหลดไฟล์ในภาษา HTML โดย TINYBLOB นั้น
จะมีเนื้อที่ให้เก็บข้อมูลได้ 256 ไบต์

BLOB : สำหรับเก็บข้อมูลประเภทไบนารี เช่นเดียวกับ TINYBLOB แต่สามารถเก็บข้อมูลได้ 64KB

MEDIUMBLOB : สำหรับเก็บข้อมูลประเภทไบนารี เช่นเดียวกับ TINYBLOB แต่เก็บข้อมูลได้ 16MB

LONGBLOB : สำหรับเก็บข้อมูลประเภทไบนารี เช่นเดียวกับ TINYBLOB แต่เก็บข้อมูลได้ 4GB ข้อสังเกต ข้อมูลประเภท BLOB นั้น แม้จะมีประโยชน์ในเรื่องของการเก็บข้อมูลประเภท BINARY ให้อยู่กับตัวฐานข้อมูล ทำให้สะดวกเวลาสืบค้นก็ตาม แต่มันก็ทำให้ฐานข้อมูลมีขนาดใหญ่ เกินความจำเป็นด้วย ทำให้เกิดความไม่สะดวกในการสำรองฐานข้อมูลในกรณีที่มีข้อมูลอัพโหลดไป เก็บมากๆ โดยปกติแล้ว จะใช้วิธีการอัพโหลดไปเก็บไว้ในโฟลเดอร์ แล้วเก็บลิงก์ไปยังไฟล์เหล่านั้นเป็นฟิลด์ชนิด VARCHAR มากกว่า

SET : สำหรับเก็บข้อมูลที่เป็นกลุ่มของข้อมูลที่ยอมให้เลือกได้ 1 ค่าหรือหลายๆ ค่า ซึ่งสามารถกำหนด ได้ถึง 64 ค่า