Chapter 8

How to work with data types

Objectives

Applied

- Code queries that convert data from one data type to another.

Knowledge

- Describe the data that can be stored in any of the character, numeric, date/time, and large object data types.
- Describe ENUM and SET data types.

MySQL data type categories

- Character
- Numeric
- Date and time
- Large Object (LOB)
- Spatial

The character types

Type	Bytes
CHAR(M)	Mx3
VARCHAR(M)	L+1

How the character types work

Data type	Original value	Value stored	Bytes used
CHAR(2)	'CA'	'CA'	6
CHAR(10)	'CA'	'CA '	30
VARCHAR(10)	'CA'	'CA'	3
VARCHAR(20)	'California'	'California'	11
VARCHAR(20)	'New York'	'New York'	9
VARCHAR(20)	"Murach's MySQL"	"Murach's MySQL"	15

Terms to know

- Latin1 character set
- UTF-8 character set
- Unicode standard

The integer types

Type	Bytes
BIGINT	8
INT	4
MEDIUMINT	3
SMALLINT	2
TINYINT	1

How the UNSIGNED and ZEROFILL attributes work

Data type	Original value	Value stored	Value displayed
INT	99	99	99
INT	-99	-99	-99
INT UNSIGNED	99	99	99
INT UNSIGNED	-99	ERROR	ERROR
INT ZEROFILL	99	99	0000000099
INT(4) ZEROFILL	99	99	0099

The fixed-point type

Type	Bytes
`DECIMAL(M, D)`	Vary

The floating-point types

Type	Bytes
`DOUBLE`	8
`FLOAT`	4

How the fixed-point and floating-point types work

Data type	Original value	Value stored	Bytes used
`DECIMAL(9,2)`	`1.2`	`1.20`	5
`DECIMAL(9,2)`	`1234567.89`	`1234567.89`	5
`DECIMAL(9,2)`	`-1234567.89`	`-1234567.89`	5
`DECIMAL(18,9)`	`1234567.89`	`1234567.890000000`	8
`DOUBLE`	`1234567.89`	`1234567.89`	8
`FLOAT`	`1234567.89`	`1234570`	4

Terms to know

- Real number
- Precision
- Scale
- Exact numeric type
- Floating-point number
- Approximate numeric type

The date and time types

Type	Bytes
`DATE`	3
`TIME`	3
`DATETIME`	8
`TIMESTAMP`	4
`YEAR[(2|4)]`	1

How MySQL interprets date/time literals

Literal value	Value stored in DATE column
'2014-08-15'	2014-08-15
'2014-8-15'	2014-08-15
'14-8-15'	2014-08-15
'20140815'	2014-08-15
20140815	2014-08-15
'2014.08.15'	2014-08-15
'14/8/15'	2014-08-15
'8/15/14'	ERROR
'2014-02-31'	ERROR

How MySQL interprets date/time literals (continued)

Literal value	Value stored in TIME column
`'7:32'`	`07:32:00`
`'19:32:11'`	`19:32:11`
`'193211'`	`19:32:11`
`193211`	`19:32:11`
`'19:61:11'`	`ERROR`

Literal value	Value stored in DATETIME or TIMESTAMP column
`'2014-08-15 19:32:11'`	`2014-08-15 19:32:11`
`'2014-08-15'`	`2014-08-15 00:00:00`

The ENUM and SET types

Type	Bytes
`ENUM`	1-2
`SET`	1-8

How values are stored in ENUM columns

Value	Stored in column ENUM ('Yes', 'No', 'Maybe')
`'Yes'`	`'Yes'`
`'No'`	`'No'`
`'Maybe'`	`'Maybe'`
`'Possibly'`	`''`

How values are stored in SET columns

Value	Stored in column SET ('Pepperoni', 'Mushrooms', 'Olives')
`'Pepperoni'`	`'Pepperoni'`
`'Mushrooms'`	`'Mushrooms'`
`'Pepperoni, Bacon'`	`'Pepperoni'`
`'Olives, Pepperoni'`	`'Pepperoni, Olives'`

The large object types

Type	Bytes
LONGBLOB	L+4
MEDIUMBLOB	L+3
BLOB	L+2
TINYBLOB	L+1
LARGETEXT	L+4
MEDIUMTEXT	L+3
TEXT	L+2
TINYTEXT	L+1

Terms to know

- BLOB (binary large object) types
- CLOB (character large object) types

Implicitly convert data a number to a string

```
SELECT invoice_total, CONCAT('$', invoice_total)
FROM invoices
```

invoice_total	CONCAT('$', invoice_total)
3813.33	$3813.33
40.20	$40.20

Implicitly convert a string to a number

```
SELECT invoice_number, 989319/invoice_number
FROM invoices
```

invoice_number	989319/invoice_number
989319-457	1
263253241	0.0037580505988908225
963253234	0.0010270601385803393

Implicitly convert a date to a number

```
SELECT invoice_date, invoice_date + 1
FROM invoices
```

invoice_date	invoice_date + 1
2014-04-08	20140409
2014-04-10	20140411

The syntax of the CAST function

```
CAST(expression AS cast_type)
```

The syntax of the CONVERT function

```
CONVERT(expression, cast_type)
```

Cast types you can use in these functions

```
CHAR[(N)]
DATE
DATETIME
TIME
SIGNED [INTEGER]
UNSIGNED [INTEGER]
DECIMAL[(M[,D])]
```

A statement that uses the CAST function

```
SELECT invoice_id, invoice_date, invoice_total,
    CAST(invoice_date AS CHAR(10)) AS char_date,
    CAST(invoice_total AS SIGNED) AS integer_total
FROM invoices
```

invoice_id	invoice_date	invoice_total	char_date	integer_total
1	2014-04-08	3813.33	2014-04-08	3813
2	2014-04-10	40.20	2014-04-10	40
3	2014-04-13	138.75	2014-04-13	139

A statement that uses the CONVERT function

```
SELECT invoice_id, invoice_date, invoice_total,
    CONVERT(invoice_date, CHAR(10)) AS char_date,
    CONVERT(invoice_total, SIGNED) AS integer_total
FROM invoices
```

invoice_id	invoice_date	invoice_total	char_date	integer_total
1	2014-04-08	3813.33	2014-04-08	3813
2	2014-04-10	40.20	2014-04-10	40
3	2014-04-13	138.75	2014-04-13	139

The FORMAT and CHAR functions

```
FORMAT(number,decimal)
CHAR(value1[,value2]...)
```

FORMAT function examples

Function	Result
`FORMAT(1234567.8901,2)`	`1,234,567.89`
`FORMAT(1234.56,4)`	`1,234.5600`
`FORMAT(1234.56,0)`	`1,235`

CHAR function examples for common control characters

Function	Control character
`CHAR(9)`	Tab
`CHAR(10)`	Line feed
`CHAR(13)`	Carriage return

A statement that uses the CHAR function

```
SELECT CONCAT(vendor_name, CHAR(13,10),
vendor_address1, CHAR(13,10),
       vendor_city, ', ', vendor_state, ' ',
vendor_zip_code)
FROM vendors
WHERE vendor_id = 1;
```

```
US Postal Service
Attn:  Supt. Window Services
Madison, WI 53707
```