



The lecture will start soon at
08:30



Aalto University
School of Business

MySQL for Data Analytics

Lecturer: Yong Liu

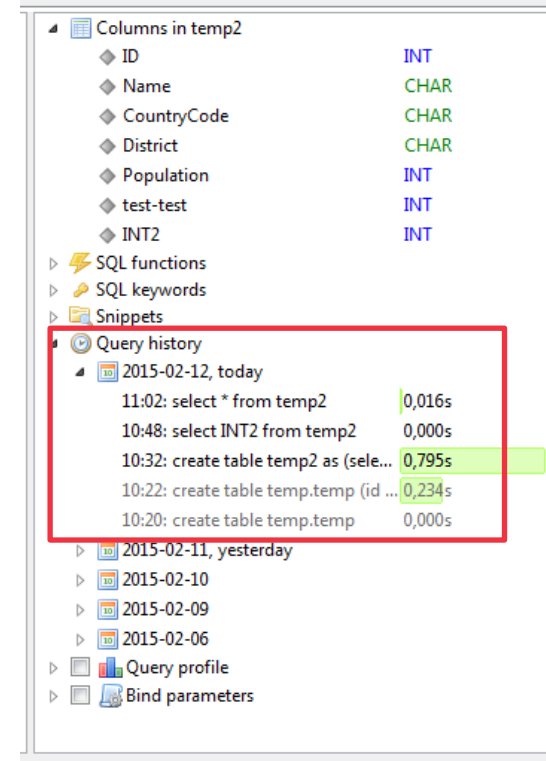
Contact me at: Yong.liu@aalto.fi

Tips

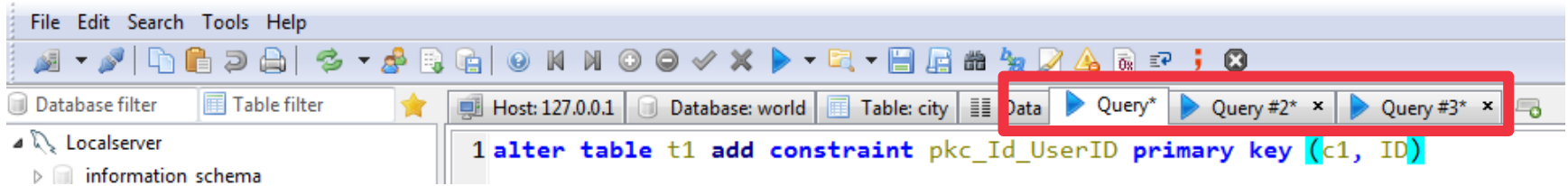
- **MySQL command is NOT case-sensitive.**
- **The names of database, table, columns are not case-sensitive in Windows, but case-sensitive in most varieties of Unix.**

Tips for HeidiSQL (1)

- HeidiSQL provides “Query history”.
- Remember to use this function!



Tips for HeidiSQL(2)



- In HeidiSQL, you can have several query-windows!

Content

- **Difference between remote and local server**
- **Exporting data to be a .csv file (Tips)**
- **Writing comments (annotations) for a query**
- **Create, drop and change (default) database**
- **Create and drop table in a database**
- **Understanding data types**

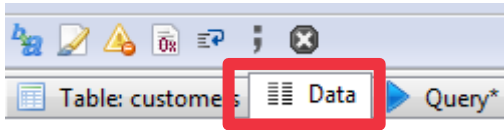
Section 1

Remote server	Local server
Main workstation—a supercomputer (Hostname : Johnson.org.aalto.fi) <ul style="list-style-type: none">➤ Your data is normally hosted and operated by a supercomputer➤ The speed of data manipulation could be very fast	Main workstation—your own laptop (Hostname: 127.0.0.1) <ul style="list-style-type: none">➤ Your data is hosted and operated locally.➤ The speed of data manipulation is determined by the capability of your own laptop
Network is required! <ul style="list-style-type: none">➤ Can only be used in student network in our case	Network is not necessary! <ul style="list-style-type: none">➤ Can be used everywhere with your own laptop
Permission <ul style="list-style-type: none">➤ Often restricted user right➤ E.g., cannot create database	Permission <ul style="list-style-type: none">➤ Full user right
Security of data <ul style="list-style-type: none">➤ Your data is secure even if your own computer collapses.	Security of data <ul style="list-style-type: none">➤ You may lose your data if your system collapses.

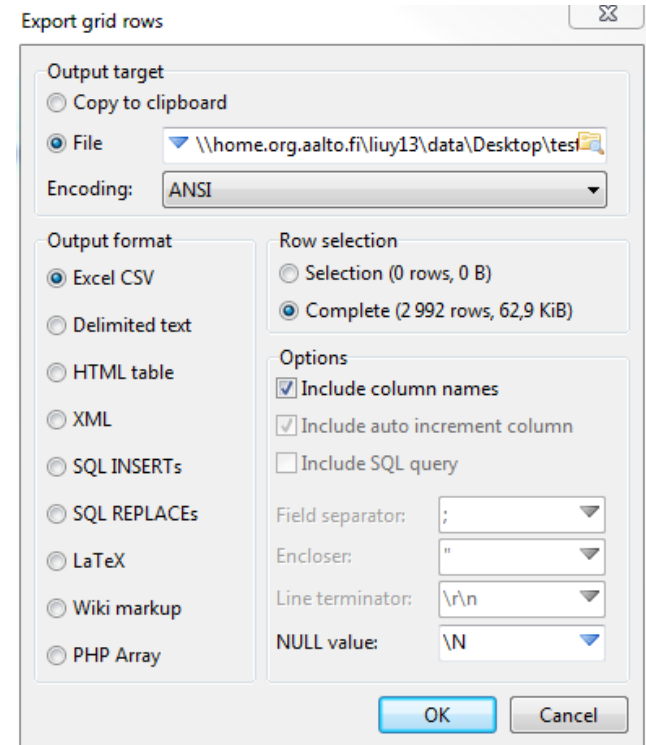
Section 2: export data to be a csv file

- Select the table that you want to export.

- Activate the data windows



- Tool → export grid rows



HeidiSQL Tips (1): showing more records

Host: 127.0.0.1 Database: classicmodels Table: orderdetails Data Query*

classicmodels.orderdetails: 2 996 rows total (approximately), limited to 1 000

orderNumber	productCode	quantityOrdered	priceEach	orderLineNumber
10 100	S18_1749	30	136	3
10 100	S18_2248	50	55.00	3
10 100	S18_4409	10	10.00	3
10 100	S24_3969	10	10.00	3
10 101	S18_2325	10	10.00	3
10 101	S18_2795	10	10.00	3
10 101	S24_1937	10	10.00	3
10 101	S24_2022	10	10.00	3
10 102	S18_1342	10	10.00	3
10 102	S18_1367	10	10.00	3

Export grid rows

Output target

☐ Copy to clipboard

☒ File

Encoding: ANSI

Output format

☒ Excel CSV

☐ Delimited text

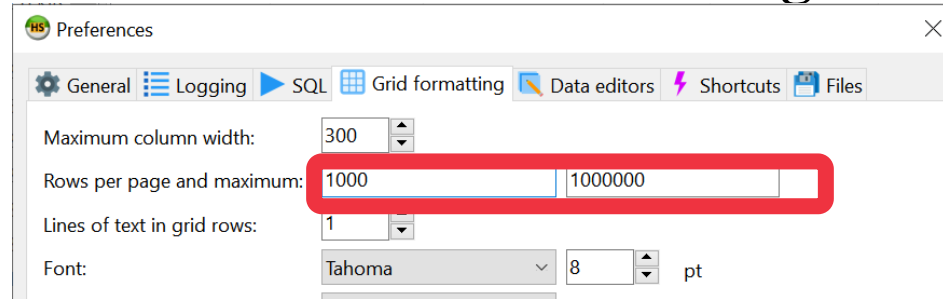
Row selection

☐ Selection (1 rows, 19 B)

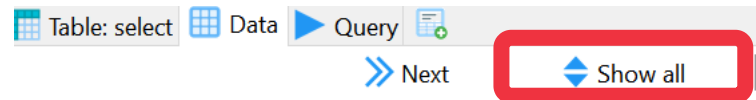
☒ Complete (1 000 rows, 21,0 KiB)

HeidiSQL Tips (2): showing more records

- **Solution 1:**
 - **Tools → Preferences → Grid formatting**



- **Solution 2:**
 - **Fast key: ctrl + End**



Section 3: Adding comments

```
1 -- -----
2 -- Host:                127.0.0.1
3 -- Server version:      5.6.23-log - MySQL Community Server (GPL)
4 -- Server OS:           Win32
5 -- HeidiSQL Version:    9.1.0.4904
6 -- -----
7
8 /*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
9 /*!40101 SET NAMES utf8mb4 */;
10 /*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0 */;
11 /*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
12
13 -- Dumping structure for table temp.chile
14 CREATE TABLE IF NOT EXISTS `chile` (
```

Adding comments: methods

- “**#**” [hashtag] character to the end of the line.
- “**--**” sequence to the end of the line.
Must be followed by at least one whitespace or control character (such as a space, tab, newline, and so on).
- **/*** sequence to the following ***/** sequence.
Extend a comment to over multiple lines

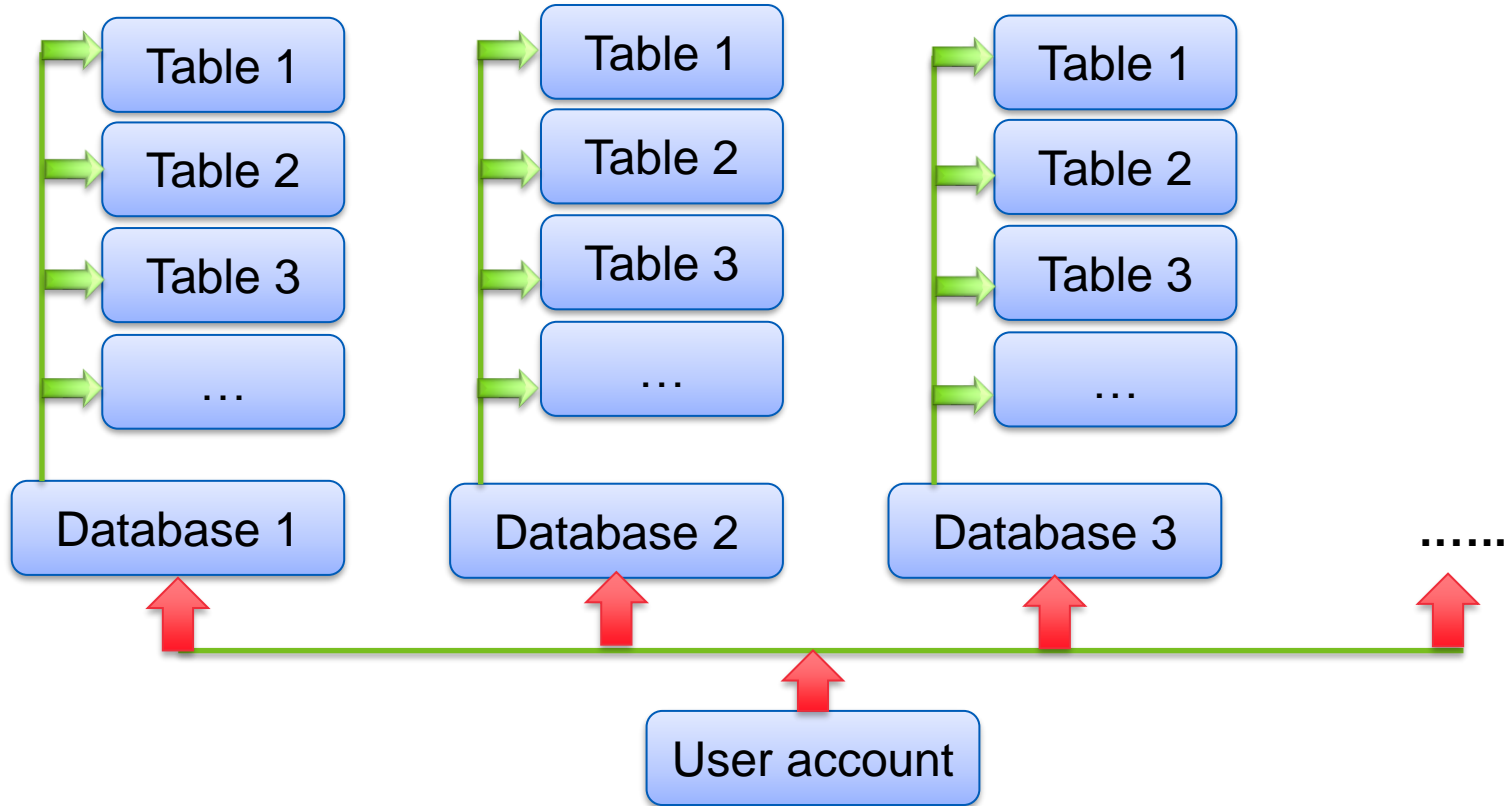
Examples of adding comments

```
mysql> SELECT 1+1;      # This comment continues to the end of line
mysql> SELECT 1+1;      -- This comment continues to the end of line
mysql> SELECT 1 /* this is an in-line comment */ + 1;
mysql> SELECT 1+
/*
this is a
multiple-line comment
*/
1;
```

A semicolon ends a command.

However, in HeidiSQL, if you run **only one command**, you don't have to type semicolon.

Section 4: Managing database

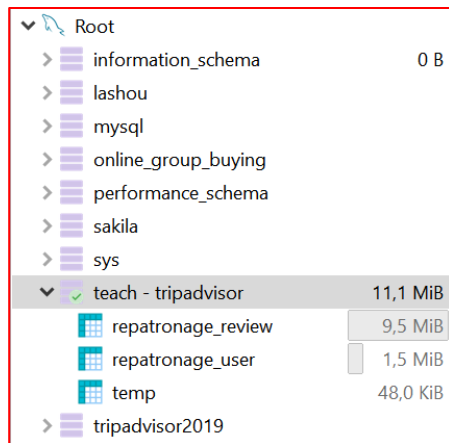


Tips

- Typically, you should use the same database to host all relevant data (tables) belonging to the same research project.

Don't distribute your tables that belong to the same research project across different databases.

- Click to activate a database before operating any command on tables belonging to that database.



Different databases

Only with Root account

– full admin rights

- **Creating a database**
 - Create database ``for_example``
- **Removing a database**
 - Drop database ``for_example``
- **Change default database**
 - use ``for_example``
- All these operations can be done via clicking on HeidiSQL's interface.

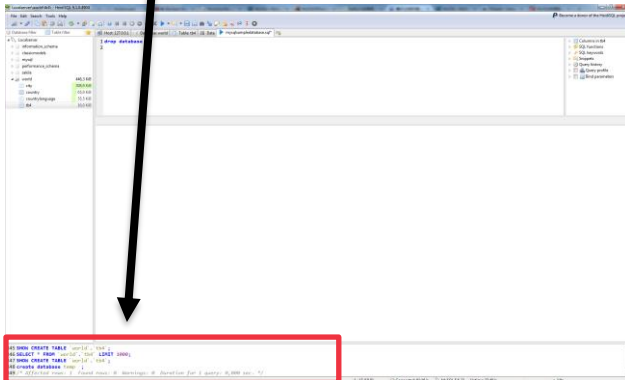
Statistics of running MySQL commands

```
348 create database temp ;
```

Command to create
a database named
as 'temp'

```
349 /* Affected rows: 1 Found rows: 0 Warnings: 0 Duration for 1 query: 0,000 sec. */
```

No error message: command OK



Report: identifying error

```
350 drop database xx ;
```

```
351 /* SQL Error (1008): Can't drop database 'xx'; database doesn't exist */
```

```
352 /* Affected rows: 0 Found rows: 0 Warnings: 0 Duration for 0 of 1 quer
```

What is the error?



Tips: Always briefly browse 'report' after running a command

Section 5: Create a table

- You can create a table via clicking the buttons of HeidiSQL.
- Alternatively, you could use commands: E.g.


```
create table TableName (Variable1 datatype [constraint],  
                        Variable2 datatype [constraint],  
                        Variable3 datatype [constraint],  
                        .....  
                        );
```



Required field

Example

We will talk about
“primary key” later.



```
create table tb4 (  
    id          int(10)    primary key,  
    name        varchar(20)  
);
```

```
create table TableName (  
    Variable1    datatype [constraint],  
    Variable2    datatype [constraint],  
    Variable3    datatype [constraint],  
    .....  
);
```

New MySQL vocabulary: “create”, “table”, “int”, “varchar”, “primary key”

Drop table

- Which one do you think is the correct command to remove a table?
 - a) **Remove table** tableName;
 - b) **Delete table** tableName;
 - c) **Drop table** tableName;
 - d) **Drop** tableName;

This one does not work!



Please use Presemo to provide your answer at: **presemo.aalto.fi/drm**

Drop table

- Command to drop a table
 - **drop table** TableName;
- If you want drop multiple tables in one command

- **drop table** TableName1, TableName2
TableName3;

Use comma in a command to describe 'equal' subjects.

Restrictions on table and column names

- 1. The names cannot exceed 18 characters.**
- 2. The names must start with a letter.**
- 3. The names can contain letters, numbers, and underscores(_)**
- 4. The names cannot contain spaces.**

Tips: Don't use reserved words as table or column name

AND	BEFORE	BETWEEN
BY	CALL	CASE
CHANGE	CHAR	CHARACTER
COLUMN	CURRENT_DATE	CURRENT_TIME
CURRENT_TIMESTAMP	CURRENT_USER	DATABASES
DEFAULT	DELETE	DESC
DESCRIBE	DISTINCT	FOREIGN
FROM	FULLTEXT	INDEX
INSERT	INTERVAL	KEY
KEYS	LIKE	LIMIT
LONG	MATCH	NOT
OPTION	READ	REPEAT
REQUIRE	RETURN	TABLE
TO	USER	UTC_TIME



See the full list of reserved words at: <http://dev.mysql.com/doc/mysql-d-version-reference/en/mysql-d-version-reference-reservedwords-5-7.html>

Tips

- **‘Date’ and ‘Year’ are often used in different data files as variables names, but they are reserved words in MySQL.**
- **Handling reserved words by HeidiSQL.**

If a reserved word is used as column name

- For instance, a column is named as **select**.

```
SELECT `select` from table_name
```

Please note that it is ` (back quote). It is not ' !

Where you can find



Section 6: Data types

- Consistency
- Validation
- Compactness
- Performance

```
create table TableName (Variable1 datatype [constraint],  
                        Variable2 datatype [constraint],  
                        Variable3 datatype [constraint],  
                        .....  
                        );
```

MySQL data types

- **Numeric Types**
- **Date and Time Types**
- **String Types**
- **Spatial Data Types (Not covered in the course)**

Numeric types

- **Integer types (Exact value)**
 - E.g. **Int**
- **Fix-point types (Exact value)**
 - **Decimal**
- **Floating-point (Approximate value)**
 - E.g. **Float**

Unsigned vs. signed

```
Create Table xx (  
    variable1 INT UNSIGNED,  
    variable2 INT);
```

New MySQL vocabulary: “**signed**”, “**unsignd**”.

Numeric types: Integer Types

Type	Storage (Bytes)	Value range (Unsigned)	Value range (Signed)
TINYINT	1	0~255	-128~127
SMALLINT	2	0~65,535	-32,768~32,767
MEDIUMINT	3	0~16,777,215	-8,388,608~8,388,607
INT	4	0~4,294,967,295	-2,147,483,648~2,147,483,647
BIGINT	8	0~18446744073709551615	-9223372036854775808~9223372036854775807

‘Signed’ is a default setting.

Questions (1)

- Which numeric type is good for a variable representing the number of dates in a month (e.g., the number of working days), or the number of the dates of a year?

TinyINT for the dates of a month

SmallINT for the dates of a year

Type	Storage (Bytes)	Value range (Unsigned)	Value range (Signed)
TINYINT	1	0~255	-128~127
SMALLINT	2	0~65535	-32768~32767
MEDIUMINT	3	0~16777215	-8388608~8388607
INT	4	0~4294967295	-2147483648~2147483647
BIGINT	8	0~18446744073709551615	-9223372036854775808~9223372036854775807

Questions (2)

- Do you think it makes sense to set up the type of “Annual Salary” for individual employees to be ‘smallint’? **No**

Type	Storage (Bytes)	Value range (Unsigned)	Value range (Signed)
SMALLINT	2	0~65535	-32768~32767
MEDIUMINT	3	0~16777215	-8388608~8388607



- **Does field size affect query time?**
 - **The short answer is yes!**
- **Do we need to have precisely-sized variables?**
 - **It depends on context!**

Numeric types: Exact Value

- The “**Decimal (M, D)**” stores exact numeric data values.
- **M** is the maximum number of digits. It has a range of 1 to 65.
- **D** is the number of digits to the right of the decimal point (the scale). It has a range of 0 to 30 and must be no larger than M.

For instance:


- **Decimal(5,2)** stores any value with five digits and two decimals, so values that can be stored in the column range from **-999.99 to 999.99**.
- **Decimal(5,2) = Dec(5,2)**

MySQL Grammar -- Insert

- **Insert Into table_name**

(column1, column2, column3,...)

Values (value1, value2, value3,...)



Specifying which
columns you want to
insert values

Example

- `CREATE TABLE IF NOT EXISTS `decimal_test` (salary
DECIMAL(5,3)) ;`
- `INSERT INTO `decimal_test` (salary) VALUES (1.2345) ;`
- `INSERT INTO `decimal_test` (salary) VALUES (1.2) ;`
- `INSERT INTO `decimal_test` (salary) VALUES (12.345) ;`
- `INSERT INTO `decimal_test` (salary) VALUES (123.45) ;`

- `CREATE TABLE `decimal_test` (salary DECIMAL(5,3));`
- `INSERT INTO `decimal_test` (salary) VALUES (1.2345);`
- `INSERT INTO `decimal_test` (salary) VALUES (1.2);`
- `INSERT INTO `decimal_test` (salary) VALUES (12.345);`
- `INSERT INTO `decimal_test` (salary) VALUES (123.45);`

salary
1,235
1,200
12,345

/ SQL Error (1264): Out of range value for column 'salary' at row 1 */*

`CREATE TABLE `decimal_test` (salary DECIMAL);`

123.45 cannot be inserted

Default settings for Decimal is decimal(10,0)

Insert by typing in HeidiSQL

The screenshot shows the HeidiSQL interface. On the left, the 'Database filter' pane shows the 'classicmodels' database selected, with the 'offices' table highlighted. The main pane displays the 'Data' view of the 'offices' table, showing 7 rows. A right-click context menu is open over the data, with the 'Insert row' option highlighted. The menu also includes options like 'Copy', 'Paste', 'Duplicate row', 'Post', 'Cancel editing', and 'Delete selected row(s)'. The 'Data' tab in the top toolbar is highlighted with a red box.

officeCode	city	phone	addressLin
1	San Francisco	+1 650 219 4782	100 Marke
2	Boston	+1 215 837 0825	1550 Cour
3	NYC	+1 212 555 3000	523 East 5
4	Paris	+33 14 723 4404	43 Rue Jo
5	Tokyo	+81 33 224 5000	4-1 Kioich
6	Sydney	+61 2 9264 2451	5-11 Went
7	London	+44 20 7877 2041	25 Old Bro

1. Select "data" view
2. Right click on data
3. Select "insert row"

Numeric types: Floating-Point (Approximate Value)

Types	Storage	Negative value	Positive value
FLOAT	4	-3.402823466E+38～ -1.175494351E-38	0 and 1.175494351E-38～ 3.402823466E+38
DOUBLE	8	-1.7976931348623157E+308～ -2.2250738585072014E-308	0 and 2.2250738585072014E-308～ 1.7976931348623157E+308

Decimal requires more storage space, such as 9 bytes for Decimal(19,9).

<https://www.mysqltutorial.org/mysql-decimal/>

<https://dev.mysql.com/doc/refman/8.0/en/storage-requirements.html>

<http://stackoverflow.com/questions/6949673/mysql-min-max-for-double-type>

Comparing decimal and float

```
create table t1 (c1 float(10,2), c3 decimal(10,2));
```

```
insert into t1 values (9876543.12, 9876543.12);
```

```
insert into t1 values (1234567.23, 1234567.23);
```

c1	c3
9 876 543,00	9 876 543,12
1 234 567,25	1 234 567,23

Difference between decimal, float and double



For money, **always** decimal. It's why it was created.

493

If numbers must add up correctly or balance, use decimal. This includes any financial storage or calculations, scores, or other numbers that people might do by hand.



If the exact value of numbers is not important, use double for speed. This includes graphics, physics or other physical sciences computations where there is already a "number of significant digits".

- **<http://stackoverflow.com/questions/1165761/decimal-vs-double-which-one-should-i-use-and-when/1165788#1165788>**

When you need to compare values, use decimal!

MySQL INT(1) or INT(10)

- **An unsigned int has the max value of 4,294,967,295 no matter if its int (1) or int(10) and will use 4 bytes of data.**

Date and Time Types

- **Year**
- **Date ('YYYY-MM-DD')**
- **Datetime ('YYYY-MM-DD HH:MM:SS')**

Range: '1000-01-01 00:00:00' to '9999-12-31 23:59:59'.

- **Timestamp**

Range: '1970-01-01 00:00:01' to '2038-01-19 03:14:07' UTC.

UTC: Coordinated Universal Time

**It is very likely to reach its lower limit when TIMESTAMP is used --
e.g. storing birthdate.**

Bad format time and date?

- `CREATE TABLE `test` (delivery date, delivery2 datetime);`
- `insert into `test` values (20110208, 2011020811111);`
- `insert into `test` values ('2011-02-08', '2011-02-08 11:11:12');`
- `insert into `test` values ('2011*02*08', '2011#02#08 11#11#13');`

delivery	delivery2
2011-02-08	2011-02-08 11:11:11
2011-02-08	2011-02-08 11:11:12
2011-02-08	2011-02-08 11:11:13

Current_time and now()

- `CREATE TABLE `test` (delivery datetime);`
- `insert into `test` (delivery) values (now());`
- `insert into `test` (delivery) values (current_time);`

delivery
2015-02-13 13:57:49
2015-02-13 13:57:49

If you need only date or time
Curdate() for current date
Curtime() for current time

- `CREATE TABLE `test` (delivery date, delivery2 time);`
- `insert into `test` values (curdate(), curtime());`

delivery	delivery2
2015-03-01	09:34:47

Using 'Check' to set Constraints

```
1 CREATE TABLE flight_infor
2 ( departure datetime CHECK (departure > '2020-01-01 00:00:01'),
3   arrival datetime CHECK (arrival > '2020-01-01 00:00:01'),
4   Planned_customer INT CHECK (Planned_customer > 0),
5   onboarding_customer INT CHECK (onboarding_customer > 0),
6   CHECK (Planned_customer > onboarding_customer),
7   CHECK (departure < arrival)
8 );
```

String Types

- **Char [range: 0 to 255]**
Char(30) can hold up to 30 characters.
- **Varchar [range: 0 to 65,535]**

Question:

What will happen by running these code?

```
CREATE TABLE `test`(story char(12));  
insert into `test` (story) values ("MySQL is good");
```

```
/* SQL Error (1406): Data too long for column  
'story' at row 1 */
```

Compare Char and Varchar

Value	CHAR (4)	Storage Required	VARCHAR (4)	Storage Required
' '	' '	4 bytes	' '	1 byte
'ab'	'ab '	4 bytes	'ab'	3 bytes
'abcd'	'abcd'	4 bytes	'abcd'	5 bytes

**'Varchar' could save space if
the data in a column is
variable in length**

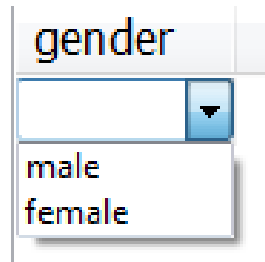
MySQL Spatial Data



- **<http://www.percona.com/blog/2013/10/21/using-the-new-spatial-functions-in-mysql-5-6-for-geo-enabled-applications/>**

ENUM type

- An **ENUM** is a string object with a value chosen from a list of permitted values that are enumerated explicitly in the column specification at table creation time.
- `CREATE TABLE staff (gender ENUM('male', 'female')) ;`



A screenshot of a database application showing a dropdown menu for the 'gender' column. The dropdown is open, displaying two options: 'male' and 'female'.

Spot the errors!

Assuming you are creating a table to store the information of about 200,000 employees for a company.

```
create table employee
(User_Num bigint,
Last_name varchar (50),
First name varcha(50),
Street varchar (100)
Payment float(10.2),
);
```

Errors:

1. 'First name' should be 'First_name'
2. varcha should be varchar
3. Missing comma in 'Street varchar(15)'
4. Payment should better be in decimal type
5. Redundant comma 'Payment float(10.2),'
6. Payment float(10.2), dot should be comma
7. User_Num bigint should be int.

Please use Presemo to provide your
answer or VOTE answers at:
presemo.aalto.fi/drm



Set not null and auto_increment

Host: 130.233.248.81 Database: classicmodels Table: countries

Basic Options Indexes Foreign keys Partitions CREATE code ALTER code

Columns: + Add - Remove ▲ Up ▼ Down

#	Name	Datatype	Length/Set	Unsign...	Allow NULL	Zerofill	Default	Comment
1	COUNTRY_ID	VARCHAR	2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default	
2	COUNTRY_NAME	ENUM	'Finland', 'Sw...	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL	
3	REGION_ID	INT		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		

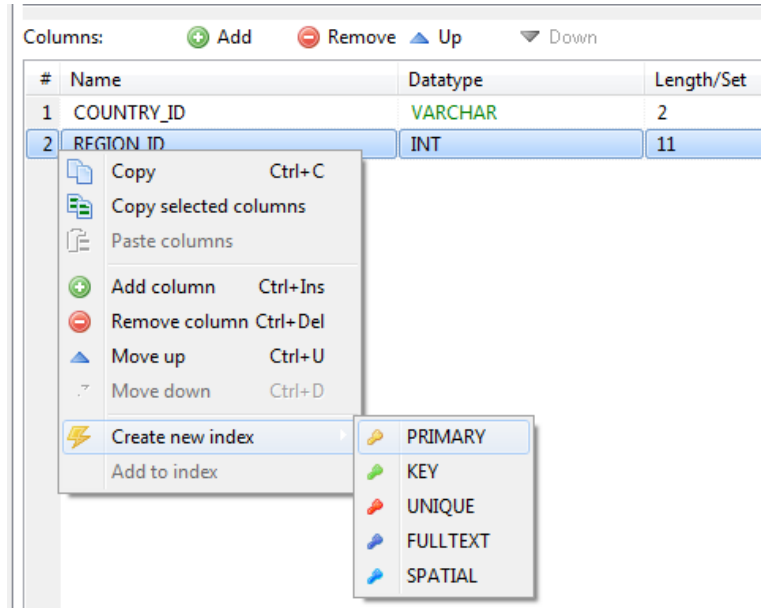
Default value options for COUNTRY_ID:

- ☐ No default value
- ☐ Custom:
- ☐ NULL
- ☐ CURRENT_TIMESTAMP
- ☐ ON UPDATE CURRENT_TIMESTAMP
- ☒ AUTO_INCREMENT

OK Cancel

If you set a variable to be “**Auto_Increment**”, you must define the variable to be the *primary key* as well.

Set up primary key



Go to “basic” view →
Right click on the
column that you want to
set to be the primary key
→ ‘Create new index’ →
‘Primary’

Spot error

```
CREATE TABLE `countries` (  
    `COUNTRY_ID` int DEFAULT NULL Primary Key,  
    `REGION_ID` int(11) NOT NULL AUTO_INCREMENT  
)
```

/ SQL Error (1067): Invalid default value for 'COUNTRY_ID' */*

/ SQL Error (1075): Incorrect table definition; there can be only one auto column and it must be defined as a key */*