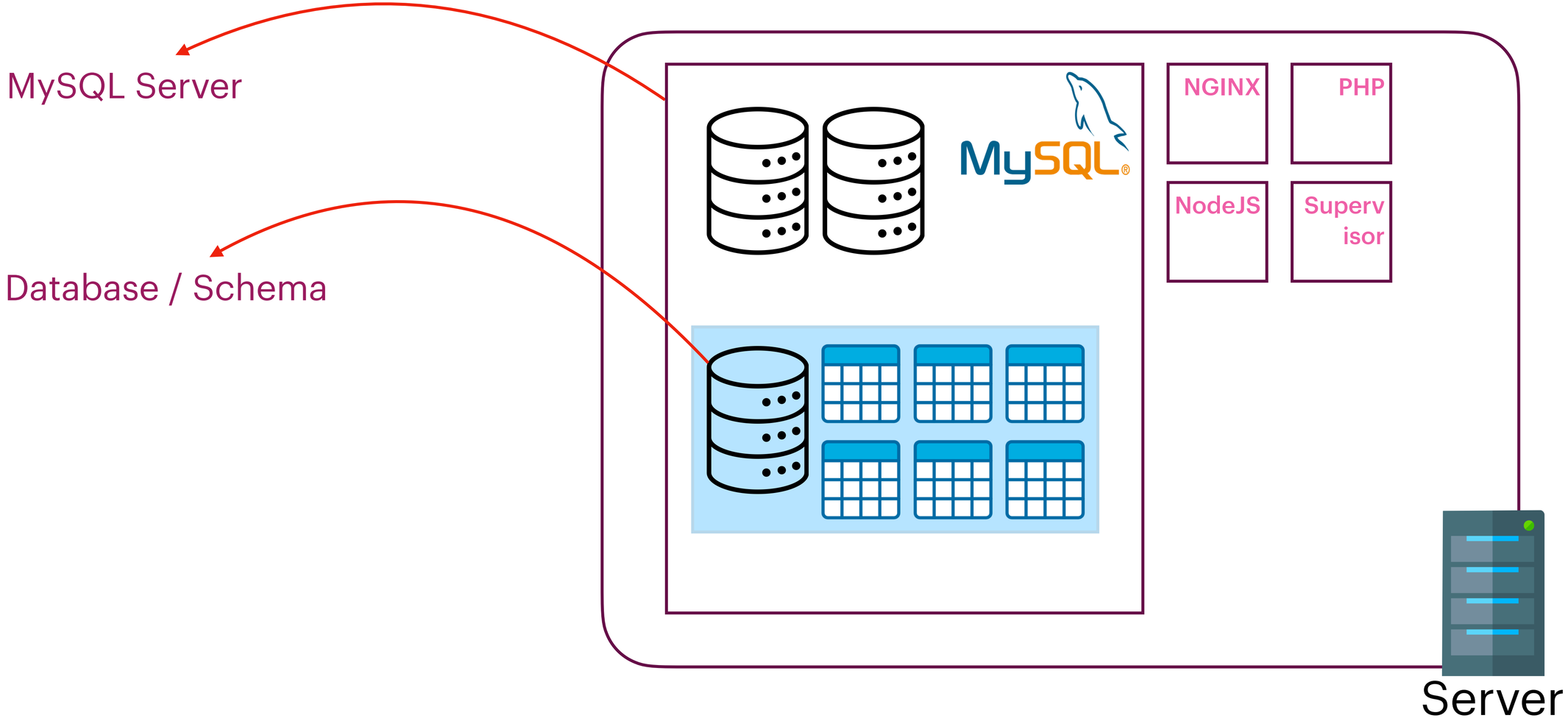
# MySQL Schema

## Managing Database, Tables, Columns

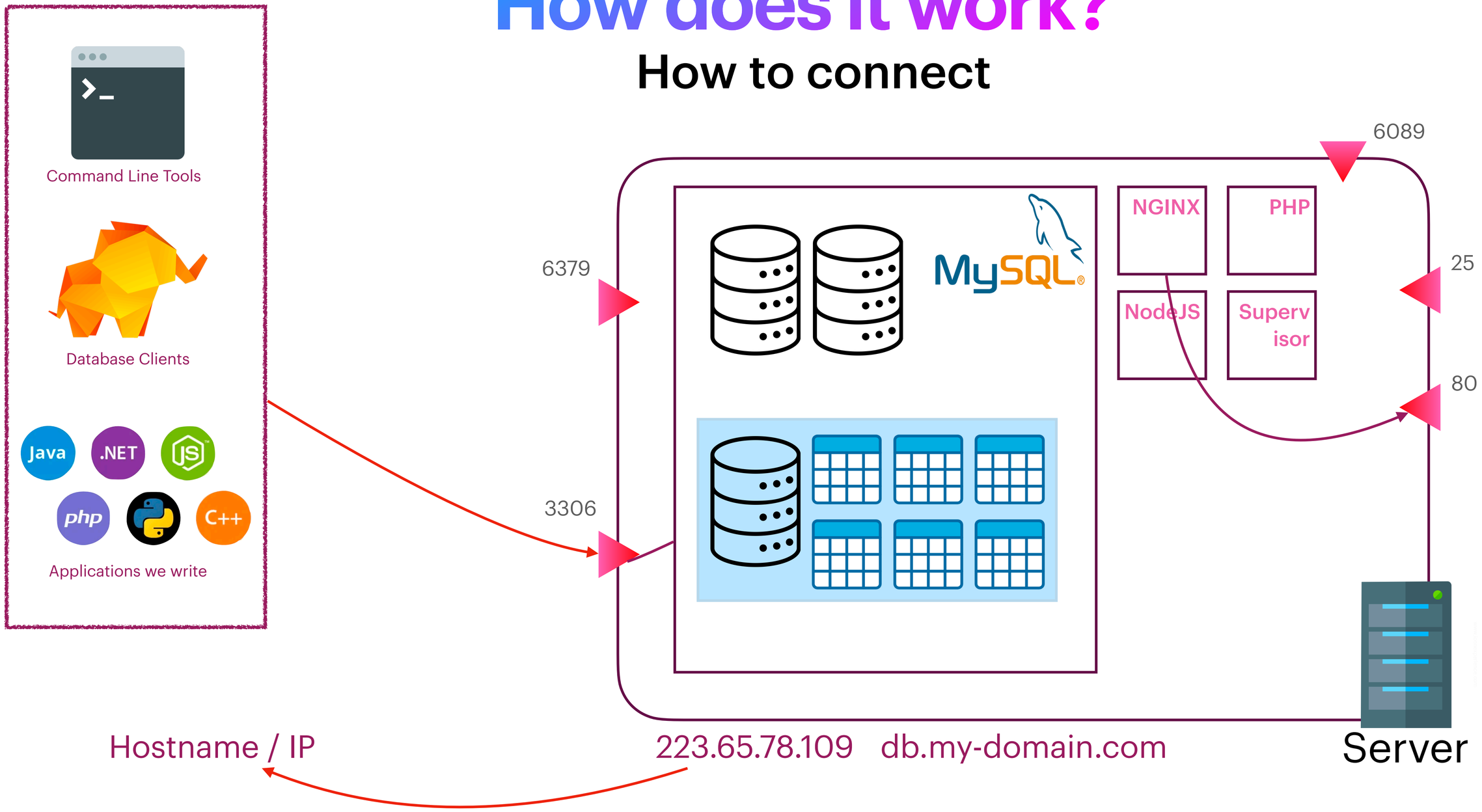
# How does it work?

## The BIG Picture



# How does it work?

## How to connect



# Let's Make A Database

```
CREATE DATABASE 'dokan';    — Create a new Database/Schema in the server
USE 'dokan';                — Start using a Database
```

Sample docker-compose.yml for MySQL and Adminer:

https://gist.github.com/ajaxray/ef6d595ccf181d9048e52a654e5687f4

# Add Tables

```
CREATE TABLE table_name (
    column1 datatype,
    column2 datatype,
    column3 datatype,
   ....
);
```

# Add Tables

```
CREATE TABLE customers (
  id datatype,
  name datatype,
  email datatype,
  phone datatype,
  password datatype,
  created_at datatype
);
```

# Add Tables

```sql
CREATE TABLE customers (
  id INT,
  name VARCHAR(100),
  email VARCHAR(100),
  phone CHAR(15),
  password CHAR(32),
  created_at DATETIME
);
```

# Add Tables

```
CREATE TABLE customers (
  id INT,
  name VARCHAR(100),
  email VARCHAR(100),
  phone CHAR(15),
  password CHAR(32),
  created_at DATETIME
);
```

Type

Length

# Add Tables

```sql
CREATE TABLE customers (
  id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100) NOT NULL,
  email VARCHAR(100),
  phone CHAR(15),
  password CHAR(32),
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP
);
```

# Fine-tune the Schema

## Ground Rules

- ☑ Smaller is better
- ☑ The closest native solution is better
- ☑ Reflect the reality
- ☑ Compact is faster and more efficient

# Data Types

LOREM IPSUM DOLOR

Reg. No.

Date of Issue: / /

Photo Here

ADMISSION FORM

Surname:

Name:

Father's Name:

Mother's Name:

Aadhar Card No.:

Date of Birth: Format (DD/MM/YY) e.g. 07/12/2000

Gender: Male Female Phone:

Place of Birth:

City: Dist:

State:

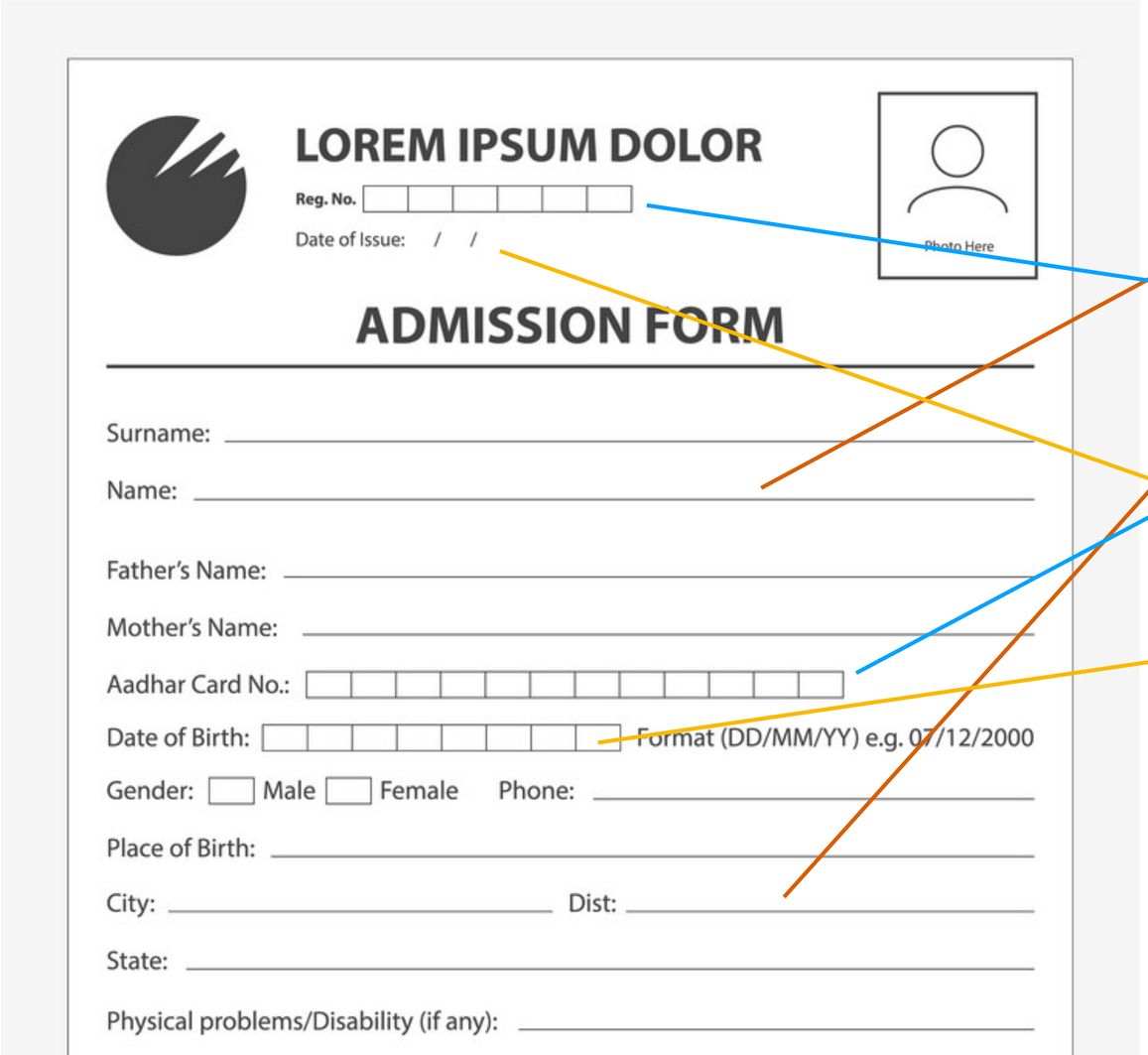
Physical problems/Disability (if any):

Texts

Numeric

Date/Time

# Data Types



LOREM IPSUM DOLOR

Reg. No.

Date of Issue: / /

Photo Here

ADMISSION FORM

Surname:

Name:

Father's Name:

Mother's Name:

Aadhar Card No.:

Date of Birth: Format (DD/MM/YY) e.g. 07/12/2000

Gender: Male Female Phone:

Place of Birth:

City: Dist:

State:

Physical problems/Disability (if any):

Texts

Numeric

Date/Time

# Data Types

Texts

Numeric

Date/Time

# Data Types

Texts ⇒ CHAR, VARCHAR, TEXT, BLOB, BINARY, ENUM

Numeric ⇒ INTEGER, FLOAT, DOUBLE, DECIMAL, BIT, BOOL

Date/Time ⇒ DATE, DATETIME, TIMESTAMP, TIME, YEAR

# Data Types - INT

| TYPE | Storage |
|---|---|
| TINYINT | 1 |
| SMALLINT | 2 |
| MEDIUMNINT | 3 |
| INT | 4 |
| BIGINT | 8 |

# Data Types - INT

| TYPE | Storage |
|---|---|
| TINYINT | 1 |
| SMALLINT | 2 |
| MEDIUMNINT | 3 |
| INT | 4 |
| BIGINT | 8 |

```
00000000   = 0
11111111   = 255
```

# Data Types - INT

| TYPE | Storage |
|---|---|
| TINYINT | 1 |
| SMALLINT | 2 |
| MEDIUMNINT | 3 |
| INT | 4 |
| BIGINT | 8 |

```
00000000   = 0
11111111   = 255
-1111111   = -128 (-1 to -128)
+1111111   = 127 (0 to 127)
```

# Data Types - INT

| TYPE | Storage | Min Signed | Max Signed | Max Unsigned |
|---|---|---|---|---|
| TINYINT | 1 | -128 | 127 | 255 |
| SMALLINT | 2 | -32,768 | 32,767 | 65,535 |
| MEDIUMNINT | 3 | -83,88,608 | 83,88,607 | 1,67,77,215 |
| INT | 4 | -214,74,83,648 | 214,74,83,647 | 429,49,67,295 |
| BIGINT | 8 | -2^63<br>-9223372036854775808 | 2^63 - 1<br>9223372036854775807 | 2^64 - 1<br>1,84,467,44,07,370,95,51,615 |

# Data Types - INT

```
CREATE TABLE int_length (
  id INT AUTO_INCREMENT PRIMARY KEY,
  test_int INT,
  test_int_6 INT(6),
  test_int_2 INT(2)
);
```

What does this numbers means?

Which one takes more storage space?

# Data Types - INT

```
CREATE TABLE int_length (
  id INT AUTO_INCREMENT PRIMARY KEY,
  test_int INT ZEROFILL,              999 => 0000000999
  test_int_6 INT(6) ZEROFILL,         999 => 000999
  test_int_2 INT(2) ZEROFILL          999 => 999
);
```

- ☑ The length in INT is just a min **display length**
- ☑ Only effective with ZEROFILL
- ☑ Has no impact on storage length

# Data Types - DECIMAL

## Available Types

**DECIMAL** => Fixed precision floating-point numbers

**NUMERIC** => Same as Decimal, just an alias

**FLOAT** => Floating-point numbers with approximate values

**DOUBLE** => Same as FLOAT, but with double size and more precise

# Data Types - DECIMAL

## How to define

```
price DECIMAL (10, 2);
```

# Data Types - DECIMAL

## How to define

```
price DECIMAL (10, 2);
```

Maximum number of digits

Digits after the decimal

# Data Types - DECIMAL

## How to define

```
price DECIMAL (10, 2);
```

Maximum number of digits

Digits after the decimal

Which of the following values are acceptable for the length column defined as:

```
length DECIMAL (4, 2);
```

A. 11.02

B. 128.43

C. 4611.02

D. 8.00

# Data Types - DECIMAL

## How to define

```
price DECIMAL (10, 2);
```

Maximum number of digits

Digits after the decimal

Which of the following values are acceptable for the length column defined as:

```
length DECIMAL (4, 2);
```

✅ A. 11.02

❌ B. 128.43

❌ C. 4611.02

✅ D. 8.00

# Data Types - Strings

## How to define

```sql
CREATE TABLE customers (
  id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100) NOT NULL,
  email VARCHAR(100),
  phone CHAR(15),
  password CHAR(32),
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP
);
```

# Data Types - Strings

## CHAR vs VARCHAR

```
email VARCHAR(100)
```

```
password CHAR(32)
```

# Data Types - Strings

## CHAR vs VARCHAR

```
email VARCHAR(100)
```

Variable length string

```
password CHAR(32)
```

Fixed length string

# Data Types - Strings

## CHAR vs VARCHAR

```
email VARCHAR(100)
```

### Variable length string

```
abc@example.com
ab@c.com
a.very.long.email.address@domain.tld
```

```
password CHAR(32)
```

### Fixed length string

```
$A$005$THISISACOMBINATIONOFINVALIDS
$A$005$THISI
XYZ
```

# Data Types - Strings

## Space in bytes

```
CREATE TABLE customers (
  id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100) NOT NULL CHARSET utf8mb4 COLLATE utf8mb4_0900_ai_ci,
  email VARCHAR(100) CHARSET utf8mb4 COLLATE utf8mb4_0900_ai_ci,
  phone CHAR(15) CHARSET utf8mb4 COLLATE utf8mb4_0900_ai_ci,
  password CHAR(32) CHARSET utf8mb4 COLLATE utf8mb4_0900_ai_ci,
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP
);
```

**CHARSET** => A defined **set of characters** that are acceptable for this column

Full list: information_schema.CHARACTER_SETS

**COLLATE** => A set of rules that determine how two or more strings are compared or sorted

Full list: information_schema.COLLATIONS

# Data Types - Strings

## Looooooong texts

| As TEXT | Range |
|---|---|
| TINYTEXT | 255 Characters |
| TEXT | 65,535 Characters |
| MEDIUMNTEXT | 16 MB |
| LONGTEXT | 4 GB |

How to handle

Indexing?

Sorting?

Files?

# Data Types - Strings

## Binary strings

| As TEXT | Range |
|---|---|
| TINYBLOB | 255 Bytes |
| BLOB | 65,535 Bytes |
| MEDIUMNBLLOB | 16 MB |
| LONGBLOB | 4 GB |

# Data Types - Enums

## How to define

```
CREATE TABLE shipment (
  id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
  type ENUM('digital', 'local', 'overseas') NOT NULL,
  status ENUM('pending', 'assigned', 'picked', 'dropped', 'confirmed'),
  … more columns…
);
```

- ☑ Only Valid values are accepted
- ☑ Uses very small size of space to store
- ☑ Human friendly and self explanatory

- Reflecting changes of business logic
- Portability challange
- Ordering by index (underlaying int)

# Data Types - Date-Time

## Available Types

| Type | Range | Purpose |
|---|---|---|
| DATE | 1,000 to 9,999 | Stores a Date (Year, Month, Date) |
| TIME | 10-day range expressed in hours, minutes, and seconds | Stores a time (Hours, Minutes, Seconds) |
| DATETIME | 1000-01-01 00:00:00 to '9999-12-31 23:59:59' | Stores Date and Time (without timezone) |
| TIMESTAMP | 1970-01-01 00:00:01.000000 to 2038-01-19 03:14:07.999999 | Stores Date and Time (with timezone) |
| YEAR | 1901 and 2155 | Only a Year value (2 or 4 digits) |

# Data Types - Date-Time

## DATETIME vs TIMESTAMP

| Type | DATETIME | TIMESTAMP |
|---|---|---|
| Range | '**1000**-01-01 00:00:00' to '**9999**-12-31 23:59:59' | '**1970**-01-01 00:00:01.000000' to '**2038**-01-19 03:14:07.999999' |
| TIMEZONE | No | Yes |
| Storage | 8 Bytes | 4 Bytes |
| Performance | Comparatively slower | Comparatively faster |

# Modify The Schema

## Updating Tables

```
ALTER TABLE table_name
alter_operation column_name [options];
```

**ADD**

```
ALTER TABLE products
ADD COLUMN sku CHAR(8);
```

**DROP COLUMN**

```
ALTER TABLE products
DROP COLUMN weight;
```

**MODIFY COLUMN**

```
ALTER TABLE products
MODIFY COLUMN image_url VARCHAR(1000);
```

# Modify The Schema

## Dropping Things

```
TRUNCATE TABLE table_name;
```

— Clear all the data from a table, but not the table itself

```
DROP TABLE table_name;
```

— Remove an existing table from a database

```
DROP DATABASE database_name;
```

— Remove an existing Database from the server

# Let’s discuss a sample Schema


https://gist.github.com/ajaxray/05de489f214fabbfd0c5b775e988f255

# Questions?