xampp里面使用mysql数据库

1. 打开xampp，启动shell

2，输入mysqladmin -u root password "no744634" 回车

3，xampp文件夹下，的 phpMyAdmin\config.inc.php 文件

4，$cfg['Servers'][$i]['password']="no744634" 保存

5，コマンドプロンプトを起動する 或者直接在xampp的shell里面输入

c:\xampp\mysql\bin

6,mysql -u root -p

7,输入密码 no744634

1,CREATE DATABASE IF NOT EXISTS mydb1;

2,use mydb1;

//或者show databases; 查看有哪些数据库

use test;

**Integer type**

**range of values**

|  |  |  |
| --- | --- | --- |
| Type | Range of values (signed integer)  有符号的的整数  符号ありの場合 | Range of values (unsigned integer)  无符号的整数  符号なしの場合 |
| TINYINT | -128 to 127 百 3位 | 0 to 255 |
| SMALLINT | -32768 to 32767 万 5位 | 0 to 65535 |
| MEDIUMINT | -8388608 to 8388607 百万 7位 | 0 to 16777215 |
| INT OR INTEGER | -2147483648 to 2147483648  十亿10位 | 0 to 4294967295 |
| BIGINT | -9223372036854775808 o  9223372036854775807 | 0 to 18446744073709551615 |

//建表

CREATE TABLE IF NOT EXSIST test(id TINYINT UNSIGNED);

//赋值

INSERT INTO test(id) values (100); //只能插入整数 且在0到255之间

**Display width(表示幅)**

**幅を指定すると指定した幅で数値が表示される。ただ、本来使用可能な桁数よりも小さい表示幅を指定した場合に、表示幅よりも大きい値を指定すると表示幅の設定は無視される**

**ZEROFILL**

**データ型にZEROFILLを付けると入力したデータの桁数が未満の場合、データの前が０で埋められる。ZEROFILLを付けると自動的にUNSIGNEDが付きます。**

**If ZEROFILL is specified, MySQL automatically adds the UNSIGNED attribute to the column**

**ZEROFILL which indicates that when a value is displayed it is padding with zeros up to the display width specified in the column definition.**

**Values that are longer than the display width are displayed in full.**

**If no display width is specified, the unsigned integer default value of 3,5,7,10 or 19 is assumed.**

CREATE TABLE IF NOT EXISTS test(id TINYINT ZEROFILL,id2 TINYINT);

INSERT INTO test(id,id2) values(24,25);

INSERT INTO test(id,id2) VALUES(7,8);

INSERT INTO test(id,id2) VALUES(110,120);

SELECT \* FROM test;

+------+------+

| id | id2 |

+------+------+

| 024 | 25 |

| 007 | 8 |

| 110 | 120 |

+------+------+

id的24 和 7 前面被加了零

//罗列所有的表格

SHOW TABLES;

//删除表格

DROP TABLE test;

DROP TABLE IF EXISTS test;

CREATE TABLE test(id SMALLINT(3) ZEROFILL,id2 SMALLINT ZEROFILL);

INSERT INTO test(id,id2) VALUES (28,28);

INSERT INTO test(id,id2) VALUES (1234,15);

SELECT \* FROM test;

+------+-------+

| id | id2 |

+------+-------+

| 028 | 00028 |

| 1234 | 00015 |

+------+-------+

the main reason for specifying a display width is to use it in conjunction with the ZEROFILL clause;

只显示3位数字，不足的以零填充。超过3位数字的显示全部。

**Fixed point types(固定小数型)（小数就用DECIMAL）**

**NUMERIC　は　DECIMAL　の別名（べつめい）です。They are equivalent.**

**MySQLではDECIMALを使うことが好まれるです。**

**DECIMALの省略表記としてDECを使うこともできる。**

DECIMAL(5,2);

//A field specified as DECIMAL(5,2) numbers from -999.99 to 999.99 can be stored

//the first number called the precision,indicates how many significant digits can be stored

//the second number called the scale,indicates the number of digits that can be stored after the decimal point.

//the DECIMAL can be followed by the optional key words UNSIGNED ZEROFILL.

DECIMAL(10);

//このように、小数点以下を指定しない場合は、DECIMAL(10,0)として扱われ、小数点以下はないものとして扱われる。

DECIMAL;

//このように、何も指定しない場合は、デフォルトの10桁として、DECIMAL(10,0)　が設定されたものとして処理される

// DECIMALの最大桁数は65桁で、小数点以下は30桁までです。

**Floating point types（浮動小数型）**

DOUBLE FLOAT REAL

// FLOAT and REAL are equivalent.they specify that single precision will be used.

// DOUBLE is specify that double precision will be used.

// useage is same as fixed point types.

// FLOAT は小数点以下7桁までです。

// DOUBLE　は小数点以下14までです。

// MySQLでは 誤差を起こさせないため、DOUBLE を使うべきです

DECIMAL と　DOUBLE の違い

１、DOUBLEのメリットを理解するまで、Webサイトを作る場合はDECIMALを使ったほうが無難です。

**Character types**

|  |  |  |
| --- | --- | --- |
| **Type** | **Range of length** | **details** |
| **CHAR(length)** | **1 to 255 characters.**  **Specified as an integer**  **If length is omitted a value of one is assumed.** | **A fixed length string.**  **string of fewer characters are stored right-padded with spaces.**  **固定長文字列、郵便番号、社員番号データの桁数が決まっているものに向ける。** |
| **VARCHAR(length)** | **1 to 255 characters.**  **必须跟上长度** | **可変長文字列,氏名、書籍名などデータの桁数が変動可能のものに向ける**  **e.g. varchar(10);**  **半角10文字、全角5文字。** |
| **BLOG or TEXT** | **1 to 65535** | **BLOG stands for “binary large object” Sorting or comparison of stored data is case sensitive**  **and case insensitive with TEXT** |
| **TINYBLOG**  **or**  **TINYTEXT** | **1 to 255** |  |
| **MEDIUMBLOG**  **or**  **MEDIUMTEXT** | **1 to 16777215** |  |
| **LONGBLOG**  **or**  **LONGTEXT** | **1 to 4294967295** |  |

**Enumeration types**

**ENUM(“M”,”F”)**

the field must take one of a specified set of values,which are separated by commas.

リストの中の文字列定数の一つを指定する

CREATE TABLE IF NOT EXISTS test(color ENUM("red","blue","black"));

INSERT INTO test(color) VALUES ("blue");

INSERT INTO test(color) VALUES (NULL);

//NULL はリストに含まれていない場合でも、データとして指定することができる。

INSERT INTO test(color) VALUES(2);

//リストされた文字列には順にインデックスが割り当てられる。１から始まる。１は "red"

SELECT \* FROM test;

+-------+

| color |

+-------+

| blue |

| NULL |

| blue |

+-------+

//查询的时候也可以用序号查询

SELECT \* FROM test WHERE color="blue";

+-------+

| color |

+-------+

| blue |

| blue |

+-------+

SELECT \* FROM test WHERE color=2;

+-------+

| color |

+-------+

| blue |

| blue |

+-------+

**Data and time types**

|  |  |  |
| --- | --- | --- |
| **type** | **format** | **example** |
| **DATE** | **“YYYY-MM-DD”** | **‘2020-02-28’** |
| **YEAR** | **“YYYY”** | **“2020”** |
| **TIME** | **“HH:MM:SS”** | **“17:08:57”** |
| **DATETIME** | **“YYYY-MM-DD HH:MM:SS”** | **‘2020-02-28 17:08:57 ’** |
| **TIMESTAMP** | **“YYYY-MM-DD HH:MM:SS”** | **‘2020-02-28 17:08:57 ’** |

TIMESTAMP is the same as DATETIME with the current date and time automaticially converted to UTC.

it is automatically set when a record is inserted into a table and automatically update when the record is update.

it should be not include explicitly in the INSERT INTO and UPDATE commands.

//他们有很多不同，以后再说

**NOT NULL**

// NOT NULL specified that it is not permitted to give the field a value of NULL.

CREATE TABLE IF NOT EXISTS test(num1 INT NOT NULL,num2 INT,num3 INT);

INSERT INTO test(num1,num2,num3) VALUES(null,null,null); //报错Column 'num1' cannot be null

INSERT INTO test(num1) VALUES(25);

SELECT \* FROM test;

+------+------+------+

| num1 | num2 | num3 |

+------+------+------+

| 25 | NULL | NULL |

+------+------+------+

可以看出当字段属性中没有NOT NULL且没有被赋值的时候，内容默认为null

// If a field is specified as NOT NULL,any attempt to not assign it a value

// by an INSERT INTO or UPLOAD commandwill cause the DEFAULT value to be used.

CREATE TABLE IF NOT EXISTS test(num1 INT NOT NULL DEFAULT 100,num2 INT,num3 INT);

show columns from test; //查看每个字段的属性

+-------+---------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+-------+---------+------+-----+---------+-------+

| num1 | int(11) | NO | | 100 | |

| num2 | int(11) | YES | | NULL | |

| num3 | int(11) | YES | | NULL | |

+-------+---------+------+-----+---------+-------+

NSERT INTO test(num2) VALUES(3);

SELECT \* FROM test;

+------+------+------+

| num1 | num2 | num3 |

+------+------+------+

| 100 | 3 | NULL |

+------+------+------+

**DEFAULT**

DEFAULT value: specified a value to be given to a field if no value is assigned by an INSERT INTO command or an UPDATE command.

1,if the field is numeric，the default value must be a number.

INT(3) DEFAULT 0

DECIMAL DEFAULT 6.5

2,if the field is string the default must be a string value or NULL.

VARCHAR(6) DEFAULT "YES"

VARCHAR(6) DEFAULT '' //空字符串

VARCHAR(6) DEFAULT NULL

3.in the case of an enumeration field the default value must be one of the specified values;

ENUM("GOOD","BAD") DEFAULT "BAD"

If it is not appropriate to make the specified value the default，the

NOT NULL DEFAULT should not be used.

4,BLOG and TEXT fields cannot be given a default value.

Change a default value

1,we can change or set a default value of a field by including DEFAULT

Default-value in the field specification. e.g.

ALTER TABLE tablename MODIFY fieldname INT NOT NULL DEFAULT 10;

2,the only change is to default value

ALTER TABLE tablename ALTER fieldname SET DEFAULT 10;

3,remove the default value

ALTER TABLE tablename ALTER fieldname DROP DEFAULT ;

**AUTO\_INCREMENT clause**

1,there may be at most one auto\_increment field for any table.

2,it is normally disirable (值得向往的，值得做的) for an auto\_increment field,

also to be sepcified as PRIMARY KEY.

还有好多内容不知道

//set a starting value for an AUTO\_INCREMENT field.

Alter TABLE tablename AUTO\_INCREMENT = unsigned integer.

**PRIMARY KEY clause**

1, Although it is not essential(必须的),it is recommended for

efficiency（效率） reasons that a primary key field is always

explicitly(明确的) given a NOT NULL specification(明确规定，具体说明).

Change a primary key

1,remove an existing primary key

ALTER TABLE tablename DROP fieldname;

2,create a new primary key from one field

ALTER TABLE tablename ADD PRIMARY KEY(fieldname);

3,create a primary key from more than one field

ALTER TABLE tablename ADD PRIMARY KEY(fieldname1,fieldname2);

**Make a backup of a table**

1,create new a table test2 with same structure and field definitions of test

in the same database.

CREATE TABLE test2 LiKE test;

2,create new a table test3 with same structure and field definitions of test2

in the different database.

CREATE TABLE test3 LIKE test.test2;

// the two tables can be in different databases, that are accessible to the user, in which case each table name should be preceded by the name of a

database followed by a dot.

//在另一个数据库mydb1里复制test数据库下的test2表的结构。命名为test3

3,If we want the contents of test to be copied to test2,

thus giving an identical copy of the original table.

CREATE TABLE test2 LiKE test;

INSERT INTO test2 SELECT \* FROM test;

4.If we want the contents of test.test to be copied to mydb1.test2

CREATE TABLE test2 LIKE test.test;

INSERT INTO test2 SELECT \* FROM test.test;

**查看表的结构**

// find the structure of a table.

DESC test;

or

show create table test;

or

show columns from test;

//change the name of a table

ALTER TABLE oldname RENAMT TO newname; //alter not alert

**chenge the name and structure of a Field**

// chenge the name and structure of a Field;

//不能只更改字段名，其属性就算不该也要再写一遍

ALTER TABLE tablename CHANGE oldfieldname newfieldname field-specification;

//change the structure of a field

ALTER TABLE tablename MODIFY fieldname field-specification;

If we are altering the specification of the PRIMARY KEY field,

there is no need to include the PRIMARY KEY clause.

Once this is set it remains in force until it is removed using an

ALTER TABLE tablename DROP PRIMARY KEY command.

**新建表格**

CREATE TABLE mytable1(

refnum INT NOT NULL DEFAULT 0 PRIMARY KEY,

Forname VARCHAR(30) NOT NULL,

surname VARCHAR(30) NOT NULL,

sex ENUM("M","F"),

occupation VARCHAR(30),

cityBorn VARCHAR(30),

yearBorn YEAR,

numchild INT

);

**adding a record to the table**

1,Field values that are purely numerical need not be enclosed in quotes,

It is best to omit the quote if the field is of a numerical type, but

for efficiency reasons we recommend that the quotes are retained in

the case of atext field even when the value itself is purely numerical.

2,If the list of field names before the VALUES keyword includes all the

Fields in the correct "left to right" order we can omit it and it will

assumed bythe MySQL system as a default.

INSERT INTO tablename VALUES (“zhang”,”wuhan”);

INSERT INTO tablename (name,birthplace) VALUES(“zhang”,”wuhan”);

3.MySQL systems require a semicolon at the end of each command but this

is not applicable when executing MySQL commands using php.

(php里面的数据库命令行不需要分号)

//adding a record to the table

INSERT INTO mytable1 values("2461","Ann","Williams","F","doctor","Pairs","1997","2");

//not all field values need to be entered and the field names can be placed in any order

INSERT INTO mytable1(Forname,Surname,occupation,refnum,cityBorn) values ("Martin","Johnson","plumber","1851","London");

INSERT INTO mytable1 VALUES("2547","Mary","Johnson","F","technician","Pairs","1989","3");

INSERT INTO mytable1 VALUES("634","James","Robinson","M","none","Geneva","2007","0");

INSERT INTO mytable1 VALUES("1927","bryan","Brown","M","engineer","Toronto","1987","2");

INSERT INTO mytable1 VALUES("4821","Sarah","Green","F","engineer","Pairs","1982","1");

INSERT INTO mytable1 VALUES("3842","Frances","Bryce","F","translator","Northampton","1980","2");

**REPLACE INTO**

The REPLACE INTO or REPLACE command (the word INTO is optional) is the same as INSERT INTOexcept that if an exiting row has the same primary key value as a new row to be inserted,theexiting row is deleted .In that situation an INSERT INTO command would fail.

REPLACE INTO mytable1 VALUES("634","James","Robinson","M","none","Geneva","2007","3");

**DELETE**

Delete a record

DELETE FROM mytable1 WHERE refnum =1972;

DELETE FROM mytable1 WHERE refnum =1972 LIMIT 1;

DELETE FROM mytable1 WHERE occupation = "engineer" AND yearBorn = 1987;

DELETE FROM mytable1 WHERE occupation = "engineer" AND yearBorn = 1987 AND numchild=1;

Delete all records preserves structure of the table

TRUNCATE mytable1;

Delete a table from database

DROP TABLE mytable1;

**Changing a table**

**Add a New Field**

ALTER TABLE tablename ADD fieldname field-specification;

//The field-specification can include all the clauses illustrate earlier.

If there is nothing following the field-specification the new field is placed in the right-most position.We can use a clause AFTER fieldnameN to place it after fieldnameN or FIRST to place it in the left-most position.

ALTER TABLE tablename ADD fieldname field-specification AFTER fieldnameN;

ALTER TABLE tablename ADD fieldname field-specification FIRST;

Delete a field(column)

ALTER TABLE mytable1 DROP passportnum;

ALTER TABLE mytable1 DROP COLUMN passportnum;

(The word COLUMN is optional)

ALTER TABLE mytable1 DROP COLUNM passportnum, DROP occupation,DROP Forname;

(The word COLUMN after DROP is optional each time)

Updating a table

//Fill the missing values for Martin Johnson.

UPDATE mytable1 SET sex ="M",occupation="butcher",

yearBorn="1970",numchild ="99" WHERE refnum="1851" LIMIT 1;

//set all the values in the extra column to the value "no"

ALTER TABLE mytable1 ADD extra varchar(10);

UPDATE mytable1 SET extra ="no";

//set the value in the extra field to "yes" for all the women

//with more than two children

//use the 99 to indicate 'unknown'

UPDATE mytable1 SET extra ="yes" WHERE sex="F" AND (numchild>2 AND numchild !=99);

illustrate some further facilities of update 1

//change the value of field extra for Martin Johnson to the value "5734"

// the 5734 is a string

UPDATE mytable1 SET extra ="5734" WHERE refnum="1851";

// change the extra field to be of type INT(6);

ALTER TABLE mytable1 CHANGE extra extra INT(6);

//yes and no are replaced by zero ;

//string '5734' has been replaced by the integer value 5734;

UPDATE mytable1 SET extra=yearBorn-1;

or

UPDATE mytable1 SET extra=2020-yearBorn;

illustrate some further facilities of update 2

ALTER TABLE mytable1 CHANGE extra extra varchar(50);

UPDATE mytable1 SET extra =concat(Forname," ",Surname);

//If the value in the Forname or Surname field subsequently changes,

//the value of extra will not change

UPDATE mytable1 SET extra ="yes" WHERE refnum=4821 LIMIT 1;

//it does not spend time searching for another one

UPDATE mytable1 SET extra =NULL where yearBorn <1980;

//NULL is indicate there is no value

WHERE yearBorn IS NULL

or

WHERE yearBorn IS NOT NULL

//test whether a field is or not null

ALTER TABLE mytable1 DROP extra;

//extra has now outlived its usefulness so delete it;

**ORDER BY**

SELECT \* from mytable1;

SELECT Forname,Surname,sex From mytable1;

//当用php代码来执行这两句代码时，返回的是1（成功） 或空字符串（失败）

SELECT \* FROM mytable1;

SELECT \* FROM mytable1 ORDER BY refnum;

SELECT \* FROM mytable1 ORDER BY refnum ASC;

// they are arranged in ascending order of refnum;

SELECT \* FROM mytable1 ORDER BY refnum DESC;

SELECT \* FROM mytable1 ORDER BY Surname; //字母顺

SELECT \* FROM mytable1 ORDER BY Surname DESC; //反字母顺

SELECT \* FROM mytable1 ORDER BY Surname,Forname DESC;

// Surname 反字母顺，如果两个记录一样，Forname反字母顺；

//we may have a field named size with values"small","average","large".

//self-explannatory 自定义

SELECT \* FROM mytable1 ORDER BY FIELD(size,"small","average","large");

SELECT \* FROM mytable1 order by FIELD(season,"spring","winter","Autumn","summer");

SELECT \* FROM mytable1 order by FIELD(numchild,2,3,4,5,6,7,8,9,0,99);

SELECT \* FROM mytable1 order by FIELD(numchild,2,3,4,5,6,7,8,9,0,99) DESC;

**WHERE CLAUSES**

Note that if there is a WHERE clause it must come before ORDER BY clause.

SELECT \* FROM mytable1

WHERE sex="F" AND (year>1980 OR numchild != 99)

AND NOT(occupation="doctor" OR occupation="engineer");

// if i1,i2,i3,i4,i5 are integer fields

WHERE i1+i2 < i3-i4+i5;

WHERE GREATEST(i1,i2,i3)>99; // return the largest one

WHERE LEAST(i1,i2,i3)>LEAST(i4,i5);

//BETWEEN a AND b

//NOT BETWEEN a AND b

//operator can be used with either numerical or character fields

//(aより大きいか等しく、bより小さいか等しい場合)

SELECT \* FROM mytable1 WHERE Forname BETWEEN "Ann" AND "James" ORDER BY Forname;

//这样可以

SELECT \* FROM mytable1 WHERE Forname BETWEEN "James" AND "Ann" ORDER BY Forname;

//这样不行

**Displaying values that are not Fields**

SELECT CONCAT(Forname," ",Surname),sex FROM mytable1

WHERE numchild>1 AND numchild !=99

ORDER BY Surname,Forname;

//using CONCAT function can display values that are not explicitly

//given field in the table

**Limit clause**

// the four youngest people in the table。展示4条记录

SELECT \* FROM mytable1 ORDER BY yearBorn DESC LIMIT 4;

//从第五条记录开始展示10条记录

SELECT \* FROM mytable1 ORDER BY yearBorn DESC LIMIT 4,10;

//to save time

SELECT \* FROM mytable1 WHERE refnum=2461 LIMIT 1;