数据库的基本操作：

1.数据库的增、删、改、查(查看数据库)

2．字符集

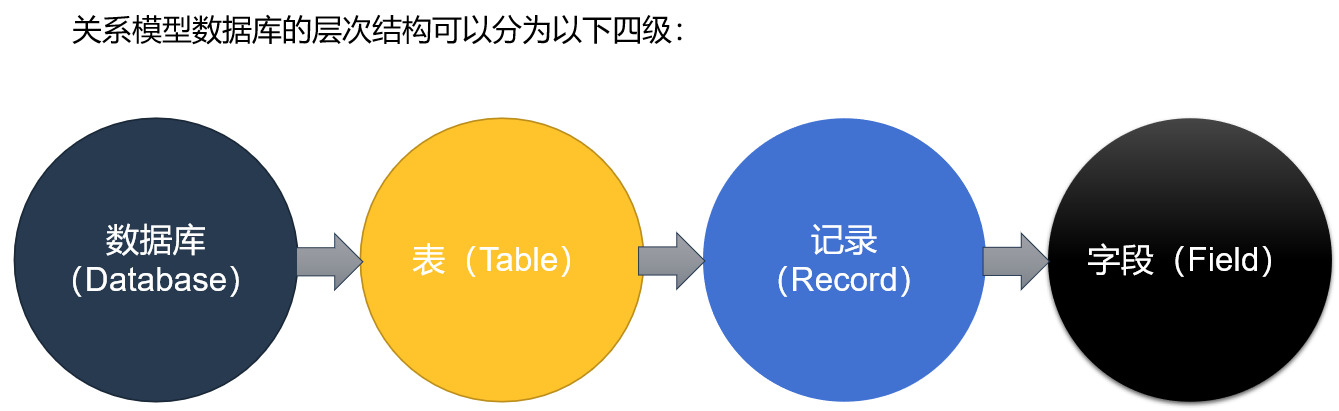
3．数据表的增加

4．常见的数据类型

库、表：create ,drop ,alter ,show database;

数据：insert，delete ,update ,select

1.数据库的基本操作



1.1 数据库的基本概念

什么是数据库？

数据库可以看作是存储数据对象的一个容器

数据库本身是数据库管理系统的核心和基础

不同数据库的数据对象相对隔离

在mysql中，数据库可以分为系统数据库和用户数据库两大类.

**系统数据库：**

Information\_schema:主要存储了系统中的一些数据对象信息，例如用户表信息，列信息，权限信息，字符集信息和分区信息。

Performance\_schema:主要存储数据库服务器的性能参数。

MySQL：存储数据库的用户、权限设置、关键字等，MySQL自己需要使用的控制和管理信息

Sys:是系统数据库，包括了存储过程、自定义函数等信息。

用户数据库：用户数据库是用户实际需求所创建的数据库

[例1.1]查看当前拥有的数据库语句：

show databases;

1.2 创建数据库

创建数据库语法：

create databases 数据库名 [charset=字符集]

数据库名：

* 一般由字母和下划线组成，不允许有空格，可以是英文单词、英文短语或相应缩写；
* 不允许是MySQL关键字；
* 长度最好不超过128位；
* 不能与其他数据库同名

[例1.2.1]创建数据库db\_shop。

Create database db\_shop;

查看指定数据库的创建信息语法：

Show create database数据库名称;

[例1.2.2]：查看db\_shop的创建信息

Show create database db\_shop;

1.3 修改数据库

修改数据库语法：

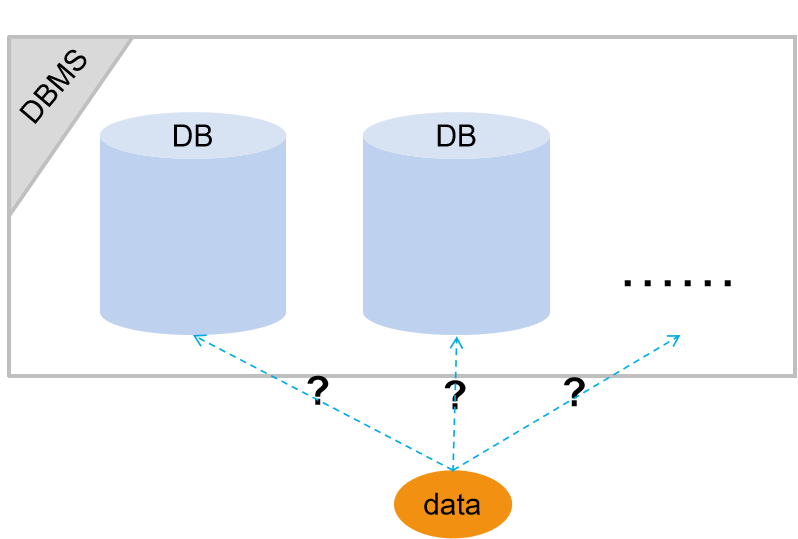
**Alter databases 数据库名字 {character set 字符集 | collate 校对规则}**

{}意味着必须写，但是可以选择去写{}中的内容任意一个

[例1.3.1]：修改db\_shop，将字符集由utf8变成latin1

Alter database db\_shop character set = latin1

1.4选择数据库



[例1.4.1 ]两个数据库A和B中都有user表，当我要查询A库中的user表中数据时，怎样选取？

1. 插入数据时写全，库.表

Insert into xgl.student values(‘a’)；

1. 直接用use进入数据库

Use xgl；

1.5 删除数据库

删除数据库语法：

Drop database xgl;

[例1.5.1]：删除db\_shop数据库

Drop database db\_shop;

2.字符集

       字符集是指一套文字符号及其编码和排序规则的集合。实际应用中，用户通常会向MySQL系统中插入大量文字，但计算机只能存储二进制码，这时就需要通过字符集将文字转换为二进制码再存入数据库。

      MySQL字符集包括字符集和排序规则两个概念，其中字符集定义MySQL存储字符串的方式，排序规则定义字符串的排序方式。每个字符集至少对应一个排序规则

[例2.1.1]导入xkgl数据库时，选择latin1的字符集，看一下数据表里的数据有什么特点。

编码方式的不同，导致出现无法识别的字符，支持中文字符集一般为GBK或者UTF8的编码方式。

用户应该在向数据库插入数据之前确定字符集的种类。MySQL默认的字符集为latin1，字符集和排序规则分为四个级别：\_数据库级别\_、\_表的级别 、服务器的级别、和某个库的级别\_。

查看当前数据库级别的字符集:

SHOW VARIABLES LIKE 'character\_set\_server';

查看当前数据表级别的字符集:

SHOW VARIABLES LIKE 'character\_set\_database';

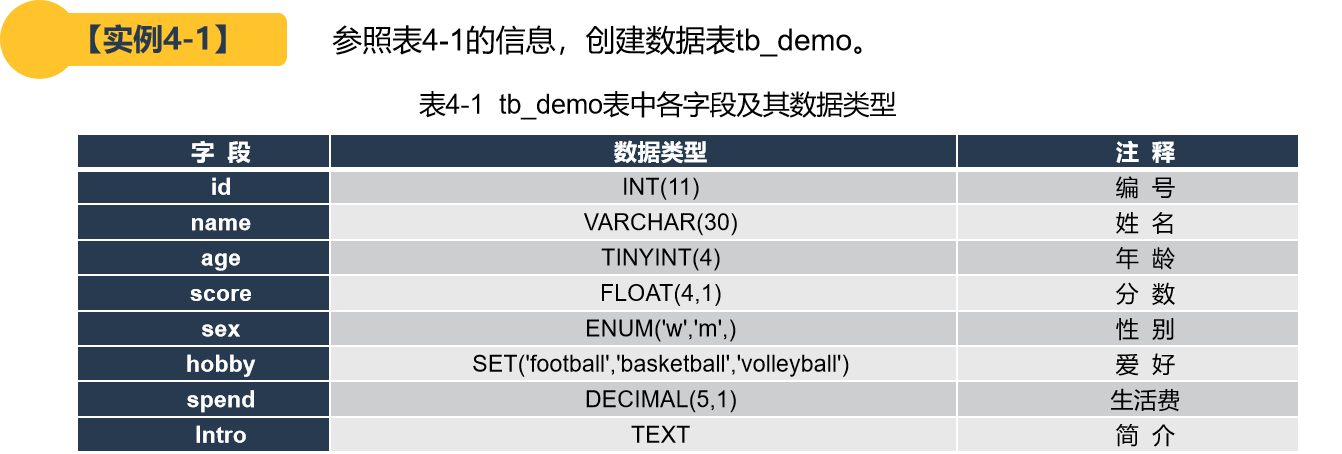
数据库默认的字符集与服务器设置的字符集相同，数据库的字符集和排序规则可以在数据库创建时指定，也可以在数据库创建之后修改。

创建数据库时定义字符集语法:

Create database skgl character set latin1;

3 数据表操作（包含数据类型以及完整性约束）

1.创建数据表

[例1.1]根据下图信息，实现该数据表

创建数据表，实际上是规定 列名及属性\_和实现完整约束的过程的过程，基本语法形式如下：

Create table 表名(

字段名1 数据类型1 [完整性约束],

字段名2 数据类型2 [],

···

字段名n 数据类型n []

)

列名及属性：数据表的基本构成，规定了这一序列存放了什么数据，以及相应的数据类型。

完整的约束：一定程度上限制了对数据的处理，确保数据的正确性。

数据表命名应遵循以下原则：

* 长度最好不超过30个字符；
* 多个单词之间使用下划线“\_”分隔，不允许有空格；
* 不允许为MySQL关键字；
* 不允许与同一数据库中的其他数据表同名。

2.数据类型

       数据表由多个字段构成，每个字段可以指定不同的数据类型，数据类型用于规定数据的存储格式、约束和有效范围。在定义数据表字段时，应根据实际情况选择合适的数据类型。MySQL提供的数据类型主要包括数值类型、日期和时间类型、字符串类型等。

2.1数值型：

|  |  |  |  |
| --- | --- | --- | --- |
| **数据类型** | **所占字节** | **存储范围（有符号）** | **存储范围（无符号）** |
| **TINYINT** | 1字节 | -128～127 | 0～255 |
| **SMALLINT** | 2字节 | -32768～32767 | 0～65535 |
| **MEDIUMINT** | 3字节 | -8388608～8388607 | 0～16777215 |
| **INT** | 4字节 | -2147483648～2147483647 | 0～4294967295 |
| **BIGINT** | 8字节 | -9223372036854775808～9223372036854775807 | 0～18446744073709551615 |

2.1.1整型

[实验]在课堂的tb\_demo的age字段中插入300，能够成功吗？如果能，截图，如果不能，说出错误提示是什么？

2.1.2浮点数类型和定点数类型：

浮点数类型分为两种，单精度（FLOAT）和双精度（DOUBLE），而定点数类型只有DECIMAL一种。

**浮点数类型和定点数类型都可以使用类型名称后面加“(M,D)”的形式来定义**

* M称为精度，表示数值的位数（整数位+小数位）；
* D称为标度，表示小数点后的位数。

|  |  |  |  |
| --- | --- | --- | --- |
| **数据类型** | **占用字节** | **负数存储范围** | **非负数存储范围** |
| **FLOAT** | 4字节 | -3.402823466E+38～-1.175494351E-38 | 0和1.175494351E-38～3.402823466E+38 |
| **DOUBLE** | 8字节 | -1.7976931348623157E+308～-2.2250738585072014E-308 | 0和2.2250738585072014E-308～1.7976931348623157E+308 |
| **DECIMAL** | M+2字节 | 与DOUBLE相同 | 与DOUBLE相同 |

[实验]在课堂的tb\_demo插入如下数据（分数90.16），数据能插入成功吗？插入成功的数据是输入的数据吗？

能插入成功，但不是输入的数据

2.2 日期类型

MySQL 5.7版本中用于存储日期和时间的数据类型有YEAR，DATE，TIME，DATETIME和TIMESTAMP，每一种类型都有固定的存储范围。

向日期和时间类型的字段插入数据时，最好使用引号将值包含起来。

Insert into t(y) values(“1991”)

|  |  |  |  |
| --- | --- | --- | --- |
| **数据类型** | **占用字节** | **存储格式** | **存储范围** |
| **YEAR** | 1字节 | YYYY | 1901～2155 |
| **DATE** | 3字节 | YYYY-MM-DD | 1000-01-01～9999-12-31 |
| **TIME** | 3字节+小数秒存储 | HH:MM:SS | -838:59:59～838:59:59 |
| **DATETIME** | 5字节+小数秒存储 | YYYY-MM-DD HH:MM:SS | 1000-01-01 00:00:00～9999-12-31 23:59:59 |
| **TIMESTAMP** | 4字节+小数秒存储 | YYYY-MM-DD HH:MM:SS | 1970-01-01 00:00:01 UTC～2038-01-19 03:14:07 UTC |

**ＴＩＭＥＳＴＡＭＰ时间戳数据类型，当该行数据被插入修改的时候，时间会随之更新**

2.3 字符串类型

MySQL支持的字符串数据类型包括CHAR，VARCHAR，BINARY，VARBINARY，BLOB，TEXT，ENUM和SET等。

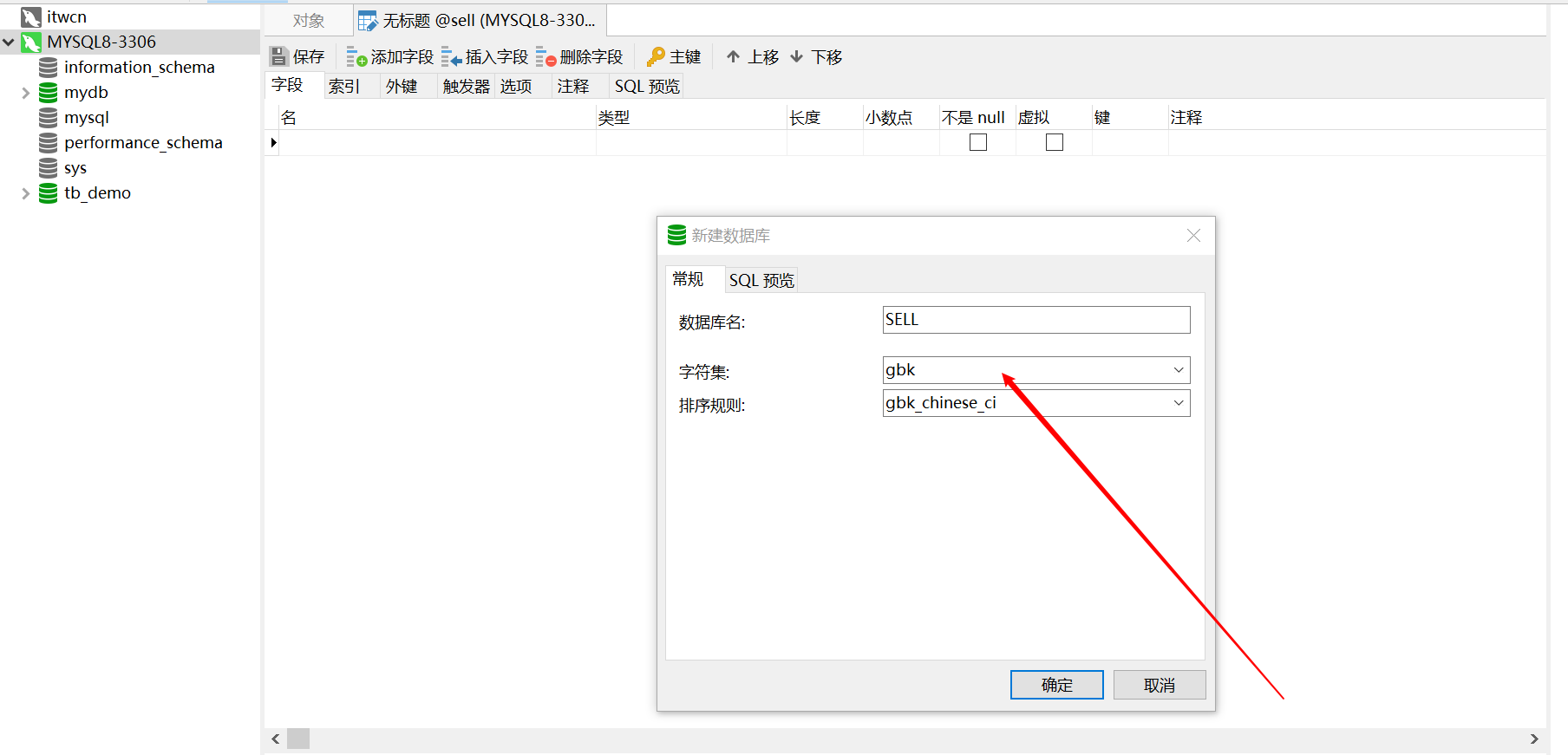
|  |  |  |  |
| --- | --- | --- | --- |
| **类型名称** | **占用字节** | **存储范围** | **备注** |
| **CHAR(M)** | M\*w字节 | 0<=M<=255 | ｍ的意思是该列最多可存储几个字符，定义时m不可少。  较短的字符串用ｃｈａr，ｃｈａｒ类型的长度是定长 |
| **VARCHAR(M)** | L+1字节 | 0<=M<=65535 | 较短的字符串用ｃｈａｒ， |
| **BINARY(N)** | N字节 | 0<=N<=255 | N的意思是该列最多可存储几个字节，定义时ｎ不可少  存放较短的二进制数据，比如图片 |
| **VARBINARY(N)** | L+1字节 | 0<=N<=65535 | 短二进制，变长 |
| **BLOB** | L+2字节 | L<216 | 存放较大的二进制数据，比如音频视频等 |
| **TEXT** | L+2字节 | L<216 | 较长的文本字符串 |
| **ENUM** | 1或2字节 | 0～65535 | 枚举，适用于单选项的数据 |
| **SET** | 1、2、3、4或8字节 | 最多64个成员 | 适用于多选项的数据 |

[例3.1]实现该数据表

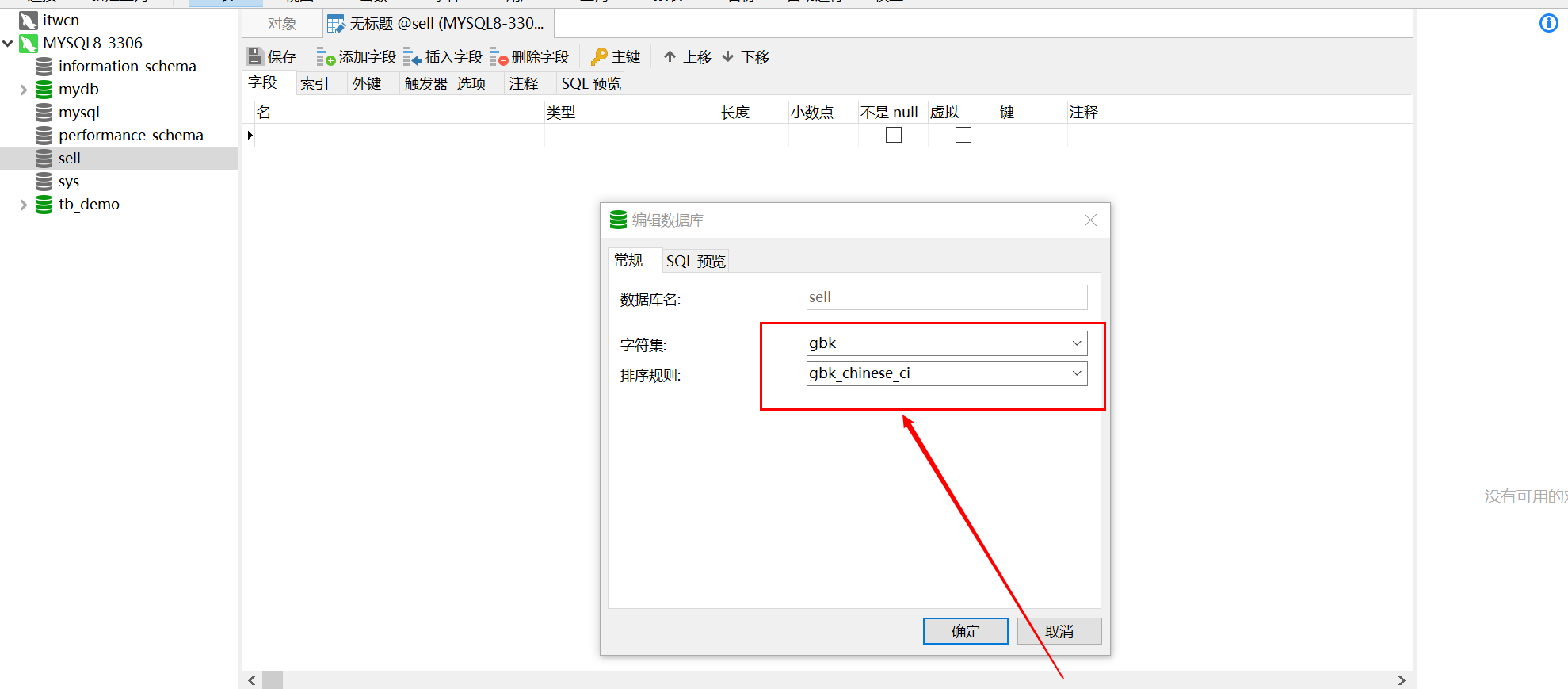
实验题：

1.建立一个商品销售数据库SELL，创建数据库的时候字符集设置为gbk，创建好后，查看SELL数据库目前默认的字符集

建立一个名为SELL的数据库，并设置字符编码方式为gbk



查看创建好的SELL数据库的默认字符集



当前账户所拥有的数据库



2.查看目前账户有哪些数据库

Show database;

3.使用数据库SELL

Use sell;

4.创建 customer（顾客表），sales（订单表），product（产品表），detailed（订单细则表）

信息分别如下：（数据类型可依据存储示例来自己决定)，在SELL库中建立起以上四个表。

顾客表：

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名 | 数据类型 | 含义 | 存储示例 |
| Cid | int(10) | 顾客号 | 例如：20200101（固定长度） |
| Cname | varchar(10) | 顾客名 | 例如：张三 |
| Cphone | char(11) | 电话 | 例如：18482112222 |
| VIP | ENUM(‘S’,’R’,’J’,’O’) | 会员 | 目前会员只有”食品会员“，”日用品会员“，”家居会员"三种，可以同时拥有0-3个会员身份 |

建表语句：

mysql> use sell;

Database changed

mysql> create table customer(

-> Cid int(10) comment '顾客号',

-> Cname varchar(10) comment '顾客名',

-> Cphone char(11) comment '电话',

-> VIP set('食','日','家') comment '会员'

-> );

Query OK, 0 rows affected (0.08 sec)

产品表：

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名 | 数据类型 | 含义 | 存储示例 |
| Pid | varchar(5) | 产品号 | 例如：K1001，长度固定 |
| Pname | varchar(10) | 产品名 | 例如果冻， 最多不超过10个字 |
| Pprice | float(4,2) | 单价 | 例：20.12（精确到小数点后2位） |

建表语句：

mysql> create table products(

-> Pid varchar(5) comment '产品号',

-> Pname varchar(10) comment '产品名',

-> Pprice float(6,2) comment '单价'

-> );

Query OK, 0 rows affected (0.08 sec)

**订单表**

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名 | 数据类型 | 含义 | 存储示例 |
| Oid | char(12) | 订单编号 | 例如：D20200728001，定长 |
| Cid | char | 顾客号 |  |
| Cnumber | int | 订货数量 | 整数 |
| Cotime | DATETIME | 订货日期 | 自动获取下单时间，格式为年月日时分秒 |
| Cdtime | DATE | 交货日期 | 记录交货时间，格式为年月日 |

建表语句：

mysql> create table sales(

-> Oid char(12) comment '订单编号',

-> Cid char comment '顾客号',

-> Cnumber int comment '订货数量',

-> Cotime DATETIME comment '订货日期',

-> Cdtime DATE comment '交货日期'

-> );

Query OK, 0 rows affected (0.07 sec)

订单明细表：

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名 | 数据类型 | 含义 | 存储示例 |
| Oid | char | 订单编号 |  |
| Pid | varchar | 产品号 |  |
| Dnumber | int | 订货数量 | 整数 |
| Dmoney | float(100,2) | 金额 | 精确到小数点后2位 |

建表语句：

mysql> create table detailed(

-> Oid char(12) comment '订单编号',

-> Pid varchar(5) comment '产品号',

-> Dnumber int comment '订货数量',

-> Dmoney decimal(7,2) comment '金额',

-> );

Query OK, 0 rows affected (0.07 sec)