阿里云天池SQL训练营task1

一、初识数据库

1.1 DBMS的种类

1.2 RDBMS的常见系统结构

1.3 数据库安装（必须学习）

1.3.1 阿里云MySQL服务器使用介绍

1.3.2 本地MySQL环境搭建方法介绍

二、初识 SQL

2.1 概念介绍

2.2 SQL的基本书写规则

2.3 数据库的创建（ CREATE DATABASE 语句）

2.4 表的创建（ CREATE TABLE 语句）

2.5 命名规则

2.6 数据类型的指定

2.7 约束的设置

2.8 表的删除和更新

2.9 向 product 表中插入数据

三、练习题

3.1

3.2

3.3

3.4

目录

一、初识数据库

1.1 DBMS的种类

1.2 RDBMS的常见系统结构

1.3 数据库安装

1.3.1 阿里云MySQL服务器使用介绍

1.3.2 本地MySQL环境搭建方法介绍

二、初识 SQL
2.1 概念介绍
2.2 SQL的基本书写规则
2.3 数据库的创建（ CREATE DATABASE 语句）
2.4 表的创建（ CREATE TABLE 语句）
2.5 命名规则
2.6 数据类型的指定
2.7 约束的设置
2.8 表的删除和更新
2.9 向 product 表中插入数据
三、练习题

一、初识数据库

数据库是将大量数据保存起来，通过计算机加工而成的可以 进行高效访问的数据集合。该数据集合称为数据库（Database，DB）。用来管理数据库的计算机系统称为数据库管理系统（Database Management System，DBMS）。

1.1 DBMS的种类

DBMS 主要通过数据的保存格式（数据库的种类）来进行分类，现阶段主要有以下 5 种类型.

- 层次数据库（Hierarchical Database，HDB）
- 关系数据库（Relational Database，RDB）

这种类型的 DBMS 称为关系数据库管理系统（Relational Database Management System，RDBMS）。比较具有代表性的 RDBMS 有如下 5 种。

* Oracle Database：甲骨文公司的RDBMS

* SQL Server：微软公司的RDBMS

* DB2：IBM公司的RDBMS

* PostgreSQL：开源的RDBMS

* MySQL：开源的RDBMS

- 面向对象数据库（Object Oriented Database，OODB）
- XML数据库（XML Database，XMLDB）

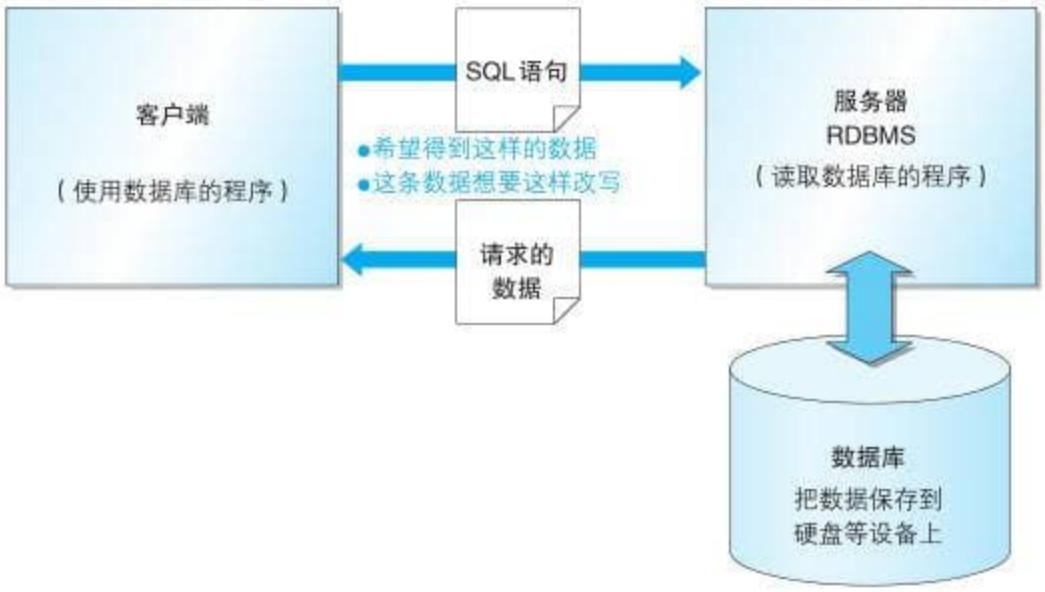
- 键值存储系统（Key-Value Store，KVS），举例：MongoDB

本课程将向大家介绍使用 SQL 语言的数据库管理系统，也就是关系数据库管理系统（RDBMS）的操作方法。

1.2 RDBMS的常见系统结构

使用 RDBMS 时，最常见的系统结构就是客户端 / 服务器类型（C/S类型）这种结构（图 1-3）

图1-3　使用RDBMS时的系统结构



1.3 数据库安装（必须学习）

本次学习大家可以选择使用阿里云数据库服务器或者本地安装数据库进行学习，在下面对应的学习教程中也给告诉了大家如何创建本次学习需要的数据库表和数据，所以大家必须使用一个方式安装数据库，才能完成后面学习。

1.3.1 阿里云MySQL服务器使用介绍

节约篇幅，具体相关介绍以及给大家写到pdf里了，大家点击链接即可进入查看：
http://tianchi-media.oss-cn-beijing.aliyuncs.com/dragonball/SQL/other/阿里云MySQL服务器使用介绍.pdf
优点： 操作使用方便，未来趋势（数据上云），导入、导出数据方便，运行速度快。
缺点： 需要付费购买，不过现在对开发者有优惠活动，基础版本 1核1G，存储空间20G的，目前优惠价半年只需9.9元，一杯奶茶钱不到。

1.3.2 本地MySQL环境搭建方法介绍

节约篇幅，具体相关介绍以及给大家写到pdf里了，大家点击链接即可进入查看：
http://tianchi-media.oss-cn-beijing.aliyuncs.com/dragonball/SQL/other/本地MySQL环境搭建方法介绍.pdf

优点： 免费，增强动手能力。

缺点： 安装、配置麻烦，数据导入、导出耗时长。

二、初识 SQL

2.1 概念介绍

图1-6　表的示例（商品表）

商品编号	商品名称	商品种类	销售单价	进货单价	登记日期
0001	T恤衫	衣服	1000	500	2009-09-20
0002	打孔器	办公用品	500	320	2009-09-11
0003	运动T恤	衣服	4000	2800	
0004	菜刀	厨房用具	3000	2800	2009-09-20
0005	高压锅	厨房用具	6800	5000	2009-01-15
0006	叉子	厨房用具	500		2009-09-20
0007	擦菜板	厨房用具	880	790	2008-04-28
0008	圆珠笔	办公用品	100		2009-11-11

列名（数据的项目名称）
行（记录）
列（字段）
单元格

数据库中存储的表结构类似于excel中的行和列，在数据库中，行称为**记录**，它相当于一条记录，列称为**字段**，它代表了表中存储的数据项目。

行和列交汇的地方称为单元格，一个单元格中只能输入一条记录。

SQL是为操作数据库而开发的语言。国际标准化组织（ISO）为 SQL 制定了相应的标准，以此为基准的SQL 称为标准 SQL（相关信息请参考专栏——标准 SQL 和特定的 SQL）。

完全基于标准 SQL 的 RDBMS 很少，通常需要根据不同的 RDBMS 来编写特定的 SQL 语句，原则上，本课程介绍的是标准 SQL 的书写方式。

根据对 RDBMS 赋予的指令种类的不同，SQL 语句可以分为以下三类.

- DDL

DDL（Data Definition Language，数据定义语言） 用来创建或者删除存储数据用的数据库以及数据库中的表等对象。DDL 包含以下几种指令。

- CREATE： 创建数据库和表等对象
- DROP： 删除数据库和表等对象
- ALTER： 修改数据库和表等对象的结构
- DML

DML（Data Manipulation Language，数据操纵语言） 用来查询或者变更表中的记录。DML 包含以下几种指令。

- SELECT： 查询表中的数据
- INSERT： 向表中插入新数据
- UPDATE： 更新表中的数据
- DELETE： 删除表中的数据
- DCL

DCL（Data Control Language，数据控制语言） 用来确认或者取消对数据库中的数据进行的变更。除此之外，还可以对 RDBMS 的用户是否有权限操作数据库中的对象（数据库表等）进行设定。DCL 包含以下几种指令。

- COMMIT： 确认对数据库中的数据进行的变更
- ROLLBACK： 取消对数据库中的数据进行的变更
- GRANT： 赋予用户操作权限
- REVOKE： 取消用户的操作权限

实际使用的 SQL 语句当中有 90% 属于 DML，本课程会以 DML 为中心进行讲解。

2.2 SQL的基本书写规则

- SQL语句要以分号（;）结尾
- SQL 不区分关键字的大小写，但是插入到表中的数据是区分大小写的
- win 系统默认不区分表名及字段名的大小写
- linux / mac 默认严格区分表名及字段名的大小写
- 本教程已统一调整表名及字段名的为小写，以方便初学者学习使用。
- 常数的书写方式是固定的

'abc', 1234, '26 Jan 2010', '10/01/26', '2010-01-26'…

- 单词需要用半角空格或者换行来分隔

SQL 语句的单词之间需使用半角空格或换行符来进行分隔，且不能使用全角空格作为单词的分隔符，否则会发生错误，出现无法预期的结果。

请大家认真查阅《附录1 – SQL 语法规范》，养成规范的书写习惯。

2.3 数据库的创建（CREATE DATABASE 语句）

语法：

CREATE DATABASE < 数据库名称 > ;

创建本课程使用的数据库

```SQL
CREATE DATABASE shop;
```

2.4 表的创建（CREATE TABLE 语句）

语法：

```SQL
CREATE TABLE < 表名 >
( < 列名 1> < 数据类型 > < 该列所需约束 > ,
 < 列名 2> < 数据类型 > < 该列所需约束 > ,
 < 列名 3> < 数据类型 > < 该列所需约束 > ,
 < 列名 4> < 数据类型 > < 该列所需约束 > ,
 < 该表的约束 1> , < 该表的约束 2> ,……);
```

创建本课程用到的商品表

```SQL
CREATE TABLE product(
  product_id CHAR(4) NOT NULL,
  product_name VARCHAR(100) NOT NULL,
  product_type VARCHAR(32) NOT NULL,
  sale_price INTEGER,
  purchase_price INTEGER,
  regist_date DATE,
  PRIMARY KEY(product_id)
)  ;
```

2.5 命名规则

- 只能使用半角英文字母、数字、下划线（_）作为数据库、表和列的名称

- 名称必须以半角英文字母开头

表1–3 商品表和 product 表列名的对应关系

商品表中的列名	Product表定义的列名
商品编号	product_id
商品名称	product_name
商品种类	product_type
销售单价	sale_price
进货单价	purchase_price
登记日期	regist_date

2.6 数据类型的指定

数据库创建的表，所有的列都必须指定数据类型，每一列都不能存储与该列数据类型不符的数据。

四种最基本的数据类型

- INTEGER 型

用来指定存储整数的列的数据类型（数字型），不能存储小数。

- CHAR 型

用来存储定长字符串，当列中存储的字符串长度达不到最大长度的时候，使用半角空格进行补足，由于会浪费存储空间，所以一般不使用。

- VARCHAR 型

用来存储可变长度字符串，定长字符串在字符数未达到最大长度时会用半角空格补足，但可变长字符串不同，即使字符数未达到最大长度，也不会用半角空格补足。

- DATE 型

用来指定存储日期（年月日）的列的数据类型（日期型）。

2.7 约束的设置

约束是除了数据类型之外，对列中存储的数据进行限制或者追加条件的功能。

NOT NULL是非空约束，即该列必须输入数据。

PRIMARY KEY是主键约束，代表该列是唯一值，可以通过该列取出特定的行的数据。

2.8 表的删除和更新

- 删除表的语法：

DROP TABLE < 表名 > ;

- 删除 product 表

需要特别注意的是，删除的表是无法恢复的，只能重新插入，请执行删除操作时无比要谨慎。

DROP TABLE product;

- 添加列的 ALTER TABLE 语句

ALTER TABLE < 表名 > ADD COLUMN < 列的定义 >;

- 添加一列可以存储100位的可变长字符串的 product_name_pinyin 列

ALTER TABLE product ADD COLUMN product_name_pinyin VARCHAR(100);

- 删除列的 ALTER TABLE 语句

ALTER TABLE < 表名 > DROP COLUMN < 列名 >;

- 删除 product_name_pinyin 列

ALTER TABLE product DROP COLUMN product_name_pinyin;

ALTER TABLE 语句和 DROP TABLE 语句一样，执行之后无法恢复。误添的列可以通过 ALTER TABLE 语句删除，或者将表全部删除之后重新再创建。
【扩展内容】

- 清空表内容

```sql
TRUNCATE TABLE TABLE_NAME;
```

优点：相比**drop``/``delete，truncate**用来清除数据时，速度最快。

- 数据的更新

基本语法：

```sql
UPDATE <表名>
SET <列名> = <表达式> [, <列名2>=<表达式2>...];
WHERE <条件>;  -- 可选，非常重要。
ORDER BY 子句;  --可选
LIMIT 子句; --可选
```

使用 update 时要注意添加 where 条件，否则将会将所有的行按照语句修改

```sql
-- 修改所有的注册时间
UPDATE product
  SET regist_date = '2009-10-10';
-- 仅修改部分商品的单价
UPDATE product
  SET sale_price = sale_price * 10
 WHERE product_type = '厨房用具';
```

使用 UPDATE 也可以将列更新为 NULL（该更新俗称为NULL清空）。此时只需要将赋值表达式右边的值直接写为 NULL 即可。

```sql
-- 将商品编号为0008的数据（圆珠笔）的登记日期更新为NULL
UPDATE product
  SET regist_date = NULL
 WHERE product_id = '0008';
```

和 INSERT 语句一样， UPDATE 语句也可以将 NULL 作为一个值来使用。

但是，只有未设置 NOT NULL 约束和主键约束的列才可以清空为NULL。如果将设置了上述约束的列更新为 NULL，就会出错，这点与INSERT 语句相同。

多列更新

UPDATE 语句的 SET 子句支持同时将多个列作为更新对象。

```sql
-- 基础写法，一条UPDATE语句只更新一列
UPDATE product
  SET sale_price = sale_price * 10
 WHERE product_type = '厨房用具';
UPDATE product
  SET purchase_price = purchase_price / 2
 WHERE product_type = '厨房用具';
```

该写法可以得到正确结果，但是代码较为繁琐。可以采用合并的方法来简化代码。

```sql
-- 合并后的写法
UPDATE product
  SET sale_price = sale_price * 10,
      purchase_price = purchase_price / 2
 WHERE product_type = '厨房用具';
```

需要明确的是，SET 子句中的列不仅可以是两列，还可以是三列或者更多。

2.9 向 product 表中插入数据

为了学习INSERT语句用法，我们首先创建一个名为productins的表，建表语句如下：

```sql
CREATE TABLE productins
(product_id    CHAR(4)      NOT NULL,
 product_name   VARCHAR(100) NOT NULL,
 product_type   VARCHAR(32)  NOT NULL,
 sale_price     INTEGER      DEFAULT 0,
 purchase_price INTEGER ,
 regist_date    DATE ,
 PRIMARY KEY (product_id));
```

基本语法：

INSERT INTO <表名> (列1, 列2, 列3, ……) VALUES (值1, 值2, 值3, ……);

对表进行全列 INSERT 时，可以省略表名后的列清单。这时 VALUES子句的值会默认按照从左到右的顺序赋给每一列。

SQL 复制代码

```sql
-- 包含列清单
INSERT INTO productins (product_id, product_name, product_type, sale_price, purchase_price, regist_date) VALUES ('0005', '高压锅', '厨房用具', 6800, 5000, '2009-01-15');
-- 省略列清单
INSERT INTO productins  VALUES ('0005', '高压锅', '厨房用具', 6800, 5000, '2009-01-15');
```

原则上，执行一次 INSERT 语句会插入一行数据。插入多行时，通常需要循环执行相应次数的 INSERT 语句。其实很多 RDBMS 都支持一次插入多行数据

SQL 复制代码

```sql
-- 通常的INSERT
INSERT INTO productins VALUES ('0002', '打孔器',  '办公用品', 500, 320, '2009-09-11'); INSERT INTO productins VALUES ('0003', '运动T恤',  '衣服', 4000, 2800, NULL); INSERT INTO productins VALUES ('0004', '菜刀',  '厨房用具', 3000, 2800, '2009-09-20');
-- 多行INSERT （ DB2、SQL、SQL Server、 PostgreSQL 和 MySQL多行插入）
INSERT INTO productins VALUES ('0002', '打孔器',  '办公用品', 500, 320, '2009-09-11'), ('0003', '运动T恤', '衣服', 4000, 2800, NULL), ('0004', '菜刀',  '厨房用具', 3000, 2800, '2009-09-20');   -- Oracle中的多行INSERT
INSERT ALL INTO productins VALUES ('0002', '打孔器', '办公用品', 500, 320, '2009-09-11') INTO productins VALUES ('0003', '运动T恤', '衣服', 4000, 2800, NULL) INTO productins VALUES ('0004', '菜刀', '厨房用具', 3000, 2800, '2009-09-20') SELECT * FROM DUAL;
-- DUAL是Oracle特有（安装时的必选项）的一种临时表A。因此“SELECT *FROM DUAL” 部分也只是临时性的，并没有实际意义。
```

INSERT 语句中想给某一列赋予 NULL 值时，可以直接在 VALUES子句的值清单中写入 NULL。想要插入 NULL 的列一定不能设置 NOT NULL 约束。

```sql
INSERT INTO productins (product_id, product_name, product_type,
sale_price, purchase_price, regist_date) VALUES ('0006', '叉子',  '厨房用具', 500, NULL, '2009-09-20');
```

还可以向表中插入默认值（初始值）。可以通过在创建表的CREATE TABLE 语句中设置DEFAULT约束来设定默认值。

```sql
CREATE TABLE productins
(product_id CHAR(4) NOT NULL,
  (略)
 sale_price INTEGER
  (略)   DEFAULT 0, -- 销售单价的默认值设定为0;
 PRIMARY KEY (product_id));
```

可以使用INSERT … SELECT 语句从其他表复制数据。

```sql
-- 将商品表中的数据复制到商品复制表中
INSERT INTO productocpy (product_id, product_name, product_type,
sale_price, purchase_price, regist_date)
SELECT product_id, product_name, product_type, sale_price,
purchase_price, regist_date
FROM Product;
```

- 本课程用表插入数据sql如下：

```sql
-- DML ： 插入数据
STARTTRANSACTION;
INSERT INTO product VALUES('0001', 'T恤衫', '衣服', 1000, 500, '2009-09-20'); INSERT INTO product VALUES('0002', '打孔器', '办公用品', 500, 320, '2009-09-11');
INSERT INTO product VALUES('0003', '运动T恤', '衣服', 4000, 2800, NULL);
INSERT INTO product VALUES('0004', '菜刀', '厨房用具', 3000, 2800, '2009-09-20');
INSERT INTO product VALUES('0005', '高压锅', '厨房用具', 6800, 5000, '2009-01-15');
INSERT INTO product VALUES('0006', '叉子', '厨房用具', 500, NULL, '2009-09-20');
INSERT INTO product VALUES('0007', '擦菜板', '厨房用具', 880, 790, '2008-04-28');
INSERT INTO product VALUES('0008', '圆珠笔', '办公用品', 100, NULL, '2009-11-11');
COMMIT;
```

三、练习题

3.1

编写一条 CREATE TABLE 语句，用来创建一个包含表 1-A 中所列各项的表 Addressbook （地址簿），并为 regist_no （注册编号）列设置主键约束

表1-A 表 Addressbook （地址簿）中的列

列的含义	列的名称	数据类型	约束
注册编号	regist_no	整数型	不能为NULL、主键
姓名	name	可变长字符串类型(长度为128)	不能为NULL
住址	address	可变长字符串类型(长度为256)	不能为NULL
电话号码	tel_no	定长字符串类型(长度为10)	
邮箱地址	mail_address	定长字符串类型(长度为20)	

SQL | 复制代码

```sql
CREATE TABLE Addressbook
 (
 regist_no INTEGER NOT NULL,
 name VARCHAR(128) NOT NULL,
 address VARCHAR(256) NOT NULL,
 tel_no CHAR(10) ,
 mail_address CHAR(20) ,
 PRIMARY KEY (regist_no));
```

3.2

假设在创建练习1.1中的 Addressbook 表时忘记添加如下一列 postal_code （邮政编码）了，请把此列添加到 Addressbook 表中。

列名： postal_code

数据类型： 定长字符串类型（长度为 8）

约束： 不能为 NULL

```sql
-- [Oracle]
ALTER TABLE Addressbook ADD (postal_code CHAR(8)) NOT NULL;
-- [SQL Server]
ALTER TABLE Addressbook ADD postal_code CHAR(8) NOT NULL;
/*
 [DB2] 无法添加。
在DB2中，如果要为添加的列设定NOT NULL约束，
需要像下面这样指定默认值，或者删除NOT NULL约束，
否则就无法添加新列。
*/
-- [DB2 修正版]
ALTER TABLE Addressbook ADD COLUMN postal_code CHAR(8) NOT NULL DEFAULT
'0000-
000';
```

3.3

编写 SQL 语句来删除 Addressbook 表。

```sql
DROP TALBE Addressbook;
```

3.4

编写 SQL 语句来恢复删除掉的 Addressbook 表。

删除后无法使用命令恢复，需要重新创建。

本教程相关练习题答案可以在天池官方公众号：**天池大数据科研平台** 回复：SQL训练营 获取。