# SQL语句查询

## 排序

通过order by语句，可以将查询出的结果进行排序。放置在select语句的最后。

格式:

SELECT \* FROM 表名 ORDER BY 排序字段ASC|DESC;

ASC 升序 (默认)

DESC 降序

#1.查询所有商品信息,使用价格排序(降序)

SELECT \* FROM product ORDER BY price DESC;

#2.在价格排序(降序)的基础上，以分类排序(降序)

SELECT \* FROM product ORDER BY price DESC,category\_id DESC;

#3.显示商品的价格(去重复)，并排序(降序)

SELECT DISTINCT price FROM product ORDER BY price DESC;

## 聚合

之前我们做的查询都是横向查询，它们都是根据条件一行一行的进行判断，而使用聚合函数查询是纵向查询，他是对查询后的结果的列进行计算，然后返回一个单一的值；另外聚合函数会忽略空值。

今天我们学习如下五个聚合函数：

* count：统计指定列不为NULL的记录行数；
* sum：计算指定列的数值和，如果指定列类型不是数值类型，那么计算结果为0；
* max：计算指定列的最大值，如果指定列是字符串类型，那么使用字符串排序运算；
* min：计算指定列的最小值，如果指定列是字符串类型，那么使用字符串排序运算；
* avg：计算指定列的平均值，如果指定列类型不是数值类型，那么计算结果为0；

#1 查询商品的总条数

SELECT COUNT(\*) FROM product;

#2 查询价格大于200商品的总条数

SELECT COUNT(\*) FROM product WHERE price > 200;

#3 查询分类为'c001'的所有商品价格的总和

SELECT SUM(price) FROM product WHERE category\_id = 'c001';

#4 查询分类为'c002'所有商品的平均价格

SELECT AVG(price) FROM product WHERE category\_id = 'c002';

#5 查询商品的最大价格和最小价格

SELECT MAX(price),MIN(price) FROM product;

## 分组

分组查询是指使用group by字句对查询信息进行分组。

* 格式：

SELECT 字段1,字段2… FROM 表名GROUP BY分组字段 HAVING 分组条件;

分组操作中的having子语句，是用于在分组后对数据进行过滤的，作用类似于where条件。

* having与where的区别:
  + having是在分组后对数据进行过滤.

where是在分组前对数据进行过滤

* + having后面可以使用统计函数过滤数据

where后面不可以使用统计函数。

#0 统计所有商品的个数

#1 统计各个分类商品的个数

SELECT category\_id ,COUNT(\*) FROM product GROUP BY category\_id ;

#2 统计各个分类商品的个数,且只显示个数大于1的信息

SELECT category\_id ,COUNT(\*) FROM product GROUP BY category\_id HAVING COUNT(\*) > 1;

## 分页查询

分页查询在项目开发中常见，由于数据量很大，显示屏长度有限，因此对数据需要采取分页显示方式。例如数据共有30条，每页显示5条，第一页显示1-5条，第二页显示6-10条。

* 格式：

SELECT 字段1，字段2... FROM 表明 LIMIT M,N

M: 整数，表示从第几条索引开始，计算方式 （当前页-1）\*每页显示条数

N: 整数，表示查询多少条数据

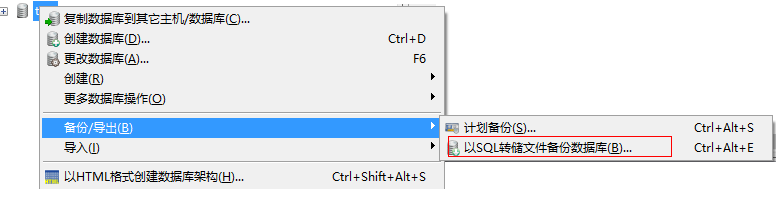
SELECT 字段1，字段2... FROM 表明 LIMIT 0,5

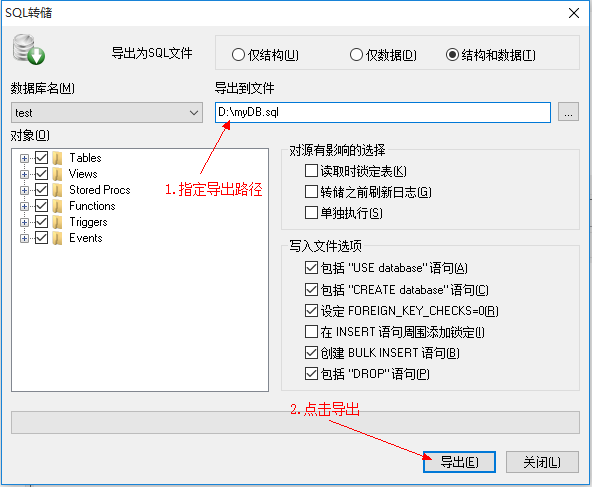
SELECT 字段1，字段2... FROM 表明 LIMIT 5,5

# SQL备份与恢复

## SQL备份

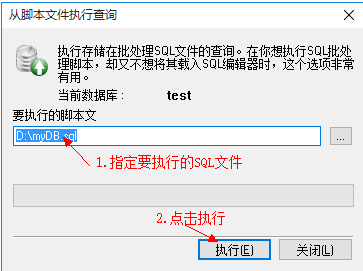
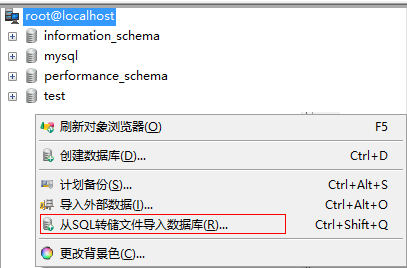
选中数据库，右键”备份/导出”，指定导出路径，保存成.sql文件即可。





## SQL恢复

数据库列表区域右键“从SQL转储文件导入数据库”，指定要执行的SQL文件，执行即可。



# SQL约束

## 数据完整性

添加数据完整性=添加表约束

分类：实体完整性，域完整性，引用完整性

实体完整性： 对数据行的约束，比如:主键约束，唯一约束

域完整性： 对数据列的约束，比如:该列的数据类型, 默认约束，非空约束

引用完整性： 外键约束(多表的关系)

## 主键约束

PRIMARY KEY 约束唯一标识数据库表中的每条记录,每条记录中被主键约束 约束的字段不能相同。

主键必须是唯一的值。

主键列不能是 NULL 值。

每个表都应该有且只能有一个主键。

### 添加主键约束

* 方式一：创建表时，在字段描述处，声明指定字段为主键：

CREATE TABLE Persons

(

Id\_P int PRIMARY KEY,

LastName varchar(255),

FirstName varchar(255),

Address varchar(255),

City varchar(255)

)

* 方式二：创建表时，在constraint约束区域，声明指定字段为主键：
  + 格式：[constraint 名称] primary key (字段列表)
  + 关键字constraint可以省略，如果需要为主键命名，constraint不能省略，主键名称一般没用。
  + 字段列表需要使用小括号括住，如果有多字段需要使用逗号分隔。声明两个以上字段为主键，我们称为联合主键。

CREATE TABLE Persons

(

FirstName varchar(255),

LastName varchar(255),

Address varchar(255),

City varchar(255),

CONSTRAINT pk\_PersonID PRIMARY KEY (FirstName,LastName)

)

或

CREATE TABLE Persons

(

FirstName varchar(255),

LastName varchar(255),

Address varchar(255),

City varchar(255),

PRIMARY KEY (FirstName)

)

* 方式三：创建表之后，通过修改表结构，声明指定字段为主键：

ALTER TABLE Persons ADD [CONSTRAINT 名称] PRIMARY KEY (字段列表)

CREATE TABLE Persons

(

FirstName varchar(255),

LastName varchar(255),

Address varchar(255),

City varchar(255)

)

ALTER TABLE Persons ADD PRIMARY KEY (FirstName)

### 删除主键约束

如需撤销 PRIMARY KEY 约束，请使用下面的 SQL：

ALTER TABLE Persons DROP PRIMARY KEY

## 自动增长列

我们通常希望在每次插入新记录时，数据库自动生成字段的值。

我们可以在表中使用 auto\_increment（自动增长列）关键字，自动增长列类型必须是整形，自动增长列必须为键(一般是主键)。

* 下列 SQL 语句把 "Persons" 表中的 "P\_Id" 列定义为 auto-increment 主键

CREATE TABLE Persons

(

P\_Id int PRIMARY KEY AUTO\_INCREMENT,

LastName varchar(255),

FirstName varchar(255),

Address varchar(255),

City varchar(255)

)

* 向persons添加数据时，可以不为P\_Id字段设置值，也可以设置成null，数据库将自动维护主键值：

INSERT INTO Persons (FirstName,LastName) VALUES ('Bill','Gates')

INSERT INTO Persons (P\_Id,FirstName,LastName) VALUES (NULL,'Bill','Gates')

* 面试：delete和truncate的区别
  + Delete删除表中的数据，但不重置auto-increment记录数。
  + Truncate删除表中的数据，auto-increment记录数将重置。Truncate其实先删除表然后再创建表。
* 扩展：默认地，AUTO\_INCREMENT 的开始值是 1，如果希望修改起始值，请使用下列 SQL 语法：

ALTER TABLE Persons AUTO\_INCREMENT=100

## 非空约束NOT NULL

NOT NULL 约束强制列不接受 NULL 值。

NOT NULL 约束强制字段始终包含值。这意味着，如果不向字段添加值，就无法插入新记录或者更新记录。

* 方式一：创建表，下面的 SQL 语句强制 "Id\_P" 列和 "LastName" 列不接受 NULL 值：

CREATE TABLE Persons

(

Id\_P int NOT NULL,

LastName varchar(255) NOT NULL,

FirstName varchar(255),

Address varchar(255),

City varchar(255)

)

* 方式二：修改表结构

添加非空约束 ALTER TABLE student MODIFY LastName varchar(255) NOT NULL

删除非空约束 ALTER TABLE student MODIFY LastName varchar(255)

## 唯一约束

UNIQUE 约束唯一标识数据库表中的每条记录。

UNIQUE 和 PRIMARY KEY 约束均为列或列集合提供了唯一性的保证。

PRIMARY KEY 拥有自动定义的 UNIQUE 约束。

请注意，每个表可以有多个 UNIQUE 约束，但是每个表只能有一个 PRIMARY KEY 约束。

### 添加唯一约束

与主键添加方式相同，共有3种，

* 方式1：创建表时，在字段描述处，声明唯一：

CREATE TABLE Persons

(

Id\_P int UNIQUE,

LastName varchar(255) NOT NULL,

FirstName varchar(255),

Address varchar(255),

City varchar(255)

)

* 方式2：创建表时，在约束区域，声明唯一：

CREATE TABLE Persons

(

Id\_P int,

LastName varchar(255) NOT NULL,

FirstName varchar(255),

Address varchar(255),

City varchar(255),

CONSTRAINT 名称UNIQUE (Id\_P)

)

* 方式3：创建表后，修改表结构，声明字段唯一：

ALTER TABLE Persons ADD [CONSTRAINT 名称] UNIQUE (Id\_P)

### 删除唯一约束

* 如需撤销 UNIQUE 约束，请使用下面的 SQL：

ALTER TABLE Persons DROP INDEX 约束名称

* 如果添加唯一约束时，没有设置约束名称，默认是当前字段的字段名。

唯一约束与主键约束的区别：

主键：唯一、不能为空、一个表只能有一个主键，非业务数据

唯一：唯一、可以有空值，但只能有一个空值。一个表可以有多个唯一约束。

## 默认约束

在添加数据中，如果该字段不指定值，采用默认值处理

* 方式一： 创建表，字段处声明

CREATE TABLE Persons

(

Id\_P int,

LastName varchar(255) NOT NULL,

FirstName varchar(255),

Address varchar(255) DEFAULT '北京',

City varchar(255)

)

* 方式二： 修改表结构

ALTER TABLE Persons MODIFY Address VARCHAR(255) DEFAULT '北京'

删除默认约束ALTER TABLE Persons MODIFY Address VARCHAR(255)