SQL

> 结构化查询语言
>
> SQL 是用于访问和处理数据库的标准的计算机语言**
>
> 与数据库系统进行交流的语言，通过SQL对数据库进行操作

- SQL 指结构化查询语言
- SQL 使我们有能力访问数据库
- SQL 是一种 ANSI 的标准计算机语言

SQL 能做什么？

- SQL 面向数据库执行查询
- SQL 可从数据库取回数据
- SQL 可在数据库中插入新的记录
- SQL 可更新数据库中的数据
- SQL 可从数据库删除记录
- SQL 可创建新数据库
- SQL 可在数据库中创建新表
- SQL 可在数据库中创建存储过程
- SQL 可在数据库中创建视图
- SQL 可以设置表、存储过程和视图的权限

> 存在着很多不同版本的 SQL 语言
>
> 除了 SQL 标准之外，大部分 SQL 数据库程序都拥有它们自己的私有扩展

在您的网站中使用 SQL

要创建发布数据库中数据的网站，您需要以下要素：

- RDBMS 数据库程序（比如 MS Access, SQL Server, MySQL）
- 服务器端脚本语言（比如 PHP 或 ASP）
- SQL
- HTML / CSS

RDBMS

> 关系数据库管理系统（Relational Database Management System：RDBMS）

RDBMS 是 SQL 的基础，同样也是所有现代数据库系统的基础，比如 MS SQL Server, IBM DB2, Oracle, MySQL 以及 Microsoft Access。

RDBMS 中的数据存储在被称为表（tables）的数据库对象中。

表是相关的数据项的集合，它由列和行组成。

数据库表

一个数据库通常包含一个或多个表。每个表由一个名字标识（例如"客户"或者"订单"）。表包含带有数据的记录（行）。

下面的例子是一个名为 "Persons" 的表：

Id	LastName	FirstName	Address	City
1	Adams	John	Oxford Street	London
2	Bush	George	Fifth Avenue	New York
3	Carter	Thomas	Changan Street	Beijing

上面的表包含三条记录（每一条对应一个人）和五个列（Id、姓、名、地址和城市）。

H2 SQL 语句

> 您需要在数据库上执行的大部分工作都由 SQL 语句完成
>
> SQL对大小写不敏感
>
> 某些数据库系统要求在每条 SQL 命令的末端使用分号，这样就可以在对服务器的相同请求中执行一条以上的语句

```
从表中选取LastName列的数据
SELECT LastName FROM Persons
```

H3 DML DDL

> 把 SQL 分为两个部分：数据操作语言 (DML) 和 数据定义语言 (DDL)
>
> DML（Data Manipulation Language）
>
> DDL (Data Definition Language)

查询和更新指令构成了 SQL 的 DML 部分：

- *SELECT* - 从数据库表中获取数据
- *UPDATE* - 更新数据库表中的数据
- *DELETE* - 从数据库表中删除数据
- *INSERT INTO* - 向数据库表中插入数据

数据定义语言 (DDL) 使我们有能力创建或删除表格。我们也可以定义索引（键），规定表之间的链接，以及施加表间的约束

SQL 中最重要的 DDL 语句:

```
CREATE DATABASE - 创建新数据库
ALTER DATABASE - 修改数据库
CREATE TABLE - 创建新表
ALTER TABLE - 变更（改变）数据库表
DROP TABLE - 删除表
CREATE INDEX - 创建索引（搜索键）
DROP INDEX - 删除索引
```

H2 SELECT 和 SELECT * 语句

> SELECT 语句用于从表中选取数据，结果被存储在一个结果表中（称为结果集）

```
SQL语法
SELECT 列名称 FROM 表名称
或者
SELECT * FROM 表名称

获取名为 "LastName" 和 "FirstName" 的列的内容（从名为 "Persons" 的数据库表）
SELECT LastName,FirstName FROM Persons
```

"Persons" 表:

Id	LastName	FirstName	Address	City
1	Adams	John	Oxford Street	London
2	Bush	George	Fifth Avenue	New York
3	Carter	Thomas	Changan Street	Beijing

结果:

LastName	FirstName
Adams	John
Bush	George
Carter	Thomas

从 "Persons" 表中选取所有的列。

请使用符号 * 取代列的名称，就像这样：

```
星号（*）是选取所有列的快捷方式
SELECT * FROM Persons
```

H3 在结果集（result-set）中导航

由 SQL 查询程序获得的结果被存放在一个结果集中。大多数数据库软件系统都允许使用编程函数在结果集中进行导航，比如：Move-To-First-Record、Get-Record-Content、Move-To-Next-Record 等等。

H2 SQL SELECT DISTINCT 语句

```
关键词 DISTINCT 用于返回唯一不同的值
SELECT DISTINCT 列名称 FROM 表名称

如果要从 "Company" 列中选取所有的值，我们需要使用 SELECT 语句
SELECT Company FROM Orders
```

"Orders"表:

Company	OrderNumber
IBM	3532
W3School	2356
Apple	4698
W3School	6953

结果:

Company
IBM
W3School
Apple
W3School

请注意，在结果集中，W3School 被列出了两次。

如需从 Company" 列中仅选取唯一不同的值，我们需要使用 SELECT DISTINCT 语句:

```
SELECT DISTINCT Company FROM Orders
```

结果:

Company
IBM
W3School
Apple

现在，在结果集中，"W3School" 仅被列出了一次。

H2 SQL WHERE 子句

> WHERE 子句用于规定选择的标准

有条件地从表中选取数据，可将 WHERE 子句添加到 SELECT 语句

```
1  SELECT 列名称 FROM 表名称 WHERE 列 运算符 值;
```

下面的运算符可在 WHERE 子句中使用:

操作符	描述
=	等于
<>	不等于
>	大于
<	小于
>=	大于等于
<=	小于等于
BETWEEN	在某个范围内
LIKE	搜索某种模式

注释: 在某些版本的 SQL 中，操作符 <> 可以写为 !=。

使用 WHERE 子句

如果只希望选取居住在城市 "Beijing" 中的人，我们需要向 SELECT 语句添加 WHERE 子句：

```
SELECT * FROM Persons WHERE City='Beijing'
```

"Persons" 表

LastName	FirstName	Address	City	Year
Adams	John	Oxford Street	London	1970
Bush	George	Fifth Avenue	New York	1975
Carter	Thomas	Changan Street	Beijing	1980
Gates	Bill	Xuanwumen 10	Beijing	1985

结果：

LastName	FirstName	Address	City	Year
Carter	Thomas	Changan Street	Beijing	1980
Gates	Bill	Xuanwumen 10	Beijing	1985

- SQL 使用单引号来环绕 *文本值*（大部分数据库系统也接受双引号）。如果是 *数值*，请不要使用引号

文本值：

```
这是正确的：
SELECT * FROM Persons WHERE FirstName='Bush'

这是错误的：
SELECT * FROM Persons WHERE FirstName=Bush
```

数值：

```
这是正确的：
SELECT * FROM Persons WHERE Year>1965

这是错误的：
SELECT * FROM Persons WHERE Year>'1965'
```

SQL AND & OR 运算符

> AND 和 OR 运算符用于基于一个以上的条件对记录进行过滤

与语言中的 and or 用法相同

LastName	FirstName	Address	City
Adams	John	Oxford Street	London
Bush	George	Fifth Avenue	New York
Carter	Thomas	Changan Street	Beijing
Carter	William	Xuanwumen 10	Beijing

使用 AND 来显示所有姓为 "Carter" 并且名为 "Thomas" 的人：

```
SELECT * FROM Persons WHERE FirstName='Thomas' AND
LastName='Carter'
```

结果：

LastName	FirstName	Address	City
Carter	Thomas	Changan Street	Beijing

or**同理**

H4 结合 AND 和 OR 运算符

我们也可以把 AND 和 OR 结合起来（使用圆括号来组成复杂的表达式）：

```
SELECT * FROM Persons WHERE (FirstName='Thomas' OR
FirstName='William')
AND LastName='Carter'
```

结果：

LastName	FirstName	Address	City
Carter	Thomas	Changan Street	Beijing
Carter	William	Xuanwumen 10	Beijing

H2 SQL ORDER BY 子句

> ORDER BY 语句用于对结果集进行排序
>
> 默认为升序，使用DESC关键字改为降序

原始的表 (用在例子中的)：

Orders 表：

Company	OrderNumber
IBM	3532
W3School	2356
Apple	4698
W3School	6953

以字母顺序显示公司名称：

从Orders表中选中 company，ordernumber列进行操作

```
1  SELECT Company, OrderNumber FROM Orders ORDER BY Company
```

结果：

Company	OrderNumber
Apple	4698
IBM	3532
W3School	6953
W3School	2356

以字母顺序显示公司名称（Company），并以数字顺序显示顺序号（OrderNumber）：

```
1  SELECT Company, OrderNumber FROM Orders ORDER BY Company,
   OrderNumber
```

结果：

Company	OrderNumber
Apple	4698
IBM	3532
W3School	2356
W3School	6953

以逆字母顺序显示公司名称：

```
1  SELECT Company, OrderNumber FROM Orders ORDER BY Company DESC
```

注意：当结果中列出现相同数据。只有这一次，在第一列中有相同的值时，第二列是以升序排列的。如果第一列中有些值为 nulls 时，情况也是这样的。

H2 SQL INSERT INTO 语句

> INSERT INTO 语句用于向表格中插入新的行

H3 语法

```
1  insert 插入      value 值
2  INSERT INTO 表名称 VALUES (值1, 值2,....)
```

我们也可以指定所要插入数据的列：

```
1  table  表
2  INSERT INTO table_name (列1, 列2,...) VALUES (值1, 值2,....)
```

插入新的行

"Persons" 表：

LastName	FirstName	Address	City
Carter	Thomas	Changan Street	Beijing

SQL 语句：

```
INSERT INTO Persons VALUES ('Gates', 'Bill', 'Xuanwumen 10', 'Beijing')
```

结果：

LastName	FirstName	Address	City
Carter	Thomas	Changan Street	Beijing
Gates	Bill	Xuanwumen 10	Beijing

在指定的列中插入数据

"Persons" 表：

LastName	FirstName	Address	City
Carter	Thomas	Changan Street	Beijing
Gates	Bill	Xuanwumen 10	Beijing

SQL 语句：

```
指定数据添加
INSERT INTO Persons (LastName, Address) VALUES ('Wilson', 'Champs-Elysees')
```

结果：

LastName	FirstName	Address	City
Carter	Thomas	Changan Street	Beijing
Gates	Bill	Xuanwumen 10	Beijing
Wilson		Champs-Elysees	

SQL UPDATE 语句

> Update 语句用于修改表中的数据

###

```
UPDATE 表名称 SET 列名称 = 新值 WHERE 列名称 = 某值
```

Person:

LastName	FirstName	Address	City
Gates	Bill	Xuanwumen 10	Beijing
Wilson		Champs-Elysees	

更新某一行中的一个列

我们为 lastname 是 "Wilson" 的人添加 firstname：

```
UPDATE Person SET FirstName = 'Fred' WHERE LastName = 'Wilson'
```

结果：

LastName	FirstName	Address	City
Gates	Bill	Xuanwumen 10	Beijing
Wilson	Fred	Champs-Elysees	

更新某一行中的若干列

我们会修改地址（address），并添加城市名称（city）：

```
UPDATE Person SET Address = 'Zhongshan 23', City = 'Nanjing'
WHERE LastName = 'Wilson'
```

结果：

LastName	FirstName	Address	City
Gates	Bill	Xuanwumen 10	Beijing
Wilson	Fred	Zhongshan 23	Nanjing

SQL DELETE 语句

> DELETE 语句用于删除表中的行

```
DELETE FROM 表名称 WHERE 列名称 = 值
```

Person:

LastName	FirstName	Address	City
Gates	Bill	Xuanwumen 10	Beijing
Wilson	Fred	Zhongshan 23	Nanjing

删除某行

"Fred Wilson" 会被删除：

```
DELETE FROM Person WHERE LastName = 'Wilson'
```

结果:

LastName	FirstName	Address	City
Gates	Bill	Xuanwumen 10	Beijing

删除所有行

可以在不删除表的情况下删除所有的行。这意味着表的结构、属性和索引都是完整的：

```
DELETE FROM table_name
```

或者：

```
DELETE * FROM table_name
```