一些最重要的 SQL 命令

* **SELECT** - 从数据库中提取数据
* **UPDATE** - 更新数据库中的数据
* **DELETE** - 从数据库中删除数据
* **INSERT INTO** - 向数据库中插入新数据
* **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

SELECT:

SELECT \* FROM table\_name;

SELECT CustomerName,City FROM Customers;

在表中，一个列可能会包含多个重复值，有时您也许希望仅仅列出不同（distinct）的值。

SELECT DISTINCT City FROM Customers;

WHERE 子句用于提取那些满足指定标准的记录。

SELECT \* FROM Customers WHERE Country='Mexico';

SELECT \* FROM Customers WHERE CustomerID=1;

SELECT \* FROM Customers WHERE Country='Germany' AND City='Berlin';

SELECT \* FROM Customers WHERE Country='Germany' AND (City='Berlin' OR City='München');

ORDER BY 关键字默认按照升序对记录进行排序。如果需要按照降序对记录进行排序，您可以使用 DESC 关键字。

SELECT \* FROM Customers ORDER BY Country;

SELECT \* FROM Customers ORDER BY Country DESC;

SELECT \* FROM Customers ORDER BY Country,CustomerName;

INSERT INTO 语句可以有两种编写形式。

第一种形式无需指定要插入数据的列名，只需提供被插入的值即可：

INSERT INTO table\_name VALUES (value1,value2,value3,...);

第二种形式需要指定列名及被插入的值：

INSERT INTO *table\_name* (*column1*,*column2*,*column3*,...) VALUES (*value1*,*value2*,*value3*,...);

INSERT INTO Customers (CustomerName, ContactName, Address, City, PostalCode, Country) VALUES ('Cardinal','Tom B. Erichsen','Skagen 21','Stavanger','4006','Norway');

UPDATE 语句用于更新表中已存在的记录。

假设我们要把客户 "Alfreds Futterkiste" 更新为一个新的联系人和城市。

我们使用下面的 SQL 语句：

UPDATE Customers  
SET ContactName='Alfred Schmidt', City='Hamburg'  
WHERE CustomerName='Alfreds Futterkiste';

DELETE 语句用于删除表中的行。

DELETE FROM Customers  
WHERE CustomerName='Alfreds Futterkiste' AND ContactName='Maria Anders';

您可以在不删除表的情况下，删除表中所有的行。这意味着表结构、属性、索引将保持不变：

DELETE FROM *table\_name*;  
or  
DELETE \* FROM *table\_name*;

## SELECT TOP 子句

SELECT TOP 子句用于规定要返回的记录的数目。

SELECT TOP 子句对于拥有数千条记录的大型表来说，是非常有用的。

**注释：**并非所有的数据库系统都支持 SELECT TOP 子句。

## MySQL

SELECT column\_name(s)  
FROM table\_name  
LIMIT number;

LIKE 操作符用于在 WHERE 子句中搜索列中的指定模式。

下面的 SQL 语句选取 City 以字母 "s" 开始的所有客户：

SELECT \* FROM Customers  
WHERE City LIKE 's%';

下面的 SQL 语句选取 City 以字母 "s" 结尾的所有客户：

SELECT \* FROM Customers  
WHERE City LIKE '%s';

## SQL 通配符

在 SQL 中，通配符与 SQL LIKE 操作符一起使用。

SQL 通配符用于搜索表中的数据。

在 SQL 中，可使用一下通配符：

|  |  |
| --- | --- |
| % | 替代 0 个或多个字符 |
| \_ | 替代一个字符 |
| [*charlist*] | 字符列中的任何单一字符 |
| [^*charlist*] or [!*charlist*] | 不在字符列中的任何单一字符 |

下面的 SQL 语句选取 City 以 "b"、"s" 或 "p" 开始的所有客户：

SELECT \* FROM Customers  
WHERE City LIKE '[bsp]%';

下面的 SQL 语句选取 City 以 "a"、"b" 或 "c" 开始的所有客户：

SELECT \* FROM Customers  
WHERE City LIKE '[a-c]%';

下面的 SQL 语句选取 City 不以 "b"、"s" 或 "p" 开始的所有客户

SELECT \* FROM Customers  
WHERE City LIKE '[!bsp]%';

IN 操作符允许您在 WHERE 子句中规定多个值。

SELECT column\_name(s)  
FROM table\_name  
WHERE column\_name IN (value1,value2,...);

BETWEEN 操作符选取介于两个值之间的数据范围内的值。这些值可以是数值、文本或者日期。

SELECT column\_name(s)  
FROM table\_name  
WHERE column\_nameBETWEEN value1 AND value2;

通过使用 SQL，可以为表名称或列名称指定别名（Alias）。

基本上，创建别名是为了让列名称的可读性更强。

SELECT column\_name AS alias\_name  
FROM table\_name;

如果列名称包含空格，要求使用双引号或方括号：

在下面的 SQL 语句中，我们把四个列（Address、City、PostalCode 和 Country）结合在一起，并创建一个名为 "Address" 的别名：

SELECT CustomerName, Address+', '+City+', '+PostalCode+', '+Country AS Address  
FROM Customers;

SQL JOIN 子句用于把来自两个或多个表的行结合起来，基于这些表之间的共同字段。

SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate  
FROM Orders  
INNER JOIN Customers  
ON Orders.CustomerID=Customers.CustomerID;

不同的 SQL JOIN

在我们继续讲解实例之前，我们先列出您可以使用的不同的 SQL JOIN 类型：

* **INNER JOIN**：如果表中有至少一个匹配，则返回行
* **LEFT JOIN**：即使右表中没有匹配，也从左表返回所有的行
* **RIGHT JOIN**：即使左表中没有匹配，也从右表返回所有的行
* **FULL JOIN**：只要其中一个表中存在匹配，则返回行

INNER JOIN 与 JOIN 是相同的。

INNER JOIN 关键字在表中存在至少一个匹配时返回行。如果 "Customers" 表中的行在 "Orders" 中没有匹配，则不会列出这些行。

UNION 操作符用于合并两个或多个 SELECT 语句的结果集。

请注意，UNION 内部的每个 SELECT 语句必须拥有相同数量的列。列也必须拥有相似的数据类型。同时，每个 SELECT 语句中的列的顺序必须相同。

SELECT City FROM Customers  
UNION  
SELECT City FROM Suppliers  
ORDER BY City;

SELECT INTO 语句从一个表复制数据，然后把数据插入到另一个新表中。

只复制德国的客户插入到新表中：

SELECT \*  
INTO CustomersBackup2013  
FROM Customers  
WHERE Country='Germany';