## TOP 子句

TOP 子句用于规定要返回的记录的数目。

对于拥有数千条记录的大型表来说，TOP 子句是非常有用的。

注释：并非所有的数据库系统都支持 TOP 子句。

SQL Server 的语法：

SELECT TOP number|percent column\_name(s) FROM table\_name

MySQL 和 Oracle 中的 SQL SELECT TOP 是等价的

MySQL 语法

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

例子

SELECT \* FROM Persons LIMIT 5

Oracle 语法

SELECT column\_name(s) FROM table\_name WHERE ROWNUM <= number

例子

SELECT \* FROM Persons WHERE ROWNUM <= 5

原始的表 (用在例子中的)：

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 |
| 4 | Obama | Barack | Pennsylvania Avenue | Washington |

SQL TOP 实例

现在，我们希望从上面的 "Persons" 表中选取头两条记录。

我们可以使用下面的 SELECT 语句：

SELECT TOP 2 \* FROM Persons

结果：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | LastName | FirstName | Address | City |
| 1 | Adams | John | Oxford Street | London |
| 2 | Bush | George | Fifth Avenue | New York |

SQL TOP PERCENT 实例

现在，我们希望从上面的 "Persons" 表中选取 50% 的记录。

我们可以使用下面的 SELECT 语句：

SELECT TOP 50 PERCENT \* FROM Persons

结果：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | LastName | FirstName | Address | City |
| 1 | Adams | John | Oxford Street | London |
| 2 | Bush | George | Fifth Avenue | New York |

## LIKE

操作符用于在 WHERE 子句中搜索列中的指定模式。

LIKE 操作符用于在 WHERE 子句中搜索列中的指定模式。

SQL LIKE 操作符语法

SELECT column\_name(s) FROM table\_name WHERE column\_name LIKE pattern

原始的表 (用在例子中的)：

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 |

LIKE 操作符实例

例子 1

现在，我们希望从上面的 "Persons" 表中选取居住在以 "N" 开始的城市里的人：

我们可以使用下面的 SELECT 语句：

SELECT \* FROM Persons WHERE City LIKE 'N%'

提示："%" 可用于定义通配符（模式中缺少的字母）。

结果集：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | LastName | FirstName | Address | City |
| 2 | Bush | George | Fifth Avenue | New York |

例子 2

接下来，我们希望从 "Persons" 表中选取居住在以 "g" 结尾的城市里的人：

我们可以使用下面的 SELECT 语句：

SELECT \* FROM Persons WHERE City LIKE '%g'

结果集：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | LastName | FirstName | Address | City |
| 3 | Carter | Thomas | Changan Street | Beijing |

例子 3

接下来，我们希望从 "Persons" 表中选取居住在包含 "lon" 的城市里的人：

我们可以使用下面的 SELECT 语句：

SELECT \* FROM Persons WHERE City LIKE '%lon%'

结果集：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | LastName | FirstName | Address | City |
| 1 | Adams | John | Oxford Street | London |

例子 4

通过使用 NOT 关键字，我们可以从 "Persons" 表中选取居住在不包含 "lon" 的城市里的人：

我们可以使用下面的 SELECT 语句：

SELECT \* FROM Persons WHERE City NOT LIKE '%lon%'

结果集：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | LastName | FirstName | Address | City |
| 2 | Bush | George | Fifth Avenue | New York |
| 3 | Carter | Thomas | Changan Street | Beijing |

## SQL 通配符

在搜索数据库中的数据时，SQL 通配符可以替代一个或多个字符。

SQL 通配符必须与 LIKE 运算符一起使用。

在 SQL 中，可使用以下通配符：

|  |  |
| --- | --- |
| 通配符 | 描述 |
| % | 替代一个或多个字符 |
| \_ | 仅替代一个字符 |
| [charlist] | 字符列中的任何单一字符 |
| [^charlist]  或者  [!charlist] | 不在字符列中的任何单一字符 |

原始的表 (用在例子中的)：

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 |

使用 % 通配符

例子 1

现在，我们希望从上面的 "Persons" 表中选取居住在以 "Ne" 开始的城市里的人：

我们可以使用下面的 SELECT 语句：

SELECT \* FROM Persons WHERE City LIKE 'Ne%'

结果集：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | LastName | FirstName | Address | City |
| 2 | Bush | George | Fifth Avenue | New York |

例子 2

接下来，我们希望从 "Persons" 表中选取居住在包含 "lond" 的城市里的人：

我们可以使用下面的 SELECT 语句：

SELECT \* FROM Persons WHERE City LIKE '%lond%'

结果集：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | LastName | FirstName | Address | City |
| 1 | Adams | John | Oxford Street | London |

现在，我们希望从上面的 "Persons" 表中选取名字的第一个字符之后是 "eorge" 的人：我们可以使用下面的 SELECT 语句：

SELECT \* FROM Persons WHERE FirstName LIKE '\_eorge'

结果集：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | LastName | FirstName | Address | City |
| 2 | Bush | George | Fifth Avenue | New York |

接下来，我们希望从 "Persons" 表中选取的这条记录的姓氏以 "C" 开头，然后是一个任意字符，然后是 "r"，然后是任意字符，然后是 "er"：

我们可以使用下面的 SELECT 语句：

SELECT \* FROM Persons WHERE LastName LIKE 'C\_r\_er'

结果集：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | LastName | FirstName | Address | City |
| 3 | Carter | Thomas | Changan Street | Beijing |

使用 [charlist] 通配符

现在，我们希望从上面的 "Persons" 表中选取居住的城市以 "A" 或 "L" 或 "N" 开头的人：

SELECT \* FROM Persons WHERE City LIKE '[ALN]%'

结果集：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | LastName | FirstName | Address | City |
| 1 | Adams | John | Oxford Street | London |
| 2 | Bush | George | Fifth Avenue | New York |

现在，我们希望从上面的 "Persons" 表中选取居住的城市不以 "A" 或 "L" 或 "N" 开头的人：我们可以使用下面的 SELECT 语句：

SELECT \* FROM Persons WHERE City LIKE '[!ALN]%'

结果集：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | LastName | FirstName | Address | City |
| 3 | Carter | Thomas | Changan Street | Beijing |

IN 操作符

IN 操作符允许我们在 WHERE 子句中规定多个值。

SQL IN 语法

SELECT column\_name(s) FROM table\_name

WHERE column\_name IN (value1,value2,...)

原始的表 (在实例中使用：)

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 |

IN 操作符实例

现在，我们希望从上表中选取姓氏为 Adams 和 Carter 的人：

我们可以使用下面的 SELECT 语句：

SELECT \* FROM Persons WHERE LastName IN ('Adams','Carter')

结果集：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | LastName | FirstName | Address | City |
| 1 | Adams | John | Oxford Street | London |
| 3 | Carter | Thomas | Changan Street | Beijing |

## BETWEEN 操作符

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

SQL BETWEEN 语法

SELECT column\_name(s) FROM table\_name WHERE column\_name

BETWEEN value1 AND value2

原始的表 (在实例中使用：)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | LastName | FirstName | Address | City |
| 1 | Adams | John | Oxford Street | London |
| 2 | Bush | George | Fifth Avenue | New York |
| 3 | Carter | Thomas | Changan Street | Beijing |
| 4 | Gates | Bill | Xuanwumen 10 | Beijing |

BETWEEN 操作符实例

如需以字母顺序显示介于 "Adams"（包括）和 "Carter"（不包括）之间的人，请使用下面的 SQL：

SELECT \* FROM Persons WHERE LastName BETWEEN 'Adams' AND 'Carter'

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | LastName | FirstName | Address | City |
| 1 | Adams | John | Oxford Street | London |
| 2 | Bush | George | Fifth Avenue | New York |

重要事项：不同的数据库对 BETWEEN...AND 操作符的处理方式是有差异的。某些数据库会列出介于 "Adams" 和 "Carter" 之间的人，但不包括 "Adams" 和 "Carter" ；某些数据库会列出介于 "Adams" 和 "Carter" 之间并包括 "Adams" 和 "Carter" 的人；而另一些数据库会列出介于 "Adams" 和 "Carter" 之间的人，包括 "Adams" ，但不包括 "Carter" 。

所以，请检查你的数据库是如何处理 BETWEEN....AND 操作符的！

（我的是前后都包括）

如需使用上面的例子显示范围之外的人，请使用 NOT 操作符：

SELECT \* FROM Persons WHERE LastName NOT BETWEEN 'Adams' AND 'Carter'

结果集：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | LastName | FirstName | Address | City |
| 3 | Carter | Thomas | Changan Street | Beijing |
| 4 | Gates | Bill | Xuanwumen 10 | Beijing |

## SQL Alias

表的 SQL Alias 语法

SELECT column\_name(s) FROM table\_name AS alias\_name

列的 SQL Alias 语法

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

Alias 实例: 使用表名称别名

假设我们有两个表分别是："Persons" 和 "Product\_Orders"。我们分别为它们指定别名 "p" 和 "po"。

现在，我们希望列出 "John Adams" 的所有定单。

我们可以使用下面的 SELECT 语句：

SELECT po.OrderID, p.LastName, p.FirstName FROM Persons AS p, Product\_Orders AS po WHERE p.LastName='Adams' AND p.FirstName='John'

不使用别名的 SELECT 语句：

SELECT Product\_Orders.OrderID, Persons.LastName, Persons.FirstName

FROM Persons, Product\_Orders

WHERE Persons.LastName='Adams' AND Persons.FirstName='John'

从上面两条 SELECT 语句您可以看到，别名使查询程序更易阅读和书写。

Alias 实例: 使用一个列名别名

表 Persons:

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

SQL:

SELECT LastName AS Family, FirstName AS Name FROM Persons

结果：

|  |  |
| --- | --- |
| Family | Name |
| Adams | John |
| Bush | George |
| Carter | Thomas |

## SQL join 用于根据两个或多个表中的列之间的关系，从这些表中查询数据。

Join 和 Key

有时为了得到完整的结果，我们需要从两个或更多的表中获取结果。我们就需要执行 join。

数据库中的表可通过键将彼此联系起来。主键（Primary Key）是一个列，在这个列中的每一行的值都是唯一的。在表中，每个主键的值都是唯一的。这样做的目的是在不重复每个表中的所有数据的情况下，把表间的数据交叉捆绑在一起。

请看 "Persons" 表：

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

请注意，"Id\_P" 列是 Persons 表中的的主键。这意味着没有两行能够拥有相同的 Id\_P。即使两个人的姓名完全相同，Id\_P 也可以区分他们。

接下来请看 "Orders" 表：

|  |  |  |
| --- | --- | --- |
| Id\_O | OrderNo | Id\_P |
| 1 | 77895 | 3 |
| 2 | 44678 | 3 |
| 3 | 22456 | 1 |
| 4 | 24562 | 1 |
| 5 | 34764 | 65 |

请注意，"Id\_O" 列是 Orders 表中的的主键，同时，"Orders" 表中的 "Id\_P" 列用于引用 "Persons" 表中的人，而无需使用他们的确切姓名。

请留意，"Id\_P" 列把上面的两个表联系了起来。

引用两个表

我们可以通过引用两个表的方式，从两个表中获取数据：

谁订购了产品，并且他们订购了什么产品？

SELECT Persons.LastName, Persons.FirstName, Orders.OrderNo

FROM Persons, Orders WHERE Persons.Id\_P = Orders.Id\_P

结果集：

|  |  |  |
| --- | --- | --- |
| LastName | FirstName | OrderNo |
| Adams | John | 22456 |
| Adams | John | 24562 |
| Carter | Thomas | 77895 |
| Carter | Thomas | 44678 |

SQL JOIN - 使用 Join

除了上面的方法，我们也可以使用关键词 JOIN 来从两个表中获取数据。

如果我们希望列出所有人的定购，可以使用下面的 SELECT 语句：

SELECT Persons.LastName, Persons.FirstName, Orders.OrderNo

FROM Persons INNER JOIN Orders ON Persons.Id\_P = Orders.Id\_P

ORDER BY Persons.LastName

结果集：

|  |  |  |
| --- | --- | --- |
| LastName | FirstName | OrderNo |
| Adams | John | 22456 |
| Adams | John | 24562 |
| Carter | Thomas | 77895 |
| Carter | Thomas | 44678 |

不同的 SQL JOIN

除了我们在上面的例子中使用的 INNER JOIN（内连接），我们还可以使用其他几种连接。

下面列出了您可以使用的 JOIN 类型，以及它们之间的差异。

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，就返回行

SQL INNER JOIN 关键字

在表中存在至少一个匹配时INNER JOIN 关键字返回行。INNER JOIN 关键字语法

SELECT column\_name(s) FROM table\_name1 INNER JOIN table\_name2

ON table\_name1.column\_name=table\_name2.column\_name

注释：INNER JOIN 与 JOIN 是相同的。

"Persons" 表：

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

"Orders" 表：

|  |  |  |
| --- | --- | --- |
| Id\_O | OrderNo | Id\_P |
| 1 | 77895 | 3 |
| 2 | 44678 | 3 |
| 3 | 22456 | 1 |
| 4 | 24562 | 1 |
| 5 | 34764 | 65 |

现在，我们希望列出所有人的定购。您可以使用下面的 SELECT 语句：

SELECT Persons.LastName, Persons.FirstName, Orders.OrderNo FROM Persons INNER JOIN Orders ON Persons.Id\_P=Orders.Id\_P ORDER BY Persons.LastName

|  |  |  |
| --- | --- | --- |
| LastName | FirstName | OrderNo |
| Adams | John | 22456 |
| Adams | John | 24562 |
| Carter | Thomas | 77895 |
| Carter | Thomas | 44678 |

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

