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，而like用在Where子句里面。

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

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

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

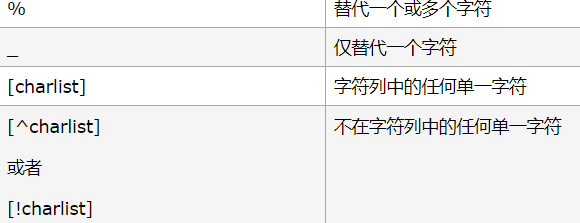
SELECT \* FROM Persons WHERE City NOT LIKE '%lon%'

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

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

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

**选取居住的城市以/不以 "A" 或 "L" 或 "N" 开头的人**



*简单的sql语句*

注意：

1. SQL对大小写不敏感！select = SELECT
2. SQL命令最后的分号，是某些数据库系统要求（MS Access 和 SQL Server 2000，则不必在每条 SQL 语句之后使用分号）的；另外分好是数据库系统中分隔每条SQL语句的标准方法，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

*增删改查*

1. SQL是结构化查询语言（用于执行查询的语法。但是 SQL 语言也包含用于更新、插入和删除记录的语法）、SQL DML是数据操作语言、SQL DDL是数据定义语言（创建和删除表格、定义索引（键）、规定表之间的链接、施加表间约束）

*重要的 DDL语句*



create alter drop

database table index

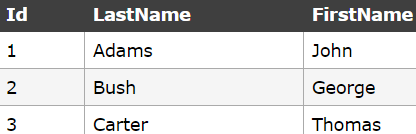
***重要的DML语句：select***

Select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result-set

SELECT LastName,FirstName FROM Persons



SELECT \* FROM Persons



注意：**函数调用访问数据**

在result-set中导航——大多数数据库软件系统，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

***重要的DML语句：select distinct***

表中可能包含重复值，关键词 DISTINCT 用于返回唯一不同的值。

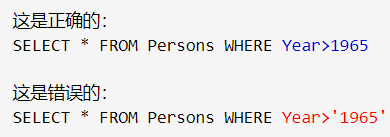
SELECT DISTINCT 列名称 FROM 表名称

***重要的DML语句：select … where***

SELECT 列名称 FROM 表名称 WHERE 列 运算符 值

SELECT \* FROM Persons WHERE City='Beijing'

注意，条件值使用单引号，SQL使用单引号来环绕文本值



***重要的DML语句：select … order by***

Order by语句用于根据指定的列对结果集进行排序（默认升序，降序添加DESC关键字）

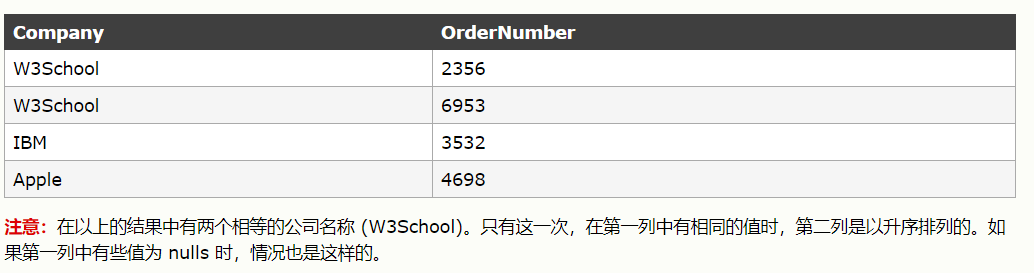
SELECT Company, OrderNumber FROM Orders ORDER BY Company（以字母顺序显示公司名称）

SELECT Company, OrderNumber FROM Orders ORDER BY Company, OrderNumber（以字母顺序显示公司名称（Company），并以数字顺序显示顺序号（OrderNumber））

SELECT Company, OrderNumber FROM Orders ORDER BY Company DESC（按字母倒序排序）

SELECT Company, OrderNumber FROM Orders ORDER BY Company DESC, OrderNumber ASC（以逆字母顺序显示公司名称，并以数字顺序显示顺序号）

注意，如果倒序之后出现相等的名称，或者nulls时：



***重要的DML语句：INSERT INTO***

向表格中插入新的行

INSERT INTO 表名称 VALUES (值1, 值2,....)

INSERT INTO table\_name (列1, 列2,...) VALUES (值1, 值2,....)

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

INSERT INTO Persons (LastName, Address) VALUES ('Wilson', 'Champs-Elysees')

***重要的DML语句：update***

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

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

***重要的DML语句：delete***

DELETE FROM Person WHERE LastName = 'Wilson'

注意：删除所有行（不删除表的情况下删除所有行，即表的结构、属性、索引都是仍完整的）

DELETE FROM table\_name

DELETE \* FROM table\_name

***重要的DML语句：select top***

**TOP 子句用于规定要返回的记录的数目**。这对于拥有数千条记录的大型表来说，TOP 子句是非常有用的。

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

（MySQL，select…limit和select top是等价的）SELECT \* FROM Persons **LIMIT** number

（Oracle，select…rownum和select top是等价的）SELECT column\_name(s) FROM table\_name WHERE **ROWNUM** <= number

***重要的DML语句：select …where…like、select …where…not like***

在where子句中搜索列中的指定模式。

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

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

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

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

***重要的DML语句：select …where…in***

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

***重要的DML语句：select …where…between、select …where…not between***

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

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

***重要的DML语句：select …as***

表的SQL Alias：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

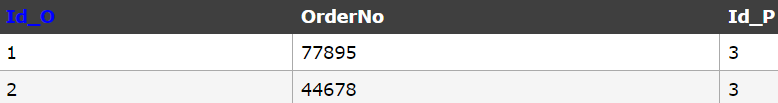
***重要的DML语句：SQL inner join…on（内连接）、left join…on（即使右表没有匹配，也从左表返回的行）、right join…on（即使左表没有匹配，也从右表返回的行）、full join…on（只要其中一个表存在匹配就返回行）***

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**在避免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**。

应用：通过引用多个表，从多个表中获取数据

SELECT Persons.LastName, Persons.FirstName, Orders.OrderNo FROM Persons, Orders WHERE Persons.Id\_P = Orders.Id\_P



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

FROM Persons

**INNER JOIN** Orders

**ON** Persons.Id\_P = Orders.Id\_P

ORDER BY Persons.LastName