SQL 查询语句大全集锦

MYSQL 查询语句大全集锦

1:使用 SHOW 语句找出在服务器上当前存在什么数据库：

mysql> SHOW DATABASES;

2:2、创建一个数据库 MYSQLDATA

mysql> CREATE DATABASE MYSQLDATA;

3:选择你所创建的数据库

mysql> USE MYSQLDATA; (按回车键出现 Database changed 时说明操作成功！)

4:查看现在的数据库中存在什么表

mysql> SHOW TABLES;

5:创建一个数据库表

mysql> CREATE TABLE MYTABLE (name VARCHAR(20), sex CHAR(1));

6:显示表的结构：

mysql> DESCRIBE MYTABLE;

7:往表中加入记录

mysql> insert into MYTABLE values (” hyq”,” M”);

8:用文本方式将数据装入数据库表中（例如 D:/mysql.txt）

mysql> LOAD DATA LOCAL INFILE “D:/mysql.txt” INTO TABLE MYTABLE;

9:导入.sql 文件命令（例如 D:/mysql.sql）

mysql>use database;

mysql>source d:/mysql.sql;

10:删除表

mysql>drop TABLE MYTABLE;

11:清空表

mysql>delete from MYTABLE;

12:更新表中数据

mysql>update MYTABLE set sex=” f” where name=’ hyq’ ;

以下是无意中在网络看到的使用 MySql 的管理心得,

在 windows 中 MySql 以服务形式存在，在使用前应确保此服务已经启动，未启动可用 net start mysql 命令启动。而 Linux 中启动时可用“/etc/rc.d/init.d/mysqld start”命令，注意启动者应具有管理员权限。

刚安装好的 MySql 包含一个含空密码的 root 帐户和一个匿名帐户，这是很大的安全隐患，对于一些重要的应用我们应将安全性尽可能提高，在这里应把匿名帐户删除、 root 帐户设置密码，可用如下命令进行：

use mysql;

delete from User where User=”″;

update User set Password=PASSWORD(’ newpassword’) where User=’root’;

如果要对用户所用的登录终端进行限制，可以更新 User 表中相应用户的 Host 字段，在进行了以上更改后应重新启动数据库服务，此时登录时可用如下类似命令：

mysql -uroot -p;

mysql -uroot -pnewpassword;

mysql mydb -uroot -p;

mysql mydb -uroot -pnewpassword;

上面命令参数是常用参数的一部分，详细情况可参考文档。此处的 mydb 是要登录的数据库的名称。

在 进行开发和实际应用中，用户不应该只用 root 用户进行连接数据库，虽然使用 root 用户进行测试时很方便，但会给系统带来重大安全隐患，也不利于管理技 术的提高。我们给一个应用中使用的用户赋予最恰当的数据库权限。如一个只进行数据插入的用户不应赋予其删除数据的权限。MySql 的用户管理是通过 User 表来实现的，添加新用户常用的方法有两个，一是在 User 表插入相应的数据行，同时设置相应的权限；二是通过 GRANT 命令创建具有某种权限的用 户。其中 GRANT 的常用用法如下：

grant all on mydb.* to NewUserName@HostName identified by “password” ;

grant usage on *.* to NewUserName@HostName identified by “password”;

grant select,insert,update on mydb.* to NewUserName@HostName identified by “password”;

grant update,delete on mydb.TestTable to NewUserName@HostName identified by “password”;

若 要给此用户赋予他在相应对象上的权限的管理能力，可在 GRANT 后面添加 WITH GRANT OPTION 选项。而对于用插入 User 表添加的用户，Password 字段应用 PASSWORD 函数进行更新加密，以防不轨之人窃看密码。对于那些已

经不用的用户应给予清除，权限过界的用户应及时回收权限，回收权限可以通过更新 User 表相应字段， 也可以使用 REVOKE 操作。

下面给出本人从其它资料(www.cn-java.com)获得的对常用权限的解释：

全局管理权限：

FILE: 在 MySQL 服务器上读写文件。

PROCESS: 显示或杀死属于其它用户的服务线程。

RELOAD: 重载访问控制表，刷新日志等。

SHUTDOWN: 关闭 MySQL 服务。

数据库/数据表/数据列权限：

ALTER: 修改已存在的数据表(例如增加/删除列)和索引。

CREATE: 建立新的数据库或数据表。

DELETE: 删除表的记录。

DROP: 删除数据表或数据库。

INDEX: 建立或删除索引。

INSERT: 增加表的记录。

SELECT: 显示/搜索表的记录。

UPDATE: 修改表中已存在的记录。

特别的权限：

ALL: 允许做任何事(和 root 一样)。

USAGE: 只允许登录－其它什么也不允许做。

一、 简单查询

简单的 Transact-SQL 查询只包括选择列表、FROM 子句和 WHERE 子句。它们分别说明所查询列、查询的表或视图、以及搜索条件等。

例如，下面的语句查询 testtable 表中姓名为“张三”的 nickname 字段和 email 字段。

复制内容到剪贴板

代码:SELECT `nickname`,`email`FROM `testtable`WHERE `name`='张三'

(一) 选择列表

选择列表(select_list)指出所查询列，它可以是一组列名列表、星号、表达式、变量(包括局部变量和全局变量)等构成。

1、选择所有列

例如，下面语句显示 testtable 表中所有列的数据：

复制内容到剪贴板

代码:SELECT * FROM testtable

2、选择部分列并指定它们的显示次序

查询结果集合中数据的排列顺序与选择列表中所指定的列名排列顺序相同。

例如：

复制内容到剪贴板

代码:SELECT nickname,email FROM testtable

3、更改列标题

在选择列表中，可重新指定列标题。定义格式为：

列标题=列名

列名 列标题

如果指定的列标题不是标准的标识符格式时，应使用引号定界符，例如，下列语句使用汉字显示列标题：

复制内容到剪贴板

代码:SELECT 昵称=nickname,电子邮件=email FROM testtable

4、删除重复行

SELECT 语句中使用 ALL 或 DISTINCT 选项来显示表中符合条件的所有行或删除其中重复的数据行，默认为 ALL。使用 DISTINCT 选项时，对于所有重复的数据行在 SELECT 返回的结果集合中只保留一行。

5、限制返回的行数

使用 TOP n [PERCENT]选项限制返回的数据行数，TOP n 说明返回 n 行，而 TOP n PERCENT 时，说明 n 是表示一百分数，指定返回的行数等于总行数的百分之几。

例如：

复制内容到剪贴板

代码:SELECT TOP 2 * FROM `testtable`

复制内容到剪贴板

代码:SELECT TOP 20 PERCENT * FROM `testtable`

(二) FROM 子句

FROM 子句指定 SELECT 语句查询及与查询相关的表或视图。在 FROM 子句中最多可指定 256 个表或视图，它们之间用逗号分隔。

在 FROM 子句同时指定多个表或视图时，如果选择列表中存在同名列，这时应使用对象名限定这些列所属的表或视图。例如在 usertable 和 citytable 表中同时存在 cityid 列，在查询两个表中的 cityid 时应使用下面语句格式加以限定：

复制内容到剪贴板

代码:SELECT `username`,citytable.cityid

FROM `usertable`,`citytable`

WHERE usertable.cityid=citytable.cityid 在 FROM 子句中可用以下两种格式为表或视图指定别名：

复制内容到剪贴板

代码:表名 as 别名

表名 别名例如上面语句可用表的别名格式表示为：

复制内容到剪贴板

代码:SELECT `username`,b.cityid

FROM usertable a,citytable b

WHERE a.cityid=b.cityidSELECT 不仅能从表或视图中检索数据，它还能够从其它查询语句所返回的结果集合中查询数据。

例如：

复制内容到剪贴板

代码:SELECT a.au_fname+a.au_lname

FROM authors a,titleauthor ta

(SELECT `title_id`,`title`

FROM `titles`

WHERE ` ytd_sales`>10000

) AS t

WHERE a.au_id=ta.au_id

AND ta.title_id=t.title_id 此例中，将 SELECT 返回的结果集合给予一别名 t，然后再从中检索数据。

(三) 使用 WHERE 子句设置查询条件

WHERE 子句设置查询条件，过滤掉不需要的数据行。例如下面语句查询年龄大于 20 的数据：

复制内容到剪贴板

代码:SELECT * FROM usertable WHERE age>20 WHERE 子句可包括各种条件运算符：

比较运算符(大小比较)：>、>=、=、、!>、!=10 AND age 复制内容到剪贴板

代码:SELECT * FROM `usertable` ORDER BY `age` DESC,`userid` ASC 另外，可以根据表达式进行排序。

二、 联合查询

UNION 运算符可以将两个或两个以上上 SELECT 语句的查询结果集合合并成一个结果集合显示，即执行联合查询。

UNION 的语法格式为：

复制内容到剪贴板

代码:select_statement

UNION [ALL] selectstatement

[UNION [ALL] selectstatement][…n]其中 selectstatement 为待联合的 SELECT 查询语句。

ALL 选项表示将所有行合并到结果集合中。不指定该项时，被联合查询结果集合中的重复行将只保留一行。

联合查询时，查询结果的列标题为第一个查询语句的列标题。因此，要定义列标题必须在第一个查询语句中定义。要对联合查询结果排序时，也必须使用第一查询语句中的列名、列标题或者列序号。

在使用 UNION 运算符时，应保证每个联合查询语句的选择列表中有相同数量的表达式，并且每个查询选择表达式应具有相同的数据类型，或是可以自动将它们转换为相同的数据类型。在自动转换时，对于数值类型，系统将低精度的数据类型转换为高精度的数据类型。

在包括多个查询的 UNION 语句中，其执行顺序是自左至右，使用括号可以改变这一执行顺序。例如：

查询 1 UNION (查询 2 UNION 查询 3)

三、连接查询

通过连接运算符可以实现多个表查询。连接是关系数据库模型的主要特点，也是它区别于其它类型数据库管理系统的一个标志。

在关系数据库管理系统中，表建立时各数据之间的关系不必确定，常把一个实体的所有信息存放在一个表中。当检索数据时，通过连接操作查询出存放在多个表中的不同实体的信息。连接操作给用户带来很大的灵活性，他们可以在任何时候增加新的数据类型。为不同实体创建新的表，尔后通过连接进行查询。

连接可以在 SELECT 语句的 FROM 子句或 WHERE 子句中建立，似是而非在 FROM 子句中指出连接时有助于

将连接操作与 WHERE 子句中的搜索条件区分开来。所以，在 Transact-SQL 中推荐使用这种方法。

SQL-92 标准所定义的 FROM 子句的连接语法格式为：

代码:FROM join_table join_type join_table [ON (join_condition)]其中 join_table 指出参与连接操作的表名，

连接可以对同一个表操作，也可以对多表操作，对同一个表操作的连接又称做自连接。

join_type 指出连接类型，可分为三种：内连接、外连接和交叉连接。

内连接(INNER JOIN)使用比较运算符进行表间某(些)列数据的比较操作，并列出这些表中与连接条件相匹配的数据行。根据所使用的比较方式不同，内连接又分为等值连接、自然连接和不等连接三种。

外连接分为左外连接(LEFT OUTER JOIN 或 LEFT JOIN)、右外连接(RIGHT OUTER JOIN 或 RIGHT JOIN)和全外连接(FULL OUTER JOIN 或 FULL JOIN)三种。与内连接不同的是，外连接不只列出与连接条件相匹配的行，而是列出左表(左外连接时)、右表(右外连接时)或两个表(全外连接时)中所有符合搜索条件的数据行。

交叉连接(CROSS JOIN)没有 WHERE 子句，它返回连接表中所有数据行的笛卡尔积，其结果集合中的数据行数等于第一个表中符合查询条件的数据行数乘以第二个表中符合查询条件的数据行数。

连接操作中的 ON (join_condition) 子句指出连接条件，它由被连接表中的列和比较运算符、逻辑运算符等构成。

无论哪种连接都不能对 text、ntext 和 image 数据类型列进行直接连接，但可以对这三种列进行间接连接。例如：

代码:SELECT p1.pub_id,p2.pub_id,p1.pr_info

FROM pub_info AS p1 INNER JOIN pub_info AS p2

ON DATALENGTH(p1.pr_info)=DATALENGTH(p2.pr_info)(一)内连接

内连接查询操作列出与连接条件匹配的数据行，它使用比较运算符比较被连接列的列值。

内连接分三种：

1、等值连接：在连接条件中使用等于号(=)运算符比较被连接列的列值，其查询结果中列出被连接表中的所有列，包括其中的重复列。

2、不等连接： 在连接条件使用除等于运算符以外的其它比较运算符比较被连接的列的列值。这些运算符包

括>、>=、、!。

3、自然连接：在连接条件中使用等于(=)运算符比较被连接列的列值，但它使用选择列表指出查询结果集合中所包括的列，并删除连接表中的重复列。

例，下面使用等值连接列出 authors 和 publishers 表中位于同一城市的作者和出版社：

代码:SELECT *

FROM authors AS a INNER JOIN publishers AS p

ON a.city=p.city 又如使用自然连接，在选择列表中删除 authors 和 publishers 表中重复列(city 和 state)：

复制内容到剪贴板

代码:SELECT a.*,p.pub_id,p.pub_name,p.country

FROM authors AS a INNER JOIN publishers AS p

ON a.city=p.city

(二)外连接内连接时，返回查询结果集合中的仅是符合查询条件(WHERE 搜索条件或 HAVING 条件)和连接条件的行。而采用外连接时，它返回到查询结果集合中的不仅包含符合连接条件的行，而且还包括左表(左外连接时)、右表(右外连接时)或两个边接表(全外连接)中的所有数据行。

如下面使用左外连接将论坛内容和作者信息连接起来：

代码:SELECT a.*,b.* FROM `luntan` LEFT JOIN usertable as b

ON a.username=b.username 下面使用全外连接将 city 表中的所有作者以及 user 表中的所有作者，以及他们所在的城市：

代码:SELECT a.*,b.*

FROM city as a FULL OUTER JOIN user as b

ON a.username=b.username

(三)交叉连接

交叉连接不带 WHERE 子句，它返回被连接的两个表所有数据行的笛卡尔积，返回到结果集合中的数

据行数等于第一个表中符合查询条件的数据行数乘以第二个表中符合查询条件的数据行数。

例，titles 表中有 6 类图书，而 publishers 表中有 8 家出版社，则下列交叉连接检索到的记录数将等于 6*8=48 行。

代码:SELECT `type`,`pub_name`

FROM `titles` CROSS JOIN `publishers`

ORDER BY `type`

SQL 核心语句(非常实用的几个技巧)

_ArticleContent1_lblContent>插入数据

向表中添加一个新记录，你要使用 SQL INSERT 语句。这里有一个如何使用这种语句的例子：

代码:INSERT mytable (mycolumn) VALUES (‘some data’) 这个语句把字符串’some data’插入表 mytable 的 mycolumn 字段中。将要被插入数据的字段的名字在第一个括号中指定，实际的数据在第二个括号中给出。

INSERT 语句的完整句法如下：

代码:INSERT [INTO] {table_name|view_name} [(column_list)] {DEFAULT VALUES | Values_list | select_statement} 如果一个表有多个字段，通过把字段名和字段值用逗号隔开，你可以向所有的字段中插入数据。假设表 mytable 有三个字段 first_column,second_column,和 third_column。

下面的 INSERT 语句添加了一条三个字段都有值的完整记录：

代码:INSERT mytable (first_column,second_column,third_column)

VALUES (‘some data’,’ some more data’,’ yet more data’) 注意

你可以使用 INSERT 语句向文本型字段中插入数据。但是，如果你需要输入很长的字符串，你应该使用 WRITETEXT 语句。这部分内容对本书来说太高级了，因此不加讨论。要了解更多的信息，请参考 Microsoft SQL Sever 的文档。

如果你在 INSERT 语句中只指定两个字段和数据会怎么样呢？换句话说，你向一个表中插入一条新记录，但有一个字段没有提供数据。在这种情况下，有下面的四种可能：

如果该字段有一个缺省值，该值会被使用。例如，假设你插入新记录时没有给字段 third_column 提供数据，而这

个字段有一个缺省值’some value’。在这种情况下，当新记录建立时会插入值’some value’。

如果该字段可以接受空值，而且没有缺省值，则会被插入空值。

如果该字段不能接受空值，而且没有缺省值，就会出现错误。你会收到错误信息：

The column in table mytable may not be null.

最后，如果该字段是一个标识字段，那么它会自动产生一个新值。当你向一个有标识字段的表中插入新记录时，只要忽略该字段，标识字段会给自己赋一个新值。

注意

向一个有标识字段的表中插入新记录后，你可以用 SQL 变量@@identity 来访问新记录

的标识字段的值。考虑如下的 SQL 语句：

复制内容到剪贴板

代码:INSERT mytable (first_column) VALUES(‘some value’) [code]

[code]INSERT anothertable(another_first,another_second)

VALUES(@@identity,’ some value’) 如果表 mytable 有一个标识字段，该字段的值会被插入表 anothertable 的 another_first 字段。这是因为变量@@identity 总是保存最后一次插入标识字段的值。

字段 another_first 应该与字段 first_column 有相同的数据类型。但是，字段 another_first 不能是应该标识字段。Another_first 字段用来保存字段 first_column 的值。

删除记录

要从表中删除一个或多个记录，需要使用 SQL DELETE 语句。你可以给 DELETE 语句提供 WHERE 子句。WHERE 子句用来选择要删除的记录。例如，下面的这个 DELETE 语句只删除字段 first_column 的值等于’Delete Me’的记录：

代码:DELETE mytable WHERE first_column=’Deltet Me’ DELETE 语句的完整句法如下：

复制内容到剪贴板

代码:DELETE [FROM] {table_name|view_name} [WHERE clause] 在 SQL SELECT 语句中可以使用的任何条件都可以在 DELECT 语句的 WHERE 子句 中使用。例如，下面的这个 DELETE 语句只删除那些 first_column 字段的值

为’goodbye’或 second_column 字段的值为’so long’的记录：

代码:DELETE mytable WHERE first_column=’goodby’ OR second_column=’so long’ 如果你不给 DELETE 语句

提供 WHERE 子句，表中的所有记录都将被删除。你不应该有这种想法。如果你想删除应该表中的所有记录，应使用第十章所讲的 TRUNCATE TABLE 语句。

注意

为什么要用 TRUNCATE TABLE 语句代替 DELETE 语句？当你使用 TRUNCATE TABLE 语句时，记录的删除是不作记录的。也就是说，这意味着 TRUNCATE TABLE 要比 DELETE 快得多。

更新记录

要修改表中已经存在的一条或多条记录，应使用 SQL UPDATE 语句。同 DELETE 语句一样，UPDATE 语句可以使用 WHERE 子句来选择更新特定的记录。请看这个例子：

代码:UPDATE mytable SET first_column=’Updated!’ WHERE second_column=’Update Me!’ 这个 UPDATE 语句更新所有 second_column 字段的值为’Update Me!’的记录。对所有被选中的记录，字段 first_column 的值被置为’Updated!’。

下面是 UPDATE 语句的完整句法：

代码:UPDATE {table_name|view_name} SET [{table_name|view_name}]

{column_list|variable_list|variable_and_column_list}

[,{column_list2|variable_list2|variable_and_column_list2}…

[,{column_listN|variable_listN|variable_and_column_listN}]]

[WHERE clause] 注意

你可以对文本型字段使用 UPDATE 语句。但是，如果你需要更新很长的字符串，应使用 UPDATETEXT 语句。这部分内容对本书来说太高级了，因此不加讨论。要了解更多的信息，请参考 Microsoft SQL Sever 的文档。

如果你不提供 WHERE 子句，表中的所有记录都将被更新。有时这是有用的。例如，如果你想把表 titles 中的所有书的价格加倍，你可以使用如下的 UPDATE 语句：

你也可以同时更新多个字段。例如，下面的 UPDATE 语句同时更新 first_column,second_column,和 third_column 这三个字段：

代码:UPDATE mytable SET first_column=’ Updated!’

Second_column=’ Updated!’

Third_column=’ Updated!’

WHERE first_column=’ Update Me1’ 技巧

SQL 忽略语句中多余的空格。你可以把 SQL 语句写成任何你最容易读的格式。

用 SELECT 创建记录和表

你也许已经注意到，INSERT 语句与 DELETE 语句和 UPDATE 语句有一点不同，它一次只操作一个记录。然而，有一个方法可以使 INSERT 语句一次添加多个记录。要作到这一点，你需要把 INSERT 语句与 SELECT 语句结合起来，象这样：

代码:INSERT mytable (first_column,second_column)

SELECT another_first,another_second

FROM anothertable

WHERE another_first=’ Copy Me!’ 这个语句从 anothertable 拷贝记录到 mytable.只有表 anothertable 中字段 another_first 的值为’ Copy Me！’ 的记录才被拷贝。

当为一个表中的记录建立备份时，这种形式的 INSERT 语句是非常有用的。在删除一个表中的记录之前，你可以先用这种方法把它们拷贝到另一个表中。

如果你需要拷贝整个表，你可以使用 SELECT INTO 语句。例如，下面的语句创建了一个名为 newtable 的新表，该表包含表 mytable 的所有数据：

代码:SELECT * INTO newtable FROM mytable 你也可以指定只有特定的字段被用来创建这个新表。要做到这一点，只需在字段列表中指定你想要拷贝的字段。另外，你可以使用 WHERE 子句来限制拷贝到新表中的记录。下面的例子只拷贝字段 second_columnd 的值等于’ Copy Me!’ 的记录的 first_column 字段。

代码:SELECT first_column INTO newtable

FROM mytable

WHERE second_column=’ Copy Me!’ 使用 SQL 修改已经建立的表是很困难的。例如，如果你向一个表中添加了一

个字段，没有容易的办法来去除它。另外，如果你不小心把一个字段的数据类型给错了，你将没有办法改变它。但是，使用本节中讲述的 SQL 语句，你可以绕过这两个问题。

例如，假设你想从一个表中删除一个字段。使用 SELECT INTO 语句，你可以创建该表的一个拷贝，但不包含要删除的字段。这使你既删除了该字段，又保留了不想删除的数据。

如果你想改变一个字段的数据类型，你可以创建一个包含正确数据类型字段的新表。创建好该表后，你就可以结合使用 UPDATE 语句和 SELECT 语句，把原来表中的所有数据拷贝到新表中。通过这种方法，你既可以修改表的结构，又能保存原有的数据。

_ArticleContent1_lblContent>插入数据

向表中添加一个新记录，你要使用 SQL INSERT 语句。这里有一个如何使用这种语句的例子：

复制内容到剪贴板

代码:INSERT mytable (mycolumn) VALUES (‘some data’) 这个语句把字符串’some data’插入表 mytable 的 mycolumn 字段中。将要被插入数据的字段的名字在第一个括号中指定，实际的数据在第二个括号中给出。

INSERT 语句的完整句法如下：

复制内容到剪贴板

代码:INSERT [INTO] {table_name|view_name} [(column_list)] {DEFAULT VALUES | Values_list | select_statement} 如果一个表有多个字段，通过把字段名和字段值用逗号隔开，你可以向所有的字段中插入数据。假设表 mytable 有三个字段 first_column,second_column,和 third_column。下面的 INSERT 语句添加了一条三个字段都有值的完整记录：

复制内容到剪贴板

代码:INSERT mytable (first_column,second_column,third_column) VALUES (‘some data’,’ some more data’,’ yet more data’)

[code]

注意

你可以使用 INSERT 语句向文本型字段中插入数据。但是，如果你需要输入很长的字符串，你应该使用 WRITETEXT 语句。这部分内容对本书来说太高级了，因此不加讨论。要了解更多的信息，请参考 Microsoft SQL Sever 的文档。

如果你在 INSERT 语句中只指定两个字段和数据会怎么样呢？换句话说，你向一个表中插入一条新记录，但有一个字段没有提供数据。在这种情况下，有下面的四种可能：

如果该字段有一个缺省值，该值会被使用。例如，假设你插入新记录时没有给字段 third_column 提供数据，而这个字段有一个缺省值’some value’。在这种情况下，当新记录建立时会插入值’some value’。

如果该字段可以接受空值，而且没有缺省值，则会被插入空值。

如果该字段不能接受空值，而且没有缺省值，就会出现错误。你会收到错误信息：

The column in table mytable may not be null.

最后，如果该字段是一个标识字段，那么它会自动产生一个新值。当你向一个有标识字段的表中插入新记录时，只要忽略该字段，标识字段会给自己赋一个新值。

注意

向一个有标识字段的表中插入新记录后，你可以用 SQL 变量@@identity 来访问新记录

的标识字段的值。考虑如下的 SQL 语句：

[code]INSERT mytable (first_column) VALUES(‘some value’)

复制内容到剪贴板

代码:INSERT anothertable(another_first,another_second) VALUES(@@identity,’ some value’) 如果表 mytable 有一个标识字段，该字段的值会被插入表 anothertable 的 another_first 字段。这是因为变量@@identity 总是保存最后一次插入标识字段的值。

字段 another_first 应该与字段 first_column 有相同的数据类型。但是，字段 another_first 不能是应该标识字段。Another_first 字段用来保存字段 first_column 的值。

删除记录

要从表中删除一个或多个记录，需要使用 SQL DELETE 语句。你可以给 DELETE 语句提供 WHERE 子句。WHERE 子句用来选择要删除的记录。例如，下面的这个 DELETE 语句只删除字段 first_column 的值等于’Delete Me’的记录：

复制内容到剪贴板

代码:DELETE mytable WHERE first_column=’Deltet Me’ DELETE 语句的完整句法如下：

复制内容到剪贴板

代码:DELETE [FROM] {table_name|view_name} [WHERE clause] 在 SQL SELECT 语句中可以使用的任何条件都可以在 DELECT 语句的 WHERE 子句 中使用。例如，下面的这个 DELETE 语句只删除那些 first_column 字段的值为’goodbye’或 second_column 字段的值为’so long’的记录：

复制内容到剪贴板

代码:DELETE mytable WHERE first_column=’goodby’ OR second_column=’so long’ 如果你不给 DELETE 语句提供 WHERE 子句，表中的所有记录都将被删除。你不应该有这种想法。如果你想删除应该表中的所有记录，应使用第十章所讲的 TRUNCATE TABLE 语句。

注意

为什么要用 TRUNCATE TABLE 语句代替 DELETE 语句？当你使用 TRUNCATE TABLE 语句时，记录的删除是不作记录的。也就是说，这意味着 TRUNCATE TABLE 要比 DELETE 快得多。

更新记录

要修改表中已经存在的一条或多条记录，应使用 SQL UPDATE 语句。同 DELETE 语句一样，UPDATE 语句可以使用 WHERE 子句来选择更新特定的记录。请看这个例子：

复制内容到剪贴板

代码:UPDATE mytable SET first_column=’Updated!’ WHERE second_column=’Update Me!’ 这个 UPDATE 语句更新所有 second_column 字段的值为’Update Me!’的记录。对所有被选中的记录，字段 first_column 的值被置为’Updated!’。

下面是 UPDATE 语句的完整句法：

复制内容到剪贴板

代码:UPDATE {table_name|view_name} SET [{table_name|view_name}]

{column_list|variable_list|variable_and_column_list}

[,{column_list2|variable_list2|variable_and_column_list2}…

[,{column_listN|variable_listN|variable_and_column_listN}]]

[WHERE clause] 注意

你可以对文本型字段使用 UPDATE 语句。但是，如果你需要更新很长的字符串，应使用 UPDATETEXT 语句。这部分内容对本书来说太高级了，因此不加讨论。要了解更多的信息，请参考 Microsoft SQL Sever 的文档。

如果你不提供 WHERE 子句，表中的所有记录都将被更新。有时这是有用的。例如，如果你想把表 titles 中的所有书的价格加倍，你可以使用如下的 UPDATE 语句：

你也可以同时更新多个字段。例如，下面的 UPDATE 语句同时更新 first_column,second_column,和 third_column 这三个字段：

复制内容到剪贴板

代码:UPDATE mytable SET first_column=’ Updated!’

Second_column=’ Updated!’

Third_column=’ Updated!’

WHERE first_column=’ Update Me1’ 技巧

SQL 忽略语句中多余的空格。你可以把 SQL 语句写成任何你最容易读的格式。

用 SELECT 创建记录和表

你也许已经注意到，INSERT 语句与 DELETE 语句和 UPDATE 语句有一点不同，它一次只操作一个记录。然而，有一个方法可以使 INSERT 语句一次添加多个记录。要作到这一点，你需要把 INSERT 语句与 SELECT 语句结合起来，象这样：

复制内容到剪贴板

代码:INSERT mytable (first_column,second_column)

SELECT another_first,another_second

FROM anothertable

WHERE another_first=’ Copy Me!’ 这个语句从 anothertable 拷贝记录到 mytable.只有表 anothertable 中字段 another_first 的值为’ Copy Me！’ 的记录才被拷贝。

当为一个表中的记录建立备份时，这种形式的 INSERT 语句是非常有用的。在删除一个表中的记录之前，你可以先用这种方法把它们拷贝到另一个表中。

如果你需要拷贝整个表，你可以使用 SELECT INTO 语句。例如，下面的语句创建了一个名为 newtable 的新表，该表包含表 mytable 的所有数据：

复制内容到剪贴板

代码:SELECT * INTO newtable FROM mytable 你也可以指定只有特定的字段被用来创建这个新表。要做到这一点，只需在字段列表中指定你想要拷贝的字段。另外，你可以使用 WHERE 子句来限制拷贝到新表中的记录。下面的例子只拷贝字段 second_columnd 的值等于’ Copy Me!’ 的记录的 first_column 字段。

复制内容到剪贴板

代码:SELECT first_column INTO newtable

FROM mytable

WHERE second_column=’ Copy Me!’ 使用 SQL 修改已经建立的表是很困难的。例如，如果你向一个表中添加了一

个字段，没有容易的办法来去除它。另外，如果你不小心把一个字段的数据类型给错了，你将没有办法改变它。但是，使用本节中讲述的 SQL 语句，你可以绕过这两个问题。

例如，假设你想从一个表中删除一个字段。使用 SELECT INTO 语句，你可以创建该表的一个拷贝，但不包含要删除的字段。这使你既删除了该字段，又保留了不想删除的数据。

如果你想改变一个字段的数据类型，你可以创建一个包含正确数据类型字段的新表。创建好该表后，你就可以结合使用 UPDATE 语句和 SELECT 语句，把原来表中的所有数据拷贝到新表中。通过这种方法，你既可以修改表的结构，又能保存原有的数据。

SQL 语法,SQL 语句大全,SQL 基础

ＳＱＬ 语法参考手册(SQL)/数据类型

2006-07-24 07:42

《ＳＱＬ 语法参考手册(SQL)》

DB2 提供了关连式资料库的查询语言 ＳＱＬ (Structured Query Language)，是一种非常口语化、既易学又易懂的语法。此一语言几乎是每个资料库系统都必须提供的，用以表示关连式的*作，包含了资料的定义（ＤＤＬ）以及资料的处理（ＤＭＬ）。SQL 原来拼成 SEQUEL，这语言的原型以"系统 R"的名字在 IBM 圣荷西实验室完成，经过 IBM 内部及其他的许多使用性及效率测试，其结果相当令人满意，并决定在系统 R 的技术基础发展出来 IBM 的产品。而且美国国家标准学会（ANSI）及国际标准化组织（ISO）在 1987 遵循一个几乎是以 IBM SQL 为基础的标准关连式资料语言定义。

一、资料定义 ＤＤＬ（Data Definition Language)

资料定语言是指对资料的格式和形态下定义的语言，他是每个资料库要建立时候时首先要面对的，举凡资料分哪些表格关系、表格内的有什麼栏位主键、表格和表格之间互相参考的关系等等，都是在开始的时候所必须规划好的。

１、建表格：

Create TABLE table_name(
column1 DATATYPE [NOT NULL] [NOT NULL PRIMARY KEY],
column2 DATATYPE [NOT NULL],
...）

说明：

DATATYPE --是资料的格式，详见表。

NUT NULL --可不可以允许资料有空的（尚未有资料填入）。

PRIMARY KEY --是本表的主键。

２、更改表格

Alter TABLE table_name
ADD COLUMN column_name DATATYPE

说明：增加一个栏位（没有删除某个栏位的语法。

Alter TABLE table_name
ADD PRIMARY KEY (column_name)

说明：更改表得的定义把某个栏位设为主键。

Alter TABLE table_name
Drop PRIMARY KEY (column_name)

说明：把主键的定义删除。

３、建立索引

Create INDEX index_name ON table_name (column_name)

说明：对某个表格的栏位建立索引以增加查询时的速度。
４、删除
Drop table_name
Drop index_name
二、的资料形态 DATATYPEs
smallint
16 位元的整数。
interger
32 位元的整数。
decimal(p,s)
p 精确值和 s 大小的十进位整数，精确值 p 是指全部有几个数(digits)大小值，s 是指小数点後有几位数。如果没有特别指定，则系统会设为 p=5; s=0 。
float
32 位元的实数。
double
64 位元的实数。
char(n)
n 长度的字串，n 不能超过 254。
varchar(n)
长度不固定且其最大长度为 n 的字串，n 不能超过 4000。
graphic(n)
和 char(n) 一样，不过其单位是两个字元 double-bytes， n 不能超过 127。这个形态是为了支援两个字元长度的字体，例如中文字。
vargraphic(n)
可变长度且其最大长度为 n 的双字元字串，n 不能超过 2000。
date
包含了 年份、月份、日期。
time
包含了 小时、分钟、秒。
timestamp
包含了 年、月、日、时、分、秒、千分之一秒。
三、资料*作 ＤＭＬ （Data Manipulation Language)
资料定义好之後接下来的就是资料的*作。资料的*作不外乎增加资料（insert)、查询资料（query）、更改资料（update) 、删除资料（delete）四种模式，以下分 别介绍他们的语法：
１、增加资料：
Insert INTO table_name (column1,column2,...)
valueS (value1,value2, ...)
说明：
1.若没有指定 column 系统则会按表格内的栏位顺序填入资料。
2.栏位的资料形态和所填入的资料必须吻合。
3.table_name 也可以是景观 view_name。
Insert INTO table_name (column1,column2,...)
Select columnx,columny,... FROM another_table
说明：也可以经过一个子查询（subquery）把别的表格的资料填入。
２、查询资料：
基本查询
Select column1,columns2,...
FROM table_name
说明：把 table_name 的特定栏位资料全部列出来

Select *
FROM table_name
Where column1 = xxx
[AND column2 〉 yyy] [OR column3 〈〉 zzz]
说明：
1.’*’表示全部的栏位都列出来。
2.Where 之後是接条件式，把符合条件的资料列出来。
Select column1,column2
FROM table_name
orDER BY column2 [DESC]
说明：ORDER BY 是指定以某个栏位做排序，[DESC]是指从大到小排列，若没有指明，则是从小到大排列
组合查询
组合查询是指所查询得资料来源并不只有单一的表格，而是联合一个以上的
表格才能够得到结果的。
Select *
FROM table1,table2
Where table1.colum1=table2.column1
说明：
1.查询两个表格中其中 column1 值相同的资料。
2.当然两个表格相互比较的栏位，其资料形态必须相同。
3.一个复杂的查询其动用到的表格可能会很多个。
整合性的查询：
Select COUNT (*)
FROM table_name
Where column_name = xxx
说明：
查询符合条件的资料共有几笔。
Select SUM(column1)
FROM table_name
说明：
1.计算出总和，所选的栏位必须是可数的数字形态。
2.除此以外还有 AVG() 是计算平均、MAX()、MIN()计算最大最小值的整合性查询。
Select column1,AVG(column2)
FROM table_name
GROUP BY column1
HAVING AVG(column2) 〉 xxx
说明：
1.GROUP BY: 以 column1 为一组计算 column2 的平均值必须和 AVG、SUM 等整合性查询的关键字一起使用。
2.HAVING : 必须和 GROUP BY 一起使用作为整合性的限制。
复合性的查询
Select *
FROM table_name1
Where EXISTS (
Select *
FROM table_name2
Where conditions)
说明：

1.Where 的 conditions 可以是另外一个的 query。
2.EXISTS 在此是指存在与否。
Select *
FROM table_name1
Where column1 IN (
Select column1
FROM table_name2
Where conditions)
说明：
1. IN 後面接的是一个集合，表示 column1 存在集合里面。
2. Select 出来的资料形态必须符合 column1。
其他查询
Select *
FROM table_name1
Where column1 LIKE 'x%'
说明：LIKE 必须和後面的'x%' 相呼应表示以 x 为开头的字串。
Select *
FROM table_name1
Where column1 IN ('xxx','yyy',..)
说明：IN 後面接的是一个集合，表示 column1 存在集合里面。
Select *
FROM table_name1
Where column1 BETWEEN xx AND yy
说明：BETWEEN 表示 column1 的值介於 xx 和 yy 之间。
3、更改资料：
Update table_name
SET column1='xxx'
Where conditoins
说明：
1.更改某个栏位设定其值为'xxx'。
2.conditions 是所要符合的条件、若没有 Where 则整个 table 的那个栏位都会全部被更改。
4、删除资料：
Delete FROM table_name
Where conditions
说明：删除符合条件的资料。
说明：关于 Where 条件后面如果包含有日期的比较，不同数据库有不同的表达式。具体如下：
(1)如果是 ACCESS 数据库，则为：Where mydate〉#2000-01-01#
(2)如果是 ORACLE 数据库，则为：Where mydate〉cast('2000-01-01' as date)
或：Where mydate〉to_date('2000-01-01','yyyy-mm-dd')
在 Delphi 中写成：
thedate='2000-01-01';
query1.SQL.add('select * from abc where mydate〉cast('+''''+thedate+''''+' as date)');
如果比较日期时间型，则为：
Where mydatetime〉to_date('2000-01-01 10:00:01','yyyy-mm-dd hh24:mi:ss')
Recordset 对象一些有用的属性"/〉 引用来自 增加一个 ：Recordset 对象一些有用的属性
rs.CursorType=
rs.CursorLocation=
rs.LockType =
rs.CacheSize=

rs.Pagesize=
rs.Pagecount=
rs.RecordCount=
’’---- CursorType Values ----
Const adOpenForwardOnly = 0 仅向前
Const adOpenKeyset = 1 键集游标
Const adOpenDynamic = 2 动态游标
Const adOpenStatic = 3 静态游标
’’---- LockType Values ----
Const adLockReadOnly = 1 默认值，只读
Const adLockPessimistic = 2 保守式记录锁定
Const adLockOptimistic = 3 开放式记录锁定，只在调用 Update 方法时锁定记录
Const adLockBatchOptimistic = 4 开放式批更新
’’---- CursorLocation Values ----
Const adUseServer = 2
Const adUseClient = 3
Set rs=Server.CreateObject("ADODB.Rrecordset")
rs.Open.sqlst,conn,1,1 ’读取
rs.Open sqlst,conn,1,2 ’新增，修改，或删除）
下一页 ：《SQL SERVER 的数据类型》
〉〉〉---------我想分页！--这么长的文章，在这里来个分页多好啊！哈哈----------〈〈〈
《SQL SERVER 的数据类型》
1.SQL SERVER 的数据类型

数据类弄是数据的一种属性，表示数据所表示信息的类型。任何一种计算机语言都定义了自己的数据类型。当然，不同的程序语言都具有不同的特点，所定义的数据类型的各类和名称都或多或少有些不同。SQLServer 提供了 25 种数据类型：

- ·Binary [(n)]
- ·Varbinary [(n)]
- ·Char [(n)]
- ·Varchar[(n)]
- ·Nchar[(n)]
- ·Nvarchar[(n)]
- ·Datetime
- ·Smalldatetime
- ·Decimal[(p[,s])]
- ·Numeric[(p[,s])]
- ·Float[(n)]
- ·Real
- ·Int
- ·Smallint
- ·Tinyint
- ·Money
- ·Smallmoney
- ·Bit
- ·Cursor
- ·Sysname
- ·Timestamp
- ·Uniqueidentifier
- ·Text

·Image

·Ntext

(1)二进制数据类型

二进制数据包括 Binary、Varbinary 和 Image

Binary 数据类型既可以是固定长度的(Binary),也可以是变长度的。

Binary[(n)] 是 n 位固定的二进制数据。其中，n 的取值范围是从 1 到 8000。其存储窖的大小是 n + 4 个字节。

Varbinary[(n)] 是 n 位变长度的二进制数据。其中，n 的取值范围是从 1 到 8000。其存储窖的大小是 n + 4 个字节，不是 n 个字节。

在 Image 数据类型中存储的数据是以位字符串存储的，不是由 SQL Server 解释的，必须由应用程序来解释。例如，应用程序可以使用 BMP、TIEF、GIF 和 JPEG 格式把数据存储在 Image 数据类型中。

(2)字符数据类型

字符数据的类型包括 Char，Varchar 和 Text

字符数据是由任何字母、符号和数字任意组合而成的数据。

Varchar 是变长字符数据，其长度不超过 8KB。Char 是定长字符数据，其长度最多为 8KB。超过 8KB 的 ASCII 数据可以使用 Text 数据类型存储。例如，因为 Html 文档全部都是 ASCII 字符，并且在一般情况下长度超过 8KB，所以这些文档可以 Text 数据类型存储在 SQL Server 中。

(3)Unicode 数据类型

Unicode 数据类型包括 Nchar,Nvarchar 和 Ntext

在 Microsoft SQL Server 中，传统的非 Unicode 数据类型允许使用由特定字符集定义的字符。在 SQL Server 安装过程中，允许选择一种字符集。使用 Unicode 数据类型，列中可以存储任何由 Unicode 标准定义的字符。在 Unicode 标准中，包括了以各种字符集定义的全部字符。使用 Unicode 数据类型，所战胜的窖是使用非 Unicode 数据类型所占用的窖大小的两倍。

在 SQL Server 中，Unicode 数据以 Nchar、Nvarchar 和 Ntext 数据类型存储。使用这种字符类型存储的列可以存储多个字符集中的字符。当列的长度变化时，应该使用 Nvarchar 字符类型，这时最多可以存储 4000 个字符。当列的长度固定不变时，应该使用 Nchar 字符类型，同样，这时最多可以存储 4000 个字符。当使用 Ntext 数据类型时，该列可以存储多于 4000 个字符。

(4)日期和时间数据类型

日期和时间数据类型包括 Datetime 和 Smalldatetime 两种类型

日期和时间数据类型由有效的日期和时间组成。例如，有效的日期和时间数据包括“4/01/98 12:15:00:00:00 PM”和“1:28:29:15:01AM 8/17/98”。前一个数据类型是日期在前，时间在后一个数据类型是霎时间在前，日期在后。在 Microsoft SQL Server 中，日期和时间数据类型包括 Datetime 和 Smalldatetime 两种类型时，所存储的日期范围是从 1753 年 1 月 1 日开始，到 9999 年 12 月 31 日结束(每一个值要求 8 个存储字节)。使用 Smalldatetime 数据类型时，所存储的日期范围是 1900 年 1 月 1 日 开始，到 2079 年 12 月 31 日结束(每一个值要求 4 个存储字节)。

日期的格式可以设定。设置日期格式的命令如下：

Set DateFormat {format | @format _var|

其中，format | @format_var 是日期的顺序。有效的参数包括 MDY、DMY、YMD、YDM、MYD 和 DYM。在默认情况下，日期格式为 MDY。

例如，当执行 Set DateFormat YMD 之后，日期的格式为年 月 日 形式；当执行 Set DateFormat DMY 之后，日期的格式为日 月有年 形式

（5）数字数据类型

数字数据只包含数字。数字数据类型包括正数和负数、小数（浮点数）和整数

整数由正整数和负整数组成，例如 39、25、0-2 和 33967。在 Micrsoft SQL Server 中，整数存储的数据类型是 Int，Smallint 和 Tinyint。Int 数据类型存储数据的范围大于 Smallint 数据类型存储数据的范围，而 Smallint 据类型存储数据的范围大于 Tinyint 数据类型存储数据的范围。使用 Int 数据狗昔存储数据的范围是从 -2 147 483 648 到 2 147 483 647（每一个值要求 4 个字节存储空间）。使用 Smallint 数据类型时，存储数据的范围从 -32 768 到 32 767（每一个值要求 2 个字节存储空间）。使用 Tinyint 数据类型时，存储数据的范围是从 0 到 255（每一个值要求 1 个字节存储空间）。

精确小娄数据在 SQL Server 中的数据类型是 Decimal 和 Numeric。这种数据所占的存储空间根据该数据的位数后的位数来确定。

在 SQL Server 中，近似小数数据的数据类型是 Float 和 Real。例如，三分之一这个分数记作。3333333，当使用近似数据类型时能准确表示。因此，从系统中检索到的数据可能与存储在该列中数据不完全一样。

（6）货币数据表示正的或者负的货币数量 。

在 Microsoft SQL Server 中，货币数据的数据类型是 Money 和 Smallmoney

Money 数据类型要求 8 个存储字节，Smallmoney 数据类型要求 4 个存储字节。

（7）特殊数据类型

特殊数据类型包括前面没有提过的数据类型。特殊的数据类型有 3 种，即 Timestamp、Bit 和 Uniqueidentifier。

Timestamp 用于表示 SQL Server 活动的先后顺序，以二进投影的格式表示。Timestamp 数据与插入数据或者日期和时间没有关系。

Bit 由 1 或者 0 组成。当表示真或者假、ON 或者 OFF 时，使用 Bit 数据类型。例如，询问是否是每一次访问的客户机请求可以存储在这种数据类型的列中。

Uniqueidentifier 由 16 字节的十六进制数字组成，表示一个全局唯一的。当表的记录行要求唯一时，GUID 是非常有用。例如，在客户标识号列使用这种数据类型可以区别不同的客户。

2.用户定义的数据类型

用户定义的数据类型基于在 Microsoft SQL Server 中提供的数据类型。当几个表中必须存储同一种数据类型时，并且为保证这些列有相同的数据类型、长度和可空性时，可以使用用户定义的数据类型。例如，可定义一种称为 postal_code 的数据类型，它基于 Char 数据类型。

当创建用户定义的数据类型时，必须提供三个数：数据类型的名称、所基于的系统数据类型和数据类型的可空性。

（1）创建用户定义的数据类型

创建用户定义的数据类型可以使用 Transact-SQL 语句。系统存储过程 sp_addtype 可以来创建用户定义的数据类型。其语法形式如下：

sp_addtype {type},[,system_data_bype][,’null_type’]

其中，type 是用户定义的数据类型的名称。system_data_type 是系统提供的数据类型，例如 Decimal、Int、Char 等等。 null_type 表示该数据类型是如何处理空值的，必须使用单引号引起来，例如’NULL’、’NOT NULL’或者’NONULL’。

例子：

Use cust

Exec sp_addtype ssn,’Varchar(11)’,"Not Null’

创建一个用户定义的数据类型 ssn，其基于的系统数据类型是变长为 11 的字符，不允许空。

例子：

Use cust

Exec sp_addtype birthday,datetime,’Null’

创建一个用户定义的数据类型 birthday，其基于的系统数据类型是 DateTime，允许空。

例子：

Use master

Exec sp_addtype telephone,’varchar(24),’Not Null’

Eexc sp_addtype fax,’varchar(24)’,’Null’

创建两个数据类型，即 telephone 和 fax

（2）删除用户定义的数据类型

当用户定义的数据类型不需要时，可删除。删除用户定义的数据类型的命令是 sp_droptype {’type’}。

例子：

Use master

Exec sp_droptype ’ssn’

注意：当表中的列还正在使用用户定义的数据类型时，或者在其上面还绑定有默认或者规则时，这种用户定义的数据类型不能删除。

以下为 SQL SERVER7.0 以上版本的字段类型说明。SQL SERVER6.5 的字段类型说明请参考 SQL SERVER 提供的说明。

字段类型 描述

bit 0 或 1 的整型数字

int 从-2^31(-2,147,483,648)到 2^31(2,147,483,647)的整型数字

smallint 从-2^15(-32,768)到 2^15(32,767)的整型数字

tinyint 从 0 到 255 的整型数字

decimal 从-10^38 到 10^38-1 的定精度与有效位数的数字

numeric decimal 的同义词

money 从-2^63(-922,337,203,685,477.5808)到 2^63-1(922,337,203,685,477.5807)的货币数据，最小货币单位千分之十

smallmoney 从-214,748.3648 到 214,748.3647 的货币数据，最小货币单位千分之十

float 从-1.79E+308 到 1.79E+308 可变精度的数字

real 从-3.04E+38 到 3.04E+38 可变精度的数字

datetime 从 1753 年 1 月 1 日到 9999 年 12 日 31 的日期和时间数据，最小时间单位为百分之三秒或 3.33 毫秒

smalldatetime 从 1900 年 1 月 1 日到 2079 年 6 月 6 日的日期和时间数据，最小时间单位为分钟

timestamp 时间戳，一个数据库宽度的唯一数字

uniqueidentifier 全球唯一标识符 GUID

char 定长非 Unicode 的字符型数据，最大长度为 8000

varchar 变长非 Unicode 的字符型数据，最大长度为 8000

text 变长非 Unicode 的字符型数据，最大长度为 2^31-1(2G)

nchar 定长 Unicode 的字符型数据，最大长度为 8000

nvarchar 变长 Unicode 的字符型数据，最大长度为 8000

ntext 变长 Unicode 的字符型数据，最大长度为 2^31-1(2G)

binary 定长二进制数据，最大长度为 8000

varbinary 变长二进制数据，最大长度为 8000

image 变长二进制数据，最大长度为 2^31-1(2G)

〉〉〉---------我想分页！--这么长的文章，在这里来个分页多好啊！哈哈----------〈〈〈

《SQL 语句的基本语法》

一.Select 语句的完整语法为：

Select[ALL|DISTINCT|DISTINCTROW|TOP]

{*|talbe.*|[table.]field1[AS alias1][,[table.]field2[AS alias2][,…]]}

FROM tableexpression[,…][IN externaldatabase]

[Where…]

[GROUP BY…]

[HAVING…]

[ORDER BY…]

[WITH OWNERACCESS OPTION]

说明：

用中括号([])括起来的部分表示是可选的，用大括号({})括起来的部分是表示必须从中选择其中的一个。

1 FROM 子句

FROM 子句指定了 Select 语句中字段的来源。FROM 子句后面是包含一个或多个的表达式(由逗号分开)，其中的表达式可为单一表名称、已保存的查询或由 INNER JOIN、LEFT JOIN 或 RIGHT JOIN 得到的复合结果。如果表或查询存储在外部数据库，在 IN 子句之后指明其完整路径。

例：下列 SQL 语句返回所有有定单的客户：

Select orderID,Customer.customerID

FROM orders Customers

Where orders.CustomerID=Customers.CustomeersID

2 ALL、DISTINCT、DISTINCTROW、TOP 谓词
(1) ALL 返回满足 SQL 语句条件的所有记录。如果没有指明这个谓词，默认为 ALL。
例：Select ALL FirstName,LastName
FROM Employees
(2) DISTINCT 如果有多个记录的选择字段的数据相同，只返回一个。
(3) DISTINCTROW 如果有重复的记录，只返回一个
(4) TOP 显示查询头尾若干记录。也可返回记录的百分比，这是要用 TOP N PERCENT 子句（其中 N 表示百分比）
例：返回 5%定货额最大的定单
Select TOP 5 PERCENT*
FROM [order Details]
orDER BY UnitPrice*Quantity*(1-Discount) DESC
3 用 AS 子句为字段取别名
如果想为返回的列取一个新的标题，或者，经过对字段的计算或总结之后，产生了一个新的值，希望把它放到一个新的列里显示，则用 AS 保留。
例：返回 FirstName 字段取别名为 NickName
Select FirstName AS NickName ,LastName ,City
FROM Employees
例：返回新的一列显示库存价值
Select ProductName ,UnitPrice ,UnitsInStock ,UnitPrice*UnitsInStock AS valueInStock
FROM Products
二 .Where 子句指定查询条件
1 比较运算符
比较运算符 含义
= 等于
〉 大于
〈 小于
〉= 大于等于
〈= 小于等于
〈〉 不等于
!〉 不大于
!〈 不小于
例：返回 96 年 1 月的定单
Select orderID, CustomerID, orderDate
FROM orders
Where orderDate〉#1/1/96# AND orderDate〈#1/30/96#
注意：
Mcirosoft JET SQL 中，日期用‘#’定界。日期也可以用 Datevalue()函数来代替。在比较字符型的数据时，要加上单引号”，尾空格在比较中被忽略。
例：
Where orderDate〉#96-1-1#
也可以表示为：
Where orderDate〉Datevalue(‘1/1/96’)
使用 NOT 表达式求反。
例：查看 96 年 1 月 1 日以后的定单
Where Not orderDate〈=#1/1/96#
2 范围（BETWEEN 和 NOT BETWEEN）
BETWEEN …AND…运算符指定了要搜索的一个闭区间。
例：返回 96 年 1 月到 96 年 2 月的定单。

Where orderDate Between #1/1/96# And #2/1/96#
3 列表（IN ，NOT IN）
IN 运算符用来匹配列表中的任何一个值。IN 子句可以代替用 OR 子句连接的一连串的条件。
例：要找出住在 London、Paris 或 Berlin 的所有客户
Select CustomerID, CompanyName, ContactName, City
FROM Customers
Where City In('London',' Paris',' Berlin')
4 模式匹配(LIKE)
LIKE 运算符检验一个包含字符串数据的字段值是否匹配一指定模式。
LIKE 运算符里使用的通配符
通配符 含义
？ 任何一个单一的字符
* 任意长度的字符
# 0~9 之间的单一数字
[字符列表] 在字符列表里的任一值
[！字符列表] 不在字符列表里的任一值
- 指定字符范围，两边的值分别为其上下限
例：返回邮政编码在（171）555-0000 到（171）555-9999 之间的客户
Select CustomerID ,CompanyName,City,Phone
FROM Customers
Where Phone Like '(171)555-####'
LIKE 运算符的一些样式及含义
样式 含义 不符合
LIKE 'A*' A 后跟任意长度的字符 Bc,c255
LIKE'5
' 5*5 555
LIKE'5?5' 5 与 5 之间有任意一个字符 55,5wer5
LIKE'5##5' 5235，5005 5kd5,5346
LIKE'[a-z]' a-z 间的任意一个字符 5,%
LIKE'[!0-9]' 非 0-9 间的任意一个字符 0,1
LIKE'[[]' 1,*
三 .用 ORDER BY 子句排序结果
orDER 子句按一个或多个（最多 16 个）字段排序查询结果，可以是升序（ASC）也可以是降序（DESC），缺省是升序。ORDER 子句通常放在 SQL 语句的最后。
orDER 子句中定义了多个字段，则按照字段的先后顺序排序。
例：
Select ProductName,UnitPrice, UnitInStock
FROM Products
orDER BY UnitInStock DESC , UnitPrice DESC, ProductName
orDER BY 子句中可以用字段在选择列表中的位置号代替字段名，可以混合字段名和位置号。
例：下面的语句产生与上列相同的效果。
Select ProductName,UnitPrice, UnitInStock
FROM Products
orDER BY 1 DESC , 2 DESC,3
四 .运用连接关系实现多表查询
例：找出同一个城市中供应商和客户的名字
Select Customers.CompanyName, Suppliers.ComPany.Name
FROM Customers, Suppliers
Where Customers.City=Suppliers.City

例：找出产品库存量大于同一种产品的定单的数量的产品和定单
Select ProductName,OrderID, UnitInStock, Quantity
FROM Products, [Order Deails]
Where Product.productID=[Order Details].ProductID
AND UnitsInStock〉Quantity
另一种方法是用 Microsof JET SQL 独有的 JNNER JOIN
语法：
FROM table1 INNER JOIN table2
ON table1.field1 comparision table2.field2
其中 comparision 就是前面 Where 子句用到的比较运算符。
Select FirstName,lastName,OrderID,CustomerID,OrderDate
FROM Employees
INNER JOIN orders ON Employees.EmployeeID=Orders.EmployeeID
注意：
INNER JOIN 不能连接 Memo OLE Object Single Double 数据类型字段。
在一个 JOIN 语句中连接多个 ON 子句
语法：
Select fields
FROM table1 INNER JOIN table2
ON table1.field1 compopr table2.field1 AND
ON table1.field2 compopr table2.field2 or
ON table1.field3 compopr table2.field3
也可以
Select fields
FROM table1 INNER JOIN
（table2 INNER JOIN [( ]table3
[INNER JOER] [( ]tablex[INNER JOIN]
ON table1.field1 compopr table2.field1
ON table1.field2 compopr table2.field2
ON table1.field3 compopr table2.field3
外部连接返回更多记录，在结果中保留不匹配的记录，不管存不存在满足条件的记录都要返回另一侧的所有记录。
FROM table [LEFT|RIGHT]JOIN table2
ON table1.field1comparision table.field2
用左连接来建立外部连接，在表达式的左边的表会显示其所有的数据
例：不管有没有定货量，返回所有商品
Select ProductName ,OrderID
FROM Products
LEFT JOIN orders ON Products.PrductsID=Orders.ProductID
右连接与左连接的差别在于：不管左侧表里有没有匹配的记录，它都从左侧表中返回所有记录。
例：如果想了解客户的信息，并统计各个地区的客户分布，这时可以用一个右连接，即使某个地区没有客户，也要返回客户信息。
空值不会相互匹配，可以通过外连接才能测试被连接的某个表的字段是否有空值。
Select *
FROM talbe1
LEFT JOIN table2 ON table1.a=table2.c
1 连接查询中使用 Iif 函数实现以 0 值显示空值
Iif 表达式： Iif(IsNull(Amount,0,Amout)
例：无论定货大于或小于￥50，都要返回一个标志。
Iif([Amount]〉50,?Big order?,?Small order?)

五. 分组和总结查询结果
在 SQL 的语法里，GROUP BY 和 HAVING 子句用来对数据进行汇总。GROUP BY 子句指明了按照哪几个字段来分组，而将记录分组后，用 HAVING 子句过滤这些记录。
GROUP BY 子句的语法
Select fidldlist
FROM table
Where criteria
[GROUP BY groupfieldlist [HAVING groupcriteria]]
注：Microsoft Jet 数据库 Jet 不能对备注或 OLE 对象字段分组。
GROUP BY 字段中的 Null 值以备分组但是不能被省略。
在任何 SQL 合计函数中不计算 Null 值。
GROUP BY 子句后最多可以带有十个字段，排序优先级按从左到右的顺序排列。
例：在'WA'地区的雇员表中按头衔分组后，找出具有同等头衔的雇员数目大于 1 人的所有头衔。
Select Title ,Count(Title) as Total
FROM Employees
Where Region = 'WA'
GROUP BY Title
HAVING Count(Title)〉1
JET SQL 中的聚积函数
聚集函数 意义
SUM ( ) 求和
AVG ( ) 平均值
COUNT ( ) 表达式中记录的数目
COUNT (*) 计算记录的数目
MAX 最大值
MIN 最小值
VAR 方差
STDEV 标准误差
FIRST 第一个值
LAST 最后一个值
六. 用 Parameters 声明创建参数查询
Parameters 声明的语法:
PARAMETERS name datatype[,name datatype[, …]]
其中 name 是参数的标志符,可以通过标志符引用参数.
Datatype 说明参数的数据类型.
使用时要把 PARAMETERS 声明置于任何其他语句之前.
例:
PARAMETERS[Low price] Currency,[Beginning date]datatime
Select orderID ,OrderAmount
FROM orders
Where orderAMount〉[low price]
AND orderDate〉=[Beginning date]
七. 功能查询
所谓功能查询,实际上是一种操作查询,它可以对数据库进行快速高效的操作.它以选择查询为目的,挑选出符合条件的数据,再对数据进行批处理.功能查询包括更新查询,删除查询,添加查询,和生成表查询.
1 更新查询
Update 子句可以同时更改一个或多个表中的数据.它也可以同时更改多个字段的值.
更新查询语法:
Update 表名

SET 新值
Where 准则
例:英国客户的定货量增加 5%,货运量增加 3%
Update OEDERS
SET orderAmount = orderAmount *1.1
Freight = Freight*1.03
Where ShipCountry = 'UK'
2 删除查询
Delete 子句可以使用户删除大量的过时的或冗于的数据.
注:删除查询的对象是整个记录.
Delete 子句的语法:
Delete [表名.*]
FROM 来源表
Where 准则
例: 要删除所有 94 年前的定单
Delete *
FROM orders
Where orderData〈#94-1-1#
3 追加查询
Insert 子句可以将一个或一组记录追加到一个或多个表的尾部.
INTO 子句指定接受新记录的表
valueS 关键字指定新记录所包含的数据值.
Insert 子句的语法:
INSETR INTO 目的表或查询(字段 1,字段 2,…)
valueS(数值 1,数值 2,…)
例:增加一个客户
Insert INTO Employees(FirstName,LastName,title)
valueS('Harry','Washington','Trainee')
4 生成表查询
可以一次性地把所有满足条件的记录拷贝到一张新表中.通常制作记录的备份或副本或作为报表的基础.
Select INTO 子句用来创建生成表查询语法:
Select 字段 1,字段 2,…
INTO 新表[IN 外部数据库]
FROM 来源数据库
Where 准则
例:为定单制作一个存档备份
Select *
INTO ordersArchive
FROM orders
八. 联合查询
UNION 运算可以把多个查询的结果合并到一个结果集里显示.
UNION 运算的一般语法:
[表]查询 1 UNION [ALL]查询 2 UNION …
例:返回巴西所有供给商和客户的名字和城市
Select CompanyName,City
FROM Suppliers
Where Country = 'Brazil'
UNION
Select CompanyName,City

FROM Customers
Where Country = 'Brazil'
注:
缺省的情况下,UNION 子句不返回重复的记录.如果想显示所有记录,可以加 ALL 选项
UNION 运算要求查询具有相同数目的字段.但是,字段数据类型不必相同.
每一个查询参数中可以使用 GROUP BY 子句 或 HAVING 子句进行分组.要想以指定的顺序来显示返回的数据,可以在最后一个查询的尾部使用 OREER BY 子句.
九. 交叉查询
交叉查询可以对数据进行总和,平均,计数或其他总和计算法的计算,这些数据通过两种信息进行分组:一个显示在表的左部,另一个显示在表的顶部.
Microsoft Jet SQL 用 TRANSFROM 语句创建交叉表查询语法:
TRANSFORM aggfunction
Select 语句
GROUP BY 子句
PIVOT pivotfield[IN(value1 [,value2[,…]])]
Aggfounction 指 SQL 聚积函数,
Select 语句选择作为标题的的字段,
GROUP BY 分组
说明:
Pivotfield 在查询结果集中创建列标题时用的字段或表达式,用可选的 IN 子句限制它的取值.
value 代表创建列标题的固定值.
例:显示在 1996 年里每一季度每一位员工所接的定单的数目:
TRANSFORM Count(OrderID)
Select FirstName&''&LastName AS FullName
FROM Employees INNER JOIN orders
ON Employees.EmployeeID = orders.EmployeeID
Where DatePart("yyyy",OrderDate)= '1996'
GROUP BY FirstName&''&LastName
orDER BY FirstName&''&LastName
POVOT DatePart("q",OrderDate)&'季度'
十 .子查询
子查询可以理解为 套查询.子查询是一个 Select 语句.
1 表达式的值与子查询返回的单一值做比较
语法:
表达式 comparision [ANY|ALL|SOME](子查询)
说明:
ANY 和 SOME 谓词是同义词,与比较运算符(=,〈,〉,〈〉,〈=,〉=)一起使用.返回一个布尔值 True 或 False.ANY 的意思是,表达式与子查询返回的一系列的值逐一比较,只要其中的一次比较产生 True 结果,ANY 测试的返回 True 值(既 Where 子句的结果),对应于该表达式的当前记录将进入主查询的结果中.ALL 测试则要求表达式与子查询返回的一系列的值的比较都产生 True 结果,才回返回 True 值.
例:主查询返回单价比任何一个折扣大于等于 25%的产品的单价要高的所有产品
Select * FROM Products
Where UnitPrice〉ANY
(Select UnitPrice FROM[Order Details] Where Discount〉0.25)
2 检查表达式的值是否匹配子查询返回的一组值的某个值
语法:
[NOT]IN(子查询)
例:返回库存价值大于等于 1000 的产品.
Select ProductName FROM Products

Where ProductID IN
(Select PrdoctID FROM [Order DEtails]
Where UnitPrice*Quantity〉= 1000)
3 检测子查询是否返回任何记录
语法:
[NOT]EXISTS (子查询)
例:用 EXISTS 检索英国的客户
Select ComPanyName,ContactName
FROM orders
Where EXISTS
(Select *
FROM Customers
Where Country = 'UK' AND
Customers.CustomerID= orders.CustomerID)
〉〉〉---------我想分页！--这么长的文章，在这里来个分页多好啊！哈哈----------〈〈〈
Sql Server 和 Access 操作数据库结构 Sql 语句
下面是 Sql Server 和 Access 操作数据库结构的常用 Sql，希望对你有所帮助。
内容由海娃整理，不正确与不完整之处还请提出，谢谢。
新建表：
create table [表名]
(
[自动编号字段] int IDENTITY (1,1) PRIMARY KEY ,
[字段 1] nVarChar(50) default '默认值' null ,
[字段 2] ntext null ,
[字段 3] datetime,
[字段 4] money null ,
[字段 5] int default 0,
[字段 6] Decimal (12,4) default 0,
[字段 7] image null ,
)
删除表：
Drop table [表名]
插入数据：
Insert INTO [表名] (字段 1,字段 2) VALUES (100,'51WINDOWS.NET')
删除数据：
Delete FROM [表名] Where [字段名]〉100
更新数据：
Update [表名] SET [字段 1] = 200,[字段 2] = '51WINDOWS.NET' Where [字段三] = 'HAIWA'
新增字段：
Alter TABLE [表名] ADD [字段名] NVARCHAR (50) NULL
删除字段：
Alter TABLE [表名] Drop COLUMN [字段名]
修改字段：
Alter TABLE [表名] Alter COLUMN [字段名] NVARCHAR (50) NULL
重命名表：(Access 重命名表，请参考文章：在 Access 数据库中重命名表)
引用来自 在 Access 数据库中重命名表
Dim Conn,ConnStr,oCat,oTbl
ConnStr = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & Server.MapPath("data.mdb")
Set oCat=Server.CreateObject("ADOX.Catalog")

oCat.ActiveConnection = ConnStr
Set oTbl = Server.CreateObject("ADOX.Table")
Set oTbl = oCat.Tables("OldTable") '要重命名的表名：OldTable
oTbl.Name = "NewTable" '新表名
Set oCat = Nothing
Set oTbl = Nothing
sp_rename '表名', '新表名', 'OBJECT'
新建约束：
Alter TABLE [表名] ADD CONSTRAINT 约束名 CHECK ([约束字段] 〈= '2000-1-1')
删除约束：
Alter TABLE [表名] Drop CONSTRAINT 约束名
新建默认值
Alter TABLE [表名] ADD CONSTRAINT 默认值名 DEFAULT '51WINDOWS.NET' FOR [字段名]
删除默认值
Alter TABLE [表名] Drop CONSTRAINT 默认值名
删除 Sql Server 中的日志，减小数据库文件大小
dump transaction 数据库名 with no_log
backup log 数据库名 with no_log
dbcc shrinkdatabase(数据库名)
exec sp_dboption '数据库名', 'autoshrink', 'true'
SQL 查询语句精华大全
2009-08-17 10:16
简单的 Transact-SQL 查询只包括选择列表、FROM 子句和 WHERE 子句。它们分别
说明所查询列、查询的
表或视图、以及搜索条件等。
例如,下面的语句查询 testtable 表中姓名为“张三”的 nickname 字段和 email
字段。

```
SELECT nickname,email
FROM testtable
WHERE name='张三'
```

(一) 选择列表
选择列表(select_list)指出所查询列，它可以是一组列名列表、星号、表达式、
变量(包括局部变
量和全局变量)等构成。

1、选择所有列
例如，下面语句显示 testtable 表中所有列的数据：

```
SELECT *
FROM testtable
```

2、选择部分列并指定它们的显示次序
查询结果集合中数据的排列顺序与选择列表中所指定的列名排列顺序相同。
例如：

```
SELECT nickname,email
FROM testtable
```

3、更改列标题
在选择列表中，可重新指定列标题。定义格式为：

列标题=列名
列名 列标题
如果指定的列标题不是标准的标识符格式时，应使用引号定界符，例如，下列语句使用汉字显示列
标题：
```
SELECT 昵称=nickname,电子邮件=email
FROM testtable
```

4、删除重复行
SELECT 语句中使用 ALL 或 DISTINCT 选项来显示表中符合条件的所有行或删除其中重复的数据行，默认
为 ALL。使用 DISTINCT 选项时，对于所有重复的数据行在 SELECT 返回的结果集合中只保留一行。

5、限制返回的行数
使用 TOP n [PERCENT]选项限制返回的数据行数，TOP n 说明返回 n 行，而 TOP n PERCENT 时，说明 n 是
表示一百分数，指定返回的行数等于总行数的百分之几。
例如：
```
SELECT TOP 2 *
FROM testtable
SELECT TOP 20 PERCENT *
FROM testtable
```

(二) FROM 子句
FROM 子句指定 SELECT 语句查询及与查询相关的表或视图。在 FROM 子句中最多可指定 256 个表或视图，
它们之间用逗号分隔。
在 FROM 子句同时指定多个表或视图时，如果选择列表中存在同名列，这时应使用对象名限定这些列
所属的表或视图。例如在 usertable 和 citytable 表中同时存在 cityid 列，在查询两个表中的 cityid 时应
使用下面语句格式加以限定：
```
SELECT username,citytable.cityid
FROM usertable,citytable
WHERE usertable.cityid=citytable.cityid
```
在 FROM 子句中可用以下两种格式为表或视图指定别名：
表名 as 别名
表名 别名
例如上面语句可用表的别名格式表示为：
```
SELECT username,b.cityid
FROM usertable a,citytable b
WHERE a.cityid=b.cityid
```
SELECT 不仅能从表或视图中检索数据，它还能够从其它查询语句所返回的结果集合中查询数据。
例如：
```
SELECT a.au_fname+a.au_lname
FROM authors a,titleauthor ta
(SELECT title_id,title
```

FROM titles
WHERE ytd_sales>10000
) AS t
WHERE a.au_id=ta.au_id
AND ta.title_id=t.title_id
此例中，将 SELECT 返回的结果集合给予一别名 t，然后再从中检索数据。

(三) 使用 WHERE 子句设置查询条件
WHERE 子句设置查询条件，过滤掉不需要的数据行。例如下面语句查询年龄大于 20 的数据：
SELECT *
FROM usertable
WHERE age>20
WHERE 子句可包括各种条件运算符：
比较运算符(大小比较)：>、>=、=、<、<=、<>、!>、!<
范围运算符(表达式值是否在指定的范围)：BETWEEN…AND…
NOT BETWEEN…AND…
列表运算符(判断表达式是否为列表中的指定项)：IN (项 1,项 2……)
NOT IN (项 1,项 2……)
模式匹配符(判断值是否与指定的字符通配格式相符):LIKE、NOT LIKE
空值判断符(判断表达式是否为空)：IS NULL、NOT IS NULL
逻辑运算符(用于多条件的逻辑连接)：NOT、AND、OR
1、范围运算符例：age BETWEEN 10 AND 30 相当于 age>=10 AND age<=30
2、列表运算符例：country IN ('Germany','China')
3、模式匹配符例：常用于模糊查找，它判断列值是否与指定的字符串格式相匹配。可用于 char、
varchar、text、ntext、datetime 和 smalldatetime 等类型查询。
可使用以下通配字符：
百分号%：可匹配任意类型和长度的字符，如果是中文，请使用两个百分号即%%。
下划线_：匹配单个任意字符，它常用来限制表达式的字符长度。
方括号[]：指定一个字符、字符串或范围，要求所匹配对象为它们中的任一个。
[^]：其取值也[] 相同，但它要求所匹配对象为指定字符以外的任一个字符。
例如：
限制以 Publishing 结尾，使用 LIKE '%Publishing'
限制以 A 开头：LIKE '[A]%'
限制以 A 开头外：LIKE '[^A]%'
4、空值判断符例 WHERE age IS NULL
5、逻辑运算符：优先级为 NOT、AND、OR
(四)查询结果排序
使用 ORDER BY 子句对查询返回的结果按一列或多列排序。ORDER BY 子句的语法格式为：
ORDER BY {column_name [ASC|DESC]} [,…n]
其中 ASC 表示升序，为默认值，DESC 为降序。ORDER BY 不能按 ntext、text 和 image 数据类型进行排
序。
例如：
SELECT *
FROM usertable
ORDER BY age desc,userid ASC

另外，可以根据表达式进行排序。

二、 联合查询

UNION 运算符可以将两个或两个以上上 SELECT 语句的查询结果集合合并成一个结果集合显示，即执行联
合查询。UNION 的语法格式为：

select_statement
UNION [ALL] selectstatement
[UNION [ALL] selectstatement][…n]

其中 selectstatement 为待联合的 SELECT 查询语句。
ALL 选项表示将所有行合并到结果集合中。不指定该项时，被联合查询结果集合中的重复行将只保留一
行。
联合查询时，查询结果的列标题为第一个查询语句的列标题。因此，要定义列标题必须在第一个查询语
句中定义。要对联合查询结果排序时，也必须使用第一查询语句中的列名、列标题或者列序号。
在使用 UNION 运算符时，应保证每个联合查询语句的选择列表中有相同数量的表达式，并且每个查询选
择表达式应具有相同的数据类型，或是可以自动将它们转换为相同的数据类型。在自动转换时，对于数值类
型，系统将低精度的数据类型转换为高精度的数据类型。
在包括多个查询的 UNION 语句中，其执行顺序是自左至右，使用括号可以改变这一执行顺序。例如：

查询 1 UNION (查询 2 UNION 查询 3)

三、连接查询

通过连接运算符可以实现多个表查询。连接是关系数据库模型的主要特点，也是它区别于其它类型
数据库管理系统的一个标志。
在关系数据库管理系统中，表建立时各数据之间的关系不必确定，常把一个实体的所有信息存放在
一个表中。当检索数据时，通过连接操作查询出存放在多个表中的不同实体的信息。连接操作给用户带
来很大的灵活性，他们可以在任何时候增加新的数据类型。为不同实体创建新的表，尔后通过连接进行
查询。
连接可以在 SELECT 语句的 FROM 子句或 WHERE 子句中建立，似是而非在 FROM 子句中指出连接时有助于
将连接操作与 WHERE 子句中的搜索条件区分开来。所以，在 Transact-SQL 中推荐使用这种方法。
SQL-92 标准所定义的 FROM 子句的连接语法格式为：

FROM join_table join_type join_table
[ON (join_condition)]

其中 join_table 指出参与连接操作的表名，连接可以对同一个表操作，也可以对多表操作，对同一
个表操作的连接又称做自连接。
join_type 指出连接类型，可分为三种：内连接、外连接和交叉连接。内连接(INNER JOIN)使用比

较运算符进行表间某(些)列数据的比较操作，并列出这些表中与连接条件相匹配的数据行。根据所使用
的比较方式不同，内连接又分为等值连接、自然连接和不等连接三种。
外连接分为左外连接(LEFT OUTER JOIN 或 LEFT JOIN)、右外连接(RIGHT OUTER JOIN 或 RIGHT JOIN)
和全外连接(FULL OUTER JOIN 或 FULL JOIN)三种。与内连接不同的是，外连接不只列出与连接条件相匹
配的行，而是列出左表(左外连接时)、右表(右外连接时)或两个表(全外连接时)中所有符合搜索条件的
数据行。
交叉连接(CROSS JOIN)没有 WHERE 子句，它返回连接表中所有数据行的笛卡尔积，其结果集合中的
数据行数等于第一个表中符合查询条件的数据行数乘以第二个表中符合查询条件的数据行数。
连接操作中的 ON (join_condition) 子句指出连接条件，它由被连接表中的列和比较运算符、逻辑
运算符等构成。
无论哪种连接都不能对 text、ntext 和 image 数据类型列进行直接连接，但可以对这三种列进行间接
连接。例如：

```
SELECT p1.pub_id,p2.pub_id,p1.pr_info
FROM pub_info AS p1 INNER JOIN pub_info AS p2
ON DATALENGTH(p1.pr_info)=DATALENGTH(p2.pr_info)
```

(一)内连接
内连接查询操作列出与连接条件匹配的数据行，它使用比较运算符比较被连接列的列值。内连接分
三种：
1、等值连接：在连接条件中使用等于号(=)运算符比较被连接列的列值，其查询结果中列出被连接
表中的所有列，包括其中的重复列。
2、不等连接： 在连接条件使用除等于运算符以外的其它比较运算符比较被连接的列的列值。这些
运算符包括>、>=、<=、<、!>、!<和<>。
3、自然连接：在连接条件中使用等于(=)运算符比较被连接列的列值，但它使用选择列表指出查询
结果集合中所包括的列，并删除连接表中的重复列。
例，下面使用等值连接列出 authors 和 publishers 表中位于同一城市的作者和出版社：

```
SELECT *
FROM authors AS a INNER JOIN publishers AS p
ON a.city=p.city
```

又如使用自然连接，在选择列表中删除 authors 和 publishers 表中重复列(city 和 state)：

```
SELECT a.*,p.pub_id,p.pub_name,p.country
FROM authors AS a INNER JOIN publishers AS p
ON a.city=p.city
```

(二)外连接
内连接时，返回查询结果集合中的仅是符合查询条件(WHERE 搜索条件或

HAVING 条件)和连接条件
的行。而采用外连接时,它返回到查询结果集合中的不仅包含符合连接条件的行,而且还包括左表(左外
连接时)、右表(右外连接时)或两个边接表(全外连接)中的所有数据行。
如下面使用左外连接将论坛内容和作者信息连接起来:
SELECT a.*,b.* FROM luntan LEFT JOIN usertable as b
ON a.username=b.username
下面使用全外连接将 city 表中的所有作者以及 user 表中的所有作者,以及他们所在的城市:
SELECT a.*,b.*
FROM city as a FULL OUTER JOIN user as b
ON a.username=b.username

(三)交叉连接
交叉连接不带 WHERE 子句,它返回被连接的两个表所有数据行的笛卡尔积,返回到结果集合中的数
据行数等于第一个表中符合查询条件的数据行数乘以第二个表中符合查询条件的数据行数。
例,titles 表中有 6 类图书,而 publishers 表中有 8 家出版社,则下列交叉连接检索到的记录数将等
于 6*8=48 行。
SELECT type,pub_name
FROM titles CROSS JOIN publishers
ORDER BY typeSQL 核心语句(非常实用的几个技巧)插入数据

向表中添加一个新记录,你要使用 SQL INSERT 语句。这里有一个如何使用这种语句的例子:

INSERT mytable (mycolumn) VALUES (‘some data’)

这个语句把字符串’some data’插入表 mytable 的 mycolumn 字段中。将要被插入数据的字段的名字在第一个括号中指定,实际的数据在第二个括号中给出。

INSERT 语句的完整句法如下:

INSERT [INTO] {table_name|view_name} [(column_list)] {DEFAULT VALUES |

Values_list | select_statement}

如果一个表有多个字段,通过把字段名和字段值用逗号隔开,你可以向所有的字段中插入数据。假设表 mytable 有三个字段 first_column,second_column,和 third_column。下面的 INSERT 语句添加了一条三个字段都有值的完整记录:

INSERT mytable (first_column,second_column,third_column)

VALUES (‘some data’,’ some more data’,’ yet more data’)

注意

你可以使用 INSERT 语句向文本型字段中插入数据。但是,如果你需要输入很长

的字符串，你应该使用 WRITETEXT 语句。这部分内容对本书来说太高级了，因此不加讨论。要了解更多的信息，请参考 Microsoft SQL Sever 的文档。

如果你在 INSERT 语句中只指定两个字段和数据会怎么样呢？换句话说，你向一个表中插入一条新记录，但有一个字段没有提供数据。在这种情况下，有下面的四种可能：

如果该字段有一个缺省值，该值会被使用。例如，假设你插入新记录时没有给字段 third_column 提供数据，而这个字段有一个缺省值’some value’。在这种情况下，当新记录建立时会插入值’some value’。

如果该字段可以接受空值，而且没有缺省值，则会被插入空值。

如果该字段不能接受空值，而且没有缺省值，就会出现错误。你会收到错误信息：

The column in table mytable may not be null.

最后，如果该字段是一个标识字段，那么它会自动产生一个新值。当你向一个有标识字段的表中插入新记录时，只要忽略该字段，标识字段会给自己赋一个新值。

注意

向一个有标识字段的表中插入新记录后，你可以用 SQL 变量@@identity 来访问新记录

的标识字段的值。考虑如下的 SQL 语句：

INSERT mytable (first_column) VALUES(‘some value’)

INSERT anothertable(another_first,another_second)

VALUES(@@identity,’ some value’)

如果表 mytable 有一个标识字段，该字段的值会被插入表 anothertable 的 another_first 字段。这是因为变量@@identity 总是保存最后一次插入标识字段的值。

字段 another_first 应该与字段 first_column 有相同的数据类型。但是，字段 another_first 不能是应该标识字段。Another_first 字段用来保存字段 first_column 的值。

删除记录

要从表中删除一个或多个记录，需要使用 SQL DELETE 语句。你可以给 DELETE 语句提供 WHERE 子句。WHERE 子句用来选择要删除的记录。例如，下面的这个 DELETE 语句只删除字段 first_column 的值等于’Delete Me’的记录：

DELETE mytable WHERE first_column=’Deltet Me’

DELETE 语句的完整句法如下：

```
DELETE [FROM] {table_name|view_name} [WHERE clause]
```

在 SQL SELECT 语句中可以使用的任何条件都可以在 DELECT 语句的 WHERE 子句中使用。例如，下面的这个 DELETE 语句只删除那些 first_column 字段的值为’goodbye’或 second_column 字段的值为’so long’的记录：

```
DELETE mytable WHERE first_column=’goodby’ OR second_column=’so long’
```

如果你不给 DELETE 语句提供 WHERE 子句，表中的所有记录都将被删除。你不应该有这种想法。如果你想删除应该表中的所有记录，应使用第十章所讲的 TRUNCATE TABLE 语句。

注意

为什么要用 TRUNCATE TABLE 语句代替 DELETE 语句？当你使用 TRUNCATE TABLE 语句时，记录的删除是不作记录的。也就是说，这意味着 TRUNCATE TABLE 要比 DELETE 快得多。

更新记录

要修改表中已经存在的一条或多条记录，应使用 SQL UPDATE 语句。同 DELETE 语句一样，UPDATE 语句可以使用 WHERE 子句来选择更新特定的记录。请看这个例子：

```
UPDATE mytable SET first_column=’Updated!’ WHERE second_column=’Update Me!’
```

这个 UPDATE 语句更新所有 second_column 字段的值为’Update Me!’的记录。对所有被选中的记录，字段 first_column 的值被置为’Updated!’。

下面是 UPDATE 语句的完整句法：

```
UPDATE {table_name|view_name} SET [{table_name|view_name}]

{column_list|variable_list|variable_and_column_list}

[,{column_list2|variable_list2|variable_and_column_list2}…

[,{column_listN|variable_listN|variable_and_column_listN}]]

[WHERE clause]
```

注意

你可以对文本型字段使用 UPDATE 语句。但是，如果你需要更新很长的字符串，应使用 UPDATETEXT 语句。这部分内容对本书来说太高级了，因此不加讨论。要了解更多的信息，请参考 Microsoft SQL Sever 的文档。

如果你不提供 WHERE 子句，表中的所有记录都将被更新。有时这是有用的。例如，

如果你想把表 titles 中的所有书的价格加倍，你可以使用如下的 UPDATE 语句：

你也可以同时更新多个字段。例如，下面的 UPDATE 语句同时更新 first_column,second_column,和 third_column 这三个字段：

UPDATE mytable SET first_column=’ Updated!’

Second_column=’ Updated!’

Third_column=’ Updated!’

WHERE first_column=’ Update Me1’

技巧

SQL 忽略语句中多余的空格。你可以把 SQL 语句写成任何你最容易读的格式。

用 SELECT 创建记录和表

你也许已经注意到，INSERT 语句与 DELETE 语句和 UPDATE 语句有一点不同，它一次只操作一个记录。然而，有一个方法可以使 INSERT 语句一次添加多个记录。要作到这一点，你需要把 INSERT 语句与 SELECT 语句结合起来，象这样：

INSERT mytable (first_column,second_column)

SELECT another_first,another_second

FROM anothertable

WHERE another_first=’ Copy Me!’

这个语句从 anothertable 拷贝记录到 mytable.只有表 anothertable 中字段 another_first 的值为’ Copy Me！’ 的记录才被拷贝。

当为一个表中的记录建立备份时，这种形式的 INSERT 语句是非常有用的。在删除一个表中的记录之前，你可以先用这种方法把它们拷贝到另一个表中。

如果你需要拷贝整个表，你可以使用 SELECT INTO 语句。例如，下面的语句创建了一个名为 newtable 的新表，该表包含表 mytable 的所有数据：

SELECT * INTO newtable FROM mytable

你也可以指定只有特定的字段被用来创建这个新表。要做到这一点，只需在字段列表中指定你想要拷贝的字段。另外，你可以使用 WHERE 子句来限制拷贝到新表中的记录。下面的例子只拷贝字段 second_columnd 的值等于’ Copy Me!’ 的记录的 first_column 字段。

SELECT first_column INTO newtable

FROM mytable

WHERE second_column=’ Copy Me!’

使用 SQL 修改已经建立的表是很困难的。例如，如果你向一个表中添加了一个字段，没有容易的办法来去除它。另外，如果你不小心把一个字段的数据类型给错了，你将没有办法改变它。但是，使用本节中讲述的 SQL 语句，你可以绕过这两个问题。

例如，假设你想从一个表中删除一个字段。使用 SELECT INTO 语句，你可以创建该表的一个拷贝，但不包含要删除的字段。这使你既删除了该字段，又保留了不想删除的数据。

如果你想改变一个字段的数据类型，你可以创建一个包含正确数据类型字段的新表。创建好该表后，你就可以结合使用 UPDATE 语句和 SELECT 语句，把原来表中的所有数据拷贝到新表中。通过这种方法，你既可以修改表的结构，又能保存原有的数据。