SqlServer存储过程详解

http://www.cnblogs.com/sunniest/p/4386296.html

1.创建存储过程的基本语法模板：

复制代码

```
if (exists (select * from sys.objects where name = 'pro_name'))
    drop proc pro_name
go
create proc pro_name
    @param_name param_type [=default_value]
as
begin
    sql语句
end
```

复制代码

ps:[]表示非必写内容。sys.objects存储的是本数据库中的信息，不仅仅存储表名，还有存储过程名　、视图名、触发器等等。

例如：

复制代码

```
 1 if (exists (select * from sys.objects where name = 'USP_GetAllUser'))
 2     drop proc USP_GetAllUser
 3 go
 4 create proc USP_GetAllUser
 5 @UserId int =1
 6 as
 7 set nocount on;
 8 begin
 9     select * from UserInfo where Id=@UserId
10 end
```

复制代码

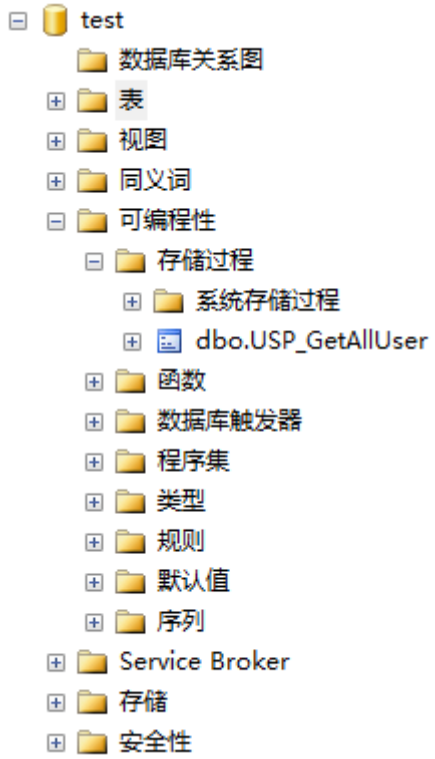
ps:SQL Server 实用工具将 GO 解释为应将当前的 Transact-SQL 批处理语句发送给 SQL Server 的信号。当前批处理语句是自上一 GO 命令后输入的所有语句，若是第一条 GO 命令，则是从特殊会话或脚本的开始处到这条 GO 命令之间的所有语句。

2.调用方法：

```
exec dbo.USP_GetAllUser 2;
```

ps:一般在执行存储过程是，最好加上架构名称，例如 dbo.USP_GetAllUser 这样可以可以减少不必要的系统开销，提高性能。 因为如果在存储过程名称前面没有加上架构名称，SQL SERVER 首先会从当前数据库sys schema（系统架构）开始查找，如果没有找到，则会去其它schema查找，最后在dbo架构（系统管理员架构）里面查找。

3.查看本数据库中存在的存储过程



依次展开数据库、可编程性、存储过程，即可看到已创建的存储过程。

4.修改存储过程

```
alter proc proc_name
as
    sql语句
```

5.存储过程中的输出参数的使用

复制代码

```
if (exists(select * from  sys.objects where name='GetUser'))
    drop proc GetUser
go
create proc GetUser
    @id int output,
    @name varchar(20) out
as
begin
    select @id=Id,@name=Name from UserInfo where Id=@id
end

go
declare
@name varchar(20),
@id int;
set @id=3;
exec dbo.GetUser @id,@name out;
select @id,@name;
print Cast(@id as varchar(10))+'-'+@name;
```

复制代码

ps:参数output为该参数可以输出

6.分页获取数据的存储过程

复制代码

```
if (exists(select * from  sys.objects where name='GetUserByPage'))
    drop proc GetUserByPage
go
create proc GetUserByPage
    @pageIndex int,
    @pageSize int
as
declare
@startIndex int,
@endIndex int;
```

```
set @startIndex =  (@pageIndex-1)*@pageSize+1;
set @endIndex = @startIndex + @pageSize -1 ;
begin
    select Id,Name from
    (
        select *,row_number()over (order by Id)as number from UserInfo
    )t where t.number>=@startIndex and t.number<=@endIndex
end

go
exec dbo.GetUserByPage 2,4;
```

复制代码

7.存储过程中事务的创建

复制代码

```
if (exists(select * from  sys.objects where name='JayJayToTest'))
    drop proc JayJayToTest
go
create proc JayJayToTest
    @GiveMoney int,
    @UserName nvarchar(20)
as
beginset nocount on;
    begin tran;
    begin try
        update BankTest set Money = Money-@GiveMoney where Name=@UserName;
        update BankTest set Money = Money+@GiveMoney where Name='test';
        commit;
    end try
    begin catch
        rollback tran;
        print ('发生异常，事务进行回滚');
    end catch
end
```

```
go
exec JayJayToTest 10,'jayjay'
```

复制代码

8.了解存储过程的执行计划

SELECT * FROM sys.[syscacheobjects]查看当前缓存的执行计划

如果执行存储过程时成功通过解析阶段，则 Microsoft SQL Server 查询优化器将分析存储过程中的 Transact-SQL 语句并创建一个执行计划。执行计划描述执行存储过程的最快方法，所依据的信息包括：

1.表中的数据量。

2.表的索引的存在及特征，以及数据在索引列中的分布。

3.WHERE 子句条件所使用的比较运算符和比较值。

4.是否存在联接以及 UNION、GROUP BY 和 ORDER BY 关键字。

查询优化器在分析完存储过程中的这些因素后，将执行计划置于内存中。分析存储过程和创建执行计划的过程称为编译。优化的内存中的执行计划将用来执行该查询。执行计划将驻留在内存中，直到重新启动 SQL Server 或其他对象需要存储空间时为止。如果随后执行了存储过程，而现有执行计划仍留在内存中，则 SQL Server 将重用现有执行计划。如果执行计划不再位于内存中，则创建新的执行计划。

重新编译执行计划(create proc JayJayToTest with recompile)

创建存储过程时在其定义中指定 WITH RECOMPILE 选项，表明 SQL Server 将不对该存储过程计划进行高速缓存；该存储过程将在每次执行时都重新编译。当存储过程的参数值在各次执行间都有较大差异，导致每次均需创建不同的执行计划时，可使用 WITH RECOMPILE 选项。此选项并不常用，因为每次执行存储过程时都必须对其进行重新编译，这样会使存储过程的执行变慢。

由于数据库的新状态，数据库内的某些更改可能会导致执行计划效率低下或不再有效。SQL Server 检测这些使执行计划无效的更改，并将计划标记为无效。此后，必须为执行查询的下一个连接重新编译新的计划。导致计划无效的情况包括：

1.对查询所引用的表或视图进行任何结构更改（ALTER TABLE 和 ALTER VIEW）。

2.通过语句（如 UPDATE STATISTICS）显式生成或者自动生成新的分发内容统计。

3.除去执行计划所使用的索引。

4.显式调用 sp_recompile。

5.对键的大量更改（其他用户对由查询引用的表使用 INSERT 或 DELETE 语句所产生的修改）。

6.对于带触发器的表，inserted 或 deleted 表内的行数显著增长。