УДК 681.3
Ч.А.КУЗЕКЕЕВА, Ф.Д.АКЗАМОВА

ИСПОЛЬЗОВАНИЕ ХРАНИМЫХ ПРОЦЕДУР ПРИ ВЗАИМОДЕЙСТВИИ БАЗЫ ДАННЫХ С СУБД MS SQL SERVER

MS SQL Server маалыматтар базасынын башкаруу системада сакталып калуучу процедураларын пайдалуунун муктаждыгы, анан оң жана терс жактары каралган.

Рассматривается целесообразность использования хранимых процедур при работе с СУБД MS SQL Server. Приведены как преимущества, так и недостатки работы с хранимыми процедурами, а также особенности их использования в среде программирования.

The article considers reasonability of stored procedures in the workflow of a MS SQL Server DBMS. Gives an overview of the pros and cons of working with stored procedures and particularity of their usage in the software development sphere.

Любая база данных SQL Server состоит из набора таблиц, содержащих данные, и дополнительных объектов, создаваемых для обработки данных. К таким объектам относятся представления, триггеры и хранимые процедуры /1/.

Хранимые процедуры обычно составляют значительную часть программного обеспечения SQL Server, а начиная с версии SQL Server 2005, обеспечивается взаимодействие хранимых процедур с инфраструктурой .NET, благодаря чему предоставляемые ими возможности еще больше расширяются /1/.

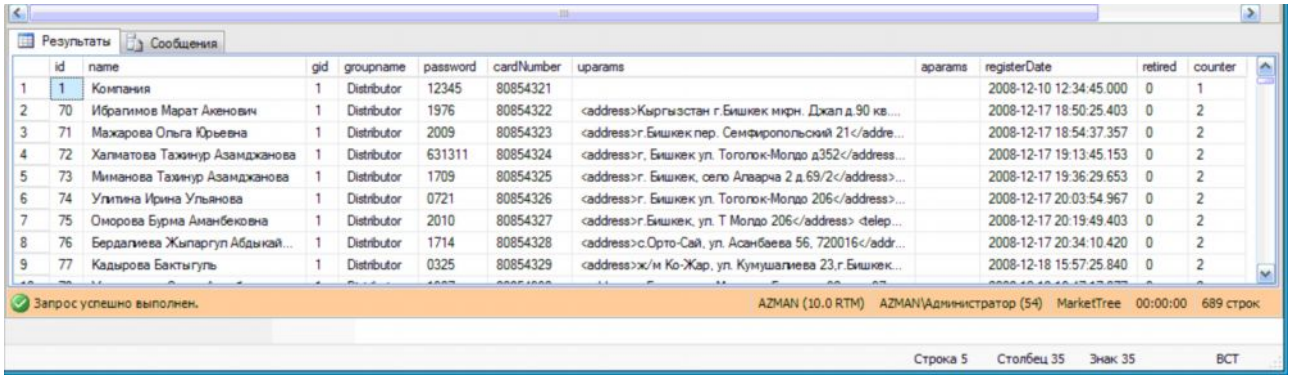
В данной статье рассмотрен способ взаимодействия приложения с базой данных посредством хранимых процедур в сравнении с SQL запросами /1/.

Хранимая процедура представляет собой оформленный особым образом сценарий (вернее, пакет), который хранится в базе данных, а не в отдельном файле. Хранимые процедуры отличаются от сценариев тем, что в них допускается использование входных и выходных параметров, а также возвращаемых значений, которые фактически не могут использоваться в обычном сценарии /1/.

Для иллюстрации работы процедуры приведем небольшой пример создания хранимой процедуры, используя БД MarketTree

```
Use MarketTree
   GO
CREATE PROCEDURE ShowDistributors    // создание хранимой процедуры
   AS
SELECT * FROM users      // выбор источника данных (таблицы)
   GO EXEC  ShowDistributors  // здесь мы вызываем хранимую процедуру на выполнение
```

Откомпилировав нашу процедуру, мы получим следующий результат:



	id	name	gid	groupname	password	cardNumber	uparams	aparams	registerDate	retired	counter
1	1	Компания	1	Distributor	12345	80854321			2008-12-10 12:34:45.000	0	1
2	70	Ибрагимов Марат Акенович	1	Distributor	1976	80854322	<address>Кыргызстан г.Бишкек мкрн. Джал д.90 кв....		2008-12-17 18:50:25.403	0	2
3	71	Мажарова Ольга Юрьевна	1	Distributor	2009	80854323	<address>г.Бишкек пер. Семферопольский 21</addre...		2008-12-17 18:54:37.357	0	2
4	72	Халматова Тахинур Азамджанова	1	Distributor	631311	80854324	<address>г, Бишкек ул. Тоголок-Молдо д352</address...		2008-12-17 19:13:45.153	0	2
5	73	Миманова Тахинур Азамджанова	1	Distributor	1709	80854325	<address>г. Бишкек, село Алварча 2 д.69/2</address>...		2008-12-17 19:36:29.653	0	2
6	74	Улитина Ирина Ульянова	1	Distributor	0721	80854326	<address>г. Бишкек ул. Тоголок-Молдо 206</address>...		2008-12-17 20:03:54.967	0	2
7	75	Оморова Бурма Аманбековна	1	Distributor	2010	80854327	<address>г.Бишкек, ул. Т Молдо 206</address> <telep...		2008-12-17 20:19:49.403	0	2
8	76	Бердалиева Жыпаргул Абдыкай...	1	Distributor	1714	80854328	<address>с.Орто-Сай, ул. Асанбаева 56, 720016</addr...		2008-12-17 20:34:10.420	0	2
9	77	Кадырова Бактыгуль	1	Distributor	0325	80854329	<address>ж/м Ко-Жар, ул. Кумушалиева 23,г.Бишкек...		2008-12-18 15:57:25.840	0	2

Так для чего же нужны хранимые процедуры, если мы можем использовать привычные и, на первый взгляд, более понятные SQL-запросы? И стоит ли полностью переходить к использованию хранимых процедур и забыть об SQL-запросах? В пользу использования хранимых процедур говорят достаточно много фактов, такие как простота, безопасность и производительность. Но в то же время следует учесть, что написать по-настоящему переносимый код хранимой процедуры практически невозможно. Хранимые процедуры сложнее в написании, чем основные операторы SQL, их подготовка требует большей квалификации и опыта. Ниже более подробно будут рассмотрены достоинства, недостатки, а так же особенности использования хранимых процедур.

Трудности в работе с хранимыми процедурами

1. **Проблема совместимости**. К сожалению, не все СУБД поддерживают работу с хранимыми процедурами, и если есть необходимость в совмещении приложения с максимальным количеством СУБД, то возможно, стоит использовать динамическую генерацию запросов. Однако следует заметить, что метод вызова самих хранимых процедур (их имена и метод передачи им данных) может быть достаточно переносимым, поэтому, если все-таки встанет вопрос о переходе на другую СУБД, то по крайней мере код приложения, возможно, не придется изменять /2/.
2. **Сложность внедрения.** Хранимые процедуры практически невозможно внедрить в уже существующий проект, написанный с использованием динамической генерации запросов. Подобные действия могут привести к полнейшей реорганизации кода работы с базой данных /3/.
3. **Передача сложных типов данных**. Бывает, что процедуре нужно будет передать более сложные данные, нежели число или строку, такие как, например, массив данных. В подобных случаях легче, все-таки, прибегнуть к построению запроса непосредственно в теле приложения, так как при использовании хранимой процедуры массив данных нужно будет преобразовывать в строку, а затем в теле хранимой процедуры будет происходить процесс обратного преобразования. В подобных случаях построение запроса в теле приложения значительно проще /3/.

Преимущества использования хранимых процедур

1. **Повышение скорости работы БД**. Так как хранимая процедура хранится в скомпилированном виде, то СУБД не нужно тратить время на постоянную компиляцию запроса при каждом его выполнении. Нет необходимости в генерации запросов, так как для вызова процедуры к исполнению достаточно лишь команды, которая гораздо короче, нежели запрос, поэтому на ее выполнение требуется меньшее количество времени для передачи команды на сервер /2/.
2. **Большая степень свободы**. Хранимые процедуры поддерживают: входные и выходные параметры, локальные переменные, операторы условного ветвления, циклы, вызовы встроенных команд и других процедур, а так же исполнение DDL-операторов /2/.

```
USE [MarketTree]
GO
/****** Object:  StoredProcedure [dbo].[addDistributor]    Script Date: 06/02/2009 16:40:09
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:              <Author,,Name>
-- Create date:      <Create Date,,>
```

```
-- Description:<Description,,>
-- =============================================
ALTER PROCEDURE [dbo].[addDistributor]
	-- Add the parameters for the stored procedure here
	@Name varchar(128) = '',
	@GID int = 0,
	@GroupName varchar(128) = '',
	@Password varchar(50) = '',
	@CardNumber varchar(50) = '',
	@UParams text = NULL,
	@AParams text = NULL,
	@Retired tinyint = 0,
	@ParentId int = 0
	AS
BEGIN
	BEGIN TRANSACTION;
	-- SET NOCOUNT ON added to prevent extra result sets from
	-- interfering with SELECT statements.
	SET NOCOUNT ON;
	DECLARE @UserId int;
	DECLARE @GlobalCounter int;
	SELECT @GlobalCounter  = counter FROM sys_settings WHERE id = 1;
	INSERT INTO users (name, gid, groupname, password, cardNumber, uparams, aparams,
registerDate, retired, counter)
	VALUES (@Name, @GID, @GroupName, @Password, @CardNumber, @UParams,
@AParams, CURRENT_TIMESTAMP, @Retired , @GlobalCounter);
	SELECT @UserId = id FROM users WHERE cardNumber = @CardNumber;
	INSERT INTO Structure (uid, pid, sid, spaid) VALUES(@UserId, @ParentId, 0, 0);
	INSERT INTO UserCart (uid, amount,discount) VALUES(@UserId, 0,0);
	INSERT INTO accountStatus (uid, totalPurchases, monthPurchases, params, lastAction,
salaryCounted, ownPurchases, structureVolume, structureVolumeInTheMonth, prevGroup,
currentGroup, nextGroup )
	VALUES (@UserId, 0, 0, '', CURRENT_TIMESTAMP, 0, 0, 0, 0, 0, 0, 0 );
  -- Insert statements for procedure here
	COMMIT TRANSACTION;
END
```

Пример хранимой процедуры с входным параметром.

3. **Упрощение кода приложения**. Для вызова хранимой процедуры необходимо знать только имя и список параметров (аналогично вызову обычных функций/методов в теле приложения). Подобный подход уменьшает размер кода и улучшает его читабельность, что положительно влияет на качество конечного продукта /2/.

4. **Безопасность**. Использование хранимых процедур позволяет значительно снизить угрозу возникновения уязвимости типа SQL-injection [1]. Хотя, конечно, при соблюдении правил добавления данных в запрос можно получить защищенный от взлома SQL-код, но не всегда это получается, так что безопаснее использовать хранимые процедуры. Ну, а кроме защиты от взлома, так же можно устанавливать права доступа к объектам базы данных для КАЖДОЙ хранимой процедуры, что также способствует повышению уровня безопасности приложения. Как видно из рисунка 1, в СУБД имеется

[1] SQL-injection - это один из распространенных способов взлома сайтов и программ, работающих с базами данных, основанный на внедрении в запрос произвольного SQL-кода.

список всех хранимых процедур, и к каждой из них можно обратиться отдельно, задав необходимые параметры /4/.

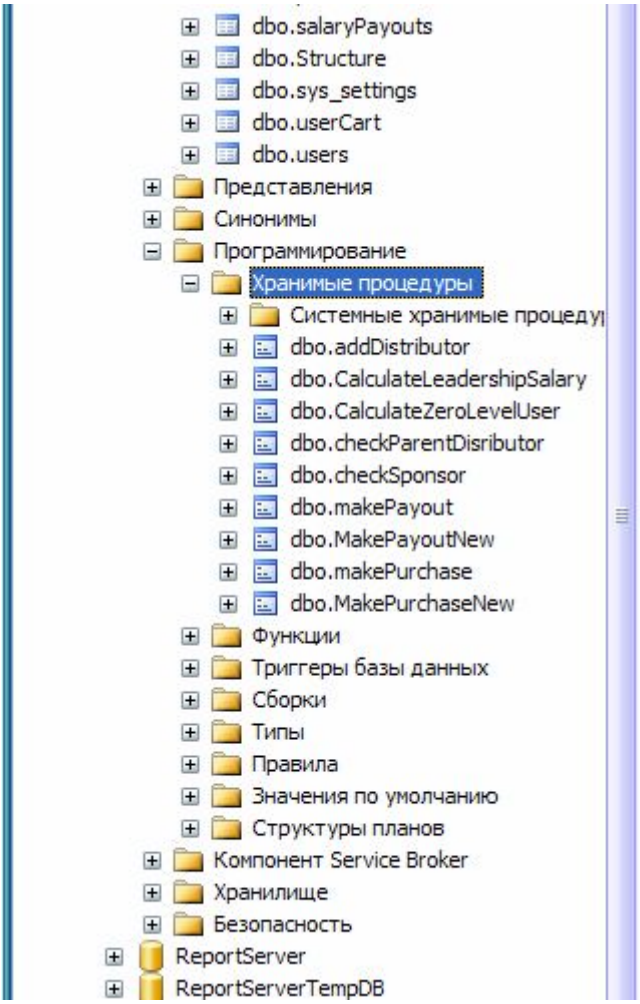


Рис.1. Перечень процедур

5. **Защита приложения от изменений структуры БД**. Если возникает необходимость в изменении структуры базы данных, например, добавить/удалить/переименовать таблицу или столбец, то организация доступа через хранимые процедуры не требует внесения изменений в код приложения до тех пор, пока имя хранимой процедуры и список параметров (а также ожидаемый результат) остаются прежними. Если приложение генерирует SQL-запросы, то необходимо внести изменения во все фрагменты кода, отвечающие за генерацию запросов /5/.

6. **Снижение количества ошибок и упрощение отладки**. Чаще всего ошибки в работе приложения с БД возникают по следующим причинам:

- в приложении используются некорректные значения для генерации SQL-запросов;
- ошибки в самом запросе;
- фрагмент кода приложения, отвечающий за генерацию SQL-запроса, содержит ошибку и не способен правильно построить нужный запрос /4/.

Если доступ к БД построен на основе хранимых процедур, то:

- легко узнать, какие значения попадают в хранимую процедуру, достаточно распечатать список аргументов в момент вызова хранимой процедуры;
- не нужно гадать по коду приложения, какой именно запрос должен получиться в том или ином месте программы, достаточно посмотреть на тело хранимой процедуры, что значительно упрощает процесс отладки /4/.

Особенности работы с хранимыми процедурами

1. Хранимые процедуры служат исключительно для доступа к данным (извлечение/ обновление/ удаление) и ни для чего больше. Использование процедур в иных целях является ошибкой /1/.

2. Хранимая процедура может вернуть более одного результата. В коде вызова хранимой процедуры необходимо делать итерацию по всем возвращаемым результатам и обрабатывать каждый из них в отдельности /1/.

3. Достаточно сложно передать в хранимую процедуру массив значений. Наиболее популярным решением является передача массива в хранимую процедуру в виде строки, содержащей элементы массива, разделенные специальным символом (вертикальная черта "|"), далее параметр анализируется в теле хранимой процедуры /1/.

4. Если в теле хранимой процедуры необходимо динамически генерировать SQL-запрос, то обязательно нужно экранировать кавычки и спецсимволы во всех переданных в процедуру параметрах, участвующих в построении запроса, например, следующим образом /6/:

```
SELECT * FROM table WHERE name = 'Д\'Артаньян'
```

Иначе процедура будет содержать потенциальную уязвимость типа SQL-injection.

Из всех перечисленных выше достоинств, недостатков и особенностей в применении хранимых процедур можно отметить, что в любом случае использовать их или нет - окончательное решение остается за разработчиком, и ответ здесь лишь один «все зависит от обстоятельств». Очевидно, что хранимые процедуры не являются универсальным средством решения всех проблем, но, без всякого сомнения, они являются наиважнейшим средством для создания программного обеспечения /7/.

Список литературы

1. Виейра Р. Программирование БД. MS SQL Server 2005. – М.: Диалектика, 2007. - С. 441-450.
2. Нильсен П. SQL Server 2005. Библия пользователя. – М.: Диалектика, 2008. - С. 465–478.
3. Хендерсон К. Профессиональное руководство по SQL Server: хранимые процедуры, XML, HTML. – СПб: Питер, 2005. - С. 30-41.
4. http://www.sql-school.info
5. http://www.sql.ru
6. http://www.phpfaq.ru
7. http://www.osp.ru. Моран Б. Стоит ли отказываться от хранимых процедур? // Системный администратор. – 2009. - № 1.