**Использование хранимых процедур без параметров**

Простейший вид хранимой процедуры SQL Server, которую можно вызвать, представляет собой хранимую процедуру, которая не содержит параметров и возвращает один результирующий набор. Драйвер Microsoft JDBC Driver для SQL Server содержит класс [SQLServerStatement](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/reference/sqlserverstatement-class?view=sql-server-ver15), который может использоваться для вызова такого вида хранимых процедур и обработки возвращаемых ими данных.

При использовании драйвера JDBC для вызова хранимых процедур без параметров необходимо использовать escape-последовательность SQL call. Синтаксис escape-последовательности call без параметров:

{call procedure-name}

**Примечание**

Дополнительные сведения об использовании escape-последовательностей в SQL см. в [**этой статье**](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/using-sql-escape-sequences?view=sql-server-ver15).

Для примера создайте следующую хранимую процедуру в образце базы данных [AdventureWorks](http://msftdbprodsamples.codeplex.com/):

SQL

CREATE PROCEDURE GetContactFormalNames

AS

BEGIN

SELECT TOP 10 Title + ' ' + FirstName + ' ' + LastName AS FormalName

FROM Person.Contact

END

Эта хранимая процедура возвращает один результирующий набор, который содержит один столбец данных, представляющий собой сочетание названия, имени, фамилии 10 основных контактов, которые хранятся в таблице Person.Contact.

В приведенном ниже примере открытое соединение с образцом базы данных [AdventureWorks](http://msftdbprodsamples.codeplex.com/) передается в функцию, а метод [executeQuery](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/reference/executequery-method-sqlserverstatement?view=sql-server-ver15) используется для вызова хранимой процедуры GetContactFormalNames.

Java

public static void executeSprocNoParams(Connection con) throws SQLException {

try(Statement stmt = con.createStatement();) {

ResultSet rs = stmt.executeQuery("{call dbo.GetContactFormalNames}");

while (rs.next()) {

System.out.println(rs.getString("FormalName"));

}

}

}

**Использование хранимых процедур с входными параметрами**

Хранимая процедура SQL Server, которую можно вызвать, содержит один или несколько параметров IN, то есть параметров, которые могут быть использованы для передачи данных в хранимую процедуру. В Microsoft JDBC Driver для SQL Server предусмотрен класс [SQLServerPreparedStatement](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/reference/sqlserverpreparedstatement-class?view=sql-server-ver15), который может быть использован для вызова этого вида хранимых процедур и обработки возвращаемых ими данных.

При использовании драйвера JDBC для вызова хранимой процедуры с параметрами IN следует использовать escape-последовательность SQL call вместе с методом [prepareCall](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/reference/preparecall-method-sqlserverconnection?view=sql-server-ver15) класса [SQLServerConnection](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/reference/sqlserverconnection-class?view=sql-server-ver15). Ниже приводится синтаксис escape-последовательности call с параметрами IN.

{call procedure-name[([parameter][,[parameter]]...)]}

**Примечание**

Дополнительные сведения об использовании escape-последовательностей в SQL см. в [**этой статье**](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/using-sql-escape-sequences?view=sql-server-ver15).

При создании escape-последовательности call укажите параметры IN при помощи (символ вопросительного знака (?)). Этот символ выполняет роль заполнителя для значений параметра, которые будут переданы хранимой процедуре. Чтобы указать значение для параметра, можно использовать один из методов задания из класса SQLServerPreparedStatement. Метод задания, который можно использовать, определяется типом данных параметра IN.

Во время передачи значения методу задания следует указать не только фактическое значение, которое будет использоваться в параметре, но также порядковый номер параметра в хранимой процедуре. Например, если хранимая процедура содержит единственный параметр IN, его порядковый номер будет 1. Если хранимая процедура содержит два параметра, порядковый номер первого значения будет 1, а второго — 2.

В качестве примера вызова хранимой процедуры, содержащей параметр IN, используйте хранимую процедуру uspGetEmployeeManagers в образце базы данных [AdventureWorks](http://msftdbprodsamples.codeplex.com/). Эта хранимая процедура принимает один параметр входных данных с именем EmployeeID, который является целым значением, и возвращает рекурсивный список служащих и их менеджеров на основе указанного EmployeeID. Ниже приводится код Java для вызова этой хранимой процедуры.

Java

public static void executeSprocInParams(Connection con) throws SQLException {

try(PreparedStatement pstmt = con.prepareStatement("{call dbo.uspGetEmployeeManagers(?)}"); ) {

pstmt.setInt(1, 50);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

System.out.println("EMPLOYEE:");

System.out.println(rs.getString("LastName") + ", " + rs.getString("FirstName"));

System.out.println("MANAGER:");

System.out.println(rs.getString("ManagerLastName") + ", " + rs.getString("ManagerFirstName"));

System.out.println();

}

}

}

**Использование хранимых процедур с выходными параметрами**

Вызываемая хранимая процедура SQL Server — это процедура, возвращающая один или несколько параметров OUT, т. е. параметров, используемых хранимой процедурой для возврата данных вызывающему приложению. Драйвер Microsoft JDBC Driver для SQL Server содержит класс [SQLServerCallableStatement](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/reference/sqlservercallablestatement-class?view=sql-server-ver15), который может быть использован для вызова этого типа хранимых процедур и обработки возвращаемых ими данных.

При вызове хранимой процедуры этого типа с помощью драйвера JDBC следует использовать escape-последовательность SQL call совместно с методом [prepareCall](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/reference/preparecall-method-sqlserverconnection?view=sql-server-ver15) класса [SQLServerConnection](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/reference/sqlserverconnection-class?view=sql-server-ver15). Ниже приводится синтаксис escape-последовательности call с параметрами OUT:

{call procedure-name[([parameter][,[parameter]]...)]}

**Примечание**

Дополнительные сведения об использовании escape-последовательностей в SQL см. в [**этой статье**](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/using-sql-escape-sequences?view=sql-server-ver15).

При создании escape-последовательности call укажите параметры OUT при помощи символа "?". (символ вопросительного знака (?)). Этот символ выполняет роль заполнителя для значений параметра, которые будут возвращены из хранимой процедуры. Чтобы указать значение параметра OUT, необходимо указать тип данных всех параметров с помощью метода [registerOutParameter](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/reference/registeroutparameter-method-sqlservercallablestatement?view=sql-server-ver15) класса SQLServerCallableStatement до выполнения хранимой процедуры.

Значение, указываемое для параметра OUT в методе registerOutParameter, должно представлять собой один из типов данных JDBC, содержащихся в java.sql.Types, который, в свою очередь, выполняет сопоставление с одним из собственных типов данных SQL Server. Дополнительные сведения о JDBC и типах данных SQL Server см. в статье [Основные сведения о типах данных драйвера JDBC](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/understanding-the-jdbc-driver-data-types?view=sql-server-ver15).

При передаче значения методу registerOutParameter для параметра OUT необходимо указать не только тип данных, который будет использоваться для параметра, но также порядковое размещение или имя параметра в хранимой процедуре. Например, если в хранимой процедуре имеется один параметр OUT, то первое порядковое значение будет 1, а второе порядковое значение — 2.

**Примечание**

Драйвер JDBC не поддерживает использование типов данных CURSOR, SQLVARIANT, TABLE и TIMESTAMP SQL Server в качестве параметров OUT.

Для примера создайте следующую хранимую процедуру в образце базы данных [AdventureWorks](http://msftdbprodsamples.codeplex.com/):

SQL

CREATE PROCEDURE GetImmediateManager

@employeeID INT,

@managerID INT OUTPUT

AS

BEGIN

SELECT @managerID = ManagerID

FROM HumanResources.Employee

WHERE EmployeeID = @employeeID

END

Эта хранимая процедура возвращает один параметр OUT (managerID), являющийся целым числом, на основе указанного параметра IN (employeeID), которое также является целым числом. Значение, возвращаемое в параметре OUT, представляет собой ManagerID, основанный на EmployeeID, который содержится в таблице HumanResources.Employee.

В следующем примере открытое соединение с образцом базы данных [AdventureWorks](http://msftdbprodsamples.codeplex.com/) передается в функцию, а метод [execute](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/reference/execute-method-sqlserverstatement?view=sql-server-ver15) используется для вызова хранимой процедуры GetImmediateManager:

Java

public static void executeStoredProcedure(Connection con) throws SQLException {

try(CallableStatement cstmt = con.prepareCall("{call dbo.GetImmediateManager(?, ?)}");) {

cstmt.setInt(1, 5);

cstmt.registerOutParameter(2, java.sql.Types.INTEGER);

cstmt.execute();

System.out.println("MANAGER ID: " + cstmt.getInt(2));

}

}

В примере для определения параметров используются порядковые местоположения. Или можно определить параметр при использовании имени вместо порядкового местоположения. В следующем примере кода изменяется предыдущий пример для иллюстрации использования именованных параметров в приложении Java. Обратите внимание, что имена параметров соответствуют именам параметров в определении хранимой процедуры:

JavaКопировать

public static void executeStoredProcedure(Connection con) throws SQLException {

try(CallableStatement cstmt = con.prepareCall("{call dbo.GetImmediateManager(?, ?)}"); ) {

cstmt.setInt("employeeID", 5);

cstmt.registerOutParameter("managerID", java.sql.Types.INTEGER);

cstmt.execute();

System.out.println("MANAGER ID: " + cstmt.getInt("managerID"));

}

}

**Примечание**

В этих примерах метод класса SQLServerCallableStatement используется для запуска хранимой процедуры. Он используется, поскольку хранимая процедура не возвратила результирующий набор. Если она возвратила результирующий набор, будет использован метод [**executeQuery**](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/reference/executequery-method-sqlserverstatement?view=sql-server-ver15).

Хранимые процедуры могут возвращать счетчики обновлений и несколько результирующих наборов. Драйвер Microsoft JDBC Driver для SQL Server соответствует спецификации JDBC 3.0, которая определяет, что множественные результирующие наборы и счетчики обновления должны быть получены до получения параметров OUT. Таким образом, приложение должно извлечь все объекты ResultSet и счетчики обновлений, прежде чем извлекать параметры OUT при использовании методов CallableStatement.getter. В противном случае объекты ResultSet и счетчики обновления, которые не были извлечены, будут потеряны при извлечении параметров OUT. Дополнительные сведения о счетчиках обновлений и нескольких результирующих наборах см. в статьях [Использование хранимых процедур со счетчиком обновлений](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/using-a-stored-procedure-with-an-update-count?view=sql-server-ver15) и [Использование нескольких результирующих наборов](https://docs.microsoft.com/ru-ru/sql/connect/jdbc/using-multiple-result-sets?view=sql-server-ver15).