Лабораторная работа 4

СОЗДАНИЕ ХРАНИМЫХ ПРОЦЕДУР В MICROSOFT SQL SERVER

Цель работы – научиться создавать и использовать хранимые процедуры на сервере БД.

Содержание работы:

1. Проработка примеров по ходу лабораторной работы.
2. Выполнение индивидуальных заданий по вариантам.

Пояснения к выполнению работы

Для освоения программирования хранимых процедур используем пример базы данных с названием **Books**.

Хранимые процедуры представляют собой набор команд, состоящий из одного или нескольких операторов SQL или функций и сохраняемый в базе данных в откомпилированном виде.

Типы хранимых процедур

Системные хранимые процедуры предназначены для выполнения различных административных действий. Практически все действия по администрированию сервера выполняются с их помощью. Можно сказать, что системные хранимые процедуры являются интерфейсом, обеспечивающим работу с системными таблицами. Системные хранимые процедуры имеют префикс sp_, хранятся в системной базе данных и могут быть вызваны в контексте любой другой базы данных.

Пользовательские *хранимые процедуры* реализуют те или иные действия. *Хранимые процедуры* – полноценный объект базы данных. Вследствие этого каждая *хранимая процедура* располагается в конкретной базе данных, где и выполняется.

Временные *хранимые процедуры* существуют лишь некоторое время, после чего автоматически уничтожаются сервером. Они делятся на локальные и глобальные. Локальные временные *хранимые процедуры* могут быть вызваны только из того соединения, в котором созданы. При создании такой процедуры ей необходимо дать имя, начинающееся с одного символа #. Как и все временные объекты, *хранимые процедуры* этого типа автоматически удаляются при отключении пользователя, перезапуске или остановке сервера. Глобальные временные *хранимые процедуры* доступны для любых соединений сервера, на котором имеется такая же процедура. Для ее определения достаточно дать ей имя, начинающееся с символов ##. Удаляются эти процедуры при перезапуске или остановке сервера, а также при закрытии соединения, в контексте которого они были созданы.

Создание, изменение хранимых процедур

При создании хранимой процедуры следует учитывать, что она будет иметь те же права доступа к объектам базы данных, что и создавший ее пользователь; определение параметров хранимой процедуры, хранимые процедуры могут обладать входными и выходными параметрами; разработка кода хранимой процедуры. Код процедуры может со-

держать последовательность любых команд SQL, включая вызов других хранимых процедур.

Синтаксис оператора создания новой или изменения имеющейся хранимой процедуры в обозначениях MS SQL Server:

```
{CREATE | ALTER PROC[EDURE] имя_процедуры [;номер] [{@имя_параметра тип_данных } [VARYING ] [=default][OUTPUT] ][,...n] [WITH { RECOMPILE | ENCRYPTION }] [FOR REPLICATION] AS sql_оператор [...n]
```

Рассмотрим параметры данной команды.

Используя префиксы sp, #, ##, создаваемую процедуру можно определить в качестве системной или временной. Как видно из синтаксиса команды, не допускается указывать имя владельца, которому будет принадлежать создаваемая процедура, а также имя базы данных, где она должна быть размещена. Таким образом, чтобы разместить создаваемую хранимую процедуру в конкретной базе данных, необходимо выполнить команду CREATE PROCEDURE в контексте этой базы данных. При обращении из тела хранимой процедуры к объектам той же базы данных можно использовать укороченные имена, т. е. без указания имени базы данных. Когда же требуется обратиться к объектам, расположенным в других базах данных, указание имени базы данных обязательно.

Для передачи входных и выходных данных в создаваемой хранимой процедуре имена параметров должны начинаться с символа @. В одной хранимой процедуре можно задать множество параметров, разделенных запятыми. В теле процедуры не должны применяться локальные переменные, чьи имена совпадают с именами параметров этой процедуры.

Для определения типа данных параметров хранимой процедуры подходят любые типы данных SQL, включая определенные пользователем. Однако тип данных CURSOR может быть использован только как выходной параметр хранимой процедуры, т.е. с указанием ключевого слова OUTPUT.

Наличие ключевого слова OUTPUT означает, что соответствующий параметр предназначен для возвращения данных из хранимой процедуры. Однако это вовсе не означает, что параметр не подходит для передачи значений в хранимую процедуру. Указание ключевого слова OUTPUT предписывает серверу при выходе из хранимой процедуры присвоить текущее значение параметра локальной переменной, которая была указана при вызове процедуры в качестве значения параметра. Отметим, что при указании ключевого слова OUTPUT значение соответствующего параметра при вызове процедуры может быть задано только с помощью локальной переменной. Не разрешается использование любых выражений или констант, допустимое для обычных параметров.

Ключевое слово VARYING применяется совместно с параметром OUTPUT, имеющим тип CURSOR. Оно определяет, что выходным параметром будет результирующее множество.

Ключевое слово DEFAULT представляет собой значение, которое будет принимать соответствующий параметр по умолчанию. Таким образом, при вызове процедуры можно не указывать явно значение соответствующего параметра.

Так как сервер кэширует план исполнения запроса и компилированный код, при последующем вызове процедуры будут использоваться уже готовые значения. Однако в

некоторых случаях все же требуется выполнять перекомпиляцию кода процедуры. Указание ключевого слова RECOMPILE предписывает системе создавать план выполнения хранимой процедуры при каждом ее вызове.

Параметр FOR REPLICATION востребован при репликации данных и включении создаваемой хранимой процедуры в качестве статьи в публикацию.

Ключевое слово ENCRYPTION предписывает серверу выполнить шифрование кода хранимой процедуры, что может обеспечить защиту от использования авторских алгоритмов, реализующих работу хранимой процедуры.

Ключевое слово AS размещается в начале собственно тела хранимой процедуры. В теле процедуры могут применяться практически все команды SQL, объявляться транзакции, устанавливаться блокировки и вызываться другие хранимые процедуры. Выход из хранимой процедуры можно осуществить посредством команды RETURN.

Удаление хранимой процедуры

DROP PROCEDURE {имя_процедуры} [,...n]

Выполнение хранимой процедуры

Для выполнения хранимой процедуры используется команда:

```
[[ EXEC[UTE] имя_процедуры [;номер] [[@имя_параметра=]{значение
| @имя_переменной} [OUTPUT ]|[DEFAULT ]][,...n]
```

Если вызов хранимой процедуры не является единственной командой в пакете, то присутствие команды EXECUTE обязательно. Более того, эта команда требуется для вызова процедуры из тела другой процедуры или триггера.

Использование ключевого слова OUTPUT при вызове процедуры разрешается только для параметров, которые были объявлены при создании процедуры с ключевым словом OUTPUT.

Когда же при вызове процедуры для параметра указывается ключевое слово DEFAULT, то будет использовано значение по умолчанию. Естественно, указанное слово DEFAULT разрешается только для тех параметров, для которых определено значение по умолчанию.

Из синтаксиса команды EXECUTE видно, что имена параметров могут быть опущены при вызове процедуры. Однако в этом случае пользователь должен указывать значения для параметров в том же порядке, в каком они перечислялись при создании процедуры. Присвоить параметру значение по умолчанию, просто пропустив его при перечислении, нельзя. Если же требуется опустить параметры, для которых определено значение по умолчанию, достаточно явного указания имен параметров при вызове хранимой процедуры. Более того, таким способом можно перечислять параметры и их значения в произвольном порядке.

Отметим, что при вызове процедуры указываются либо имена параметров со значениями, либо только значения без имени параметра. Их комбинирование не допускается.

Использование RETURN в хранимой процедуре

Позволяет выйти из процедуры в любой точке по указанному условию, а также позволяет передать результат выполнения процедуры числом, по которому можно судить о

качестве и правильности выполнения процедуры. Пример создания процедуры без параметров:

```
CREATE PROCEDURE Count_Books AS
Select count(ISBN) from EXEMPLAR
Go
```

Пример создания процедуры с входным параметром:

```
CREATE PROCEDURE Count_Books_Pages @Count_pages as Int AS
Select count(ISBN) from BOOKS WHERE Pages>=@Count_pages
Go
```

Вызывается процедура следующим образом:

```
EXEC Count_Books_Pages 100
```

Пример создания процедуры с входными параметрами:

```
CREATE PROCEDURE Count_Books_Title @Count_pages as Int, @Title AS
Char(10)
AS
Select count(ISBN) from Books WHERE Pages>=@Count_pages AND Title
LIKE @Title
Go
```

```
EXEC Count_Books_Title 100, 'П%'
```

Пример создания процедуры с входными параметрами и выходным параметром:

```
CREATE PROCEDURE Count_Books_Itogo @Count_pages Int, @Title
Char(10), @Itogo Int OUTPUT
AS
Select @Itogo = count(ISBN) from Books
WHERE Pages>=@Count_pages AND Title LIKE @Title
Go
```

Запуск производится с помощью набора команд:

```
Declare @q As int
EXEC Count_Books_Itogo 100, 'П%', @q output select @q
```

Пример создания процедуры с входными параметрами и RETURN:

```
CREATE PROCEDURE checkname @param int
AS
IF (SELECT author FROM books WHERE author = @param) = 'Пушкин А.С.'
RETURN 1
ELSE
RETURN 2
```

Запуск производится с помощью набора команд:

```
DECLARE @return_status int
EXEC @return_status = checkname 1 SELECT 'Return Status' =
@return_status
```

Пример создания процедуры без параметров для увеличения значения ключевого поля в таблице Exemplar в 2 раза:

```
CREATE PROC update_proc
AS
UPDATE EXEMPLAR SET ID_Exemplar = ID_Exemplar*2
```

Процедура не возвращает никаких данных.

```
EXEC update_proc
```

Пример процедуры с входным параметром для получения всей информации о конкретном читателе:

```
CREATE PROC select_reader @k CHAR(30)
AS
SELECT * FROM Readers WHERE LAST_NAME=@k
```

```
EXEC select_author 'Петров'
```
или
```
EXEC select_author @k='Петров'
```

Пример создания процедуры с входным параметром и значением по умолчанию для увеличения значения ключевого поля в таблице Purchases в заданное количество раза (по умолчанию в 2 раза):

```
CREATE PROC update_proc @p INT = 2
AS
UPDATE EXEMPLAR SET ID_Exemplar = ID_Exemplar*@p
```

Процедура не возвращает никаких данных.

```
EXEC update_proc 4 или
EXEC update_proc @p = 4 или
EXEC update_proc --будет использовано значение по умолчанию.
```

Пример создания процедуры с входным и выходным параметрами. Создать процедуру для определения количества книг, выданных за указанный период:

```
CREATE PROC count_books
@d1 SMALLDATETIME, @d2 SMALLDATETIME,
@c INT OUTPUT AS
SELECT @c=count(ISBN) from EXEMLAR
WHERE DATA_IN BETWEEN @d1 AND @d2
SET @c = ISNULL(@c,0)
```

```
DECLARE @c2 INT
EXEC count_books '01-jun-2017', '01-jul-2017', @c2
OUTPUT SELECT @c2
```

Варианты заданий к лабораторной работе

Общие положения

В утилите SQL Server Management Studio создать новую страницу для кода (кнопка «Создать запрос»). Программно сделать активной свою БД с помощью оператора Use. Создать хранимые процедуры с помощью операторов Create procedure, причем самостоятельно определить имена процедур. Каждая процедура будет выполнять SQL запросы. Причем код SQL запросов нужно изменить таким образом, чтобы в них можно было передавать значения полей, по которым осуществляется поиск.

Например, исходное задание и запрос:

/*Выбрать из справочника поставщиков (таблица Deliveries) названия компаний, телефоны и ИНН (поля Name_company, Phone и INN), у которых название компании (поле Name_company) 'ОАО МИР'.

SELECT Name_company, Phone, INN FROM Deliveries WHERE

Name_company = 'ОАО МИР'

*/

--В данной работе будет создана процедура:

```
CREATE PROC select_name_company @comp CHAR(30) AS
SELECT Name_company, Phone, INN FROM Deliveries WHERE Name_company
= @comp
```

--Для запуска процедуры используется команда: EXEC select_name_company 'ОАО МИР'

Список заданий

Задания выполнять для индивидуальной БД. Если задания выполняются для БД из курсовой работы, необходимо согласовать их с преподавателем до выполнения.

Вариант 1

1. Получить список экзаменов на выбранную дату. Если не указана дата, получать на последнюю дату, когда были экзамены.
2. Проверить корректность расписания экзаменов (уникальность комбинации "время – дата – аудитория"; между экзаменами в одной группе должно пройти не менее трёх дней).
3. Организовать ввод данных в таблицу расписания экзаменов, учитывая группы, кафедры, отдельных преподавателей. При этом рекомендуется использовать предыдущую процедуру корректности расписания.
4. Получить отчет по результатам зачётов и экзаменов итоговых значений (количество оценок «Отлично», «Хорошо», «Удовлетворительно» или A, B, C, D, E; количество неявок; средний балл по группе). Вид оценок задать входным параметром.

   Оценки по 100-балльной шкале: A 90-100 5 (отлично); B 80-89 4 (хорошо); C 75-79 4 (хорошо); D 70-74 3 (удовлетворительно); E 60-69 3 (удовлетворительно).

Вариант 2

1. Получить список людей предпенсионного возраста (не более 2-х лет до пенсии).
2. Получить список сотрудников с количеством детей каждого.
3. Получить список ветеранов (работающих в институте не менее указанного количества лет). По умолчанию количество отработанных лет – 30. Отдельно отметить среди них юбиляров текущего года.
4. Организовать ввод данных заполнения ставок с проверкой свободных вакансий (не должно быть заполнено ставок больше, чем их свободно)

Вариант 3

1. Получить список проданных и сданных в аренду помещений с учетом за указанный период. Если период не указан – за все время.
2. Получить список помещений, предназначенных для аренды или продажи. Входные параметры процедуры должны учитывать жилые и нежилые помещения, по умолчанию – все помещения.
3. Получить среднюю стоимость аренды/продажи помещений по районам. Район получать в виде входного параметра, если не указан – для всех районов.
4. Организовать ввод информации по ведению списков жилых и нежилых помещений, предназначенных для аренды и/или продажи (использовать вставку или изменение строк в соответствующие таблицы).

Вариант 4

1. Получить список студентов, выполняющих курсовые и дипломные работы (по группам и по руководителям). Тип работы (курсовая или дипломная), группу и преподавателя получить в виде входных параметров. По умолчанию – получать список дипломников для всех групп и руководителей.
2. Получить расписание занятий на семестр по группам. Группу получать в виде входного параметра, по умолчанию – для всех групп.
3. Получить расписание для преподавателей. Преподавателя получать в виде входного параметра.
4. Организовать ввод информации по курсовым и дипломным работам с учетом типа работы (курсовая, ВКР или магистерская диссертация) студента, руководителя, темы работы (использовать вставку или изменение строк в соответствующие таблицы).

Вариант 5

1. Организовать поиск литературы по требуемым разделу, теме, автору, ключевому слову, с заданием интересующего периода. Критерии поиска задавать в виде входных параметров. Если параметр не указан – не использовать соответствующий критерий поиска.
2. Построить рейтинг изданий (по количеству читателей и дате последней выдачи).
3. Получить список должников по месяцам и годам. Месяцы или годы передавать в виде параметров.
4. Организовать ввод информации по выдаче литературы (использовать вставку или изменение строк в соответствующие таблицы).

Вариант 6

1. Получить наличие свободных мест в палатах отдельно для мужчин и для женщин. Учет «мужчина или женщина» организовать через входной параметр.
2. Получить количество прооперированных пациентов (из них – с осложнениями и умерших);
3. Получить пациентов, которым были оказаны платные услуги с указанием вида услуги и ее стоимости. Организовать с помощью входных параметров ввод пациента и вид услуги. По умолчанию – для всех пациентов и все виды услуг.
4. Организовать ввод информации поступления пациентов по отделениям (использовать вставку или изменение строк в соответствующие таблицы).

Вариант 7

1. Получить список товаров, запасы которых подходят к концу.
2. Определить 3 лучших продавца указанного месяца. Месяц получать в виде входного параметра, по умолчанию – последний перед текущим (предыдущий).
3. Получить список лучших поставщиков за период. Период получать в виде входного параметра, по умолчанию – за все время.

4. Организовать ввод информации по продажам с учетом проданного товара, цены, количества, отдела, продавца, даты, поставщика (если есть такая возможность) и др. (использовать вставку или изменение строк в соответствующие таблицы).

Вариант 8

1. Получить список номеров, освобождающихся на указанную дату. Если дата не указана, получить список на сегодня и завтра.
2. Проверить наличие брони по имени клиента и/или названию организации. Если имя или название не указаны, то показать все забронированные номера.
3. Выдать информации по конкретному номеру. Если номер не указан, - по последнему освобожденному.
4. Организовать ввод информации по заселению в номер с учетом постояльца, бронирования и пр. (использовать вставку или изменение строк в соответствующие таблицы).

Вариант 9

1. Найти свободное место на рейс по следующим требованиям: дата, класс, номер рейса, конечная цель поездки. Требования оформить в виде входных параметров. Если параметр не указан, то не использовать соответствующий критерий поиска.
2. Проверить наличие брони по имени клиента и/или названию организации. Если не указывать имени клиента или названия, то показать всю наличную бронь.
3. Получить информацию по конкретному рейсу. Если параметр не указан, получить информацию по ближайшему по времени рейсу.
4. Организовать ввод информации по продаже билетов в оба конца (использовать вставку или изменение строк в соответствующие таблицы).

Вариант 10

1. Получить информацию по соревнованиям за указанный период. Если период не указан, то получить информацию за последний месяц.
2. Подобрать возможные кандидатуры на участие в соревнованиях (вида спорта, соответствующего уровня мастерства, возраста и без травм). Требования оформить в виде входных параметров. Параметр «вид спорта» обязателен к заданию. Остальные параметры могут быть не указаны, в таком случае не использовать соответствующий критерий поиска.
3. Получить информацию по конкретному спортсмену. Если конкретный спортсмен не указан, получить информацию по самому рейтинговому спортсмену.
4. Организовать ввод информации о проведении соревнования (использовать вставку или изменение строк в соответствующие таблицы).

Вариант 11

1. Получить список туров в определенную страну. Если страна не определена, получить полный список туров с группировкой по странам.
2. Получить сведения о фирмах, имеющих путевки в определенную страну с применением фильтра цен. Организовать ввод страны и фильтра цен в виде параметров. Если фильтр цен не указан, показать сведения без учета фильтра цен.
3. Получить информацию об определенной туристической фирме с учетом реализуемых путевок. Организовать ввод фирмы в виде входного параметра. Если параметр не указан, получить информацию о фирме с самым большим набором предложений.
4. Организовать ввод информации по картотеке турфирм (использовать вставку или изменение строк в соответствующие таблицы).

Вариант 12

1. Получить информацию о том, за кем из читателей закреплены книги, количество

экземпляров которых в библиотеке не превышает указанного числа. По умолчанию это число равно 2.

2. Получить информацию, сколько читателей в процентном отношении имеют начальное образование, среднее, высшее.
3. Получить информацию об определенном читателе с учетом того, какие книги закреплены за ним. Если читатель не определен, получить информацию о последнем обслуженном читателе.
4. Организовать ввод информации о приеме книги в фонд библиотеки (использовать вставку или изменение строк в соответствующие таблицы).

Контрольные вопросы

1. Что такое хранимая процедура?
2. Где выполняются хранимые процедуры?
3. Как активизируются хранимые процедуры?
4. В чем преимущества использования хранимых процедур?
5. Какие типы хранимых процедур имеются в SQL Server?