**Лабораторная работа № 10. Обработка результатов запросов с помощью курсоров**

**КУРНОСЕНКО СОФЬЯ**

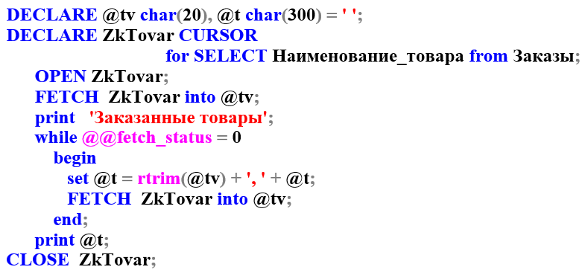
**Курсор** является программной конструкцией, которая дает возможность пользователю обрабатывать строки результирующего набора запись за записью. Курсоры бывают *локальные* и *глобальные* (по умолчанию), *статические* и *динамические* (по умолчанию).

1.Разработать сценарий, формирующий список дисциплин на кафедре ИСиТ. В отчет должны быть выведены краткие названия (поле SUBJECT) из таблицы SUBJECT в одну строку через запятую.

Использовать встроенную функцию RTRIM.

***→ Теория (методичка)***

Пусть требуется создать список заказанных товаров из таблицы **Заказы**. Наименования товаров должны выводиться в одну строку через запятую. Курсор объявляется в операторе DECLARE:



Курсор открывается с помощью оператора OPEN.

Оператор FETCH считывает одну строку из результирующего набора и продвигает указатель на следующую строку. Количество переменных в списке после ключевого слова INTO должно быть равно количеству столбцов результирующего набора, а порядок их должен соответствовать порядку перечисления столбцов в SELECT-списке.

После выполнения FETCH проверяется значение функции **@@fetch\_status**, которая возвращает значение 0, если оператор FETCH выполнен успешно; −1, если достигнут конец результирующего набора и строка не считывается; −2, если выбранная строка отсутствует в БД. В зависимости от полученного результата цикл продолжается и считывается следующая строка, или цикл заканчивается.

Курсор закрывается с помощью оператора CLOSE.

***→ Выполнение заданий***

-- 1

declare subjectCur cursor

for select SUBJECT from SUBJECT

where PULPIT = 'ИСиТ';

declare @subject char(10), @allSubjects char(300) = '';

open subjectCur;

fetch subjectCur into @subject;

print 'Дисциплины на кафедре ИСиТ:' + char(10);

while @@fetch\_status = 0

begin

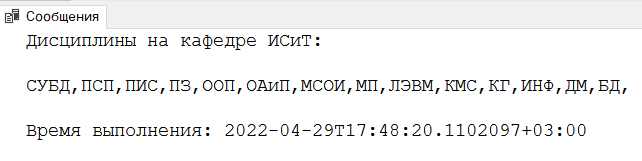
set @allSubjects = rtrim(@subject) + ','+ @allSubjects;

fetch subjectCur into @subject;

end;

print @allSubjects;

close subjectCur;



2. Разработать сценарий, демонстрирующий отличие глобального курсора от локального на примере базы данных X\_UNIVER.

***→ Теория (методичка)***

*Локальный* курсор может применяться в рамках одного пакета и ресурсы, выделенные ему при объявлении, освобождаются сразу после завершения работы пакета.

**DECLARE Tovary CURSOR LOCAL**

**for SELECT Наименование, Цена from Товары;**

**DECLARE @tv char(20), @cena real;**

**OPEN Tovary;**

**fetch Tovary into @tv, @cena;**

**print '1. '+@tv+cast(@cena as varchar(6));**

**go**

**DECLARE @tv char(20), @cena real;**

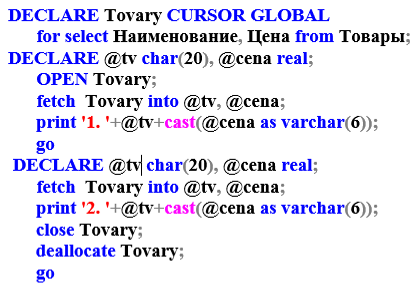
**fetch Tovary into @tv, @cena;**

**print '2. '+@tv+cast(@cena as varchar(6));**

**go**

Выполняется первый пакет, а при попытке выполнить следующий появляется сообщение, что курсора с именем **Tovary** не существует.

*Глобальный* курсор может быть объявлен, открыт и использован в разных пакетах. Выделенные ему при объявлении ресурсы освобождаются только после выполнения оператора DEALLOCATE или при завершении сеанса пользователя.



***→ Выполнение заданий***

-- локальный курсор

declare locCur cursor local

for

select PULPIT, PULPIT\_NAME

from PULPIT;

declare @pul char(20), @pulName varchar(100);

open locCur;

fetch locCur into @pul, @pulName;

print '1. ' + @pul + ' ' + @pulName;

go

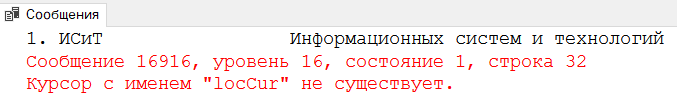
------------------------------------------

declare @pul char(20), @pulName varchar(100);

fetch locCur into @pul, @pulName;

print '2. ' + @pul + ' ' + @pulName;

go

******

-- глобальный

declare globCur cursor global

for

select PULPIT, PULPIT\_NAME

from PULPIT;

declare @pul char(20), @pulName varchar(100);

open globCur;

fetch globCur into @pul, @pulName;

print '1. ' + @pul + ' ' + @pulName;

go

------------------------------------------

declare @pul char(20), @pulName varchar(100);

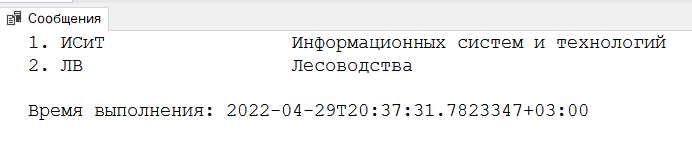
fetch globCur into @pul, @pulName;

print '2. ' + @pul + ' ' + @pulName;

close globCur;

deallocate globCur; -- удаления курсора и освобождения памяти, процессора и других системных ресурсов, выделенных для курсора

go

******

3. Разработать сценарий, демонстрирующий отличие статических курсоров от динамических на примере базы данных X\_UNIVER.

***→ Теория (методичка)***

Открытие *статического* курсора приводит к выгрузке результирующего набора во временную таблицу системной БД **TEMPDB**, и все дальнейшие операции осуществляются с этой таблицей.

После открытия курсора все текущие изменения в исходных таблицах не будут отражаться в результирующем наборе.

**DECLARE @tid char(10), @tnm char(40), @tgn char(1);**

**DECLARE Zakaz CURSOR LOCAL STATIC**

**for SELECT Наименование\_товара, Цена\_продажи, Количество**

**FROM dbo.Заказы where Заказчик = 'Луч';**

**open Zakaz;**

**print 'Количество строк : '+cast(@@CURSOR\_ROWS as varchar(5));**

**UPDATE Заказы set Количество = 5 where Наименование\_товара = 'Стул';**

**DELETE Заказы where Наименование\_товара = 'Шкаф';**

**INSERT Заказы (Номер\_заказа, Наименование\_товара, Цена\_продажи,**

**Количество, Дата\_поставки, Заказчик)**

**values (18, 'Шкаф', 340, 1, '2014-08-02', 'Луч');**

**FETCH Zakaz into @tid, @tnm, @tgn;**

**while @@fetch\_status = 0**

**begin**

**print @tid + ' '+ @tnm + ' '+ @tgn;**

**fetch Zakaz into @tid, @tnm, @tgn;**

**end;**

**CLOSE Zakaz;**

Здесь значение функции @@CURSOR\_ROWS равно **−n** (количество записей) при асинхронной выборке, равно **n** при синхронной выборке, равно **0**, если курсор не открыт. При выполнении курсора все изменения (UPDATE, DELETE и INSERT) в исходной таблице **Заказы** не отражаются на результате выборки строк.

Если заменить LOCAL STATICна LOCAL DYNAMIC, то изменения будут отражаться в результирующем наборе.

***→ Выполнение заданий***

Курсоры объявляем с атрибутом scroll (вообще, он установлен по умолчанию), чтобы можно было двигаться по записям курсора не только от первой записи до последней, но и, как в нашем случае, считать последнюю запись.

-- статический курсор

declare studentStaticCur cursor static scroll

for select IDSTUDENT, NAME

from STUDENT;

-- переменные для статического курсора

declare @idForStatic int, @nameForStatic nvarchar(100); -- переменные, куда будут считываться значения курсора

open studentStaticCur; -- открываем статический курсор

-- модифицируем таблицу (новая строка попадет в конец таблицы)

insert into STUDENT(NAME)

values('Курносенко Софья Андреевна');

print 'В таблицу добавлена строка ''Курносенко Софья Андреевна''!';

print 'Последняя строка таблицы STUDENT по версии статического курсора:';

-- считываем последнюю запись в статическом курсоре

fetch last from studentStaticCur into @idForStatic, @nameForStatic;

print cast(@idForStatic as varchar(7)) + ' ' + @nameForStatic;

close studentStaticCur;

----------------------------------------------------

-- динамический курсор

declare studentDynamicCur cursor dynamic scroll

for select IDSTUDENT, NAME

from STUDENT;

declare @idForDynamic int, @nameForDynamic nvarchar(100); -- переменные, куда будут считываться значения курсора

open studentDynamicCur; -- открываем динамический курсор

-- модифицируем таблицу (новая строка попадет в конец таблицы)

insert into STUDENT(NAME)

values('Олимпиева Ольга Денисовна');

print 'В таблицу добавлена строка ''Олимпиева Ольга Денисовна''!';

print 'Последняя строка таблицы STUDENT по версии динамического курсора:';

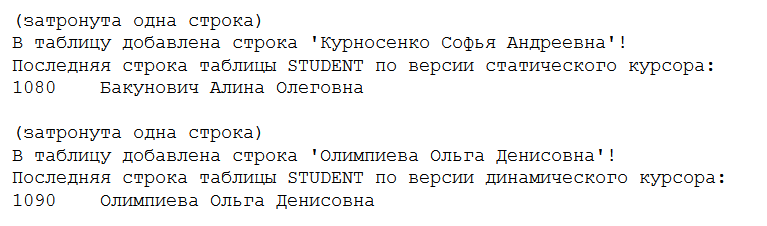
-- считываем последнюю запись в динамическом курсоре

fetch last from studentDynamicCur into @idForDynamic, @nameForDynamic;

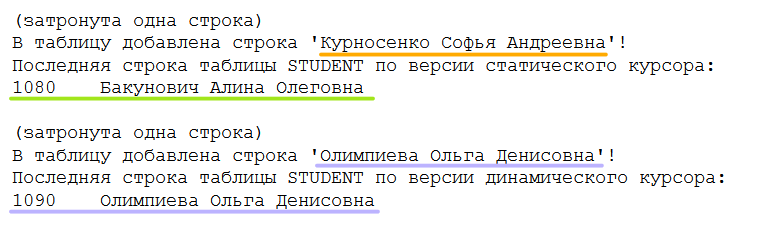
print cast(@idForDynamic as varchar(7)) + ' ' + @nameForDynamic;

close studentDynamicCur;

deallocate studentDynamicCur;

******

Видно, что статический курсор изменений не отразил, в отличии от динамического:

******

4. Разработать сценарий, демонстрирующий свойства навигации в результирующем наборе курсора с атрибутом SCROLL на примере базы данных X\_UNIVER.

Использовать все известные ключевые слова в операторе FETCH.

***→ Теория (методичка)***

По умолчанию для курсора установлен атрибут SCROLL, позволяющий применять оператор FETCH с дополнительными опциями позиционирования.

В примере ниже выводятся названия товаров, которые заказаны фирмой с названием «Луч». Перед каждым названием выводится номер строки результирующего набора, определяемый функцией ROW\_NUMBER(). Выбор строки определяется соответствующим ключевым словом в операторе FETCH.

**DECLARE @tc int, @rn char(50);**

**DECLARE Primer1 cursor local dynamic SCROLL**

**for SELECT row\_number() over (order by Наименование\_товара) N,**

**Наименование\_товара FROM dbo.Заказы**

**where Заказчик = 'Луч'**

**OPEN Primer1;**

**FETCH Primer1 into @tc, @rn;**

**print 'следующая строка : ' + cast(@tc as varchar(3))+ rtrim(@rn);**

**FETCH LAST from Primer1 into @tc, @rn;**

**print 'последняя строка : ' + cast(@tc as varchar(3))+ rtrim(@rn);**

**. . . . . . . . . . . . . . . . . . . . . . . . .**

**CLOSE Primer1;**

Можно дописать этот пример, используя другие ключевые слова: FIRST (первая строка), NEXT (следующая строка за текущей), PRIOR (предыдущая строка от текущей), ABSOLUTE 3 (третья строка от начала), ABSOLUTE −3 (третья строка от конца), RELATIVE 5 (пятая строка вперед от текущей), RELATIVE −5 (пятая строка назад от текущей).

***→ Выполнение заданий***

-- создаем переменные для хранения строк курсора

declare @pul varchar(20),

@allPulpits varchar(100) = '',

@rowNum int;

-- создаем курсор

declare curTask\_4 cursor scroll

for select row\_number() over (order by PULPIT) as [Номер строки],

PULPIT as [Кафедра]

from PULPIT;

-- выведем все кафедры из таблицы PULPIT

open curTask\_4;

print 'Все кафедры из таблицы PULPIT :';

fetch curTask\_4 into @rowNum, @pul;

while @@fetch\_status = 0

begin

set @allPulpits = @allPulpits + rtrim(@pul) + ',';

fetch curTask\_4 into @rowNum, @pul;

end;

print @allPulpits;

close curTask\_4;

-- применение атрибута scroll

open curTask\_4;

fetch first from curTask\_4 into @rowNum, @pul;

print char(10) + 'Первая запись : ' +

cast(@rowNum as varchar(10)) + ' -> ' + @pul;

fetch next from curTask\_4 into @rowNum, @pul;

print 'Следующая строка от текущей : ' +

cast(@rowNum as varchar(10)) + ' -> ' + @pul;

fetch last from curTask\_4 into @rowNum, @pul;

print 'Последняя запись : ' +

cast(@rowNum as varchar(10)) + ' -> ' + @pul;

fetch prior from curTask\_4 into @rowNum, @pul;

print 'Предыдущая строка от текущей : ' +

cast(@rowNum as varchar(10)) + ' -> ' + @pul;

fetch absolute 5 from curTask\_4 into @rowNum, @pul;

print 'Пятая запись с начала : ' +

cast(@rowNum as varchar(10)) + ' -> ' + @pul;

fetch absolute -5 from curTask\_4 into @rowNum, @pul;

print 'Пятая запись с конца : ' +

cast(@rowNum as varchar(10)) + ' -> ' + @pul;

fetch relative 3 from curTask\_4 into @rowNum, @pul;

print 'Третья строка вперед от текущей : ' +

cast(@rowNum as varchar(10)) + ' -> ' + @pul;

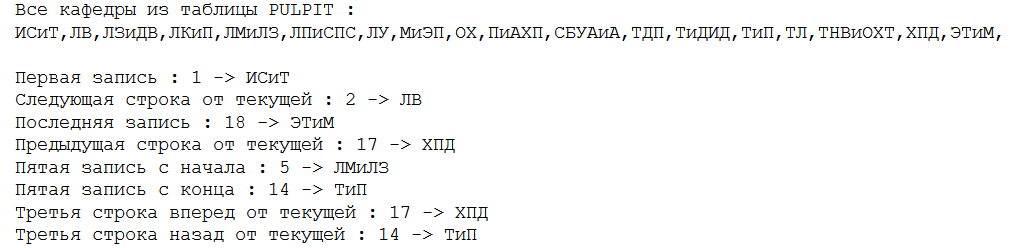
fetch relative -3 from curTask\_4 into @rowNum, @pul;

print 'Третья строка назад от текущей : ' +

cast(@rowNum as varchar(10)) + ' -> ' + @pul;

close curTask\_4;

deallocate curTask\_4;

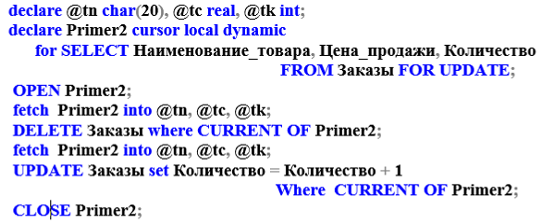
******

5. Создать курсор, демонстрирующий применение конструкции CURRENT OF в секции WHERE с использованием операторов UPDATE и DELETE.

***→ Теория (методичка)***

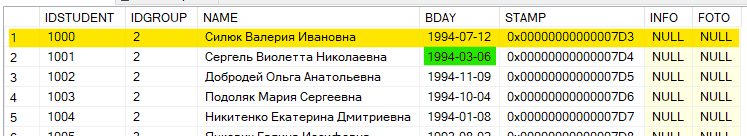
Курсоры с установленным свойством FOR UPDATE помимо чтения данных из строк с помощью оператора FETCH, могут эти строки изменять или удалять с помощью операторов UPDATE и DELETE, если в секции WHERE эти операторы используют операцию CURRENT OF, для которой указывается имя курсора (в примере ниже Primer2). Такой формат операторов позволяет удалять или изменять строки в таблице (в примере таблица **Заказы**), соответствующих *текущей* позиции курсора в результирующем наборе.

В примере удаляется первая строка в таблице **Заказы** и увеличивается на единицу количество товаров в следующей строке.



***→ Выполнение заданий***

Первые записи в таблице до редактирования посредством курсора:

******

declare @bday date;

declare curTask\_5 cursor local dynamic

for select BDAY

from STUDENT for update;

open curTask\_5;

fetch curTask\_5 into @bday;

delete STUDENT where current of curTask\_5; -- удаляем из таблицы STUDENT запись, считанную курсором

fetch curTask\_5 into @bday;

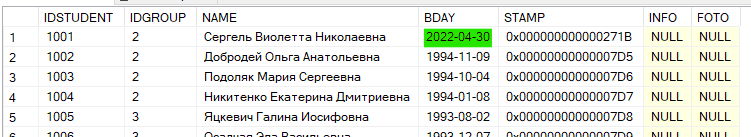
update STUDENT

set BDAY = getdate()

where current of curTask\_5; -- изменяем в таблице STUDENT запись, считанную курсором

close curTask\_5;

После:

******

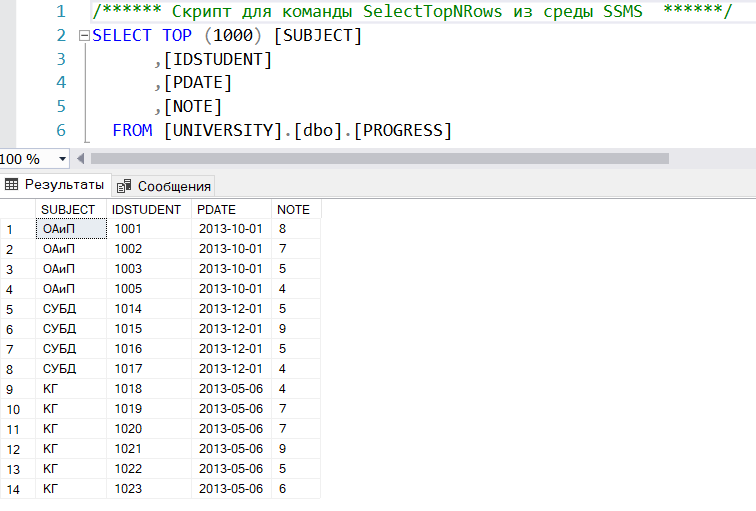
6. Разработать SELECT-запрос, с помощью которого из таблицы PROGRESS удаляются строки, содержащие информацию о студентах, получивших оценки ниже 4 (использовать объединение таблиц PROGRESS, STUDENT, GROUPS).

Разработать SELECT-запрос, с помощью которого в таблице PROGRESS для студента с конкретным номером IDSTUDENT корректируется оценка (увеличивается на единицу).

***→ Выполнение заданий***

***6.1***

В таблице PROGRESS нет студентов с оценкой ниже 4:



Буду удалять студентов с оценкой 4, затем восстановлю исходные таблицы:

-- в таблице PROGRESS нет студентов с оценкой ниже 4

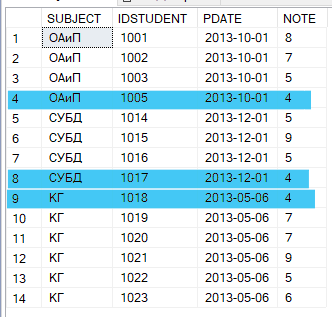
-- так что буду удалять студентов с оценкой равной 4

-- перед этим сделаю копию таблицы PROGRESS, чтобы потом восстановить все как было

select \* into PROGRESS\_copy

from PROGRESS;

Студенты с ИД 1005, 1017, 1018 будут удалены из таблицы PROGRESS, так как их оценки равны 4:



Создаем курсор и удаляем студентов:

-- курсор

declare curTask\_6\_1 cursor local dynamic

for select STUDENT.NAME as [ФИО],

PROGRESS.NOTE as [Оценка]

from STUDENT inner join PROGRESS

on STUDENT.IDSTUDENT = PROGRESS.IDSTUDENT

inner join GROUPS

on STUDENT.IDGROUP = GROUPS.IDGROUP

where PROGRESS.NOTE = 4

for update;

-- удаление из таблицы PROGRESS информации о студентах с баллом = 4

open curTask\_6\_1;

fetch curTask\_6\_1;

while @@fetch\_status = 0

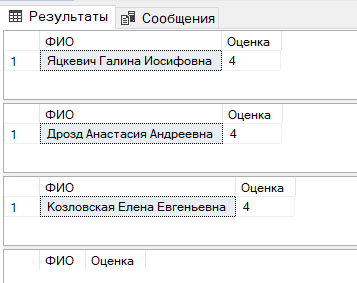
begin

delete PROGRESS where current of curTask\_6\_1;

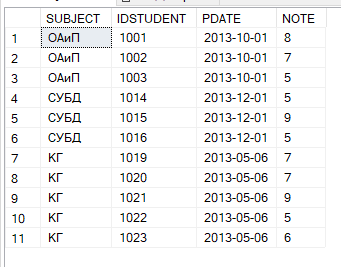
fetch curTask\_6\_1;

end;

close curTask\_6\_1;



Теперь таблица PROGRESS:



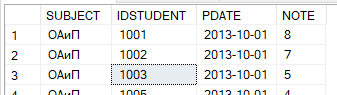
Теперь удалю таблицу PROGRESS, а PROGRESS\_copy переименую в PROGRESS:

-- удаление покалеченной таблицы и восстановление исходного варианта из копии

drop table PROGRESS;

exec sp\_rename 'PROGRESS\_copy', 'PROGRESS';

***6.2***



Увеличиваем на 1 оценку 1003 студента:

declare curTask\_6\_2 cursor local dynamic

for select IDSTUDENT

from PROGRESS

where IDSTUDENT = 1003

for update;

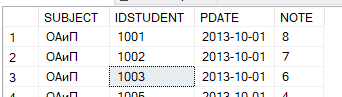
open curTask\_6\_2;

fetch curTask\_6\_2;

update PROGRESS set NOTE += 1

where current of curTask\_6\_2;

close curTask\_6\_2;



7. Разработать курсоры для базы данных **X\_MyBASE** и продемонстрировать их работу.

***→ Выполнение заданий***

***1***

-- 1 (вывести в строку через запятую список фамилий водителей со стажем более 5 лет)

declare curTask\_1 cursor local

for select Фамилия

from Водители

where Стаж > 5;

declare @surname nvarchar(50), @allSurnames nvarchar(200) = '';

open curTask\_1;

print 'Водители со стажем более 5 лет: ';

fetch curTask\_1 into @surname;

while @@fetch\_status = 0

begin

set @allSurnames = @surname + ',' + @allSurnames;

fetch curTask\_1 into @surname;

end;

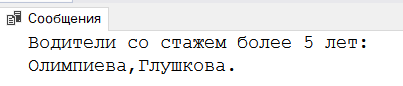
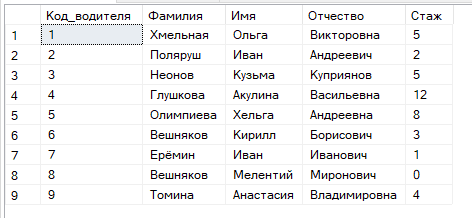
close curTask\_1;

-- убираем последнюю запятую в строке, затем добавляем точку

set @allSurnames = substring(@allSurnames, 1, len(@allSurnames)-1) + '.';

print @allSurnames;

Исходная таблица и результат запроса:

******

***2***

-- 2.1

-- локальный курсор

declare LocCurTask\_2 cursor local

for select Название

from Маршруты;

declare @name nvarchar(50);

open LocCurTask\_2;

fetch LocCurTask\_2 into @name;

print '1. ' + @name;

go

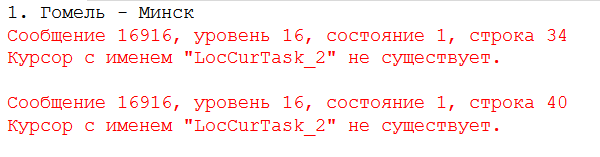
------------------------ конец 1 пакета

declare @name nvarchar(50);

fetch LocCurTask\_2 into @name;

print '2. ' + @name;

close LocCurTask\_2;

******

-- 2.2

-- глобальный курсор

declare GlobCurTask\_2 cursor global

for select Название

from Маршруты;

declare @nameG nvarchar(50);

open GlobCurTask\_2;

fetch GlobCurTask\_2 into @nameG;

print '1. ' + @nameG;

go

------------------------ конец 1 пакета

declare @nameG nvarchar(50);

fetch GlobCurTask\_2 into @nameG;

print '2. ' + @nameG;

close GlobCurTask\_2;

deallocate GlobCurTask\_2;

******

***3***

-- 3

-- статический курсор

declare statCur cursor static local scroll

for select Название

from Маршруты;

declare @nameForStat nvarchar(50);

open statCur;

insert into Маршруты(Название)

values('Заславль - Коссово');

print 'В таблицу добавлена строка ''Заславль - Коссово''!';

print 'Последняя строка таблицы ''Маршруты'' по версии статического курсора:';

fetch last from statCur into @nameForStat;

print @nameForStat;

close statCur;

-- динамический курсор

declare dynamicCur cursor dynamic local scroll

for select Название

from Маршруты;

declare @nameForDyn nvarchar(50);

open dynamicCur;

insert into Маршруты(Название)

values('Миоры - Туров');

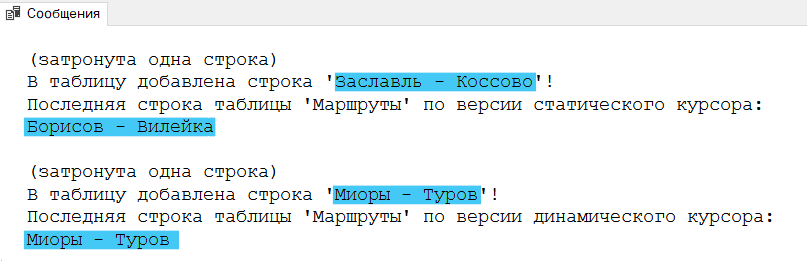
print 'В таблицу добавлена строка ''Миоры - Туров''!';

print 'Последняя строка таблицы ''Маршруты'' по версии динамического курсора:';

fetch last from dynamicCur into @nameForDyn;

print @nameForDyn;

close dynamicCur;

******

***4***

-- 4

-- scroll

declare curTask\_4 cursor scroll

for select row\_number() over (order by Название),

Название

from Маршруты;

declare @name\_4 nvarchar(50),

@allNames nvarchar(250) = '',

@num int;

-- Выведем все записи курсора через запятую

open curTask\_4;

print 'Все записи курсора:';

fetch curTask\_4 into @num, @name\_4;

while @@fetch\_status = 0

begin

set @allNames = @allNames + @name\_4 + ', ';

if (@num % 3 = 0)

set @allNames = @allNames + char(10);

fetch curTask\_4 into @num, @name\_4;

end;

close curTask\_4;

-- убираем последнюю запятую в строке, затем добавляем точку

set @allNames = substring(@allNames, 1, len(@allNames)-1) + '.';

print @allNames;

-- применение scroll

open curTask\_4;

fetch first from curTask\_4 into @num, @name\_4;

print char(10) + 'Первая запись : ' +

cast(@num as varchar(5)) + ' -> ' + @name\_4;

fetch next from curTask\_4 into @num, @name\_4;

print 'Следующая строка от текущей : ' +

cast(@num as varchar(5)) + ' -> ' + @name\_4;

fetch last from curTask\_4 into @num, @name\_4;

print 'Последняя запись : ' +

cast(@num as varchar(5)) + ' -> ' + @name\_4;

fetch prior from curTask\_4 into @num, @name\_4;

print 'Предыдущая строка от текущей : ' +

cast(@num as varchar(5)) + ' -> ' + @name\_4;

fetch absolute 3 from curTask\_4 into @num, @name\_4;

print 'Третья запись с начала : ' +

cast(@num as varchar(5)) + ' -> ' + @name\_4;

fetch absolute -3 from curTask\_4 into @num, @name\_4;

print 'Третья запись с конца : ' +

cast(@num as varchar(5)) + ' -> ' + @name\_4;

fetch relative 2 from curTask\_4 into @num, @name\_4;

print 'Вторая запись вперед от текущей : ' +

cast(@num as varchar(5)) + ' -> ' + @name\_4;

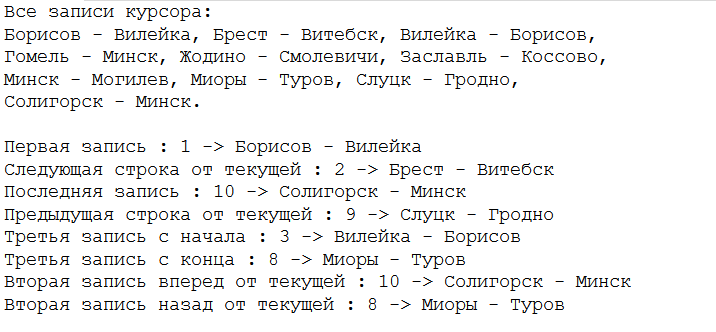
fetch relative -2 from curTask\_4 into @num, @name\_4;

print 'Вторая запись назад от текущей : ' +

cast(@num as varchar(5)) + ' -> ' + @name\_4;

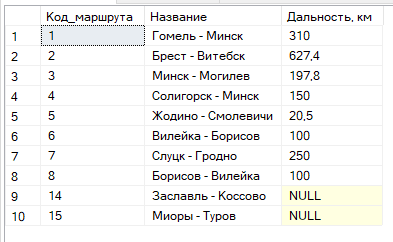
close curTask\_4;

deallocate curTask\_4;

******

***5***

Исходная таблица:

******

Запрос:

-- 5

declare @name\_5 nvarchar(50);

declare curTask\_5 cursor dynamic

for select Название

from Маршруты

for update;

open curTask\_5;

fetch last from curTask\_5 into @name\_5;

delete Маршруты where current of curTask\_5;

fetch last from curTask\_5 into @name\_5;

update Маршруты

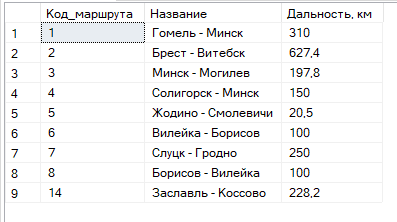
set [Дальность, км] = 228.2

where current of curTask\_5;

close curTask\_5;

deallocate curTask\_5;

Таблица после выполнения запроса:

******

***6.1***

Исходная таблица:



-- удалим из таблицы "Водители" водителей со стажем меньше года

select \* into cop

from Водители;

declare curTask\_6\_1 cursor local dynamic

for select Код\_водителя

from Водители

where Стаж < 1

for update;

open curTask\_6\_1;

fetch curTask\_6\_1;

while @@fetch\_status = 0

begin

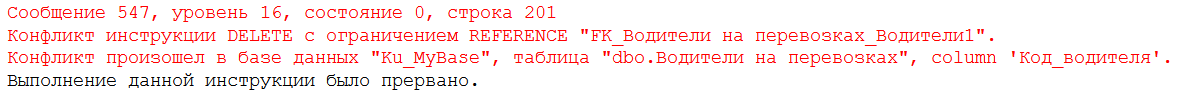
delete Водители where current of curTask\_6\_1;

fetch curTask\_6\_1;

end;

close curTask\_6\_1;

Не получилось из ограничения целостности:

******

***6.2***

Исходная таблица:



-- 6.1

declare curTask\_6\_2 cursor local dynamic

for select Код\_водителя

from Водители

where Код\_водителя = 7

for update;

open curTask\_6\_2;

fetch curTask\_6\_2;

update Водители set Стаж += 1

where current of curTask\_6\_2

close curTask\_6\_2;

После запроса:

