БАЗЫ ДАННЫХ

Лекция 10 Курсоры

Курсоры

- **Курсор –** механизм, позволяющий обрабатывать отдельные строки, полученные в результате select-запроса.

Курсоры

- **Курсор –** область памяти сервера, предназначенная для хранения и обработки результата select-запроса.

Курсоры

1. Курсор объявляется в операторе DECLARE.
2. Курсор открывается с помощью оператора OPEN.
3. С помощью оператора FETCH считывается одна или несколько строк результирующего набора, связанного с курсором SELECT-оператора, и обрабатывается нужным образом. Результат каждого считывания проверяется с помощью системной функции @@FETCH_STATUS.
4. Курсор закрывается оператором CLOSE.
5. Если курсор глобальный, то он должен быть освобожден с использованием оператора DEALLOCATE.

Курсоры

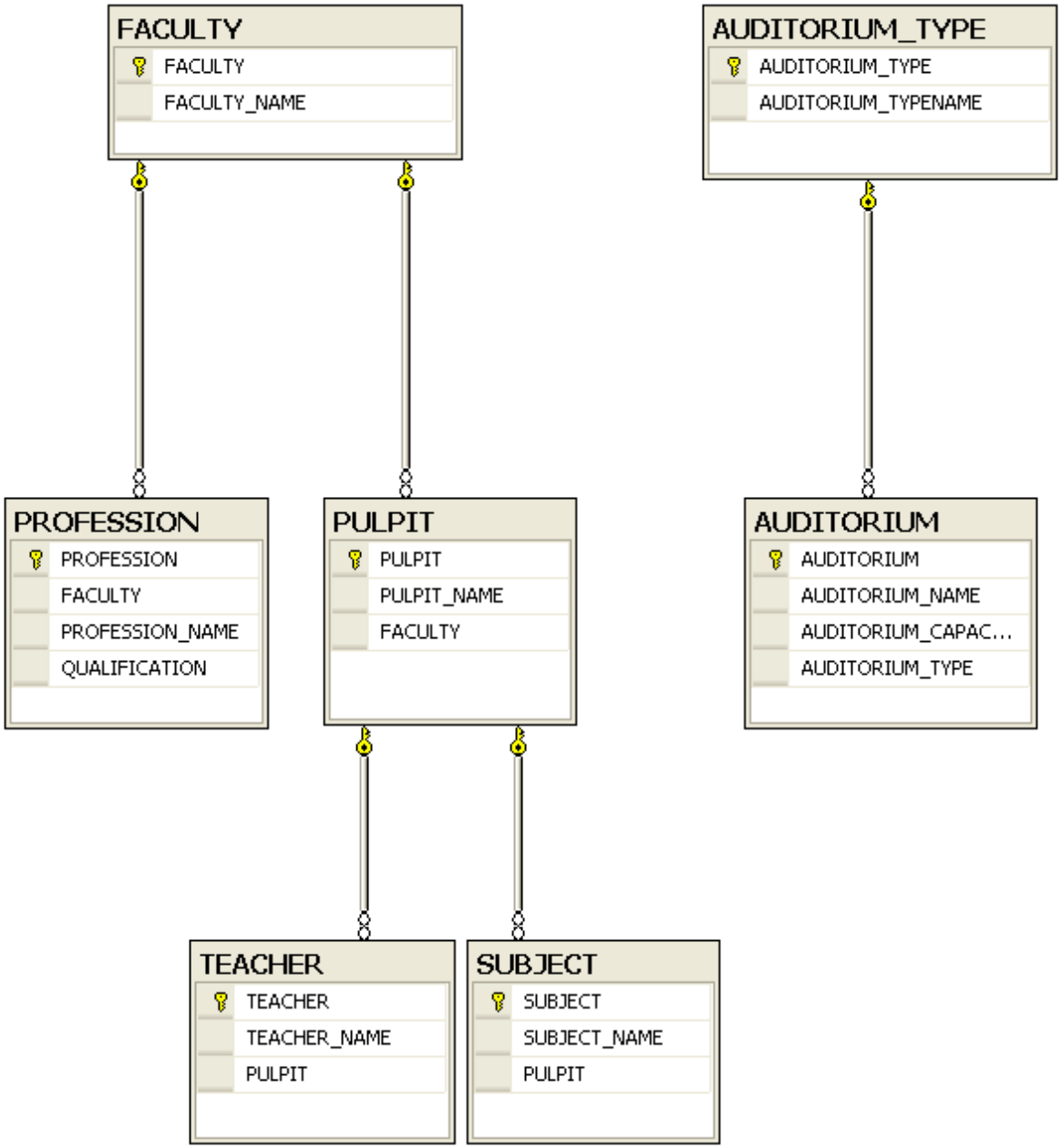
```
declare @tid char(10), @tnm char(40), @tgn char(1);
declare c_teacher cursor global                                    -- 1
                  for select TEACHER, TEACHER_NAME, GENDER         -- 1
                      from TEACHER where PULPIT = 'ПОиСОИ';        -- 1

 open c_teacher;                                                   -- 2
 fetch  c_teacher into @tid, @tnm, @tgn;                           -- 3
 while @@fetch_status = 0                                          -- 3
 begin
    print @tid + ' '+ @tnm + ' '+ @tgn;
    fetch  c_teacher into @tid, @tnm, @tgn;                        -- 3
 end;
 close  c_teacher;                                                 -- 4
 deallocate c_teacher;                                             -- 5
```

Курсоры

```
-- DROP TABLE SUBJECT
CREATE TABLE SUBJECT
    (
     SUBJECT      CHAR(10)     NOT NULL,
     SUBJECT_NAME VARCHAR(50)  NOT NULL,
     PULPIT       CHAR(10)     NOT NULL,
     CONSTRAINT PK_SUBJECT PRIMARY KEY(SUBJECT),
     CONSTRAINT FK_SUBJECT_PULPIT FOREIGN  KEY(PULPIT)  REFERENCES PULPIT(PULPIT)
    )
```

SUBJECT	SUBJECT_NAME	PULPIT
ИНФ	Информацтонные технологии	ИСиТ
КГ	Компьютерная геометрия	ИСиТ
КМС	Компьютерные мультимедийные системы	ИСиТ
БД	Базы данных	ИСиТ
МСОИ	Моделирование систем обработки информации	ИСиТ
ОАиП	Основы алгоритмизации и программирования	ИСиТ
ПЗ	Представление знаний в компьютерных системах	ИСиТ
ПИС	Проектирование информационных систем	ИСиТ
СУБД	Системы управления базами данных	ИСиТ
ПСП	Пограммирование сетевых приложений	ИСиТ
ЛВ	Лесоводство	ЛЗиДВ
ТиОЛ	Технология и оборудование лесозаготовок	ЛМиЛЗ
ОСПиЛ...	Основы садовопаркового и лесопаркового хозя...	ЛПиС...
ИГ	Инженерная геодезия	ЛУ
ЭП	Экономика природопользования	МиЭП
БЛЗиП...	Биология лесных зверей и птиц с осн. охотов.	ОВ
ОХ	Органическая химия	ОХ
ПМАПЛ	Полиграфические машины, автоматы и поточны...	ПОиС...
ОПП	Организация полиграфического производства	ПОиС...
ВТЛ	Водный транспорт леса	ТЛ
ТОПИ	Технология обогащения полезных ископаемых	ТНВи...
ТРИ	Технология резиновых изделий	ТНХС...
ПЭХ	Прикладная электрохимия	ХТЭП...
ЭТ	Экономическая теория	ЭТиМ


FACULTY
FACULTY
FACULTY_NAME
AUDITORIUM_TYPE
AUDITORIUM_TYPE
AUDITORIUM_TYPENAME
PROFESSION
PROFESSION
FACULTY
PROFESSION_NAME
QUALIFICATION
PULPIT
PULPIT
PULPIT_NAME
FACULTY
AUDITORIUM
AUDITORIUM
AUDITORIUM_NAME
AUDITORIUM_CAPAC...
AUDITORIUM_TYPE
TEACHER
TEACHER
TEACHER_NAME
PULPIT
SUBJECT
SUBJECT
SUBJECT_NAME
PULPIT

Курсоры – пример

```
declare ccc cursor
        for select subject, pulpit from subject
        read_only
declare @s char(10), @p char(10)

open ccc
fetch ccc into @s, @p
print @s+' '+@p
close ccc

go
```

```
БД          ИСиТ
```

Курсоры

- Глобальные курсоры
- Локальные курсоры

Курсоры

```
declare ccc cursor
            for select subject, pulpit from subject
            read_only
declare @s char(10), @p char(10)

open ccc
fetch ccc into @s, @p
print @s+' '+@p
close ccc

go
```

```
Сообщение 16915, уровень 16, состояние 1, строка 2
Курсор с именем "ccc" уже существует.
```

Курсоры

```
declare ccc cursor
              for select subject, pulpit from subject
              read_only
declare @s char(10), @p char(10)

open ccc
fetch ccc into @s, @p
print @s+' '+@p
close ccc
deallocate ccc
go
```

Курсоры

```
declare ccc cursor local
            for select subject, pulpit from subject
            read_only
declare @s char(10), @p char(10)

open ccc
fetch ccc into @s, @p
print @s+' '+@p
close ccc
-- deallocate ccc
go
```

Курсоры

```
declare ccc cursor global
            for select subject, pulpit from subject
            read_only
declare @s char(10), @p char(10)

open ccc
fetch ccc into @s, @p
print @s+' '+@p
close ccc
deallocate ccc
go
```

Типы курсоров

- Динамические
- Статические

Курсоры

- **Динамический курсор –** изменения данных отображаются в динамике

Курсоры

- **Статический курсор –** данные выбраны один раз и произошедшие изменения не видны

Курсоры

- Асинхронное заполнение статических курсоров оптимизирует производительность
- Статические курсоры используют рабочие таблицы базы данных **tempdb** для хранения строк, составляющих курсор
- Если в соответствии с прогнозом оптимизатора запросов SQL Server ожидаемое число строк, возвращаемых в курсоре, превысит значение параметра **sp_configure cursor threshold**, сервер запускает отдельный поток для заполнения рабочей таблицы

Курсоры

- Функция @@CURSOR_ROWS сообщает число строк в курсоре.
- @@CURSOR_ROWS для курсора, рабочая таблица которого продолжает заполняться, возвращается отрицательное число. Абсолютное значение возвращенного числа дает число строк в рабочей таблице, заполненных на данный момент времени.
- Например, если функция @@CURSOR_ROWS выбрана, пока идет заполнение набора ключей или курсора, управляемого набором ключей, но в наборе ключей уже имеется 143 ключа, функция возвращает значение -143.

Курсоры

- @@CURSOR_ROWS
  - -n – количество записей при асинхронной выборке,
  - n – количество записей при синхронной выборке,
  - 0 – курсор не открыт

Курсоры

```
declare ccc cursor local
            for select subject, pulpit from subject
            read_only
 declare @s char(10), @p char(10)
 print ' @@CURSOR_ROWS = '+ CAST(@@CURSOR_ROWS as varchar(10))
 open ccc
 print ' @@CURSOR_ROWS = '+ CAST(@@CURSOR_ROWS as varchar(10))
 fetch ccc into @s, @p
 print @s+' '+@p+' @@CURSOR_ROWS = '+ CAST(@@CURSOR_ROWS as varchar(10))
 fetch ccc into @s, @p
 print @s+' '+@p+' @@CURSOR_ROWS = '+ CAST(@@CURSOR_ROWS as varchar(10))
 close ccc
```

```
 @@CURSOR_ROWS = 0
 @@CURSOR_ROWS = -1
БД          ИСиТ        @@CURSOR_ROWS = -1
БЛЗиПсОО    ОВ          @@CURSOR_ROWS = -1
```

Курсоры

```
declare ccc cursor local dynamic
            for select subject, pulpit from subject
            read_only
declare @s char(10), @p char(10)
print ' @@CURSOR_ROWS = '+ CAST(@@CURSOR_ROWS as varchar(10))
open ccc
print ' @@CURSOR_ROWS = '+ CAST(@@CURSOR_ROWS as varchar(10))
fetch ccc into @s, @p
print @s+' '+@p+' @@CURSOR_ROWS = '+ CAST(@@CURSOR_ROWS as varchar(10))
fetch ccc into @s, @p
print @s+' '+@p+' @@CURSOR_ROWS = '+ CAST(@@CURSOR_ROWS as varchar(10))
close ccc
```

```
 @@CURSOR_ROWS = 0
 @@CURSOR_ROWS = -1
ЗД          ИСиТ        @@CURSOR_ROWS = -1
ЗЛЗиПс00    ОВ          @@CURSOR_ROWS = -1
```

Курсоры

```
declare ccc cursor local static
            for select subject, pulpit from subject
            read_only
declare @s char(10), @p char(10)
print ' @@CURSOR_ROWS = '+ CAST(@@CURSOR_ROWS as varchar(10))
open ccc
print ' @@CURSOR_ROWS = '+ CAST(@@CURSOR_ROWS as varchar(10))
fetch ccc into @s, @p
print @s+' '+@p+' @@CURSOR_ROWS = '+ CAST(@@CURSOR_ROWS as varchar(10))
fetch ccc into @s, @p
print @s+' '+@p+' @@CURSOR_ROWS = '+ CAST(@@CURSOR_ROWS as varchar(10))
close ccc
```

```
 @@CURSOR_ROWS = 0
 @@CURSOR_ROWS = 24
БД          ИСиТ        @@CURSOR_ROWS = 24
БЛЗиПс00    ОВ          @@CURSOR_ROWS = 24
```

Курсоры

- @@FETCH_STATUS – возвращает состояние последней инструкции FETCH, вызванной в любом курсоре, открытом в данном соединении

Курсоры

- @@FETCH_STATUS
  - 0 – успешная выборка,
  - -1 – вышли за диапазон таблицы,
  - -2 – запись удалена после открытия курсора

Курсоры

```
declare ccc cursor local
            for select subject, pulpit from subject
            read_only
declare @s char(10), @p char(10)
print ' @@FETCH_STATUS = '+ CAST(@@FETCH_STATUS as varchar(10))
open ccc
print ' @@FETCH_STATUS = '+ CAST(@@FETCH_STATUS as varchar(10))
fetch ccc into @s, @p
while @@FETCH_STATUS = 0
  begin
    print @s+' '+@p+' @@FETCH_STATUS = '+ CAST(@@FETCH_STATUS as varchar(10))
    fetch ccc into @s, @p
  end
print ' @@FETCH_STATUS = '+ CAST(@@FETCH_STATUS as varchar(10))
close ccc
print ' @@FETCH_STATUS = '+ CAST(@@FETCH_STATUS as varchar(10))
go
```

Курсоры

```
 @@FETCH_STATUS = -1
 @@FETCH_STATUS = -1
БД          ИСиТ        @@FETCH_STATUS = 0
БЛЗиПсОО    ОВ          @@FETCH_STATUS = 0
ВТЛ         ТЛ          @@FETCH_STATUS = 0
ИГ          ЛУ          @@FETCH_STATUS = 0
ИНФ         ИСиТ        @@FETCH_STATUS = 0
КГ          ИСиТ        @@FETCH_STATUS = 0
КМС         ИСиТ        @@FETCH_STATUS = 0
ЛВ          ЛЗиДВ       @@FETCH_STATUS = 0
МСОИ        ИСиТ        @@FETCH_STATUS = 0
ОАиП        ИСиТ        @@FETCH_STATUS = 0
ОПП         ПОиСОИ      @@FETCH_STATUS = 0
ОСПиЛПХ     ЛПиСПС      @@FETCH_STATUS = 0
ОХ          ОХ          @@FETCH_STATUS = 0
ПЗ          ИСиТ        @@FETCH_STATUS = 0
ПИС         ИСиТ        @@FETCH_STATUS = 0
ПМАПЛ       ПОиСОИ      @@FETCH_STATUS = 0
ПСП         ИСиТ        @@FETCH_STATUS = 0
ПЭХ         ХТЭПиМЭЕ    @@FETCH_STATUS = 0
СУБД        ИСиТ        @@FETCH_STATUS = 0
ТиОЛ        ЛМиЛЗ       @@FETCH_STATUS = 0
ТОПИ        ТНВиОХТ     @@FETCH_STATUS = 0
ТРИ         ТНХСиППМ    @@FETCH_STATUS = 0
ЭП          МиЭП        @@FETCH_STATUS = 0
ЭТ          ЭТиМ        @@FETCH_STATUS = 0
 @@FETCH_STATUS = -1
 @@FETCH_STATUS = -1
```

Курсоры – SCROLL

```
declare ccc cursor local scroll
            for select subject, pulpit from subject order by pulpit
            --read_only  -- не совместен со scroll
declare @s char(10), @p char(10)
open ccc
fetch ccc into @s, @p
while @@FETCH_STATUS = 0
  begin
    print @s+' '+@p
    fetch ccc into @s, @p
  end
close ccc
go
```

Курсоры

ИНФ	ИСиТ
КГ	ИСиТ
КМС	ИСиТ
ЗД	ИСиТ
МСОИ	ИСиТ
ОАиП	ИСиТ
ЛЭ	ИСиТ
ЛИС	ИСиТ
СУБД	ИСиТ
ЛСП	ИСиТ
ПВ	ЛЗиДВ
ГиОЛ	ЛМиЛЭ
ОСПиЛПХ	ЛПиСПС
ИГ	ЛУ
ЭП	МиЭП
ЗЛЗиПсОО	ОВ
ОХ	ОХ
ЛМАПЛ	ПОиСОИ
ОПП	ПОиСОИ
ЭТЛ	ТЛ
ГОПИ	ТНВиОХТ
ГРИ	ТНХСиППМ
ЛЭХ	ХТЭПиМЭЕ
ЭТ	ЭТиМ

```
declare ccc cursor local scroll
            for select subject, pulpit from subject order by pulpit
declare @s char(10), @p char(10)
open ccc

fetch last from ccc into @s, @p
print @s+' '+@p

fetch first from ccc into @s, @p
print @s+' '+@p

fetch absolute 10 from ccc into @s, @p
print @s+' '+@p

fetch relative 5 from ccc into @s, @p
print @s+' '+@p

fetch relative -5 from ccc into @s, @p
print @s+' '+@p

fetch absolute -10 from ccc into @s, @p
print @s+' '+@p

fetch next from ccc into @s, @p
print @s+' '+@p

fetch prior from ccc into @s, @p
print @s+' '+@p

close ccc
go
```

```
ЭТ          ЭТиМ
ИНФ         ИСиТ
ПСП         ИСиТ
ЭП          МиЭП
ПСП         ИСиТ
ЭП          МиЭП
БЛЗиПс00    ОВ
ЭП          МиЭП
```

Курсоры

```
declare @s char(10), @ps char(10),  @p char(10)
declare ccc cursor local dynamic scroll
            for select subject, pulpit from subject  where pulpit = @ps
set @ps = 'ИСиТ'
open ccc
fetch ccc into @s, @p
while @@FETCH_STATUS = 0
  begin
    print @s+' '+@p
    fetch ccc into @s, @p
  end
close ccc
go
```

```
Выполнение команд успешно завершено.
```

Курсоры

```
declare @s char(10), @ps char(10) = 'ИСиТ',  @p char(10)
declare ccc cursor local dynamic scroll
             for select subject, pulpit from subject  where pulpit = @ps
-- set @ps = 'ИСиТ'
open ccc
fetch ccc into @s, @p
while @@FETCH_STATUS = 0
  begin
    print @s+' '+@p
    fetch ccc into @s, @p
  end
close ccc
go
```

БД	ИСиТ
ИНФ	ИСиТ
КГ	ИСиТ
КМС	ИСиТ
МСОИ	ИСиТ
ОАиП	ИСиТ
ПЗ	ИСиТ
ПИС	ИСиТ
ПСП	ИСиТ
СУБД	ИСиТ

Курсоры

```
declare @s char(10), @ps char(10),  @p char(10)
set @ps  = 'ИСиТ'

declare ccc cursor local dynamic scroll
            for select subject, pulpit from subject  where pulpit = @ps
-- set @ps = 'ИСиТ'
open ccc
fetch ccc into @s, @p
while @@FETCH_STATUS = 0
  begin
    print @s+' '+@p
    fetch ccc into @s, @p
  end
close ccc
go
```

БД	ИСиТ
ИНФ	ИСиТ
КГ	ИСиТ
КМС	ИСиТ
МСОИ	ИСиТ
ОАиП	ИСиТ
ПЗ	ИСиТ
ПИС	ИСиТ
ПСП	ИСиТ
СУБД	ИСиТ

Курсоры – UPDATE CURRENT OF

```
declare @s char(10), @ps char(10),  @p char(10), @n varchar(200)
declare ccc cursor local dynamic scroll
            for select subject, pulpit, subject_name from subject
open ccc
fetch ccc into @s, @p, @n
while @@FETCH_STATUS = 0
 begin
  if @s = 'БД' update subject set subject_name = 'Самая важная дисциплина!!!'
                                 where  current of ccc
  fetch ccc into @s, @p, @n
 end
close ccc
go
```

Курсоры – DELETE CURRENT OF

```
declare @s char(10), @ps char(10),  @p char(10), @n varchar(200)
declare ccc cursor local dynamic scroll
            for select subject, pulpit, subject_name from subject
open ccc
fetch ccc into @s, @p, @n
while @@FETCH_STATUS = 0
 begin
   if @s = 'БД' delete subject where  current of ccc
   fetch ccc into @s, @p, @n
 end
close ccc
go
```

Вложенные курсоры

```
SET NOCOUNT ON;

DECLARE @vendor_id int, @vendor_name nvarchar(50),
    @message varchar(80), @product nvarchar(50);

PRINT '-------- Vendor Products Report --------';

DECLARE vendor_cursor CURSOR FOR
SELECT BusinessEntityID, Name
FROM Purchasing.Vendor
WHERE PreferredVendorStatus = 1
ORDER BY BusinessEntityID;

OPEN vendor_cursor

FETCH NEXT FROM vendor_cursor
INTO @vendor_id, @vendor_name

WHILE @@FETCH_STATUS = 0
BEGIN
    PRINT ' '
    SELECT @message = '----- Products From Vendor: ' +
        @vendor_name

    PRINT @message
```

```
-- Declare an inner cursor based
-- on vendor_id from the outer cursor.
  DECLARE product_cursor CURSOR FOR
SELECT v.Name
FROM Purchasing.ProductVendor pv, Production.Product v
WHERE pv.ProductID = v.ProductID AND
pv.BusinessEntityID = @vendor_id  -- Variable value from the outer cursor

OPEN product_cursor
FETCH NEXT FROM product_cursor INTO @product
IF @@FETCH_STATUS <> 0
    PRINT '          <<None>>'

WHILE @@FETCH_STATUS = 0
 BEGIN
    SELECT @message = '          ' + @product
    PRINT @message
    FETCH NEXT FROM product_cursor INTO @product
 END
CLOSE product_cursor
DEALLOCATE product_cursor
```

```
        -- Get the next vendor.
    FETCH NEXT FROM vendor_cursor
    INTO @vendor_id, @vendor_name
END
CLOSE vendor_cursor;
DEALLOCATE vendor_cursor;
```

Вложенные курсоры

```
-------- Vendor Products Report --------

----- Products From Vendor: Australia Bike Retailer
        Thin-Jam Lock Nut 9
        Thin-Jam Lock Nut 10
        Thin-Jam Lock Nut 1
        Thin-Jam Lock Nut 2
        Thin-Jam Lock Nut 15
        Thin-Jam Lock Nut 16
        Thin-Jam Lock Nut 5
        Thin-Jam Lock Nut 6
        Thin-Jam Lock Nut 3
        Thin-Jam Lock Nut 4
        Thin-Jam Lock Nut 13
        Thin-Jam Lock Nut 14
        Thin-Jam Lock Nut 7
        Thin-Jam Lock Nut 8
        Thin-Jam Lock Nut 12
        Thin-Jam Lock Nut 11

----- Products From Vendor: Allenson Cycles
        Seat Post
```

Вложенные курсоры

```
----- Products From Vendor: Morgan Bike Accessories
        HL Grip Tape

----- Products From Vendor: Cycling Master
        <<None>>

----- Products From Vendor: Chicago Rent-All
        Reflector

----- Products From Vendor: Greenwood Athletic Company
        LL Mountain Pedal
        ML Mountain Pedal

----- Products From Vendor: Compete Enterprises, Inc
        Guide Pulley
        Tension Pulley
        HL Road Pedal
```

Курсоры

- CURSOR_STATUS – скалярная функция, позволяющая при вызове хранимой процедуры определить, вернула ли она курсор и результирующий набор для данного параметра

Курсоры

```
DECLARE @a1 INT, @a2 INT, @a3 INT, @a4 INT, @a5 INT;

DECLARE CURS_SUBJECTS CURSOR FOR
SELECT SUBJECT FROM SUBJECT WHERE PULPIT='ИСиТ' ;
    SET @a1 = CURSOR_STATUS('GLOBAL', 'CURS_SUBJECTS');
DECLARE  @S1 NCHAR(5), @S NCHAR(300) ='';

OPEN CURS_SUBJECTS;
    SET @a2 = CURSOR_STATUS('GLOBAL', 'CURS_SUBJECTS');
FETCH CURS_SUBJECTS INTO @S1;
    SET @a3 = CURSOR_STATUS('GLOBAL', 'CURS_SUBJECTS');
PRINT 'Список кратких наименований предметов на кафедре ИСиТ: ';
WHILE (@@FETCH_STATUS=0)
    BEGIN
    SET @S=RTRIM(@S1)+', '+@S;
    FETCH CURS_SUBJECTS INTO @S1;
    END;
PRINT LEFT(@S, LEN(@S)-1);
    SET @a4 = CURSOR_STATUS('GLOBAL', 'CURS_SUBJECTS');
CLOSE CURS_SUBJECTS;
    SET @a5 = CURSOR_STATUS('GLOBAL', 'CURS_SUBJECTS');

    SELECT  @a1 AS 'AFTER DECLARE',
            @a2 AS 'AFTER OPEN',
            @a3 AS 'AFTER 1 FETCH',
            @a4 AS 'AFTER ALL FETCH',
            @a5 AS 'AFTER CLOSE';
```

Results | Messages

	AFTER declare	AFTER OPEN	AFTER 1 FETCH	AFTER ALL FETCH	AFTER CLOSE
1	-1	1	1	1	-1

Процедуры, поддерживающие курсоры

- sp_cursor_list
- sp_describe_cursor
- sp_describe_cursor_columns
- sp_describe_cursor_tables

sp_cursor_list

```
DECLARE @REPORT CURSOR;
EXEC SP_CURSOR_LIST @CURSOR_RETURN = @REPORT OUTPUT, @CURSOR_SCOPE =3;
OPEN @REPORT;
FETCH NEXT FROM @REPORT;
WHILE (@@FETCH_STATUS=0)
BEGIN
FETCH NEXT FROM @REPORT;
END;
CLOSE @REPORT;
```

Results | Messages

	reference_name	cursor_name	cursor_scope	status	model	concurrency	scrollable
1	CURS_SUBJECTS	CURS_SUBJECTS	2	-1	3	3	0

	reference_name	cursor_name	cursor_scope	status	model	concurrency	scrollable	open_

open_status	cursor_rows	fetch_status	column_count	row_count	last_operation	cursor_handle
0	0	-1	1	0	6	180150009

tatus	cursor_rows	fetch_status	column_count	row_count	last_operation	cursor_handle

sp_describe_cursor

```
-----------------------------------
DECLARE @REPORT CURSOR;
EXEC SP_describe_CURSOR @CURSOR_return = @REPORT OUTPUT,
                        @CURSOR_Source ='global',
                        @CURSOR_Identity ='CURS_SUBJECTS';

OPEN @REPORT;
FETCH NEXT FROM @REPORT;
WHILE (@@FETCH_STATUS=0)
BEGIN
FETCH NEXT FROM @REPORT;
END;
CLOSE @REPORT;
DEALLOCATE @REPORT;
```

Results | Messages

	reference_name	cursor_name	cursor_scope	status	model	concurrency	scrollable
1	CURS_SUBJECTS	CURS_SUBJECTS	2	-1	3	3	0

	reference_name	cursor_name	cursor_scope	status	model	concurrency	scrollable	open_

open_status	cursor_rows	fetch_status	column_count	row_count	last_operation	cursor_handle
0	0	-1	1	0	6	180150009

tatus	cursor_rows	fetch_status	column_count	row_count	last_operation	cursor_handle

sp_describe_cursor_columns

```
----------------------------------
-- SP_DESCRIBE_CURSOR_COLUMNS
----------------------------------
DECLARE @REPORT CURSOR;
EXEC SP_DESCRIBE_CURSOR_COLUMNS @CURSOR_return = @REPORT OUTPUT,
                          @CURSOR_Source ='global',
                          @CURSOR_Identity ='CURS_SUBJECTS';
OPEN @REPORT;
FETCH NEXT FROM @REPORT;
WHILE (@@FETCH_STATUS=0)
BEGIN
FETCH NEXT FROM @REPORT;
END;
CLOSE @REPORT;
DEALLOCATE @REPORT;

go
```

Results | Messages

column_name	ordinal_position	column_characteristics_flags	column_size	data_type_sql	column_precision
SUBJECT	0	18	10	175	0

column_name	ordinal_position	column_characteristics_flags	column_size	data_type_sql	column_precision

column_scale	order_position	order_direction	hidden_column	columnid	objectid	dbid	dbname
0	0	NULL	0	1	565577053	11	B_BSTU

column_scale	order_position	order_direction	hidden_column	columnid	objectid	dbid	dbname

sp_describe_cursor_tables

```
-----------------------------------
-- SP_DESCRIBE_CURSOR_TABLES
-----------------------------------
DECLARE @REPORT CURSOR;
EXEC SP_DESCRIBE_CURSOR_TABLES @CURSOR_return = @REPORT OUTPUT,
                        @CURSOR_Source ='global',
                        @CURSOR_Identity ='CURS_SUBJECTS';

OPEN @REPORT;
FETCH NEXT FROM @REPORT;
WHILE (@@FETCH_STATUS=0)
BEGIN
FETCH NEXT FROM @REPORT;
END;
CLOSE @REPORT;
DEALLOCATE @REPORT;
```

Results | Messages

	table_owner	table_name	optimizer_hint	lock_type	server_name	objectid	dbid	dbname
1	dbo	SUBJECT	0	0	1-VAIO	565577053	11	B_BSTU

table_owner	table_name	optimizer_hint	lock_type	server_name	objectid	dbid	dbname

Вопросы?