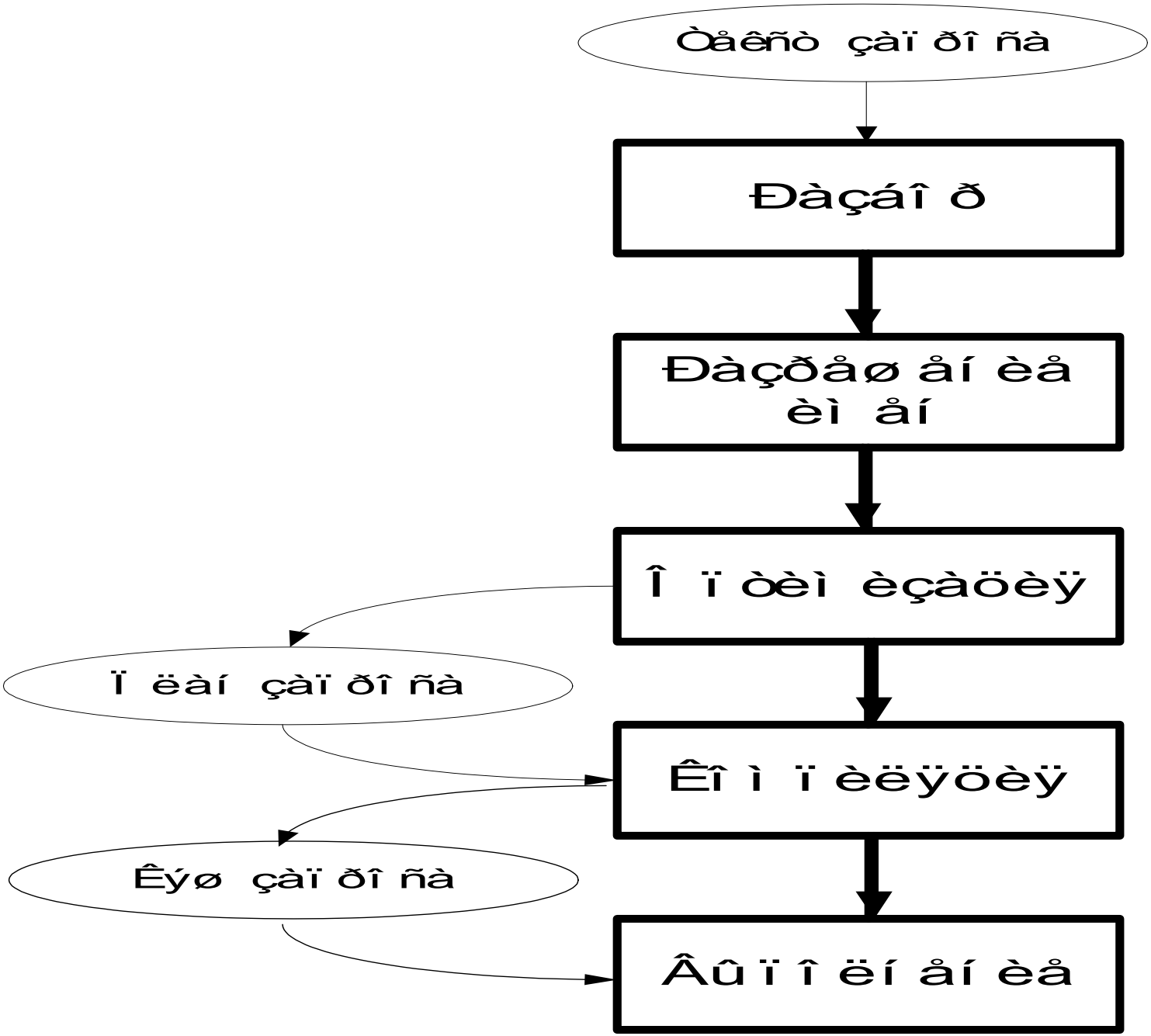
БАЗЫ ДАННЫХ

Лекция 12 Индексы


Òåêñò çàïðîñà
Ðàçáîð
Ðàçðåøåíèå èìåí
Îïòèìèçàöèÿ
Ïëàí çàïðîñà
Êîìïèëÿöèÿ
Êýø çàïðîñà
Âûïîëíåíèå

Обработка SQL-запроса

- Разбор
- Разрешение имен
- Оптимизация
- Компиляция
- Выполнение

Синтаксический разбор

- Синтаксический разбор текста запроса для проверки его на соответствие правилам языка.

Разрешение имен

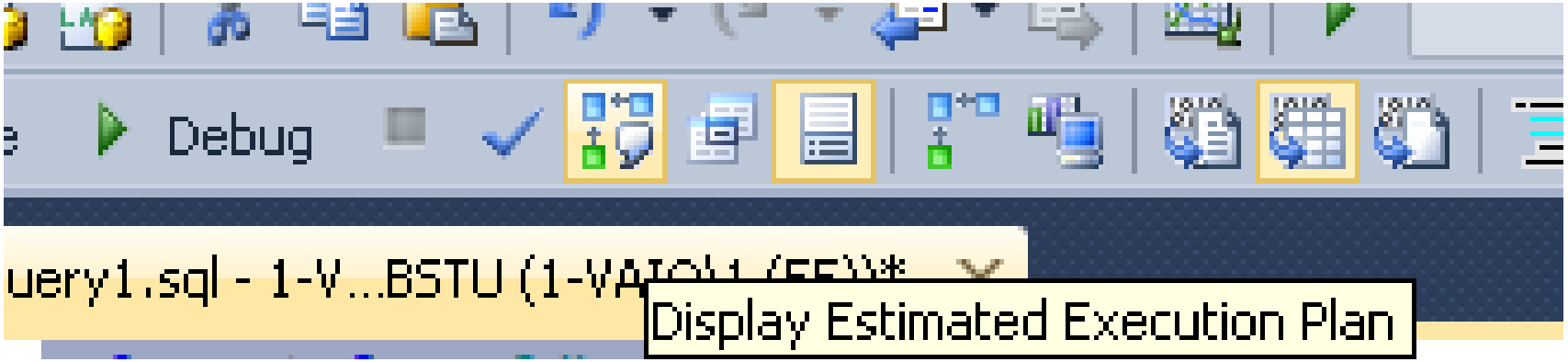
- Проверка наличия используемых в запросе объектов БД:
  - Таблиц
  - Представлений
  - Столбцов
  - Пользовательских и встроенных функций и пр.

Оптимизация

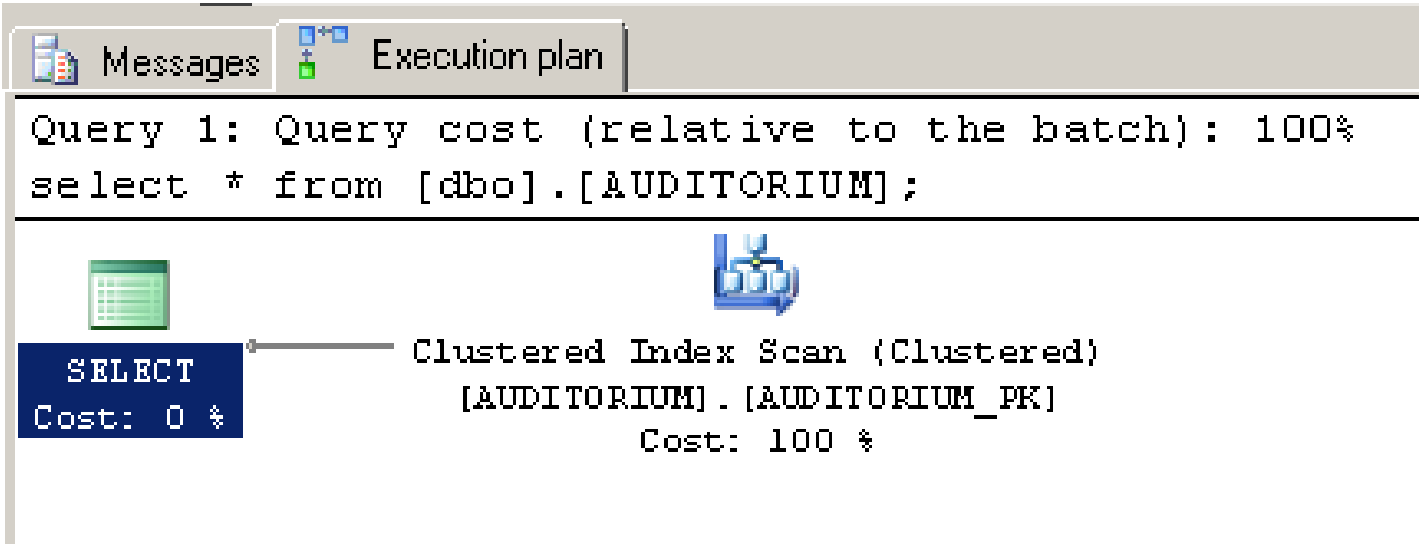
- Специальная компоненте сервера – оптимизатор
- Основная задача оптимизатора – построение плана запроса
- План запроса представляет собой алгоритм выполнения SQL-запроса

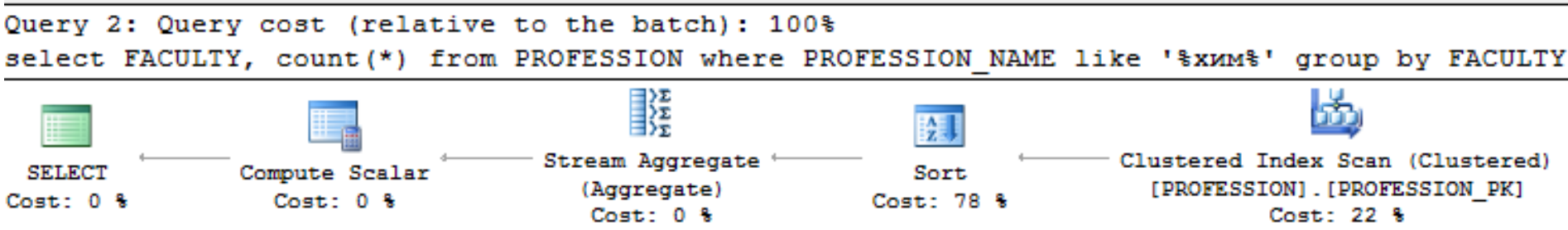
План запроса

- **Display Estimated Execution Plan** - Показать предполагаемый план выполнения



План запроса





План запроса

- Для каждого шага вычисляется стоимость – величина, пропорциональная продолжительности выполнения шага
- Суммарная стоимость шагов плана составляет стоимость всего запроса
- Задача – минимизация общей стоимости запроса

Компиляция

- Откомпилированный план запроса помещается в специальную область памяти, называемую библиотечным кэшем
- Кэш используется для ускорения выполнения будущих аналогичных запросов

Выполнение

- Откомпилированный план выполняется сервером СУБД

Индексы

- Индекс представляет собой отдельную физическую структуру данных, которая позволяет получать быстрый доступ к одной или нескольким строкам данных

Индексы

- Индексы сохраняются в страницах индексов
- Для каждой индексируемой строки имеется элемент индекса, который сохраняется на странице индексов
- Каждый элемент индекса состоит из ключа индекса и указателя

Индексы

- Индексы создаются по сбалансированному дереву B+
- B+-дерево имеет древовидную структуру, в которой все листья находятся на расстоянии одинакового количества уровней от вершины дерева
- Это свойство поддерживается при добавлении или удалении данных в индексированном столбце

Индексы

- CREATE
- ALTER
- DROP

Индексы

- Индекс всегда связан с таблицей или с подмножеством столбцов таблицы

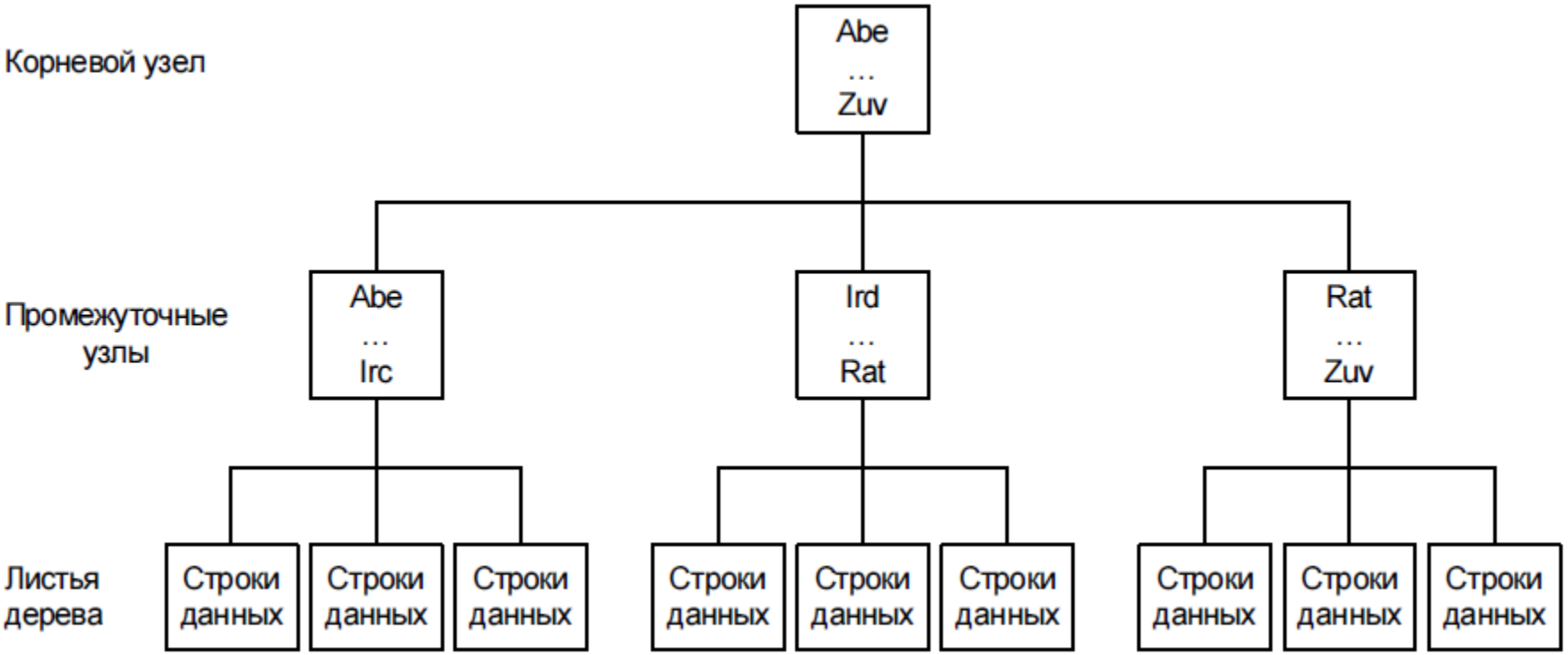
Индексы

- Кластеризованные индексы
- Некластеризованные индексы

Кластеризованный индекс

- Определяет физический порядок данных в таблице
- Может только один для одной таблицы
- Таблица перестраивается в порядке индекса
- Листья дерева индекса содержат страницы данных

Кластеризованный индекс



Кластеризованный индекс

- Создается по умолчанию для каждой таблицы, для которой определен первичный ключ
- Уникальный – в столбце, для которого определен кластеризованный индекс, каждое значение данных может встречаться только один раз
- Если кластеризованный индекс создается для столбца, содержащего повторяющиеся значения, СУБД принудительно добавляет четырехбайтовый идентификатор к строкам, содержащим дубликаты значений

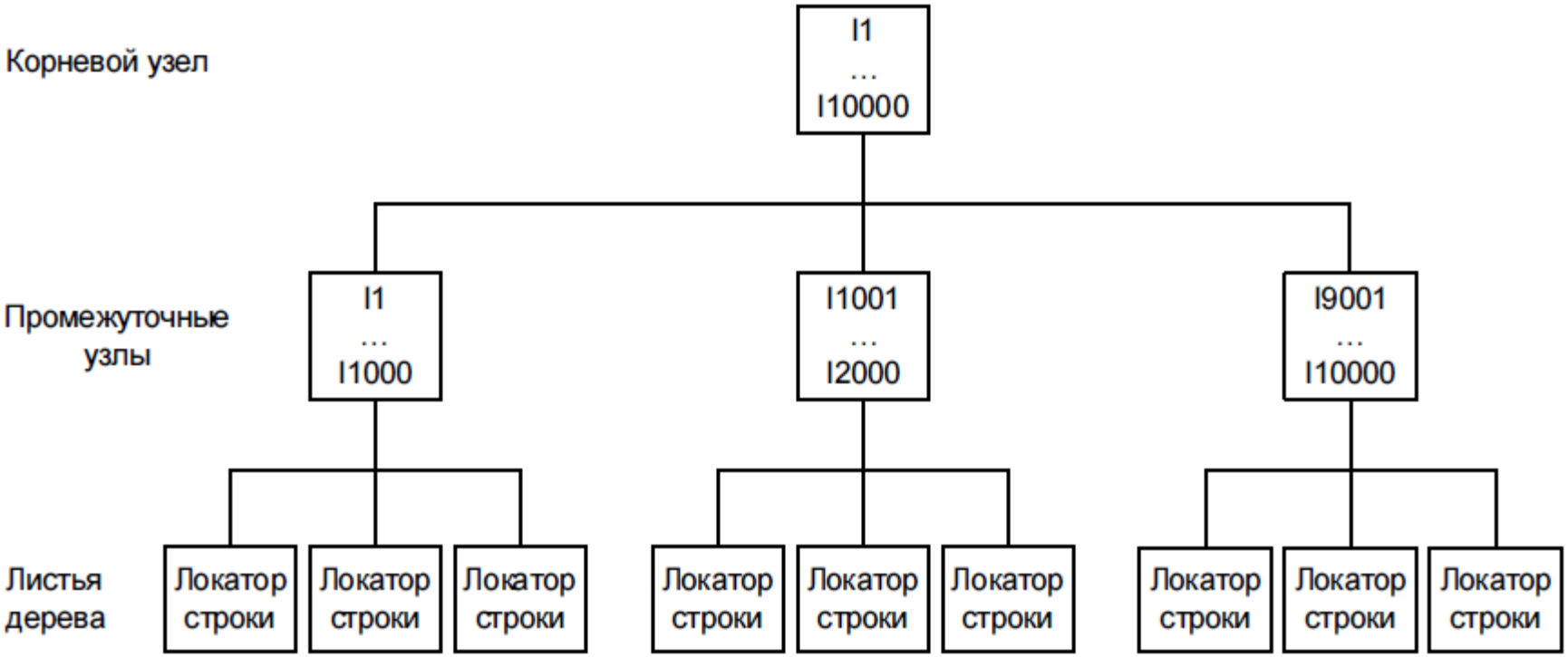
Индекс

- Кластеризованная таблица – таблица с кластеризованным индексом
- Куча (heap) – таблица без кластеризованного индекса

Некластеризованный индекс

- физически находится отдельно от таблицы
- страницы листьев состоят из ключей индекса и закладок
- может быть несколько для одной таблицы
- не изменяет физическое упорядочивание строк таблицы

Некластеризованный индекс



Некластеризованный индекс

- Если есть кластеризованный индекс, то закладка некластеризованного индекса показывает B+-дерево кластеризованного индекса таблицы
- Если нет кластеризованного индекса, закладка идентична RID — Row Identifier, состоящего из:
  - Адреса файла, в котором хранится таблица,
  - Адреса физического блока (страницы), в котором хранится строка,
  - Смещения строки в странице.

Некластеризованный индекс

- Поиск данных с использованием некластеризованного индекса в зависимости от типа таблицы:
- Куча — прохождение при поиске по структуре некластеризованного индекса, после чего строка извлекается, используя идентификатор строки.
- Кластеризованная таблица — прохождение при поиске по структуре некластеризованного индекса, после чего следует прохождение по соответствующему кластеризованному индексу.

Индексы

- **SP_HELPINDEX** - получить перечень индексов, связанных с заданной таблицей

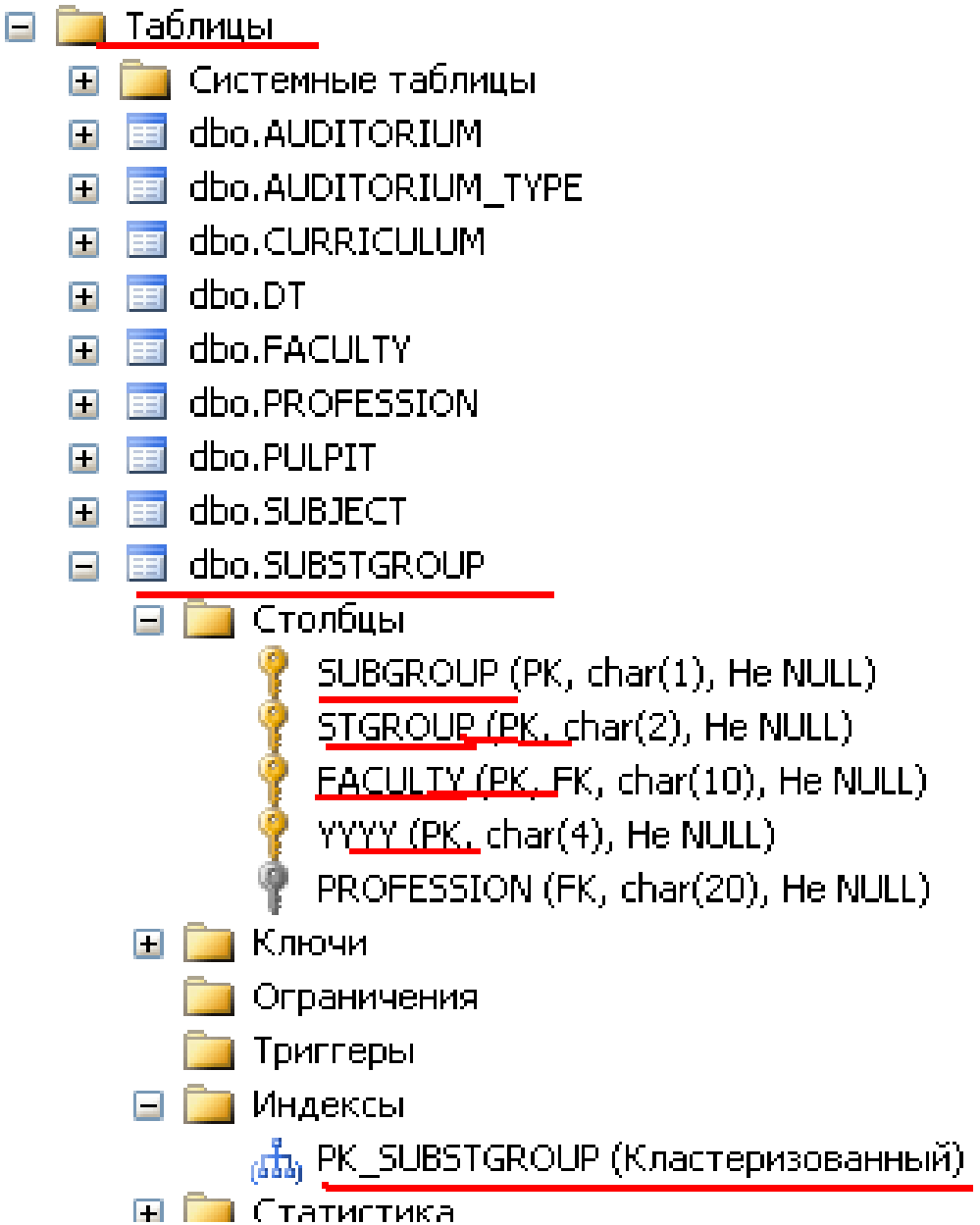
```
exec SP_HELPINDEX 'TEACHER'
```

index_name	index_description	index_keys
TEACHER_PK	clustered, unique, primary key located on FG1	TEACHER

Свойства индексов

- Индекс может иметь максимум 900 байтов и не более 16 столбцов
- Разрешено максимум 249 некластеризованных индексов для таблицы
- В UNIQUE составном индексе - однозначная комбинация значений всех столбцов каждой строки
- Если UNIQUE не указывается, то повторяющиеся значения разрешаются
- Параметр NONCLUSTED по умолчанию

Индексы


Таблицы
Системные таблицы
dbo.AUDITORIUM
dbo.AUDITORIUM_TYPE
dbo.CURRICULUM
dbo.DT
dbo.FACULTY
dbo.PROFESSION
dbo.PULPIT
dbo.SUBJECT
dbo.SUBSTGROUP
Столбцы
SUBGROUP (PK, char(1), He NULL)
STGROUP (PK, char(2), He NULL)
FACULTY (PK, FK, char(10), He NULL)
YYYY (PK, char(4), He NULL)
PROFESSION (FK, char(20), He NULL)
Ключи
Ограничения
Триггеры
Индексы
PK_SUBSTGROUP (Кластеризованный)
Статистика

Индексы - пример

```
create table #EXPLORE
(
 TKEY   int,
 CC     int identity(1,1),
 TFIELD varchar(100)
);
go
-- добавление в таблицу 10000 строк
set nocount on;      --  не выводить сообщения о вводе строк
declare @i int = 0;
while   @i < 10000
begin
  insert #EXPLORE(TKEY, TFIELD) values(floor(30000*RAND()), replicate('строка ', 10));
  if (@i%100 = 0) print @i;  -- вывести сообщение
  set @i = @i+1;
end;
go
select count(*)[количество строк] from #EXPLORE;
```

количество строк
10000

Индексы - пример

```
checkpoint;              -- фиксация БД
dbcc dropcleanbuffers; -- очистить буферный кэш
go
select * from  #EXPLORE where tkey between 15000 and 25000 order by tkey;
```

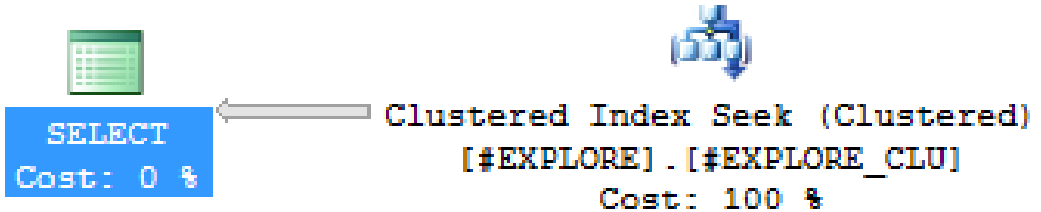
Cached plan size	16 B
Degree of Parallelism	0
Memory Grant	2672
Estimated Operator Cost	0 (0%)
Estimated Subtree Cost	0,17378
Estimated Number of Rows	3311,68

Statement

select * from #EXPLORE where tkey between 15000 and 25000 order by tkey;

Индексы - пример

```
checkpoint;                -- фиксация БД
dbcc dropcleanbuffers; -- очистить буферный кэш
go
create  clustered  index #EXPLORE_CLU on #EXPLORE(tkey asc) -- создать инде
go
select * from  #EXPLORE where tkey between 15000 and 25000 order by tkey;
```



Cached plan size	16 B
Degree of Parallelism	0
Estimated Operator Cost	0 (0%)
Estimated Subtree Cost	0,035073
Estimated Number of Rows	3311,68

Statement

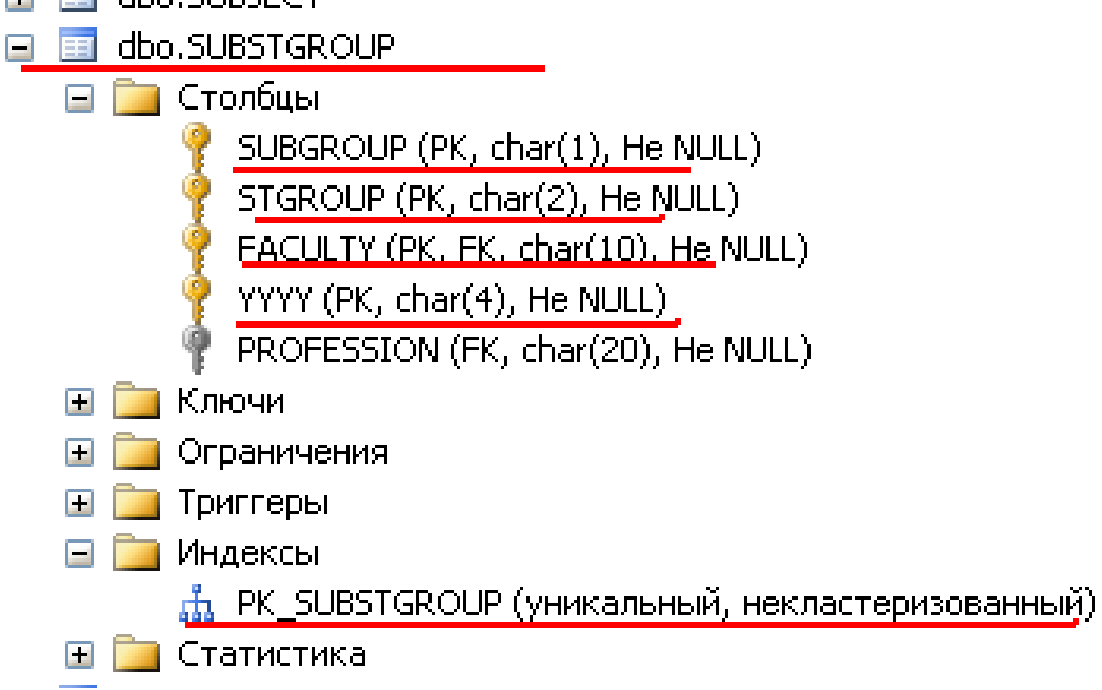
select * from #EXPLORE where tkey

Индексы

- Кластеризованные и некластеризованные
- Уникальные и неуникальные
- Простые и составные
- XML-индексы
- Пространственные индексы

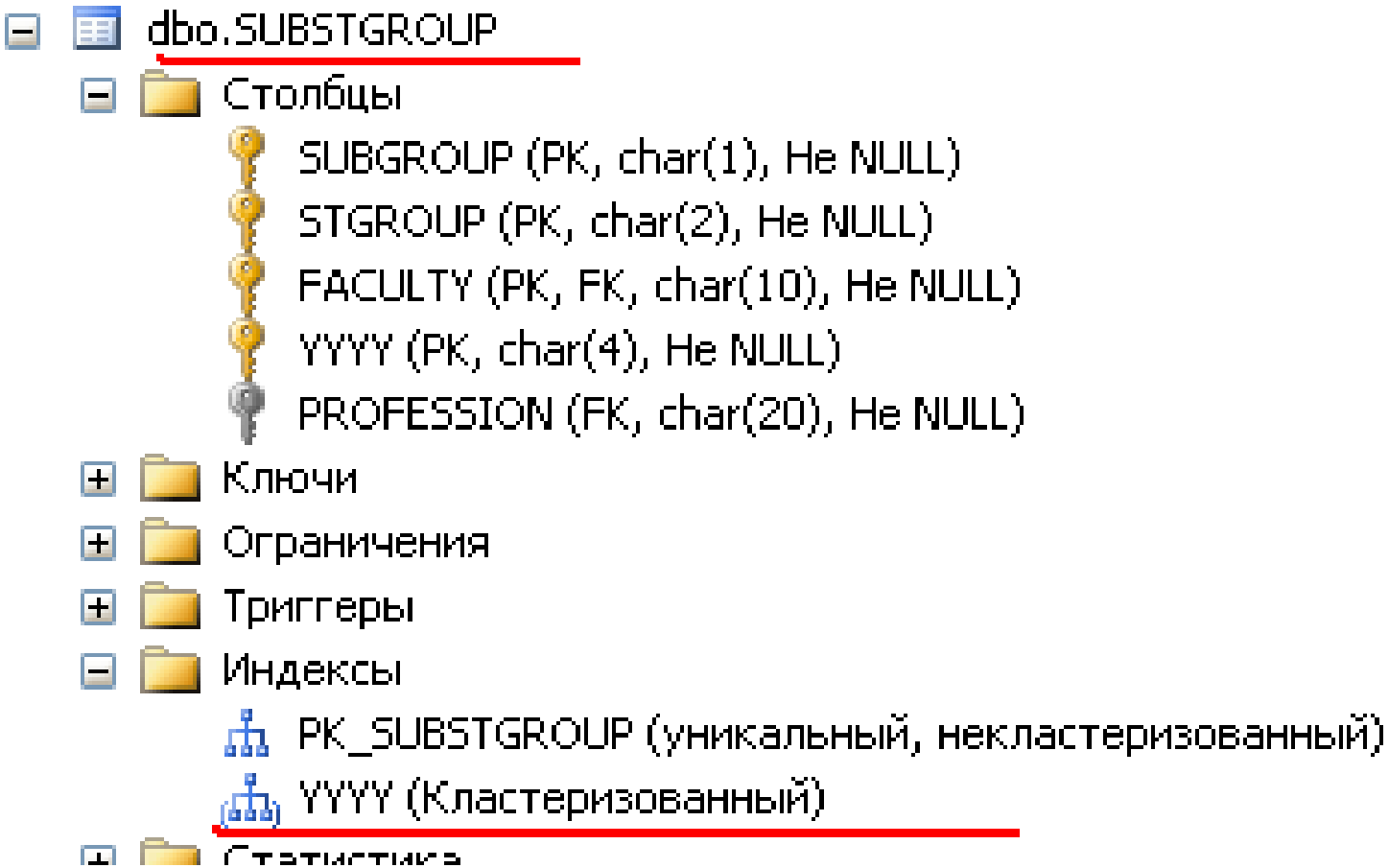
Индексы

```
CREATE TABLE SUBSTGROUP -- студенческая подгруппа
   (
    SUBGROUP    CHAR(1)    NOT NULL,   -- подгруппа
    STGROUP     CHAR(2)    NOT NULL,   -- группа
    FACULTY     CHAR(10)   NOT NULL,   -- факультет
    YYYY        CHAR(4)    NOT NULL,   -- год поступления
    PROFESSION CHAR(20)    NOT NULL,   -- специальность
    CONSTRAINT FK_SUBSTGROUP_FACULTY FOREIGN KEY(FACULTY)    REFERENCES FACULTY(FACULTY),
    CONSTRAINT FK_SUBSTGROUP_PROFESSION FOREIGN KEY(PROFESSION)    REFERENCES PROFESSION(PROFESSION),
    CONSTRAINT PK_SUBSTGROUP PRIMARY KEY NONCLUSTERED(SUBGROUP, STGROUP, FACULTY, YYYY)
   );
```



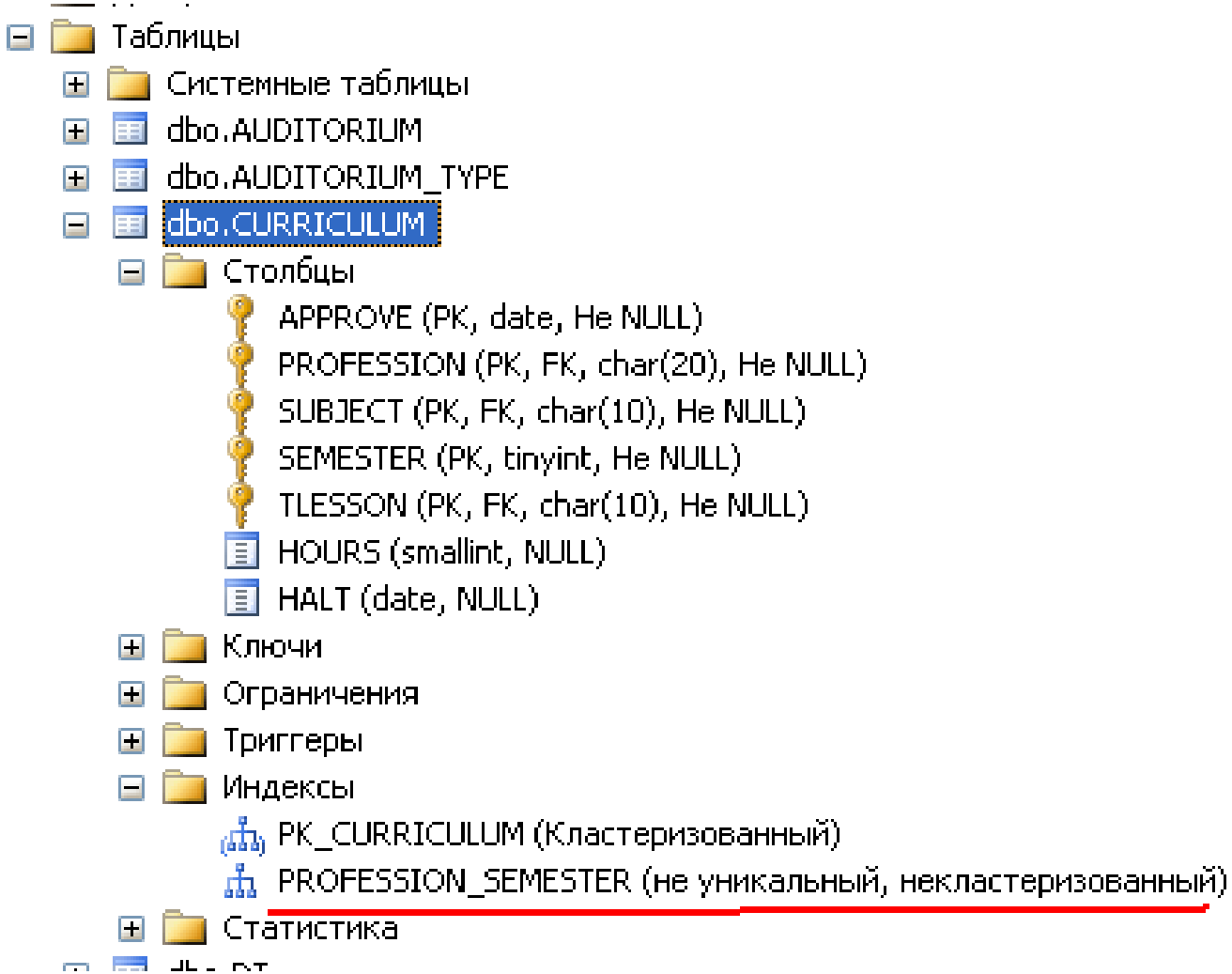
Индексы

```
create  clustered index YYYY on SUBSTGROUP(YYYY);
```



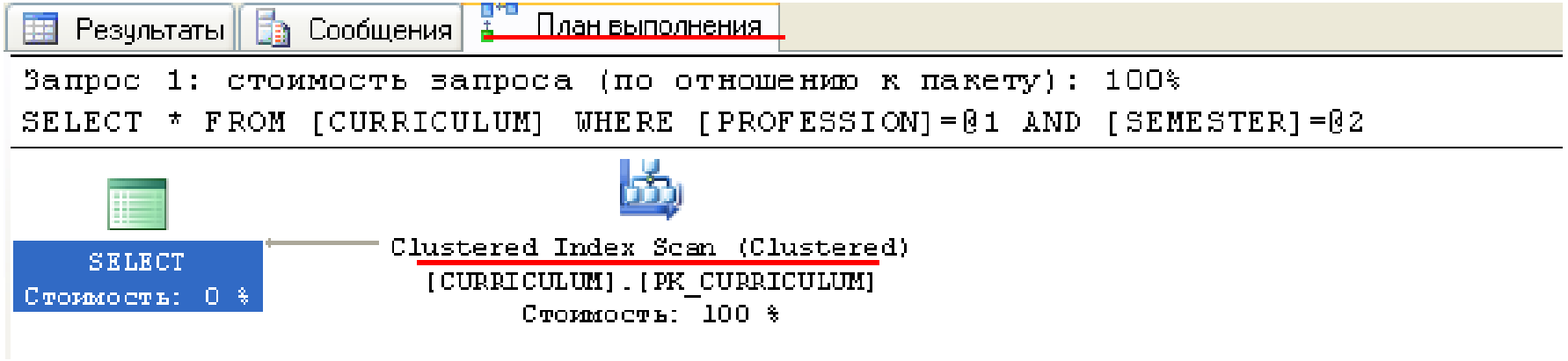
Индексы

```
create  index  PROFESSION_SEMESTER on CURRICULUM(PROFESSION, SEMESTER);
```



Индексы

```
select * from  CURRICULUM where PROFESSION = '1-40 01 02' and SEMESTER = 3
```



Индексы

Clustered Index Scan (Clustered)

Просмотр всего кластеризованного индекса или его части.

Физическая операция	Clustered Index Scan
Логическая операция	Clustered Index Scan
Фактическое количество строк	7
Предполагаемая стоимость операций ввода-вывода	0,003125
Предполагаемая стоимость процессного ресурса	0,0002153
Количество выполнений	1
Предполагаемое количество выполнений	1
Предполагаемая стоимость оператора	0,0033403 (100%)
Предполагаемая стоимость поддерева	0,0033403
Предполагаемое количество строк	7
Предполагаемый размер строки	56 Б
Фактическое число повторных привязок	0
Фактическое число сбросов на начало	0
Отсортировано	False
Идентификатор узла	0

Предикат

[Smelow].[dbo].[CURRICULUM].[SEMESTER]=(3) AND [Smelow].[dbo].[CURRICULUM].[PROFESSION]='1-40 01 02'

Объект

[Smelow].[dbo].[CURRICULUM].[PK_CURRICULUM]

Список столбцов

[Smelow].[dbo].[CURRICULUM].APPROVE; [Smelow].[dbo].[CURRICULUM].PROFESSION; [Smelow].[dbo].[CURRICULUM].SUBJECT; [Smelow].[dbo].[CURRICULUM].SEMESTER; [Smelow].[dbo].[CURRICULUM].TLESSON; [Smelow].[dbo].[CURRICULUM].HOURS; [Smelow].[dbo].[CURRICULUM].HALT

Индексы

```
checkpoint;              -- фиксация БД
dbcc dropcleanbuffers; -- очистить буферный кэш
go
select * from  #EXPLORE where tkey between 15000 and 25000 order by tkey;
```

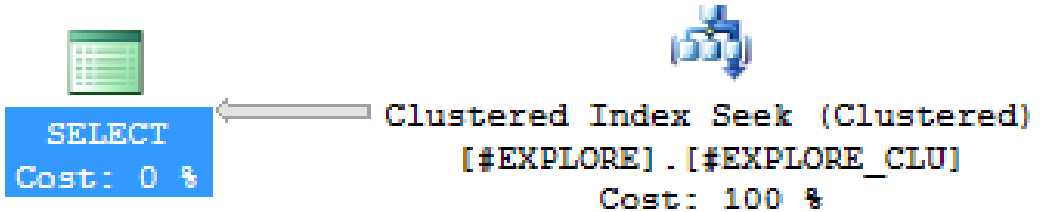
Cached plan size	16 B
Degree of Parallelism	0
Memory Grant	2672
Estimated Operator Cost	0 (0%)
Estimated Subtree Cost	0,17378
Estimated Number of Rows	3311,68

Statement

select * from #EXPLORE where tkey between 15000 and 25000 order by tkey;

Индексы

```
checkpoint;              -- фиксация БД
dbcc dropcleanbuffers; -- очистить буферный кэш
go
create  clustered  index #EXPLORE_CLU on #EXPLORE(tkey asc) -- создать инде
go
select * from  #EXPLORE where tkey between 15000 and 25000 order by tkey;
```



Cached plan size	16 B
Degree of Parallelism	0
Estimated Operator Cost	0 (0%)
Estimated Subtree Cost	0,035073
Estimated Number of Rows	3311,68

Statement

select * from #EXPLORE where tkey

Неуникальные индексы

```
select  t.cc [количество значений], count(*) [количество случаев]
  from (
        select TKEY, count(*) cc
        from #EXPLORE  group by TKEY having  count(*) > 1
        )t
  group by t.cc
  order by [количество случаев] desc
```

количество значений	количество случаев
2	1226
3	142
4	7

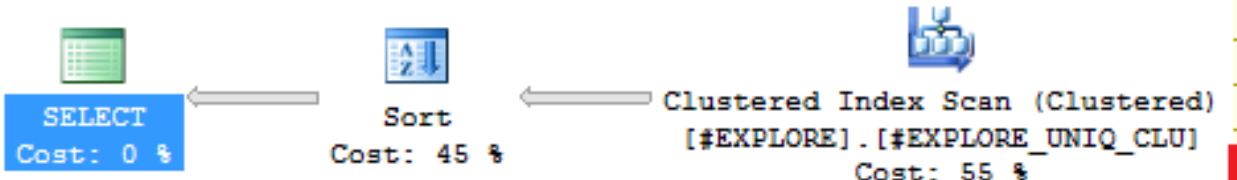
Неуникальные индексы

```
drop index  #EXPLORE_CLU on #EXPLORE
go
---- создание уникального кластеризованного индекса
create unique clustered  index #EXPLORE_UNIQ_CLU on #EXPLORE(tkey asc) -- создать индекс
```

```
Msg 1505, Level 16, State 1, Line 2
The CREATE UNIQUE INDEX statement terminated because a duplicate
key was found for the object name 'dbo.#EXPLORE__000000000001E' and the index name '#EXPLORE_UNIQ_CLU'.
The duplicate key value is (4).
The statement has been terminated.
```

Неуникальные индексы

```
---- создание уникального кластеризованного индекса
create unique clustered  index #EXPLORE_UNIQ_CLU on #EXPLORE(CC asc) -- создать индекс
go
select * from  #EXPLORE where TKEY between 15000 and 25000 order by TKEY;
```



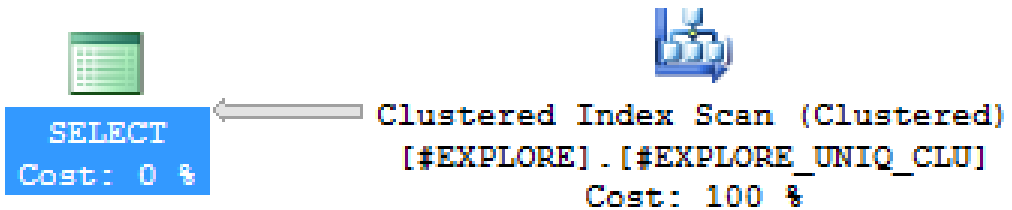
Cached plan size	16 B
Degree of Parallelism	0
Memory Grant	2672
Estimated Operator Cost	0 (0%)
Estimated Subtree Cost	0,178768
Estimated Number of Rows	3321,96

Statement

select * from #EXPLORE where TKEY between 15000 and 25000 order by TKEY;

Неуникальные индексы

```
select * from  #EXPLORE where TKEY between 15000 and 25000 order by CC;
```



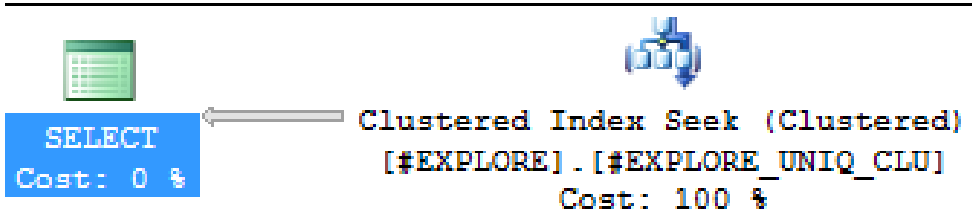
Cached plan size	16 B
Degree of Parallelism	0
Estimated Operator Cost	0 (0%)
Estimated Subtree Cost	0,0979857
Estimated Number of Rows	3321,96

Statement

select * from #EXPLORE where TKEY between 15000 and 25000 order by CC;

Неуникальные индексы

```
select * from  #EXPLORE where CC between 4500 and  6500 order by CC;
```



Cached plan size	16 B
Degree of Parallelism	0
Estimated Operator Cost	0 (0%)
Estimated Subtree Cost	0,0217795
Estimated Number of Rows	2001,07

Statement

select * from #EXPLORE where CC between 4500 and 6500 order by CC;

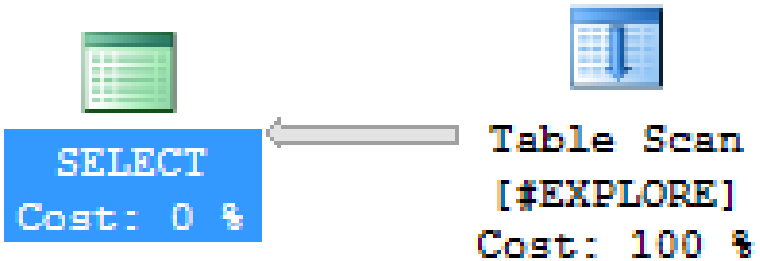
Составной индекс

```
create   index #EXPLORE_NONCLU on #EXPLORE(TKEY, CC)
go
exec SP_HELPINDEX '#EXPLORE'
```

index_name	index_description	index_keys
#EXPLORE_NONCLU	nonclustered located on PRIMARY	TKEY, CC

Некластеризованный составной индекс

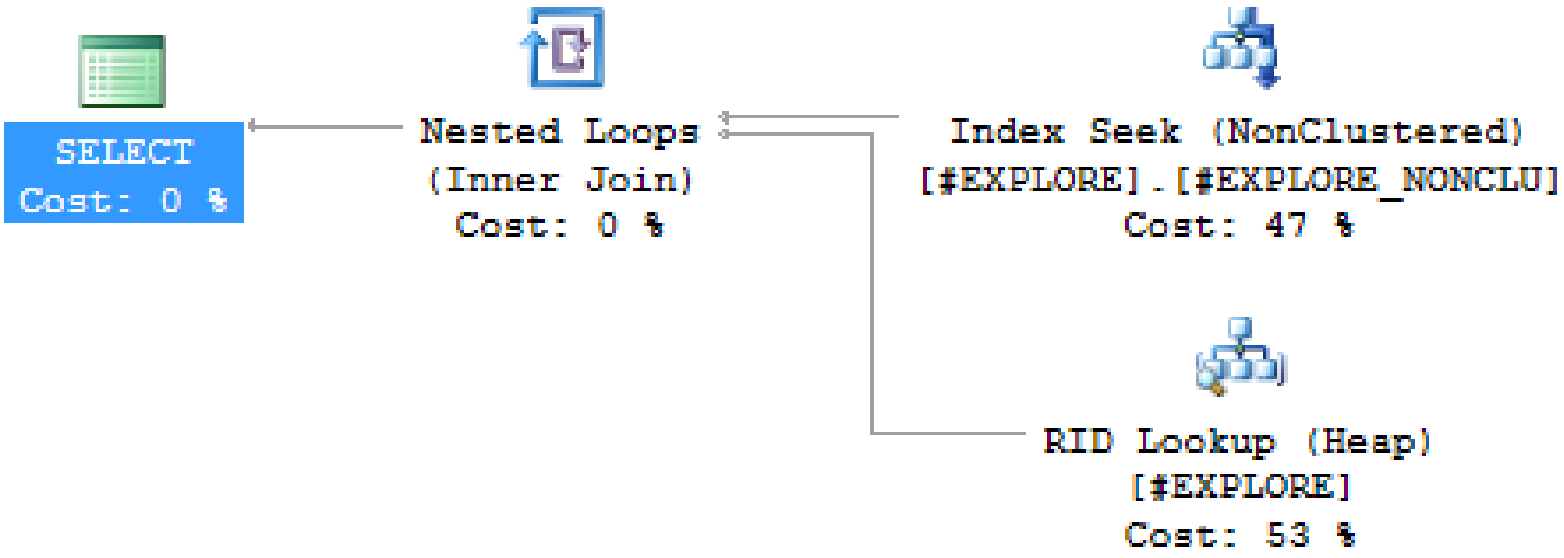
```
select * from  #EXPLORE where  TKEY > 1500 and  CC < 4500;
```




```
select * from  #EXPLORE order by TKEY, CC;
```

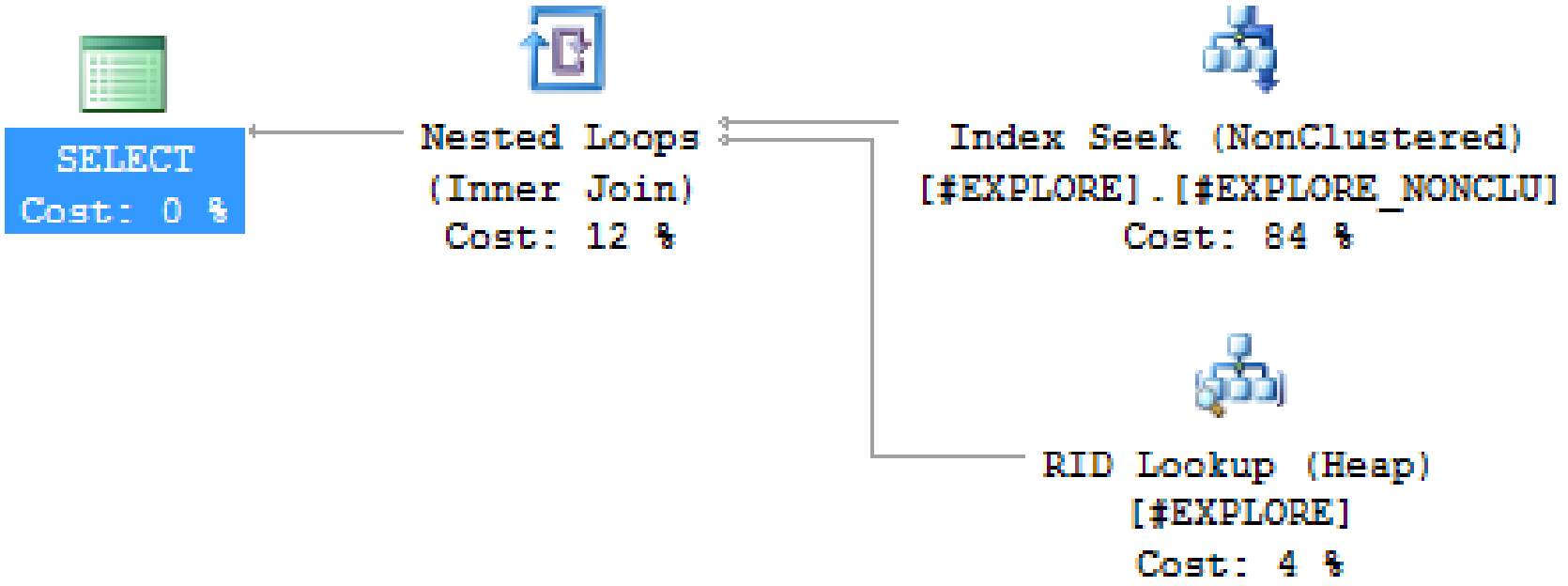
Некластеризованный составной индекс

```
select * from  #EXPLORE where  TKEY = 556 and  CC > 3
```



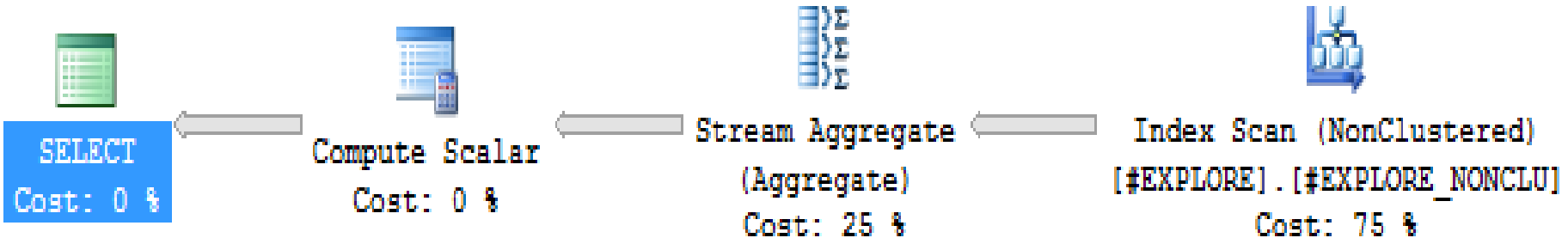
Некластеризованный составной индекс

```
select * from  #EXPLORE where  TKEY > 1000 and  CC = 2222
```



Составной индекс

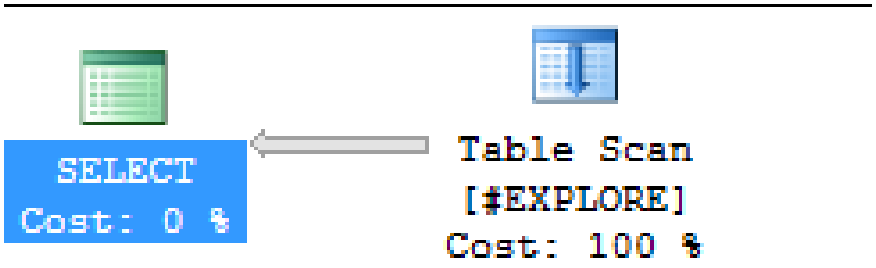
```
select max(TKEY), max(CC), count(*) from #EXPLORE group by TKEY, CC;
```



Индекс

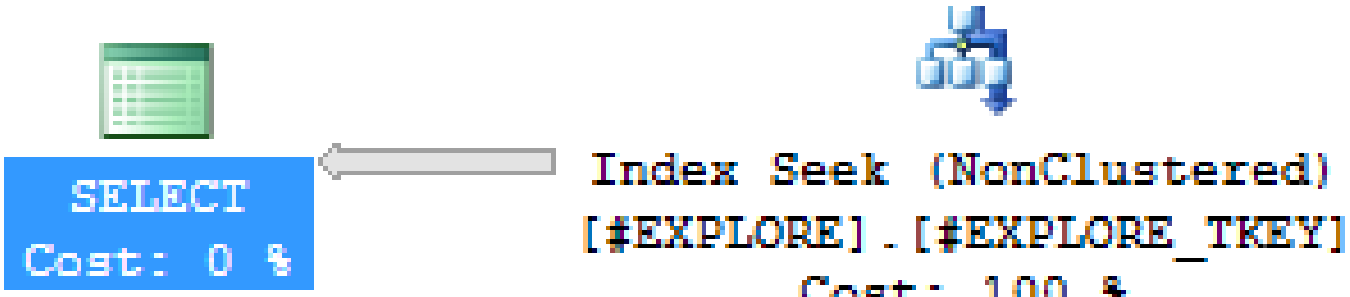
```
drop index #EXPLORE_NONCLU on #EXPLORE; -- удаление индекса
go
create   index #EXPLORE_TKEY on #EXPLORE(TKEY);
go
create   index #EXPLORE_CC on #EXPLORE(CC);
go

select * from  #EXPLORE where TKEY > 1500 and CC < 4500;
select * from  #EXPLORE where TKEY > 1500;
select * from  #EXPLORE where CC < 4500;
```

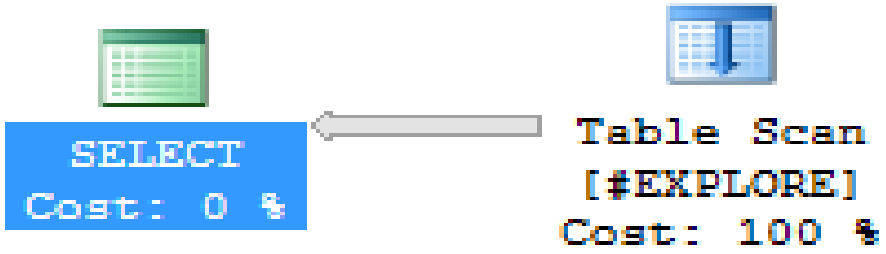


Индекс

```
select TKEY from  #EXPLORE where TKEY > 1500;
```




```
select CC      from  #EXPLORE where TKEY > 1500;
select TFIELD  from  #EXPLORE where TKEY > 1500;
```



Индекс покрытия

- Включение всех столбцов запроса в индекс может значительно повысить производительность запроса
- Фильтруемые столбцы должны быть первыми ключевыми столбцами в индексе

```
create  index #EXPLORE_WHERE_I  on #EXPLORE(TKEY) include(CC)

WHERE (TKEY >= 15000 and CC < 45000);
```

INCLUDE

- INCLUDE позволяет указать столбцы, которые добавляются к страницам листьев некластеризованного индекса
- Имена столбцов в списке INLCUDE не должны повторяться
- Если все столбцы запроса включены в индекс, то оптимизатор запросов может определить местонахождение всех значений столбцов по страницам индекса, не обращаясь к данным в таблице

Фильтрующие индексы

- Фильтрующий индекс создается только для строк таблицы, которые удовлетворяют логическому условию
- Если множество строк, выбираемое WHERE-выражением SELECT-запроса, является подмножеством множества индексируемых фильтрующим индексом строк, оптимизатор будет его применять в плане запроса
- Можно объединять с индексами покрытия

Фильтрующие индексы

```
use TEMPDB
go
create  index #EXPLORE_WHERE  on #EXPLORE(TKEY) WHERE (TKEY >= 15000 and TKEY <= 20000)
go
exec SP_HELPINDEX '#EXPLORE';
```

index_name	index_description	index_keys
#EXPLORE_TKEY	nonclustered located on PRIMARY	TKEY
#EXPLORE_WHERE	nonclustered located on PRIMARY	TKEY

```
create  index #EXPLORE_WHERE_I  on #EXPLORE(TKEY)
include(CC)  WHERE (TKEY >= 15000 and CC < 45000);
```

Вопросы?