

IBM® DB2 Universal Database™



# Учебник по Наглядному объяснению

*Версия 8*



IBM® DB2 Universal Database™



# Учебник по Наглядному объяснению

*Версия 8*

Перед тем как использовать данный документ и продукт, описанный в нем, прочтите общие сведения под заголовком *Замечания*.

Этот документ содержит информацию, которая является собственностью IBM. Она предоставляется в соответствии с лицензионным соглашением и защищена законами об авторском праве. Информация в данной публикации не включает никаких гарантий на продукт и никакое из утверждений в данном руководстве не следует понимать подобным образом.

Заказать публикации IBM можно через Интернет или у местного представителя IBM.

- Чтобы заказать публикации через Интернет, перейдите на Web-страницу Центра публикаций IBM (IBM Publications Center): [www.ibm.com/shop/publications/order](http://www.ibm.com/shop/publications/order)
- Чтобы найти местное представительство IBM, перейдите на страницу IBM Directory of Worldwide Contacts по адресу [www.ibm.com/planetwide](http://www.ibm.com/planetwide)

Чтобы заказать публикации DB2 через отдел DB2 Marketing and Sales в Соединенных Штатах или Канаде, позвоните по телефону 1-800-IBM-4YOU (426-4968).

Отсылая информацию IBM, вы тем самым даете IBM неисключительное право использовать или распространять эту информацию любым способом, как фирма сочтет нужным, без каких-либо обязательств перед вами.

© Copyright International Business Machines Corporation 2000 - 2002. Все права защищены.

# Содержание

<b>Об этом учебнике.</b> . . . . .	<b>v</b>
Информация, зависящая от среды . . . . .	vi
<b>Урок 1. Создание снимков объяснения . . . . .</b>	<b>1</b>
Создание таблиц объяснения . . . . .	1
Применение снимков объяснения . . . . .	2
Создание снимков объяснения для операторов динамического SQL . . . . .	3
Создание снимков объяснения для операторов статического SQL. . . . .	4
Что дальше . . . . .	5
<b>Урок 2. Просмотр и работа с графом плана доступа . . . . .</b>	<b>7</b>
Просмотр графа плана доступа путем выбора элемента в списке ранее объясненных операторов SQL . . . . .	7
Чтение символов на графе плана доступа . . . . .	7
Увеличение частей графа плана доступа с помощью регулятора масштаба . . . . .	8
Получение дополнительных сведений об объектах, изображенных на графе . . . . .	9
Получение статистики для таблиц, индексов и функций таблицы . . . . .	9
Получение сведений об операторах, изображенных на графе . . . . .	10
Получение статистики для функций . . . . .	10
Получение статистики для табличных пространств . . . . .	10
Получение статистики для столбцов в операторе SQL . . . . .	11
Получение информации о параметрах конфигурации и опциях связывания . . . . .	11
Изменение вида графа плана доступа . . . . .	11
Что дальше . . . . .	12
<b>Урок 3. Улучшение плана доступа для однораздельной базы данных. . . . .</b>	<b>13</b>
Работа с графами плана доступа . . . . .	13
Выполнение запроса без индексов и статистики . . . . .	14
Сбор текущей статистики для таблиц и индексов командой runstats . . . . .	17
Создание индексов в столбцах, используемых для объединения таблиц в запросе. . . . .	21
Создание дополнительных индексов в столбцах таблицы . . . . .	27
Что дальше . . . . .	29
<b>Урок 4. Улучшение плана доступа для многораздельной базы данных . . . . .</b>	<b>31</b>
Работа с графами плана доступа . . . . .	31
Выполнение запроса без индексов и статистики . . . . .	32
Сбор текущей статистики для таблиц и индексов командой runstats . . . . .	35
Создание индексов в столбцах, используемых для объединения таблиц в запросе. . . . .	39
Создание дополнительных индексов в столбцах таблицы . . . . .	43
Что дальше . . . . .	47
<b>Приложение А. Основные понятия, применяемые в программе Наглядное объяснение . . . . .</b>	<b>49</b>
План доступа . . . . .	49
Граф плана доступа . . . . .	49
Узел на графе плана доступа. . . . .	50
Кластеризация . . . . .	51
Контейнер . . . . .	51
Стоимость . . . . .	51
Указатель с блокированием . . . . .	52
Табличное пространство, управляемое базой данных (DMS) . . . . .	52
Динамический SQL . . . . .	53
Снимок объяснения . . . . .	53
Объяснимый оператор. . . . .	54
Объясненный оператор . . . . .	54
Операнд . . . . .	54
Оператор . . . . .	54
CMPEXP . . . . .	56
DELETE . . . . .	56
EISCAN . . . . .	56
FETCH. . . . .	57
FILTER. . . . .	57
GENROW . . . . .	58

GRPBY . . . . .	58	FETCH . . . . .	76
HSJOIN . . . . .	59	FILTER . . . . .	76
INSERT . . . . .	59	GENROW . . . . .	76
IXAND . . . . .	60	GRPBY . . . . .	77
IXSCAN . . . . .	60	HSJOIN . . . . .	77
MSJOIN . . . . .	61	INSERT . . . . .	78
NLJOIN . . . . .	62	IXAND . . . . .	78
PIPE . . . . .	63	IXSCAN . . . . .	79
RETURN . . . . .	63	MSJOIN . . . . .	80
RIDSCN . . . . .	63	NLJOIN . . . . .	81
RQUERY . . . . .	64	PIPE . . . . .	82
SORT . . . . .	64	RETURN . . . . .	82
TBSCAN . . . . .	65	RIDSCN . . . . .	82
TEMP . . . . .	66	RQUERY . . . . .	82
TQUEUE . . . . .	66	SORT . . . . .	83
UNION . . . . .	66	TBSCAN . . . . .	84
UNIQUE . . . . .	67	TEMP . . . . .	85
UPDATE . . . . .	67	TQUEUE . . . . .	85
Оптимизатор . . . . .	67	UNION . . . . .	85
Пакет . . . . .	67	UNIQUE . . . . .	86
Предикат . . . . .	68	UPDATE . . . . .	86
Класс оптимизации запроса . . . . .	69		
Избирательность предикатов . . . . .	70	<b>Приложение С. Основы DB2 . . . . .</b>	<b>87</b>
Объединение типа "звезда" . . . . .	70	Базы данных . . . . .	87
Статический SQL . . . . .	71	Схемы . . . . .	87
Табличные пространства, управляемые		Таблицы . . . . .	88
системой (SMS) . . . . .	72		
Табличное пространство . . . . .	72	<b>Приложение D. Замечания . . . . .</b>	<b>89</b>
Наглядное объяснение . . . . .	72	Товарные знаки . . . . .	92
<b>Приложение В. Алфавитный список</b>		<b>Индекс . . . . .</b>	<b>95</b>
<b>операторов Наглядного объяснения . . . . .</b>	<b>75</b>		
CMPEXP . . . . .	75	<b>Как связаться с IBM . . . . .</b>	<b>97</b>
DELETE . . . . .	75	Информация о продукте . . . . .	97
EISCAN . . . . .	75		

---

## Об этом учебнике

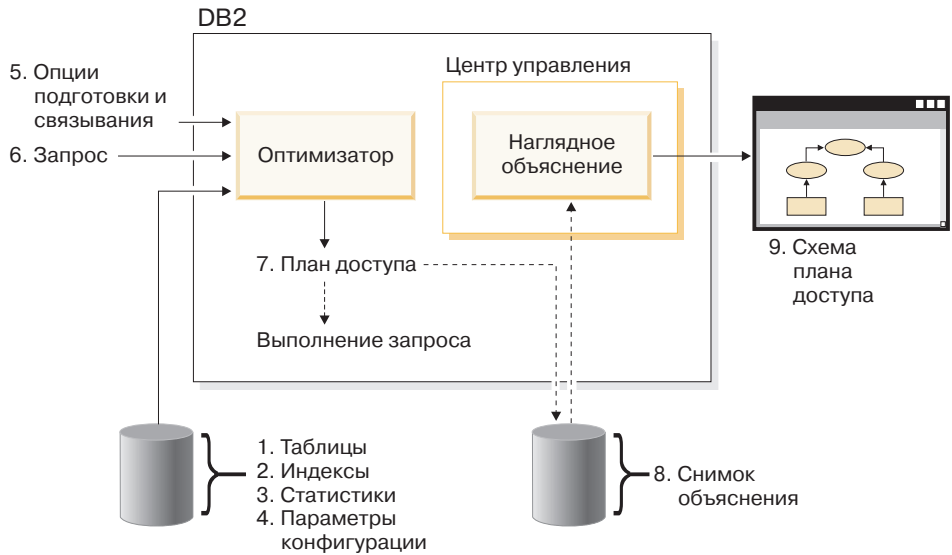
Этот учебник представляет собой руководство по возможностям функции Наглядное объяснение DB2. В уроках этого учебника описано применение Наглядного объяснения для просмотра графического представления плана доступа объясненных операторов SQL. Кроме того, описано применение полученной информации для настройки производительности запросов SQL.

С помощью оптимизатора DB2 проверяет запросы SQL и определяет оптимальный способ доступа к данным. Этот путь называется планом доступа. DB2 позволяет ознакомиться с действиями оптимизатора, просмотрев план доступа, выбранный для конкретного запроса SQL. Наглядное объяснение позволяет просмотреть план доступа в виде графа. Граф - это наглядное представление объектов базы данных, связанных с запросом (например, таблиц и индексов). Он также включает операции, выполняемые над этими объектами (например, просмотра и сортировки), и потоки данных.

Для ускорения доступа к данным при выполнении запроса можно выполнить любые или все из следующих действий:

1. Настроить формат таблицы и реорганизовать данные в ней.
2. Создать подходящие индексы.
3. Собрать текущую статистику, необходимую оптимизатору, с помощью команды **runstats**.
4. Выбрать подходящие параметры конфигурации.
5. Выбрать подходящие опции связывания.
6. Создать запросы для получения только необходимых данных.
7. Использовать план доступа.
8. Создать снимки объяснения.
9. Использовать для улучшения плана доступа граф плана доступа.

Эти действия соответствуют показанным на рисунке. (Прерывистые линии обозначают действия, обязательные в Наглядном объяснении.)



Этот учебник содержит уроки по:

- Созданию снимков объяснения. Они необходимы для просмотра графов плана доступа.
- Просмотру и работе с графом плана доступа.
- Настройке производительности и проверке изменений плана доступа.

**Примечание:** Настройка производительности включает отдельные уроки для однораздельной и многораздельной сред.

Для работы с уроками с DB2 поставляется база данных SAMPLE. Инструкции по созданию этой базы данных, если она отсутствует, приведены в руководстве *Administration Guide*.

## Информация, зависящая от среды



Информация, помеченная этим значком, относится только к средам single-partition database.



Информация, помеченная этим значком, относится только к средам partitioned database.



---

## Урок 1. Создание снимков объяснения

Это занятие описывает создание снимков объяснения. Возможность объяснения SQL применяется для захвата информации о среде, в которой компилируется оператор статического или динамического SQL. Эта информация позволит понять структуру и потенциальную производительность операторов SQL. Снимок объяснения - это краткая информация, которая собирается во время объяснения оператора SQL. Она хранится в виде двоичного объекта большого размера (BLOB) в таблице EXPLAIN\_STATEMENT. Снимок содержит следующую информацию:

- Внутреннее представление плана доступа, в том числе его операторов, а также используемых таблиц и индексов.
- Критерий, используемый оптимизатором для принятия решения. Он включает в себя статистическую информацию об объектах базы данных и общие затраты на выполнение каждого оператора.

Для просмотра графа плана доступа функции Наглядное объяснение требуется информация снимка объяснения.

---

### Создание таблиц объяснения

Перед созданием снимков объяснения необходимо убедиться, что для вашего ID пользователя созданы следующие таблицы объяснения:

- EXPLAIN\_INSTANCE
- EXPLAIN\_STATEMENT

Для того чтобы проверить, существуют ли эти таблицы, введите команду **DB2 list tables**. Если эти таблицы еще не существуют, то необходимо создать их, выполнив следующие действия:

1. Если DB2 еще не запущена, выполните команду **db2start**.
2. Из командной строки CLP DB2 установите соединение с нужной базой данных. Для выполнения данного урока подключитесь к базе данных SAMPLE командой **connect to sample**.
3. Создайте таблицы объяснения, воспользовавшись образцом командного файла, содержащимся в файле EXPLAIN.DDL в каталоге sqllib/misc. Для запуска командного файла перейдите в этот каталог и введите команду **db2 -tf EXPLAIN.DDL**. Указанный командный файл создает таблицы объяснения, префикс которых представляет собой ID пользователя, установившего соединение. У этого ID пользователя должны быть права доступа CREATETAB (для доступа к базе данных), либо права доступа SYSADM или DBADM.

---

## Применение снимков объяснения

Для изучения функции Наглядное объяснение поставляются четыре примера снимков. Информация о создании собственных снимков приведена в следующих разделах; для выполнения данного урока создание собственных снимков не требуется.

- Создание снимков объяснения для операторов динамического SQL
- Создание снимков объяснения для операторов статического SQL.

В данном запросе к примеру снимка указаны имя, отдел и размер заработной платы всех рядовых сотрудников, зарплата которых составляет более 90% от зарплаты наиболее высокооплачиваемого менеджера.

```
SELECT S.ID,S.NAME,O.DEPTNAME,SALARY+COMM
FROM ORG O, STAFF S
WHERE
  O.DEPTNUMB = S.DEPT AND
  S.JOB <> 'Mgr' AND
  S.SALARY+S.COMM > ALL( SELECT ST.SALARY*.9
                        FROM STAFF ST
                        WHERE ST.JOB='Mgr' )
ORDER BY S.NAME
```

Запрос состоит из двух частей:

1. Подзапрос (в круглых скобках) создает строки данных, содержащие 90% от зарплаты каждого менеджера. Так как подзапрос определен как ALL, то из этой таблицы считывается только наибольшее значение.
2. Основной запрос объединяет все строки таблиц ORG и STAFF с одинаковыми номерами отделов, в которых значение JOB не равно 'Mgr', а сумма оклада и комиссионного вознаграждения превышает значение, возвращенное подзапросом.

Основной запрос содержит следующие три предиката (оператора сравнения):

1. O.DEPTNUMB = S.DEPT
2. S.JOB <> 'Mgr'
3. S.SALARY+S.COMM > ALL ( SELECT ST.SALARY\*.9  
FROM STAFF ST  
WHERE ST.JOB='Mgr' )

Это следующие предикаты (в том же порядке):



1. Предикат объединения, который объединяет таблицы ORG и STAFF, если номера отделов совпадают
2. Локальный предикат для столбца JOB таблицы STAFF
3. Локальный предикат для столбцов SALARY и COMM таблицы STAFF, использующий результат подзапроса.

Для загрузки примера снимка:

1. Если DB2 еще не запущена, выполните команду **db2start**.
2. Если в вашей базе данных нет таблиц объяснения, создайте их. Для этого выполните инструкции, приведенные в разделе Создание таблиц объяснения.
3. Подключитесь к нужной базе данных. В данном учебнике описана работа с базой данных SAMPLE. Для подключения к базе данных SAMPLE введите в командной строке DB2 команду **connect to sample**.

Если она еще не создана, обратитесь к разделу по установке базы данных SAMPLE в книге *Administration Guide*.

4. Для импорта предопределенных снимков запустите командный файл DB2 с именем VESAMPL.DDL.

-  Этот файл находится в каталоге sqllib\samples\ve.
-  Этот файл находится в каталоге sqllib\samples\ve\inter.

Для запуска командного файла перейдите в указанный каталог и введите команду **db2 -tf vesampl.ddl**.

- Этот командный файл следует запустить с тем же ID пользователя, который применялся для создания таблиц объяснения.
- Данный командный файл импортирует только предопределенные снимки. Он не создает каких-либо таблиц или данных. Описанные ниже действия по настройке (например, CREATE INDEX и runstats) выполняются над таблицами и данными базы данных SAMPLE.

Теперь вы готовы просматривать графы плана доступа и работать с ними.

## Создание снимков объяснения для операторов динамического SQL

**Примечание:** Информация о создании снимков объяснения приведена в данном разделе только для справки. Благодаря наличию предопределенных снимков объяснения для выполнения этого урока создание снимков не требуется.

Для того чтобы создать снимок объяснения для оператора динамического SQL, выполните следующие действия:

1. Если DB2 еще не запущена, выполните команду **db2start**.
2. Если в вашей базе данных нет таблиц объяснения, создайте их. Для этого выполните инструкции, приведенные в разделе Создание таблиц объяснения.
3. Из командной строки CLP DB2 установите соединение с нужной базой данных. Например, для подключения к базе данных SAMPLE выполните команду **connect to sample**.

Для создания базы данных SAMPLE обратитесь к разделу, посвященному установке базы данных SAMPLE, в книге *Administration Guide*.

4. С помощью одной из следующих команд, введенных в командной строке CLP DB2, создайте снимок объяснения для оператора динамического SQL:
  - Для создания снимка объяснения без выполнения оператора SQL введите команду **set current explain snapshot=explain**.
  - Для создания снимка объяснения и выполнения оператора SQL введите команду **set current explain snapshot=yes**.

Эта команда устанавливает специальный регистр объяснения. Как только этот регистр установлен, его будут использовать все последующие операторы SQL. Более подробная информация приведена в разделах, посвященных текущим снимкам объяснения, в книге *SQL Reference*.

5. Введите операторы SQL в командной строке CLP DB2.
6. Для того чтобы просмотреть граф плана доступа для снимка, обновите окно Explained Statements History (доступное из Control Center) и дважды щелкните на снимке.
7. Необязательно: для того чтобы отключить средство работы со снимками, после передачи на выполнение операторов SQL запустите команду **set current explain snapshot=no**.

## Создание снимков объяснения для операторов статического SQL.

**Примечание:** Информация о создании снимков объяснения приведена в данном разделе только для справки. Благодаря наличию предопределенных снимков объяснения для выполнения этого урока создание снимков не требуется.

Для того чтобы создать снимок объяснения для оператора статического SQL, выполните следующие действия:

1. Если DB2 еще не запущена, выполните команду **db2start**.
2. Если в вашей базе данных нет таблиц объяснения, создайте их. Для этого выполните инструкции, приведенные в разделе Создание таблиц объяснения.
3. Из командной строки CLP DB2 установите соединение с нужной базой данных. Например, для подключения к базе данных SAMPLE выполните команду **connect to sample**.
4. Создайте для оператора статического SQL снимок объяснения, воспользовавшись опцией EXPLSNAP при связывании или подготовке вашего приложения. Например, введите команду **bind файл explsnap yes**.
5. Необязательно: Для того чтобы просмотреть граф плана доступа для снимка, обновите окно Explained Statements History (доступное из Control Center) и дважды щелкните на снимке.

Информацию об использовании опции EXPLSNAP для эквивалентных API можно найти в разделах по соответствующим API в книге *Application Development Guide*.

---

## Что дальше

Раздел “Урок 2. Просмотр и работа с графом плана доступа” на стр. 7 описывает просмотр графа плана доступа и содержимое этого графа.



---

## Урок 2. Просмотр и работа с графом плана доступа

Это занятие описывает применение окна Граф плана доступа для просмотра и работы с графом плана доступа. Граф плана доступа - это графическое представление плана доступа. Граф плана доступа содержит сведения о:

- Таблицах (и связанных с ними столбцах) и индексах
- Операторах (например, операторах просмотра, сортировки и объединения таблиц)
- Табличных пространствах и функциях.

Для просмотра графа плана доступа можно:

- Выбрать элемент в списке ранее объясненных операторов.
- Выбрать элемент в списке объяснимых операторов в пакете.
- Динамически объяснить оператор SQL.

Для работы с графами плана доступа для примеров снимков объяснения, загруженных в уроке 1, следует выбрать элемент в списке ранее объясненных операторов. Информация о других способах просмотра графов плана доступа приведена в справке по Наглядному объяснению.

---

### Просмотр графа плана доступа путем выбора элемента в списке ранее объясненных операторов SQL

Для того чтобы просмотреть граф плана доступа при выборе оператора из списка ранее объясненных операторов SQL:

1. В Центре управления найдите в дереве объектов базу данных SAMPLE.
2. Щелкните на значке базы данных правой кнопкой и выберите в меню пункт **Показать хронологию объясненных операторов**. Откроется окно Хронология объясненных операторов.
3. Просмотреть граф плана доступа можно только для тех операторов, для которых есть снимок объяснения. Для таких операторов в столбце **Снимок объяснения** будет показано значение Да. Дважды щелкните на записи, для которой в столбце Номер запроса указано 1 (возможно, для просмотра столбца **Номер запроса** потребуется прокрутить окно вправо). Будет показано окно Граф плана доступа для оператора.

**Примечание:** Граф читается снизу вверх. Первое действие запроса показано в нижней части графа, последнее - в верхней.

### Чтение символов на графе плана доступа

Граф плана доступа изображает структуру плана в виде дерева. *Узлы* дерева представляют:

- Таблицы, которые изображаются в виде прямоугольников
- Индексы, которые изображаются в виде ромбов
- Операторы (восьмиугольники). Операторы TQUEUE (параллелограммы)
- Табличные функции (шестиугольники).

Для операторов стоящий справа номер в скобках - это уникальный идентификатор узла. Номер, стоящий под оператором, - это совокупные затраты.

## **Увеличение частей графа плана доступа с помощью регулятора масштаба**

При просмотре графа плана доступа он выдается на экран полностью, поэтому мелкие детали узлов графа различимы не всегда.

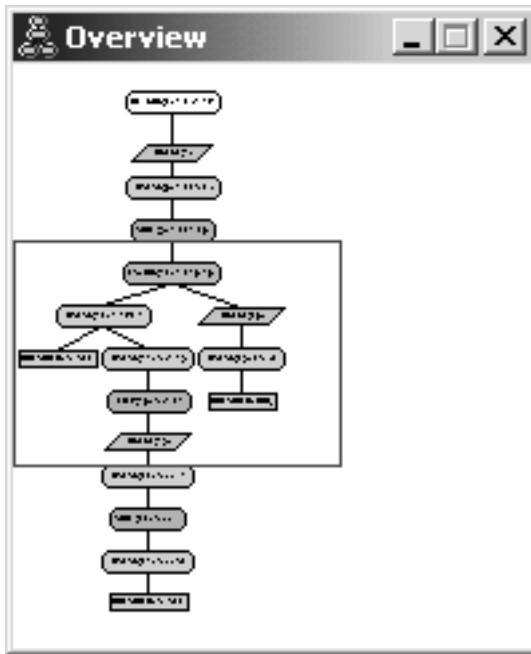
Для увеличения отдельных участков графа плана доступа в окне Граф плана доступа можно воспользоваться **регулятором масштаба**.

1. Установите указатель мыши на бегунок на полосе прокрутки регулятора масштаба в левой части графа плана доступа.
2. Переместите регулятор для выбора нужного масштаба.

Для просмотра различных частей графа плана доступа воспользуйтесь полосой прокрутки.

Для просмотра большого и сложного графа плана доступа воспользуйтесь окном Обзор графа. Это окно позволяет определить и изменить просматриваемую часть графа. В основном окне будет показана часть графа, находящаяся в рамке увеличения.





Для прокрутки окна с графом поместите указатель мыши над выделенной областью в окне Обзор графа, нажмите левую кнопку мыши и, удерживая ее, переместите выделенную область.

## Получение дополнительных сведений об объектах, изображенных на графе

Для каждого объекта графа плана доступа можно просмотреть дополнительную информацию. Можно просмотреть:

- Статистику системного каталога для таких объектов, как:
  - Таблицы, индексы или функции таблицы
  - Информацию об операторах, например, такую как затраты на них, свойства и входные параметры
  - Встроенные и пользовательские функции
  - Табличные пространства
  - Столбцы ссылок в операторе SQL
- Информацию о параметрах конфигурации и опциях связывания (параметры оптимизации).

### Получение статистики для таблиц, индексов и функций таблицы

Для того чтобы просмотреть статистику каталога для таблицы (прямоугольник), индекса (ромб) или табличной функции (шестиугольник), изображенных на графе, дважды щелкните на соответствующем узле. Для каждого из выбранных объектов появится окно Статистика, в котором будет

показана статистическая информация, полученная во время создания снимка, а также информация, существующая на данный момент в таблицах системного каталога.

Для того чтобы просмотреть статистику каталога для *нескольких* таблиц, индексов или функций таблиц, изображенных на графе, выберите их, щелкнув на соответствующих узлах (они будут выделены цветом); а затем выберите **Узел→Показать статистику**. Для каждого из выбранных объектов будет показано окно Статистика. (Окна могут закрывать друг друга; для работы может потребоваться переместить часть из них.)

Если в строке **STATS\_TIME** в столбце **Объясненные** содержится запись **Статистика не обновлялась**, это означает, что во время создания оптимизатором плана доступа статистики не существовало. Следовательно, если оптимизатору для создания плана доступа была нужна определенная статистическая информация, он использовал значения по умолчанию. В этом случае статистика обозначается строкой (**по умолчанию**).

### Получение сведений об операторах, изображенных на графе

Для просмотра статистики каталога для одного оператора (восьмиугольник), дважды щелкните на соответствующем узле. Будет показано окно Сведения об операторе со следующей информацией:

- Оценка совокупных затрат (затраты на ввод-вывод, команды CPU и полные затраты)
- Число выбранных элементов (оценка числа выбранных строк) на данный момент
- Таблицы в плане доступа, использованные и объединенные на данный момент
- Столбцы в таблицах, использованные на данный момент
- Предикаты, примененные на данный момент, включая их избирательность
- Входные параметры для каждого оператора.

Для просмотра сведений о *нескольких* операторах выберите каждый из них, щелкнув на нем, затем выберите **Узел→Показать сведения**. Для каждого из выбранных объектов будет показано окно Статистика. (Окна могут закрывать друг друга; для работы может потребоваться переместить часть из них.)

### Получение статистики для функций

Для просмотра статистики каталога для встроенных и пользовательских функций выберите **Оператор→Показать статистику→Функции**. В списке, показанном в окне Функции, выберите одну или несколько записей и нажмите кнопку **ОК**. Для каждой из выбранных функций появится окно Function Statistics.

### Получение статистики для табличных пространств

Для просмотра статистики каталога для табличных пространств выберите **Оператор→Показать статистику→Табличные пространства**. В списке, показанном в окне Табличные пространства, выберите одну или несколько

записей и нажмите кнопку **ОК**. Для каждого выбранного табличного пространства откроется окно Table Space Statistics.

## Получение статистики для столбцов в операторе SQL

Для просмотра статистики для столбцов, указанных в операторе:

1. Дважды щелкните на таблице в графе плана доступа. Будет показано окно Статистика таблицы.
2. Нажмите кнопку **Обращения к столбцам**. Откроется окно Referenced Columns со списком столбцов таблицы.
3. Выберите в списке один или несколько столбцов и нажмите кнопку **ОК**. Для каждого выбранного столбца откроется окно Referenced Column Statistics.

## Получение информации о параметрах конфигурации и опциях связывания

Для просмотра информации о параметрах конфигурации и опциях связывания (параметрах оптимизации) выберите **Оператор→Показать параметры оптимизации** в окне Граф плана доступа. Откроется окно Optimization Parameters со значениями параметров, действовавшими во время создания снимка, и текущими значениями.

---

## Изменение вида графа плана доступа

Как изменить различные параметры вида графа плана доступа:

1. В окне Граф плана доступа выберите **Вид→Параметры**. Будет показана записная книжка Параметры графа плана доступа.
2. Для изменения цвета фона перейдите на вкладку Граф.
3. Изменить цвет различных операторов можно на вкладках Основные, Дополнительные, Обновленные, Прочие.
4. Для изменения цвета узла таблицы, индекса или функции таблицы выберите вкладку Операнд.
5. На вкладке Оператор можно задать тип информации, которая выдается в узлах операторов (тип затрат или "количества элементов", представляющего собой оценку числа строк, возвращенных на данный момент).
6. На вкладке Операнд можно указать, следует ли показывать в узлах таблиц имена схем или ID пользователей.
7. На вкладке Узел можно определить, в каком виде будут показаны узлы: в двумерном или трехмерном.
8. Для того чтобы обновить граф с учетом выбранных опций и сохранить их, нажмите кнопку **Применить**.

---

## Что дальше

При работе с одностолбчатой базой данных перейдите к разделу “Урок 3. Улучшение плана доступа для одностолбчатой базы данных” на стр. 13 с описанием настройки плана доступа.

При работе с одностолбчатой базой данных перейдите к разделу “Выполнение запроса без индексов и статистики” на стр. 14 с описанием настройки плана доступа.

---

## Урок 3. Улучшение плана доступа для однораздельной базы данных

В данном разделе описано влияние различных настроек на план доступа и связанные окна простого запроса. Последовательность примеров и рисунков иллюстрирует, как с помощью команды **runstats** и добавления подходящих индексов можно снизить полные затраты на план доступа даже для простого запроса.

После того как вы освоите основные приемы работы с Наглядным объяснением, вы сможете применять и другие способы настройки запросов.

---

### Работа с графами плана доступа

Настройка производительности базы данных будет показана на четырех примерах, использующих предопределенные снимки объяснения.

Запросы, связанные со снимками объяснения, пронумерованы цифрами от 1 до 4. Во всех запросах используется один и тот же оператор SQL (описанный в Уроке 1):

```
SELECT S.ID,S.NAME,O.DEPTNAME,SALARY+COMM
FROM ORG O, STAFF S
WHERE
  O.DEPTNUMB = S.DEPT AND
  S.JOB <> 'Mgr' AND
  S.SALARY+S.COMM > ALL( SELECT ST.SALARY*.9
                        FROM STAFF ST
                        WHERE ST.JOB='Mgr' )
ORDER BY S.NAME
```

Однако, в каждом следующем случае используется больше приемов настройки, чем в предыдущем. В частности, к запросу 1 не применена настройка производительности, в то время как к запросу 4 применена наиболее полная настройка. Изменение запросов описано ниже:

#### Запрос 1

Выполнение запроса без индексов и статистики

#### Запрос 2

Сбор текущей статистики для таблиц и индексов в запросе

#### Запрос 3

Создание индексов в столбцах, используемых для объединения таблиц в запросе

## Запрос 4

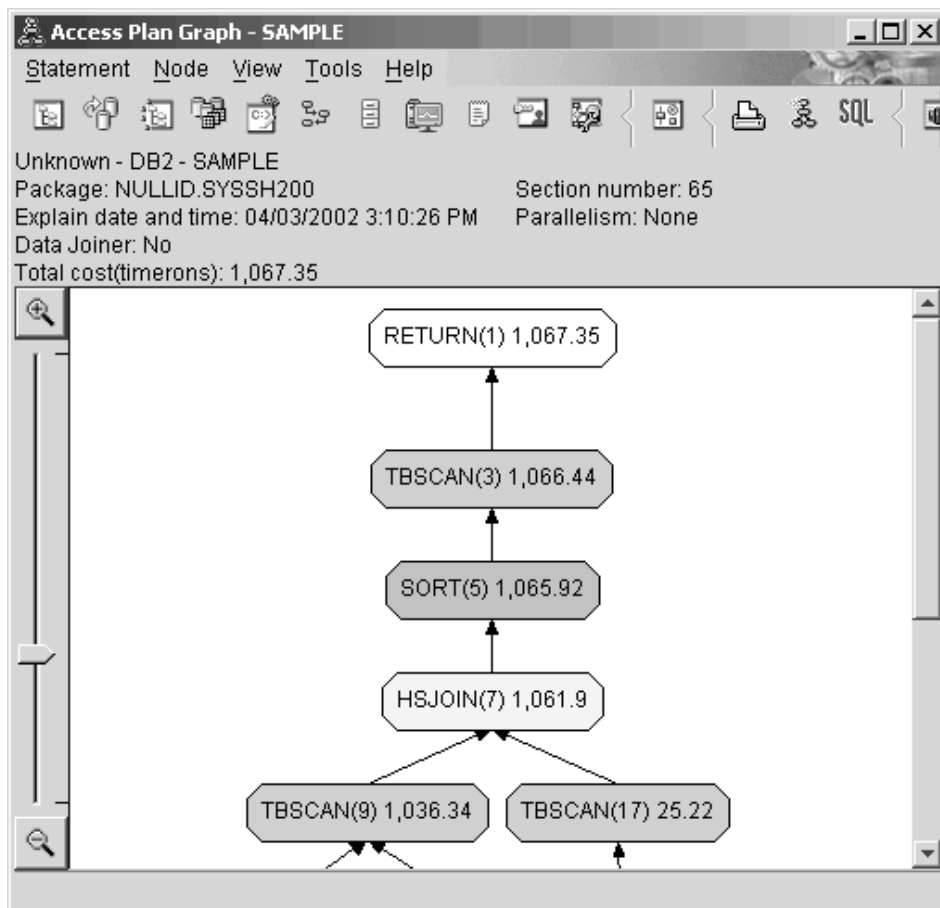
Создание дополнительных индексов в столбцах таблицы

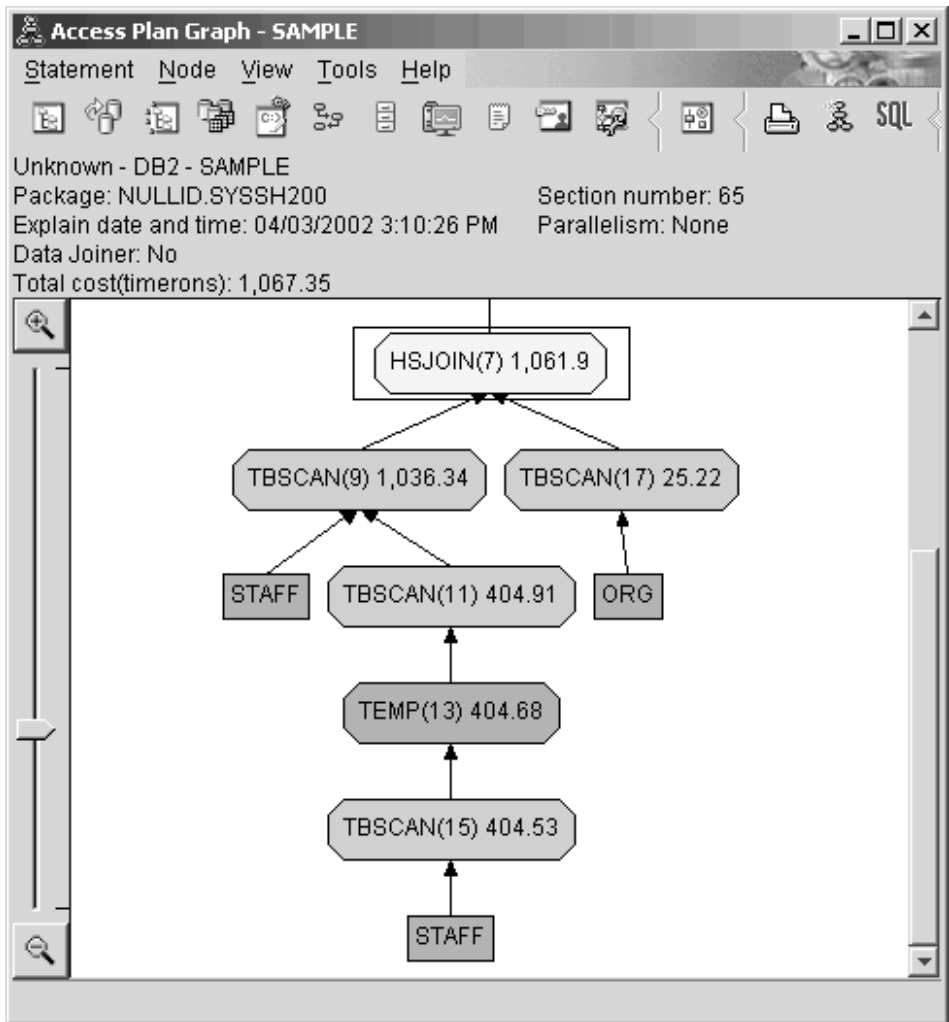
### Выполнение запроса без индексов и статистики

В этом примере план доступа создается для запроса SQL без индексов и статистики.

Для просмотра графа плана доступа для запроса (Запрос 1):

1. В Центре управления найдите в дереве объектов базу данных SAMPLE.
2. Щелкните на значке базы данных правой кнопкой и выберите в меню пункт **Показать хронологию объясненных операторов**. Откроется окно Хронология объясненных операторов.
3. Дважды щелкните на записи, для которой в столбце Номер запроса указано 1 (возможно, для просмотра столбца **Номер запроса** потребуется прокрутить окно вправо). Будет показано окно Граф плана доступа для оператора.





Для того чтобы понять, как улучшить запрос, ответьте на несколько вопросов.

1. Существует ли для каждой таблицы запроса текущая статистика?

Для того чтобы проверить это, дважды щелкните на каждом узле таблицы, показанном на графе плана доступа. В окне Статистика таблицы строка **STATS\_TIME** в столбце **Объяснен** будет содержать слова "Статистика не обновлялась", если при создании снимка не была собрана статистика.

Если текущая статистика отсутствует, оптимизатор использует статистику по умолчанию, которая может отличаться от реальной. Статистика по умолчанию обозначается словом "по умолчанию" в колонке **Объясненные** окна Table Statistics.

В соответствии с информацией в окне Table Statistics для таблицы ORG, оптимизатор использовал статистику по умолчанию (на что указывает запись рядом с объясненными значениями), так как реальная статистика при создании снимка была недоступна (на это указывает значение в строке **STATS\_TIME**).

Statistics	Explained	Current
CREATE_TIME	04/03/2002 3:05:03 PM	04/03/2002 3:05:03 PM
STATS_TIME	Statistics not updated	04/03/2002 4:25:19 PM
CARD	55(default)	8
NPAGES	1(default)	1
FPAGES	1(default)	1
COLCOUNT	5(default)	5
OVERFLOW	0(default)	0
TABLESPACE	USERSPACE1	USERSPACE1
INDEX_TABLESPACE		
LONG_TABLESPACE		
VOLATILE	No(default)	No

2. Применяет ли этот план доступа наиболее эффективные методы доступа к данным?

Этот план доступа содержит операции просмотра таблиц, а не индексов. Такие операции показаны в виде восьмиугольников и помечены TBSCAN. Операции просмотра индексов будут показаны в виде ромбов и помечены IXSCAN. Использование индекса - наиболее эффективный способ получения из таблицы небольших объемов данных.

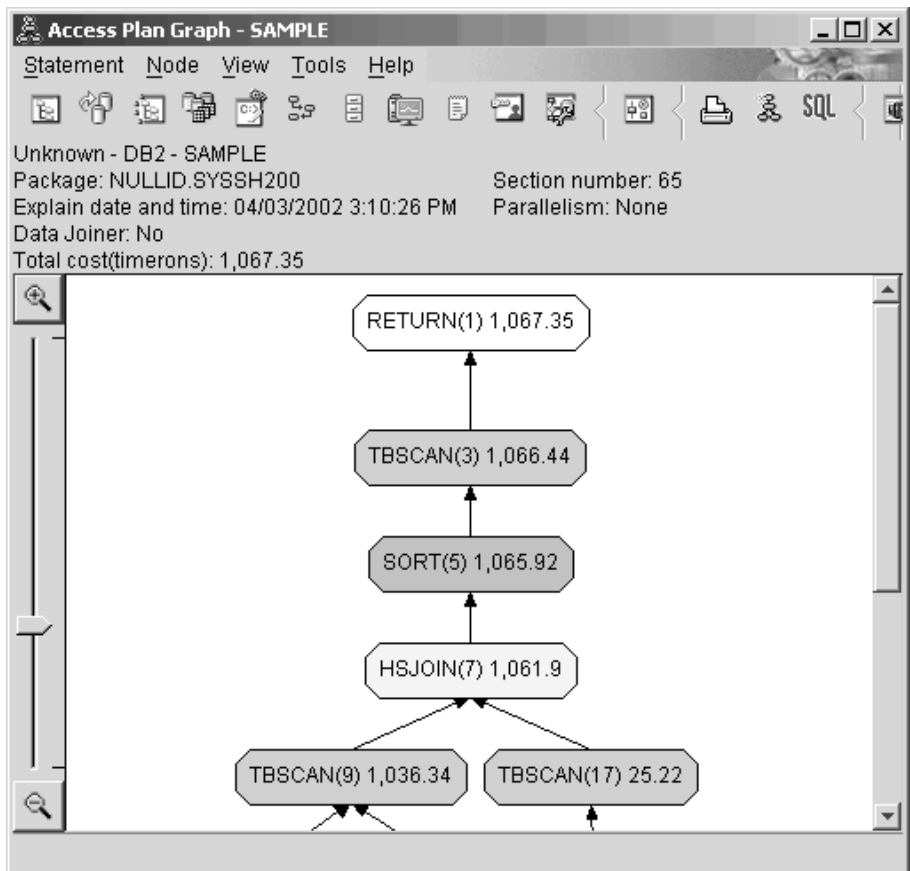
3. Насколько эффективен этот план доступа?

Эффективность плана доступа можно оценить только на основании реальной статистики. Так как оптимизатор использует статистику по умолчанию, определить, насколько эффективен данный план, невозможно.

В общем случае вы должны записать оценку полных затрат на план доступа для дальнейшего сравнения с откорректированными планами доступа. Для каждого узла указываются совокупные затраты, начиная с первых шагов запроса и до данного узла включительно.

В окне Граф плана доступа полная стоимость, приблизительно равная 1,067 единиц времени, показана для узла **RETURN (1)** в верхней части графа. В верхней части окна также показана оценка полных затрат.





#### 4. Что дальше?

В разделе Запрос 2 план доступа для простого запроса анализируется после выполнения команды **runstats**. Команда **runstats** предоставляет оптимизатору текущую статистику для всех таблиц, используемых в запросе.

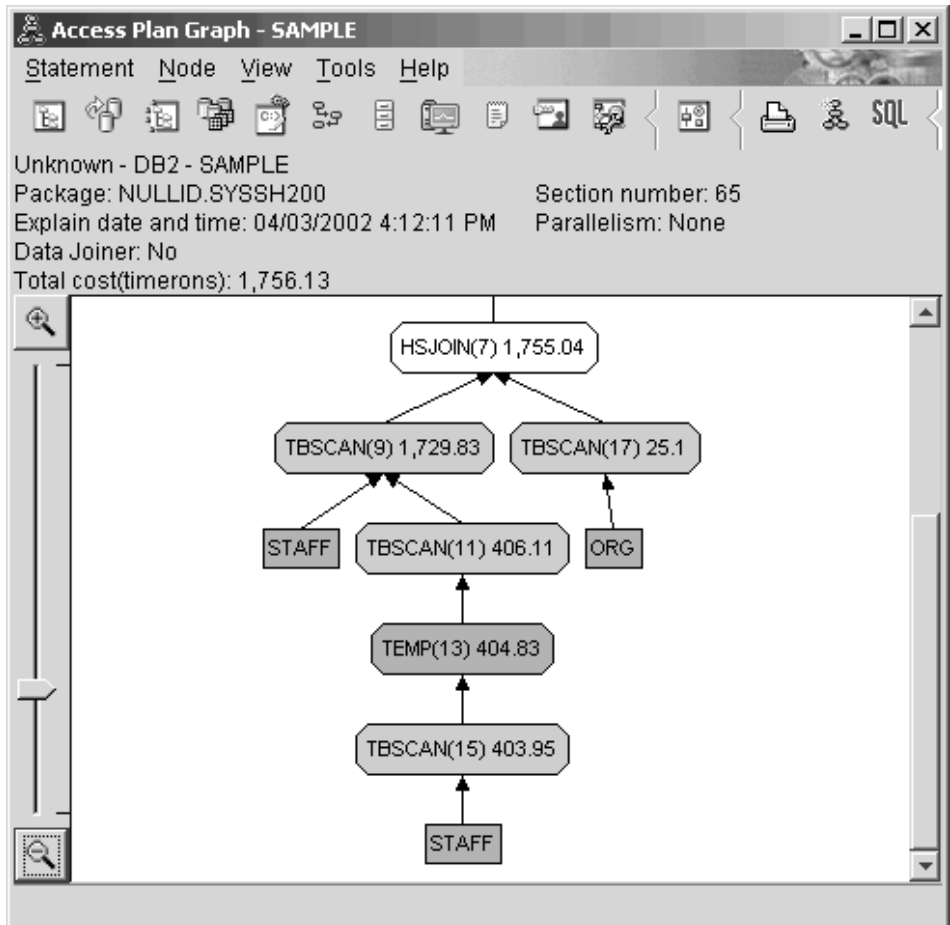
### Сбор текущей статистики для таблиц и индексов командой **runstats**

Данный пример основан на плане доступа, описанном в Запросе 1, с добавлением процедуры сбора статистики с помощью команды **runstats**.

Для сбора текущей статистики по таблицам и индексам настоятельно рекомендуется использовать команду **runstats**, особенно при интенсивном обновлении данных или при создании новых индексов уже после выполнения команды **runstats**. Это позволяет обеспечивать оптимизатор наиболее точной информацией, на основе которой он может выбрать оптимальный план доступа. Если текущая статистика недоступна, оптимизатор может использовать план доступа, основанный на статистике по умолчанию. Однако эта статистика не точна, поэтому такой план доступа наименее эффективен.

Не забывайте выполнять команду **runstats** *после* обновления таблиц; в противном случае оптимизатор может обнаружить, что таблица пуста, например, если в окне Сведения об операторе указано нулевое число элементов. В этом случае следует обновить таблицы, повторно выполнить команду **runstats** и еще раз создать поясняющие снимки для обновленных таблиц.

Для просмотра графа плана доступа для запроса (Запрос 2): Будет показано окно Граф плана доступа для данного выполнения оператора.



Для того чтобы понять, как улучшить запрос, ответьте на несколько вопросов.

1. Существует ли для каждой таблицы запроса текущая статистика?

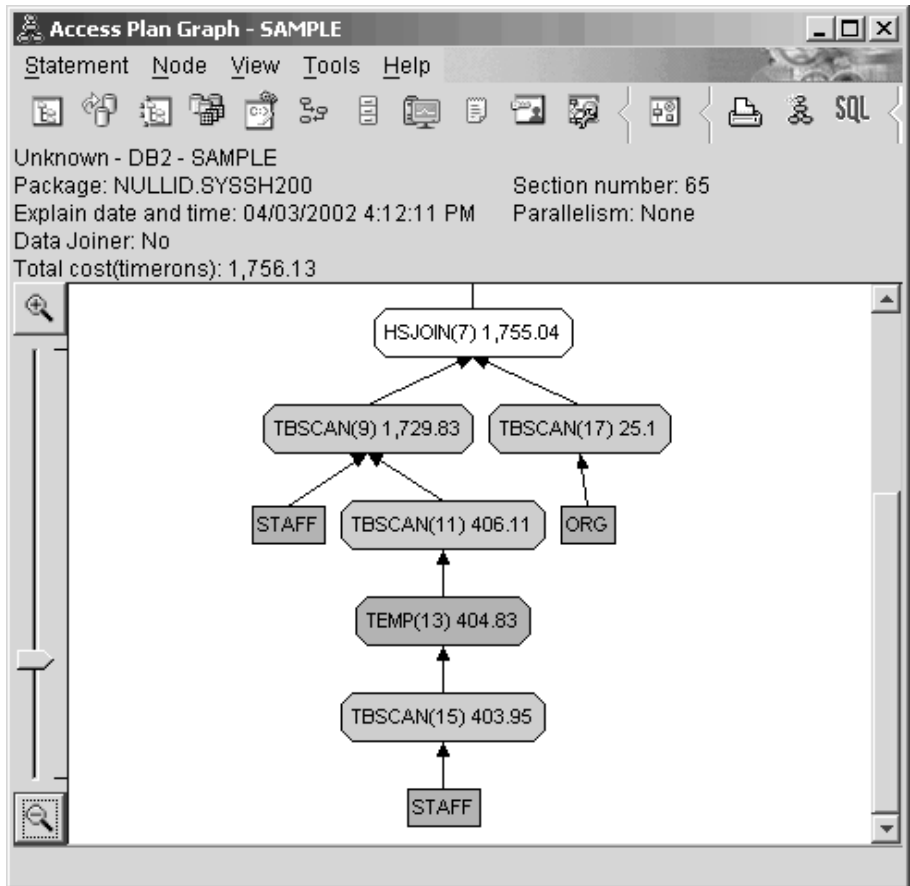
Информация в окне Table Statistics для таблицы ORG свидетельствует, что оптимизатор использовал реальную статистику (значение в поле **STATS\_TIME** представляет собой фактическое время сбора статистики).

Точность статистики зависит от того, насколько значительными были изменения содержимого таблиц после запуска команды **runstats**.

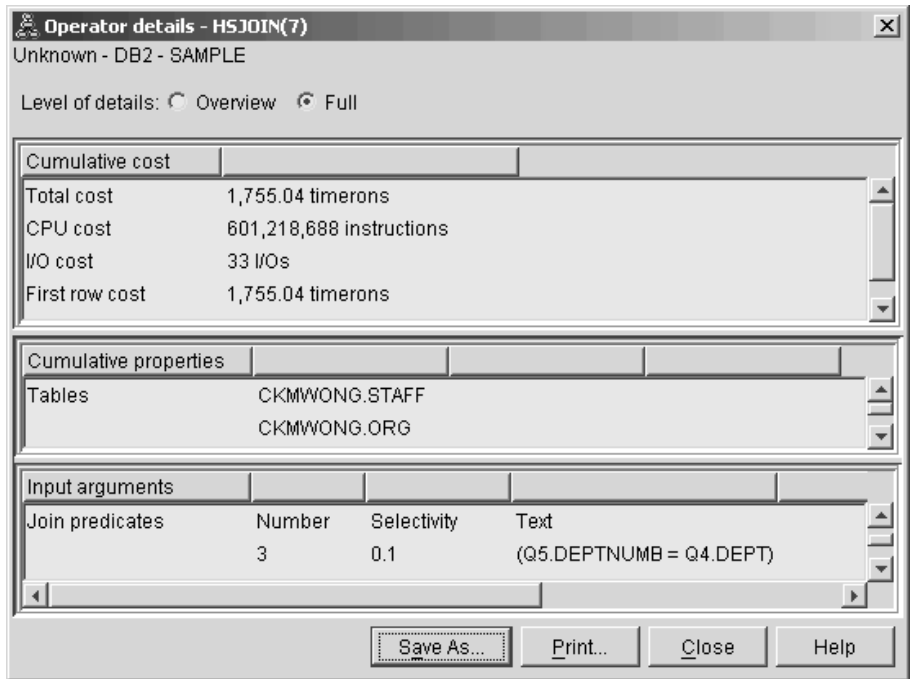
Statistics	Explained	Current
CREATE_TIME	04/03/2002 3:05:03 PM	04/03/2002 3:05:03 PM
STATS_TIME	04/03/2002 4:12:05 PM	04/03/2002 4:12:05 PM
CARD	8	8
NPAGES	1	1
FPAGES	1	1
COLCOUNT	5	5
OVERFLOW	0	0
TABLESPACE	USERSPACE1	USERSPACE1
INDEX_TABLESPACE		
LONG_TABLESPACE		
VOLATILE	No	No

2. Применяет ли этот план доступа наиболее эффективные методы доступа к данным?

Как и в Запросе 1, план доступа в Запросе 2 использует операторы просмотра таблиц (TBSCAN), а не индексов (IXSCAN). Даже несмотря на существование текущей статистики, просмотр индексов не применяется, так как для столбцов, использованных в запросе, нет индексов. Возможный способ улучшить запрос - передавать оптимизатору индексы столбцов, которые используются для объединения таблиц (т.е. столбцов, применяемых в предикатах объединения). В данном примере это первый оператор объединения результатов просмотра с помощью слияния: HSJOIN (7).



В окне Сведения об операторе для оператора HSJOIN (7) найдите раздел **Предикаты объединения** в поле **Входные параметры**. Столбцы, для которых выполняется данная операция объединения, перечислены в колонке **Текст**. В этом примере это столбцы DEPTNUMB и DEPT.



### 3. Насколько эффективен этот план доступа?

Для планов доступа, основанных на текущей, постоянно обновляемой статистике, оценка затрат (измеряемая в единицах времени) всегда реальна. Так как оценка затрат в Запросе 1 основывалась на статистике по умолчанию, мы не можем сравнивать оценки затрат двух графов планов доступа, чтобы определить, какой из них эффективнее. Сравнение этих оценок неправомерно. Для того чтобы оценить эффективность, необходимо сравнивать затраты на планы доступа, основанные только на реальной статистике.

### 4. Что дальше?

В разделе Запрос 3 рассматриваются эффекты от добавления индексов для столбцов DEPTNUMB и DEPT. Добавление индексов для столбцов, указанных в предикатах объединения, увеличивает производительность.

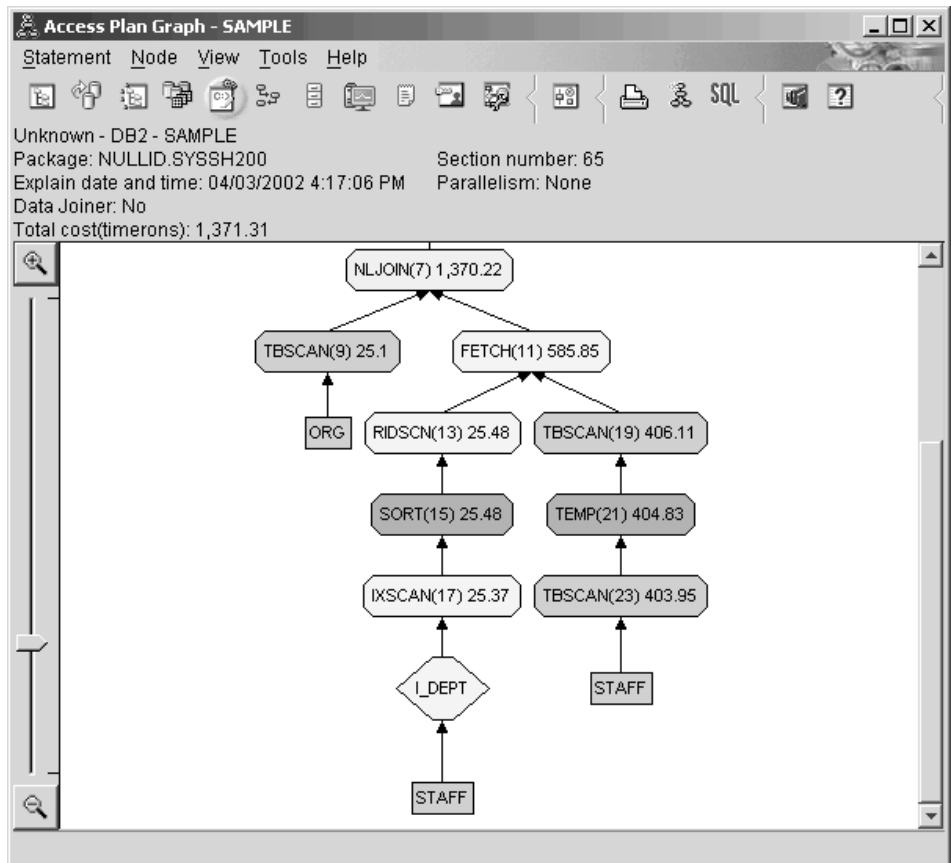
## Создание индексов в столбцах, используемых для объединения таблиц в запросе

Этот пример основан на плане доступа, описанном в Запросе 2, с добавлением процедуры создания индексов для столбца DEPT в таблице STAFF и столбца DEPTNUMB в таблице ORG.

**Примечание:** В версии 8 рекомендованные индексы могут быть созданы с помощью мастера Производительность нагрузки.

Для просмотра графа плана доступа для запроса (Запрос 3): Будет показано окно Граф плана доступа для данного выполнения оператора.

**Примечание:** Несмотря на то, что для столбца DEPTNUM был создан индекс, этот индекс не был использован оптимизатором.

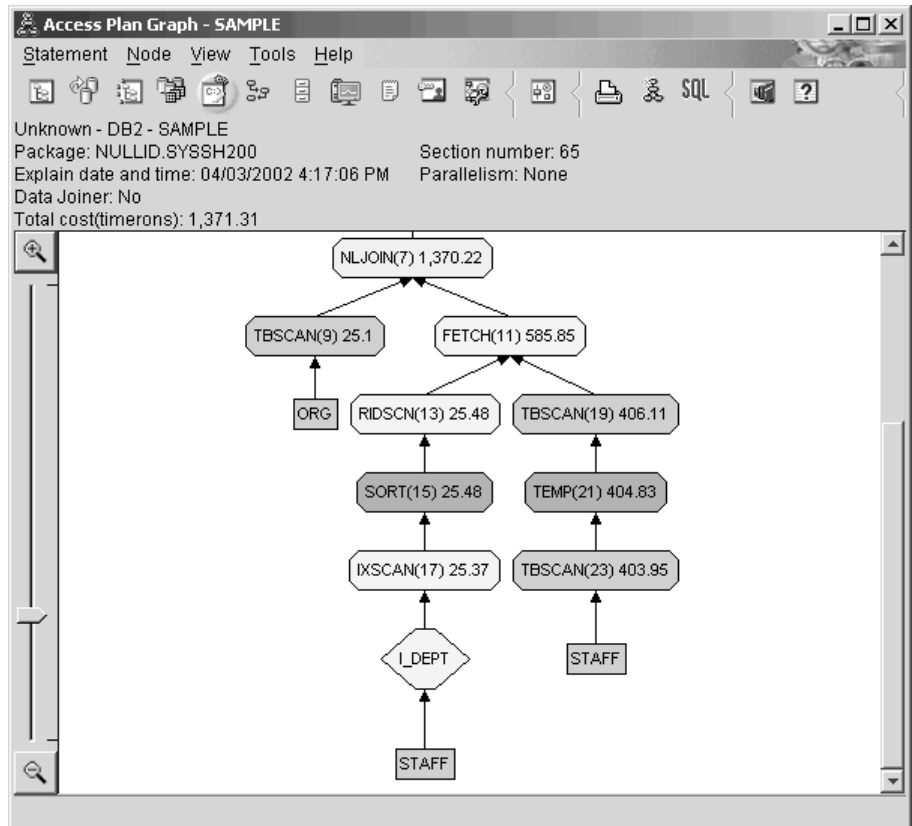


Для того чтобы понять, как улучшить запрос, ответьте на несколько вопросов.

1. Что изменилось в плане доступа с индексами?

Вместо оператора объединения результатов просмотра с помощью слияния HSJOIN (7), использовавшегося в Запросе 2, применяется оператор объединения с вложенным циклом (NLJOIN (7)). Такая замена приводит к снижению затрат, так как этот тип объединения не требует сортировки или создания временных таблиц.

Над таблицей STAFF появился дополнительный ромб узла **I\_DEPT**. Этот узел, представляющий собой индекс, созданный для DEPT, показывает, что оптимизатор выбирал считываемые строки с помощью сканирования индекса, а не таблицы.



В этой части графа плана доступа указано, что для столбца DEPT был создан новый индекс (**I\_DEPT**), а для доступа к таблице STAFF использовался оператор IXSCAN (17). В Запросе 2 для доступа к таблице STAFF применялся просмотр таблицы.

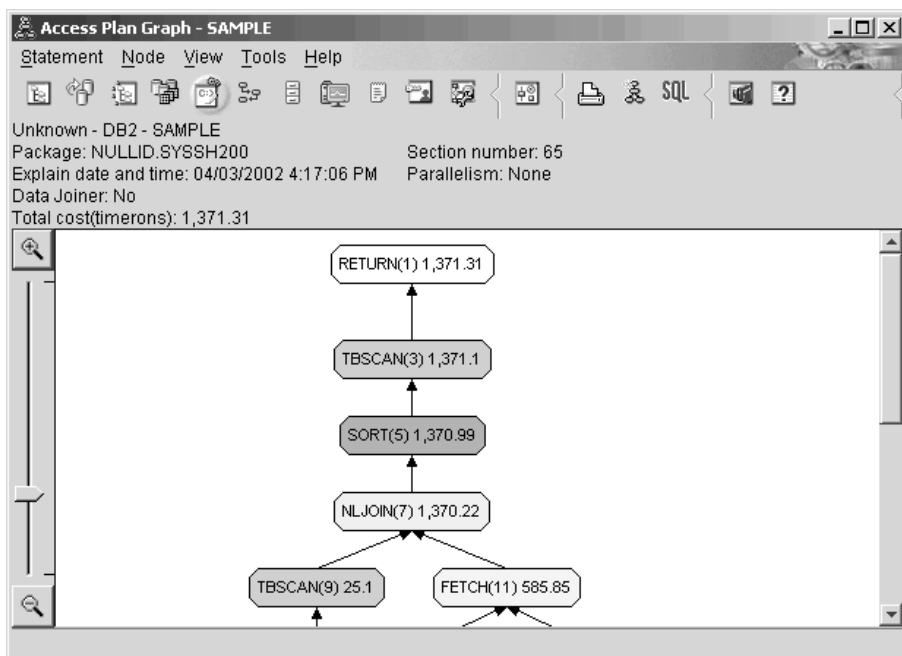
2. Применяет ли этот план доступа наиболее эффективные методы доступа к данным?

В результате добавления индекса для доступа к таблице ORG использовался узел IXSCAN (IXSCAN (17)). В Запросе 2 индекса не было, следовательно, во втором примере для доступа к таблице применялся просмотр таблицы.

Узел FETCH (FETCH (11)) показывает, что, помимо сканирования индекса для считывания столбца DEPT, оптимизатор считывает дополнительные столбцы из таблицы STAFF, используя индекс как указатель. В этом случае

затраты на комбинацию операций сканирования индекса и выборки оказываются меньше, чем на сканирование всей таблицы, применяемое в предыдущих планах доступа.

**Примечание:** Узел для таблицы STAFF указан дважды, чтобы показать его связь и с индексом для столбца DEPT, и с операцией FETCH.



План доступа для данного запроса отражает эффект создания индексов для столбцов, включенных в предикаты объединения. Индексы могут также улучшить применимость локальных предикатов. Для того чтобы понять, как добавление индексов для столбцов, на которые есть ссылки в локальных предикатах, может влиять на план доступа, рассмотрим локальные предикаты для каждой таблицы в данном запросе.

В окне Operator Details для оператора FETCH (11) посмотрите на столбцы **Совокупные свойства**. Как показано в разделе Предикаты, в предикате для данной операции выборки использовался столбец JOB.

**Примечание:** Избирательность этого предиката равна 0,69. Это означает, что с помощью этого предиката для дальнейшей обработки будут отобраны только 69% строк.



**Operator details - FETCH(11)** [X]

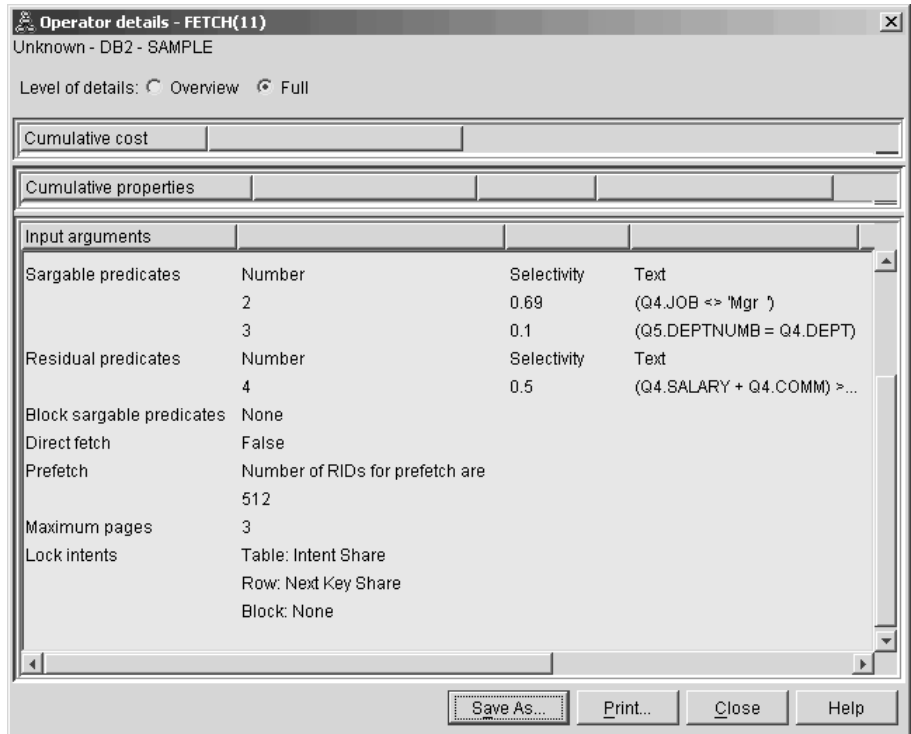
Unknown - DB2 - SAMPLE

Level of details:  Overview  Full

Cumulative cost	
Total cost	585.85 timerons
CPU cost	63,529,024 instructions
I/O cost	32.97 I/Os
First row cost	458.9 timerons

Cumulative properties			
Tables	CKMWONG.STAFF		
Columns	CKMWONG.STAFF.NAME CKMWONG.STAFF.ID CKMWONG.STAFF.COMM CKMWONG.STAFF.SALARY		
Order columns	None		
Predicates	Number	Selectivity	Text
	2	0.69	(Q4.JOB <=> 'Mgr')
	3	0.1	(Q5.DEPTNUMB = Q4.DEPT)
	4	0.5	(Q4.SALARY + Q4.COMM) > ...
Cardinality	37.2		
Total buffer pool pages used	23.39		

Save As... Print... Close Help



В окне Сведения об операторе для оператора FETCH (11) показаны столбцы, используемые в этой операции. Видно, что DEPTNAME упомянут в первой строке **Полученные столбцы** в поле **Входные параметры**.

3. Насколько эффективен этот план доступа?

Данный план доступа более эффективен с точки зрения затрат, чем предыдущий. Совокупные затраты в Запросе 3 уменьшены примерно до 959 единиц времени по сравнению с 1755 единицами в Запросе 2.

Вместе с тем, в плане доступа для Запроса 3 по отношению к таблице STAFF применяется как просмотр индекса IXSCAN (17), так и операция выборки FETCH (11). Поскольку величина затрат на просмотр в сочетании с операцией выборки меньше, чем затраты на просмотр всей таблицы, это означает, что для каждой считываемой строки один раз происходит обращение к таблице и один раз - к индексу. Попытаемся устранить это двойное обращение к таблице STAFF.

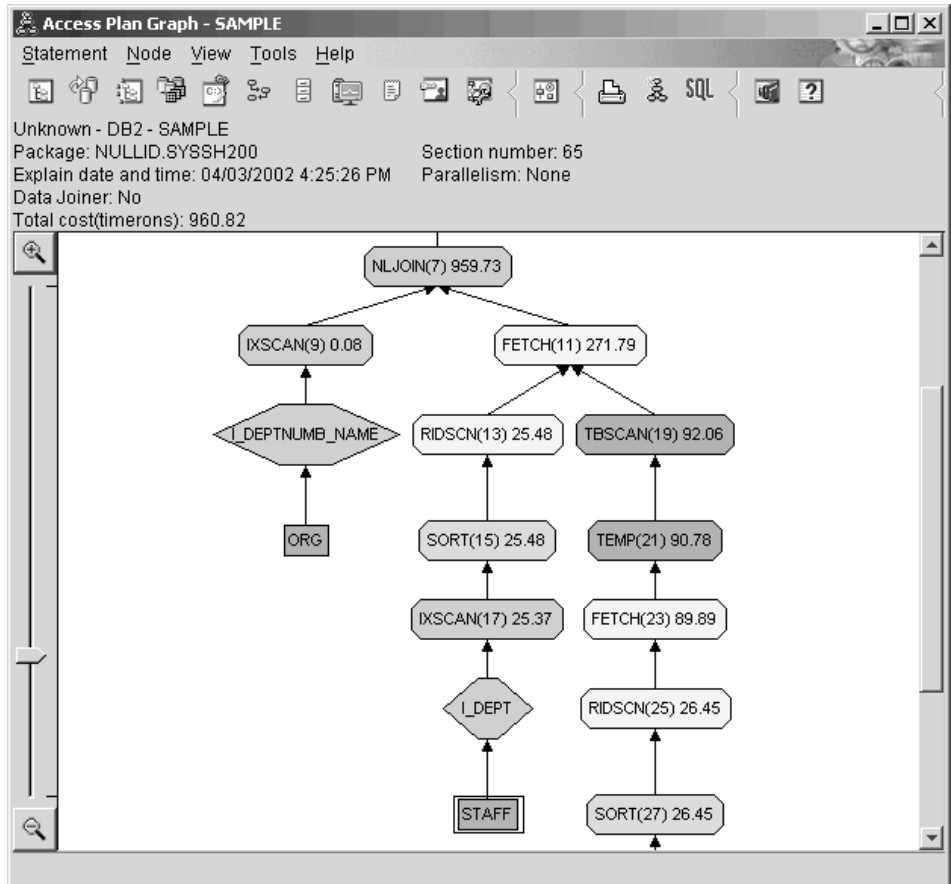
4. Что дальше?

В Запросе 4 выборка и просмотр индекса заменяются просмотром индекса без выборки. Для уменьшения стоимости выполнения запроса можно попробовать создать дополнительные индексы.

## Создание дополнительных индексов в столбцах таблицы

Данный пример основан на плане доступа, описанном в Запросе 3, с добавлением процедуры создания индексов для столбца JOB в таблице STAFF и добавлением столбца DEPTNUMB к существующему индексу в таблице ORG. (Добавление индексов может привести к дополнительному доступу)

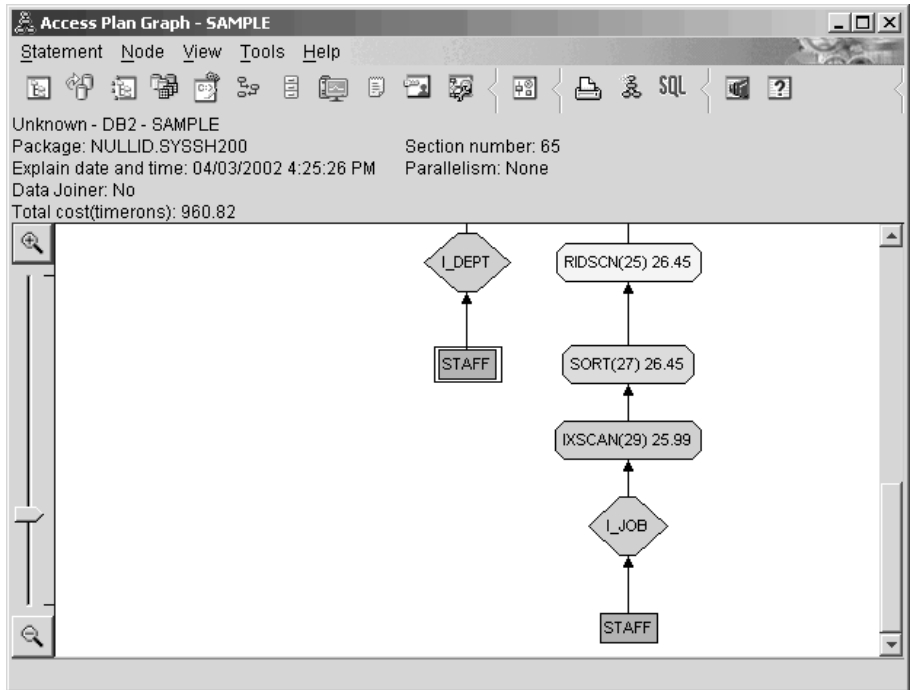
Для просмотра графа плана доступа для запроса (Запрос 4): Будет показано окно Граф плана доступа для данного выполнения оператора.



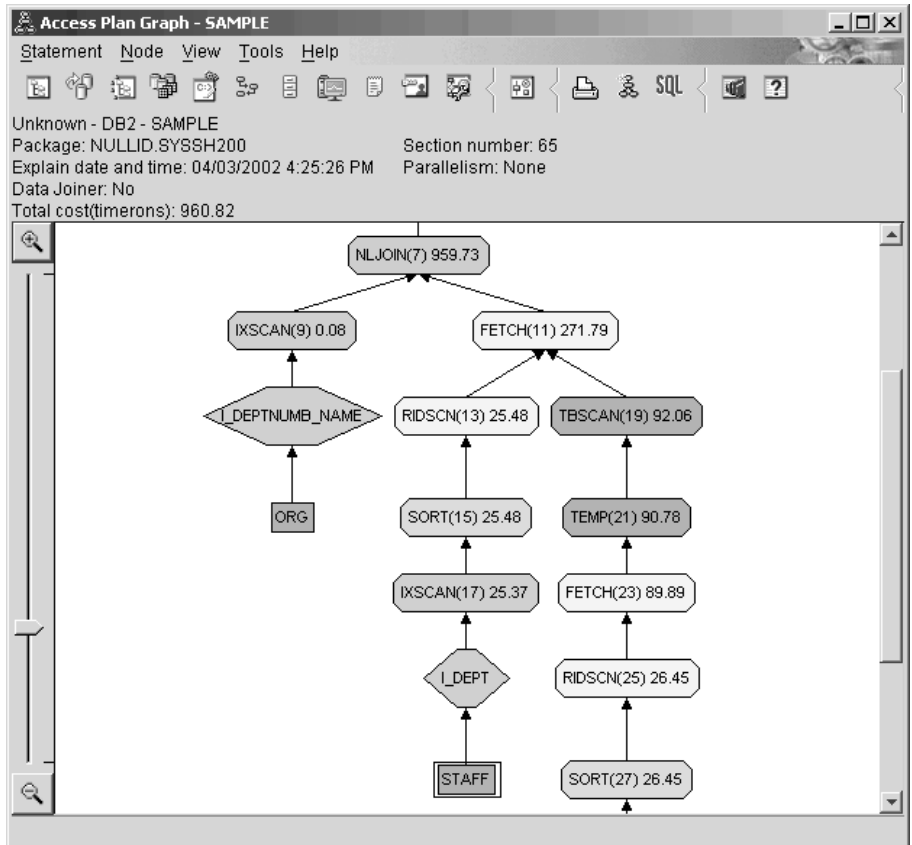
Для того чтобы понять, как улучшить запрос, ответьте на несколько вопросов.

1. Что изменилось в плане доступа в результате создания дополнительных индексов?

Оптимизатор использовал индекс, созданный для столбца JOB в таблице STAFF (показанный в виде ромба с меткой I\_JOB), для дальнейшего улучшения плана доступа.



Обратите внимание на среднюю часть графа плана доступа: для таблицы ORG вместо операторов сканирования индекса и выборки появился оператор сканирования индекса IXSCAN (9). Добавление столбца DEPTNAME к индексу в таблице ORG позволило оптимизатору исключить лишнюю операцию доступа (выборку).



2. Насколько эффективен этот план доступа?

Данный план доступа более эффективен с точки зрения затрат, чем предыдущий. Совокупные затраты в Запросе 4 уменьшены примерно до 959 единиц времени по сравнению с 1370 единицами в Запросе 3.

## Что дальше

Дополнительная информация об увеличении производительности приведена в публикации *Administration Guide*. Для оценки влияния действий по оптимизации на план доступа и производительность пользуйтесь Наглядным объяснением.



---

## Урок 4. Улучшение плана доступа для многораздельной базы данных

В данном разделе описано влияние различных настроек на план доступа и связанные окна простого запроса. Последовательность примеров и рисунков иллюстрирует, как с помощью команды **runstats** и добавления подходящих индексов можно снизить полные затраты на план доступа даже для простого запроса.

После того как вы освоите основные приемы работы с Наглядным объяснением, вы сможете применять и другие способы настройки запросов.

---

### Работа с графами плана доступа

Настройка производительности базы данных будет показана на четырех примерах, использующих predetermined снимки объяснения.

Запросы, связанные со снимками объяснения, пронумерованы цифрами от 1 до 4. Во всех запросах используется один и тот же оператор SQL (описанный в Уроке 1):

```
SELECT S.ID,S.NAME,O.DEPTNAME,SALARY+COMM
FROM ORG O, STAFF S
WHERE
  O.DEPTNUMB = S.DEPT AND
  S.JOB <> 'Mgr' AND
  S.SALARY+S.COMM > ALL( SELECT ST.SALARY*.9
                        FROM STAFF ST
                        WHERE ST.JOB='Mgr' )
ORDER BY S.NAME
```

Однако, в каждом следующем случае используется больше приемов настройки, чем в предыдущем. В частности, к запросу 1 не применена настройка производительности, в то время как к запросу 4 применена наиболее полная настройка. Изменение запросов описано ниже:

#### Запрос 1

Выполнение запроса без индексов и статистики

#### Запрос 2

Сбор текущей статистики для таблиц и индексов в запросе

#### Запрос 3

Создание индексов в столбцах, используемых для объединения таблиц в запросе

## Запрос 4

Создание дополнительных индексов в столбцах таблицы

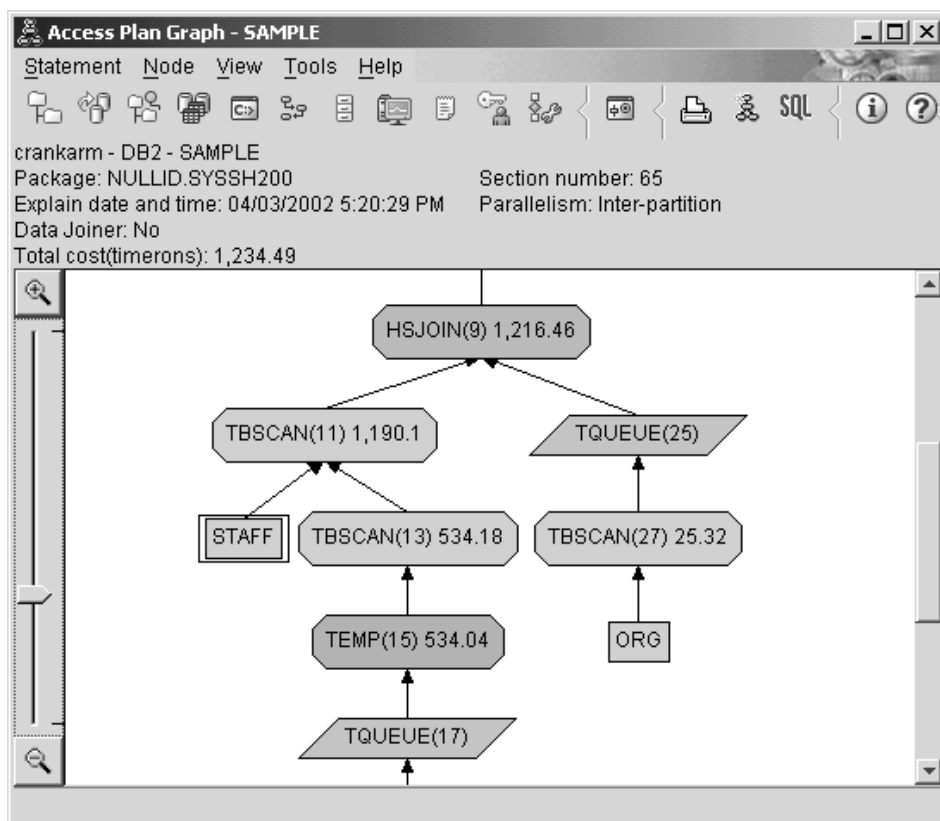
Они создавались на компьютере RS/6000 SP с семью физическими узлами и межраздельным параллелизмом.

### Выполнение запроса без индексов и статистики

В этом примере план доступа создается для запроса SQL без индексов и статистики.

Для просмотра графа плана доступа для запроса (Запрос 1):

1. В Центре управления найдите в дереве объектов базу данных SAMPLE.
2. Щелкните на значке базы данных правой кнопкой и выберите в меню пункт **Показать хронологию объясненных операторов**. Откроется окно Хронология объясненных операторов.
3. Дважды щелкните на записи, для которой в столбце Номер запроса указано 1 (возможно, для просмотра столбца **Номер запроса** потребуется прокрутить окно вправо). Будет показано окно Граф плана доступа для оператора.





Для того чтобы понять, как улучшить запрос, ответьте на несколько вопросов.

1. Существует ли для каждой таблицы запроса текущая статистика?

Для того чтобы проверить это, дважды щелкните на каждом узле таблицы, показанном на графе плана доступа. При этом для каждого узла откроется окно Table Statistics. Если с момента создания снимка сбор статистики не проводился, то в строке **STATS\_TIME** колонки **Объясненные** будет указано "Статистика не обновлялась".

Если текущая статистика отсутствует, оптимизатор использует статистику по умолчанию, которая может отличаться от реальной. Статистика по умолчанию обозначается словом "по умолчанию" в колонке **Объясненные** окна Table Statistics.

В соответствии с информацией в окне Table Statistics для таблицы ORG, оптимизатор использовал статистику по умолчанию (на что указывает запись рядом с объясненными значениями), так как реальная статистика при создании снимка была недоступна (на это указывает значение в строке **STATS\_TIME**).

Statistics	Explained	Current
CREATE_TIME	03/26/2002 1:35:42 PM	03/26/2002 1:35:42 PM
STATS_TIME	Statistics not updated	Statistics not updated
CARD	55(default)	-1
NPAGES	1(default)	-1
FPAGES	1(default)	-1
COLCOUNT	5(default)	5
OVERFLOW	0(default)	-1
TABLESPACE	USERSPACE1	USERSPACE1
INDEX_TABLESPACE		
LONG_TABLESPACE		
VOLATILE	No(default)	No

2. Применяет ли этот план доступа наиболее эффективные методы доступа к данным?

Этот план доступа содержит операторы просмотра таблиц, а не индексов. Такие операции показаны в виде восьмиугольников и помечены TBSCAN. Операции просмотра индексов будут показаны в виде ромбов и помечены

IXSCAN. Использование индекса - наиболее эффективный способ получения из таблицы небольших объемов данных.

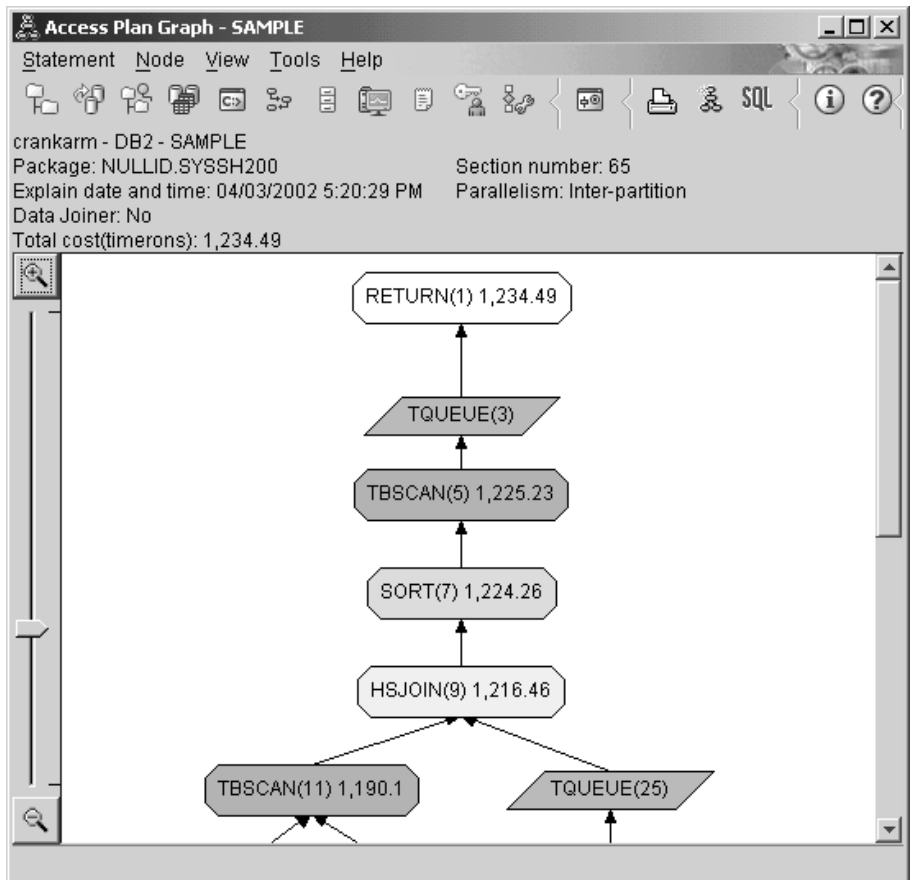
3. Насколько эффективен этот план доступа?

Эффективность плана доступа можно оценить только на основании реальной статистики. Так как оптимизатор использует статистику по умолчанию, определить, насколько эффективен данный план, невозможно.

В общем случае вы должны записать оценку полных затрат на план доступа для дальнейшего сравнения с откорректированными планами доступа. Для каждого узла указываются совокупные затраты, начиная с первых шагов запроса и до данного узла включительно.

**Примечание:** Для баз данных с несколькими разделами это совокупные затраты для узла, который использует наибольший объем ресурсов.

В окне Граф плана доступа в верхней строке графа **RETURN (1)** показаны полные затраты, примерно равные 1234 единицам времени. В верхней части окна также показана оценка полных затрат.



#### 4. Что дальше?

В разделе Запрос 2 план доступа для простого запроса анализируется после выполнения команды **runstats**. Команда **runstats** предоставляет оптимизатору текущую статистику для всех таблиц, используемых в запросе.

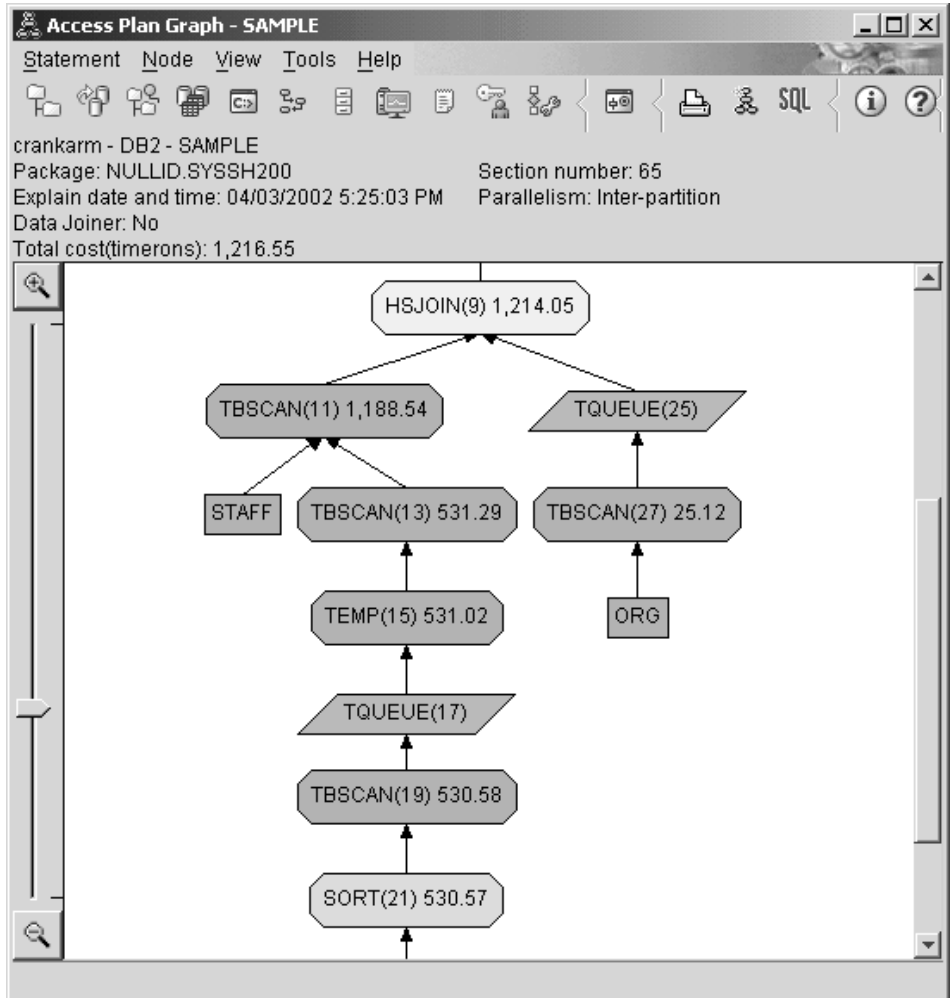
### Сбор текущей статистики для таблиц и индексов командой **runstats**

Данный пример основан на плане доступа, описанном в Запросе 1, с добавлением процедуры сбора статистики с помощью команды **runstats**.

Для сбора текущей статистики по таблицам и индексам настоятельно рекомендуется использовать команду **runstats**, особенно при интенсивном обновлении данных или при создании новых индексов уже после выполнения команды **runstats**. Это позволяет обеспечивать оптимизатор наиболее точной информацией, на основе которой он может выбрать оптимальный план доступа. Если текущая статистика недоступна, оптимизатор может использовать план доступа, основанный на статистике по умолчанию. Однако эта статистика не точна, поэтому такой план доступа наименее эффективен.

Не забывайте выполнять команду **runstats** *после* обновления таблиц; в противном случае оптимизатор может обнаружить, что таблица пуста, например, если в окне Сведения об операторе указано нулевое число элементов. В этом случае следует обновить таблицы, повторно выполнить команду **runstats** и еще раз создать поясняющие снимки для обновленных таблиц.

Для просмотра графа плана доступа для запроса (Запрос 2): Будет показано окно Граф плана доступа для данного выполнения оператора.



Для того чтобы понять, как улучшить запрос, ответьте на несколько вопросов.

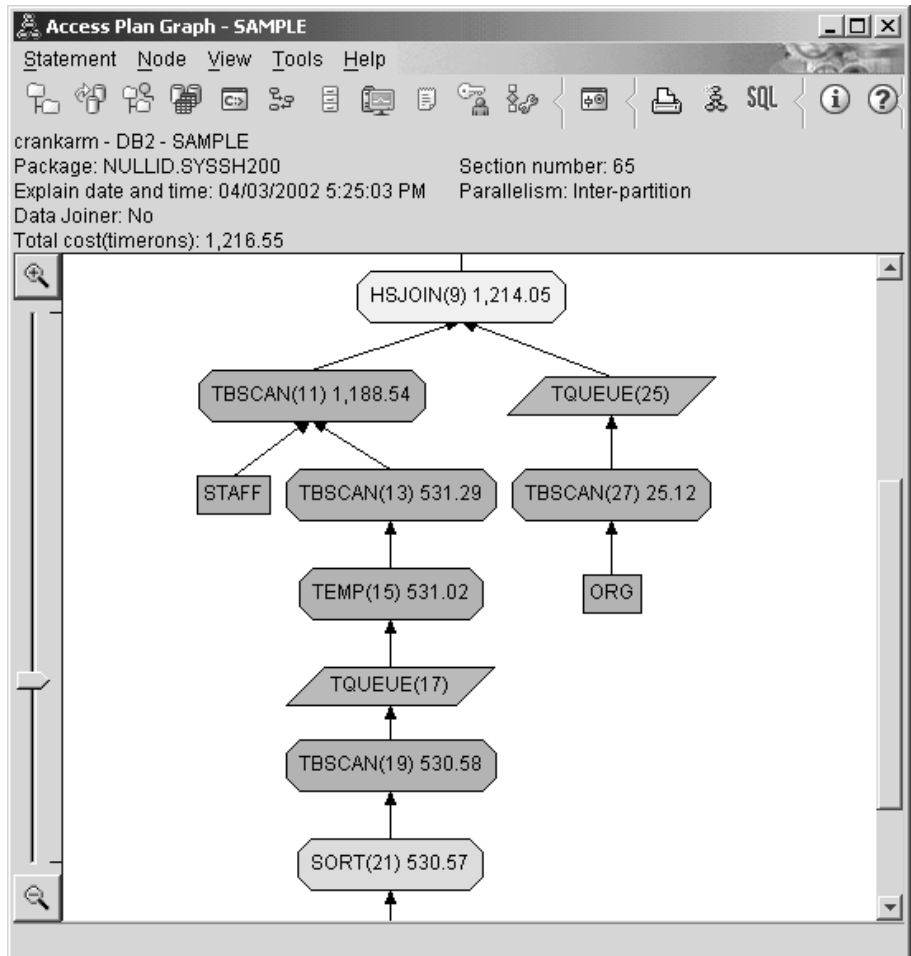
1. Существует ли для каждой таблицы запроса текущая статистика?

Информация в окне Table Statistics для таблицы ORG свидетельствует, что оптимизатор использовал реальную статистику (значение в поле **STATS\_TIME** представляет собой фактическое время сбора статистики). Точность статистики зависит от того, насколько значительными были изменения содержимого таблиц после запуска команды **runstats**.

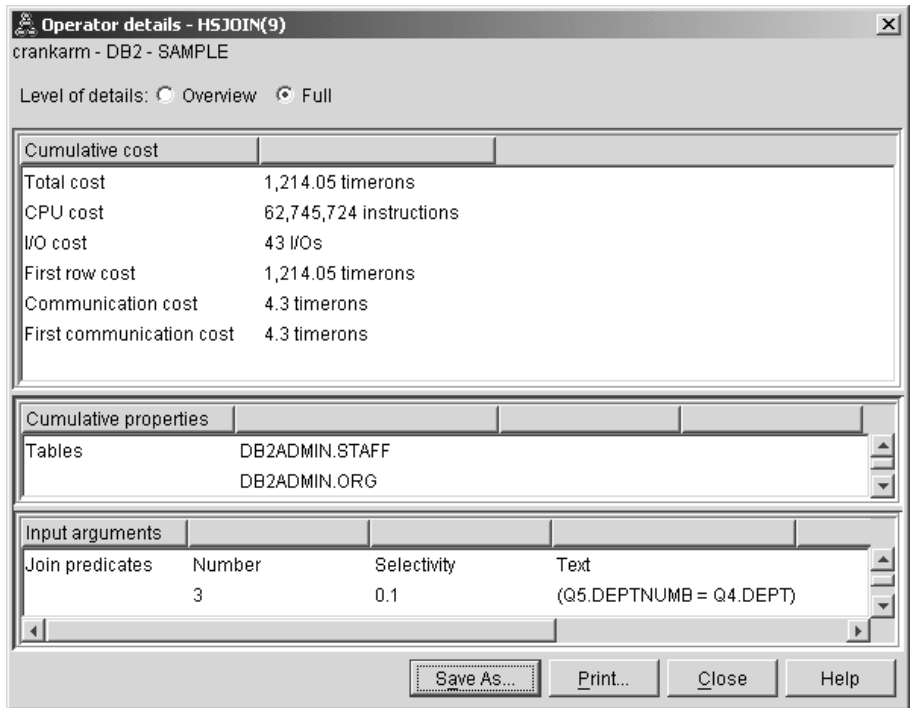
Statistics	Explained	Current
CREATE_TIME	03/26/2002 1:35:42 PM	03/26/2002 1:35:42 PM
STATS_TIME	04/03/2002 5:24:55 PM	04/03/2002 5:24:55 PM
CARD	4	8
NPAGES	1	2
FPAGES	1	2
COLCOUNT	5	5
OVERFLOW	0	0
TABLESPACE	USERSPACE1	USERSPACE1
INDEX_TABLESPACE		
LONG_TABLESPACE		
VOLATILE	No	No

2. Применяет ли этот план доступа наиболее эффективные методы доступа к данным?

Как и в Запросе 1, план доступа в Запросе 2 использует операторы просмотра таблиц (TBSCAN), а не индексов (IXSCAN). Даже несмотря на существование текущей статистики, просмотр индексов не применялся, так как для столбцов, использованных в запросе, нет индексов. Возможный способ улучшить запрос - передавать оптимизатору индексы столбцов, которые используются для объединения таблиц (т.е. столбцов, применяемых в предикатах объединения). В данном примере это первый оператор объединения результатов просмотра с помощью слияния: HSJOIN (9).



В окне Сведения об операторе для оператора HSJOIN (9) найдите раздел **Предикаты объединения** в поле **Входные параметры**. Столбцы, для которых выполняется данная операция объединения, перечислены в колонке **Текст**. В этом примере это столбцы DEPTNUMB и DEPT.



3. Насколько эффективен этот план доступа?

Для планов доступа, основанных на текущей, постоянно обновляемой статистике, оценка затрат (измеряемая в единицах времени) всегда реальна. Так как оценка затрат в Запросе 1 основывалась на статистике по умолчанию, мы не можем сравнивать оценки затрат двух графов планов доступа, чтобы определить, какой из них эффективнее. Сравнение этих оценок неправомерно. Для того чтобы оценить эффективность, необходимо сравнивать затраты на планы доступа, основанные только на реальной статистике.

4. Что дальше?

В разделе Запрос 3 рассматриваются эффекты от добавления индексов для столбцов DEPTNUMB и DEPT. Добавление индексов для столбцов, указанных в предикатах объединения, увеличивает производительность.

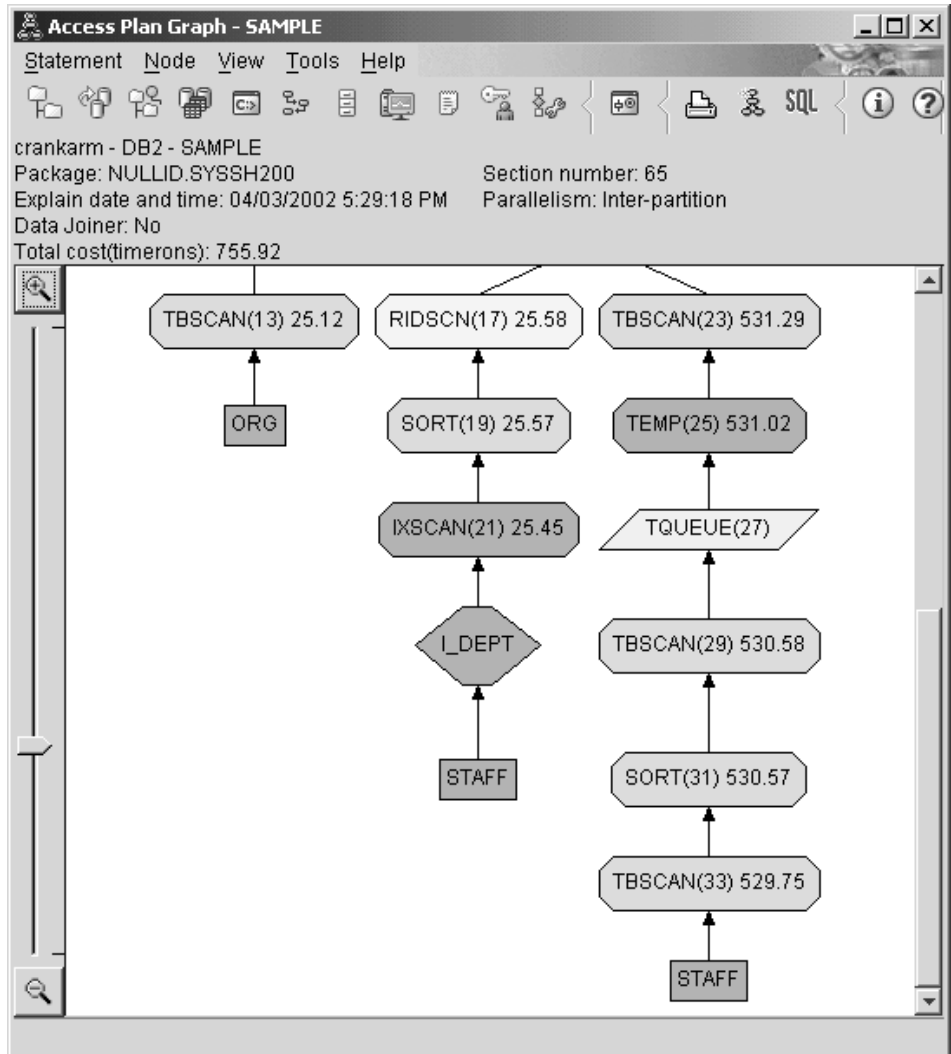
### Создание индексов в столбцах, используемых для объединения таблиц в запросе

Этот пример основан на плане доступа, описанном в Запросе 2, с добавлением процедуры создания индексов для столбца DEPT в таблице STAFF и столбца DEPTNUMB в таблице ORG.

**Примечание:** В версии 8 рекомендованные индексы могут быть созданы с помощью мастера Производительность нагрузки.

Для просмотра графа плана доступа для запроса (Запрос 3): Будет показано окно Граф плана доступа для данного выполнения оператора.

**Примечание:** Несмотря на то, что для столбца DEPTNUM был создан индекс, этот индекс не был использован оптимизатором.

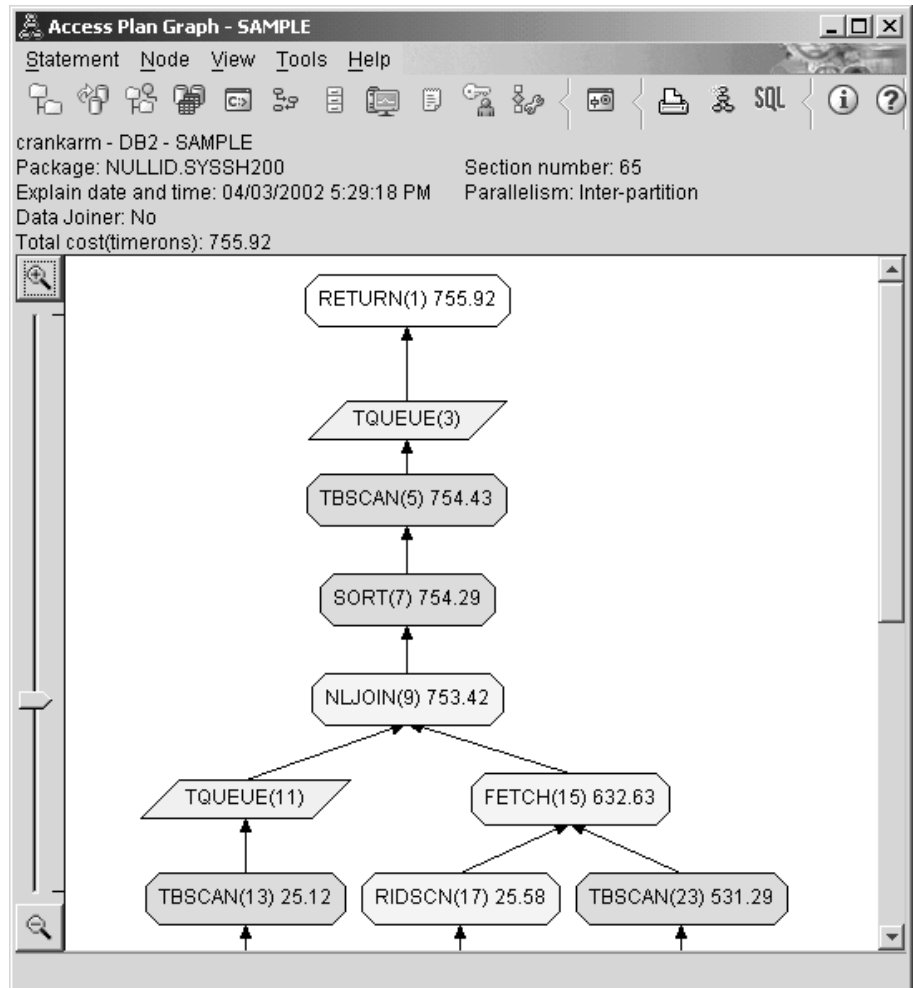


Для того чтобы понять, как улучшить запрос, ответьте на несколько вопросов.

1. Что изменилось в плане доступа с индексами?

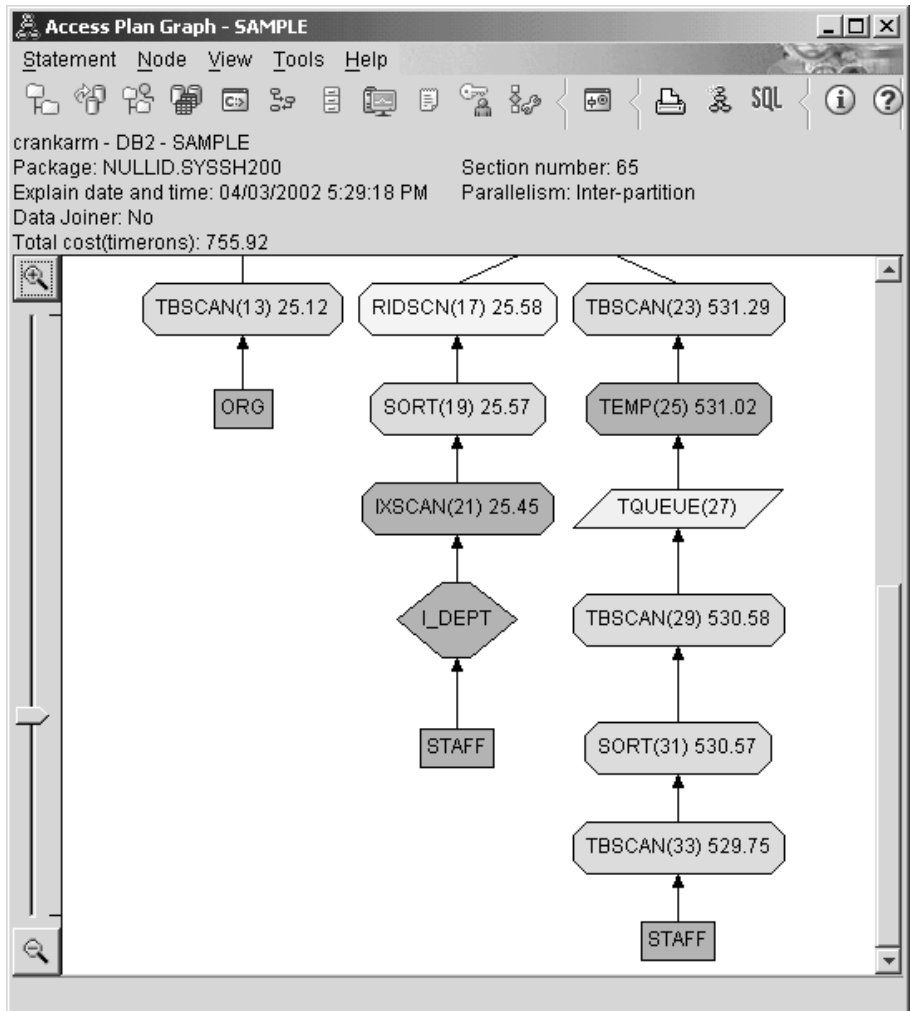


Над таблицей STAFF появился дополнительный ромб узла **I\_DEPT**. Этот узел, представляющий собой индекс, созданный для DEPT, показывает, что оптимизатор выбирал считываемые строки с помощью сканирования индекса, а не таблицы.

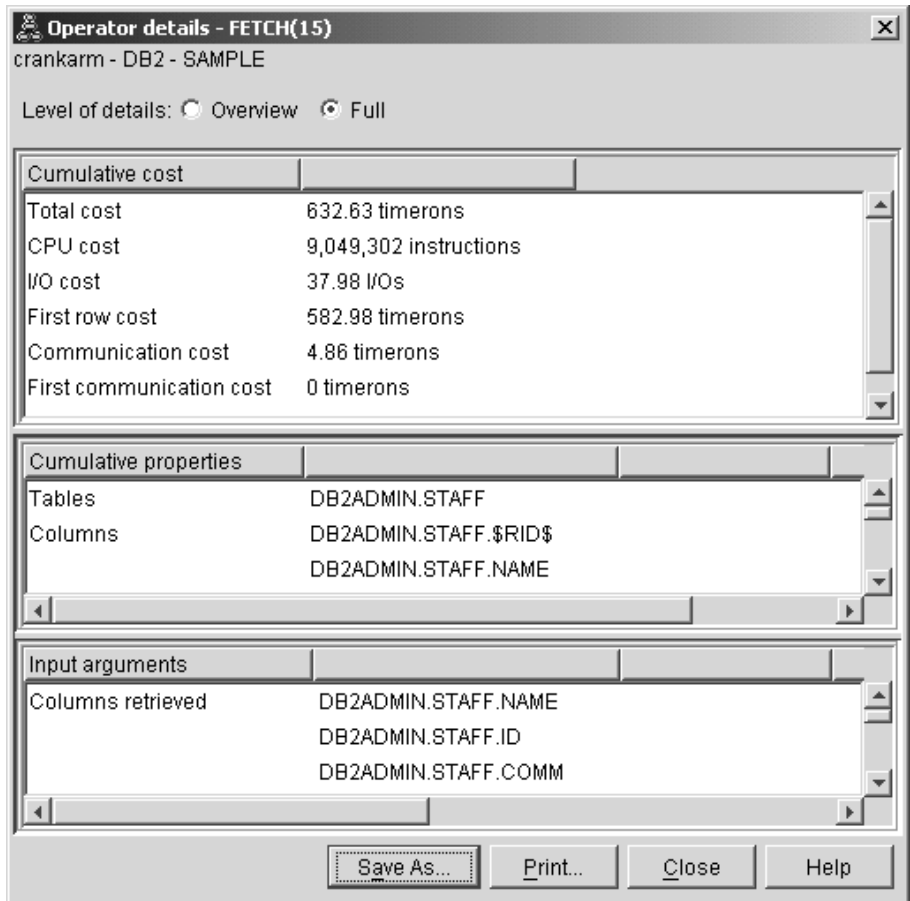


2. Применяет ли этот план доступа наиболее эффективные методы доступа к данным?

Этот план доступа отражает эффект создания индексов для столбца DEPTNUMB в таблице ORG (в результате чего появляются операторы FETCH (15) и IXSCAN (21)) и для столбца DEPT в таблице STAFF. В Запросе 2 индекса не было, следовательно, во втором примере для доступа к таблице применялось сканирование таблицы.



В окне Сведения об операторе для оператора FETCH (15) показаны столбцы, используемые в этой операции.



Затраты на комбинацию операций сканирования индекса и выборки оказываются меньше, чем на сканирование всей таблицы, применяемое в предыдущем плане доступа.

3. Насколько эффективен этот план доступа?

Данный план доступа более эффективен с точки зрения затрат, чем предыдущий. Совокупные затраты в Запросе 3 уменьшены примерно до 755 единиц времени по сравнению с 1214 единицами в Запросе 2.

4. Что дальше?

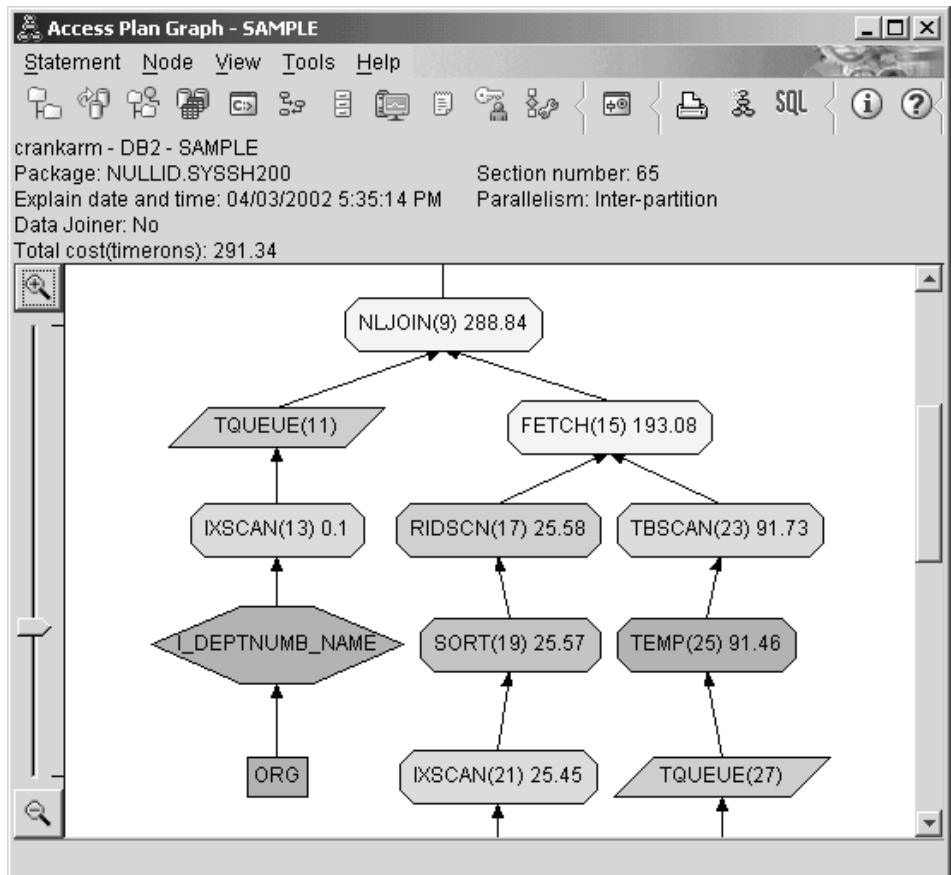
В Запросе 4 выборка и просмотр индекса заменяются просмотром индекса без выборки. Для уменьшения стоимости выполнения запроса можно попробовать создать дополнительные индексы.

### Создание дополнительных индексов в столбцах таблицы

Данный пример основан на плане доступа, описанном в Запросе 3, с добавлением процедуры создания индексов для столбца JOB в таблице STAFF и

добавлением столбца DEPTNUMB к существующему индексу в таблице ORG. (Добавление индексов может привести к дополнительному доступу)

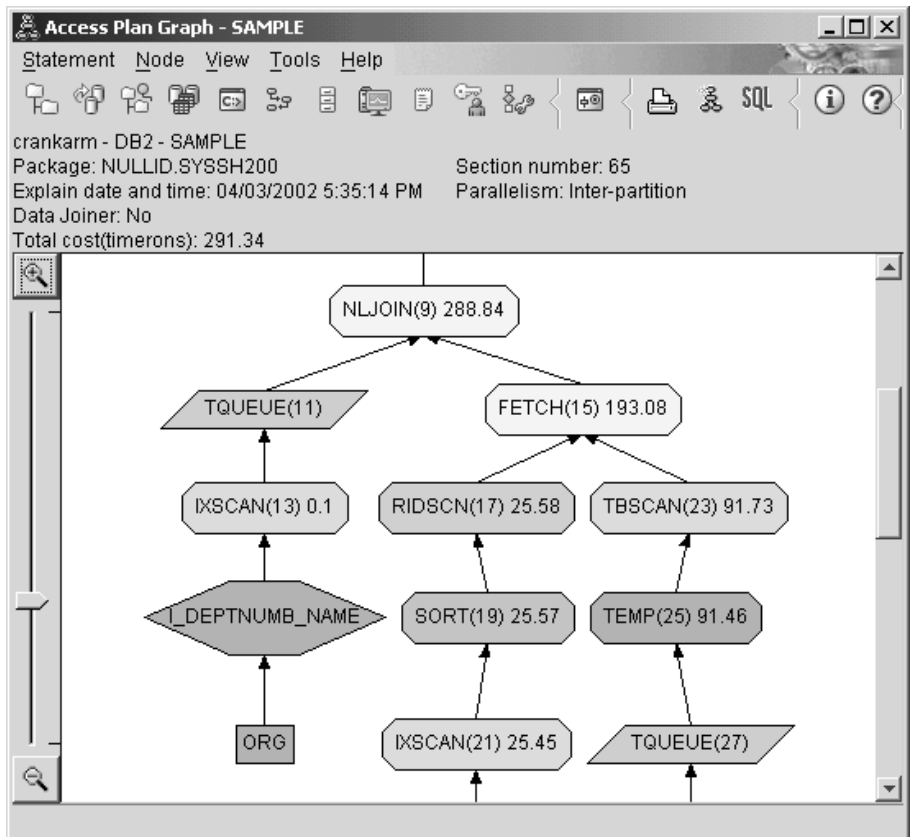
Для просмотра графа плана доступа для запроса (Запрос 4): Будет показано окно Граф плана доступа для данного выполнения оператора.



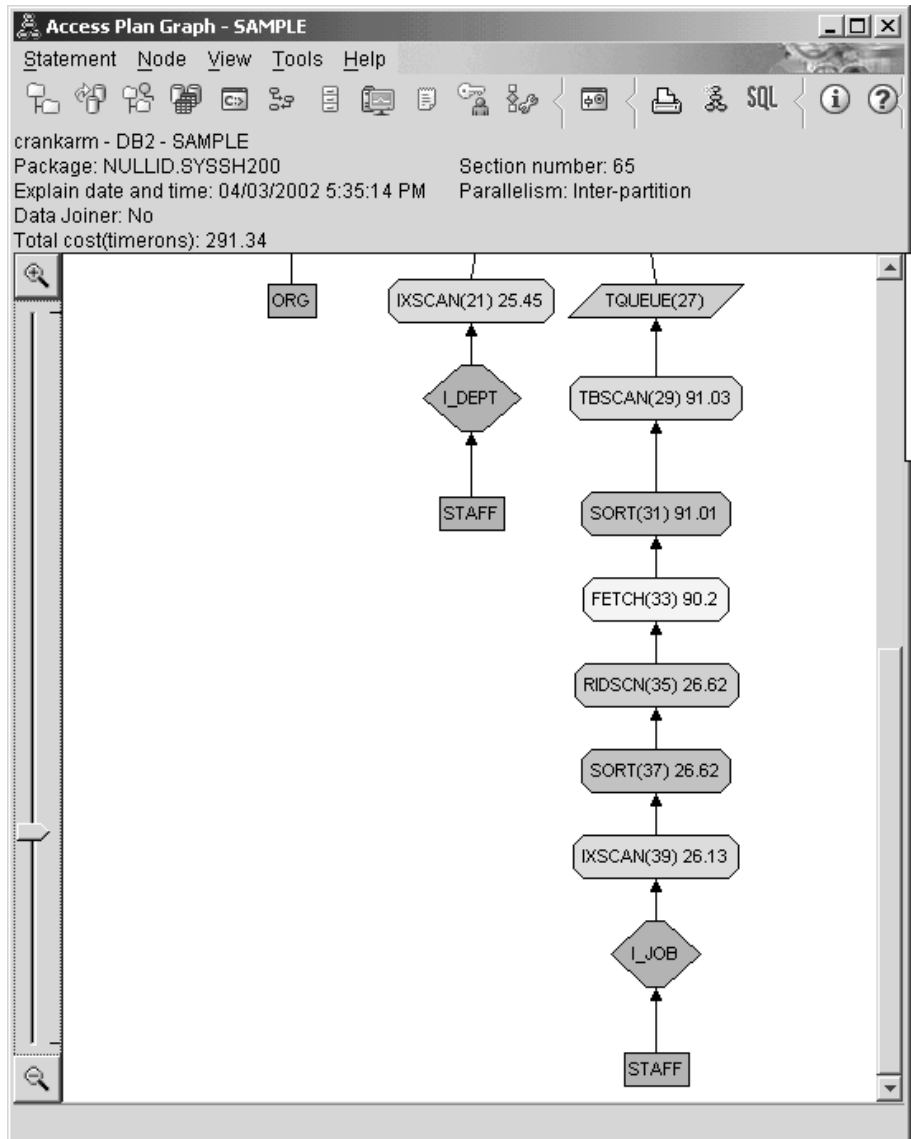
Для того чтобы понять, как улучшить запрос, ответьте на несколько вопросов.

1. Что изменилось в плане доступа в результате создания дополнительных индексов?

Обратите внимание на среднюю часть графа плана доступа: для таблицы ORG вместо оператора сканирования таблицы появился оператор сканирования индекса IXSCAN (7). Добавление столбца DEPTNAME к индексу таблицы ORG позволило оптимизатору уменьшить количество операций доступа при просмотре таблицы.



В нижней части графа плана доступа для таблицы STAFF вместо операторов сканирования индекса и выборки появился оператор сканирования индекса IXSCAN (39). Создание индекса JOB в таблице STAFF позволило оптимизатору исключить лишнюю операцию доступа (выборку).



2. Насколько эффективен этот план доступа?

Данный план доступа более эффективен с точки зрения затрат, чем предыдущий. Совокупные затраты в Запросе 4 уменьшены примерно до 288 единиц времени по сравнению с 753 единицами в Запросе 3.

---

## Что дальше

Дополнительная информация об увеличении производительности приведена в публикации *Administration Guide*. Для оценки влияния действий по оптимизации на план доступа и производительность пользуйтесь Наглядным объяснением.





---

# Приложение А. Основные понятия, применяемые в программе Наглядное объяснение

---

## План доступа

Для обработки объяснимого оператора SQL необходимы некоторые данные. *План доступа* задает порядок выполнения операций, обращающихся к этим данным. Он дает возможность просмотреть статистическую информацию о выбранных таблицах, индексах и столбцах; свойства операторов, глобальную информацию, в том числе статистическую информацию о табличных пространствах и функциях; а также параметры конфигурации, связанные с оптимизацией. С помощью программы Наглядное объяснение вы можете просмотреть план доступа оператора SQL в графической форме.

План доступа создается оптимизатором при компиляции любого объяснимого оператора SQL. Для статических операторов это происходит во время подготовки и связывания, а для динамических операторов - во время выполнения.

Важно понимать, что план доступа - это всего лишь *оценка*, полученная на основе доступной информации. Оптимизатор получает такую оценку на основе следующей информации:

- Статистической информации, хранящейся в таблицах системного каталога (если эта информация устарела, обновите ее с помощью команды `runstats`.)
- Параметров конфигурации
- Опций связывания
- Классов оптимизации запросов

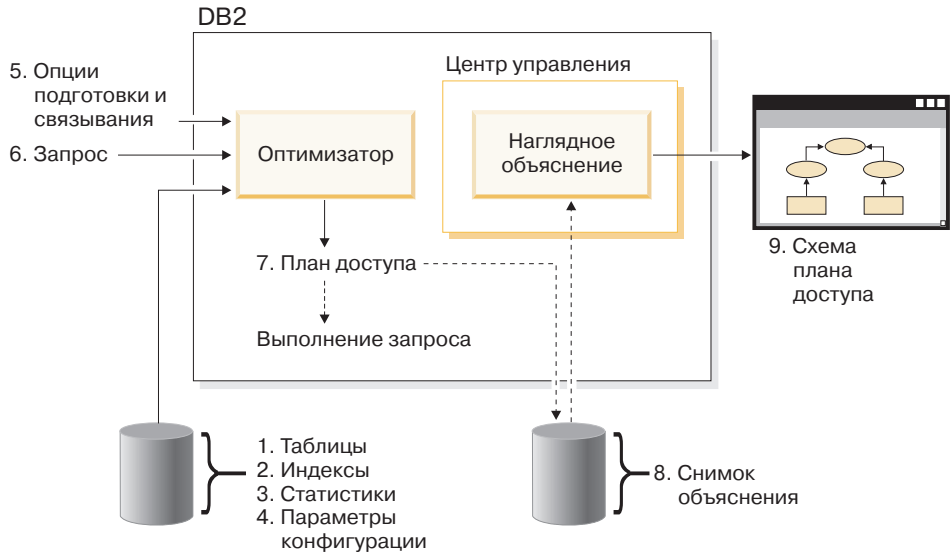
Затраты, связанные с планом доступа, - это *минимальная оценка* объема ресурсов для выполнения запроса, полученная компилятором. Фактическое время выполнения запроса может отличаться, так как оно зависит и от факторов, которые не связаны с DB2 (например, от числа других приложений, выполняющихся в то же время). Фактическое время выполнения запроса можно узнать после обработки запроса с помощью монитора производительности.

---

## Граф плана доступа

Граф плана доступа в Наглядном объяснении создается на основе нескольких источников (см. приведенный ниже рисунок). Пользуясь данными разных источников, оптимизатор выбирает план доступа, а Наглядное объяснение показывает его на *графе плана доступа*. Узлы графа представляют таблицы и индексы, а также операции над ними. Связи между узлами представляют потоки

данных.



Ниже перечислены задачи, соответствующие показанным на приведенном выше рисунке. (Прерывистые линии обозначают шаги, обязательные в Наглядном объяснении.)

1. Настроить формат таблицы и реорганизовать данные таблицы.
2. Создать подходящие индексы.
3. Собрать текущую статистику, необходимую оптимизатору, с помощью команды `runstats`.
4. Выбрать подходящие параметры конфигурации.
5. Выбрать подходящие опции связывания.
6. Создать запросы для получения только необходимых данных.
7. Создать план доступа.
8. Создать снимки объяснения.
9. Показать и использовать граф плана доступа.

Например, для работы с Наглядным объяснением сначала обновите текущую статистику, выполнив команду **runstats** по отношению к таблицам, и используемые данным оператором индексы. Статистика, параметры конфигурации, опции связывания и сам запрос используются оптимизатором при связывании пакета для создания плана доступа и снимка объяснения. Наглядное объяснение применяет полученный снимок для показа графа плана доступа для данного оператора.

## Узел на графе плана доступа

Граф плана доступа представляет собой дерево *узлов*. Узлами могут служить:

- Таблицы, которые изображаются в виде прямоугольников
- Индексы, которые изображаются в виде ромбов

- Операторы, которые изображаются в виде восьмиугольников. Операторы TQUEUE, которые изображаются в виде параллелограммов.
- Табличные функции, которые изображаются в виде шестиугольников.

---

## Кластеризация

Из-за вносимых обновлений строки на страницах данных могут изменять свое расположение, что снижает степень *кластеризации*, которая существует между индексом и страницами данных. Степень кластеризации можно повысить путем реорганизации таблицы с учетом выбранного индекса. Кластеризованный индекс наиболее эффективно работает в случае столбцов, для которых заданы предикаты диапазона, так как он повышает скорость последовательного доступа к данным в базовой таблице. В результате число страниц, просматриваемых во время выборки, уменьшается, так как все значения хранятся на одной странице данных.

В общем случае только у одного индекса таблицы может быть высокая степень кластеризации.

Для того чтобы узнать степень кластеризации индекса, дважды щелкните на его узле. Появится окно Статистика по индексу. В нем будет показана степень, или коэффициент, кластеризации. Если это значение будет небольшим, то рекомендуется реорганизовать данные таблицы.

Более подробная информация приведена в разделе, посвященном реорганизации данных таблицы, руководства *Administration Guide*.

---

## Контейнер

*Контейнер* определяет расположение данных в физической памяти. Это понятие связано с понятием табличного пространства. Контейнером может служить файл, каталог или устройство.

---

## Стоимость

*Затратами* в программе Наглядное объяснение называется оценка общего объема ресурсов, необходимых для выполнения плана доступа оператора (или элементов оператора). Затраты вычисляются на основе затрат CPU (числа инструкций) и затрат ресурсов на ввод-вывод (число операций поиска и загрузки страниц).

Затраты измеряются в *единицах времени*. Единицу времени нельзя представить в виде некоторого интервала времени. Она дает приблизительную оценку ресурсов (затрат), необходимых менеджеру баз данных для выполнения двух планов одного запроса.

В узлах операторов на графике плана доступа указаны общие затраты, которые потребуются для выполнения всех действий от начала плана доступа до данного оператора. Это значение не учитывает такие факторы, как нагрузка на систему или затраты по передаче строк данных пользователю.

---

## Указатель с блокированием

*Указатель с блокированием* позволяет менеджеру баз данных возвращать за одну операцию целый блок строк, снижая таким образом нагрузку на систему. На время обработки эти строки помещаются в кэш. Кэш создается при получении запроса OPEN CURSOR от приложения и удаляется при закрытии указателя. После обработки всех строк считывается следующий блок.

Опция BLOCKING команд **PREP** и **BIND** в сочетании с приведенными ниже параметрами позволяет задать тип указателя с объединением в блоки:

### UNAMBIG

Свойство блокирования включается только для однозначных указателей (значение по умолчанию).

**ALL** Свойство блокирования включается как для неоднозначных, так и для однозначных указателей.

**NO** Указатели с блокированием не применяются.

Подробная информация об использовании указателя с блокированием приведена в соответствующем разделе руководства *Administration Guide*.

---

## Табличное пространство, управляемое базой данных (DMS)

В базе данных существует два типа табличных пространств: пространства базы данных (DMS) и системные пространства (SMS).

Для управления табличными пространствами DMS применяется менеджер баз данных. Эти пространства создаются и настраиваются с учетом его требований.

Определение табличного пространства DMS содержит список файлов (или устройств), содержащих информацию базы данных.

Для увеличения размера табличного пространства DBS к нему можно добавить предварительно созданные файлы (или устройства). Менеджер баз данных автоматически перераспределяет данные по всем контейнерам, относящимся к табличному пространству.

В одной базе данных могут существовать табличные пространства DMS и SMS.

---

## Динамический SQL

*Операторы динамического SQL* - это операторы SQL, которые подготавливаются и выполняются во время работы прикладной программы. Существует два способа работы с динамическим SQL:

- Операторы SQL передаются на обработку в интерактивном режиме с помощью CLI или CLP
- Исходный код SQL содержится в переменных языка хоста, которые встроены в прикладную программу.

При выполнении оператора динамического SQL DB2 создает план доступа, основанный на текущей статистической информации о каталоге и параметрах конфигурации. При каждом выполнении оператора может применяться свой план доступа.

Помимо динамического SQL, может применяться статический SQL.

---

## Снимок объяснения

Программа Наглядное объяснение позволяет просматривать содержимое снимка объяснения.

*Снимок объяснения* - это краткая информация, которая собирается во время объяснения оператора SQL. Она хранится в виде двоичного объекта большого размера (BLOB) в таблице EXPLAIN\_STATEMENT. Снимок содержит следующую информацию:

- Внутреннее представление плана доступа, в том числе его операторов, а также используемых таблиц и индексов.
- Критерий, используемый оптимизатором для принятия решения. Он включает в себя статистическую информацию об объектах базы данных и общие затраты на выполнение каждого оператора.

Снимок объяснения применяется для создания графического представления плана доступа оператора SQL. Для того чтобы убедиться, что снимок объяснения создан, выполните следующие действия:

1. В менеджере баз данных должны быть созданы таблицы объяснения для хранения снимков объяснения. Более подробная информация о создании этих таблиц см. Создание таблиц объяснения в электронной справке.
2. Во время подготовки или связывания пакета, содержащего операторы статического SQL, укажите в параметре EXPLSNAP значение ALL или YES. В этом случае для всех объяснимых операторов SQL из этого пакета будет создан снимок объяснения. Более подробная информация о командах **BIND** и **PREP** приведена в руководстве *Command Reference*.
3. Во время связывания приложения, содержащего операторы динамического SQL, укажите в параметре EXPLSNAP значение ALL, либо перед

интерактивным выполнением этих операторов укажите в специальном регистре CURRENT EXPLAIN SNAPSHOT значение YES или EXPLAIN. Более подробная информация по этому вопросу приведена в разделе, посвященном снимкам объяснения, руководства *SQL Reference*.

---

## Объяснимый оператор

*Объяснимым* называется оператор SQL, для которого можно выполнить операцию объяснения.

К объяснимым относятся следующие операторы SQL:

- SELECT
- INSERT
- UPDATE
- DELETE
- VALUES

## Объясненный оператор

*Объясненным* называется оператор SQL, для которого была выполнена операция объяснения. Список объясненных операторов содержится в окне Хронология объясненных операторов.

---

## Операнд

Операндом называется объект, над которым выполняется операция. Например, таблица и индекс являются операндами таких операторов, как TBSCAN и IXSCAN.

---

## Оператор

*Операция* - это действие над данными, выполняемое в рамках плана доступа оператора SQL.

На графике плана доступа могут быть показаны следующие операции:

### **DELETE**

Удаляет строки из таблицы.

### **EISCAN**

Просматривает пользовательский индекс и создает сокращенный набор строк.

### **FETCH**

Выбирает из столбцов таблицы записи с указанным идентификатором.

**FILTER**

Фильтрует данные путем применения одного или нескольких предикатов.

**GRPBY**

Группирует строки с одинаковыми значениями в указанных столбцах, либо строки, дающие одинаковый результат при вычислении функции, а затем вычисляет функцию над полученным множеством.

**HSJOIN**

Операция объединения с помощью хеширования, в ходе которой для объединенных столбцов одной или двух таблиц выполняется операция хеширования.

**INSERT**

Вставляет строки в таблицу.

**IXAND**

Логически умножает идентификаторы строк (RID), заданные в двух и более операциях просмотра индекса.

**IXSCAN**

Просматривает индекс таблицы и создает упорядоченный набор строк. В этой операции могут быть заданы условия запуска и остановки.

**MSJOIN**

Операция объединения с помощью слияния, в которой внешняя и внутренняя таблицы должны быть упорядочены в соответствии с предикатом объединения.

**NLJOIN**

Операция объединения с помощью вложенного цикла, в ходе которой для каждой строки внешней таблицы выполняется одно обращение к внутренней таблице.

**RETURN**

Возвращает результат обработки запроса пользователю.

**RIDSCAN**

Просматривает список идентификаторов строк (RID), полученных из одного или нескольких индексов.

**SHIP**

Получает данные из удаленного источника данных. Применяется в объединенной системе.

**SORT**

Сортирует строки по заданным столбцам и, дополнительно, удаляет дубликаты записей.

**TBSCAN**

Считывает строки данных напрямую со страниц данных.

**TEMP** Сохраняет данные во временной таблице для последующего чтения (возможно, многократного).

**TQUEUE**

Передает данные таблицы от одного агента базы данных другому.

**UNION**

Объединяет строки нескольких таблиц.

**UNIQUE**

Удаляет строки, содержащие дублирующие значения в заданных столбцах.

**UPDATE**

Обновляет строки таблицы.

---

## CMPEXP

**Имя оператора:** CMPEXP

**Назначение:** Вычисление промежуточных или окончательных выражений.

(Применяется только в режиме отладки.)

---

## DELETE

**Имя оператора:** DELETE

**Назначение:** Удаление строк из таблицы.

Этот оператор выполняет необходимую операцию. Для того чтобы снизить затраты на план доступа, оптимизируйте другие операторы (просмотра, объединения), которые выбирают строки для удаления.

**Предложение по повышению производительности:**

- При удалении всех строк из таблицы воспользуйтесь оператором DROP TABLE или командой **LOAD REPLACE**.

---

## EISCAN

**Имя оператора:** EISCAN

**Назначение:** Этот оператор просматривает пользовательский индекс и создает сокращенный набор строк. При просмотре применяется множество условий запуска-прекращения поиска, предоставленных пользовательской функцией диапазона.



Данная операция позволяет сократить число рассматриваемых строк при обращении к базовой таблице (на основе предикатов).

**Предложение по повышению производительности:**

- Со временем в результате обновления базы данных индекс может стать фрагментированным, поэтому он будет занимать больше страниц, чем необходимо. Это можно исправить путем уничтожения и повторного создания индекса, либо его реорганизации.
- Если статистические данные устарели, обновите их командой `runstats`.

---

## FETCH

**Имя оператора:** FETCH

**Назначение:** Выборка столбцов из таблицы по идентификатору строки (RID).

**Предложения по повышению производительности:**

- Добавьте в индексные ключи выбранные столбцы, чтобы не требовалось обращаться к страницам данных.
- Найдите индекс, соответствующий выборке, и дважды щелкните на его узле для просмотра окна статистики. Убедитесь, что степень кластеризации для этого индекса высока.
- Увеличьте размер буфера, если операций ввода-вывода по выборке выполняется больше, чем страниц в таблице.
- Если статистические данные устарели, обновите их с помощью команды `runstats`.

Статистическая информация по частоте выборки и объему выбираемых данных указывает на избирательность предикатов, что позволяет определить, когда вместо просмотра таблицы применяется просмотр индекса. Для обновления этой статистической информации примените к таблице команду `runstats` с предложением `WITH DISTRIBUTION`.

---

## FILTER

**Имя оператора:** FILTER

**Назначение:** Приложение остаточных предикатов, позволяющее фильтровать данные на основе критериев, заданных предикатами.

**Предложения по повышению производительности:**

- Убедитесь, что вы указали только те предикаты, которые выдают нужные вам данные. Например, убедитесь, что значение избирательности для предикатов представляет именно ту часть таблицы, которая вам нужна.

- Убедитесь, что класс оптимизации равен по крайней мере 3, и оптимизатор применяет объединение вместо подзапроса. Если это невозможно, попробуйте вручную переписать запрос на языке SQL, исключив подзапросы. См., например, раздел по изменению запросов компилятором SQL в *Administration Guide*.

---

## GENROW

**Имя оператора:** GENROW

**Назначение:** Встроенная функция, порождающая таблицу строк без использования входных таблиц, индексов и операторов.

GENROW может применяться оптимизатором для создания строк данных (например, для оператора INSERT или для некоторых IN-списков, преобразуемых в объединения).

Для просмотра примерной статистической информации для таблиц, созданных функцией GENROW, дважды щелкните на ее узле.

---

## GRPBY

**Имя оператора:** GRPBY

**Назначение:** Группировка строк согласно общим значениям количества требуемых столбцов или функций. Этот оператор необходим для создания группы значений или для вычисления функций над множествами.

Если столбцы GROUP BY отсутствуют, то оператор GRPBY по-прежнему может использоваться при условии, что в списке SELECT указаны сводные функции. В этом случае GRPBY означает, что вся таблица рассматривается как единая группа.

**Предложения по повышению производительности:**

- Этот оператор выполняет необходимую операцию. Для того чтобы снизить затраты на план доступа, оптимизируйте другие операторы (просмотра, объединения), которые выбирают строки для группировки.
- Для повышения производительности в случае, когда оператор SELECT содержит одну сводную функцию и ни одного предложения GROUP BY, попробуйте сделать следующее:
  - Для сводной функции MIN(C) создайте возрастающий индекс для C.
  - Для сводной функции MAX(C) создайте убывающий индекс для C.

---

## HSJOIN

**Имя оператора:** HSJOIN

**Назначение:** Объединение с помощью хеширования, при котором табличные строки со спецификаторами хешируются для последующего прямого объединения без предварительного упорядочения содержимого таблиц.

Объединение необходимо всякий раз, когда в предложении FROM указаны ссылки на несколько таблиц. Объединение с помощью хеширования возможно тогда, когда существует предикат объединения, выравнивающий столбцы из двух разных таблиц. Предикаты объединения в этом случае должны быть в точности одного типа данных. Объединения с помощью хеширования могут появиться также из переписанного подзапроса, как в случае с оператором NLJOIN.

При объединении с помощью хеширования не требуется упорядочивать входные таблицы. Сначала просматривается внутренняя таблица объединения и создается таблица просмотра путем хеширования значений объединяемых столбцов. Затем считывается внешняя таблица с помощью хеширования значений объединяемых столбцов и их проверки в созданной таблице просмотра.

Дополнительную информацию см. в разделе, посвященном принципам объединения, в *Administration Guide*.

### **Предложения по повышению производительности:**

- Для сокращения числа объединяемых строк используйте локальные предикаты (т.е. предикаты, ссылающиеся на одну и ту же таблицу).
- Увеличьте размер кучи сортировки, чтобы в ней уместилась хеш-таблица просмотра.
- Если статистические данные устарели, обновите их с помощью команды `runstats`.

---

## INSERT

**Имя оператора:** INSERT

**Назначение:** Вставка строк в таблицу.

Этот оператор выполняет необходимую операцию. Для того чтобы снизить затраты на план доступа, оптимизируйте другие операторы (просмотра, объединения), которые выбирают строки для вставки.

---

## IXAND

**Имя оператора:** IXAND

**Назначение:** Логическое умножение (AND) результатов множественных просмотров индекса с помощью методов Dynamic Bitmap. Оператор позволяет применять предикаты, объединенные операторами AND, к множественным индексам с целью сократить до минимума число обращений к таблицам.

Этот оператор выполняется с целью:

- Сократить набор строк перед обращением к базовой таблице
- Объединить операторами AND предикаты, примененные к множественным индексам
- Объединить операторами AND результаты полуобъединений, применяемые в звездообразных объединениях.

**Предложения по повышению производительности:**

- Со временем в результате обновления базы данных индекс может стать фрагментированным, поэтому он будет занимать больше страниц, чем необходимо. Это можно исправить путем уничтожения и повторного создания индекса, либо его реорганизации.
- Если статистические данные устарели, обновите их с помощью команды `runstats`.
- В общем случае, просмотр индекса наиболее эффективен при небольшом числе строк. Для оценки числа участвующих строк оптимизатор использует статистические данные по упомянутым в предикатах столбцам. Если некоторые значения встречаются чаще остальных, важно запросить статистические данные о распределении. Для этого служит предложение `WITH DISTRIBUTION` команды `runstats`. На основе данных о неравномерном распределении оптимизатор может различать часто и редко встречающиеся значения.
- IXAND наиболее эффективен применительно к индексам отдельных столбцов, поскольку ключи запуска и останова играют важную роль при работе с ним.
- В случае звездообразных объединений создайте индексы отдельных столбцов для каждого из наиболее избираемых столбцов в фактической таблице и в связанных с ней таблицах.

---

## IXSCAN

**Имя оператора:** IXSCAN

**Назначение:** Просмотр индекса для создания сокращенного набора строк. В просмотре могут применяться необязательные условия запуска/останова; кроме того, оно может выполняться над индексируемыми предикатами, ссылающимися на столбцы индекса.

Данная операция позволяет сократить число рассматриваемых строк при обращении к базовой таблице (на основе предикатов).

Дополнительную информацию см. в разделе, посвященном просмотру индексов, в *Administration Guide*.

#### **Предложения по повышению производительности:**

- Со временем в результате обновления базы данных индекс может стать фрагментированным, поэтому он будет занимать больше страниц, чем необходимо. Это можно исправить путем уничтожения и повторного создания индекса, либо его реорганизации.
- При обращении к двум и более таблицам доступ к внутренней таблице по индексу можно сделать более эффективным, создав индекс по столбцу объединения внешней таблицы.  
Дополнительные сведения об индексах приведены в электронной справке по Наглядному объяснению.
- Если статистические данные устарели, обновите их с помощью команды `runstats`.
- В общем случае, просмотр индекса наиболее эффективен при небольшом числе строк. Для оценки числа участвующих строк оптимизатор использует статистические данные по упомянутым в предикатах столбцам. Если некоторые значения встречаются чаще остальных, важно запросить статистические данные о распределении. Для этого служит предложение `WITH DISTRIBUTION` команды **`runstats`**. На основе данных о неравномерном распределении оптимизатор может различать часто и редко встречающиеся значения.

---

## **MSJOIN**

**Имя оператора:** MSJOIN

**Назначение:** Объединение с помощью слияния. Для его применения необходимо, чтобы объединяемые строки внешней и внутренней таблиц были упорядочены по предикату объединения. Другие названия - *объединение результатов просмотра с помощью слияния* и *упорядоченное объединение с помощью слияния*.

Объединение необходимо всякий раз, когда в предложении FROM указаны ссылки на несколько таблиц. Объединение с помощью слияния возможно тогда, когда существует предикат объединения, выравнивающий столбцы из двух разных таблиц. Кроме того, это объединение может появиться из переписанного подзапроса.

При объединении с помощью слияния входные данные в объединяемых столбцах должны быть упорядоченными, поскольку таблицы обычно

просматриваются только один раз. Получить упорядоченный ввод можно с помощью индекса или упорядоченной таблицы.

Дополнительную информацию см. в разделе, посвященном принципам объединения, в *Administration Guide*.

**Предложения по повышению производительности:**

- Для сокращения числа объединяемых строк используйте локальные предикаты (т.е. предикаты, ссылающиеся на одну и ту же таблицу).

Дополнительные сведения об индексах см. в разделе Создание подходящих индексов электронной справки по Наглядному объяснению.

- Если статистические данные устарели, обновите их с помощью команды `runstats`.

---

## NLJOIN

**Имя оператора:** NLJOIN

**Назначение:** Объединение со вложенным циклом, просматривающее (обычно с помощью индекса) внутреннюю таблицу один раз для каждой строки из внешней таблицы.

Объединение необходимо всякий раз, когда в предложении FROM указаны ссылки на несколько таблиц. Для объединения со вложенным циклом предикат объединения не обязателен, но желателен, поскольку повышает производительность.

Объединение со вложенным циклом выполняется одним из следующих способов:

- Просмотр внутренней таблицы по одному разу для каждой участвующей строки из внешней таблицы.
- Поиск по индексу во внутренней таблице по одному разу для каждой участвующей строки из внешней таблицы.

Дополнительную информацию см. в разделе, посвященном принципам объединения, в *Administration Guide*.

**Предложения по повышению производительности:**

- Объединение со вложенным циклом обычно более эффективно, если для столбцов предиката объединения из внутренней таблицы (показанной справа от оператора NLJOIN) есть индекс. Если внутренняя таблица относится к типу TBSCAN, а не IXSCAN, добавьте индекс к ее объединяемым столбцам.

Другой (не столь важный) способ повысить эффективность заключается в создании индекса для объединяемых столбцов внешней таблицы, сделав последнюю упорядоченной.

Дополнительные сведения об индексах см. в разделе Создание подходящих индексов электронной справки по Наглядному объяснению.

- Если статистические данные устарели, обновите их с помощью команды `runstats`.

**Связанная информация:**

- Звездообразные объединения.

---

## PIPE

**Имя оператора:** PIPE

**Назначение:** Передача строк в исходном виде другим операторам.

(Применяется только в режиме отладки.)

---

## RETURN

**Имя оператора:** RETURN

**Назначение:** Возврат данных запроса пользователю. Это последний оператор в схеме плана доступа. Он выдает итоговые суммарные значения и затраты на план доступа.

Этот оператор выполняет необходимую операцию.

**Предложение по повышению производительности:**

- Убедитесь, что вы указали только те предикаты, которые выдают нужные вам данные. Например, убедитесь, что значение избирательности для предикатов представляет именно ту часть таблицы, которая вам нужна.

---

## RIDSCN

**Имя оператора:** RIDSCN

**Назначение:** Просмотр списка идентификаторов строк (RID), полученного по одному или нескольким индексам.

Этот оператор рассматривается оптимизатором, если:

- Предикаты объединены ключевыми словами OR или есть предикат IN. Может быть применен метод логического сложения индексов, когда результаты нескольких обращений к индексу объединяются в одной таблице.
- Для одного обращения к индексу выгодно использовать предварительную выборку из списка, поскольку сортировка идентификаторов строк перед обращением к базовым строкам повышает эффективность ввода-вывода.

---

## RQUERY

**Имя оператора:** SHIP

**Назначение:** Оператор, применяемый в системе объединения для получения данных из удаленного источника. Этот оператор рассматривается оптимизатором в том случае, если SHIP отправляет оператор SQL SELECT удаленному источнику данных для получения результатов запроса. Оператор SELECT генерируется с помощью диалекта языка SQL, поддерживаемого источником данных, и может содержать любой допустимый для последнего запрос.

**Предложения по повышению производительности:** См. главу 4 книги *Administration Guide*, том 2, *Federated Database Query and Network Tuning Information*.

---

## SORT

**Имя оператора:** SORT

**Назначение:** Сортировка строк в таблице согласно порядку, задаваемому одним или несколькими ее столбцами, с возможным исключением повторяющихся записей.

Сортировка необходима, если индекс, задающий требуемый порядок, отсутствует, или если выполнить сортировку проще, чем просмотр индекса. Сортировка обычно выполняется в конце процедуры после выборки нужных строк, либо перед объединением или группировкой данных.

Если строк слишком много, или если упорядоченные данные невозможно передать в исходной последовательности другому оператору, то необходимо создать временные таблицы, что требует больших затрат.

Дополнительную информацию о сортировке см. в книге *Administration Guide*.

**Предложения по повышению производительности:**

- Попробуйте добавить индекс к столбцам, участвующим в сортировке. Дополнительные сведения об индексах см. в разделе *Создание подходящих индексов* электронной справки по Наглядному объяснению.
- Убедитесь, что вы указали только те предикаты, которые выдают нужные вам данные. Например, убедитесь, что значение избирательности для предикатов представляет именно ту часть таблицы, которая вам нужна.
- Убедитесь, что размер временной системной таблицы достаточен для предварительной выборки, т.е. что таблица не ограничена вводом-выводом. (Для этого выберите **Statement->Show statistics->Table spaces**.)



- Если сортировку требуется выполнять часто и в большом объеме, то попробуйте увеличить значения следующих параметров конфигурации:
  - Размер кучи сортировки (sortheap). Для изменения этого параметра щелкните правой кнопкой на базе данных в разделе Control Center и выберите *Настроить* во всплывающем меню. В появившейся записной книжке выберите вкладку **Производительность**.
  - Порог кучи сортировки (sheapthres). Для изменения этого параметра щелкните правой кнопкой мыши на примере базы данных в Control Center и выберите *Настроить* во всплывающем меню. В появившейся записной книжке выберите вкладку **Производительность**.
- Если статистические данные устарели, обновите их с помощью команды `runstats`.

---

## TBSCAN

**Имя оператора:** TBSCAN

**Назначение:** Просмотр таблицы, при котором все необходимые данные (строки) считываются непосредственно из страниц данных.

Оптимизатор предпочитает этот тип просмотра индекса в следующих случаях:

- Диапазон просматриваемых значений занимает большую часть таблицы (т.е. требуется просматривать практически всю таблицу)
- Таблица невелика
- Уровень кластеризации индекса низок
- Индекс не существует

Дополнительную информацию о просмотрах таблиц и индексов см. в книге *Administration Guide*.

### **Предложения по повышению производительности:**

- Просмотр индекса более эффективен, чем просмотр таблицы, если таблица велика, а доля просматриваемых строк незначительна. Для повышения вероятности выбора оптимизатором просмотра индекса попробуйте добавить индексы к столбцам, для которых существуют предикаты выбора.

Дополнительные сведения об индексах см. в разделе Создание подходящих индексов электронной справки по Наглядному объяснению.

- Если индекс уже существует, но не использовался, то убедитесь, что для каждого из его начальных столбцов заданы предикаты выбора. Если это не так, то проверьте, высока ли степень кластеризации индекса. (Для просмотра этой информации откройте окно Статистика таблицы для таблицы, применяемой в сортировке, и нажмите кнопку *Индексы* - появится окно Статистика индекса.)

- Убедитесь, что размер временной системной таблицы достаточен для предварительной выборки, т.е. что таблица не ограничена вводом-выводом. (Для этого выберите **Statement->Show statistics->Table spaces.**)

Дополнительную информацию см. в разделе, посвященном предварительной выборке данных и занесению их в буфер, в *Administration Guide*.

- Если статистические данные устарели, обновите их с помощью команды `runstats`.

Статистическая информация по частоте выборки и объему выбираемых данных указывает на избирательность предикатов. Например, она используется при выборе между просмотром индекса и просмотром таблицы. Для обновления этой статистической информации примените к таблице команду `runstats` с предложением `WITH DISTRIBUTION`.

---

## TEMP

**Имя оператора:** TEMP

**Назначение:** Операция сохранения данных во временной таблице, из которой они впоследствии будут считаны другим оператором (возможно, несколько раз). Таблица удаляется после выполнения оператора SQL (или раньше).

Этот оператор применяется для обработки подзапросов и хранения промежуточных результатов. В некоторых ситуациях (например, если не исключено обновление оператора) он обязателен.

---

## TQUEUE

**Имя оператора:** TQUEUE

**Назначение:** Очередь таблицы, применяемая для передачи табличных данных от одного агента базы данных другому, когда запрос обрабатывается несколькими агентами. Такая обработка позволяет повысить параллелизм процесса.

Ниже перечислены типы очередей таблицы:

- **Локальная:** Очередь таблицы применяется для передачи данных между агентами базы данных в пределах одного узла. Локальная очередь таблицы используется для внутрираздельный параллелизм.
- **Не локальная:** Очередь таблицы применяется для передачи данных между агентами баз данных, находящимися на различных узлах.

---

## UNION

**Имя оператора:** UNION

**Назначение:** Конкатенация потоков строк от различных таблиц.

Этот оператор выполняет необходимую операцию. Для того чтобы снизить затраты на план доступа, оптимизируйте другие операторы (просмотра, объединения), которые выбирают строки для конкатенации.

---

## UNIQUE

**Имя оператора:** UNIQUE

**Назначение:** Исключение строк с повторяющимися значениями в указанных столбцах.

**Предложение по повышению производительности:**

- Этот оператор не требуется в том случае, если для соответствующих столбцов существует уникальный индекс.

Дополнительные сведения об индексах см. в разделе Создание подходящих индексов электронной справки по Наглядному объяснению.

---

## UPDATE

**Имя оператора:** UPDATE

**Назначение:** Обновление данных в строках таблицы.

Этот оператор выполняет необходимую операцию. Для того чтобы снизить затраты на план доступа, оптимизируйте другие операторы (просмотра, объединения), которые выбирают строки для обновления.

---

## Оптимизатор

*Оптимизатор* - это компонент компилятора SQL, который выбирает план доступа для оператора SQL языка управления данными (DML). Для этого оптимизатор оценивает затраты на выполнение оператора по нескольким планам доступа и выбирает план доступа с минимальными затратами.

---

## Пакет

*Пакет* - это объект, хранящийся в базе данных, который содержит информацию, необходимую для обработки всех операторов SQL из одного исходного файла прикладной программы. Он создается во время:

- Предварительной компиляции исходного файла с помощью команды **PREP**, либо
- Связывания файла, созданного в ходе предварительной компиляции, с помощью команды **BIND**.

---

## Предикат

*Предикат* - это элемент условия поиска, представляющий операцию сравнения. Предикаты указываются в предложениях, начинающихся со слова WHERE или HAVING.

Например, в операторе SQL:

```
SELECT * FROM SAMPLE
  WHERE NAME = 'SMITH' AND
  DEPT = 895 AND YEARS > 5
```

Содержатся следующие предикаты: NAME = 'SMITH'; DEPT = 895; и YEARS > 5.

Предикаты делятся на следующие категории, перечисленные в порядке убывания эффективности:

1. Условия начала и завершения операции просмотра индекса в прямом порядке. (Эти условия также называются предикатами, задающими диапазон.)
2. Предикаты страниц индекса, которые можно проверить с помощью индекса, так как столбцы, задаваемые в этом предикате, входят в ключ индекса.
3. Предикаты страниц данных, которые нельзя проверить с помощью индекса, однако можно проверить в тот момент, когда строки хранятся в буфере.
4. Остаточные предикаты, для обработки которых обычно требуется не только обратиться к базовой таблице, но и выполнить операцию ввода-вывода. Они применяются после удаления данных из страницы буфера. К таким предикатам относятся предикаты, содержащие подзапросы, а также предикаты, которые считывают данные LONG VARCHAR или LOB, хранящиеся отдельно от таблицы.

При выборе предикатов нужно стремиться получить максимальную избирательность, чтобы было возвращено минимальное число строк.

К наиболее эффективным (и часто используемым) относятся следующие типы предикатов:

- *Предикат объединения в виде простого равенства* задается в операции объединения с помощью слияния. Он указывается в форме table1.column = table2.column и означает, что будут объединены те таблицы, у которых значения в указанных столбцах совпадают.
- *Локальный предикат* применяется только к одной таблице.

Более подробная информация приведена в разделах, содержащих информацию о способах доступа к данным и методах его оптимизации, руководства *Administration Guide*.

---

## Класс оптимизации запроса

*Класс оптимизации запроса* - это набор правил замены запроса и способов оптимизации, применяемых при компиляции запросов.

К основным классам оптимизации запроса относятся:

- 1 Ограниченная оптимизация. Применяется в том случае, когда на ресурсы памяти и процессора наложены жесткие ограничения. Этот тип оптимизации приблизительно совпадает с оптимизацией, выполнявшейся в версии 1.
- 2 Небольшая оптимизация. На данном уровне запросы оптимизируются сильнее, чем в версии 1, однако для этого требуется значительно меньше ресурсов, чем на уровне 3 и выше, особенно в случае сложных запросов.
- 3 Значительная оптимизация. Приближается по своим свойствам к оптимизации, выполняемой в DB2 для MVS/ESA.
- 5 Обычная оптимизация. Этот уровень рекомендуется применять в том случае, если должны обрабатываться как простые транзакции, так и сложные запросы.
- 7 Обычная оптимизация. Совпадает с оптимизацией уровня 5 за исключением того, что на этом уровне для сложных запросов динамического SQL оптимизация также выполняется полностью.

Другие классы оптимизации запросов могут применяться лишь в особых случаях. К ним относятся:

- 0 Минимальная оптимизация. Применяется в том случае, когда оптимизация практически или вообще не требуется (то есть для обработки простых запросов по отношению к таблицам с эффективными индексами).
- 9 Максимальная оптимизация. Такая оптимизация требует значительных ресурсов памяти и процессора. Она применяется в том случае, если оптимизации уровня 5 недостаточно (то есть в случае очень сложных запросов, выполнение которых занимает значительное время, и которые нельзя эффективно оптимизировать на уровне 5).

В общем случае для статических запросов и запросов, обработка которых не должна занимать значительное время, рекомендуется выбирать высокий класс оптимизации, а для простых запросов, которые передаются на обработку динамически или редко выполняются, рекомендуется выбирать низкий класс оптимизации.

Для того чтобы включить функцию оптимизации запросов для операторов динамического SQL, введите в командной строке следующую команду:

```
SET CURRENT QUERY OPTIMIZATION = n;
```

где 'n' - это класс оптимизации запроса.

Для того чтобы включить функцию оптимизации запросов для операторов статического SQL, укажите опцию QUERYOPT в команде **BIND** или **PREP**.

Более подробная информация приведена в разделе руководства *Administration Guide*, содержащем рекомендации по выбору класса оптимизации.

---

## Избирательность предикатов

*Избирательность* характеризует вероятность, с которой произвольная строка таблицы соответствует предикату (для которой предикат будет истинен).

Например, предикат с избирательностью 0.01 (1%), заданный для таблицы размером в 1000000 строк, означает, что по условию, заданному этим предикатом, будет выбрано 10000 строк (1% от 1000000) и отброшено 990000 строк.

Наиболее предпочтительными являются предикаты с высоким уровнем избирательности (то есть предикаты с избирательностью не выше 0.10). По таким предикатам выбирается небольшое число строк, поэтому для последующей обработки этих строк требуется намного меньше ресурсов CPU и операций ввода-вывода.

### Пример

Предположим, что таблица содержит 1000000 строк, и запрос содержит условие 'ORDER BY', для выполнения которого требуется выполнить сортировку. Если избирательность предиката равна 0.01, то потребуется отсортировать около 10000 строк. Если же избирательность предиката равна 0.50, то потребуется отсортировать около 500000 строк - на это будет затрачено намного больше времени CPU и операций ввода-вывода.

---

## Объединение типа "звезда"

Набор операций объединения называется объединением типа "звезда", если из таблицы фактов (большая центральная таблица) в результате объединения получилось две и более таблиц ассоциаций (мелкие таблицы, содержащие описание значений столбцов таблицы фактов).

Объединение типа "звезда" состоит из трех основных компонентов:

- Частичных объединений
- Логического умножения индексов, полученных в результате частичных объединений
- Завершения частичных объединений.

Это объединение задается в виде операции IXAND, содержащей несколько вложенных операций объединения.

Частичное объединение - это особый тип объединения, результатом которого является идентификатор строк внутренней таблицы, а не объединенные столбцы внешней и внутренней таблиц.

При объединении типа "звезда" идентификаторы строк, полученные в ходе частичного объединения, используются в операции логического умножения индексов. В ходе операции логического умножения индекса фильтруются результаты нескольких объединений. Результат операции логического умножения обрабатывается с помощью логического сложения, которое упорядочивает идентификаторы строк и удаляет дубликаты строк, появившиеся после выполнения логического умножения индексов для результатов нескольких операций объединения. После этого выбираются строки из таблицы фактов. Для завершения объединения сокращенная таблица фактов объединяется со всеми таблицами ассоциаций.

#### **Рекомендации по повышению производительности:**

- Создайте индексы в таблице фактов для каждой операции объединения с таблицей ассоциаций.
- Убедитесь, что пороговый размер кучи сортировки достаточно велик для размещения двоичного фильтра операции логического умножения индексов. Для объединения типа "звезда" может потребоваться до 12 Мб памяти (3000 страниц по 4 Кб). Если применяется параллелизм внутри раздела, то двоичный фильтр размещается в том же общем сегменте памяти, что и куча базы данных, однако его размер ограничен значением параметра `sortheap` (и значением `shearphres` во всем экземпляре). Таким образом, размер общего сегмента памяти зависит от значений `sortheap` и `shearphres`, поэтому для него может потребоваться больше 12 Мб памяти.
- Задайте предикаты фильтрации для таблиц ассоциаций. Если статистическая информация устарела, обновите ее с помощью команды `runstats`.

---

## **Статический SQL**

Операторы *статического SQL* указываются непосредственно в прикладной программе. Перед выполнением приложения все такие операторы должны быть скомпилированы и связаны в один *пакет*.

Во время компиляции статического оператора DB2 создает план доступа, основанный на статистической информации каталога и значениях параметров конфигурации на момент предварительной компиляции и связывания.

Эти планы доступа используются при каждом запуске приложения. Они не изменяются до тех пор, пока пакет не будет повторно связан.

Помимо статического SQL применяется динамический SQL.

---

## Табличные пространства, управляемые системой (SMS)

В базе данных может существовать два типа табличных пространств: системные пространства (SMS) и пространства базы данных (DMS).

Управлением табличными пространствами SMS занимается операционная система. Она сохраняет информацию базы данных в области памяти, выделяемой при создании табличного пространства. Определение табличного пространства содержит список каталогов, в которых хранятся данные.

За выделение памяти на физическом носителе отвечает файловая система.

В одной базе данных могут существовать табличные пространства SMS и DMS.

---

## Табличное пространство

Для того чтобы было проще управлять базой данных, ее разделяют на несколько независимых компонентов, которые называются *табличными пространствами*.

Для каждого табличного пространства или его части можно задать набор логических устройств, на которых оно должно размещаться. Например, при создании таблицы можно указать, что индексы или столбцы, хранящие объекты большого размера (LOB), должны храниться отдельно от остальных данных таблицы.

Для повышения производительности табличное пространство можно разместить на нескольких физических устройствах (контейнерах). Однако рекомендуется, чтобы у всех устройств, или контейнеров, относящихся к одному табличному пространству, были одинаковые параметры производительности.

По способу управления табличные пространства делятся на два типа: системные пространства (SMS) и пространства базы данных (DMS).

---

## Наглядное объяснение

**Примечание:** Как и в версии 6, в данной версии программу Наглядное объяснение нельзя вызвать из командной строки. Однако ее можно вызвать из различных объектов базы данных, которые содержит Control Center. В документации по данной версии продукта эта программа по-прежнему называется Наглядное объяснение.



Программа Наглядное объяснение позволяет просмотреть графическое представление плана доступа объяснимого оператора SQL. Полученную информацию можно использовать для настройки производительности запросов SQL.

На графике плана доступа указана информация о следующих объектах:

- Таблицах (и связанных с ними столбцах) и индексах
- Операциях (например, просмотра, сортировки или объединения таблиц)
- Табличных пространствах и функциях.

Кроме того, с помощью программы Наглядное объяснение можно выполнить следующие действия:

- Просмотреть статистическую информацию, которая применялась во время оптимизации. Эту информацию можно сравнить с текущей статистической информацией каталога, чтобы узнать, позволит ли повторное связывание пакета повысить производительность.
- Узнать, применялся ли индекс для доступа к таблице. Если индекс не применялся, то с помощью программы Наглядное объяснение вы сможете узнать, для каких столбцов имеет смысл создать индекс.
- Узнать, насколько эффективными оказались внесенные изменения, путем сравнения старого и нового графиков планов доступа запроса.
- Получить информацию об отдельных операциях, входящих в план доступа, в том числе сведения об общих затратах и числе полученных строк.



---

## Приложение В. Алфавитный список операторов Наглядного объяснения

---

### СМРЕХР

**Имя оператора:** СМРЕХР

**Назначение:** Вычисление промежуточных или окончательных выражений.

(Применяется только в режиме отладки.)

---

### DELETE

**Имя оператора:** DELETE

**Назначение:** Удаление строк из таблицы.

Этот оператор выполняет необходимую операцию. Для того чтобы снизить затраты на план доступа, оптимизируйте другие операторы (просмотра, объединения), которые выбирают строки для удаления.

**Предложение по повышению производительности:**

- При удалении всех строк из таблицы воспользуйтесь оператором DROP TABLE или командой **LOAD REPLACE**.

---

### EISCAN

**Имя оператора:** EISCAN

**Назначение:** Этот оператор просматривает пользовательский индекс и создает сокращенный набор строк. При просмотре применяется множество условий запуска-прекращения поиска, предоставленных пользовательской функцией диапазона.

Данная операция позволяет сократить число рассматриваемых строк при обращении к базовой таблице (на основе предикатов).

**Предложение по повышению производительности:**

- Со временем в результате обновления базы данных индекс может стать фрагментированным, поэтому он будет занимать больше страниц, чем необходимо. Это можно исправить путем уничтожения и повторного создания индекса, либо его реорганизации.

- Если статистические данные устарели, обновите их командой `runstats`.

---

## FETCH

**Имя оператора:** FETCH

**Назначение:** Выборка столбцов из таблицы по идентификатору строки (RID).

**Предложения по повышению производительности:**

- Добавьте в индексные ключи выбранные столбцы, чтобы не требовалось обращаться к страницам данных.
- Найдите индекс, соответствующий выборке, и дважды щелкните на его узле для просмотра окна статистики. Убедитесь, что степень кластеризации для этого индекса высока.
- Увеличьте размер буфера, если операций ввода-вывода по выборке выполняется больше, чем страниц в таблице.
- Если статистические данные устарели, обновите их с помощью команды `runstats`.

Статистическая информация по частоте выборки и объему выбираемых данных указывает на избирательность предикатов, что позволяет определить, когда вместо просмотра таблицы применяется просмотр индекса. Для обновления этой статистической информации примените к таблице команду `runstats` с предложением `WITH DISTRIBUTION`.

---

## FILTER

**Имя оператора:** FILTER

**Назначение:** Приложение остаточных предикатов, позволяющее фильтровать данные на основе критериев, заданных предикатами.

**Предложения по повышению производительности:**

- Убедитесь, что вы указали только те предикаты, которые выдают нужные вам данные. Например, убедитесь, что значение избирательности для предикатов представляет именно ту часть таблицы, которая вам нужна.
- Убедитесь, что класс оптимизации равен по крайней мере 3, и оптимизатор применяет объединение вместо подзапроса. Если это невозможно, попробуйте вручную переписать запрос на языке SQL, исключив подзапросы. См., например, раздел по изменению запросов компилятором SQL в *Administration Guide*.

---

## GENROW

**Имя оператора:** GENROW

**Назначение:** Встроенная функция, порождающая таблицу строк без использования входных таблиц, индексов и операторов.

GENROW может применяться оптимизатором для создания строк данных (например, для оператора INSERT или для некоторых IN-списков, преобразуемых в объединения).

Для просмотра примерной статистической информации для таблиц, созданных функцией GENROW, дважды щелкните на ее узле.

---

## GRPBY

**Имя оператора:** GRPBY

**Назначение:** Группировка строк согласно общим значениям количества требуемых столбцов или функций. Этот оператор необходим для создания группы значений или для вычисления функций над множествами.

Если столбцы GROUP BY отсутствуют, то оператор GRPBY по-прежнему может использоваться при условии, что в списке SELECT указаны сводные функции. В этом случае GRPBY означает, что вся таблица рассматривается как единая группа.

**Предложения по повышению производительности:**

- Этот оператор выполняет необходимую операцию. Для того чтобы снизить затраты на план доступа, оптимизируйте другие операторы (просмотра, объединения), которые выбирают строки для группировки.
- Для повышения производительности в случае, когда оператор SELECT содержит одну сводную функцию и ни одного предложения GROUP BY, попробуйте сделать следующее:
  - Для сводной функции MIN(C) создайте возрастающий индекс для C.
  - Для сводной функции MAX(C) создайте убывающий индекс для C.

---

## HSJOIN

**Имя оператора:** HSJOIN

**Назначение:** Объединение с помощью хеширования, при котором табличные строки со спецификаторами хешируются для последующего прямого объединения без предварительного упорядочения содержимого таблиц.

Объединение необходимо всякий раз, когда в предложении FROM указаны ссылки на несколько таблиц. Объединение с помощью хеширования возможно тогда, когда существует предикат объединения, выравнивающий столбцы из двух разных таблиц. Предикаты объединения в этом случае должны быть в

точности одного типа данных. Объединения с помощью хеширования могут появиться также из переписанного подзапроса, как в случае с оператором NLJOIN.

При объединении с помощью хеширования не требуется упорядочивать входные таблицы. Сначала просматривается внутренняя таблица объединения и создается таблица просмотра путем хеширования значений объединяемых столбцов. Затем считывается внешняя таблица с помощью хеширования значений объединяемых столбцов и их проверки в созданной таблице просмотра.

Дополнительную информацию см. в разделе, посвященном принципам объединения, в *Administration Guide*.

#### **Предложения по повышению производительности:**

- Для сокращения числа объединяемых строк используйте локальные предикаты (т.е. предикаты, ссылающиеся на одну и ту же таблицу).
- Увеличьте размер кучи сортировки, чтобы в ней уместилась хеш-таблица просмотра.
- Если статистические данные устарели, обновите их с помощью команды runstats.

---

## **INSERT**

**Имя оператора:** INSERT

**Назначение:** Вставка строк в таблицу.

Этот оператор выполняет необходимую операцию. Для того чтобы снизить затраты на план доступа, оптимизируйте другие операторы (просмотра, объединения), которые выбирают строки для вставки.

---

## **IXAND**

**Имя оператора:** IXAND

**Назначение:** Логическое умножение (AND) результатов множественных просмотров индекса с помощью методов Dynamic Bitmap. Оператор позволяет применять предикаты, объединенные операторами AND, к множественным индексам с целью сократить до минимума число обращений к таблицам.

Этот оператор выполняется с целью:

- Сократить набор строк перед обращением к базовой таблице
- Объединить операторами AND предикаты, примененные к множественным индексам

- Объединить операторами AND результаты полуобъединений, применяемые в звездообразных объединениях.

#### **Предложения по повышению производительности:**

- Со временем в результате обновления базы данных индекс может стать фрагментированным, поэтому он будет занимать больше страниц, чем необходимо. Это можно исправить путем уничтожения и повторного создания индекса, либо его реорганизации.
- Если статистические данные устарели, обновите их с помощью команды `runstats`.
- В общем случае, просмотр индекса наиболее эффективен при небольшом числе строк. Для оценки числа участвующих строк оптимизатор использует статистические данные по упомянутым в предикатах столбцам. Если некоторые значения встречаются чаще остальных, важно запросить статистические данные о распределении. Для этого служит предложение `WITH DISTRIBUTION` команды `runstats`. На основе данных о неравномерном распределении оптимизатор может различать часто и редко встречающиеся значения.
- `IXAND` наиболее эффективен применительно к индексам отдельных столбцов, поскольку ключи запуска и останова играют важную роль при работе с ним.
- В случае звездообразных объединений создайте индексы отдельных столбцов для каждого из наиболее избираемых столбцов в фактической таблице и в связанных с ней таблицах.

---

## **IXSCAN**

**Имя оператора:** IXSCAN

**Назначение:** Просмотр индекса для создания сокращенного набора строк. В просмотре могут применяться необязательные условия запуска/останова; кроме того, оно может выполняться над индексируемыми предикатами, ссылающимися на столбцы индекса.

Данная операция позволяет сократить число рассматриваемых строк при обращении к базовой таблице (на основе предикатов).

Дополнительную информацию см. в разделе, посвященном просмотру индексов, в *Administration Guide*.

#### **Предложения по повышению производительности:**

- Со временем в результате обновления базы данных индекс может стать фрагментированным, поэтому он будет занимать больше страниц, чем необходимо. Это можно исправить путем уничтожения и повторного создания индекса, либо его реорганизации.

- При обращении к двум и более таблицам доступ к внутренней таблице по индексу можно сделать более эффективным, создав индекс по столбцу объединения внешней таблицы.  
Дополнительные сведения об индексах приведены в электронной справке по Наглядному объяснению.
- Если статистические данные устарели, обновите их с помощью команды `runstats`.
- В общем случае, просмотр индекса наиболее эффективен при небольшом числе строк. Для оценки числа участвующих строк оптимизатор использует статистические данные по упомянутым в предикатах столбцам. Если некоторые значения встречаются чаще остальных, важно запросить статистические данные о распределении. Для этого служит предложение `WITH DISTRIBUTION` команды `runstats`. На основе данных о неравномерном распределении оптимизатор может различать часто и редко встречающиеся значения.

---

## MSJOIN

**Имя оператора:** MSJOIN

**Назначение:** Объединение с помощью слияния. Для его применения необходимо, чтобы объединяемые строки внешней и внутренней таблиц были упорядочены по предикату объединения. Другие названия - *объединение результатов просмотра с помощью слияния* и *упорядоченное объединение с помощью слияния*.

Объединение необходимо всякий раз, когда в предложении FROM указаны ссылки на несколько таблиц. Объединение с помощью слияния возможно тогда, когда существует предикат объединения, выравнивающий столбцы из двух разных таблиц. Кроме того, это объединение может появиться из переписанного подзапроса.

При объединении с помощью слияния входные данные в объединяемых столбцах должны быть упорядоченными, поскольку таблицы обычно просматриваются только один раз. Получить упорядоченный ввод можно с помощью индекса или упорядоченной таблицы.

Дополнительную информацию см. в разделе, посвященном принципам объединения, в *Administration Guide*.

### **Предложения по повышению производительности:**

- Для сокращения числа объединяемых строк используйте локальные предикаты (т.е. предикаты, ссылающиеся на одну и ту же таблицу).  
Дополнительные сведения об индексах см. в разделе Создание подходящих индексов электронной справки по Наглядному объяснению.



- Если статистические данные устарели, обновите их с помощью команды `runstats`.

---

## NLJOIN

**Имя оператора:** NLJOIN

**Назначение:** Объединение со вложенным циклом, просматривающее (обычно с помощью индекса) внутреннюю таблицу один раз для каждой строки из внешней таблицы.

Объединение необходимо всякий раз, когда в предложении FROM указаны ссылки на несколько таблиц. Для объединения со вложенным циклом предикат объединения не обязателен, но желателен, поскольку повышает производительность.

Объединение со вложенным циклом выполняется одним из следующих способов:

- Просмотр внутренней таблицы по одному разу для каждой участвующей строки из внешней таблицы.
- Поиск по индексу во внутренней таблице по одному разу для каждой участвующей строки из внешней таблицы.

Дополнительную информацию см. в разделе, посвященном принципам объединения, в *Administration Guide*.

### **Предложения по повышению производительности:**

- Объединение со вложенным циклом обычно более эффективно, если для столбцов предиката объединения из внутренней таблицы (показанной справа от оператора NLJOIN) есть индекс. Если внутренняя таблица относится к типу TBSCAN, а не IXSCAN, добавьте индекс к ее объединяемым столбцам.

Другой (не столь важный) способ повысить эффективность заключается в создании индекса для объединяемых столбцов внешней таблицы, сделав последнюю упорядоченной.

Дополнительные сведения об индексах см. в разделе Создание подходящих индексов электронной справки по Наглядному объяснению.

- Если статистические данные устарели, обновите их с помощью команды `runstats`.

### **Связанная информация:**

- Звездообразные объединения.

---

## PIPE

**Имя оператора:** PIPE

**Назначение:** Передача строк в исходном виде другим операторам.

(Применяется только в режиме отладки.)

---

## RETURN

**Имя оператора:** RETURN

**Назначение:** Возврат данных запроса пользователю. Это последний оператор в схеме плана доступа. Он выдает итоговые суммарные значения и затраты на план доступа.

Этот оператор выполняет необходимую операцию.

**Предложение по повышению производительности:**

- Убедитесь, что вы указали только те предикаты, которые выдают нужные вам данные. Например, убедитесь, что значение избирательности для предикатов представляет именно ту часть таблицы, которая вам нужна.

---

## RIDSCN

**Имя оператора:** RIDSCN

**Назначение:** Просмотр списка идентификаторов строк (RID), полученного по одному или нескольким индексам.

Этот оператор рассматривается оптимизатором, если:

- Предикаты объединены ключевыми словами OR или есть предикат IN. Может быть применен метод логического сложения индексов, когда результаты нескольких обращений к индексу объединяются в одной таблице.
- Для одного обращения к индексу выгодно использовать предварительную выборку из списка, поскольку сортировка идентификаторов строк перед обращением к базовым строкам повышает эффективность ввода-вывода.

---

## RQUERY

**Имя оператора:** SHIP

**Назначение:** Оператор, применяемый в системе объединения для получения данных из удаленного источника. Этот оператор рассматривается оптимизатором в том случае, если SHIP отправляет оператор SQL SELECT удаленному источнику данных для получения результатов запроса. Оператор

SELECT генерируется с помощью диалекта языка SQL, поддерживаемого источником данных, и может содержать любой допустимый для последнего запрос.

**Предложения по повышению производительности:** См. главу 4 книги *Administration Guide*, том 2, *Federated Database Query and Network Tuning Information*.

---

## SORT

**Имя оператора:** SORT

**Назначение:** Сортировка строк в таблице согласно порядку, задаваемому одним или несколькими ее столбцами, с возможным исключением повторяющихся записей.

Сортировка необходима, если индекс, задающий требуемый порядок, отсутствует, или если выполнить сортировку проще, чем просмотр индекса. Сортировка обычно выполняется в конце процедуры после выборки нужных строк, либо перед объединением или группировкой данных.

Если строк слишком много, или если упорядоченные данные невозможно передать в исходной последовательности другому оператору, то необходимо создать временные таблицы, что требует больших затрат.

Дополнительную информацию о сортировке см. в книге *Administration Guide*.

**Предложения по повышению производительности:**

- Попробуйте добавить индекс к столбцам, участвующим в сортировке. Дополнительные сведения об индексах см. в разделе *Создание подходящих индексов* электронной справки по Наглядному объяснению.
- Убедитесь, что вы указали только те предикаты, которые выдают нужные вам данные. Например, убедитесь, что значение избирательности для предикатов представляет именно ту часть таблицы, которая вам нужна.
- Убедитесь, что размер временной системной таблицы достаточен для предварительной выборки, т.е. что таблица не ограничена вводом-выводом. (Для этого выберите **Statement->Show statistics->Table spaces**.)
- Если сортировку требуется выполнять часто и в большом объеме, то попробуйте увеличить значения следующих параметров конфигурации:
  - Размер кучи сортировки (sortheap). Для изменения этого параметра щелкните правой кнопкой на базе данных в разделе *Control Center* и выберите *Настроить* во всплывающем меню. В появившейся записной книжке выберите вкладку **Производительность**.

- Порог кучи сортировки (sheapthres). Для изменения этого параметра щелкните правой кнопкой мыши на примере базы данных в Control Center и выберите *Настроить* во всплывающем меню. В появившейся записной книжке выберите вкладку **Производительность**.
- Если статистические данные устарели, обновите их с помощью команды runstats.

---

## TBSCAN

**Имя оператора:** TBSCAN

**Назначение:** Просмотр таблицы, при котором все необходимые данные (строки) считываются непосредственно из страниц данных.

Оптимизатор предпочитает этот тип просмотра индекса в следующих случаях:

- Диапазон просматриваемых значений занимает большую часть таблицы (т.е. требуется просматривать практически всю таблицу)
- Таблица невелика
- Уровень кластеризации индекса низок
- Индекс не существует

Дополнительную информацию о просмотрах таблиц и индексов см. в книге *Administration Guide*.

### **Предложения по повышению производительности:**

- Просмотр индекса более эффективен, чем просмотр таблицы, если таблица велика, а доля просматриваемых строк незначительна. Для повышения вероятности выбора оптимизатором просмотра индекса попробуйте добавить индексы к столбцам, для которых существуют предикаты выбора.

Дополнительные сведения об индексах см. в разделе Создание подходящих индексов электронной справки по Наглядному объяснению.

- Если индекс уже существует, но не использовался, то убедитесь, что для каждого из его начальных столбцов заданы предикаты выбора. Если это не так, то проверьте, высока ли степень кластеризации индекса. (Для просмотра этой информации откройте окно Статистика таблицы для таблицы, применяемой в сортировке, и нажмите кнопку *Индексы* - появится окно Статистика индекса.)
- Убедитесь, что размер временной системной таблицы достаточен для предварительной выборки, т.е. что таблица не ограничена вводом-выводом. (Для этого выберите **Statement->Show statistics->Table spaces**.)  
Дополнительную информацию см. в разделе, посвященном предварительной выборке данных и занесению их в буфер, в *Administration Guide*.
- Если статистические данные устарели, обновите их с помощью команды runstats.

Статистическая информация по частоте выборки и объему выбираемых данных указывает на избирательность предикатов. Например, она используется при выборе между просмотром индекса и просмотром таблицы. Для обновления этой статистической информации примените к таблице команду **runstats** с предложением **WITH DISTRIBUTION**.

---

## TEMP

**Имя оператора:** TEMP

**Назначение:** Операция сохранения данных во временной таблице, из которой они впоследствии будут считаны другим оператором (возможно, несколько раз). Таблица удаляется после выполнения оператора SQL (или раньше).

Этот оператор применяется для обработки подзапросов и хранения промежуточных результатов. В некоторых ситуациях (например, если не исключено обновление оператора) он обязателен.

---

## TQUEUE

**Имя оператора:** TQUEUE

**Назначение:** Очередь таблицы, применяемая для передачи табличных данных от одного агента базы данных другому, когда запрос обрабатывается несколькими агентами. Такая обработка позволяет повысить параллелизм процесса.

Ниже перечислены типы очередей таблицы:

- **Локальная:** Очередь таблицы применяется для передачи данных между агентами базы данных в пределах одного узла. Локальная очередь таблицы используется для внутрираздельный параллелизм.
  - **Не локальная:** Очередь таблицы применяется для передачи данных между агентами баз данных, находящимися на различных узлах.
- 

## UNION

**Имя оператора:** UNION

**Назначение:** Конкатенация потоков строк от различных таблиц.

Этот оператор выполняет необходимую операцию. Для того чтобы снизить затраты на план доступа, оптимизируйте другие операторы (просмотра, объединения), которые выбирают строки для конкатенации.

---

## UNIQUE

**Имя оператора:** UNIQUE

**Назначение:** Исключение строк с повторяющимися значениями в указанных столбцах.

**Предложение по повышению производительности:**

- Этот оператор не требуется в том случае, если для соответствующих столбцов существует уникальный индекс.

Дополнительные сведения об индексах см. в разделе Создание подходящих индексов электронной справки по Наглядному объяснению.

---

## UPDATE

**Имя оператора:** UPDATE

**Назначение:** Обновление данных в строках таблицы.

Этот оператор выполняет необходимую операцию. Для того чтобы снизить затраты на план доступа, оптимизируйте другие операторы (просмотра, объединения), которые выбирают строки для обновления.

---

## Приложение С. Основы DB2

---

### Базы данных

В реляционной базе данных данные представлены как собрание таблиц. Таблица состоит из заданного числа столбцов и произвольного числа строк. Данные в каждой таблице логически связаны, а между таблицами могут быть заданы отношения. Данные можно просматривать и работать с ними на основе математических правил и операций.

Кроме самих данных, база данных содержит также информацию о структуре содержимого. В каждой базе данных содержатся: набор таблиц системного каталога, в которых описана логическая и физическая структура данных; файл конфигурации, содержащий значения параметров, связанных с базой данных; журнал восстановления, в котором регистрируются текущие транзакции и транзакции, которые могут быть архивированы.

База данных может быть как локальной, так и удаленной. База данных, физически расположенная на той рабочей станции, из которой выполняется обращение, называется локальной; расположенная на другом компьютере - удаленной.

---

### Схемы

Схема - это уникальный идентификатор, применяемый для объединения нескольких объектов базы данных (таких как таблицы, производные таблицы, индексы и алиасов). Например, таблица с именем PAYROLL может быть создана как вами, так и другим пользователем. Имя каждого объекта должно быть уникальным только в пределах схемы.

Большинство объектов базы данных имеют имя, состоящее из двух частей - имени схемы и имени объекта. При создании объекта его можно назначить в заданную схему. Если не задать схему, объект назначается в схему по умолчанию; обычно это схема ID пользователя, создавшего данный объект. Например, у пользователя по имени Smith может быть таблица с именем SMITH.PAYROLL.

Схема также является объектом базы данных. Она создается при создании первого принадлежащего схеме объекта. Схема может принадлежать отдельному пользователю, и ее владелец может управлять доступом к данным и объектам схемы.

---

## Таблицы

В реляционной базе данных данные представлены как собрание таблиц. Таблица состоит из данных, логически объединенных в столбцы и строки (записи).

Каждая таблица имеет имя; каждый столбец таблицы также имеет имя. Строки таблицы не имеют определенного порядка, но могут быть выбраны в порядке, определяемом значениями столбцов. Данные таблицы логически связаны. Все данные базы данных и таблиц собраны в табличных пространствах.



---

## Приложение D. Замечания

IBM может предлагать описанные продукты, услуги и возможности не во всех странах. Сведения о продуктах и услугах, доступных в настоящее время в вашей стране, можно получить в местном представительстве IBM. Любые ссылки на продукты, программы или услуги IBM не означают явным или неявным образом, что можно использовать только продукты, программы или услуги IBM. Разрешается использовать любые функционально эквивалентные продукты, программы или услуги, если при этом не нарушаются права IBM на интеллектуальную собственность. Однако ответственность за оценку и проверку работы любых продуктов, программ и услуг других фирм лежит на пользователе.

Фирма IBM может располагать патентами или рассматриваемыми заявками на патенты, относящимися к предмету данного документа. Получение этого документа не означает предоставления каких-либо лицензий на эти патенты. Запросы по поводу лицензий следует направлять в письменной форме по адресу:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.

По поводу лицензий, связанных с использованием наборов двухбайтных символов (DBCS), обращайтесь в отдел интеллектуальной собственности IBM в вашей стране/регионе или направьте запрос в письменной форме по адресу:

IBM World Trade Asia Corporation  
Licensing  
2-31 Roppongi 3-chome, Minato-ku  
Tokyo 106, Japan

**Следующий абзац не применяется в Великобритании или в любой другой стране/регионе, где подобные заявления противоречат местным законам: КОРПОРАЦИЯ INTERNATIONAL BUSINESS MACHINES ПРЕДСТАВЛЯЕТ ДАННУЮ ПУБЛИКАЦИЮ “КАК ЕСТЬ” БЕЗ КАКИХ-ЛИБО ГАРАНТИЙ, ЯВНЫХ ИЛИ ПОДРАЗУМЕВАЕМЫХ, ВКЛЮЧАЯ ПРЕДПОЛАГАЕМЫЕ ГАРАНТИИ СОВМЕСТИМОСТИ, РЫНОЧНОЙ ПРИГОДНОСТИ И СООТВЕТСТВИЯ ОПРЕДЕЛЕННОЙ ЦЕЛИ, НО НЕ ОГРАНИЧИВАЯСЬ ИМИ. В некоторых странах для определенных сделок подобные оговорки не допускаются; таким образом, это утверждение может не относиться к вам.**

Данная информация может содержать технические неточности и типографские опечатки. Периодически в информацию вносятся изменения, они будут включены в новые издания этой публикации. Фирма IBM может в любое время без уведомления вносить изменения и усовершенствования в продукты и программы, описанные в этой публикации.

Любые ссылки в данной информации на Web-сайты, не принадлежащие IBM, приводятся только для удобства и никоим образом не означают поддержки IBM этих Web-сайтов. Материалы этих Web-сайтов не являются частью данного продукта IBM, и вы можете использовать их только на собственную ответственность.

IBM может использовать или распространять присланную вами информацию любым способом, как фирма сочтет нужным, без каких-либо обязательств перед вами.

Если обладателю лицензии на данную программу понадобятся сведения о возможности: (i) обмена данными между независимо разработанными программами и другими программами (включая данную) и (ii) совместного использования таких данных, он может обратиться по адресу:

IBM Canada Limited  
Office of the Lab Director  
8200 Warden Avenue  
Markham, Ontario  
L6G 1C7  
CANADA

Такая информация может быть предоставлена на определенных условиях (в некоторых случаях к таким условиям может относиться оплата).

Лицензированная программа, описанная в данном документе, и все лицензированные материалы, доступные с ней, предоставляются IBM на условиях IBM Customer Agreement (Соглашения IBM с заказчиком), Международного соглашения о лицензиях на программы IBM или эквивалентного соглашения.

Приведенные данные о производительности измерены в контролируемой среде. Таким образом, результаты, полученные в других операционных средах, могут существенно отличаться от них. Некоторые показатели измерены получены в системах разработки и нет никаких гарантий, что в общедоступных системах эти показатели будут теми же. Более того, некоторые результаты могут быть получены путем экстраполяции. Реальные результаты могут отличаться от них. Пользователи должны проверить данные для своих конкретных сред.

Информация о продуктах других фирм получена от поставщиков этих продуктов, из их опубликованных объявлений или из других общедоступных

источников. Фирма IBM не проверяла эти продукты и не может подтвердить точность измерений, совместимость или прочие утверждения о продуктах других фирм. Вопросы о возможностях продуктов других фирм следует направлять поставщикам этих продуктов.

Все утверждения о будущих планах и намерениях IBM могут быть изменены или отменены без уведомлений, и описывают исключительно цели фирмы.

Эта информация может содержать примеры данных и отчетов, иллюстрирующие типичные деловые операции. Чтобы эти примеры были правдоподобны, в них включены имена лиц, названия компаний и товаров. Все эти имена и названия вымышлены и любое их сходство с реальными именами и адресами полностью случайно.

#### ЛИЦЕНЗИЯ НА КОПИРОВАНИЕ:

Эта информация может содержать примеры прикладных программ на языках программирования, иллюстрирующих приемы программирования для различных операционных платформ. Разрешается копировать, изменять и распространять эти примеры программ в любой форме без оплаты фирме IBM для целей разработки, использования, сбыта или распространения прикладных программ, соответствующих интерфейсу прикладного программирования операционных платформ, для которых эти примера программ написаны. Эти примеры не были всесторонне проверены во всех возможных условиях. Поэтому IBM не может гарантировать их надежность, пригодность и функционирование.

Каждая копия программ примеров или программ, созданных на их основе, должна содержать следующее замечание об авторских правах:

© (название вашей фирмы) (год). Части этого кода построены на основе примеров программ IBM Corp. © Copyright IBM Corp. *вставьте год или годы*. Все права защищены.

---

## Товарные знаки

Следующие термины, используемые по крайней мере в одном из документов библиотеки документации DB2 UDB, являются товарными знаками корпорации International Business Machines в Соединенных Штатах и/или в других странах.

ACF/VTAM	LAN Distance
AISPO	MVS
AIX	MVS/ESA
AIXwindows	MVS/XA
AnyNet	Net.Data
APPN	NetView
AS/400	OS/390
BookManager	OS/400
C Set++	PowerPC
C/370	pSeries
CICS	QBIC
Database 2	QMF
DataHub	RACF
DataJoiner	RISC System/6000
DataPropagator	RS/6000
DataRefresher	S/370
DB2	SP
DB2 Connect	SQL/400
DB2 Extenders	SQL/DS
DB2 OLAP Server	System/370
DB2 Universal Database	System/390
Distributed Relational Database Architecture	SystemView
DRDA	Tivoli
eServer	VisualAge
Extended Services	VM/ESA
FFST	VSE/ESA
First Failure Support Technology	VTAM
IBM	WebExplorer
IMS	WebSphere
IMS/ESA	WIN-OS/2
iSeries	z/OS
	zSeries

Следующие термины, используемые по крайней мере в одном из документов библиотеки документации DB2 UDB, являются товарными знаками или зарегистрированными товарными знаками других компаний:

Microsoft, Windows, Windows NT и логотип Windows - товарные знаки Microsoft Corporation в Соединенных Штатах и в других странах.

Intel и Pentium - товарные знаки Intel Corporation в Соединенных Штатах и/или других странах.

Java и все товарные знаки на основе Java - товарные знаки Sun Microsystems, Inc. в Соединенных Штатах и/или в других странах.

UNIX - зарегистрированный товарный знак The Open Group в Соединенных Штатах и в других странах.

Названия других компаний, продуктов и услуг могут быть товарными знаками или марками сервиса других фирм.



# Индекс

## С

CMPEXP, оператор  
определение 56, 75

## D

DELETE, оператор  
определение 56, 75

## E

EISCAN, оператор  
определение 56, 75  
EXPLAIN.DDL, файл/команда 1  
EXPLSNAP, опция (в команде  
BIND) 4

## F

FETCH, оператор  
определение 57, 76  
FILTER, оператор  
определение 57, 76

## G

GENROW, функция  
определение 58, 76  
GRPBY, оператор  
определение 58, 77

## H

HSJOIN, оператор  
определение 59, 77

## I

INSERT, оператор  
определение 59, 78  
IXAND, оператор  
определение 60, 78  
IXSCAN, оператор  
определение 60, 79

## L

LIST TABLES, команда 1

## M

MSJOIN, оператор  
определение 61, 80

## N

NLJOIN, оператор  
определение 62, 81

## P

PIPE, оператор  
определение 63, 82

## R

RETURN, оператор  
определение 63, 82  
RIDSCN, оператор  
определение 63, 82

## S

SET CURRENT EXPLAIN SNAPSHOT,  
команда 4  
SHIP, оператор  
определение 64, 82  
SORT, оператор  
определение 64, 83

## T

TBSCAN, оператор  
определение 65, 84  
TEMP, оператор  
определение 66, 85  
TQUEUE, оператор  
определение 66, 85

## U

UNION, оператор  
определение 66, 85  
UNIQUE, оператор  
определение 67, 86  
UPDATE, оператор  
определение 67, 86

## V

VESAMPL.DDL, команда 2

## B

блокировка строк  
блокировка указателя 52  
блокировка указателя  
определение 52

## V

встроенные функции, получение  
статистики 10  
выполнение запроса без индексов и  
статистики 32

## Г

граф плана доступа, изменение  
вида 11  
граф плана доступа, просмотр и  
использование 7  
граф плана доступа, сведения об  
объекте 9  
граф плана доступа, чтение  
символов 7  
графы плана доступа  
создание  
определение 49  
список использованных  
операторов 54  
узлы  
определение 50

## Д

динамический SQL  
определение 53

## З

запрос без индексов и статистики 14

## И

избирательность предикатов  
определение 70  
индексы  
кластеризация  
определение 51

## К

классы оптимизации запроса  
определение 69  
команды, EXPLAIN.DDL 1  
команды, LIST TABLES 1  
команды, SET CURRENT EXPLAIN  
SNAPSHOT 4  
команды, VESAMPL.DDL 2  
команды, опция EXPLSNAP в  
BIND 4  
контейнеры  
определение 51

## Н

Наглядное объяснение  
описание 72

## О

- объединения типа "звезда"
  - определение 70
- объясненные операторы
  - определение 54
- объясненные операторы SQL,
  - выбор 7
- объяснимые операторы
  - определение 54
- операнды
  - определение 54
- операторы динамического SQL,
  - создание снимков объяснения 3
- операторы статического SQL,
  - создание снимков объяснения 4
- операторы, получение сведений 10
- операции
  - определение 54
  - список 54
- оптимизатор
  - определение 67
- опции связывания, получение информации 11

## П

- пакеты
  - определение 67
- параметры конфигурации, получение информации 11
- план доступа, улучшение 13, 31
- планы доступа
  - определение 49
- пользовательские функции, получение статистики 10
- предикаты
  - определение 68

## Р

- регулятор масштаба, увеличение графов плана доступа 8

## С

- сбор текущей статистики для таблиц и индексов 17, 35
- снимки объяснения
  - определение 53
- снимки объяснения для операторов динамического SQL, создание 3
- снимки объяснения для операторов статического SQL, создание 4
- снимки объяснения, образцы для Наглядного объяснения 2
- снимки объяснения, создание 1
- снимки, образцы для Наглядного объяснения 2

- создание дополнительных индексов для столбцов таблицы 27, 43
- создание индексов для столбцов, использованных для объединения таблиц в запросе 21, 39
- статистика для таблиц, индексов и функций таблицы 9
- статический SQL
  - определение 71
- стоимость
  - определение 51
- столбцы в операторах SQL, получение статистики 11

## Т

- таблицы объяснения, создание 1
- табличные пространства DMS
  - определение 52
  - определение 72
- табличные пространства DMS
  - определение 52
- табличные пространства, получение статистики 10
- табличные пространства, управляемые системой
  - определение 72

## Ф

- файлы, EXPLAIN.DDL 1
- функции, получение статистики 10



---

## Как связаться с IBM

В Соединенных Штатах позвоните по одному из следующих номеров:

- 1-800-237-5511, чтобы обратиться в службу поддержки заказчиков
- 1-888-426-4343, чтобы узнать о доступных формах обслуживания.
- 1-800-IBM-4YOU (426-4968), чтобы обратиться в отдел маркетинга и продаж DB2

В Канаде позвоните по одному из следующих номеров:

- 1-800-IBM-SERV (1-800-426-7378), чтобы обратиться в службу поддержки заказчиков
- 1-800-465-9600, чтобы узнать о доступных формах обслуживания.
- 1-800-IBM-4YOU (1-800-426-4968), чтобы обратиться в отдел маркетинга и продаж DB2

Адрес отделения IBM в вашей стране или регионе можно найти на странице IBM Directory of Worldwide Contacts в Интернете по адресу [www.ibm.com/planetwide](http://www.ibm.com/planetwide)

---

## Информация о продукте

Информацию о продуктах DB2 Universal Database можно получить по телефону или в Интернете по адресу [www.ibm.com/software/data/db2/udb](http://www.ibm.com/software/data/db2/udb)

Этот сайт содержит свежую информацию по технической библиотеке, заказу книг, загружаемые клиенты, группы новостей, пакеты FixPaks, новости и ссылки на ресурсы в Интернете.

Если вы находитесь в США, позвоните по одному из следующих номеров:

- 1-800-IBM-CALL (1-800-426-2255), чтобы заказать продукты или получить общую информацию.
- 1-800-879-2755, чтобы заказать публикации.

Информацию о том, как связаться с IBM из других стран, смотрите на странице IBM Worldwide по адресу [www.ibm.com/planetwide](http://www.ibm.com/planetwide)



Напечатано в Дании