ГУАП

КАФЕДРА № 42

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| канд. тех. наук, доцент |  |  |  | А. В. Бржезовский |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
| ИНДЕКСАЦИЯ ДАННЫХ |
| по курсу: МЕТОДЫ И СРЕДСТВА ПРОЕКТИРОВАНИЯ  ИНФОРМАЦИОННЫХ СИСТЕМ И ТЕХНОЛОГИЙ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ гр. № | 4121 |  |  |  | А. М .Антонов |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2024

**Задание:**

Произвести генерацию и вставку тестовых данных в БД, выполнить запросы из ЛР 3..5 или аналогичные им, зафиксировать планы и время выполнения запросов, создать систему индексов для ускорения выполнения запросов, повторно выполнить запросы, зафиксировать планы и время выполнения.

Для исключения влияния кэширования при оценке времени выполнения запросов можно воспользоваться директивами:

checkpoint

go

dbcc freeproccache

go

dbcc freesystemcache ('all')

go

dbcc dropcleanbuffers

go

Альтернативой является перезапуск службы MS SQL.

С помощью директивы hints в операторе select задать оптимизатору решения относительно использования индексов, алгоритмов соединения таблиц, сравнить планы выполнения с созданными оптимизатором.

Индивидуальный вариант задания №5: Создайте базу данных для хранения следующих сведений: ВУЗ, студент, группа, факультет, конференция, тема доклада, программа конференции.

**Ход работы:**

Для проверки выполнения запросов и оценки эффективности выбранных индексов необходимо, чтобы в таблицах базы данных содержалось определенное количество записей. Для генерации тестовых данных воспользуемся генератором данных RedGate, на каждую таблицу были созданы от 1 000 000 (таблица «ВУЗ») до 3 000 000 (таблица «Студенты») столбцов. На рисунке 1 представлена процедура генерации данных.

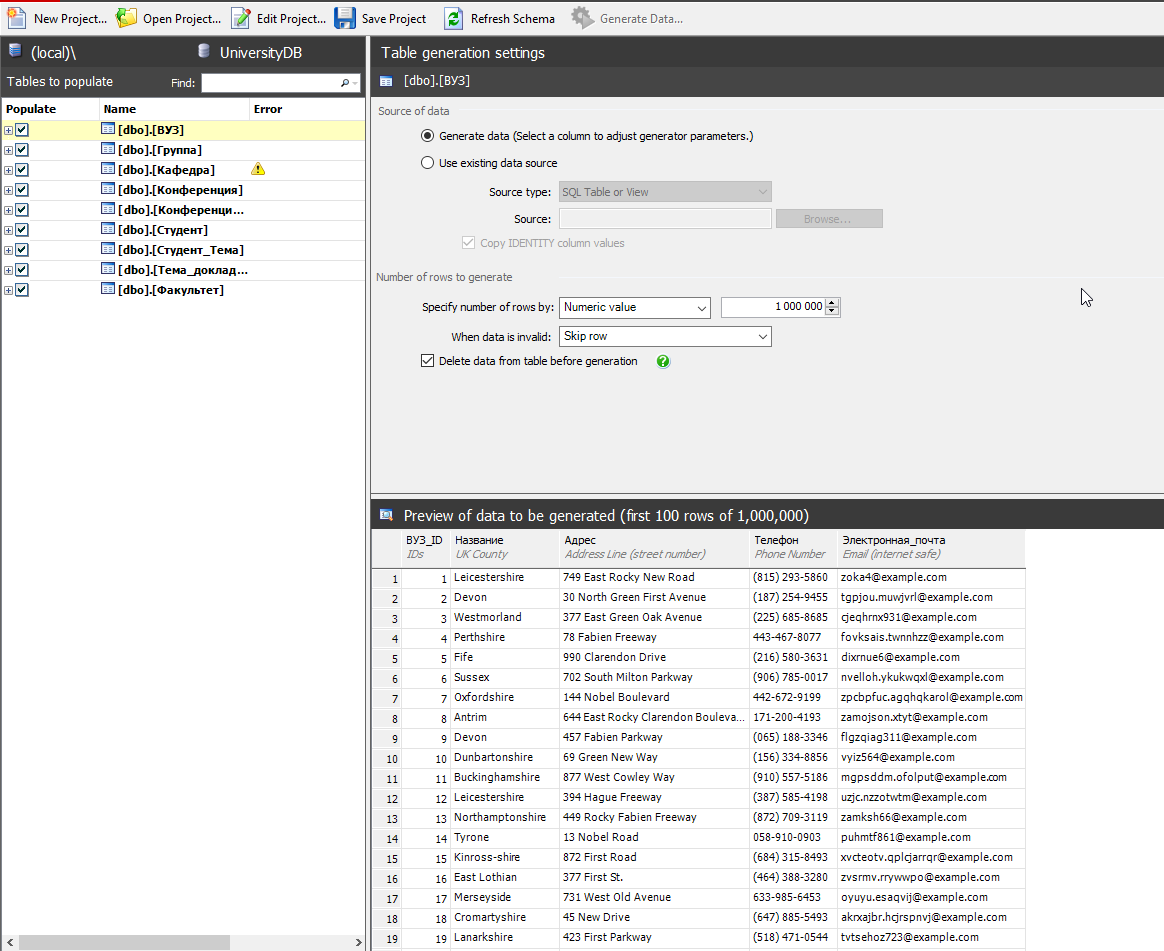


Рисунок 1 – Генерация тестовых данных

На рисунке 2 показаны индексы для нескольких таблиц базы данных. Индексы используются для ускорения поиска и выполнения запросов, особенно для полей, которые часто участвуют в соединениях (JOIN) или условиях фильтрации (WHERE).

Описание каждого индекса:

1) IX\_Студент\_Группа на таблице Студент по полю Группа\_ID - этот индекс ускоряет поиск студентов, связанных с конкретной группой, ссылаясь на поле Группа\_ID. Это полезно, когда есть запросы, связанные с получением всех студентов из одной группы.

2) IX\_Группа\_Кафедра на таблице Группа по полю Кафедра\_ID - этот индекс оптимизирует запросы, которые ищут группы, связанные с определенной кафедрой, благодаря полю Кафедра\_ID.

3) IX\_Кафедра\_Факультет на таблице Кафедра по полю Факультет\_ID - этот индекс позволяет быстро находить кафедры, относящиеся к конкретному факультету. Индекс полезен для запросов, где нужно найти кафедры по факультету.

4) IX\_Студент\_Тема на таблице Студент\_Тема по полям Студент\_ID и Тема\_ID - этот составной индекс ускоряет запросы, которые ищут темы, связанные с определенным студентом или находят студентов, связанных с определенной темой.

5) IX\_Тема\_доклада\_Конференция на таблице Тема\_доклада по полю Конференция\_ID – этот индекс оптимизирует запросы, которые ищут темы докладов, связанные с конкретной конференцией.

6) IX\_Конференция\_Факультет на таблице Конференция по полю ВУЗ\_ID - этот индекс ускоряет поиск конференций, проводимых определенным вузом, используя поле ВУЗ\_ID.

7) IX\_Группа\_Номер на таблице Группа — индекс по полю Номер. Этот индекс ускоряет поиск и фильтрацию строк в таблице Группа по значению столбца Номер, который может, например, представлять номер группы студентов.

8) IX\_Тема\_доклада\_Тема\_Конференция на таблице Тема\_доклада — составной индекс по полям Тема\_ID и Конференция\_ID. Этот индекс ускоряет запросы, в которых одновременно используются Тема\_ID и Конференция\_ID для фильтрации или соединения с другими таблицами, что позволяет быстро находить все доклады по определённой теме и конференции.

9)IX\_Конференция\_ВУЗ\_Название на таблице Конференция — составной индекс по полям ВУЗ\_ID и Название. Этот индекс полезен при запросах, которые фильтруют данные сразу по идентификатору вуза (ВУЗ\_ID) и названию конференции (Название). Индекс повышает скорость доступа к конференциям, связанным с конкретным вузом и имеющим определённое название.

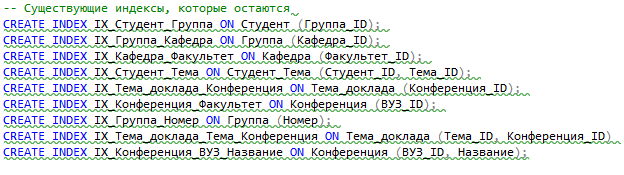


Рисунок 2 – Индексы для базы данных

Первый запрос позволяет выбрать студентов первого факультета, выступавших на конференции «Technical». На рисунках 3-4 представлены предполагаемые и действительные планы запроса. Время выполнения без использования индекса составляет 9382 мс.

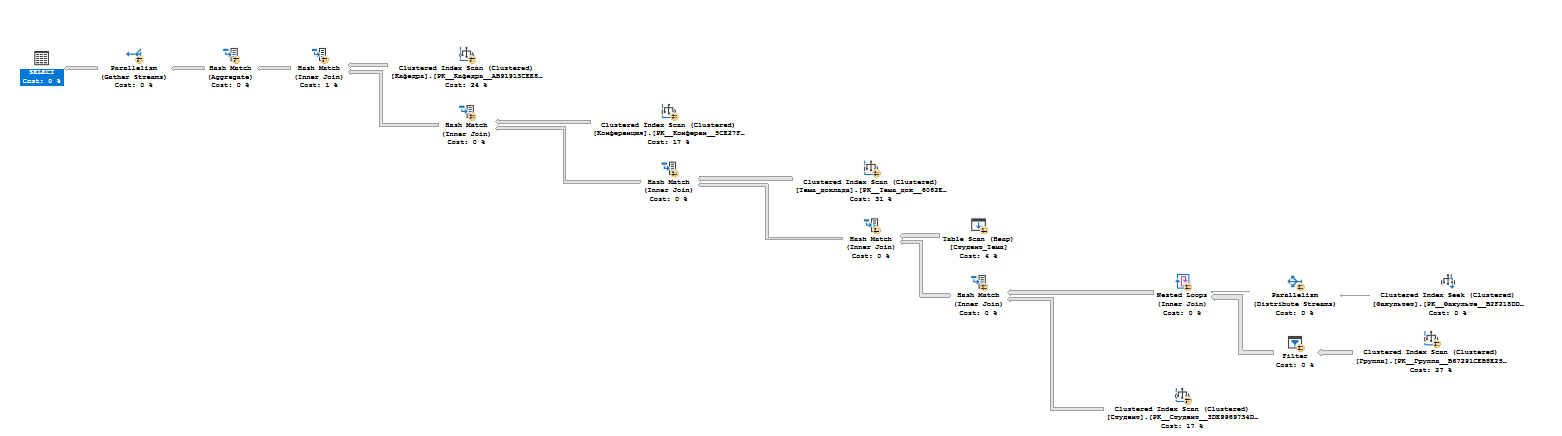


Рисунок 3 – Предполагаемый план запроса

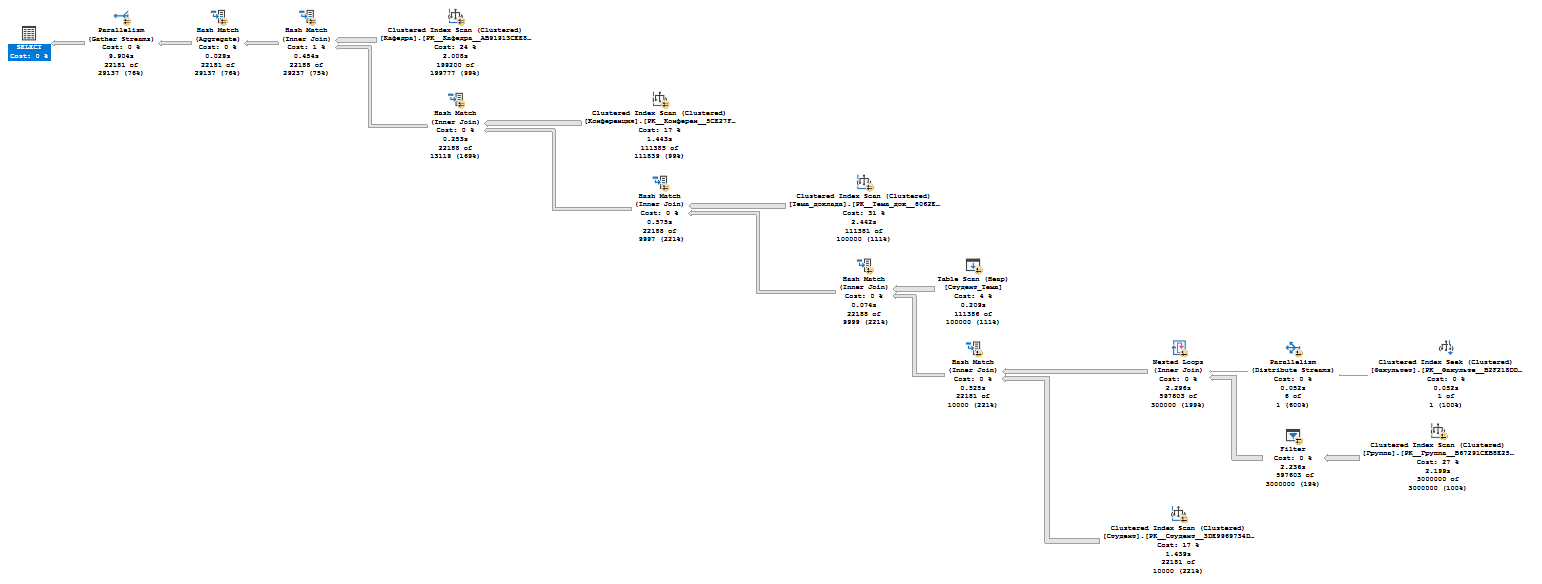


Рисунок 4 – Действительный план запроса

Создадим индексы и проверим работоспособность. Время выполнения запроса с использованием индекса составило 4408 мс, это значит, что индекс ускорил работу запроса. На рисунках 5-6 предполагаемые и действительные планы запроса с использованием индекса.

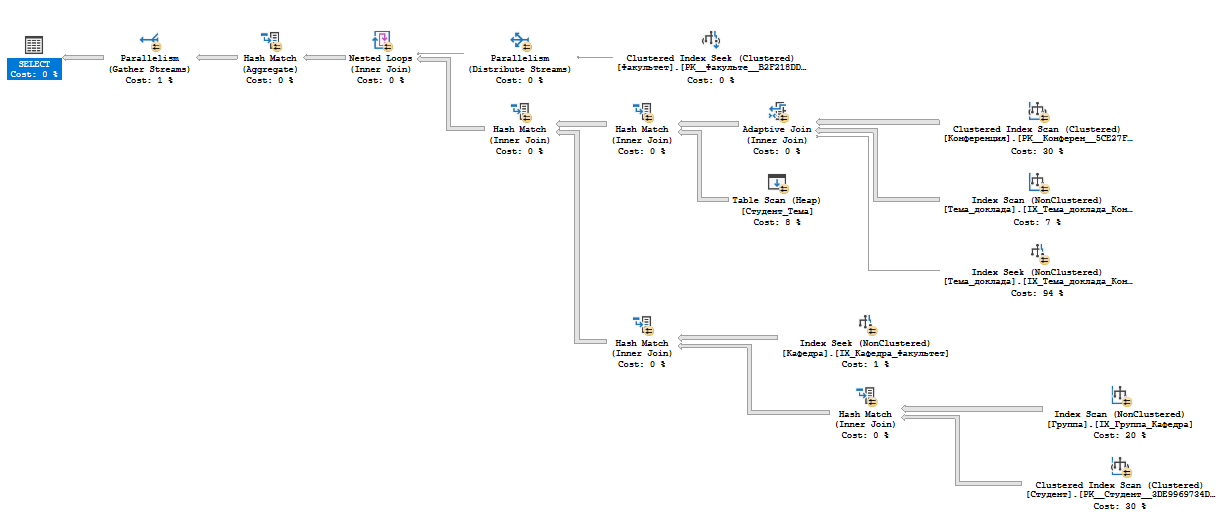


Рисунок 5 – Предполагаемый план запроса с индексом

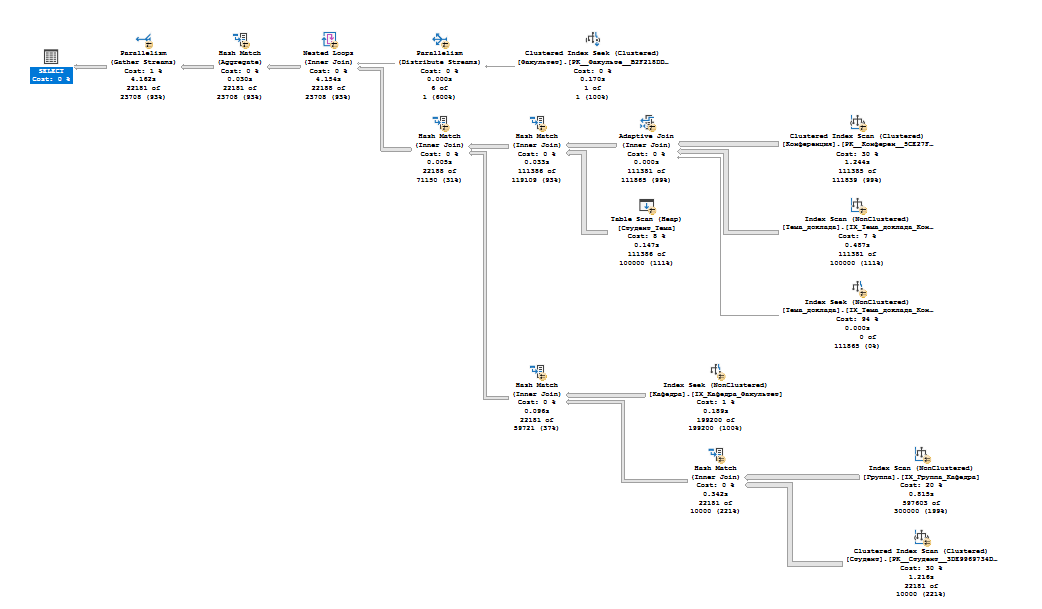


Рисунок 6 – Действительный план запроса с индексом

Также для этого запроса была применена директива HINT для управления оптимизацией запроса. Код запроса представлен на рисунке 7.

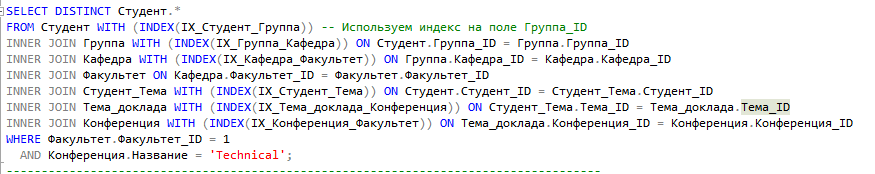


Рисунок 7 – Код запроса с hint

Время выполнения запроса с использованием индекса составило 4408 мс, а с использованием директивы HINT составило 2313 мс, что говорит о эффективности использования hint. На рисунках 8-9 представлены предполагаемые и действительные планы запроса с использованием hint.

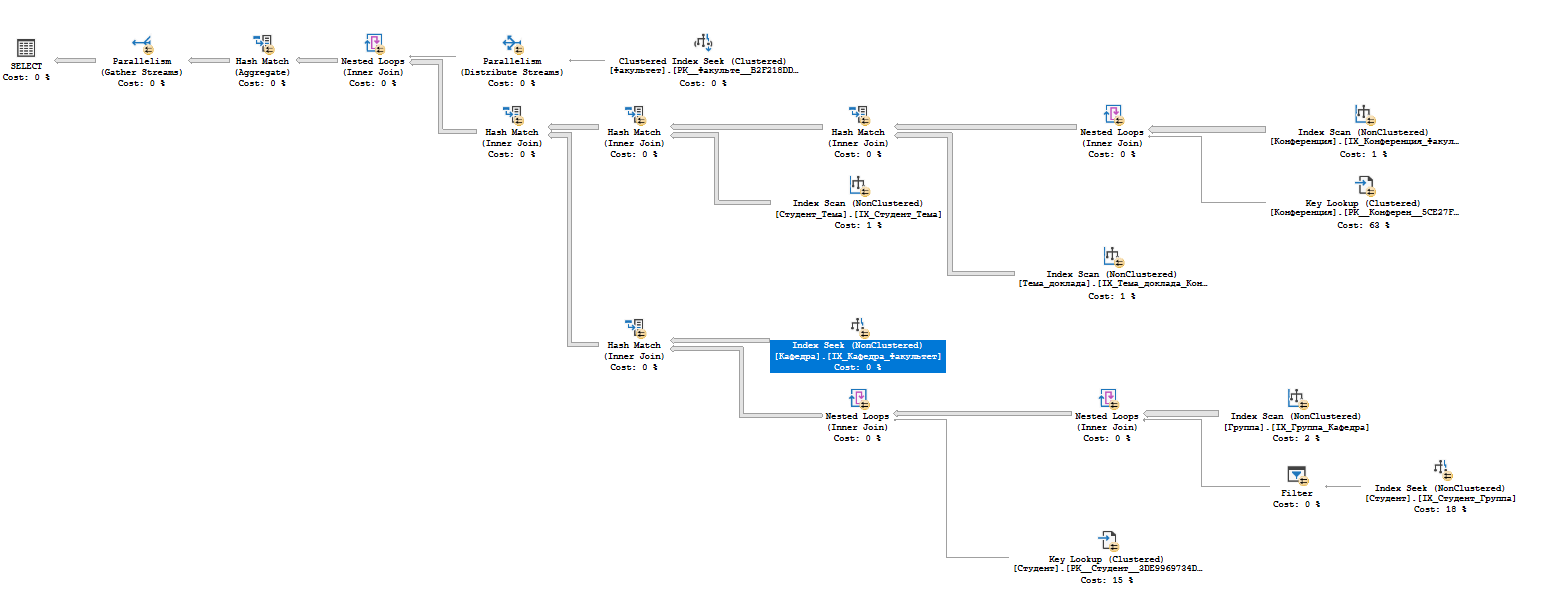


Рисунок 8 - Предполагаемый план запроса

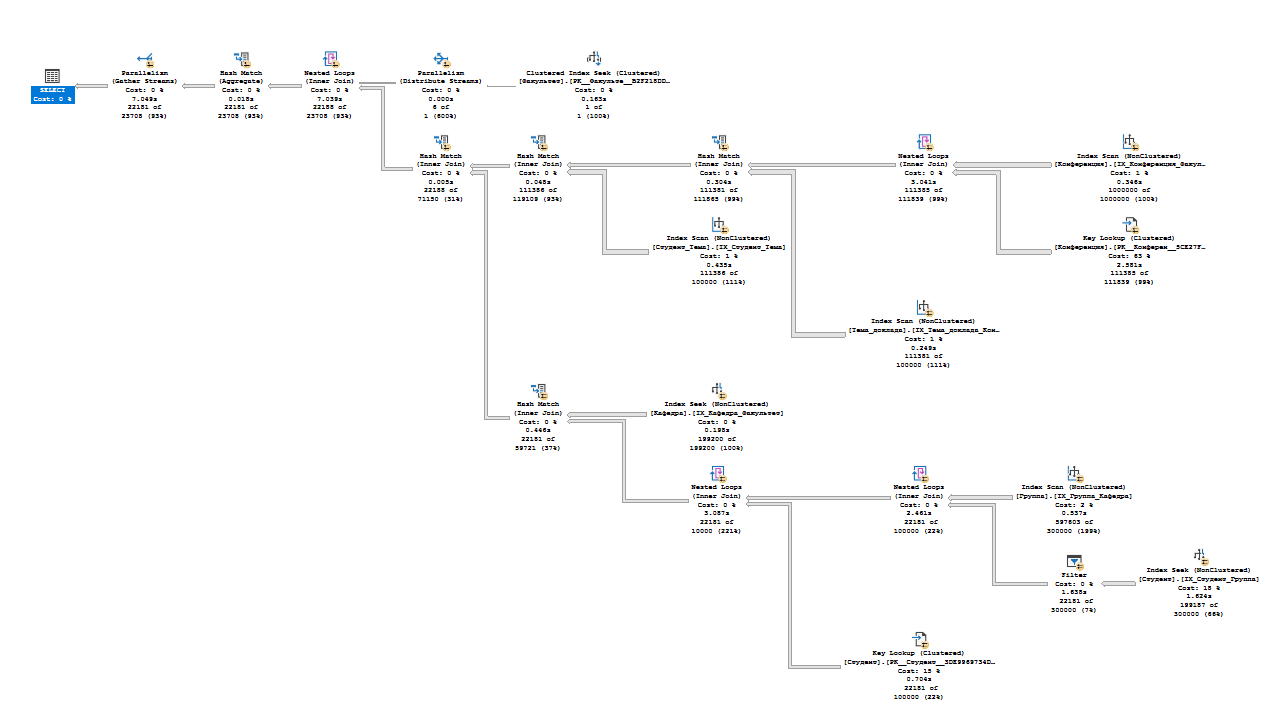


Рисунок 9 - Действительный план запроса

Второй запрос выбирает темы докладов студентов для заданной группы. Время выполнения без использования индекса составляет 5946 мс. На рисунках 10-11 представлены планы запросов.

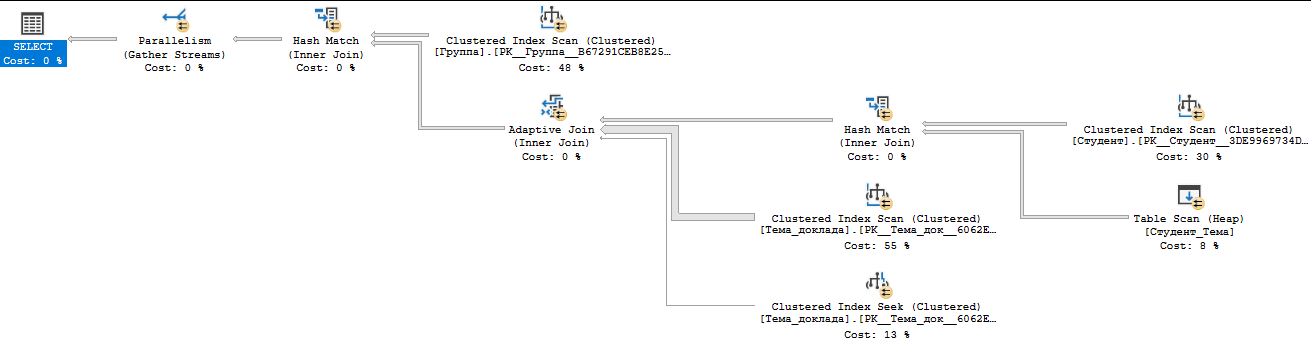


Рисунок 6 – Предполагаемый план запроса

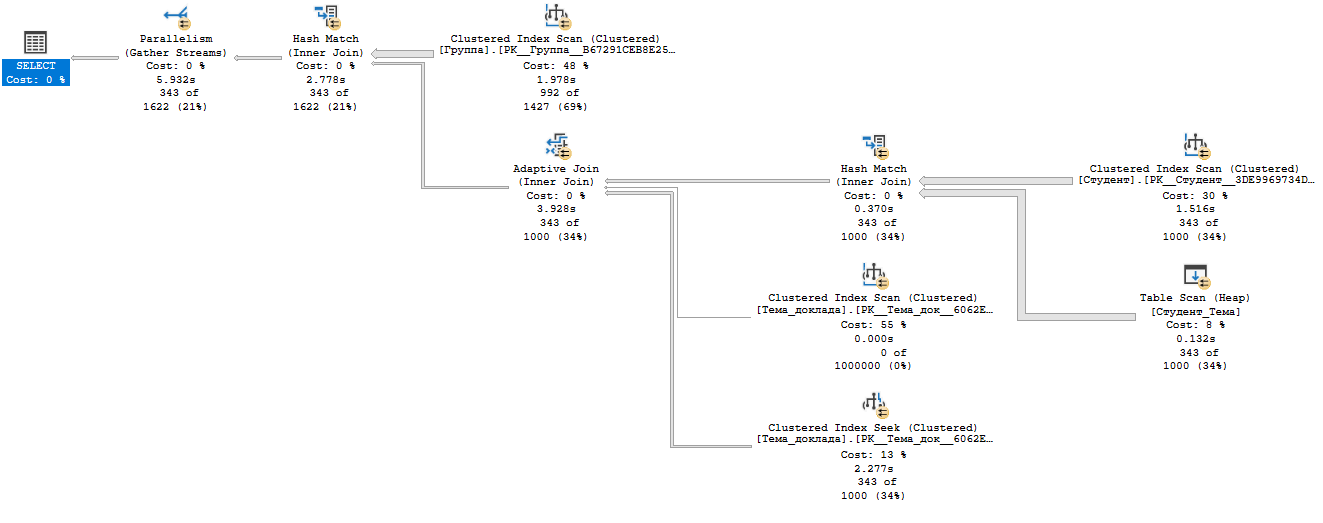


Рисунок 7 – Действительный план запроса

Создадим индекс для фильтрации по номеру группы. Время выполнения запроса с индексом составило 2418 мс, можно сказать, что индекс оптимизирует запрос. На рисунках 8-9 представлены планы запросов с индексом.

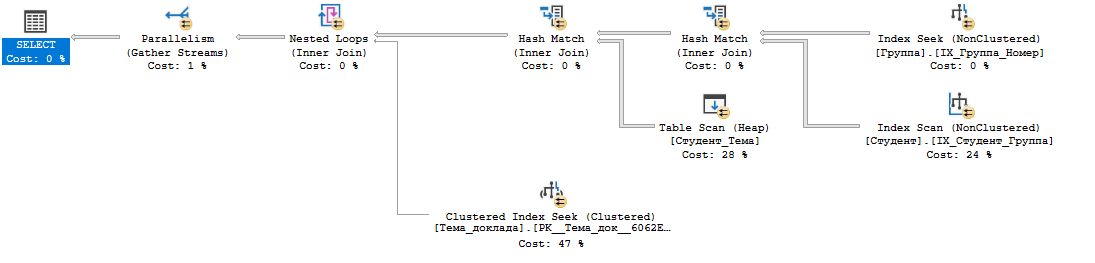


Рисунок 8 – Предполагаемый план запроса с индексом

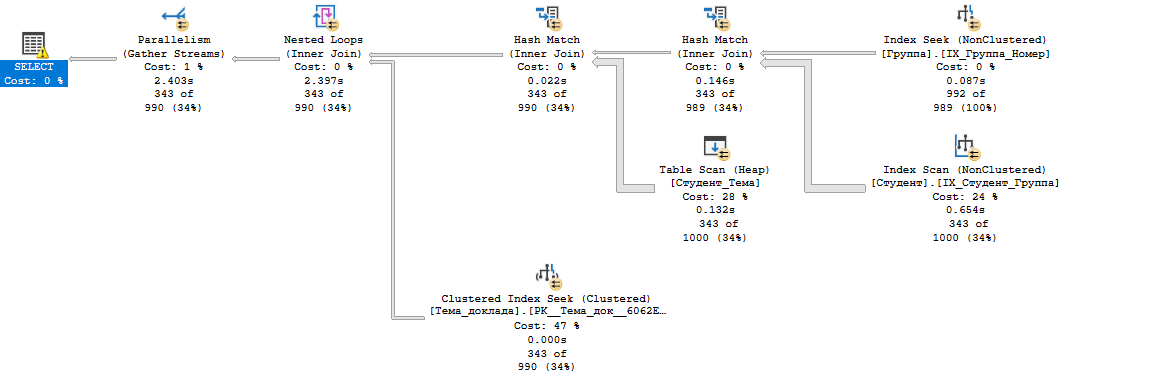


Рисунок 9 – Действительный план запроса с индексом

Третий запрос позволяет выбрать выступления, подготовленные двумя студентами различных факультетов. Планы запросов представлены на рисунках 10-11, время выполнения запроса без использования индекса составляет 9157 мс.

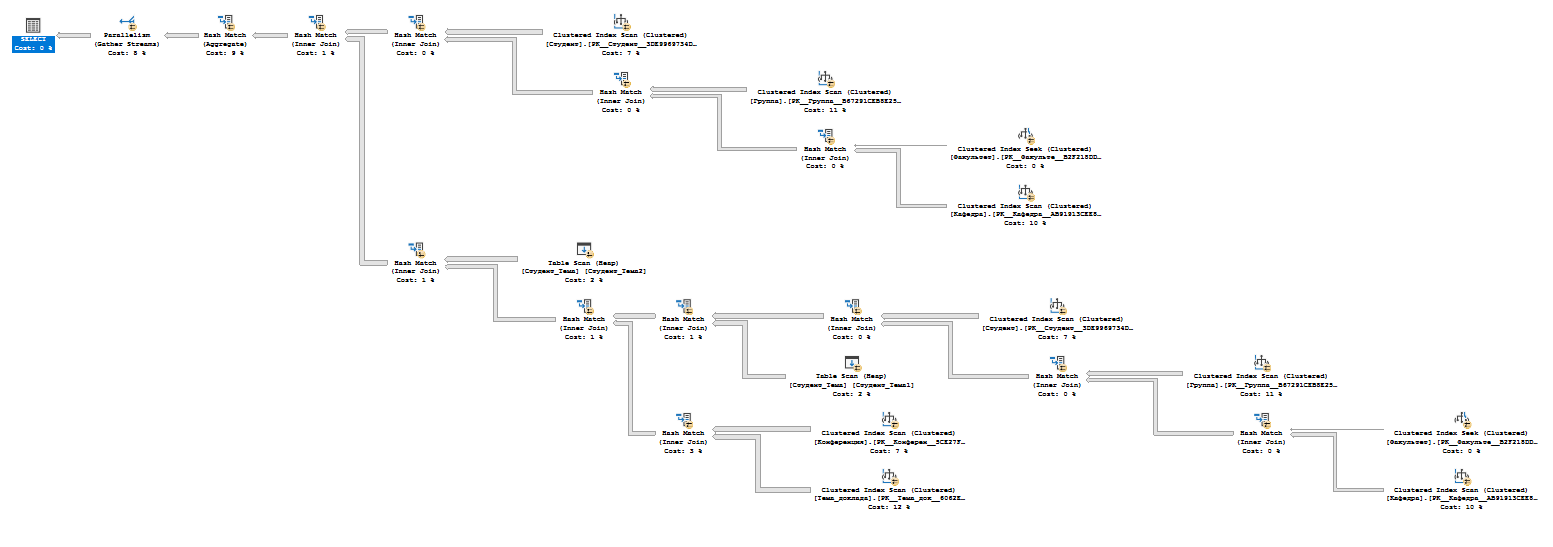


Рисунок 10 – Предполагаемый план запроса

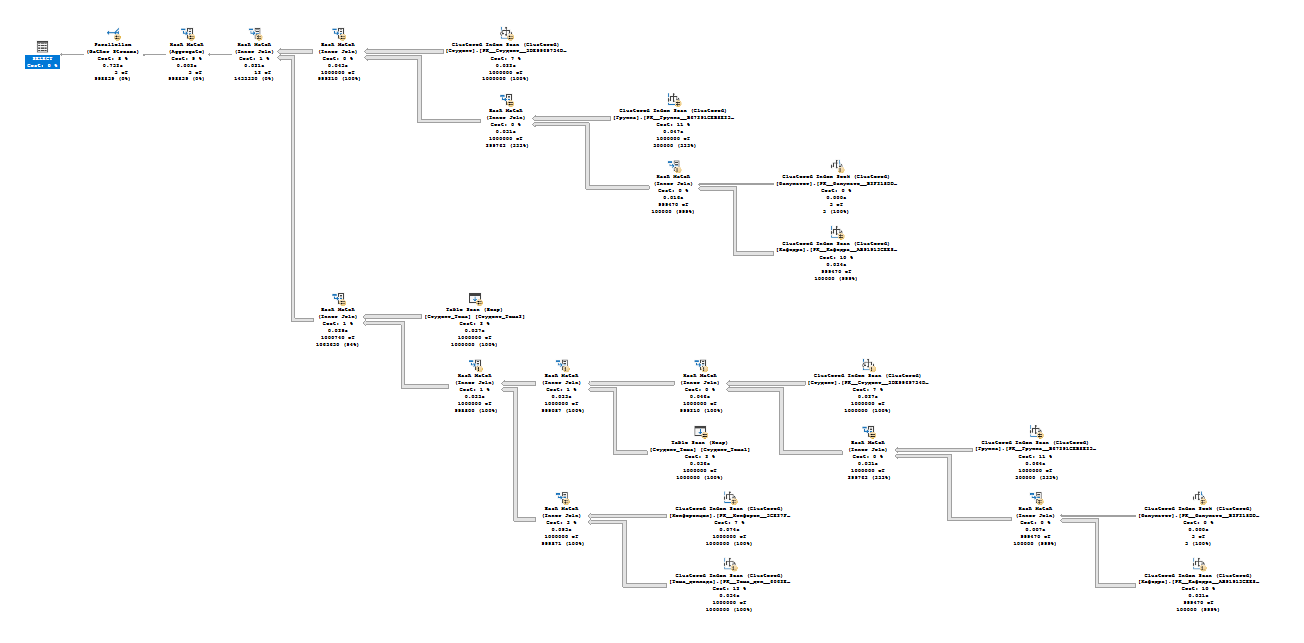


Рисунок 11 – Действительный план запроса

Время выполнения запроса с созданными индексами составило 6029 мс, индексы эффективны для этого запроса. На рисунках 12-13 показаны планы запросов с индексами.

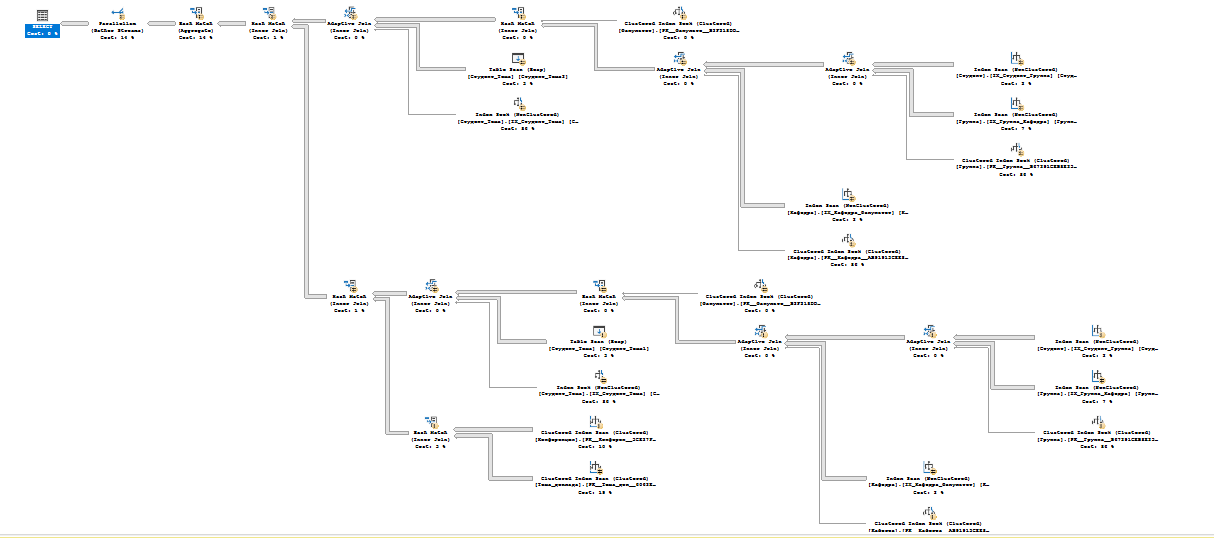


Рисунок 12 – Предполагаемый план запроса с индексами

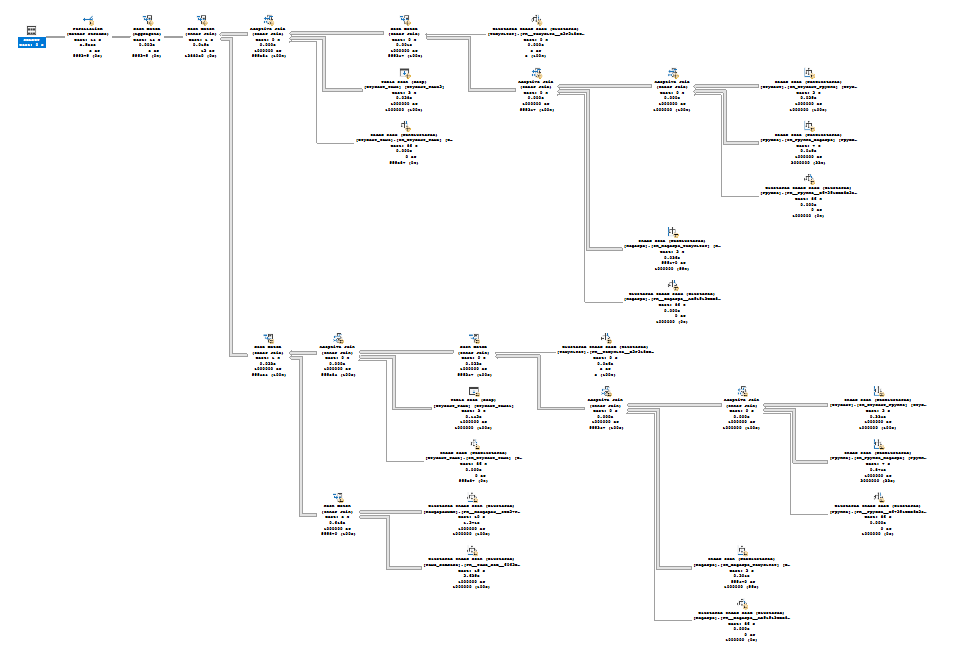


Рисунок 13 – Действительный план запроса с индексами

Четвертый запрос возвращает количество докладов для каждой конференции. Без использования индексов запрос выполнился за 7639 мс. На рисунках 14-15 представлены планы выполнения.

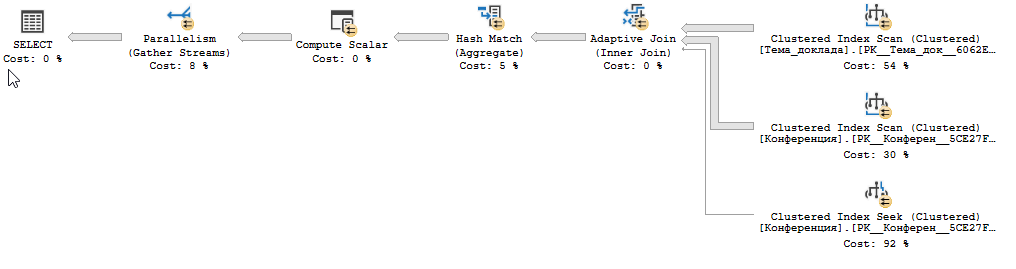


Рисунок 14 – Предполагаемый план выполнения запроса

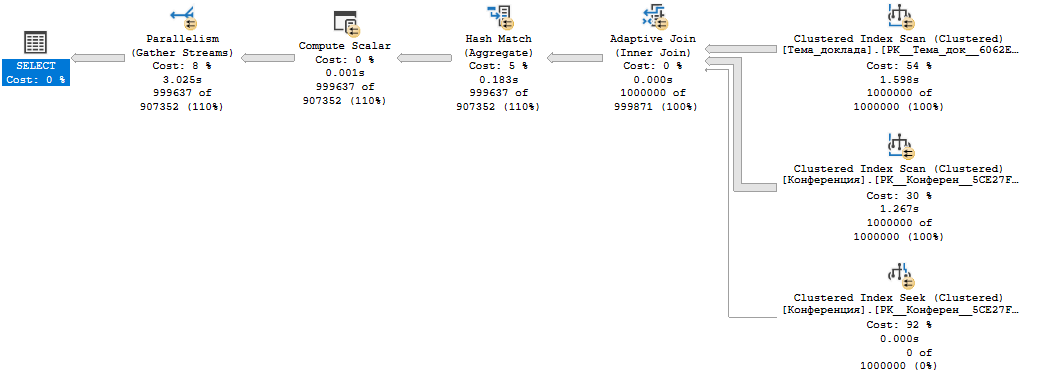


Рисунок 15 – Действительный план выполнения запроса

После создания индекса время выполнения запроса уменьшилось и составило 5790 мс. На рисунках 16-17 представлены планы запросов.

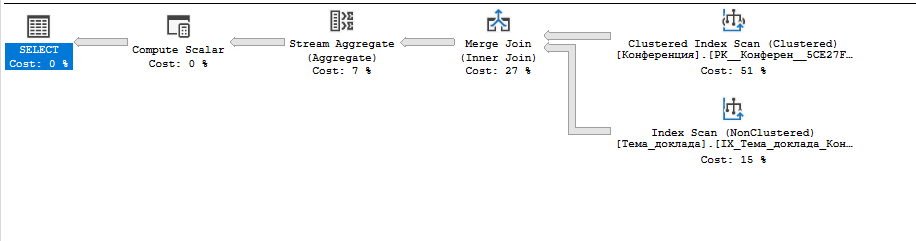


Рисунок 16 – Предполагаемый план запроса с индексом

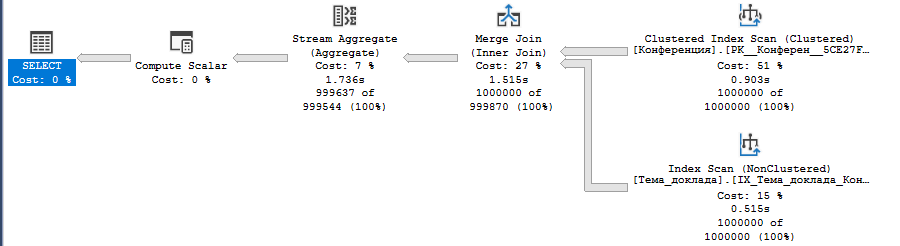


Рисунок 17 – Действительный план запроса с индексом

Пятый запрос показывает среднее количество докладов, сделанных студентами третьего факультета на конференциях. Без индексов запрос выполняется за 5659 мс. Планы запросов показаны на рисунках 18-19.

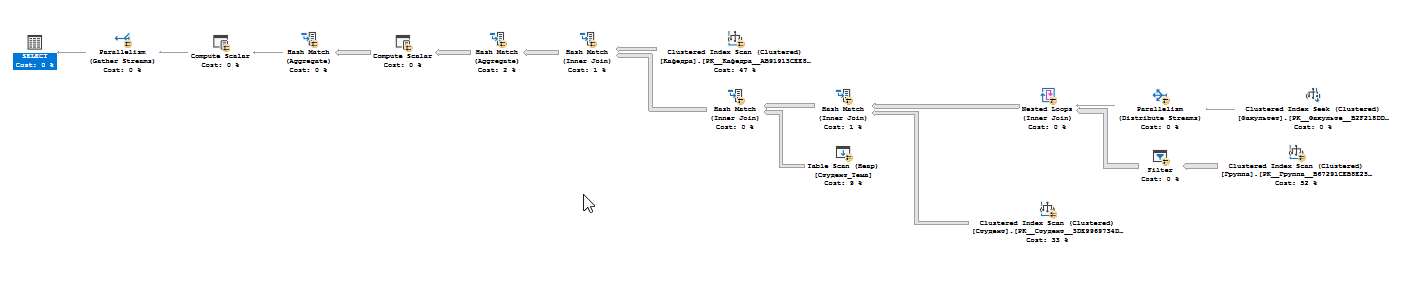


Рисунок 18 – Предполагаемый план запроса

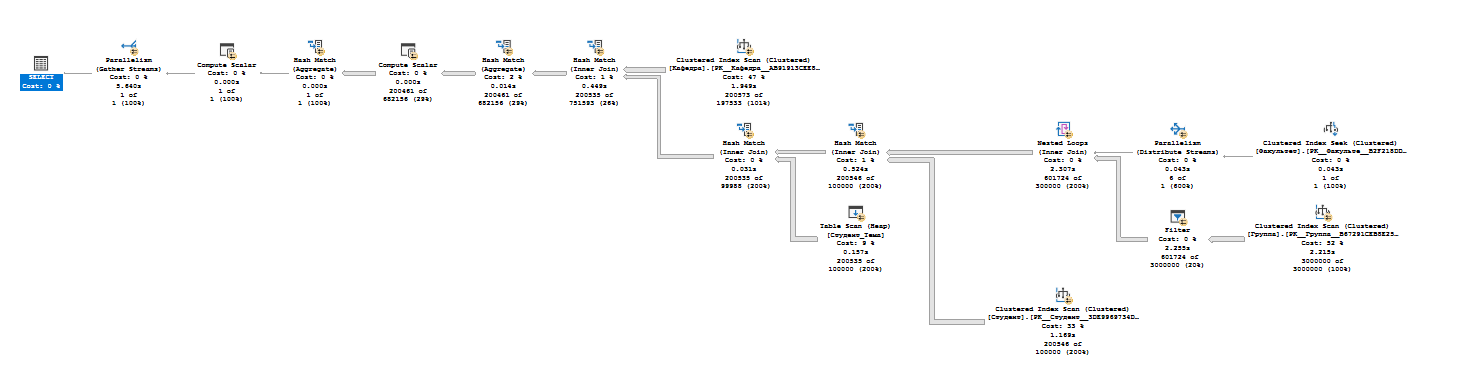


Рисунок 19 – Действительный план запроса

После создания индексов выполняется за 1721 мс. На рисунках 20-21 показаны планы запросов с индексами.

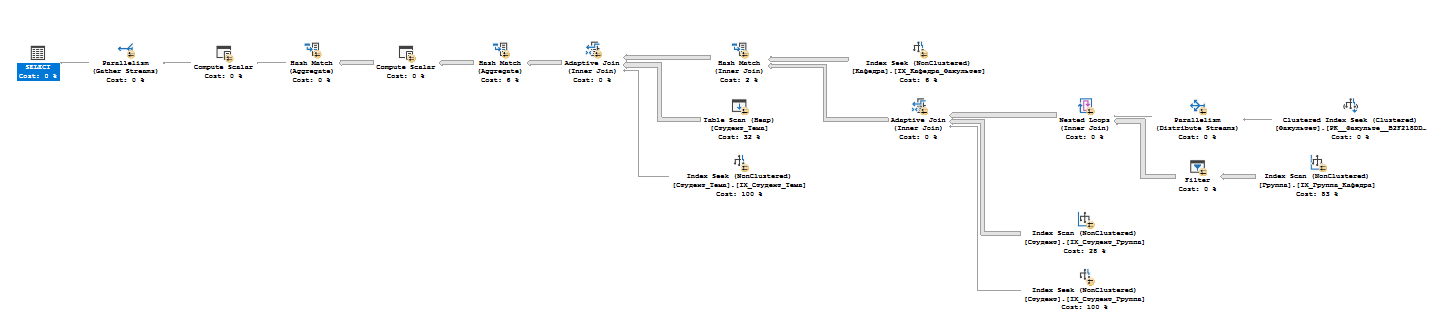


Рисунок 20 – Предполагаемый план запроса с индексами

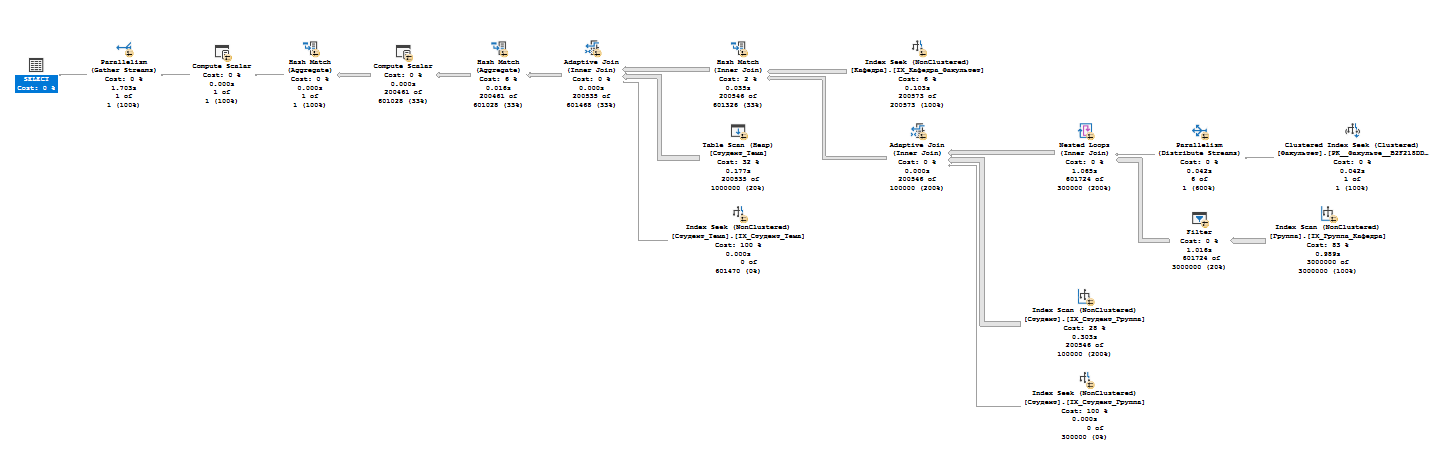


Рисунок 21 – Действительный план запроса с индексами

В шестом запросе нужно было выбрать студентов, выступивших на трех или большем числе конференций. Запрос выполняется за 4986 мс, на рисунках 22-23 представлены планы запросов.

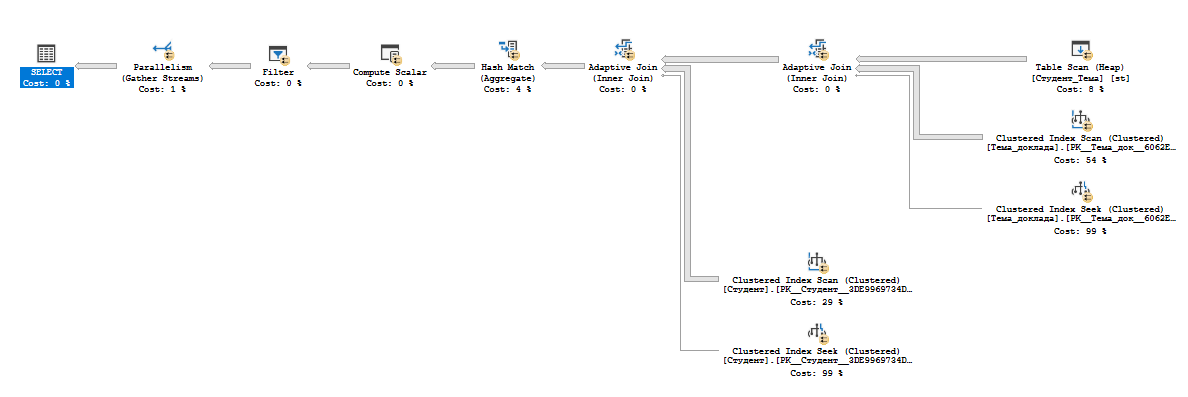


Рисунок 22 – Предполагаемый план запроса

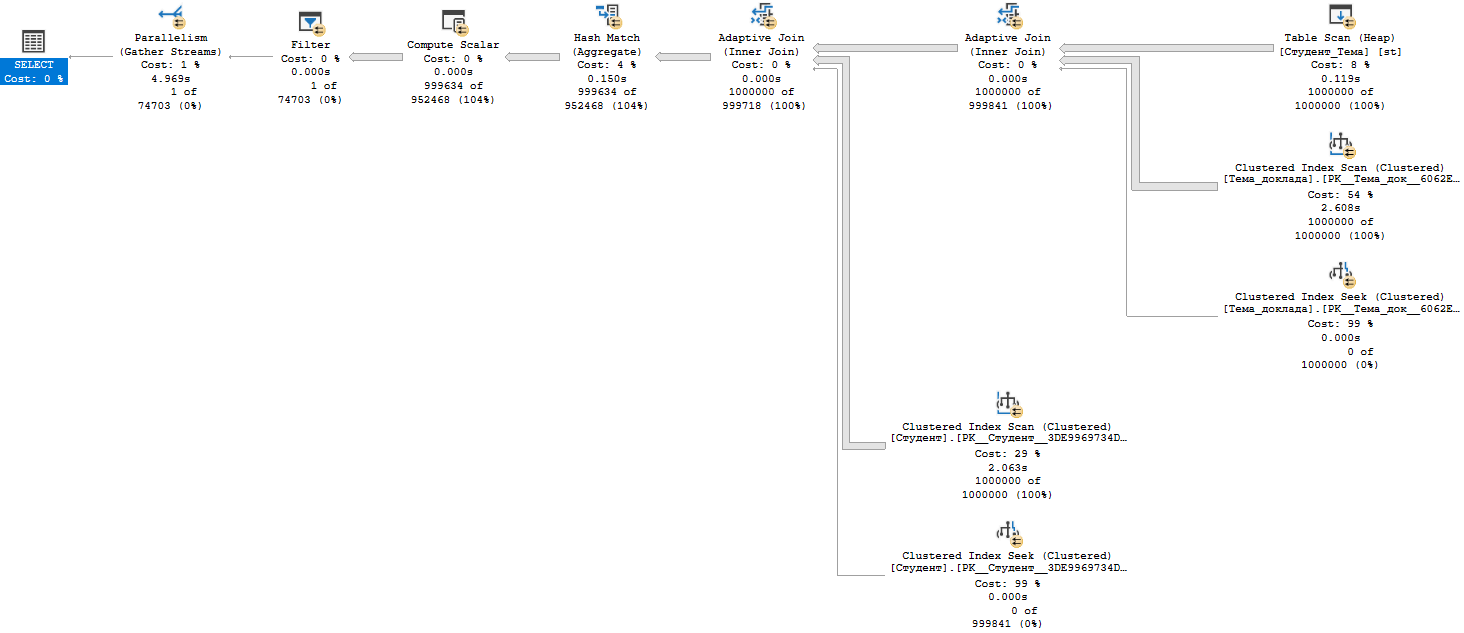


Рисунок 23 – Действительный план запроса

С добавлением индексов общее время выполнения запроса составило 3982 мс. На рисунках 24-25 представлены планы выполнения запроса с индексом.

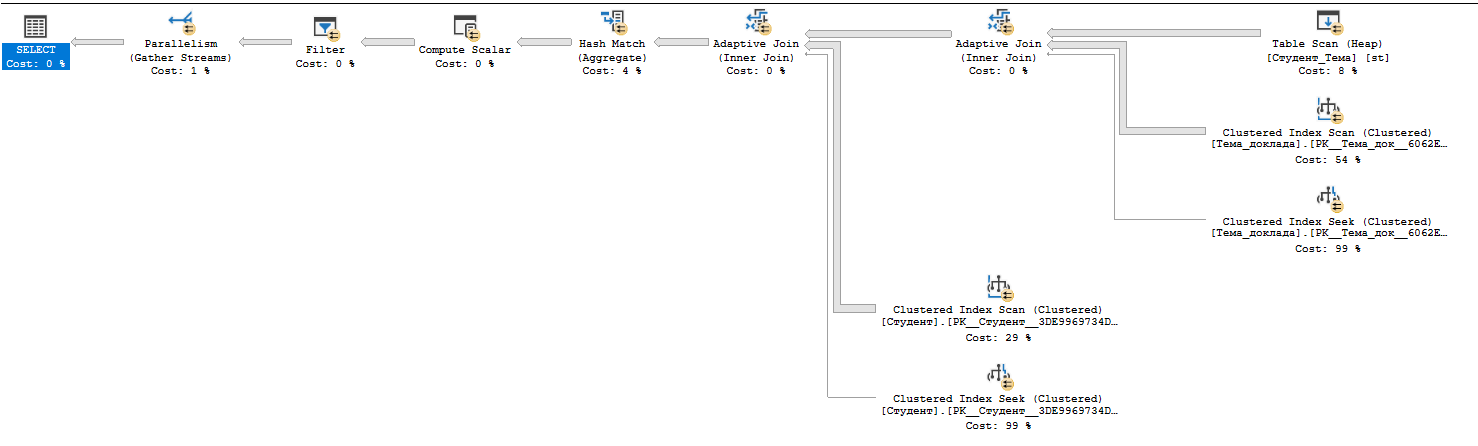


Рисунок 24 – Предполагаемый план запроса с индексом

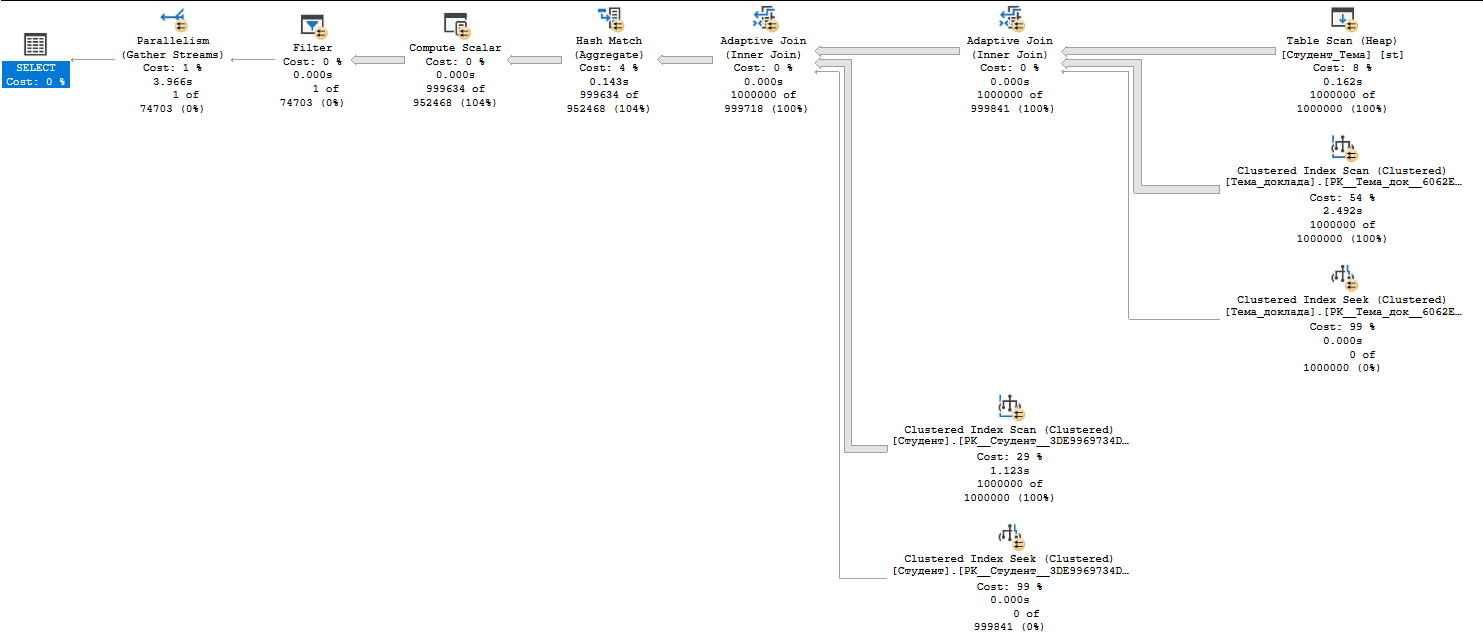


Рисунок 25 – Действительный план запроса с индексом

Седьмой запрос выводит на экран студентов четвертого факультета, не выступавших на конференциях. Время выполнения составило 5519 мс. На рисунках 26-27 представлены планы выполнения запросов.

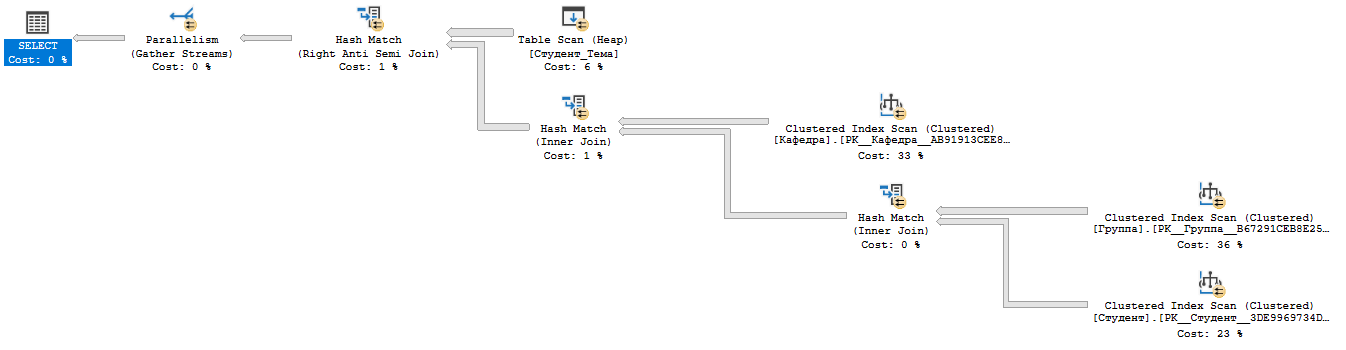


Рисунок 26 – Предполагаемый план запроса

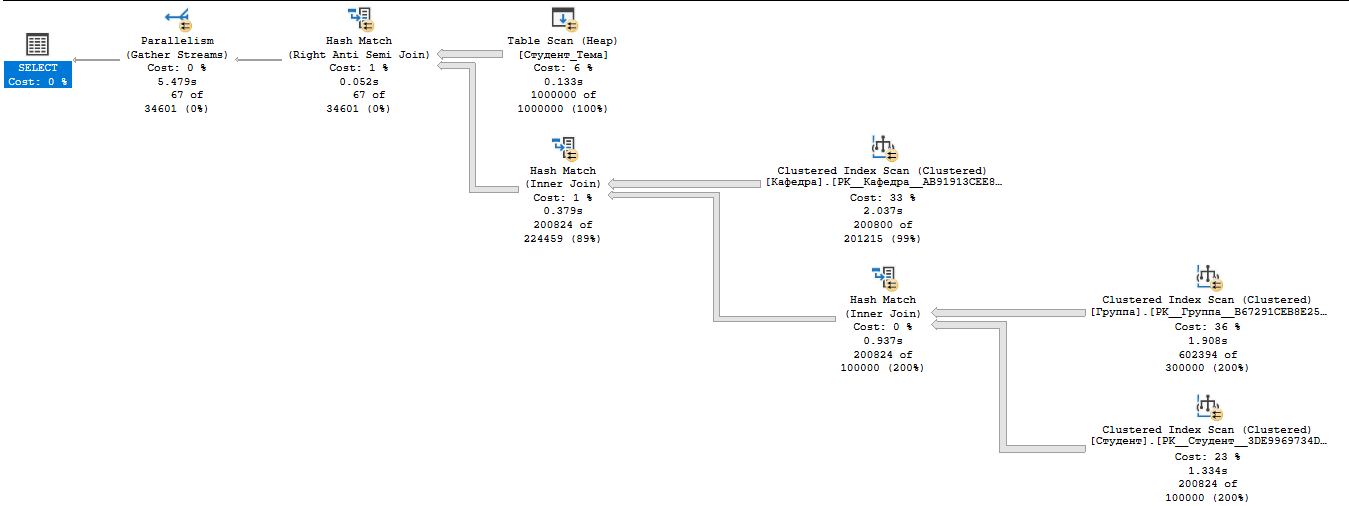


Рисунок 27 – Действительный план запроса

Время выполнения этого запроса с индексом составило 2593 мс. На рисунках 28-29 показаны планы выполнения запросов с индексом.

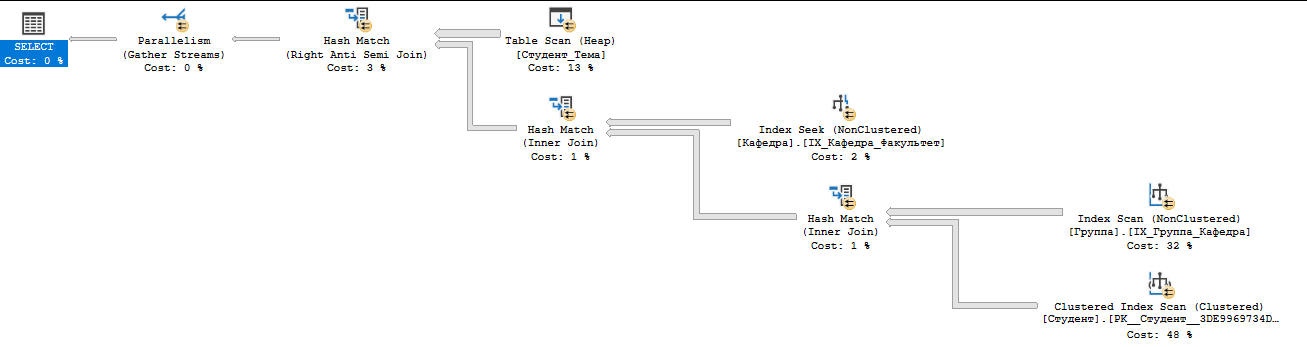


Рисунок 28 – Предполагаемый план выполнения с индексом

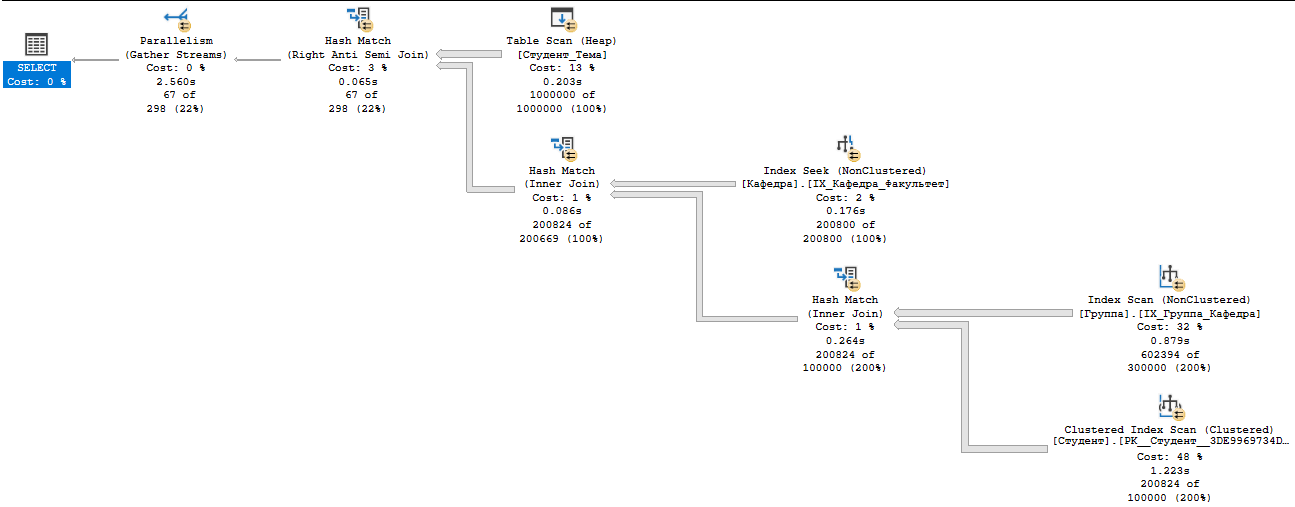


Рисунок 29 – Действительный план выполнения с индексом

В восьмом запросе нужно было найти студентов, выступивших на всех конференциях. Время выполнения без использования индекса составляет 17760 мс. На рисунках 30-31 представлены планы запросов.

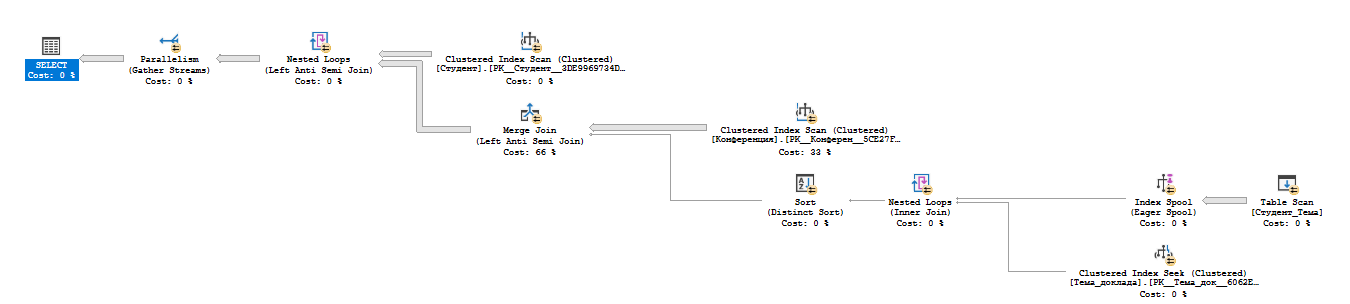


Рисунок 30 – Предполагаемый план запроса

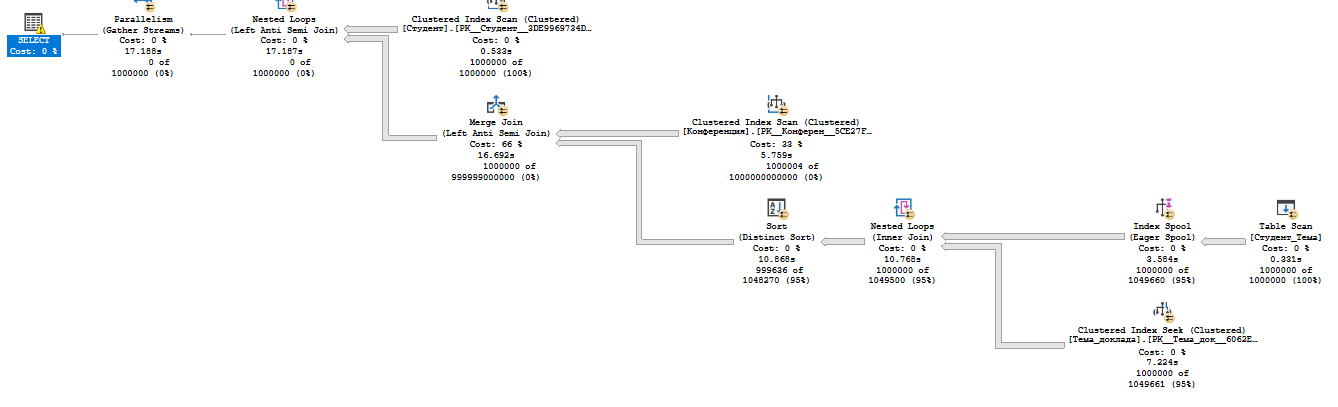


Рисунок 31 – Действительный план запроса

С добавлением индексов общее время выполнения запроса составило 9839 мс, следовательно, можно сказать индекс эффективен. На рисунках 32-33 представлены планы выполнения запроса с индексом.

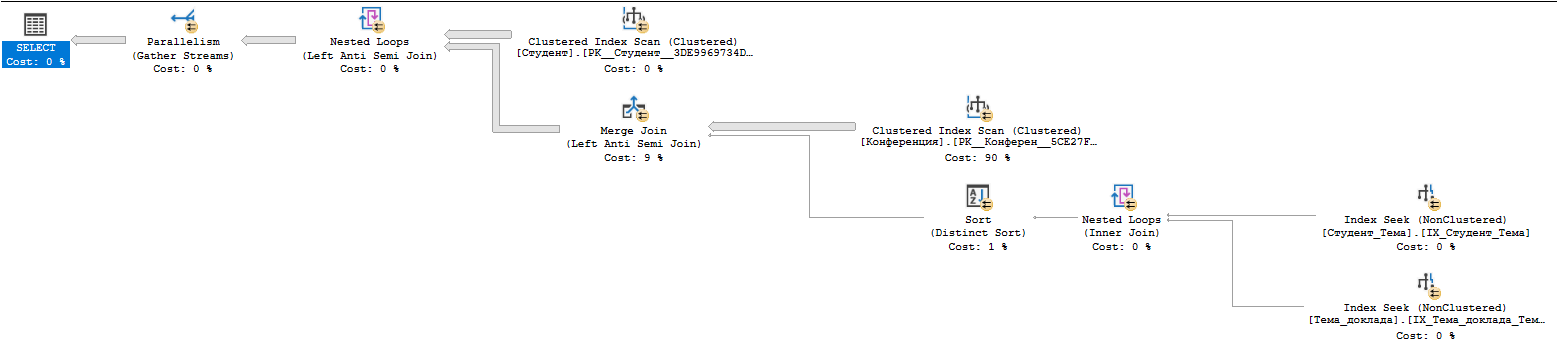


Рисунок 32 – Предполагаемый план запроса с индексом

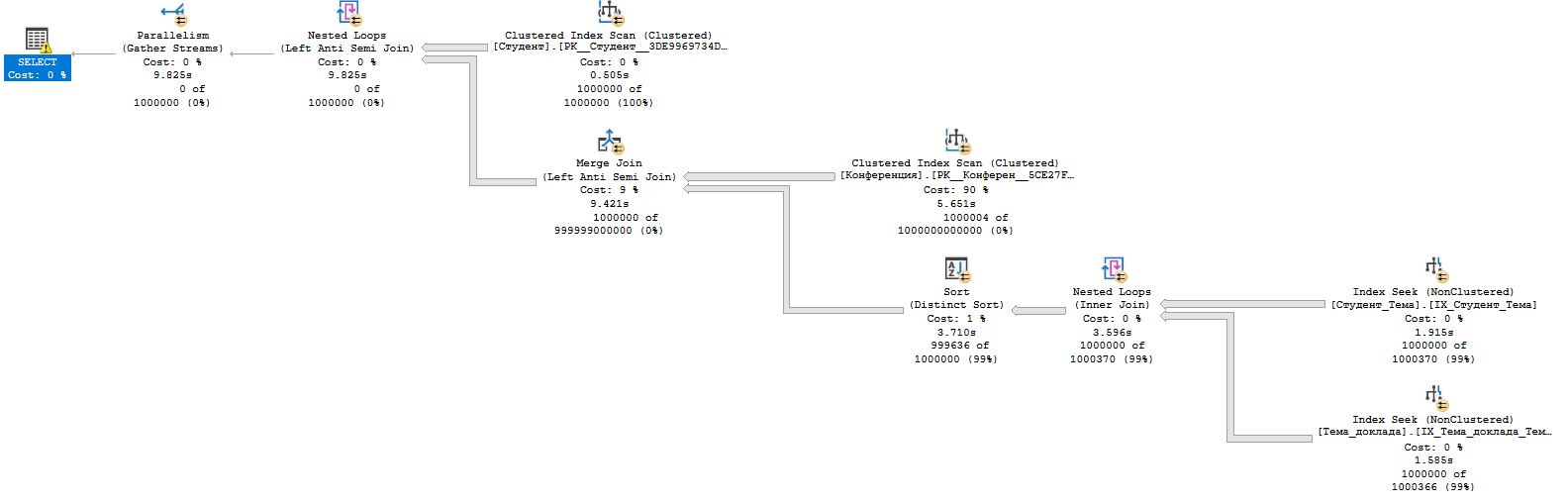


Рисунок 33 – Действительный план запроса с индексом

В девятом запросе необходимо вывести пары студентов, всегда выступающие вместе. Планы запросов представлены на рисунках 34-35, время выполнения запроса без использования индекса составляет 27292 мс

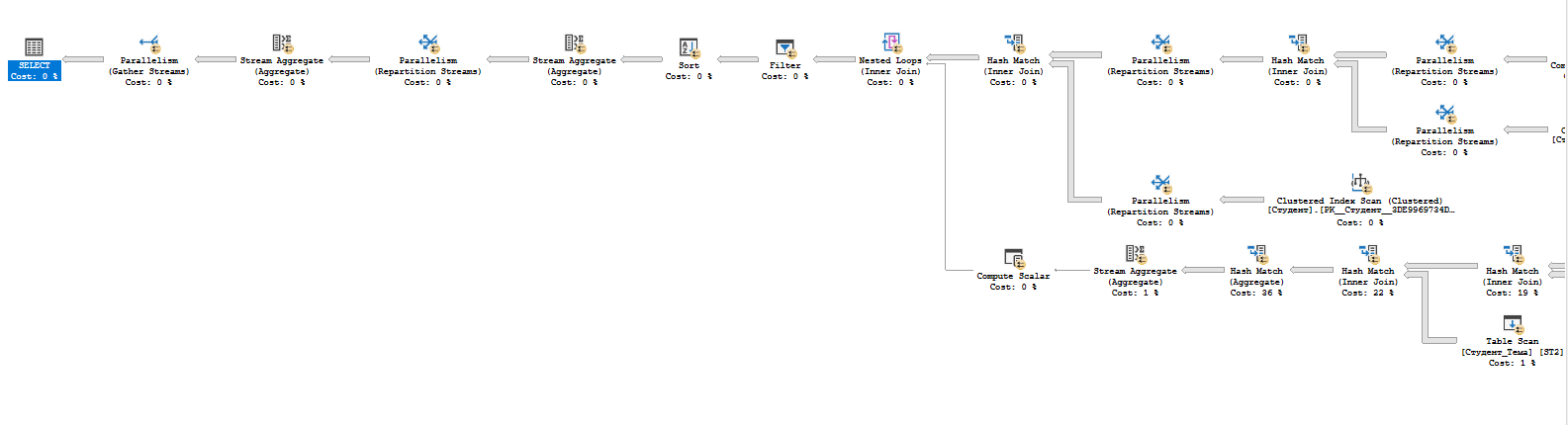


Рисунок 34 – Предполагаемый план запроса

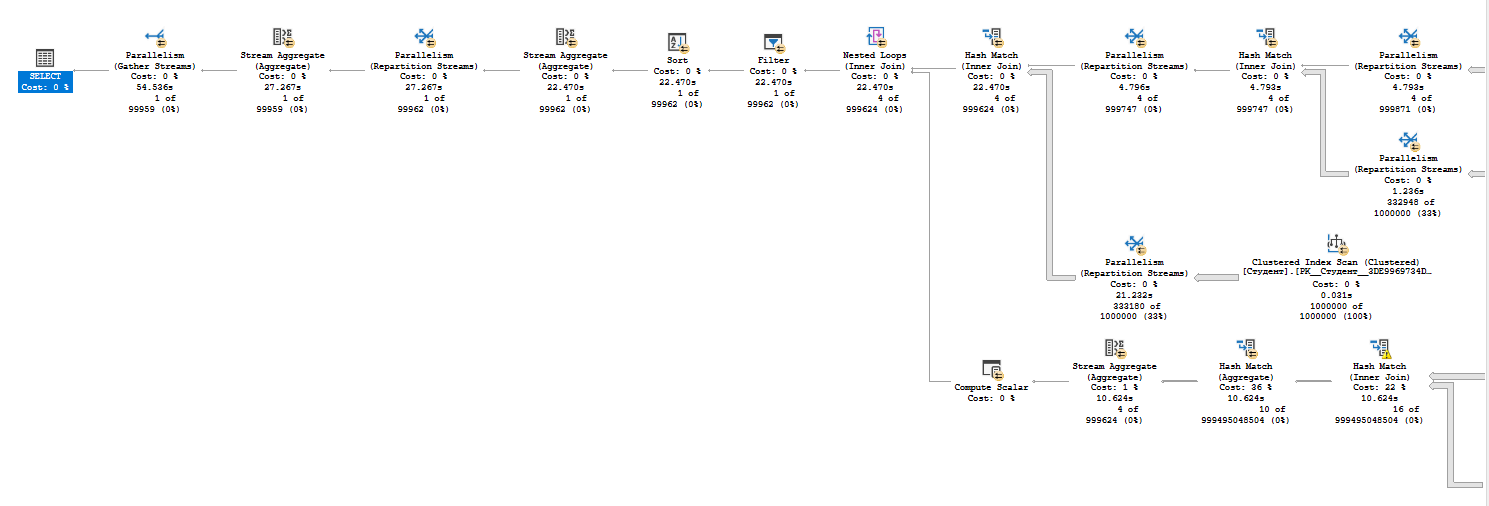


Рисунок 35 – Действительный план запроса

Время выполнения этого запроса с индексом составило 18427 мс, это означает что индекс эффективен. На рисунках 36-37 показаны планы выполнения запросов с индексом.

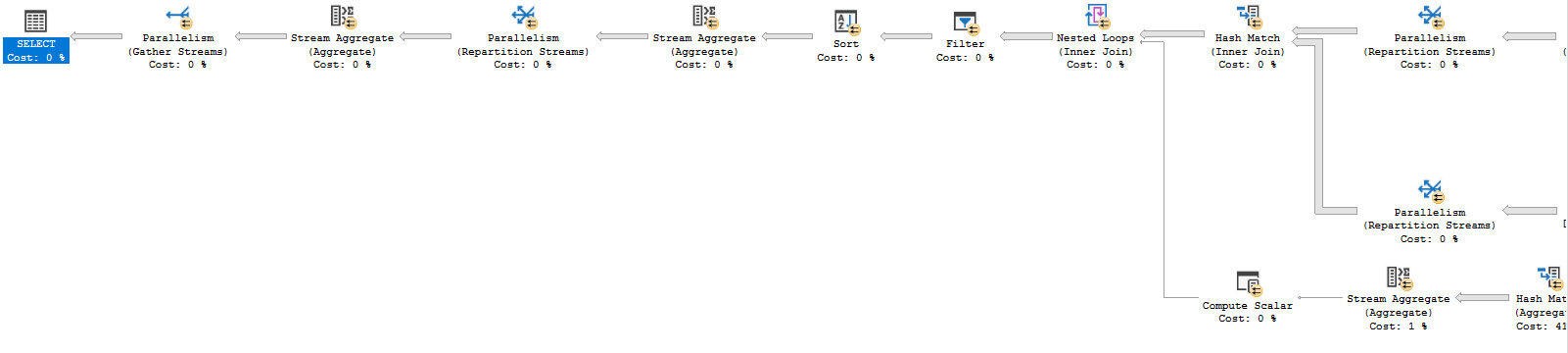


Рисунок 36 – Предполагаемый план запроса с индексом

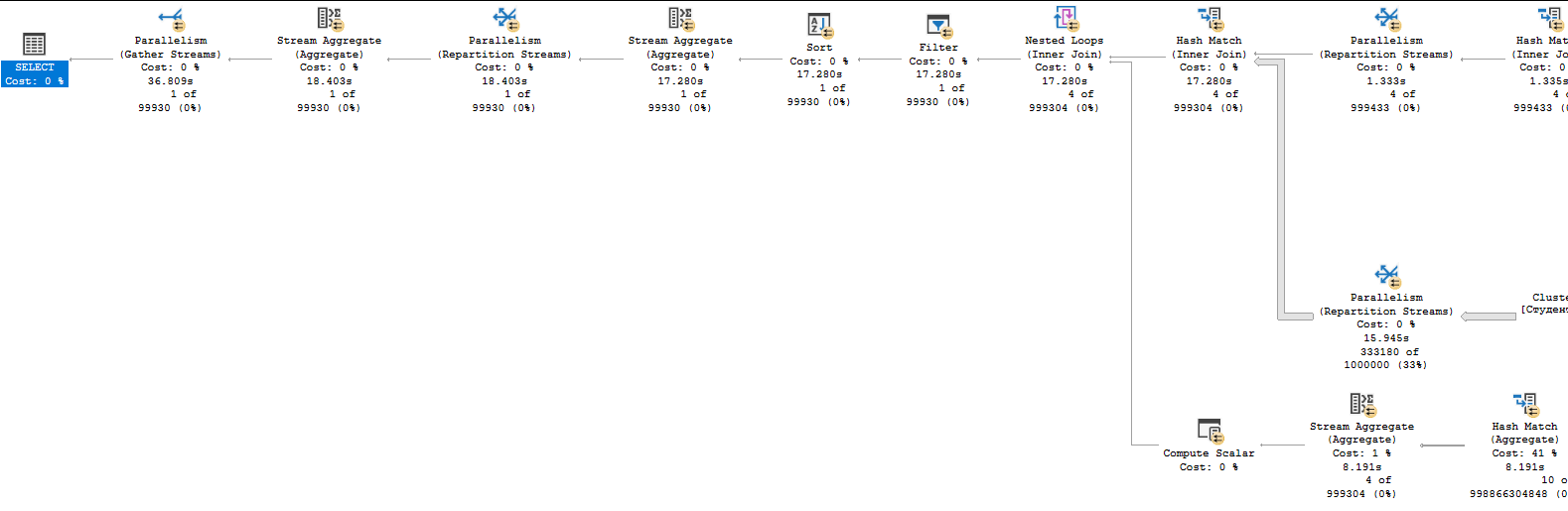


Рисунок 37 – Действительный план запроса с индексом

**Вывод**

В ходе выполнения лабораторной работы было изучено понятие индексации данных и её назначение. Были созданы индексы для выполнения запросов в собственной базе данных и проведён анализ полученных результатов. На основе анализа можно сделать вывод, что применение индексов значительно сокращает время выполнения запросов.

Приложение А

CREATE DATABASE uniDB;

GO

-- Использование созданной базы данных

USE uniDB;

GO

-- Создание таблиц

CREATE TABLE ВУЗ (

ВУЗ\_ID INT PRIMARY KEY,

Название VARCHAR(100) NOT NULL,

Адрес VARCHAR(255),

Телефон VARCHAR(20) NOT NULL,

Электронная\_почта VARCHAR(100) NOT NULL

);

CREATE TABLE Конференция (

Конференция\_ID INT PRIMARY KEY,

Название VARCHAR(100) NOT NULL,

Место VARCHAR(100) NOT NULL,

Организатор VARCHAR(100),

ВУЗ\_ID INT,

Дата DATE NOT NULL,

Время\_начала TIME,

Время\_окончания TIME,

);

CREATE TABLE Конференция\_ВУЗ (

Конференция\_ID INT not null,

ВУЗ\_ID INT not null,

FOREIGN KEY (Конференция\_ID) REFERENCES Конференция(Конференция\_ID),

FOREIGN KEY (ВУЗ\_ID) REFERENCES ВУЗ(ВУЗ\_ID)

);

CREATE TABLE Факультет (

Факультет\_ID INT PRIMARY KEY,

Название VARCHAR(50) NOT NULL,

Декан VARCHAR(50) NOT NULL,

Телефон VARCHAR(20) NOT NULL,

Электронная\_почта VARCHAR(100),

ВУЗ\_ID INT not null,

FOREIGN KEY (ВУЗ\_ID) REFERENCES ВУЗ(ВУЗ\_ID) ON DELETE NO ACTION ON UPDATE CASCADE

);

CREATE TABLE Кафедра (

Кафедра\_ID INT PRIMARY KEY,

Название VARCHAR(100) NOT NULL,

Заведующий VARCHAR(50) NOT NULL,

Телефон VARCHAR(20) NOT NULL,

Электронная\_почта VARCHAR(100),

Факультет\_ID INT NOT NULL CHECK (Факультет\_ID IN (1, 2, 3, 4, 5)),

FOREIGN KEY (Факультет\_ID) REFERENCES Факультет(Факультет\_ID)

);

CREATE TABLE Группа (

Группа\_ID INT PRIMARY KEY,

Номер VARCHAR(20) NOT NULL,

Специальность VARCHAR(50),

Курс INT,

Кафедра\_ID INT NOT NULL,

FOREIGN KEY (Кафедра\_ID) REFERENCES Кафедра(Кафедра\_ID) ON DELETE NO ACTION

);

CREATE TABLE Тема\_доклада (

Тема\_ID INT PRIMARY KEY,

Конференция\_ID INT NOT NULL,

Тема VARCHAR(200) NOT NULL,

Описание VARCHAR(200),

FOREIGN KEY (Конференция\_ID) REFERENCES Конференция(Конференция\_ID)

);

CREATE TABLE Студент (

Студент\_ID INT PRIMARY KEY,

ФИО VARCHAR(50),

Сер\_Ном\_Пасп CHAR(12) NOT NULL UNIQUE,

Гражданство VARCHAR(50) DEFAULT 'Российская Федерация',

Адрес VARCHAR(250),

Группа\_ID INT not null,

FOREIGN KEY (Группа\_ID) REFERENCES Группа(Группа\_ID) ON DELETE CASCADE

);

CREATE TABLE Студент\_Тема (

Студент\_ID INT not null,

Тема\_ID INT not null,

FOREIGN KEY (Студент\_ID) REFERENCES Студент(Студент\_ID) ON DELETE CASCADE,

FOREIGN KEY (Тема\_ID) REFERENCES Тема\_доклада(Тема\_ID) ON DELETE CASCADE

);

SELECT \* FROM dbo.Студент WHERE Студент\_ID =

SELECT \* FROM dbo.Конференция WHERE Конференция\_ID = 655

SELECT \* FROM dbo.Факультет;

SELECT \* FROM dbo.Тема\_доклада WHERE Тема\_ID = 496

SELECT \* FROM dbo.Студент\_Тема WHERE Студент\_ID = 516

SELECT \* FROM dbo.Кафедра;

SELECT \* FROM dbo.Конференция\_ВУЗ WHERE ВУЗ\_ID = 655

SELECT \* FROM dbo.ВУЗ WHERE ВУЗ\_ID = 1485

SET STATISTICS TIME ON;

-------------1 ЗАПРОС--------------------------------------------

-- Основной запрос

SELECT DISTINCT Студент.\*

FROM Студент

INNER JOIN Группа ON Студент.Группа\_ID = Группа.Группа\_ID

INNER JOIN Кафедра ON Группа.Кафедра\_ID = Кафедра.Кафедра\_ID

INNER JOIN Факультет ON Кафедра.Факультет\_ID = Факультет.Факультет\_ID

INNER JOIN Студент\_Тема ON Студент.Студент\_ID = Студент\_Тема.Студент\_ID

INNER JOIN Тема\_доклада ON Студент\_Тема.Тема\_ID = Тема\_доклада.Тема\_ID

INNER JOIN Конференция ON Тема\_доклада.Конференция\_ID = Конференция.Конференция\_ID

WHERE Факультет.Факультет\_ID = 1

AND Конференция.Название = 'Technical';

SELECT DISTINCT Студент.\*

FROM Студент WITH (INDEX(IX\_Студент\_Группа)) -- Используем индекс на поле Группа\_ID

INNER JOIN Группа WITH (INDEX(IX\_Группа\_Кафедра)) ON Студент.Группа\_ID = Группа.Группа\_ID

INNER JOIN Кафедра WITH (INDEX(IX\_Кафедра\_Факультет)) ON Группа.Кафедра\_ID = Кафедра.Кафедра\_ID

INNER JOIN Факультет ON Кафедра.Факультет\_ID = Факультет.Факультет\_ID

INNER JOIN Студент\_Тема WITH (INDEX(IX\_Студент\_Тема)) ON Студент.Студент\_ID = Студент\_Тема.Студент\_ID

INNER JOIN Тема\_доклада WITH (INDEX(IX\_Тема\_доклада\_Конференция)) ON Студент\_Тема.Тема\_ID = Тема\_доклада.Тема\_ID

INNER JOIN Конференция WITH (INDEX(IX\_Конференция\_Факультет)) ON Тема\_доклада.Конференция\_ID = Конференция.Конференция\_ID

WHERE Факультет.Факультет\_ID = 1

AND Конференция.Название = 'Technical';

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

checkpoint

go

dbcc freeproccache

go

dbcc freesystemcache ('all')

go

dbcc dropcleanbuffers

go

----------2 ЗАПРОС------------------------------------------------------

SELECT Тема\_доклада.\*

FROM Тема\_доклада

INNER JOIN Студент\_Тема ON Тема\_доклада.Тема\_ID = Студент\_Тема.Тема\_ID

INNER JOIN Студент ON Студент\_Тема.Студент\_ID = Студент.Студент\_ID

INNER JOIN Группа ON Студент.Группа\_ID = Группа.Группа\_ID

WHERE Группа.Номер = '111';

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

----------------------3 ЗАПРОС-----------------------------------------------------------

SELECT DISTINCT Конференция.\*, Тема\_доклада.\*, Студент\_Тема1.\*

FROM Студент\_Тема AS Студент\_Тема1

INNER JOIN Студент AS Студент1 ON Студент\_Тема1.Студент\_ID = Студент1.Студент\_ID

INNER JOIN Группа AS Группа1 ON Студент1.Группа\_ID = Группа1.Группа\_ID

INNER JOIN Кафедра AS Кафедра1 ON Группа1.Кафедра\_ID = Кафедра1.Кафедра\_ID

INNER JOIN Факультет AS Факультет1 ON Кафедра1.Факультет\_ID = Факультет1.Факультет\_ID

INNER JOIN Тема\_доклада ON Студент\_Тема1.Тема\_ID = Тема\_доклада.Тема\_ID

INNER JOIN Конференция ON Тема\_доклада.Конференция\_ID = Конференция.Конференция\_ID

INNER JOIN Студент\_Тема AS Студент\_Тема2 ON Студент\_Тема1.Тема\_ID = Студент\_Тема2.Тема\_ID

INNER JOIN Студент AS Студент2 ON Студент\_Тема2.Студент\_ID = Студент2.Студент\_ID

INNER JOIN Группа AS Группа2 ON Студент2.Группа\_ID = Группа2.Группа\_ID

INNER JOIN Кафедра AS Кафедра2 ON Группа2.Кафедра\_ID = Кафедра2.Кафедра\_ID

INNER JOIN Факультет AS Факультет2 ON Кафедра2.Факультет\_ID = Факультет2.Факультет\_ID

WHERE Факультет1.Факультет\_ID != Факультет2.Факультет\_ID;

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

---------------------------------------4 ЗАПРОС-----------------------------------------------------------

-- Этот запрос выбирает идентификатор конференции, её название и количество докладов, связанных с этой конференцией

SELECT

Тема\_доклада.Конференция\_ID, -- Выбираем идентификатор конференции из таблицы "Тема\_доклада"

Конференция.Название AS Наименование\_конференции, -- Выбираем название конференции из таблицы "Конференция" и называем его "Наименование\_конференции"

COUNT(\*) AS Количество\_докладов -- Считаем количество докладов для каждой конференции и называем это "Количество\_докладов"

FROM

Тема\_доклада -- Таблица, содержащая информацию о темах докладов

JOIN

Конференция ON Тема\_доклада.Конференция\_ID = Конференция.Конференция\_ID -- Объединяем таблицы по идентификатору конференции

GROUP BY

Тема\_доклада.Конференция\_ID, Конференция.Название; -- Группируем результаты по идентификатору конференции и названию конференции

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

----------------------------------------------------5 ЗАПРОС----------------------------------------------------------------------------------

-- Этот запрос вычисляет среднее количество докладов, сделанных студентами третьего факультета.

SELECT

AVG(Количество\_докладов) AS Среднее\_количество\_докладов -- Выбираем среднее количество докладов и называем его "Среднее\_количество\_докладов"

FROM

(

-- Подзапрос: здесь мы считаем количество докладов для каждого студента третьего факультета

SELECT

COUNT(\*) AS Количество\_докладов -- Считаем количество записей (докладов) для каждого студента и называем это "Количество\_докладов"

FROM

Студент\_Тема -- Таблица, содержащая информацию о докладах студентов

JOIN

Студент ON Студент\_Тема.Студент\_ID = Студент.Студент\_ID -- Объединяем таблицы по идентификатору студента

JOIN

Группа ON Студент.Группа\_ID = Группа.Группа\_ID -- Объединяем таблицы по идентификатору группы студента

JOIN

Кафедра ON Группа.Кафедра\_ID = Кафедра.Кафедра\_ID -- Объединяем таблицы по идентификатору кафедры группы

JOIN

Факультет ON Кафедра.Факультет\_ID = Факультет.Факультет\_ID -- Объединяем таблицы по идентификатору факультета кафедры

WHERE

Факультет.Факультет\_ID = 3 -- Ограничиваем результаты только студентами третьего факультета

GROUP BY

Студент.Студент\_ID -- Группируем результаты по идентификатору студента, чтобы посчитать количество докладов для каждого студента

) AS Количество\_докладов\_студентов\_третьего\_факультета; -- Даем имя подзапросу для дальнейшей ссылки

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

----------------------------------6 ЗАПРОС---------------------------------------------------------

-- Этот запрос выбирает идентификатор и ФИО студентов, которые принимали участие в трех или более конференциях.

SELECT

Студент\_ID, -- Выбираем идентификатор студента

ФИО -- Выбираем ФИО студента

FROM

(

-- Подзапрос: здесь мы считаем количество уникальных конференций, в которых участвовал каждый студент

SELECT

st.Студент\_ID, -- Выбираем идентификатор студента из таблицы "Студент\_Тема"

s.ФИО, -- Выбираем ФИО студента из таблицы "Студент"

COUNT(DISTINCT td.Конференция\_ID) AS Количество\_конференций -- Считаем количество уникальных конференций, в которых участвовал каждый студент, и называем это "Количество\_конференций"

FROM

Студент\_Тема st -- Таблица, содержащая информацию о докладах студентов

JOIN

Студент s ON st.Студент\_ID = s.Студент\_ID -- Объединяем таблицы по идентификатору студента

JOIN

Тема\_доклада td ON st.Тема\_ID = td.Тема\_ID -- Объединяем таблицы по идентификатору темы доклада

GROUP BY

st.Студент\_ID, s.ФИО -- Группируем результаты по идентификатору студента и ФИО

) AS sub -- Даем имя подзапросу для дальнейшей ссылки

WHERE

Количество\_конференций >= 3; -- Ограничиваем результаты только студентами, участвовавшими в трех или более конференциях

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

----------------------------------------------7 ЗАПРОС--------------------------------------

--студентов четвертого факультета, не выступавших на конференциях;

SELECT Студент\_ID, ФИО

FROM Студент

WHERE Группа\_ID IN (

SELECT Группа\_ID

FROM Группа

WHERE Кафедра\_ID IN (

SELECT Кафедра\_ID

FROM Кафедра

WHERE Факультет\_ID = 4

)

)

AND NOT EXISTS (

SELECT \*

FROM Студент\_Тема

WHERE Студент\_Тема.Студент\_ID = Студент.Студент\_ID

);

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

-------------------------------------------------8 ЗАПРОС-------------------------------

--студентов, выступивших на всех конференциях;

SELECT DISTINCT Студент\_ID, ФИО

FROM Студент

WHERE NOT EXISTS (

SELECT \*

FROM Конференция

WHERE NOT EXISTS (

SELECT \*

FROM Студент\_Тема

WHERE Студент\_Тема.Студент\_ID = Студент.Студент\_ID

AND Студент\_Тема.Тема\_ID IN (

SELECT Тема\_ID

FROM Тема\_доклада

WHERE Тема\_доклада.Конференция\_ID = Конференция.Конференция\_ID

)

)

);

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

-------------------------------------9 ЗАПРОС--------------------------------------------------------

--пары студентов, всегда выступающие вместе.

WITH Пары\_Студентов AS (

SELECT DISTINCT ST1.Студент\_ID AS Студент1\_ID,

ST2.Студент\_ID AS Студент2\_ID,

К.Конференция\_ID

FROM Студент\_Тема AS ST1

INNER JOIN Студент\_Тема AS ST2 ON ST1.Тема\_ID = ST2.Тема\_ID AND ST1.Студент\_ID < ST2.Студент\_ID

INNER JOIN Тема\_доклада AS ТД ON ST1.Тема\_ID = ТД.Тема\_ID

INNER JOIN Конференция AS К ON ТД.Конференция\_ID = К.Конференция\_ID

),

Количество\_Конференций AS (

SELECT Студент1\_ID, Студент2\_ID, COUNT(DISTINCT Конференция\_ID) AS Количество\_Конференций

FROM Пары\_Студентов

GROUP BY Студент1\_ID, Студент2\_ID

),

Пары\_с\_единственными\_выступлениями AS (

SELECT Студент1\_ID, Студент2\_ID

FROM Количество\_Конференций

WHERE Количество\_Конференций = (

SELECT COUNT(\*)

FROM Пары\_Студентов AS П

WHERE (П.Студент1\_ID = Количество\_Конференций.Студент1\_ID OR

П.Студент2\_ID = Количество\_Конференций.Студент2\_ID)

)

)

SELECT DISTINCT S1.ФИО AS Студент1, S2.ФИО AS Студент2

FROM Пары\_с\_единственными\_выступлениями AS ПЕ

INNER JOIN Студент AS S1 ON ПЕ.Студент1\_ID = S1.Студент\_ID

INNER JOIN Студент AS S2 ON ПЕ.Студент2\_ID = S2.Студент\_ID

ORDER BY Студент1, Студент2;

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

-- Вывести все индексы в текущей базе данных

SELECT

OBJECT\_NAME(OBJECT\_ID) AS TableName,

name AS IndexName,

type\_desc AS IndexType

FROM

sys.indexes

WHERE

OBJECT\_NAME(OBJECT\_ID) IN ('ВУЗ', 'Конференция', 'Конференция\_ВУЗ', 'Факультет', 'Кафедра', 'Группа', 'Тема\_доклада', 'Студент', 'Студент\_Тема' )

AND is\_primary\_key = 0

AND is\_unique\_constraint = 0 -- исключаем первичные ключи и уникальные ограничения

ORDER BY

TableName, IndexName;

-- Существующие индексы, которые остаются

CREATE INDEX IX\_Студент\_Группа ON Студент (Группа\_ID);

CREATE INDEX IX\_Группа\_Кафедра ON Группа (Кафедра\_ID);

CREATE INDEX IX\_Кафедра\_Факультет ON Кафедра (Факультет\_ID);

CREATE INDEX IX\_Студент\_Тема ON Студент\_Тема (Студент\_ID, Тема\_ID);

CREATE INDEX IX\_Тема\_доклада\_Конференция ON Тема\_доклада (Конференция\_ID);

CREATE INDEX IX\_Конференция\_Факультет ON Конференция (ВУЗ\_ID);

CREATE INDEX IX\_Группа\_Номер ON Группа (Номер);

CREATE INDEX IX\_Тема\_доклада\_Тема\_Конференция ON Тема\_доклада (Тема\_ID, Конференция\_ID)

CREATE INDEX IX\_Конференция\_ВУЗ\_Название ON Конференция (ВУЗ\_ID, Название);

DROP INDEX IX\_Группа\_Кафедра ON Группа;

DROP INDEX IX\_Кафедра\_Факультет ON Кафедра;

DROP INDEX IX\_Конференция\_Факультет ON Конференция;

DROP INDEX IX\_Студент\_Группа ON Студент;

DROP INDEX IX\_Студент\_Тема ON Студент\_Тема;

DROP INDEX IX\_Группа\_Номер ON Группа;

DROP INDEX IX\_Тема\_доклада\_Тема\_Конференция ON Тема\_доклада;

DROP INDEX IX\_Конференция\_ВУЗ\_Название ON Конференция;

DROP INDEX IX\_Тема\_доклада\_Конференция ON dbo.Тема\_доклада;