Санкт-Петербургский национальный исследовательский университет ИТМО

Факультет Инфокоммуникационных технологий

**Отчет по итоговому проекту**

Выполнили:

Студент 2 курса группы K32421

Козлов Всеволод Денисович

Студент 2 курса группы К32402

Горбатов Дмитрий Алексеевич

Санкт-Петербург 2023

Оглавление

[**Описание предметной области** 6](#_Toc136421200)

[**Выполнение** 7](#_Toc136421201)

[**Архитектура БД** 7](#_Toc136421202)

[**Описание атрибутов сущностей и ограничений на данные** 8](#_Toc136421203)

[**Запросы к базе данных** 16](#_Toc136421204)

[**Запрос №1:** 16](#_Toc136421205)

[Команда: 16](#_Toc136421206)

[Результат (рис 2.): 16](#_Toc136421207)

[**Запрос №2:** 17](#_Toc136421208)

[Команда: 17](#_Toc136421209)

[Результат(рис. 3): 17](#_Toc136421210)

[**Запрос №3:** 18](#_Toc136421211)

[Команда: 18](#_Toc136421212)

[Результат (рис 4.): 18](#_Toc136421213)

[**Запрос №4:** 19](#_Toc136421214)

[Команда: 19](#_Toc136421215)

[Результат (рис. 5): 19](#_Toc136421216)

[**Запрос №5:** 20](#_Toc136421217)

[Команда: 20](#_Toc136421218)

[Результат (рис. 6): 20](#_Toc136421219)

[**Запрос №6:** 21](#_Toc136421220)

[Команда: 21](#_Toc136421221)

[Результат (рис. 7): 21](#_Toc136421222)

[**Запрос №7:** 22](#_Toc136421223)

[Команда: 22](#_Toc136421224)

[Результат (рис. 8): 22](#_Toc136421225)

[**Создание представлений** 23](#_Toc136421226)

[**Представление №1:** 23](#_Toc136421227)

[Команда: 23](#_Toc136421228)

[Результат (рис. 9): 23](#_Toc136421229)

[**Представление №2:** 24](#_Toc136421230)

[Команда: 24](#_Toc136421231)

[Результат (рис. 10): 24](#_Toc136421232)

[**Запросы на модификацию данных (INSERT, UPDATE, DELETE) с использованием подзапросов** 25](#_Toc136421233)

[**Запрос №1 на update:** 25](#_Toc136421234)

[Команда: 25](#_Toc136421235)

[Скриншот до (рис. 11): 25](#_Toc136421236)

[Скриншот после (рис.12): 26](#_Toc136421237)

[**Запрос №2 на update:** 26](#_Toc136421238)

[Команда: 26](#_Toc136421239)

[Скриншот до (рис. 13): 27](#_Toc136421240)

[Скриншот после (рис. 14): 27](#_Toc136421241)

[**Запрос №1 на удаление:** 28](#_Toc136421242)

[Команда: 28](#_Toc136421243)

[Скриншот до (рис. 15): 28](#_Toc136421244)

[Скриншот после (рис. 16): 29](#_Toc136421245)

[**Запрос №2 на удаление:** 30](#_Toc136421246)

[Команда: 30](#_Toc136421247)

[Скриншот до (рис. 17): 30](#_Toc136421248)

[Скриншот после (рис.18): 31](#_Toc136421249)

[**Запрос №1 на вставку:** 31](#_Toc136421250)

[Команда: 31](#_Toc136421251)

[Скриншот до (рис. 19): 32](#_Toc136421252)

[Скриншот после (рис. 20): 32](#_Toc136421253)

[**Создание индексов** 33](#_Toc136421254)

[**Запрос с составным индексом:** 33](#_Toc136421255)

[Команда: 33](#_Toc136421256)

[План операции до создания индексов (рис. 21): 34](#_Toc136421257)

[Время до создания индексов: 35](#_Toc136421258)

[437ms 35](#_Toc136421259)

[**Добавление индексов:** 35](#_Toc136421260)

[create index try\_mark\_idx on record\_book(try\_number, mark); 35](#_Toc136421261)

[План операции после создания индексов (рис. 22): 35](#_Toc136421262)

[Время после создания индексов: 35](#_Toc136421263)

[**Запрос без составных индексов:** 36](#_Toc136421264)

[Команда: 36](#_Toc136421265)

[Время до создания индексов: 36](#_Toc136421266)

[План операции до создания индексов (рис. 23): 36](#_Toc136421267)

[**Добавление индексов:** 37](#_Toc136421268)

[Время после создания индексов: 37](#_Toc136421269)

[План операции после добавления индексов: 37](#_Toc136421270)

[План после добавления 10000 студентов (рис. 25): 39](#_Toc136421271)

[**Методы** 40](#_Toc136421272)

[№ . 1 40](#_Toc136421273)

[Описание: 40](#_Toc136421274)

[Код метода: 40](#_Toc136421275)

[Выполнение метода: 40](#_Toc136421276)

[№ . 2 41](#_Toc136421277)

[Описание: 41](#_Toc136421278)

[Код метода: 41](#_Toc136421279)

[Выполнение метода: 42](#_Toc136421280)

[№ 3 42](#_Toc136421281)

[Описание: 42](#_Toc136421282)

[Код метода: 42](#_Toc136421283)

[Выполнение метода: 43](#_Toc136421284)

[№4. Бонус 44](#_Toc136421285)

[Описание: 44](#_Toc136421286)

[Код метода: 44](#_Toc136421287)

[**Триггеры** 45](#_Toc136421288)

[№1 45](#_Toc136421289)

[Описание: 45](#_Toc136421290)

[Код: 45](#_Toc136421291)

[Выполнение: 46](#_Toc136421292)

[№2 47](#_Toc136421293)

[Описание: 47](#_Toc136421294)

[Код: 47](#_Toc136421295)

[Выполнение: 47](#_Toc136421296)

[**Выводы** 48](#_Toc136421297)

# **Цели проекта**

* Создать архитектуру БД, которая может быть использована для ведения учебного процесса в высшем учебном заведении.
* Продумать ограничения на столбцы в БД, чтобы поддерживать ее целостность
* Реализовать архитектуру БД, использую PostgreSQL
* Заполнить БД тестовыми данными для проверки работоспособности и дальнейшей оптимизации БД при помощи индексов
* Проверить работоспособность БД при помощи запросов на получение, вставку, изменение и удаление данных
* Реализовать представления
* Оптимизировать работу БД при помощи индексов
* Реализовать функции
* Реализовать триггеры

# **Описание предметной области**

БД содержит сведения о сдаче сессии студентами. Номер зачетной книжки однозначно идентифицирует студента. Каждый студент обучается в группе, причем номера групп меняются каждый очередной учебный год. Дисциплины, по которым студенты сдают промежуточную аттестацию, соотнесены с учебным планом образовательной программы, которая в свою очередь относится к направлению подготовки, реализуемом в определенном подразделении вуза. По каждой дисциплине могут проводиться лекционные, лабораторные/практические занятия и практика определенном объеме часов. По каждой дисциплине и практике проводится аттестация в формате экзамен/дифзачет/зачет. Одна дисциплина может соотноситься с несколькими учебными планами разных направлений подготовки. Каждый учебный план относится к определенному году приема. Экзамены проходят на различных площадках вуза, территориально расположенных в разных частях города или страны.

БД должна содержать следующий минимальный набор сведений: Номер зачетной книжки. Фамилия студента. Имя студента. Отчество студента. Курс. Группа. Учебный год. Семестр. Код дисциплины/практики. Название дисциплины/практики. Код направления. Название направления. Оценка. Фамилия преподавателя. Имя преподавателя. Отчество преподавателя.  Должность. Код подразделения. Подразделение. Дата сдачи экзамена/зачета/дифзачета. Аудитория. Площадка (адрес). Номер попытки (максимально 3).

Дополните исходные данные информацией: по расписанию сессии, по назначению базовой и повышенной стипендии.

# **Выполнение**

# **Архитектура БД**

В нотации idef1x (рис 1.). Архитектура также есть в формате PDF в архиве.

**Изображение выглядит как текст, Шрифт, диаграмма, снимок экрана

Автоматически созданное описание**

Рисунок 1 – нотация idef1x

# **Описание атрибутов сущностей и ограничений на данные**

Таблица 1 – Описание атрибутов сущностей

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Наименование атрибута** | **Тип** | **Первичный ключ** | | | | **Внешний ключ** | **Обязатель ность** | **Ограничения целостности** |
| **Собственный атрибут** | | | **Внешний ключ** |
| Subdivisions (подразделения) | | | | | | | | |
| id\_subdivision | INTEGER | | + | |  |  | + | Уникален, необходимо обеспечить автоматическую генерацию значения |
| name | VARCHAR(40) | |  | |  |  | + |  |
| headquater\_addres (адрес голового офиса) | VARCHAR(50) | |  | |  |  | + |  |
| type | VARCHAR(9) | |  | |  |  | + | Значение должно выбираться из списка (основной,филиал) |
| Study\_plans (Учебный план) | | | | | | | | |
| id\_study\_plan | INTEGER | | | + |  |  | + | Значение соответствует первичному ключу сущности  учебный план |
| id\_edu\_track(Код направления) | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности  направление |
| Id\_subdivision  (ID поздразделения) | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности  подразделение |
| Year (Год) | INTEGER | | |  |  |  | + | 1900>val>2030 |
| Edu\_tracks (Направление) | | | | | | | | |
| Id\_edu\_track | INTEGER | | | + |  |  | + | Уникален, необходимо обеспечить автоматическую генерацию значения |
| Code\_edu\_track  (Код направления) | VARCHAR(10) | | |  |  |  | + |  |
| Name  (Название направления) | VARCHAR(50) | | |  |  |  | + |  |
| Disciplines (Дисциплина) | | | | | | | | |
| Id\_discipline | INTEGER | | | + |  |  | + | Уникален, необходимо обеспечить автоматическую генерацию значения |
| Name | VARCHAR(70) | | |  |  |  | + |  |
| Selfstudy\_hours | INTEGER | | |  |  |  | + | >0 |
| Practice\_hours | INTEGER | | |  |  |  | + | >0 |
| Lecture\_hours | INTEGER | | |  |  |  | + | >0 |
| Study\_courses (Перечень учебных курсов) | | | | | | | | |
| Id\_study\_course | INTEGER | | | + |  |  | + | Уникален, необходимо обеспечить автоматическую генерацию значения |
| Semester | INTEGER | | |  |  |  |  | 1<=val<=8 |
| Certification | VARCHAR(9) | | |  |  |  | + | Значение должно выбираться из списка (зачет, диф.зачет, экзамен) |
| Id\_study\_plan | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности  учебный план |
| Id\_discipline | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности дисциплина |
| Record\_book (Зачетная книжка) | | | | | | | | |
| Id\_row\_record\_book | INTEGER | | | + |  |  | + | Уникален, необходимо обеспечить автоматическую генерацию значения |
| Id\_study\_course | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности  учебный курс |
| Id\_group\_row | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности студент |
| Id\_teacher | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности преподаватель |
| Mark | INTEGER | | |  |  |  | + | 2<=val<=5 |
| Try\_number | INTEGER | | |  |  |  | + | 1<=val <= 3 |
| Students (Студент) | | | | | | | | |
| Id\_record\_book | INTEGER | | | + |  |  | + | Уникален, необходимо обеспечить автоматическую генерацию значения |
| Name | VARCHAR(40) | | |  |  |  | + |  |
| Surname | VARCHAR(40) | | |  |  |  | + |  |
| Patronimic | VARCHAR(40) | | |  |  |  | - |  |
| Student\_groups (История групп) | | | | | | | | |
| Id\_group\_row | INTEGER | | | + |  |  | + | Уникален, необходимо обеспечить автоматическую генерацию значения |
| Id\_group | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности группа |
| Id\_record\_book | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности студент |
| Groups (Группа) | | | | | | | | |
| Id\_group | INTEGER | | | + |  |  |  | Уникален, необходимо обеспечить автоматическую генерацию значения |
| Id\_study\_plan | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности учебный план |
| Start\_date | DATE | | |  |  |  | + |  |
| End\_date | DATE | | |  |  |  | + | val > Начало обучения |
| Group\_name | VARCHAR(10) | | |  |  |  | + |  |
| Course | INTEGER | | |  |  |  | + | 0<= val <= 4 |
| Awarded\_scholarships (Назначенные стипендии) | | | | | | | | |
| Id\_awarded\_scholarship | INTEGER | | | + |  |  | + | Уникален, необходимо обеспечить автоматическую генерацию значения |
| Id\_scholarship | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности  стипендия |
| Id\_record\_book | INTEGER | | |  |  |  | + | Значение соответствует первичному ключу сущности  студент |
| Start\_date | DATE | | |  |  |  | + |  |
| End\_date | DATE | | |  |  |  | + | val > С |
| Scholarships (Стипендия) | | | | | | | | |
| Id\_scholarship | INTEGER | | | + |  |  | + | Уникален, необходимо обеспечить автоматическую генерацию значения |
| name | VARCHAR(100) | | |  |  |  | + |  |
| Money\_amount | MONEY | | |  |  |  | + | val>0 |
| Type\_of\_scholarship | VARCHAR(13) | | |  |  |  | + | Значение должно выбираться из списка (социальная, академическая, именная) |
| Scholarship\_payments (История выплат) | | | | | | | | |
| Id\_scholarship\_pay | INTEGER | | | + |  |  | + | Уникален, необходимо обеспечить автоматическую генерацию значения |
| Id\_awarded\_scholarship | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности  назначенные стипендии |
| Pay\_date | DATE | | |  |  |  | + |  |
| Status | VARCHAR(14) | | |  |  |  | + | Значение должно выбираться из списка (выплачено, не выплачено, обрабатывается) |
| Campuses (Корпус) | | | | | | | | |
| Id\_campus | INTEGER | | | + |  |  | + | Уникален, необходимо обеспечить автоматическую генерацию значения |
| Address | VARCHAR(70) | | |  |  |  | + |  |
| name | VARCHAR(30) | | |  |  |  | + |  |
| Rooms (Аудитория) | | | | | | | | |
| Id\_room | INTEGER | | | + |  |  | + | Уникален, необходимо обеспечить автоматическую генерацию значения |
| Id\_campus | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности  корпус |
| Room\_type | VARCHAR(12) | | |  |  |  | + | Значение должно выбираться из списка (лекционная, практическая, лаборатория) |
| Room\_number | INTEGER | | |  |  |  | + | val>0 |
| Session\_schedule (Расписание сессии) | | | | | | | | |
| Id\_schedule (id сдачи) | INTEGER | | | + |  |  | + | Уникален, необходимо обеспечить автоматическую генерацию значения |
| Id\_group | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности группа |
| Id\_teacher | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности преподаватель |
| Id\_room | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности аудитория |
| Id\_study\_course | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности перечень учебных курсов |
| Date\_time | DATE | | |  |  |  | + |  |
| Teachers (Преподаватель) | | | | | | | | |
| Id\_teacher | INTEGER | | | + |  |  | + | Уникален, необходимо обеспечить автоматическую генерацию значения |
| Name | VARCHAR(40) | | |  |  |  | + |  |
| Surname | VARCHAR(40) | | |  |  |  | + |  |
| Patronymic | VARCHAR(40) | | |  |  |  | - |  |
| Employment\_contracts (Трудовые договоры) | | | | | | | | |
| Id\_contract | INTEGER | | | + |  |  | + | Уникален, необходимо обеспечить автоматическую генерацию значения |
| Id\_teachet | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности  преподаватель |
| Share\_rate (Доля ставки) | FLOAT | | |  |  |  | + | 0.1, 0.25, 0.5, 0.75, 1 |
| Id\_subdivision | INTEGER | | |  |  | + | + | Значение соответствует первичному ключу сущности  подразделения |

# **Запросы к базе данных**

### **Запрос №1:**

Список дисциплин, которые должны быть сданы заданной группой с указанием дат сдачи и фамилий преподавателей.

### Команда:

select date\_time, surname

from session\_schedule join teachers

on session\_schedule.id\_teacher = teachers.id\_teacher

where id\_group =(select id\_group from groups where group\_name='K10');

### Результат (рис 2.):

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 2 – результат работы 1-ого запроса

### **Запрос №2:**

Список студентов, получивших двойки на первой попытке с указанием фамилии преподавателя, которым они должны пересдать экзамен.

### Команда:

select s.surname as student\_name, t.surname as teacher\_surname

from record\_book

join student\_groups sg

on sg.id\_group\_row = record\_book.id\_group\_row

join students s

on sg.id\_record\_book = s.id\_record\_book

join teachers t

on record\_book.id\_teacher = t.id\_teacher

where try\_number=1 and mark=2;

### Результат(рис. 3):

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рисунок 3 – результат работы 2-ого запроса

### **Запрос №3:**

Фамилии студентов, получивших оценки по дисциплине, которые выше среднего балла по этой дисциплине.

### Команда:

from record\_book

join student\_groups sg

on sg.id\_group\_row = record\_book.id\_group\_row

join students s

on sg.id\_record\_book = s.id\_record\_book

join (select id\_study\_course, *avg*(mark) as avg\_mark from record\_book where mark<>2 group by id\_study\_course) avg\_mark

on record\_book.id\_study\_course = avg\_mark.id\_study\_course

where record\_book.mark > avg\_mark.avg\_mark;

### Результат (рис 4.):

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рисунок 4 – результат работы 3-его запроса

### **Запрос №4:**

Рейтинговый список групп по заданному направлению по результатам сдачи сессии, упорядочить его по убыванию.

### Команда:

select sg.group\_name, *avg*(mark) from record\_book

join

(select id\_group\_row, group\_name from student\_groups

join groups g

on g.id\_group = student\_groups.id\_group

where g.id\_study\_plan = 0) sg

on sg.id\_group\_row = record\_book.id\_group\_row

where mark<>2

group by sg.group\_name

order by *avg*(mark) desc;

### Результат (рис. 5):

**Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание**

Рисунок 5 – результат работы 4-ого запроса

### **Запрос №5:**

Cписки студентов, упорядоченные по группам и фамилиям студентов, назначении на стипендии. Студент получает стипендию, если он сдал сессию без троек. Если студент не назначен на стипендию, указать 0, если назначен – 1.

### Команда:

select s.surname, sg.group\_name,

case

when *min*(mark) > 3 then 1

else 0

end as is\_awarder

from record\_book

join (select id\_record\_book, id\_group\_row, group\_name

from student\_groups

join groups g

on g.id\_group = student\_groups.id\_group) sg

on sg.id\_group\_row = record\_book.id\_group\_row

join students s

on sg.id\_record\_book = s.id\_record\_book

group by (s.surname, sg.group\_name)

order by surname, group\_name;

### Результат (рис. 6):

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рисунок 6 – результат работы 5-ого запроса

### **Запрос №6:**

Список студентов, сдавших все положенные экзамены.

### Команда:

select id\_record\_book from record\_book

join (select id\_group\_row, id\_study\_course, *max*(try\_number) as last\_try

from record\_book r2

group by (id\_group\_row, id\_study\_course)) as trys

on record\_book.try\_number=last\_try and

record\_book.id\_group\_row = trys.id\_group\_row and

record\_book.id\_study\_course = trys.id\_study\_course

join student\_groups sg

on record\_book.id\_group\_row = sg.id\_group\_row

group by sg.id\_record\_book

having *min*(mark) > 2;

### Результат (рис. 7):

Изображение выглядит как текст, число

Автоматически созданное описание

Рисунок 7 – результат работы 6-ого запроса

### **Запрос №7:**

Список студентов, получивших максимальный средний балл в своей группе.

### Команда:

select sg.id\_record\_book, sg.id\_group

from

(select id\_group\_row, *avg*(mark) as avg\_st

from record\_book

where mark<>2

group by id\_group\_row) avg\_st

join

student\_groups sg

on

sg.id\_group\_row = avg\_st.id\_group\_row

join

(select id\_group, *max*(avg\_st) as max\_group\_avg from record\_book as r1

join student\_groups sg on

r1.id\_group\_row = sg.id\_group\_row

join (select id\_group\_row, *avg*(mark) as avg\_st

from record\_book

where mark<>2

group by id\_group\_row) s\_av

on r1.id\_group\_row = s\_av.id\_group\_row

group by id\_group) as max\_group\_avg

on

sg.id\_group = max\_group\_avg.id\_group

where avg\_st=max\_group\_avg;

### Результат (рис. 8):

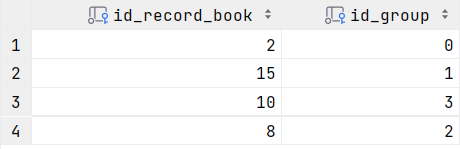


Рисунок 8 – результат работы 7-ого запроса

# **Создание представлений**

### **Представление №1:**

Cписок студентов, получивших двойки на первой попытке с указанием фамилии преподавателя, которым они должны пересдать экзамен.

### Команда:

create or replace view first\_mark\_two as

select s.surname as student\_name, t.surname as teacher\_surname

from record\_book

join student\_groups sg

on sg.id\_group\_row = record\_book.id\_group\_row

join students s

on sg.id\_record\_book = s.id\_record\_book

join teachers t

on record\_book.id\_teacher = t.id\_teacher

where try\_number=1 and mark=2;

### Результат (рис. 9):

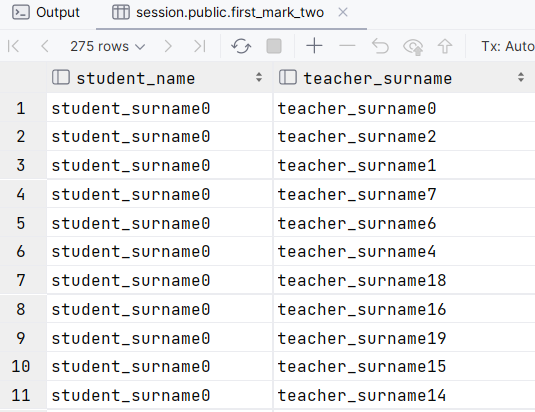
****

Рисунок 9 – представление №1

### **Представление №2:**

Создать представление для студентов, побывавших на ппа2

### Команда:

create or replace view students\_ppa2 as

select s.surname as student\_name, t.surname as teacher\_surname

from record\_book

join student\_groups sg

on sg.id\_group\_row = record\_book.id\_group\_row

join students s

on sg.id\_record\_book = s.id\_record\_book

join teachers t

on record\_book.id\_teacher = t.id\_teacher

where try\_number=2 and mark=2;

### Результат (рис. 10):

**Изображение выглядит как текст, число, снимок экрана, Шрифт

Автоматически созданное описание**

Рисунок 10 – представление №2

# **Запросы на модификацию данных (INSERT, UPDATE, DELETE) с использованием подзапросов**

### **Запрос №1 на update:**

Заменить название предмета на "nice discipline", по которому student\_surname4 получил больше всего 5.

### Команда:

create temporary table cnt\_5\_marks as

select id\_discipline, *count*(*\**) cnt\_5 from record\_book

join study\_courses sc

on record\_book.id\_study\_course = sc.id\_study\_course

where

id\_group\_row in

(select id\_group\_row from student\_groups where id\_record\_book in

(select id\_record\_book from students where surname='student\_surname4'))

and

mark=5

group by id\_discipline;

update disciplines

set name='nice discipline'

where id\_discipline=(select id\_discipline from cnt\_5\_marks

where cnt\_5=(select *max*(cnt\_5) from cnt\_5\_marks));

select *\** from disciplines where id\_discipline=8;

drop table cnt\_5\_marks;

### Скриншот до (рис. 11):

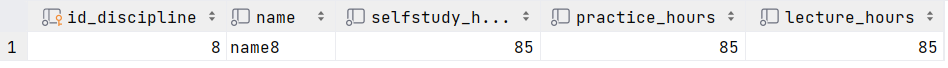


Рисунок 11 – запрос №1 – до

### Скриншот после (рис.12):

 Рисунок 12 – запрос №1 – после

### **Запрос №2 на update:**

Увеличить количество selfstudy\_hours на 10 для дисциплин с самой низкой средней успеваемостью

### Команда:

create temporary table discipline\_avg as

select d.id\_discipline, *avg*(mark) avg\_mark from record\_book

join study\_courses sc

on record\_book.id\_study\_course = sc.id\_study\_course

join

disciplines d on sc.id\_discipline = d.id\_discipline

where mark <> 2

group by d.id\_discipline;

update disciplines

set selfstudy\_hours = selfstudy\_hours + 10

where id\_discipline in

(select id\_discipline from discipline\_avg

where avg\_mark>=all(select avg\_mark from discipline\_avg)

limit 1);

drop table discipline\_avg;

### Скриншот до (рис. 13):

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рисунок 13 – запрос №2 – до

### Скриншот после (рис. 14):

Изображение выглядит как текст, число, снимок экрана, Параллельный

Автоматически созданное описание

Рисунок 14 – запрос №2 – после

### **Запрос №1 на удаление:**

Удалить подразделение, в котором не обучается ни один студент и не трудоустроен ни один преподаватель

### Команда:

delete from subdivisions

where id\_subdivision not in

(select distinct id\_subdivision from study\_plans

where id\_study\_plan in

(select distinct id\_study\_plan from groups

where id\_group in

(select distinct id\_group from student\_groups))

union

select distinct id\_subdivision from employment\_contracts);

### Скриншот до (рис. 15):

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рисунок 15 – запрос №3 – до

### Скриншот после (рис. 16):

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рисунок 16 – запрос №3 – после

### **Запрос №2 на удаление:**

Удалить учебные планы, которые были созданы позже всего и не реализует ни одной дисциплины

### Команда:

delete from study\_plans

where id\_study\_plan in

(select id\_study\_plan from study\_plans

where id\_study\_plan not in

(select distinct id\_study\_plan from study\_courses)

and year>=all(select year from study\_plans));

### Скриншот до (рис. 17):

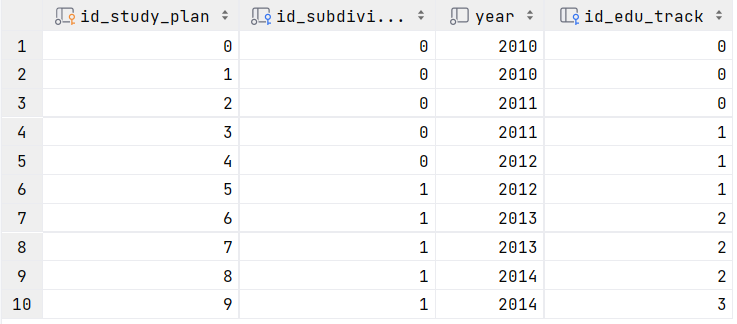


Рисунок 17 – запрос №4 – до

### Скриншот после (рис.18):

Изображение выглядит как текст, число, Шрифт, снимок экрана

Автоматически созданное описание

Рисунок 18 – запрос №4 – после

### **Запрос №1 на вставку:**

Для студента с id\_record\_book=2 во время его обучения в группе c id\_group=0. Если по дисциплине была получена 3 на 1-й попытке, то добавить 4 со второй попытки.

### Команда:

insert into record\_book

(id\_group\_row, id\_teacher, id\_study\_course, mark, try\_number)

select id\_group\_row, id\_teacher, id\_study\_course, 4, 2 from record\_book

where

id\_group\_row =

(select id\_group\_row from student\_groups

where id\_group=0 and id\_record\_book=2)

and

mark=3

and

try\_number=1;

### Скриншот до (рис. 19):

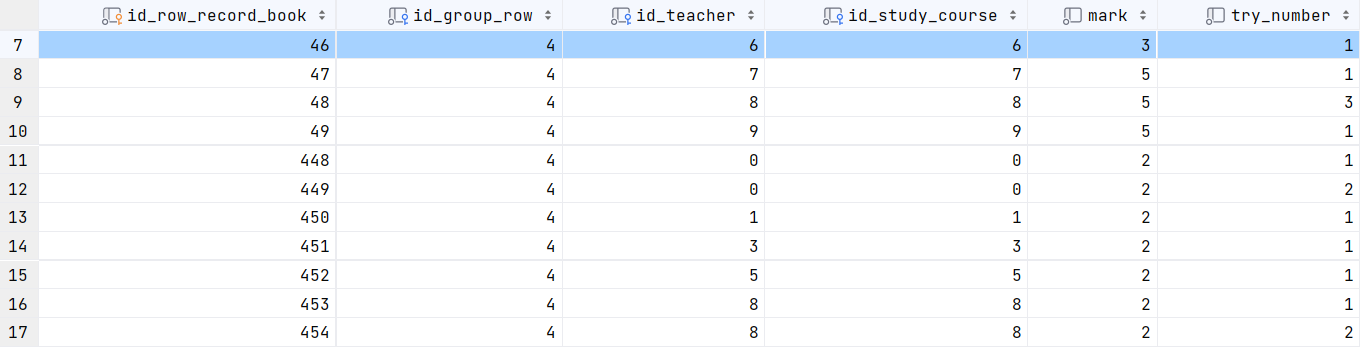


Рисунок 19 – запрос №5 – до

### Скриншот после (рис. 20):

Изображение выглядит как текст, снимок экрана, линия, число

Автоматически созданное описание

Рисунок 20 – запрос №5 – после

# **Создание индексов**

### **Запрос с составным индексом:**

Список студентов, получивших двойки на первой попытке с указанием фамилии преподавателя, которым они должны пересдать экзамен.

### Команда:

select s.surname as student\_name, t.surname as teacher\_surname

from record\_book

join student\_groups sg

on sg.id\_group\_row = record\_book.id\_group\_row

join students s

on sg.id\_record\_book = s.id\_record\_book

join teachers t

on record\_book.id\_teacher = t.id\_teacher

where try\_number=1 and mark=2;

### План операции до создания индексов (рис. 21):

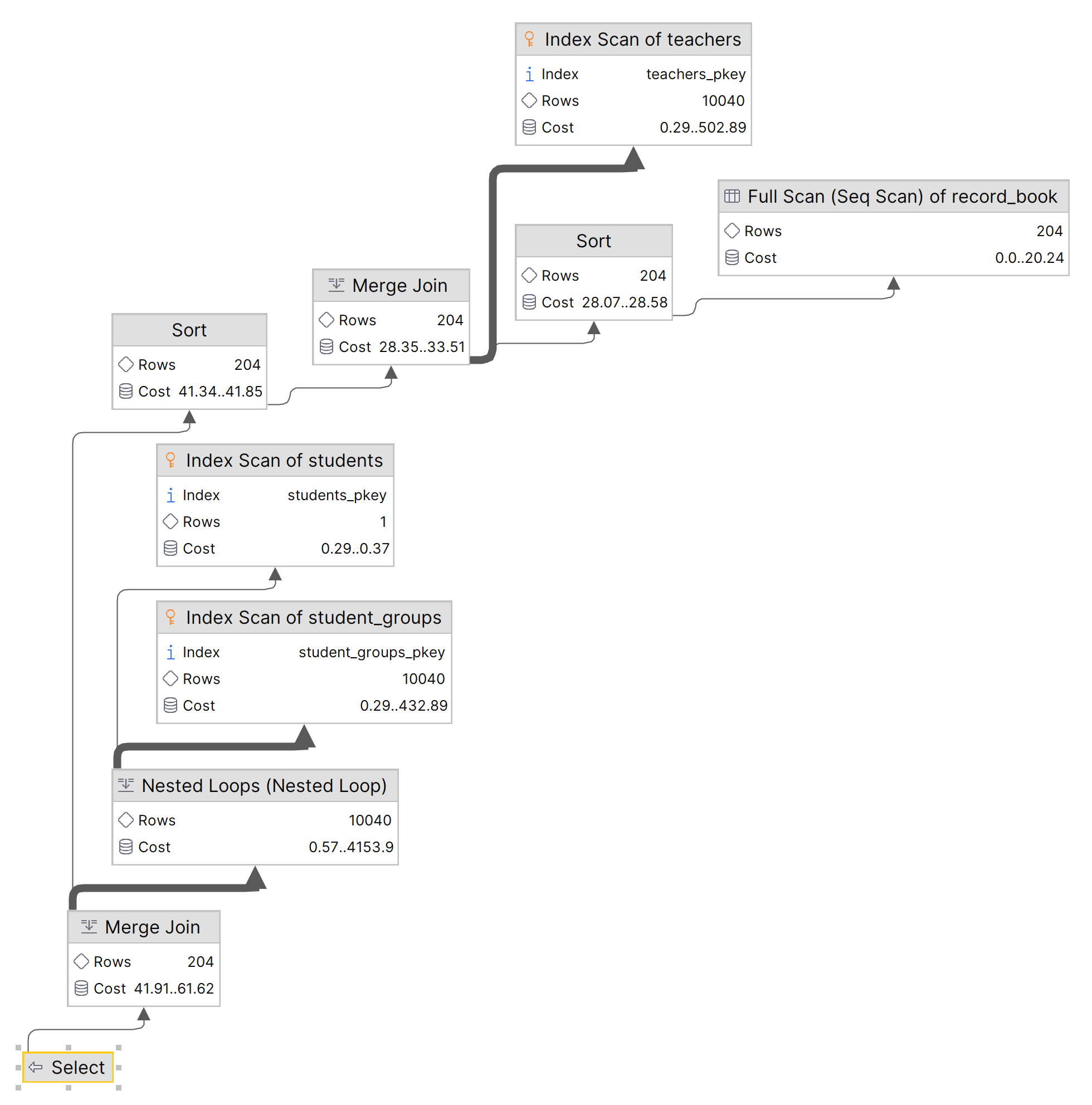
****

Рисунок 21 – план операции до создания индексов

### Время до создания индексов:

### 437ms

### **Добавление индексов:**

### create index try\_mark\_idx on record\_book(try\_number, mark);

create index record\_book\_idteacher\_idx on record\_book(id\_teacher);

create index record\_book\_group\_row\_idx on record\_book(id\_group\_row);

create index group\_recordbook\_idx on student\_groups(id\_record\_book);

### План операции после создания индексов (рис. 22):

**Изображение выглядит как текст, диаграмма, снимок экрана, План

Автоматически созданное описание**

Рисунок 22 – план операции после создания индексов

### Время после создания индексов:

58ms

### **Запрос без составных индексов:**

Получить количество 5-к по дисциплинам у студента с surname=student\_surname4

### Команда:

select id\_discipline, *count*(*\**) cnt\_5 from record\_book

join study\_courses sc

on record\_book.id\_study\_course = sc.id\_study\_course

where

id\_group\_row in

(select id\_group\_row from student\_groups where id\_record\_book in

(select id\_record\_book from students where surname='student\_surname4'))

and

mark=5

group by id\_discipline;

### Время до создания индексов:

68ms

### План операции до создания индексов (рис. 23):

**Изображение выглядит как текст, диаграмма, Параллельный, снимок экрана

Автоматически созданное описание**

Рисунок 23 – план операции до создания индексов

### **Добавление индексов:**

create index mark\_idx on record\_book(mark);

create index student\_surname\_idx on students(surname);

create index group\_recordbook\_idx on student\_groups(id\_record\_book);

### Время после создания индексов:

48ms

### План операции после добавления индексов:

План до добавления дополнительных студентов (рис. 24):

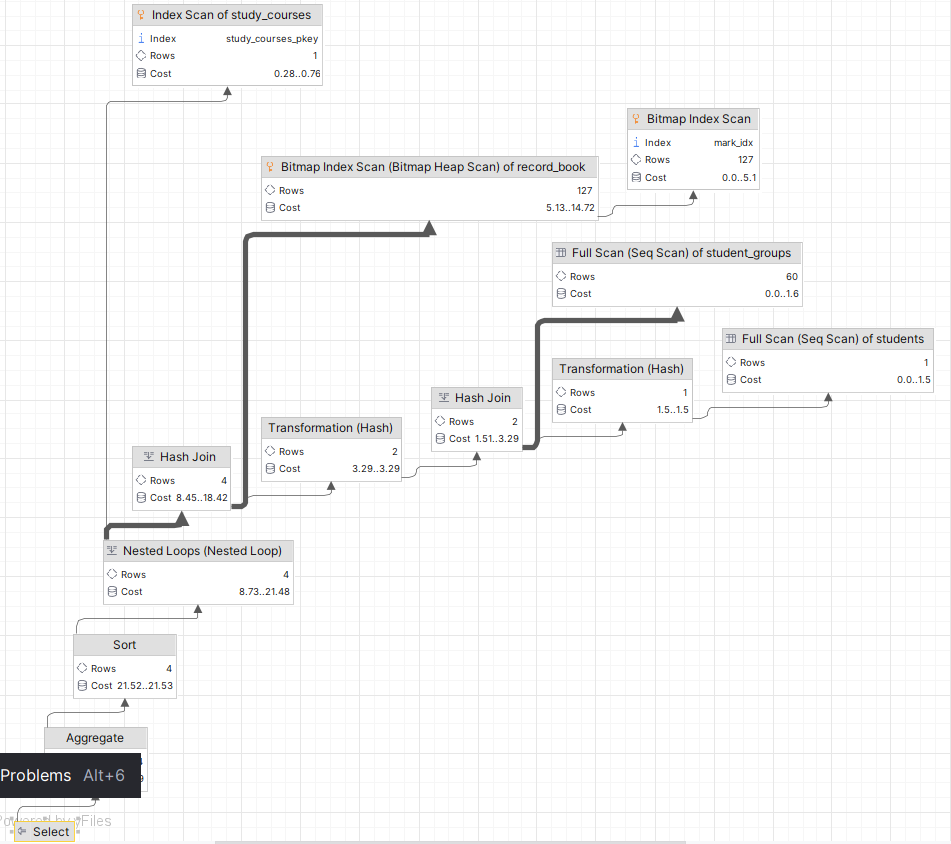


Рисунок 24 – план операции после создания индексов и добавления дополнительных студентов

### План после добавления 10000 студентов (рис. 25):

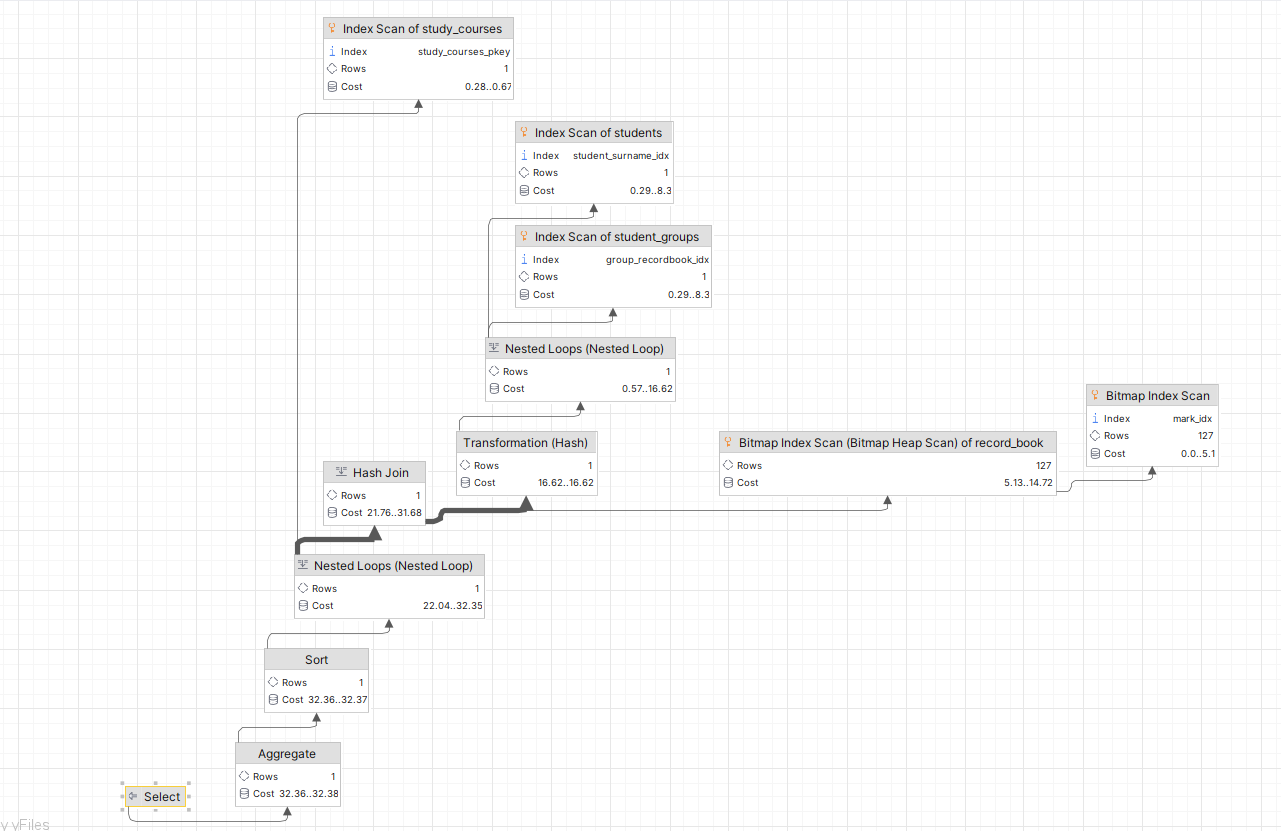


Рисунок 25 – план операции после создания индексов и добавления 10000 дополнительных студентов

# **Методы**

## № . 1

### Описание:

Повысить социальную стипендию на 10%

### Код метода:

create or replace function *increase\_social\_scholarship*()

returns void as

$$

declare

prev\_amount int;

scholarship\_name text;

begin

scholarship\_name := 'социальная имя';

select money\_amount into prev\_amount

from scholarships

where name=scholarship\_name;

update scholarships

set

money\_amount = prev\_amount \* 1.1

where

name = scholarship\_name;

end;

$$ language plpgsql;

### Выполнение метода:

До (рис. 26):

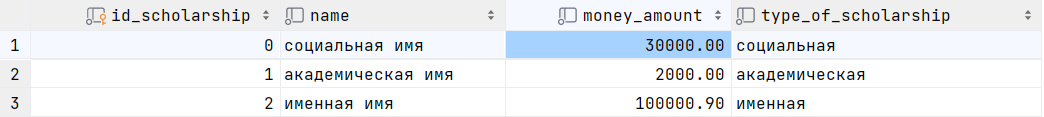


Рисунок 26 – выполнение метода до

После (рис. 27):

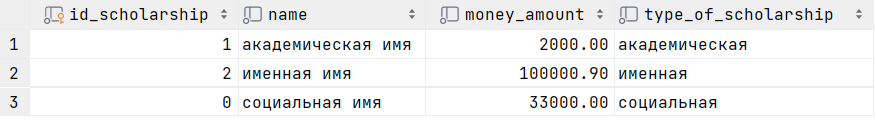


Рисунок 27 – выполнение метода после

## № . 2

### Описание:

Добавить новую группу в историю групп.

### Код метода:

create or replace function

*plus\_course*(id\_record\_book\_inp int, id\_new\_group\_inp int)

returns void as

$$

declare

current\_course int;

new\_group\_course int;

begin

select *max*(course) into current\_course from groups

where id\_group in

(select id\_group from student\_groups where id\_record\_book=id\_record\_book\_inp);

select course into new\_group\_course from groups

where id\_group = id\_new\_group\_inp;

if new\_group\_course <> current\_course+1 then

raise exception 'new course of group is not larger than current course of group by one';

end if;

insert into student\_groups(id\_record\_book, id\_group)

values (id\_record\_book\_inp, id\_new\_group\_inp);

end;

$$ language plpgsql;

### Выполнение метода:

Выведем историю групп студента 20. Он сейчас находится на 1-м курсе в группе 4 (рис. 28):

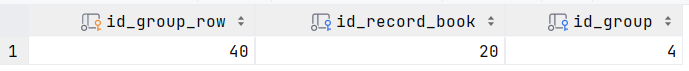


Рисунок 28 – выполнение запроса

Попытаемся добавить в группу 0 с курсом 1:

select *plus\_course*(20, 0)

Получаем ошибку:

[2023-05-06 12:26:54] [P0001] ОШИБКА: new course of group is not larger than current course of group by one

[2023-05-06 12:26:54] Где: функция PL/pgSQL plus\_course(integer,integer), строка 14, оператор RAISE

Попытаемся добавить в группу 5 с курсом 2:

select *plus\_course*(20, 5);

Посмотрим на результат (рис. 29):

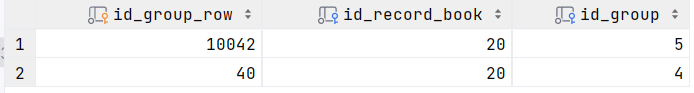


Рисунок 29 – выполнение метода

## № 3

### Описание:

Метод для изменения оценки при успешной пересдаче экзамена

### Код метода:

create or replace function *change\_mark*(input\_id\_study\_course int, input\_id\_group\_row int, new\_mark int) returns void as

$$

DECLARE

last\_try\_number int;

var\_id\_teacher int;

begin

select *max*(try\_number) into last\_try\_number from record\_book

where

id\_study\_course = input\_id\_study\_course and

id\_group\_row = input\_id\_group\_row;

select id\_teacher into var\_id\_teacher from record\_book

where

id\_study\_course = input\_id\_study\_course and

id\_group\_row = input\_id\_group\_row and

try\_number=last\_try\_number;

if last\_try\_number = 3 then

raise exception 'Невозможно пересдать экзамен, так как были использованы все попытки';

end if;

insert into record\_book(id\_group\_row, id\_teacher, id\_study\_course, mark, try\_number)

values (input\_id\_group\_row, var\_id\_teacher, input\_id\_study\_course, new\_mark, last\_try\_number+1);

end;

$$ language plpgsql;

### Выполнение метода:

Посмотрим оценки для id\_group\_row = 0 and id\_study\_course = 1 (рис. 30):

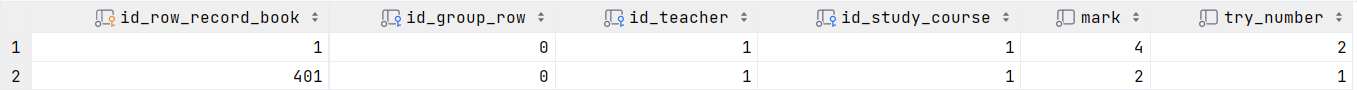


Рисунок 30 – выполнение запроса

Выполним метод:

select *change\_mark*(2, 0, 5);

Результат (рис. 31):

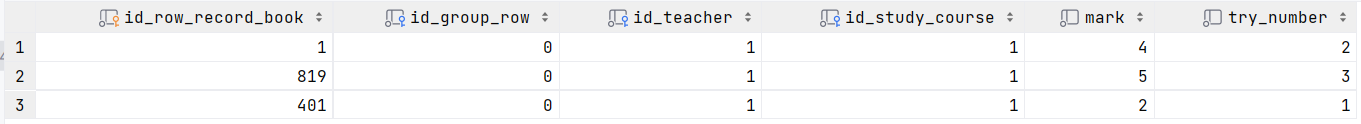


Рисунок 31 – выполнение метода

Посмотрим оценки для id\_group\_row = 0 and id\_study\_course = 2.

Результат (рис. 32):

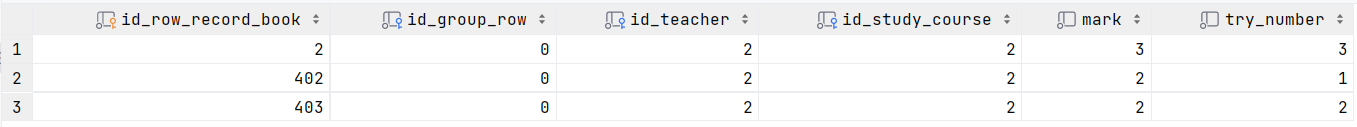


Рисунок 32 – результат запроса

Видим, что у студента использованы все попытка => ему нельзя пересдать предмета

Выполним метод:

select *change\_mark*(2, 0, 5);

Получили ошибку:

[2023-05-06 15:49:24] [P0001] ERROR: Невозможно пересдать экзамен, так как были использованы все попытки

[2023-05-06 15:49:24] Где: PL/pgSQL function change\_mark(integer,integer,integer) line 18 at RAISE

## №4.

### Описание:

У нас почему-то сбились последовательности для primary key. Мы решили написать метод, который устанавливает значение последовательностей на максимальное значение PK + 1.

### Код метода:

CREATE OR REPLACE FUNCTION *reset\_serial\_sequences*() RETURNS void AS $$

DECLARE

name\_of\_table text;

name\_of\_column text;

sequence\_name text;

max\_value bigint;

BEGIN

FOR name\_of\_table, name\_of\_column IN

SELECT table\_name, column\_name FROM information\_schema.columns

WHERE column\_default LIKE 'nextval%'

LOOP

sequence\_name := *pg\_get\_serial\_sequence*(name\_of\_table, name\_of\_column);

if name\_of\_column is not null then

EXECUTE *format*('SELECT max(%I) FROM %I', name\_of\_column, name\_of\_table) INTO max\_value;

EXECUTE *format*('SELECT setval(%L, %s)', sequence\_name, max\_value + 1);

end if;

END LOOP;

END;

$$ LANGUAGE plpgsql;

# **Триггеры**

## №1

### Описание:

Триггер для логирования событий вставки, удаления, редактирования данных в базе данных PostgreSQL

### Код:

*-- Создание таблицы для логов*

create table subdivision\_logs(

id\_log serial primary key,

operation\_type cud\_operation not null,

operation\_time timestamp without time zone,

affected\_id\_subdivision int null,

affected\_subdivision\_name text null

);

*-- Создание метода для триггера*

CREATE OR REPLACE FUNCTION *subdivision\_add\_to\_log*() RETURNS TRIGGER AS $$

DECLARE

var\_operation\_type cud\_operation;

BEGIN

IF TG\_OP = 'INSERT' THEN

var\_operation\_type := 'insert';

insert into

subdivision\_logs(operation\_type, operation\_time, affected\_id\_subdivision, affected\_subdivision\_name)

VALUES

(var\_operation\_type, *now*(), new.id\_subdivision, new.name);

RETURN NEW;

ELSIF TG\_OP = 'UPDATE' THEN

var\_operation\_type := 'update';

insert into

subdivision\_logs(operation\_type, operation\_time, affected\_id\_subdivision, affected\_subdivision\_name)

VALUES

(var\_operation\_type, *now*(), old.id\_subdivision, old.name);

RETURN NEW;

ELSIF TG\_OP = 'DELETE' THEN

var\_operation\_type := 'delete';

insert into

subdivision\_logs(operation\_type, operation\_time, affected\_id\_subdivision, affected\_subdivision\_name)

VALUES

(var\_operation\_type, *now*(), old.id\_subdivision, old.name);

RETURN OLD;

END IF;

END

$$ LANGUAGE plpgsql;

*-- Создание триггера*

create trigger log\_trigger

after insert or update or delete on subdivisions

for each row execute procedure *subdivision\_add\_to\_log*();

### Выполнение:

Выполнение insert (рис. 33):



Рисунок 33 – выполнение insert

Выполнение update (рис. 34):

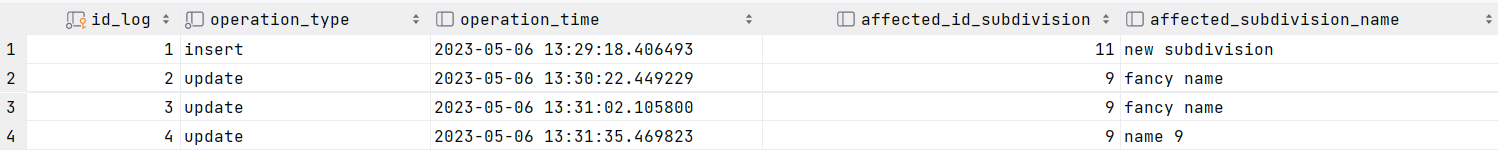


Рисунок 34 – выполнение update

Выполнение delete (рис. 35):

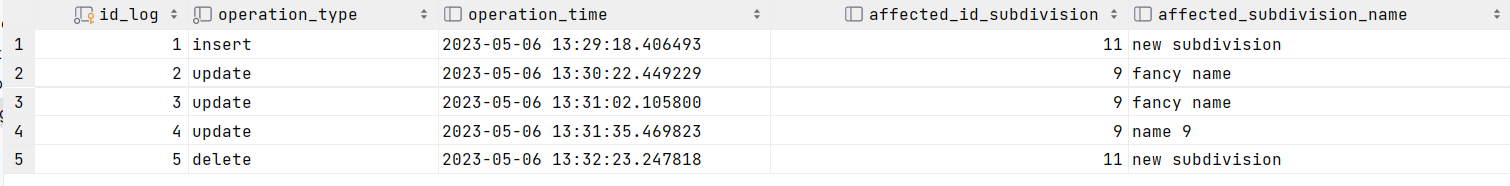


Рисунок 35 – выполнение delete

## №2

### Описание:

Триггер, который предотвращает вставку оценки, если дисциплина отсутствует в учебном плане группы.

### Код:

*-- Метод для триггера*

create or replace function *check\_study\_course*() returns trigger as

$$

declare

var\_id\_study\_courses int[];

begin

select *array\_agg*(id\_study\_course) into var\_id\_study\_courses from study\_courses

where id\_study\_plan =

(select id\_study\_plan from groups where id\_group =

(select id\_group from student\_groups where id\_group\_row=new.**id\_group\_row**));

if new.id\_study\_course <> any(var\_id\_study\_courses) then

raise EXCEPTION 'Нет такого учебного курса в учебном плане группы';

end if;

return new;

end

$$ language plpgsql;

*-- Создание триггера*

create trigger study\_course\_trigger

before insert on record\_book

for each row execute procedure *check\_study\_course*();

### Выполнение:

Попытаемся поставить оценку по дисциплине, которой нет у студента в учебном плане:

insert into record\_book(id\_group\_row, id\_teacher, id\_study\_course, mark, try\_number)

VALUES (0, 10, 10, 5, 1);

Получаем ошибку:

[2023-05-06 17:18:10] [P0001] ERROR: Нет такого учебного курса в учебном плане группы

[2023-05-06 17:18:10] Где: PL/pgSQL function check\_study\_course() line 11 at RAISE

# **Выводы**

В результате выполнения проекта мы успешно реализовали базу данных для ведения учебного процесса в высшем учебном заведении. Были реализованы функции, триггеры, представления и индексы, которые значительно расширили возможности базы данных и улучшили ее эффективность.

Благодаря выполнению проекта мы расширили свои знания в области проектирования и реализации баз данных, используя PostgreSQL