МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

ГЛАВНОЕ УПРАВЛЕНИЕ ПО ОБРАЗОВАНИЮ

МОГИЛЕВСКОГО ОБЛАСТНОГО ИСПОЛНИТЕЛЬНОГО КОМИТЕТА

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«МОГИЛЕВСКИЙ ГОСУДАРСТВЕННЫЙ ПОЛИТЕХНИЧЕСКИЙ КОЛЛЕДЖ»

**БАЗЫ ДАННЫХ И СИСТЕМЫ УПРАВЛЕНИЯ**

**БАЗАМИ ДАННЫХ**

ДОМАШНЯЯ КОНТРОЛЬНАЯ РАБОТА №1

Выполнил Учащийся группы ПО-455 М.Г.Верхов

Шифр 15

2021

**19 Что называют транзакциями? Что называют сериализацией транзакций? Опишите сериальный план выполнения транзакций.**

Понятие транзакции имеет непосредственную связь с понятием целостности БД. Очень часто БД может обладать такими ограничениями целостности, которые просто невозможно не нарушить, выполняя только один оператор изменения БД. Например, в базе данных СОТРУДНИКИ-ОТДЕЛЫ естественным ограничением целостности является совпадения значения атрибута ОТД\_РАЗМЕР в кортеже отношения ОТДЕЛЫ, описывающем данный отдел (например, отдел 320), с числом кортежей отношения СОТРУДНИКИ таких, что значение атрибута СОТР\_ОТД\_НОМЕР равно 320. Как в этом случае принять на работу в отдел 320 нового сотрудника? Независимо от того, какая операция будет выполнена первой, вставка нового кортежа в отношение СОТРУДНИКИ или модификация существующего кортежа в отношении ОТДЕЛЫ, после выполнения операции база данных окажется в нецелостном состоянии.

Поэтому для поддержания подобных ограничений целостности допускается их нарушение внутри транзакции с тем условием, чтобы к моменту завершения транзакции условия целостности были соблюдены. В системах с развитыми средствами ограничения и контроля целостности каждая транзакция начинается при целостном состоянии БД и должна оставить это состояние целостными после своего завершения. Несоблюдение этого условия приводит к тому, что вместо фиксации результатов транзакции происходит ее откат (т.е. вместо оператора COMMIT выполняется оператор ROLLBACK), и БД остается в таком состоянии, в котором находилась к моменту начала транзакции, т.е. в целостном состоянии.

Если быть немного более точным, различаются два вида ограничений целостности: немедленно проверяемые и откладываемые. К немедленно проверяемым ограничениям целостности относятся такие ограничения, проверку которых бессмысленно или даже невозможно откладывать. Примером ограничения, проверку которого откладывать бессмысленно, являются ограничения домена (возраст сотрудника не может превышать 150 лет). Более сложным ограничением, проверку которого невозможно отложить, является следующее: зарплата сотрудника не может быть увеличена за одну операцию более, чем на 100,000 рублей. Немедленно проверяемые ограничения целостности соответствуют уровню отдельных операторов языкового уровня СУБД. При их нарушениях не производится откат транзакции, а лишь отвергается соответствующий оператор.

Откладываемые ограничения целостности - это ограничения на базу данных, а не на какие-либо отдельные операции. По умолчанию такие ограничения проверяются при конце транзакции, и их нарушение вызывает автоматическую замену оператора COMMIT на оператор ROLLBACK. Однако в некоторых системах поддерживается специальный оператор насильственной проверки ограничений целостности внутри транзакции. Если после выполнения такого оператора обнаруживается, что условия целостности не выполнены, пользователь может сам выполнить оператор ROLLBACK или постараться устранить причины нецелостного состояния базы данных внутри транзакции (видимо, это осмысленно только при использовании интерактивного режима работы).

И еще одно замечание. С точки зрения внешнего представления в момент завершения транзакции проверяются все откладываемые ограничения целостности, определенные в этой базе данных. Однако при реализации стремятся при выполнении транзакции динамически выделить те ограничения целостности, которые действительно могли бы быть нарушены. Например, если при выполнении транзакции над базой данных СОТРУДНИКИ-ОТДЕЛЫ в ней не выполнялись операторы вставки или удаления кортежей из отношения СОТРУДНИКИ, то проверять упоминавшееся выше ограничение целостности не требуется (а проверка подобных ограничений вызывает достаточно большую работу).

Для того, чтобы добиться изолированности транзакций, в СУБД должны использоваться какие-либо методы регулирования совместного выполнения транзакций.

План (способ) выполнения набора транзакций называется сериальным, если результат совместного выполнения транзакций эквивалентен результату некоторого последовательного выполнения этих же транзакций.

Сериализация транзакций - это механизм их выполнения по некоторому сериальному плану. Обеспечение такого механизма является основной функцией компонента СУБД, ответственного за управление транзакциями. Система, в которой поддерживается сериализация транзакций обеспечивает реальную изолированность пользователей.

Основная реализационная проблема состоит в выборе метода сериализации набора транзакций, который не слишком ограничивал бы их параллельность. Приходящим на ум тривиальным решением является действительно последовательное выполнение транзакций. Но существуют ситуации, в которых можно выполнять операторы разных транзакций в любом порядке с сохранением сериальности. Примерами могут служить только читающие транзакции, а также транзакции, не конфликтующие по объектам базы данных.

Между транзакциями могут существовать следующие виды конфликтов:

* W-W - транзакция 2 пытается изменять объект, измененный не закончившейся транзакцией 1;
* R-W - транзакция 2 пытается изменять объект, прочитанный не закончившейся транзакцией 1;
* W-R - транзакция 2 пытается читать объект, измененный не закончившейся транзакцией 1.

Практические методы сериализации транзакций основывается на учете этих конфликтов.

Существуют два базовых подхода к сериализации транзакций - основанный на синхронизационных захватах объектов базы данных и на использовании временных меток. Суть обоих подходов состоит в обнаружении конфликтов транзакций и их устранении.

Предварительно заметим, что для каждого из подходов имеются две разновидности - пессимистическая и оптимистическая. При применении пессимистических методов, ориентированных на ситуации, когда конфликты возникают часто, конфликты распознаются и разрешаются немедленно при их возникновении. Оптимистические методы основываются на том, что результаты всех операций модификации базы данных сохраняются в рабочей памяти транзакций. Реальная модификация базы данных производится только на стадии фиксации транзакции. Тогда же проверяется, не возникают ли конфликты с другими транзакциями.

Далее мы ограничимся рассмотрением более распространенных пессимистических разновидностей методов сериализации транзакций. Пессимистические методы сравнительно просто трансформируются в свои оптимистические варианты.

Наиболее распространенным в централизованных СУБД (включающих системы, основанные на архитектуре "клиент-сервер") является подход, основанный на соблюдении двухфазного протокола синхронизационных захватов объектов БД. В общих чертах протокол состоит в том, что перед выполнением любой операции в транзакции T над объектом базы данных r от имени транзакции T запрашивается синхронизационный захват объекта r в соответствующем режиме (в зависимости от вида операции).

Основными режимами синхронизационных захватов являются:

* совместный режим - S (Shared), означающий разделяемый захват объекта и требуемый для выполнения операции чтения объекта;
* монопольный режим - X (eXclusive), означающий монопольный захват объекта и требуемый для выполнения операций занесения, удаления и модификации.

Захваты объектов несколькими транзакциями по чтению совместимы, т.е. нескольким транзакциям допускается читать один и тот же объект, захват объекта одной транзакцией по чтению не совместим с захватом другой транзакцией того же объекта по записи, и захваты одного объекта разными транзакциями по записи не совместимы.

Альтернативный метод сериализации транзакций, хорошо работающий в условиях редких конфликтов транзакций и не требующий построения графа ожидания транзакций. основан на использовании временных меток.

Основная идея метода (у которого существует множество разновидностей) состоит в следующем: если транзакция T1 началась раньше транзакции T2, то система обеспечивает такой режим выполнения, как если бы T1 была целиком выполнена до начала T2.

Для этого каждой транзакции T предписывается временная метка t, соответствующая времени начала T. При выполнении операции над объектом r транзакция T помечает его своей временной меткой и типом операции (чтение или изменение).

Перед выполнением операции над объектом r транзакция T1 выполняет следующие действия:

* Проверяет, не закончилась ли транзакция T, пометившая этот объект. Если T закончилась, T1 помечает объект r и выполняет свою операцию.
* Если транзакция T не завершилась, то T1 проверяет конфликтность операций. Если операции неконфликтны, при объекте r остается или проставляется временная метка с меньшим значением, и транзакция T1 выполняет свою операцию.
* Если операции T1 и T конфликтуют, то если t(T) > t(T1) (т.е. транзакция T является более "молодой", чем T), производится откат T и T1 продолжает работу.
* Если же t(T) < t(T1) (T "старше" T1), то T1 получает новую временную метку и начинается заново.

К недостаткам метода временных меток относятся потенциально более частые откаты транзакций, чем в случае использования синхронизационных захватов. Это связано с тем, что конфликтность транзакций определяется более грубо. Кроме того, в распределенных системах не очень просто вырабатывать глобальные временные метки с отношением полного порядка (это отдельная большая наука).

**36 Найдите учащихся, фамилии которых не начинаются с букв «М» и «Д»**

Создадим базу данных колледж. Заполним таблицы тестовыми данными. Настроим схему БД. Схема данных БД «Колледж» представлена на рисунке 1.

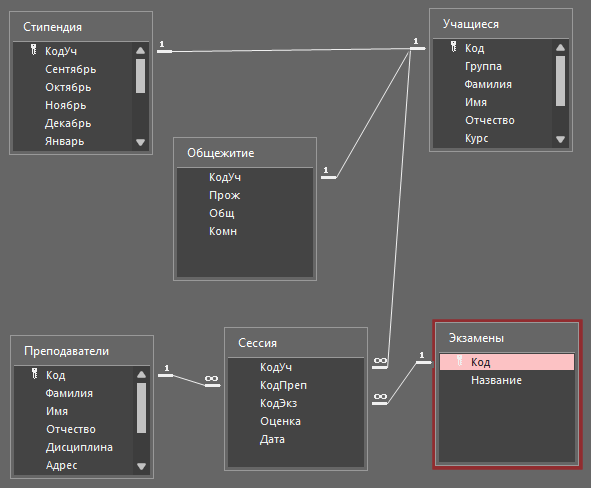


Рисунок 1. Схема БД «Колледж»

Заполним таблицы тестовыми данными. Заполненная таблица «Учащиеся представлена на рисунке 2.

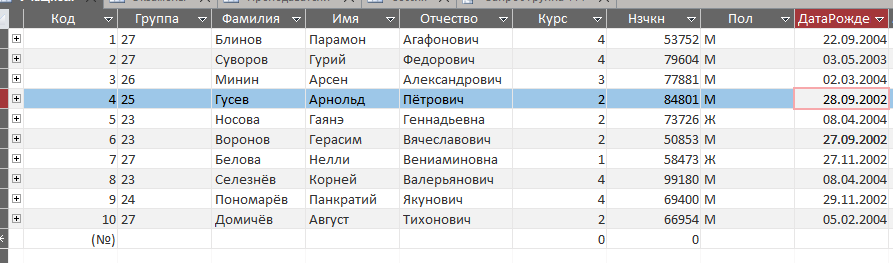


Рисунок 2. Таблица «Учащиеся»

По условию задания нам необходимо создать запрос выбирающий учащихся, фамилия которых не начинается с букв «М» и «Д». Создадим запрос «Учащиеся Запрос 36» выберем желаемые поля и укажем критерии отбора по фамилии согласно нашему заданию. Запрос в режиме конструктора представлен на рисунке 3.

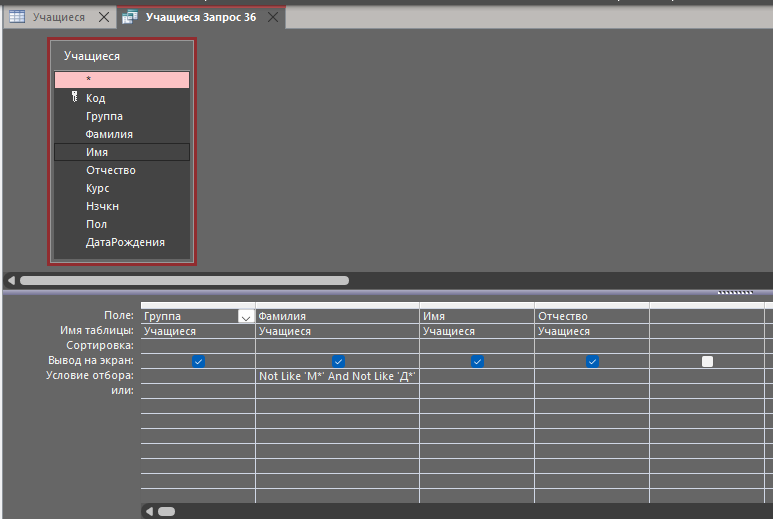


Рисунок 3. Запрос в режиме конструктора

Сохраним запрос и выполним его. Результат выполнения запроса представлен на рисунке 4.

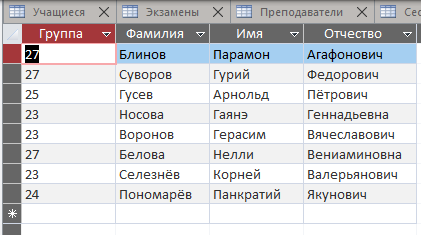


Рисунок 4. Результат выполнения запроса.

**75 Найдите средний балл по дисциплине. Наименование дисциплины - параметр запроса**

Заполним данными таблицы «Сессия», «Преподаватели», «Экзамены». Результат заполнения таблицы «Преподаватели» представлен на рисунке 5.

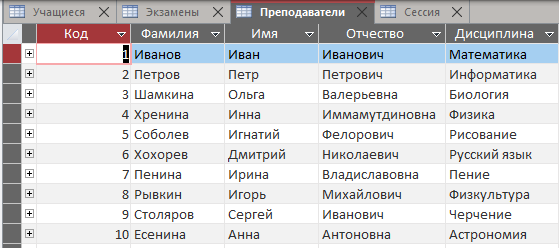


Рисунок 5. Результат заполнения таблицы «Преподаватели»

Результат заполнения таблицы «Экзамены» представлен на рисунке 6.

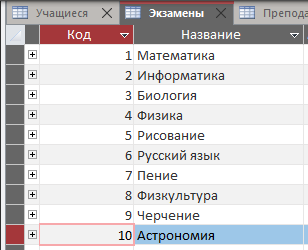


Рисунок 6. Результат заполнения таблицы «Экзамены»

Таблица «Сессия» представлена на рисунке 7.

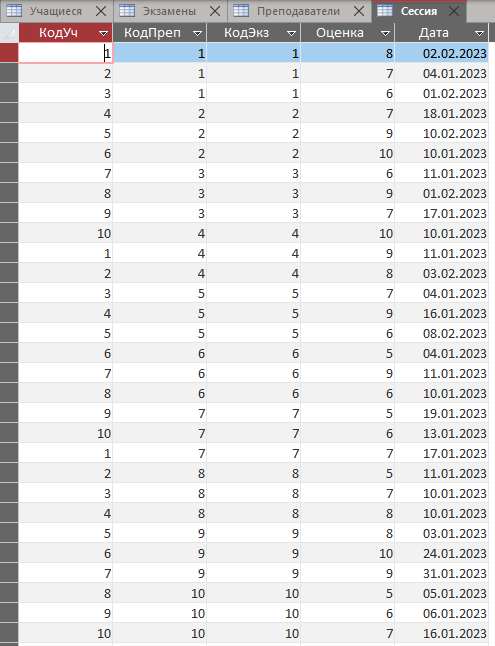


Рисунок 7. Таблица «Сессия»

По условию задания нам необходимо найти средний балл по дисциплине, которая является параметром запроса.

Создадим запрос в режиме конструктора, укажем Название дисциплины для параметра, для этого подключим таблицу Экзамены. Укажем поле для выбора по параметру, запрос на параметр и отметим необходимые к выборке поля.

Запрос в режиме конструктора представлен на рисунке 8.

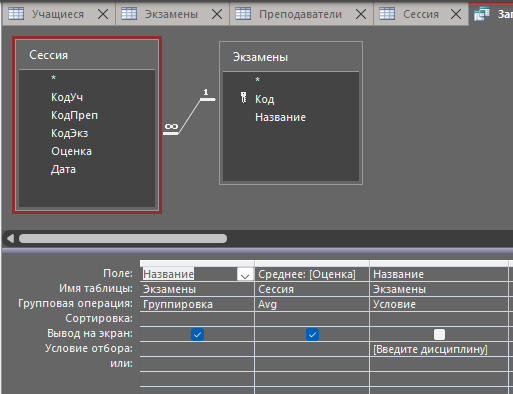


Рисунок 8. Запрос в режиме конструктора.

Из рисунка 7 (первые три строки) видим что баллы у нас 6,7 и 8, а значит по дисциплине Математика ожидается средний балл 7. Запрос Дисциплины представлен на рисунке 9.

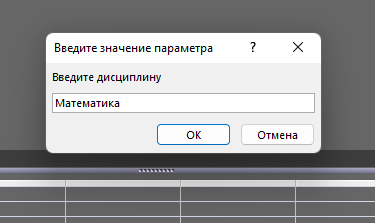


Рисунок 9. Запрос Дисциплины.

Результат выполнения запроса представлен на рисунке 10.

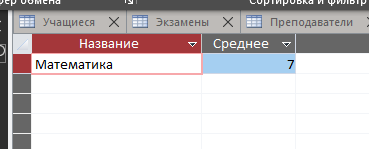


Рисунок 10. Результат выполнения запроса.

**114 Найдите в определенной группе учащихся, которые будут получать повышенную стипендию (сдали сессию на 9 и 10 и имеют средний балл больше 9,1)**

Логически запрос будет состоять из подзапроса и собственно запроса. В подзапросе мы выберем учащихся, которые получили за сессию отметки только 9 и 10, а основным запросом уже из них выберем тех у кого средний балл больше чем 9,1.

Конструктор запроса представлен на рисунке 11.

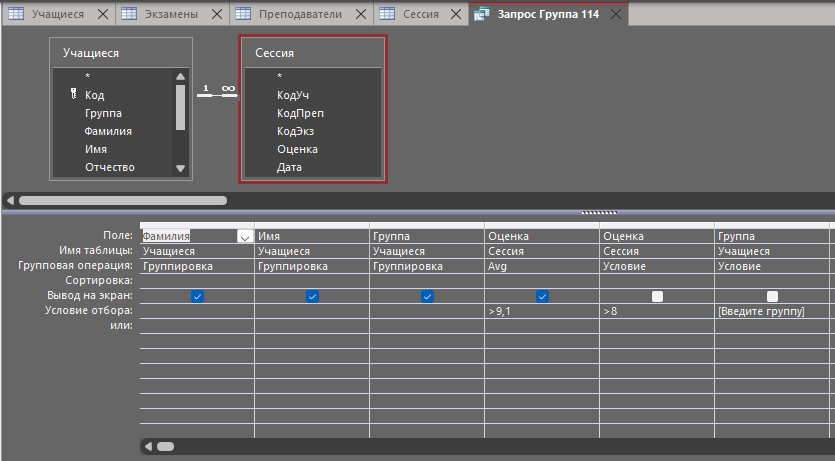


Рисунок 11. Конструктор запроса.

Соответствующий SQL запрос представлен на рисунке 12.

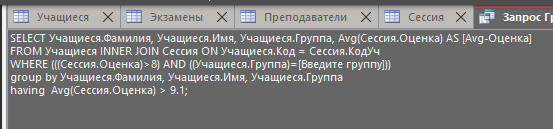


Рисунок 12. Запрос в виде SQL.

Запрос на ввод номера группы представлен на рисунке 13.

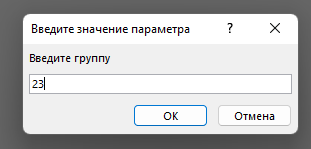


Рисунок 13. Запрос номера группы.

Результат выполнения запроса представлен на рисунке 14.

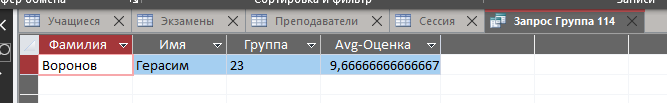


Рисунок 14. Результат выполнения запроса.

**129 Создайте перекрестный запрос «Итоги сессии в группе», содержащий поля «ФИО», «№ зачкн» и названия экзаменов. Номер группы - параметр запроса**

Для начала создадим обычный запрос «ЗапросИтог129».

Конструктор запроса представлен на рисунке 15.

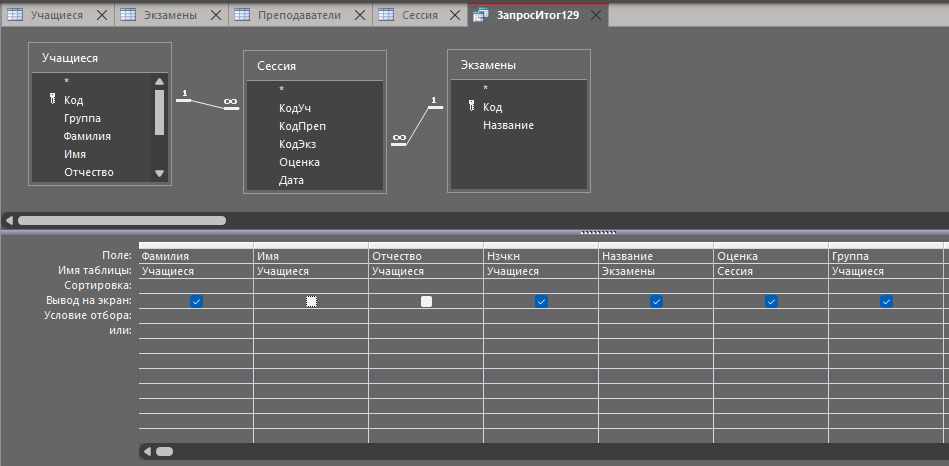


Рисунок 15. Конструктор запроса «ЗапросИтог129»

Таблица запроса представлена на рисунке 16.

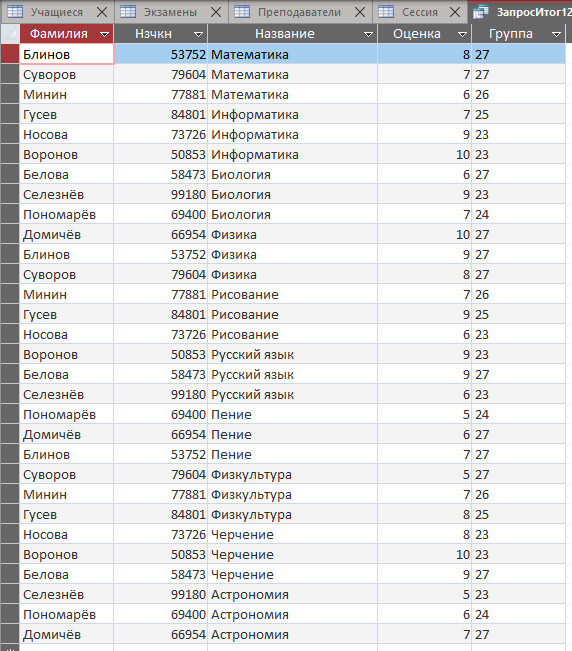


Рисунок 16. Таблица запроса.

На базе этого запроса создадим перекрестный запрос с итогами. Последовательные шаги запроса представлены на рисунках 17 – 22.

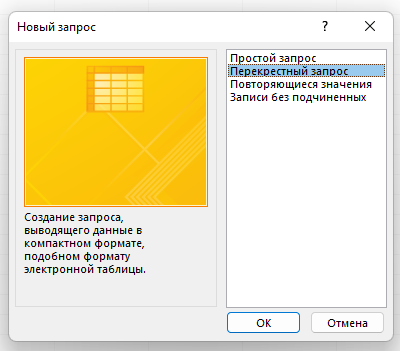


Рисунок 17. Выбор типа запроса.

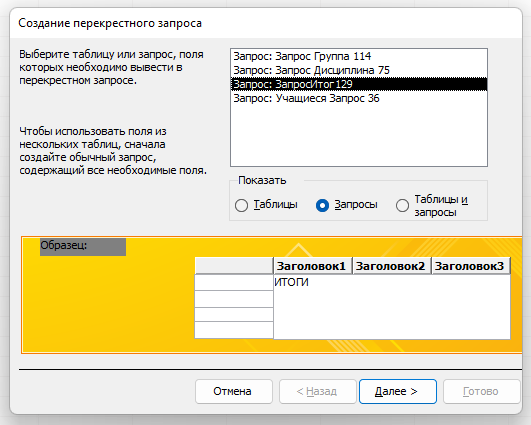


Рисунок 18. Выбор базового запроса

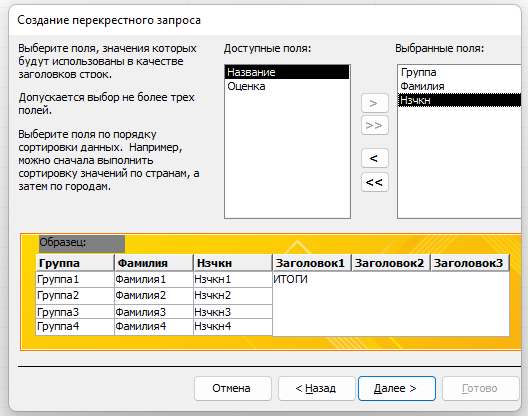


Рисунок 19. Выбор полей заголовков

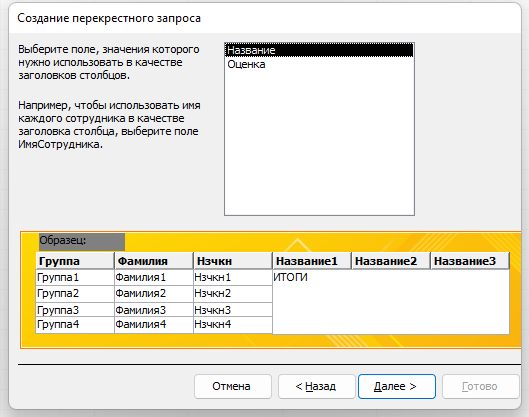


Рисунок 20. Работа с итогами

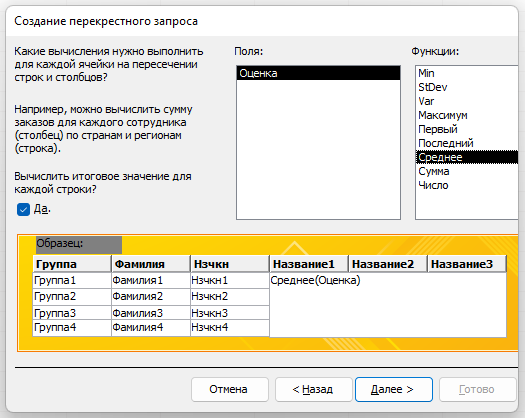


Рисунок 21. Работа с выбором типов итогов.

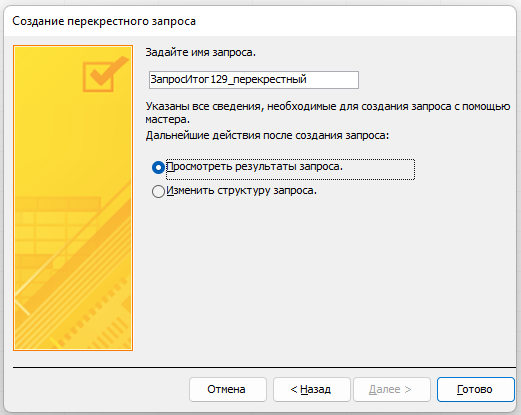


Рисунок 22. Завершение работы мастера.

Результат работы запроса представлен на рисунке 23.

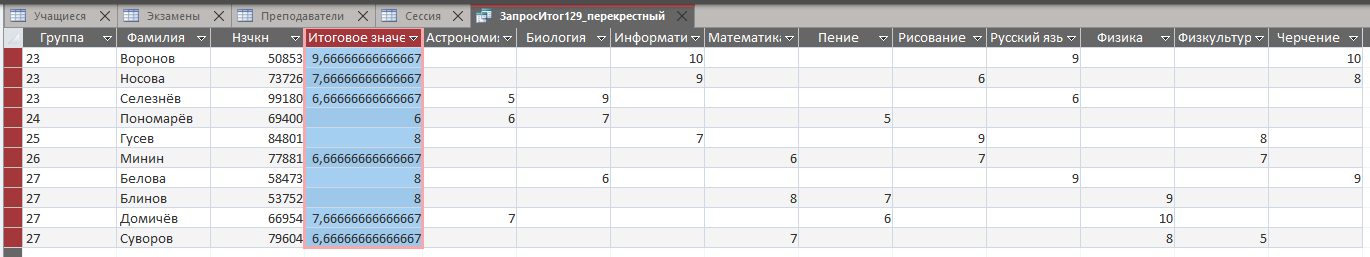
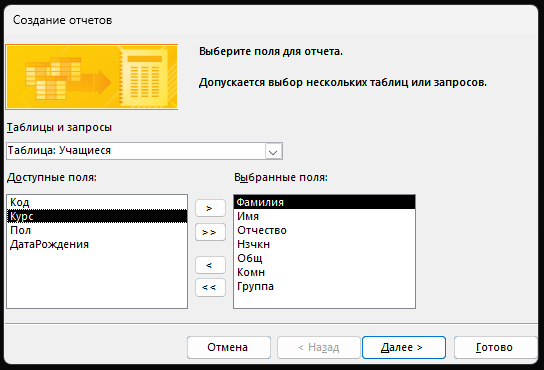
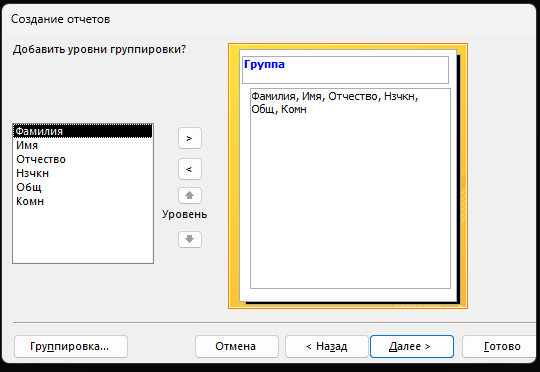


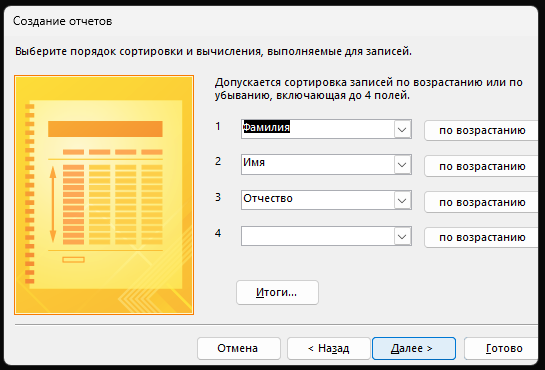
Рисунок 23. Результат работы запроса.

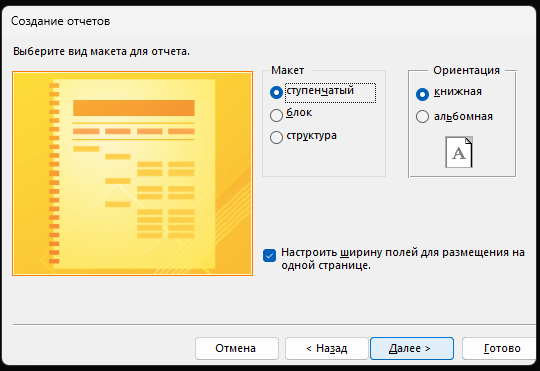
**158 Постройте составную форму «Группы колледжа». Главная форма должна содержать поле с номером группы, a подчиненная форма -поля «№ зачкн», ФИО», «Адрес» и «Дата рождения»**

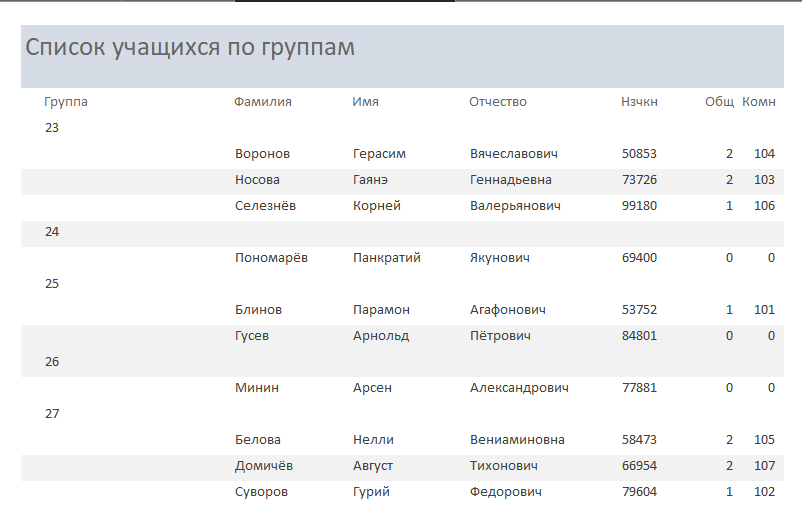
**184 Постройте Отчет «Список учащихся по группам», включающий поля «№ зачкн», «ФИО» «О6щежитие» и «Комната». Он должен быть сгруппирован по учебным группам, а список каждой группы отсортирован по фамилиям учащихся. Номер учебной группы вынести в заголовок группы. После каждой группы указать общее число**











**Список использованных источников**

1 Бен-Ган, И. Microsoft SQL Server 2012. Основы T-SQL / И. Бен-Ган. М. : Эксмо, 2015. 400 с.

2 Дейт, К.Дж. Введение в системы баз данных / К.Дж. Дейт. М. : Вильямс, 2018. 1382 с.

3 Лазицкас, Е.А. Базы данных и системы управления базами данных / Е.А. Лазицкас, И.Н. Загуменникова, П.Г. Гилевский. Минск : РИПО, 2016. 268 с.

4 Федорова, Г. Разработка и администрирование баз данных / Г. Федорова. М. : Академия, 2015. 313 с.

5 SQL справочник / К. Кляйн [и др.]. СПб. : Символ-плюс, 2016. 56 с.