**Проектирование баз данных**

**Управление параллелизмом в СУБД**

Практическая работа *«Создание запросов и фильтров»*

Цель практической работы:

1. Изучить создание запросов и фильтров.
2. Изучить процесс выполнения вычислений при помощи оператора SELECT. Встроенные функции.

Практические задания и рекомендации по их выполнению:

ЗАДАНИЕ

**Создание запросов**

Перейдём к созданию статических запросов. В обозревателе объектов все запросы БД находятся в папке “Views” («Представления») (Рис 34).

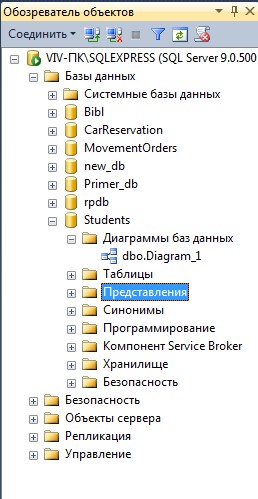


Рисунок 34 — Папка для хранения запросов в обозревателе объектов

Создадим запрос «Студенты+Специальности». Для создания нового запроса необходимо в обозревателе объектов в БД «Students» шёлкнуть правой кнопкой мыши по папке “Views” («Представления»), затем в появившемся меню выбрать пункт “New View” («Создать представление»). Появится окно “Add Table” («Добавление таблицы»), предназначенное для выбора таблиц и запросов, участвующих в новом запросе (Рис. 35).

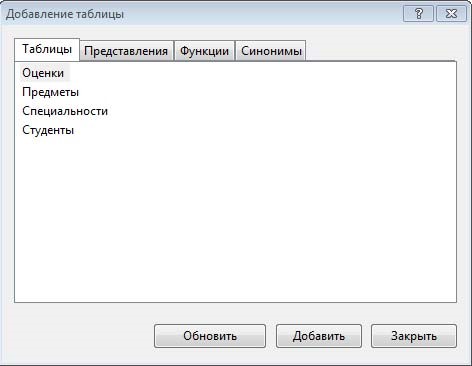
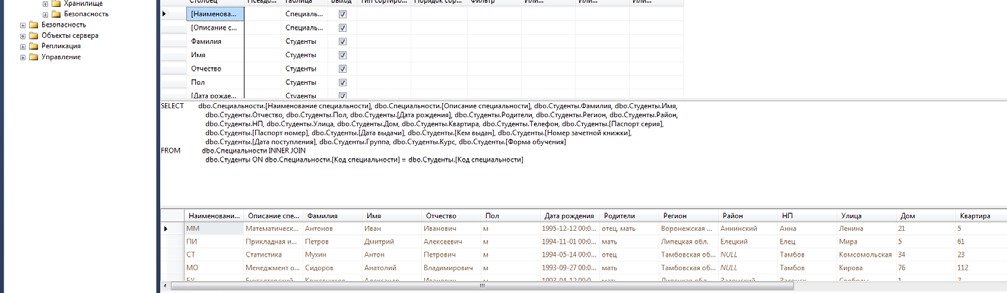
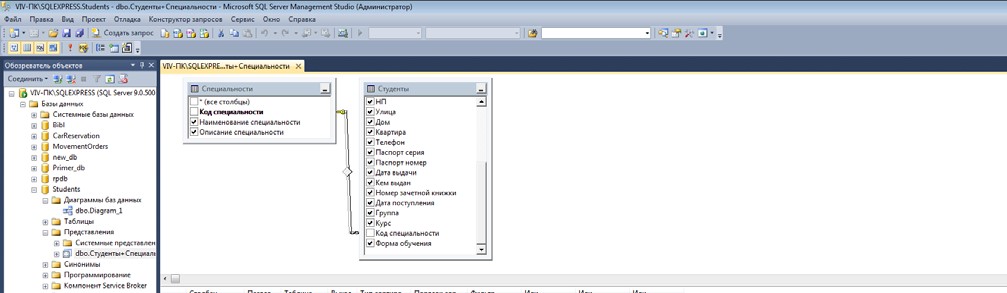


Рисунок 35 — Окно диалога для добавления таблиц и запросов в конструктор запросов

Добавим в новый запрос таблицы «Студенты» и «Специальности». Для этого в окне “Add Table” («Добавление таблицы») выделите таблицу «Студенты» и нажмите кнопку “Add” («Добавить»). Аналогично добавьте таблицу «Специальности». После добавления таблиц участвующих в запросе закройте окно “Add Table” («Добавление таблицы»), нажав кнопку “Close” («Закрыть»). Появится окно конструктора запросов (Рис. 36).

**Замечание**: Окно конструктора запросов состоит из следующих панелей:

1. **Схема данных** — отображает поля таблиц и запросов, участвующих в запросе, позволяет выбирать отображаемые поля, позволяет устанавливать связи между участниками запроса по специальным полям связи. Эта панель включается и выключается следующей кнопкой на панели инструментов ;
2. **Таблица отображаемых полей** — показывает отображаемые поля (столбец “Column” («Столбец»)), позволяет задавать им псевдонимы (столбец “Alias” («Псевдоним»)), позволяет устанавливать тип сортировки записей по одному или нескольким полям (столбец “Sort Type” («Тип сортировки»)), позволяет задавать порядок сортировки (столбец “Sort Orden” («Порядок сортировки»)), позволяет задавать условия отбора записей в фильтрах (столбцы “Filter” («Фильтр») и “Оr...” («Или…»)). Также эта таблица позволяет менять порядок отображения полей в запросе. Эта панель включается и выключается следующей кнопкой на панели инструментов ;



**1**

**2**

**3**

**4**

Рисунок 36 — Окно конструктора запросов

1. **Код SQL** — код создаваемого запроса на языке SQL. Эта панель включается и выключается следующей кнопкой на панели инструментов

;

1. **Результат** — показывает результат запроса после его выполнения. Эта панель включается и выключается следующей кнопкой на панели инструментов .

**Замечание**: Если необходимо снова отобразить окно “Add Table” («Добавление таблицы») для добавления новых таблиц или запросов, то для этого на панели инструментов нужно нажать кнопку .

**Замечание**: Если необходимо удалить таблицу или запрос из схемы данных, то для этого нужно щёлкнуть правой кнопкой мыши и в появившемся меню выбрать пункт “Remove” («Удалить»).

**Замечание**: Если перед созданием запроса таблицы не были связаны в “Database Diagram” («Диаграмма баз данных»), то это можно сделать в окне конструктора запросов. Чтобы создать связь необходимо в схеме данных перетащить мышью поле «Код специальности» таблицы «Специальности» на такое же поле таблицы «Студенты». Связь отобразиться в виде ломаной линии соединяющей эти два поля связи (Рис. 36).

**Замечание**: Если необходимо удалить связь, то для этого необходимо щёлкнуть по ней правой кнопкой мыши и в появившемся меню выбрать пункт “Remove” («Удалить»).

**Замечание**: После связывания таблиц (а также при любых изменениях в запросе) в области кода SQL будет отображаться SQL-код редактируемого запроса.

Теперь определим поля, отображаемые при выполнении запроса. Отображаемые поля обозначаются галочкой (слева от имени поля) на схеме данных, а также отображаются в таблице отображаемых полей. Чтобы сделать поле отображаемым при выполнении запроса необходимо щёлкнуть мышью по пустому квадрату (слева от имени поля) на схеме данных, в квадрате появится галочка.

**Замечание**: Если необходимо сделать поле невидимым при выполнении запроса, то нужно убрать галочку, расположенную слева от имени поля на схеме данных. Для этого просто щёлкните мышью по галочке.

**Замечание**: Если необходимо отобразить все поля таблицы, то необходимо установить галочку слева от пункта “\* (All Columns)” («Все поля»), принадлежащего соответствующей таблице на схеме данных.

Определите отображаемые поля нашего запроса, как это показано на рисунке 36 (Отображаются все поля кроме полей с кодами, то есть полей связи).

На этом настройку нового запроса можно считать законченной. Перед сохранением запроса проверим его работоспособность, выполнив его. Для запуска запроса на панели инструментов нажмите кнопку . Либо щёлкните правой кнопкой мыши в любом месте окна конструктора запросов и в появившемся меню выберите пункт “Execute SQL” («Выполнить код SQL»). Результат выполнения запроса появиться в виде таблицы в области результата (Рис. 36).

**Замечание**: Если после выполнения запроса результат не появился, а появилось сообщение об ошибке, то в этом случае проверьте, правильно ли создана связь.

Если запрос выполняется правильно, то его необходимо сохранить. Для сохранения запроса закройте окно конструктора запросов, щёлкнув мышью по кнопке закрытия , расположенной в верхнем правом углу окна конструктора (над схемой данных). Появится окно с вопросом о сохранении запроса (Рис.

37).

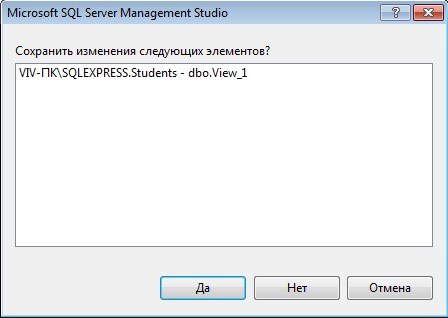


Рисунок 37 — Окно диалога сохранения запроса

В данном окне необходимо нажать кнопку “Yes” («Да»). Появится окно “Choose Name” («Выбор имени») (Рис. 38).

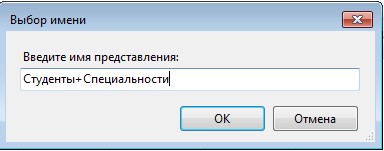


Рисунок 38 — Окно диалога для присвоения имени запросу

В данном окне зададим имя нового запроса «Студенты+Спецнальности» и нажмём кнопку «ОК». Запрос появится в папке “Views” («Представления») БД «Students» в обозревателе объектов (Рис. 39).

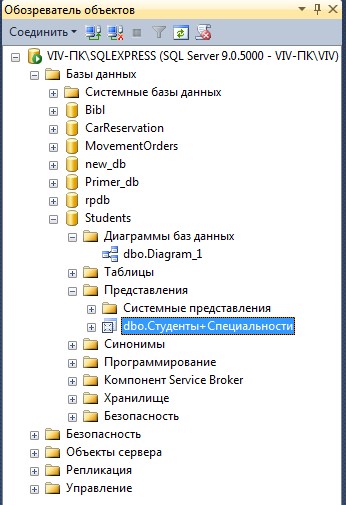


Рисунок 39 — Отображение нового запроса в обозревателе объектов

Проверим работоспособность созданного запроса вне конструктора запросов. Запустим вновь созданный запрос «Студенты+Спеииальности» без использования конструктора запросов. Для выполнения уже сохранённого запроса необходимо щёлкнуть правой кнопкой мыши по запросу и в появившемся меню выбрать пункт “Select top 1000 rows” («Выбрать первые 1000 строк»). Выполните эту операцию для запроса

«Студенты+Специальности». Результат представлен на рисунке 40.

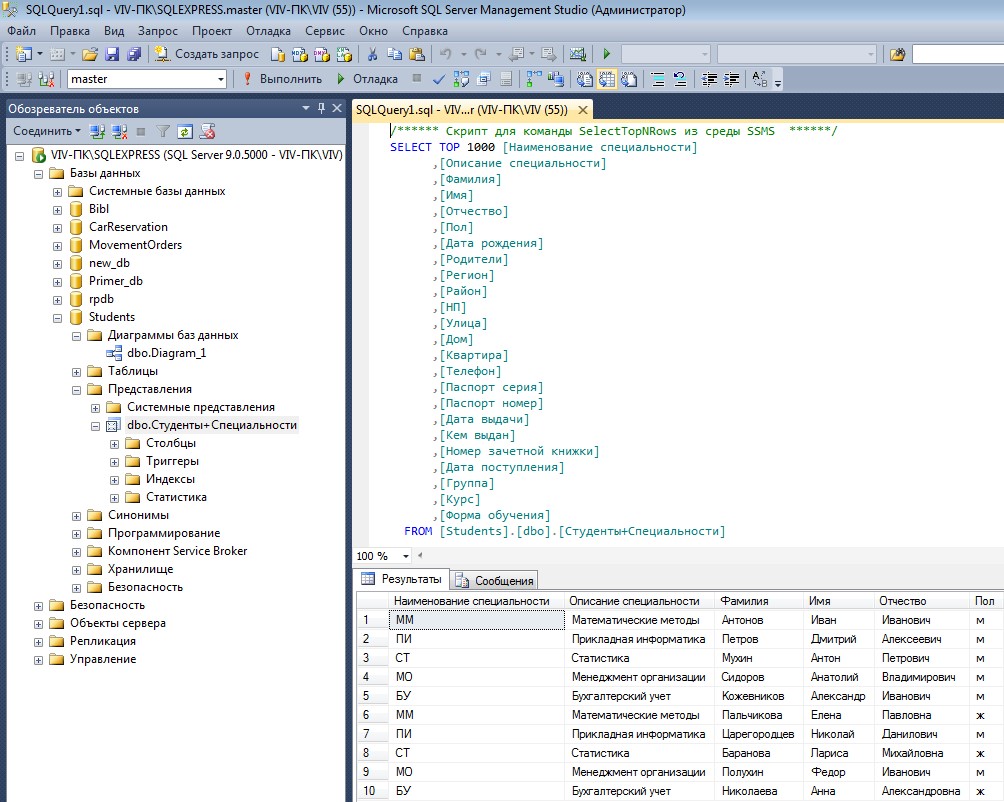


Рисунок 40 — Выполнение запроса вне конструктора запросов

Перейдём к созданию запроса «Студенты+Оценки». В обозревателе объектов в БД «Students» щелкните правой кнопкой мыши по папке “Views” («Представления»), затем в появившемся меню выберите пункт “New View” («Создать представление»). Появится окно “Add Table” («Добавление таблицы») (Рис. 35).

В окне “Add Table” («Добавление таблицы») в новый запрос добавляем таблицы «Студенты», «Предметы» и «Оценки». После добавления таблиц закройте окно “Add Table” («Добавление таблицы»), появится окно конструктора запросов.

В окне конструктора запросов определите отображаемые поля, как показано на рисунке 41.

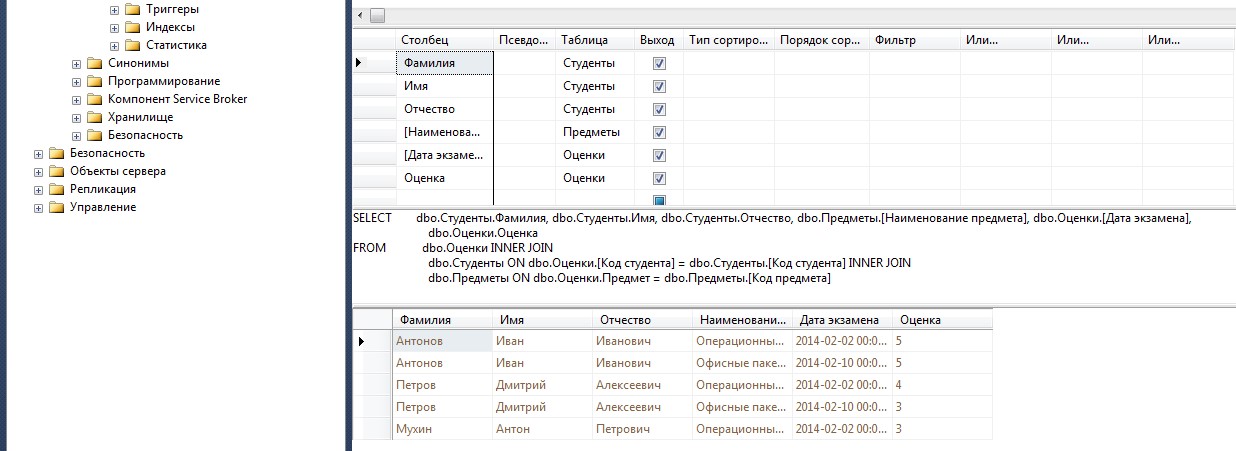
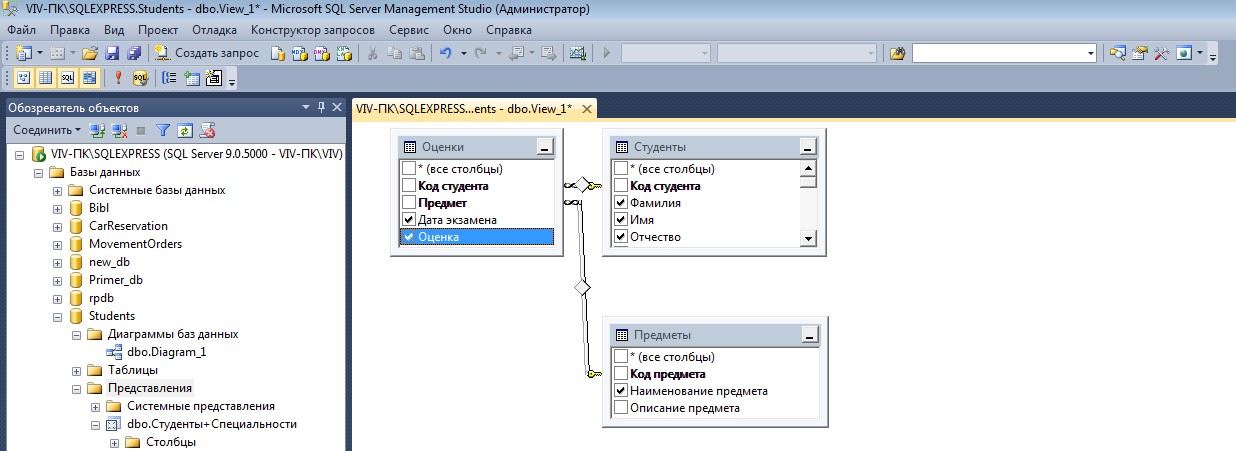


Рисунок 41 — Запрос «Студенты+Оценки» в окне конструктора запросов

Теперь поменяем порядок отображаемых полей в запросе. Для этого в таблице отображаемых полей необходимо перетащить поля мышью вверх или вниз за заголовок строки таблицы (столбец перед столбцом “Column” («Столбец»)). Расположите отображаемые поля в таблице отображаемых полей как показано на рисунке 42.

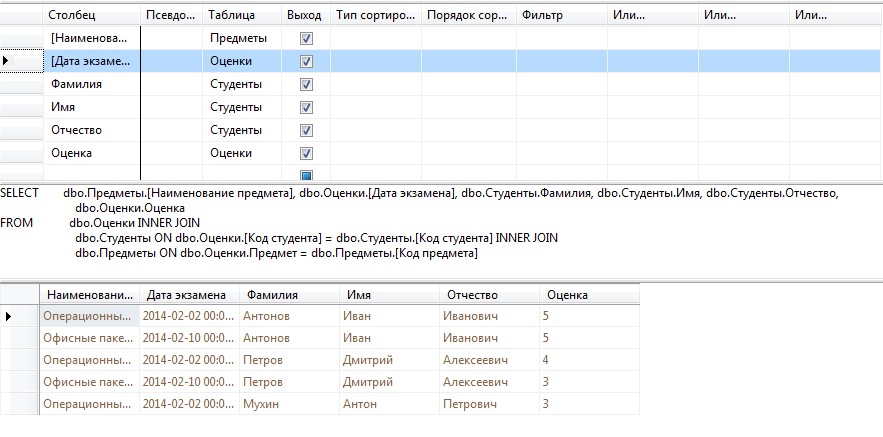


Рисунок 42 — Изменение порядка отображаемых в запросе полей

Задайте псевдонимы для каждого из полей, просто записав псевдонимы в столбце “Alias” («Псевдоним») таблицы отображаемых полей, как на рисунке 43.

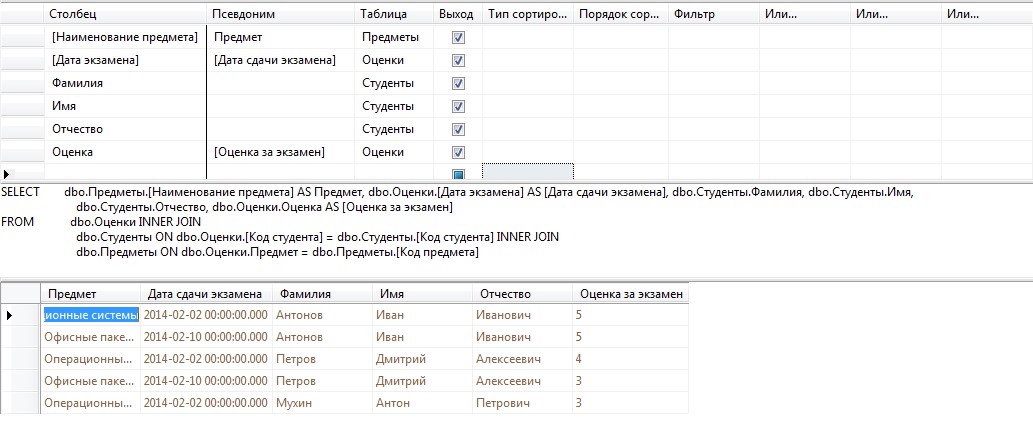


Рисунок 43 — Присвоение псевдонима полю

Проверьте работоспособность нового запроса, выполнив его. Обратите внимание на то, что реальные названия полей были заменены их псевдонимами. Закройте окно конструктора запросов. В появившемся окне “Choose Name” («Выбор имени») задайте имя нового запроса

«Студенты+Оценки» (Рис. 44).

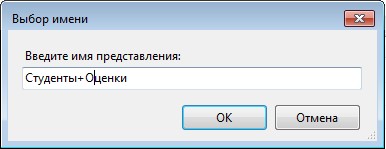


Рисунок 44 — Окно сохранения запроса «Студенты+Оценки»

Проверьте работоспособность нового запроса вне конструктора. Для этого запустите запрос. Результат выполнения запроса «Студенты+Оценки» должен выглядеть как на рисунке 45.

На этом мы заканчиваем рассмотрение обычных запросов и переходим к созданию фильтров.

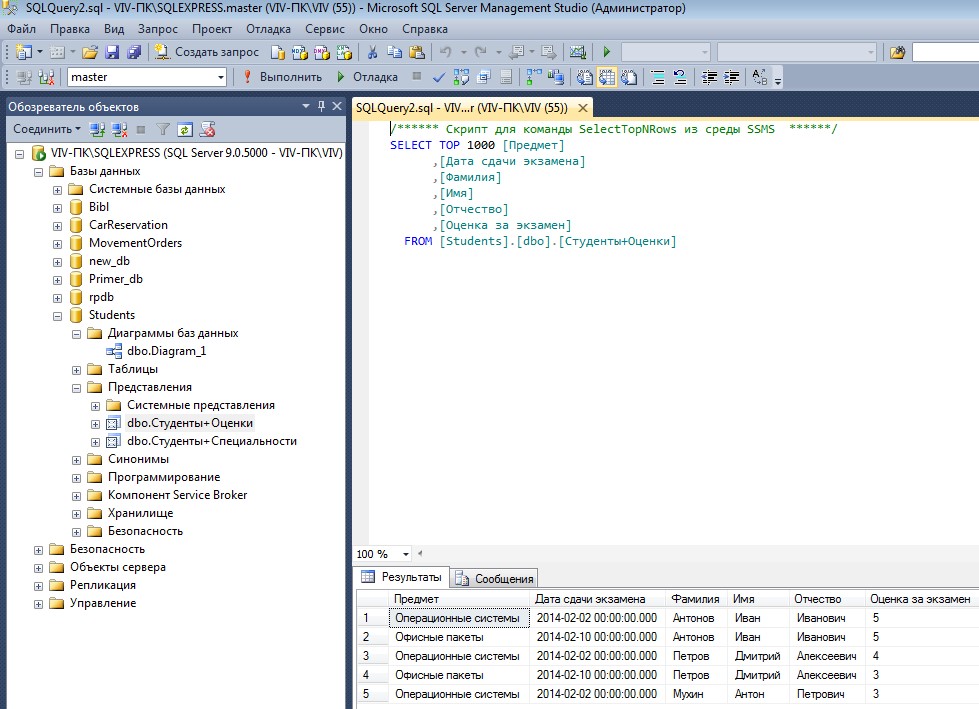


Рисунок 45 — Выполнение запроса «Студенты+Оценки» вне конструктора запросов

На основе запроса «Студенты+Специальности» создадим фильтры, отображающие студентов отдельных специальностей. Создайте новый запрос. Так как он будет основан на запросе «Студенты+Специальности», то в окне “Add Table” («Добавление таблицы») перейдите на вкладку “Views” («Представления») и добавьте в новый запрос «Студенты+Специальности» (Рис. 46). Затем закройте окно “Add Table” («Добавление таблицы»).

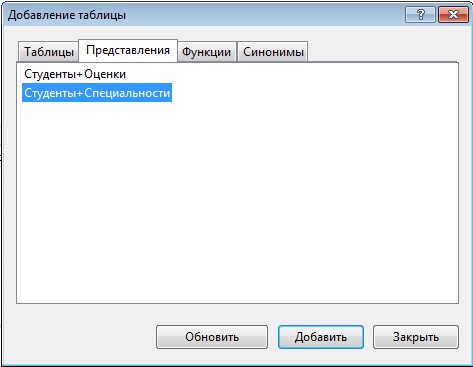


Рисунок 46 — Добавление представления в конструктор запросов

В появившемся окне конструктора запросов определите в качестве отображаемых полей все поля запроса «Студенты+Специальности» (Рис. 47).

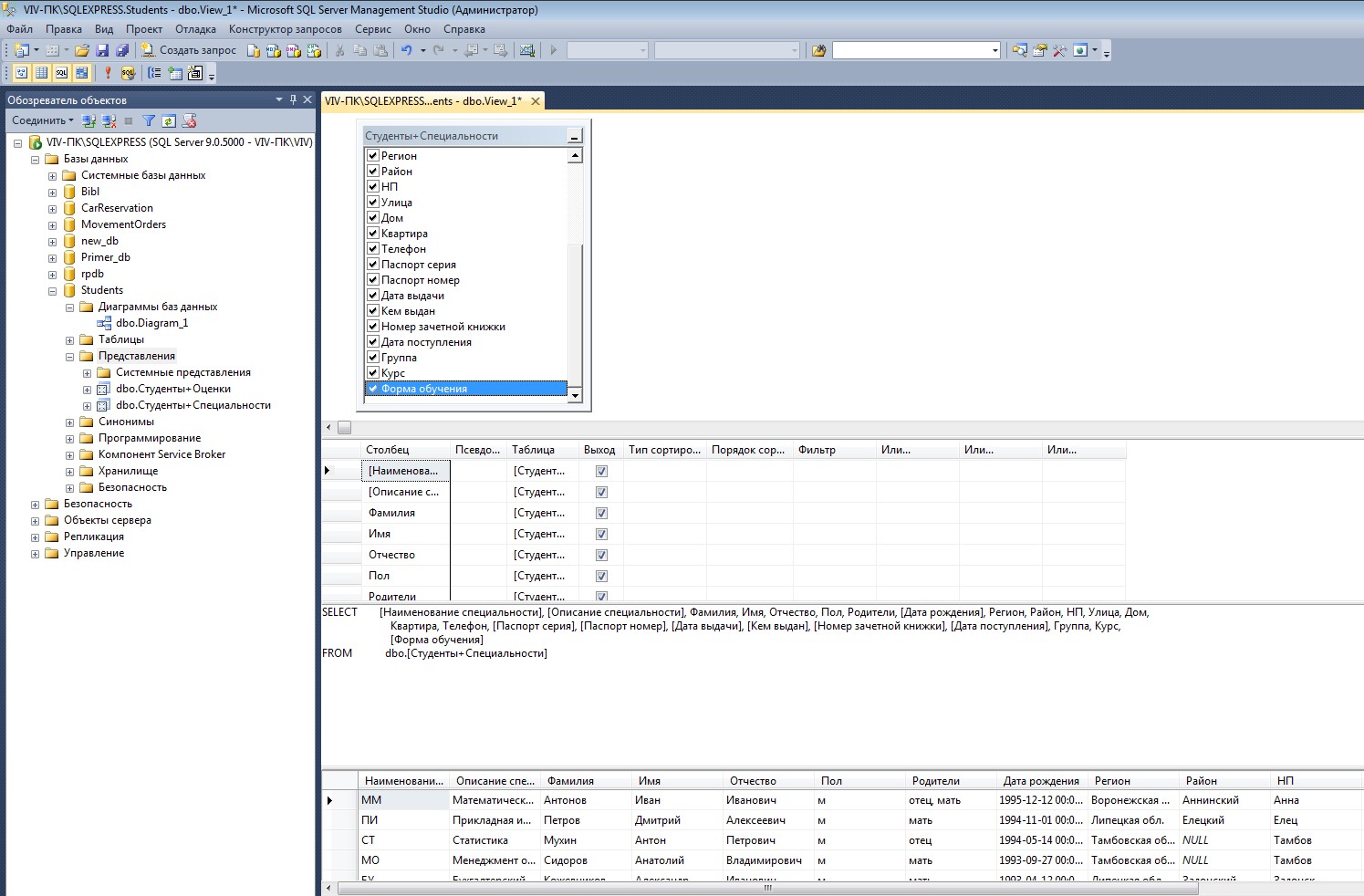


Рисунок 47 — Выбор полей для запроса

**Замечание**: Для отображения всех полей запроса, в данном случае, мы не можем использовать пункт «\* (All Columns)» (Все поля). Так как в этом случае мы не сможем установить критерий отбора записей в фильтре, а также сортировку записей.

Теперь установим критерий отбора записей в фильтре. Пусть наш фильтр отображает только студентов имеющих специальность «ММ». Для определения условия отбора записей в таблице отображаемых полей в строке, соответствующей полю, на которое накладывается условие, в столбце “Filter” («Фильтр»), необходимо задать условие. В нашем случае условие накладывается на поле «Наименование специальности». Следовательно, в строке «Наименование специальности» в столбце “Filter” («Фильтр») нужно задать следующее условие отбора «='ММ'» (Рис. 48).

В заключение настроим сортировку записей в фильтре. Пусть при выполнении фильтра сначала происходит сортировка записей по возрастанию по полю «Форма обучения», а затем по убыванию по полю «Курс». Для установки сортировки записей по возрастанию в таблице определяемых полей в строке для поля «Форма обучения» в столбце “Sort Type” («Тип сортировки»), задайте “Ascending” («По возрастанию»), а в строке для поля «Курс» — задайте “Descending” («По убыванию»). Для определения порядка сортировки для поля «Форма обучения» в столбце “Sort Order” («Порядок сортировки») поставьте 1, а для поля «Курс» поставьте 2 (Рис. 49). То есть при выполнении запроса записи сначала сортируются по полю «Форма обучения», а затем по полю «Курс».

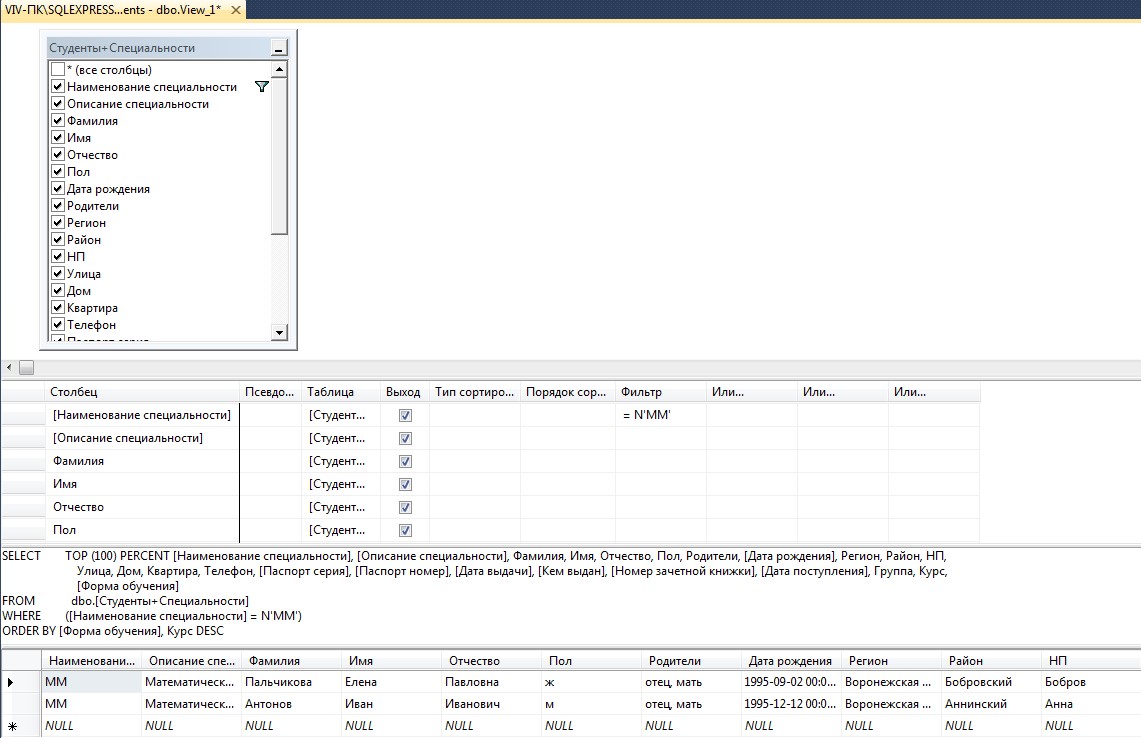


Рисунок 48 — Установка фильтра

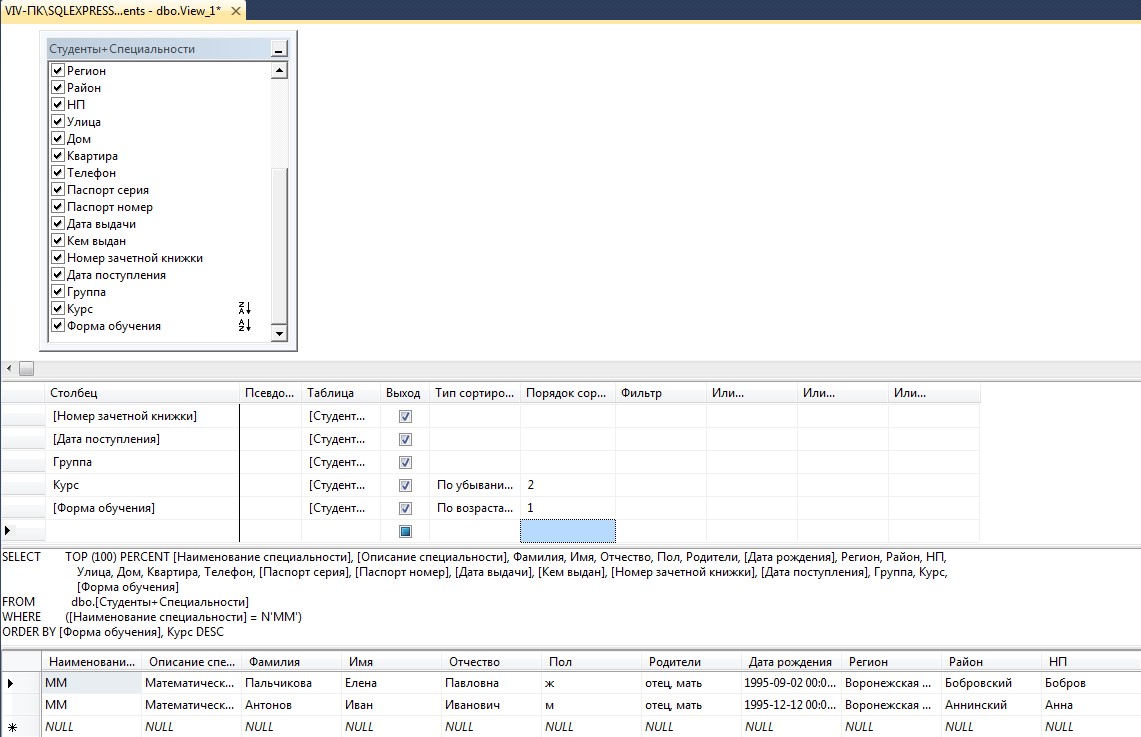


Рисунок 49 — Установка типа и порядка сортировки

**Замечание**: После установки условий отбора и сортировки записей на схеме данных напротив соответствующих полей появятся специальные значки. Значки и обозначают сортировку по возрастанию и убыванию, а значок  показывает наличие условия отбора.



После установки сортировки записей в фильтре проверим его работоспособность, выполнив его. Результат выполнения фильтра должен выглядеть как на рисунке 49. Закройте окно конструктора запросов. В качестве имени нового фильтра в окне “Choose Name” («Выбор имени») задайте «Фильтр ММ» (Рис. 50) и нажмите кнопку «ОК».

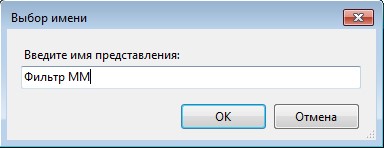


Рисунок 50 — Окно сохранения запроса с фильтром

Представление «Фильтр ММ» появится в обозревателе объектов. Выполните созданный фильтр вне окна конструктора запросов. Результат должен быть таким же, как на рисунке 51.

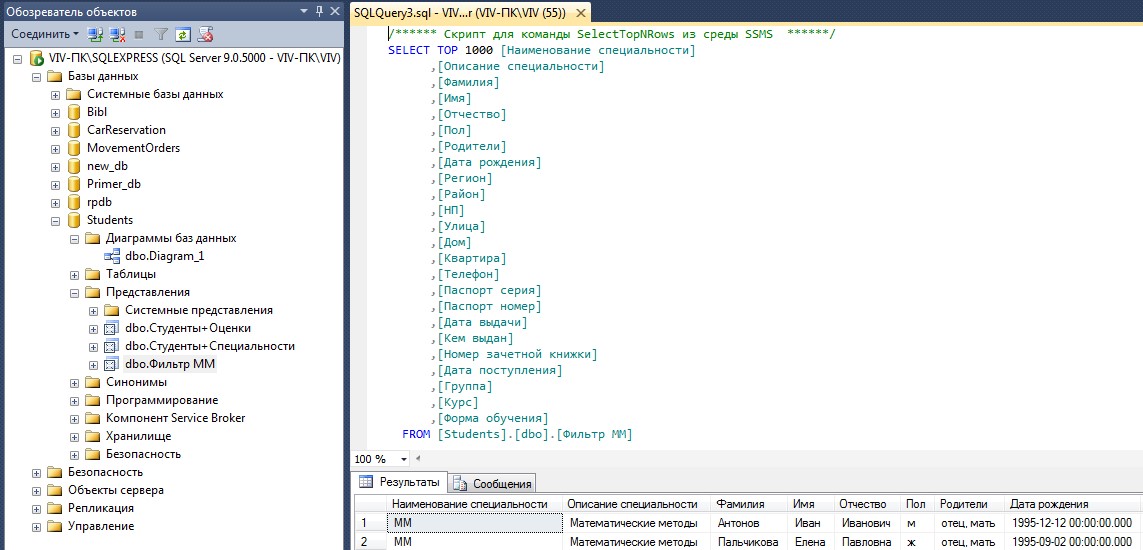


Рисунок 51 — Результат выполнения фильтра вне конструктора запросов

Самостоятельно создайте фильтры для отображения других специальностей. Данные фильтры создаются аналогично фильтру «Фильтр ММ» (смотри выше). Единственным отличием является условие отбора, накладываемое на поле «Наименование специальности», оно должно быть не «=’ММ’», а «=’'ПИ’», «=’СТ’», «=’МО’» или «=‘БУ’». При сохранении фильтров задаём их имена соответственно их условиям отбора, то есть «Фильтр ПИ», «Фильтр СТ», «Фильтр МО» или «Фильтр БУ». Проверьте созданные фильтры на работоспособность.

Теперь на основе запроса «Студенты+Специальности» создадим фильтры, отображающие студентов имеющих отдельных родителей. Для начала создадим фильтр для студентов, у которых из родителей только «отец». Создайте новый запрос и добавьте в него запрос «Студенты+Специальности» (Рис. 46). После закрытия окна “Add Table” («Добавление таблицы») сделайте отображаемыми все поля запроса (Рис. 52).

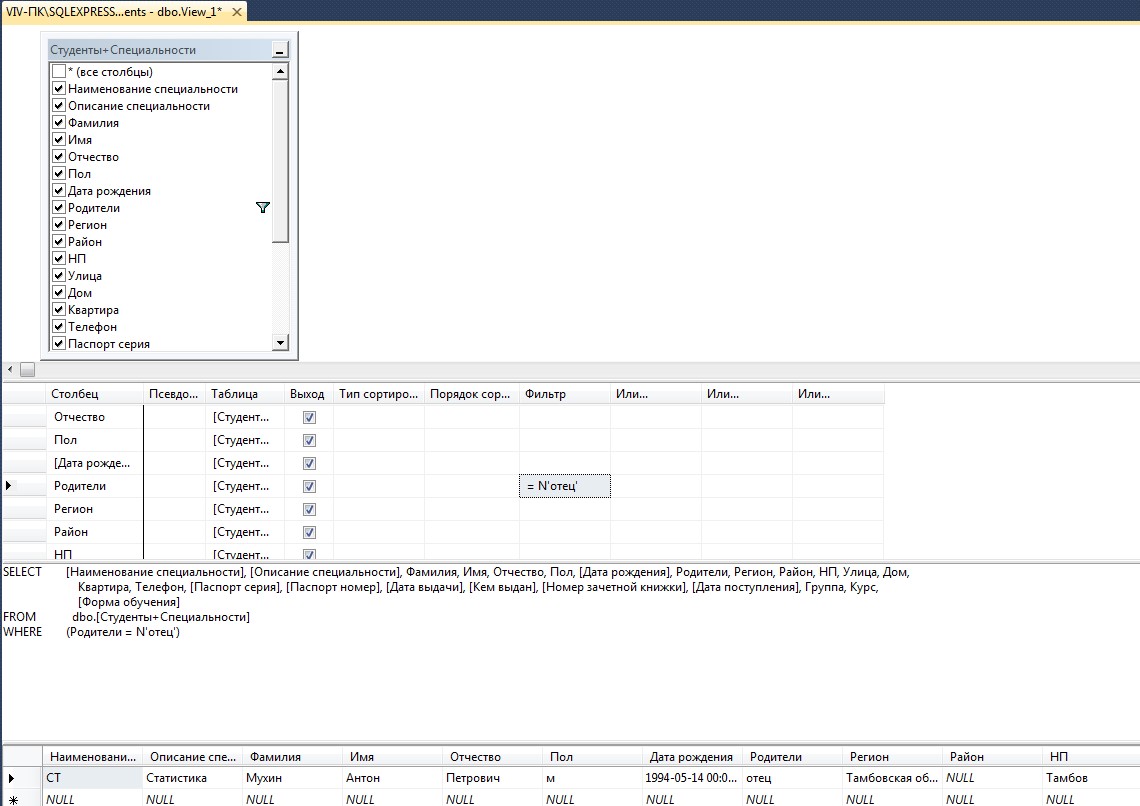


Рисунок 52 — Окно конструктора запросов

В таблице отображаемых полей в строке для поля «Родители» в столбце «Filter» («Фильтр») задайте условие отбора равное «='Отец’». Проверьте работу фильтра, выполнив его. В результате выполнения фильтра окно конструктора запросов должно выглядеть как на рисунке 52.

Закройте окно конструктора запросов. В окне “Choose Name” («Выбор имени») задайте имя нового фильтра как «Фильтр отец» (Рис. 53).

Выполните фильтр «Фильтр отец» вне конструктора запросов. Результат должен быть аналогичен рисунку 54.

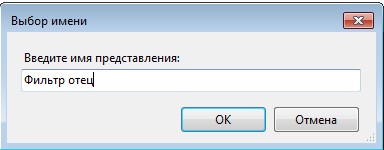


Рисунок 53 — Окно диалога для задания имени фильтру

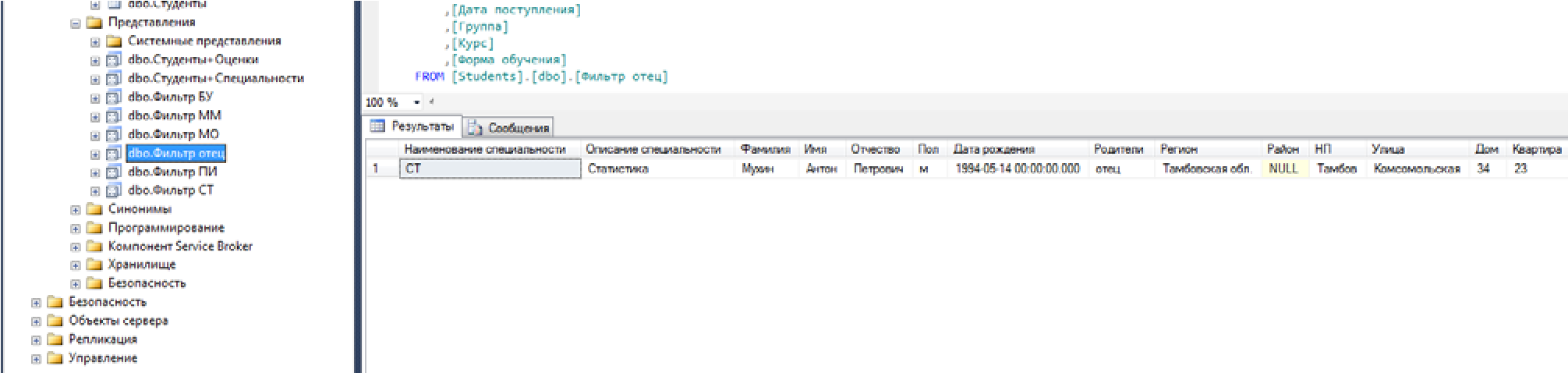


Рисунок 54 — Результат выполнения фильтра «Фильтр отец» вне конструктора запросов

Создайте фильтры для отображения студентов с другими вариантами родителей. Данные фильтры создаются аналогично фильтру «Фильтр отец» (смотри выше). Единственным отличием является условие отбора, накладываемое на поле «Родители», оно должно быть не «=’отец’'», а «='мать'», «='отец, мать"» или «=’Нет’». При сохранении фильтров задаём их имена соответственно их условиям отбора, то есть «Фильтр мать», «Фильтр отец и мать» или «Фильтр нет родителей». Проверьте созданные фильтры на работоспособность.

Наконец, создадим фильтры для отображения студентов очной и заочной формы обучения. Начнём с очной формы обучения. Создайте новый запрос и добавьте в него запрос «Студенты+Специальности». Как и ранее, сделайте все поля запроса отображаемыми (Рис. 55).

В таблице отображаемых полей в столбце “Filter” («Фильтр») в строке для поля «Форма обучения» установите условие отбора равное «=1».

**Замечание**: Поле «Форма обучения» является логическим полем, оно может принимать значения либо “True” («Истина»), либо “False” («Ложь»). В качестве синонимов этих значений в Microsoft SQL Server можно использовать 1 и 0 соответственно.

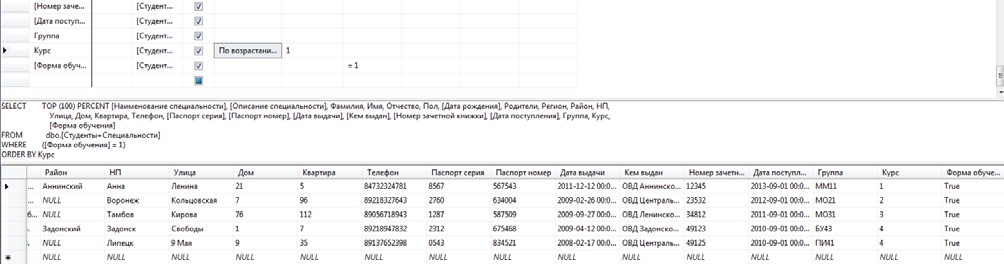
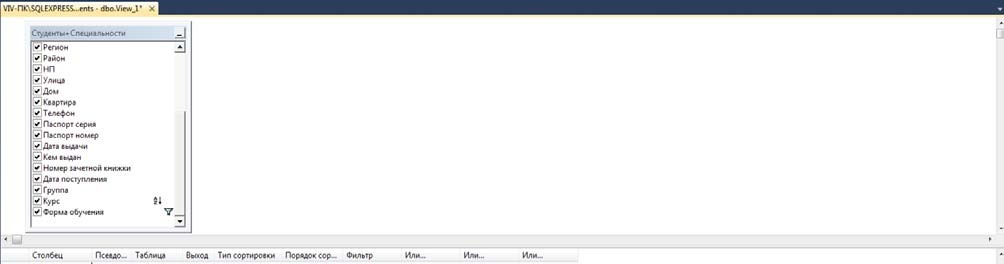


Рисунок 55 — Окно конструктора запросов

Установите сортировку по возрастанию по полю «Курс», задав в строке для этого поля в столбце “Sort Type” («Тип сортировки») значение «Ascending» («по возрастанию»).

Проверьте работу фильтра, выполнив его. После выполнения фильтра окно конструктора запросов должно выглядеть точно также как на рисунке 55.

Закройте окно конструктора запросов. Сохраните фильтр под именем «Фильтр очная форма обучения» (Рис. 56).

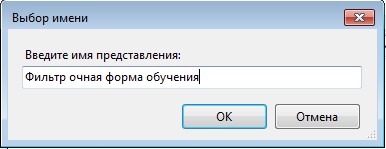


Рисунок 56 — Окно сохранения фильтра

После появления фильтра «Фильтр очная форма обучения» в обозревателе объектов выполните фильтр вне окна конструктора запросов. Результат выполнения фильтра «Фильтр очная форма обучения» представлен на рисунке 57.

Самостоятельно создайте фильтр для отображения студентов заочной формы обучения. Данный фильтр создаётся точно также, как и фильтр «Фильтр очная форма обучения». Единственным отличием является условие отбора, накладываемое на поле «Форма обучения». Оно должно быть не «=1», а «=0». При сохранении фильтра задайте его имя как «Фильтр заочная форма обучения». Проверьте созданный фильтр на работоспособность.

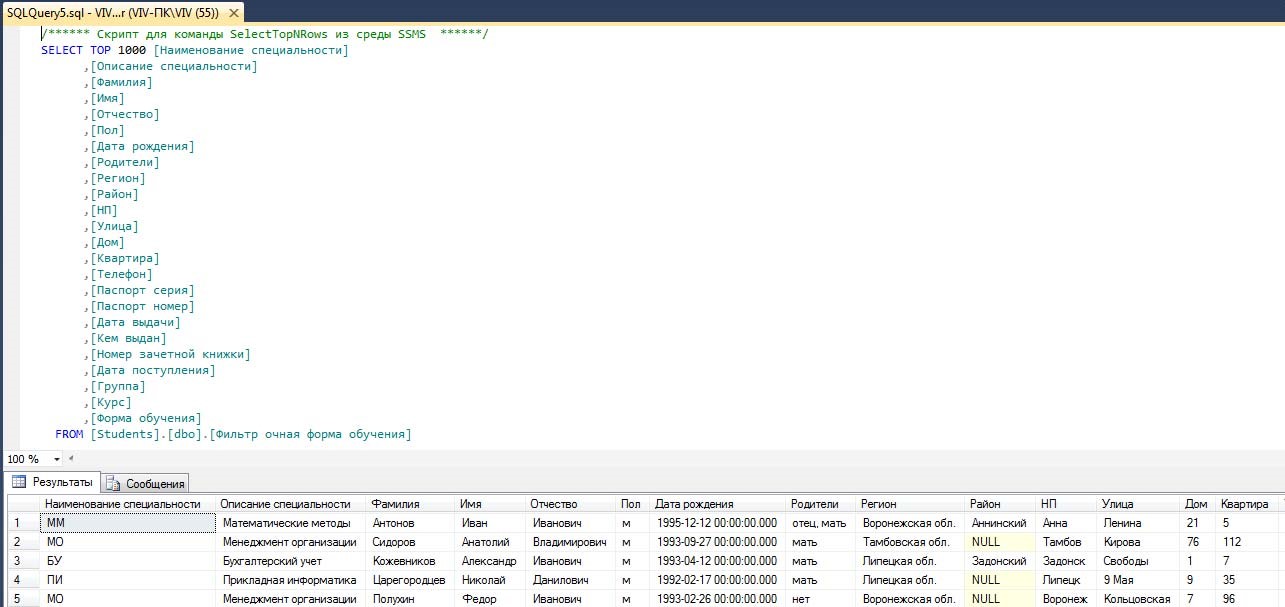


Рисунок 57 — Результат выполнения фильтра вне конструктора запросов

В итоге, после создания всех запросов и фильтров окно обозревателя объектов должно выглядеть следующим образом (Рис. 58).

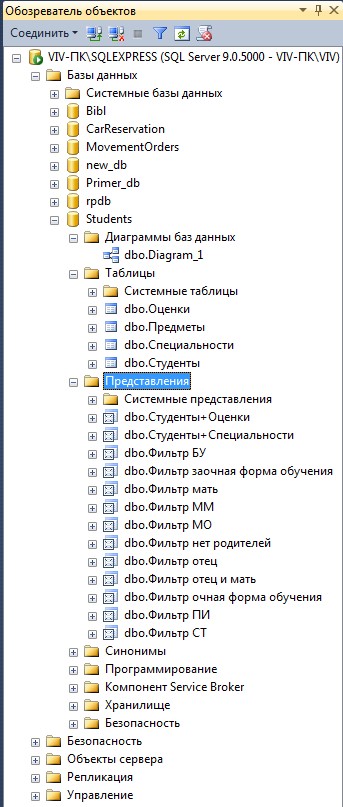


Рисунок 58 — Окно обозревателя объектов со всеми созданными объектами Контрольные вопросы:

1. Что такое запрос?
2. Как создаются запросы?
3. Для чего используются фильтры?
4. В чем состоит процесс выполнения вычислений при помощи оператора SELECT?
5. Какие встроенные функции могут использоваться в операторе SELECT?

Практическая работа «*Хранимые процедуры».*

Цель практической работы:

1. Изучить процесс создания динамических запросов при помощи хранимых процедур.
2. Научиться работать с хранимыми процедурами.

Практические задания и рекомендации по их выполнению:

ЗАДАНИЕ

**Хранимая процедура** — SQL запрос, который имеет параметры, то есть он выполняется как обычная процедура (мы задаем ее имя и передаем в хранимую процедуру значение параметров.) В зависимости от значения параметров хранимой процедуры мы получаем тот или иной результат запроса.

**Замечание**. В SQL-сервере хранимые процедуры реализуют динамические запросы, выполняемые на стороне сервера.

Перейдём к созданию хранимых процедур. Для работы с хранимыми процедурами в обозревателе объектов необходимо выделить папку “Programmability / Stored Procedures” («Программирование / Хранимые процедуры») базы данных Students (рис. 59).

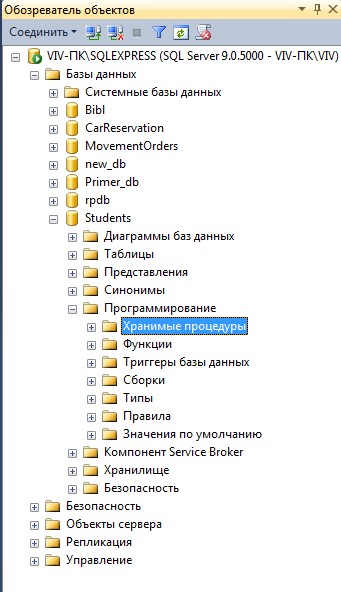
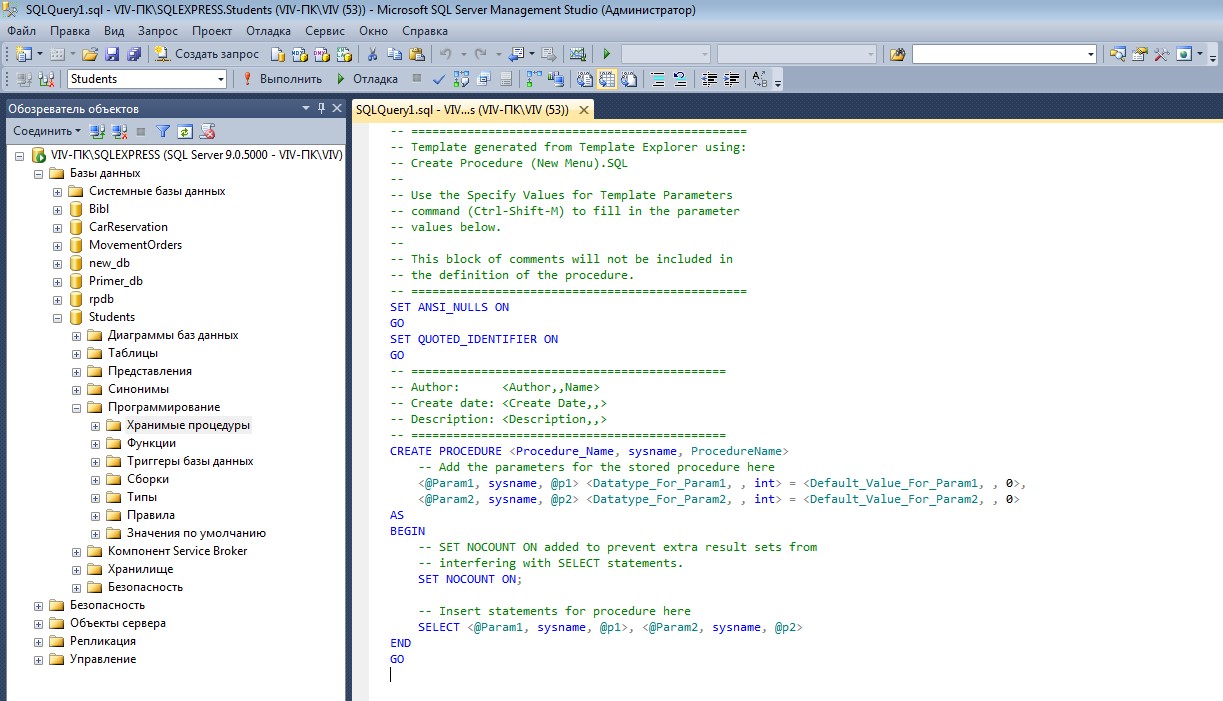


Рисунок 59 — Отображение папки Программирование / Хранимые процедуры в обозревателе объектов

Создадим процедуру, вычисляющую среднее трёх чисел. Для создания новой хранимой процедуры щёлкните правой кнопкой мыши по папке “Stored Procedures” («Хранимые процедуры») (рис.59) и в появившемся меню выберите пункт “New Stored Procedure” («Создать хранимую процедуру…»). Появится окно кода новой хранимой процедуры (рис. 60).

1

2

3

4

5

Рисунок 60 — Окно новой хранимой процедуры

Хранимая процедура имеет следующую структуру (рис. 60):

1. Область настройки параметров синтаксиса процедуры. Позволяет настраивать некоторые синтаксические правила, используемые при наборе кода процедуры. В нашем случае это:
   * SET ANSI\_NULLS ON — включает использование значений NULL (Пусто) в кодировке ANSI,
   * SET QUOTED\_IDENTIFIER ON — включает возможность использования двойных кавычек для определения идентификаторов;
2. Область определения имени процедуры (Procedure\_Name) и параметров, передаваемых в процедуру (@Paraml, @Param2). Определение
3. параметров имеет следующий синтаксис:

@<Имя параметра> <Тип данных> = Значение по умолчанию> Параметры разделяются между собой запятыми;

1. Начало тела процедуры, обозначается служебным словом BEGIN;
2. Тело процедуры, содержит команды языка программирования запросов SQL;
3. Конец тела процедуры, обозначается служебным словом END.

**Замечание:** В коде зелёным цветом выделяются комментарии. Они не обрабатываются сервером и выполняют функцию пояснений к коду. Строки комментариев начинаются с подстроки «--».

Наберём код процедуры, вычисляющей среднее трёх чисел, как это показано на рисунке 61.

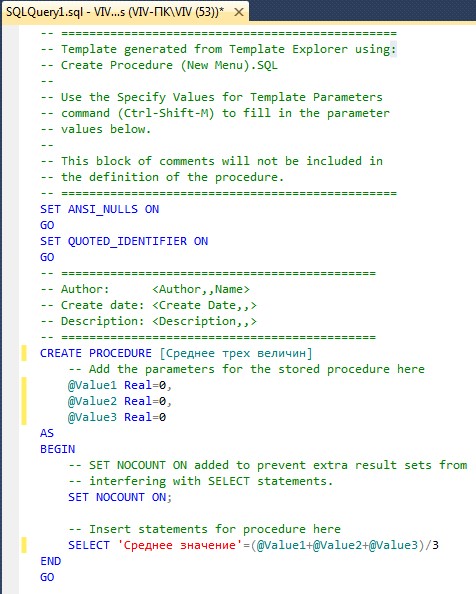


Рисунок 61 — Процедура вычисления среднего трех чисел

Рассмотрим код данной процедуры более подробно (рис. 61):

1. CREATE PROCRDURE [Среднее трёх величин] — определяет имя создаваемой процедуры как «Среднее трёх величин»;
2. @Value1 Real = 0, @Value2 Real = 0, @Value3 Real = 0 — определяют три параметра процедуры Value1, Value2 и Value3. Данным параметрам можно присвоить дробные числа (Тип данных Real), значения по умолчанию равны 0;
3. SELECT 'Среднее значение' = (@Valuel + @Value2 + @Value3)/3 — вычисляет среднее и выводит результат с подписью «Среднее значение».

Остальные фрагменты кода рассмотрены выше (рис. 60).

Для создания процедуры выполним вышеописанный код, нажав кнопку

) на панели инструментов. В нижней части окна с кодом



(



появится сообщение “Conimand(s) completed successfully.” («Выполнение команд успешно завершено.»). Закройте окно с кодом, щёлкнув мышью по кнопке закрытия , расположенной в верхнем правом углу окна с кодом процедуры. После появления окна сохранения процедуры, ответим “Yes” («Да»), в появившемся окне сохраним запрос в свою папку под именем «Среднее трех величин».

Проверим работоспособность созданной хранимой процедуры. Для запуска хранимой процедуры необходимо создать новый пустой запрос, нажав на кнопку ) на панели инструментов. В появившемся окне с пустым запросом наберите команду EXEC [Среднее трёх



(



величин] 1, 7, 9 и нажмите кнопку ) на панели инструментов (рис. 62).



(



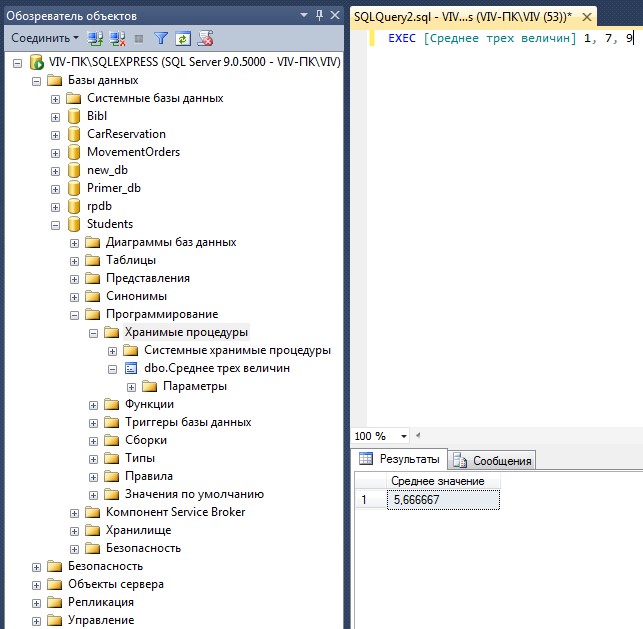


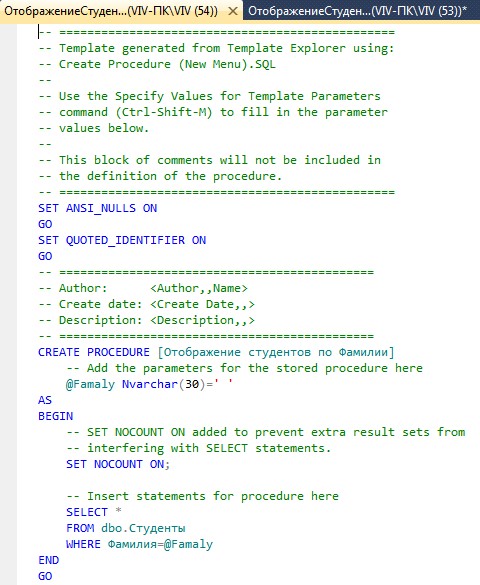
Рисунок 62 — Запуск хранимой процедуры

В нижней части окна с кодом появится результат выполнения новой хранимой процедуры: Среднее значение 5,66667 (рис. 62).

Закроем окно запроса и сохраним его под именем

«СреднееТрехВеличин1». Также процедура будет сохранена в папку “Stored Procedures” («Хранимые процедуры») под именем «dbo.Среднее трех величин».

Теперь создадим хранимую процедуру для отбора студентов из таблицы Студенты по их Фамилии. Для этого создайте новую хранимую процедуру, как это описано выше, и наберите код новой процедуры как на рисунке 63.

1

2

3

Рисунок 63 — Текст процедуры для отбора студентов по Фамилии из таблицы Студенты

Рассмотрим код процедуры «Отображение студентов по Фамилии» более подробно (рис. 63):

1. CREATE PROCRDURE [Отображение студентов по Фамилии] — определяет имя создаваемой процедуры как «Отображение студентов по Фамилии»;
2. @Famaly Varchar (30) =' ' — определяют единственный параметр процедуры Famaly. Параметру можно присвоить текстовые сроки переменной длины длиной до 30 символов (Тип данных Varсhar(30)), значения по умолчанию равны пустой строке;
3. SELECT \* FROM dbo.Студенты WHERE ФИО = @Famaly — отображает все поля (\*) из таблицы Студенты (dbo.Студенты), где значение поля Фамилия равно значению параметра Famaly (Фамилия = @Famaly).

Выполним вышеописанный код и закроем окно с кодом, как описано выше.

Проверим работоспособность созданной хранимой процедуры. Создайте новый пустой запрос. В появившемся окне с пустым запросом наберите команду EXEC [Отображение студентов по Фамилии] “Антонов” и нажмите

кнопку ) на панели инструментов (рис. 64).



(



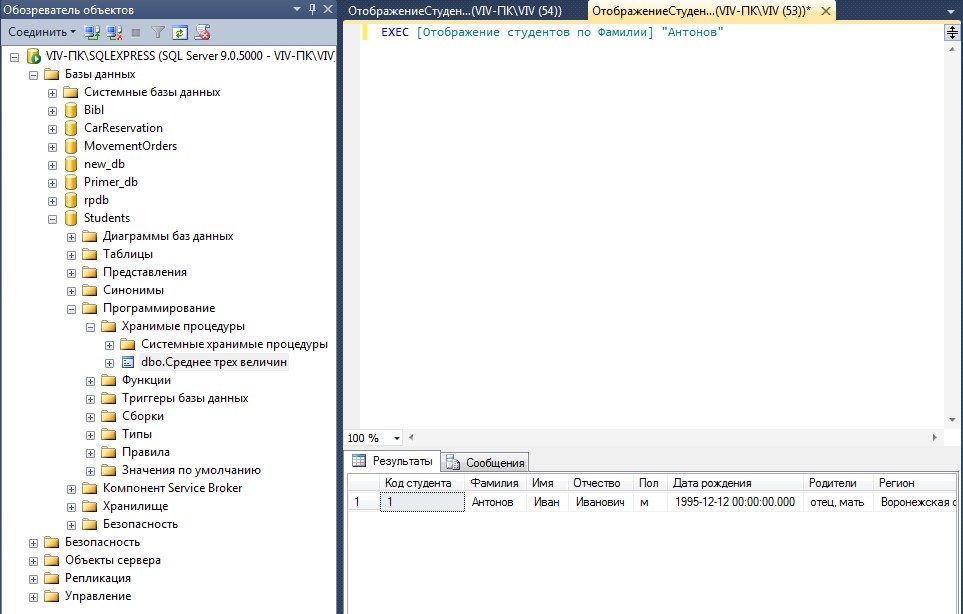
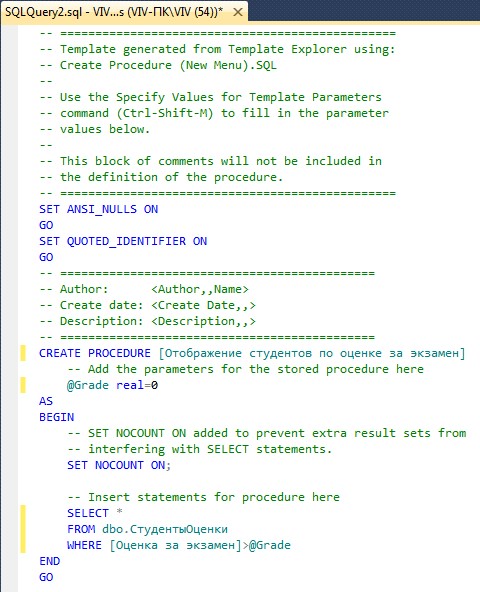


Рисунок 64 — Результат выполнения процедуры «Отображение студентов по Фамилии»

В нижней части окна с кодом появится результат выполнения хранимой процедуры «Отображение студентов по Фамилии» (Рис. 64).

Теперь перейдём к более сложной задаче — отобразить студентов, у которых оценка за экзамен выше заданной. Создайте новую хранимую процедуру и наберите код новой процедуры как на рисунке 65.

|  |
| --- |
| 1 |

2

3

Рисунок 65 — Текст хранимой процедуры «Отображение студентов по оценке за экзамен»

Рассмотрим код процедуры «Отображение студентов по оценке за экзамен» более подробно (рис. 65):

1. CREATE PROCRDURE [Отображение студентов по оценке за экзамен] — определяет имя создаваемой процедуры как «Отображение студентов по оценке за экзамен»;
2. @Grade Real = 0 — определяет параметр процедуры Grade. Параметру можно присваивать дробные числа (тип данных Real), значение по умолчанию равно 0;
3. SELECT \* FROM dbo.СтудентыОценки WHERE [Оценка за экзамен] > @Grade — отображает все поля (\*) из запроса dbo.СтудентыОценки, где оценка больше, чем значение параметра Grade ([Оценка за экзамен] > @Grade).

Выполним вышеописанный код и закроем окно с кодом, как описано выше. Проверим, как работает запрос, описанный выше. Для этого, создайте новый запрос и в нём наберите команду EXEC [Отображение студентов по оценке за экзамен] 3 и выполните её (Смотри выше) (рис. 66).

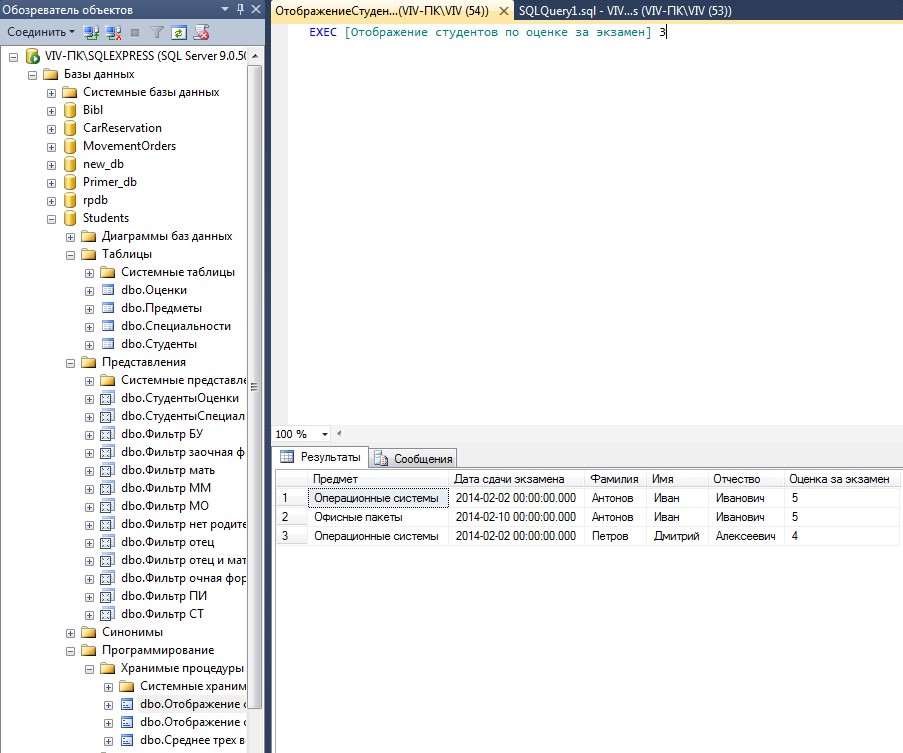


Рисунок 66 — Результат выполнения процедуры «Отображение студентов по оценке за экзамен»

В нижней части окна с кодом появится результат выполнения хранимой процедуры «Отображение студентов по оценке за экзамен» (рис. 66).

В заключение решим более сложную задачу — отображение студентов старше заданного возраста. Причём возраст будет автоматически вычисляться в зависимости от даты рождения.

Создадим новую хранимую процедуру и наберём код новой процедуры, как представлено на рисунке 67.

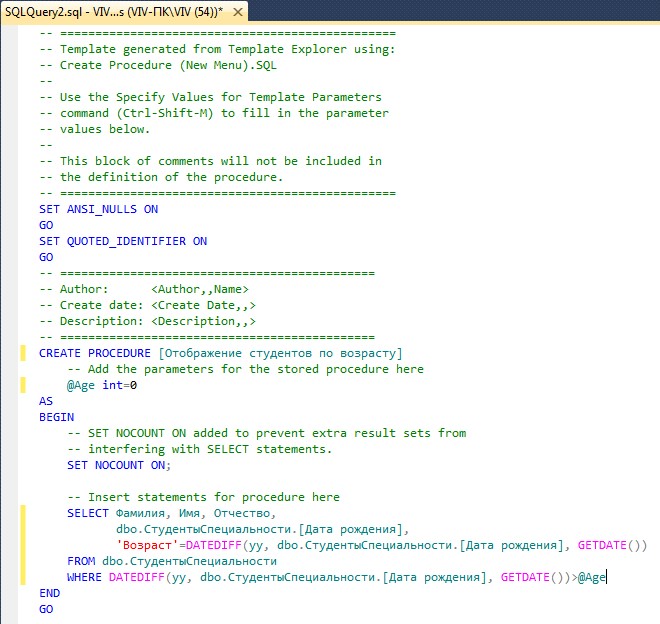


Рисунок 67 — Текст хранимой процедуры «Отображение студентов по возрасту»

Рассмотрим код создаваемой процедуры «Отображение студентов по возрасту» более подробно (рис. 67):

1. CREATE PROCRDURE [Отображение студентов по возрасту] — определяет имя создаваемой процедуры как «Отображение студентов по возрасту»;
2. @Age int = 0 — определяет параметр процедуры Age. Параметру можно присваивать целые числа (тип данных int), значение по умолчанию равно 0;
3. Фамилия, Имя, Отчество, dbo.СтудентыСпециальности.[Дата рождения], ‘Возраст’ = DATEDIFF(уу, dbo.СтудентыСпециальности.[Дата рождения], GETDATE()) — отображает из запроса dbo.СтудентыСпециальности (FROM dbo.СтудентыСпециальности) поля

Фамилия, Имя, Отчество и «Дата рождения» (dbo.СтудентыСпециальности.[Дата рождения]), а также отображает возраст студента ('Возраст') в годах (уу), вычисленный исходя из его даты рождения и текущей даты (DATEDIFF(yy, dbo.СтудентыСпециальности.[Дата рождения], GETDATE()). Более того, выводятся студенты, возраст которых больше определённого в параметре «Age» (DATEDIFF(yy, dbo.СтудентыСпециальности.[Дата рождения], GETDATE()) > @Age).

**Замечание:** Встроенная функция DATEDIFF, вычисляющая количество периодов между двумя датами, имеет следующий синтаксис: DATEDIFF (<период>,<начальная дата>, <конечная дата>).

Выполним код запроса «Отображение студентов по возрасту», а затем закроем окно с кодом, как описано выше. Проверим, как работает запрос. Для этого создадим новый запрос и в нём наберём команду EXEC [Отображение студентов по возрасту] 20 и выполним её. Должен появиться результат аналогичный результату, представленному на рис. 68.

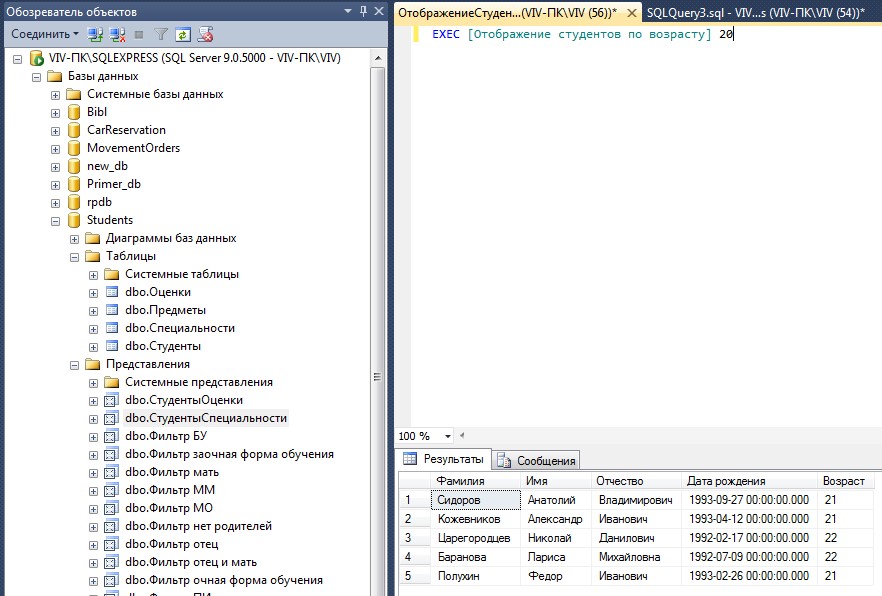


Рисунок 68 — Результат выполнения хранимой процедуры

«Отображение студентов по возрасту»

Контрольные вопросы:

1. В чем отличие динамических запросов?
2. Как создаются динамические запросов при помощи хранимых процедур?
3. Как можно работать с хранимыми процедурами?
4. Какие параметры у CREATE PROCRDURE? 5) Какой тип у CREATE PROCRDURE?