Лабораторная работа № 1.1 Язык SQL. Выборка из одной таблицы. Выборка с добавлением. Выбор уникальных строк. Извлечение диапазона строк. Сортировка.

1. Теоретические сведения.

1.1 Основные понятия языка Transact-SQL.

Transact-SQL структурированный язык запросов, являющийся инструментальным средством для описания, управления и манипулирования реляционными данными в СУБД Microsoft SQL Server.

Идентификатор. Любой объект в базе данных должен быть идентифицирован. Для этого ему присваивают имя

Правила задания идентификатора:

- Длина идентификатора от 1 до 128 символов.
- Идентификатор может включать любые символы, определенные стандартом Unicode Standard 2.0 кроме запрещенных символов.
- В идентификатор не должны входить слова, являющиеся зарезервированными.
- Первым символом в идентификаторе может быть: буква алфавита или символ подчеркивания «_».
- Для обозначения **временных объектов (переменных и параметров)** разрешается использовать в качестве первого символа «@».
- Transact-SQL не различает регистр. Например, идентификатор *Фамилия* соответствует идентификатору *фамилия*.

Типы идентификаторов:

- @ - идентификатор локальной переменной (пользовательской).
- @@ - идентификатор глобальной переменной (встроенной).
- # - идентификатор локальной таблицы или процедур
- ## - идентификатор глобальной таблицы или процедуры.
- [ ] - идентификатор группировки слов в переменную.

Идентификатор, заданный в соответствии с этими правилами называется **обычным идентификатором**. Кроме обычных идентификаторов можно задавать идентификаторы с разделителем. **Идентификаторы с разделителем** заключаются в квадратные скобки ([ ]) или в двойные кавычки (" ") и может при необходимости содержать недопустимый символ или зарезервированные слова. Примеры идентификатора с разделителем: [Общая ведомость], [Номер группы], [Код специальности].

Именование. SQL Server достаточно лояльно относиться к именам таблиц.

Можно *использовать в качестве имен русские слова, и даже имена из нескольких слов. Только если имя состоит из нескольких слов, оно должно заключаться в квадратные скобки.*

1.2 Типы данных в Transact-SQL.

Тип данных определяет диапазон значений, которые можно сохранить в переменной или в поле таблицы.

Числовые типы:

Числовые типы данных

1) ***Целочисленные типы данных*** (общее название integer):

- **SMALLINT**: хранит числа от –32 768 до 32 767. Занимает 2 байта
- **INT**: хранит числа от –2 147 483 648 до 2 147 483 647. Занимает 4 байта. Наиболее используемый тип для хранения чисел.

- **BIT**: хранит значение от 0 до 16. Может выступать аналогом булевого типа в языках программирования (в этом случае значению true соответствует 1, а значению false - 0). При значениях до 8 (включительно) занимает 1 байт, при значениях от 9 до 16 - 2 байта.
- **TINYINT**: хранит числа от 0 до 255. Занимает 1 байт. Хорошо подходит для хранения небольших чисел.
- **BIGINT**: хранит очень большие числа от -9 223 372 036 854 775 808 до 9 223 372 036 854 775 807, которые занимают в памяти 8 байт.

2) Нецелочисленные типы данных

Подразделяются на два типа: ***десятичные*** (decimal) и ***приблизительные*** (approximate).

Десятичные данные хранятся в виде последовательности цифр.

- **DECIMAL**: хранит числа с фиксированной точностью. Занимает от 5 до 17 байт в зависимости от количества чисел после запятой.

Данный тип может принимать два параметра precision и scale: DECIMAL(precision, scale).

Параметр precision представляет максимальное количество цифр, которые может хранить число. Это значение должно находиться в диапазоне от 1 до 38. По умолчанию оно равно 18.

Параметр scale представляет максимальное количество цифр, которые может содержать число после запятой. Это значение должно находиться в диапазоне от 0 до значения параметра precision. По умолчанию оно равно 0.

- **NUMERIC**: данный тип аналогичен типу DECIMAL.

Приблизительные типы используются для работы с данными, имеющими значения от очень малых величин до предельно больших. К ним относятся:

- **FLOAT**: хранит числа от –1.79E+308 до 1.79E+308. Занимает от 4 до 8 байт в зависимости от дробной части.

Может иметь форму опредеения в виде FLOAT(n), где n представляет число бит, которые используются для хранения десятичной части числа (мантиссы). По умолчанию n = 53.

- **REAL**: хранит числа от –340E+38 to 3.40E+38. Занимает 4 байта. Эквивалентен типу FLOAT(24).

Денежные типы данных

Используются для хранения данных о денежных суммах. Позволяет хранить после запятой четыре знака. ***К денежным типам относятся***:

- **SMALLMONEY**: хранит дробные значения от -214 748.3648 до 214 748.3647. Предназначено для хранения денежных величин. Занимает 4 байта. Эквивалентен типу DECIMAL(10,4).
- **MONEY**: хранит дробные значения от -922 337 203 685 477.5808 до 922 337 203 685 477.5807. Представляет денежные величины и занимает 8 байт. Эквивалентен типу DECIMAL(19,4).

Тип даты и времени

Существует встроенные типы, позволяющие хранить сведения о дате и о времени одновременно, или отдельно. С данными этого типа можно выполнять арифметические операции сложения и вычитания, а также сравнивать их. К временным типам относятся:

- **DATE**: хранит даты от 0001-01-01 (1 января 0001 года) до 9999-12-31 (31 декабря 9999 года). Занимает 3 байта.
- **TIME**: хранит время в диапазоне от 00:00:00.0000000 до 23:59:59.9999999. Занимает от 3 до 5 байт.

Может иметь форму TIME(n), где n представляет количество цифр от 0 до 7 в дробной части секунд.

- **DATETIME**: хранит даты и время от 01/01/1753 до 31/12/9999. Занимает 8 байт.
- **DATETIME2**: хранит даты и время в диапазоне от 01/01/0001 00:00:00.0000000 до 31/12/9999 23:59:59.9999999. Занимает от 6 до 8 байт в зависимости от точности времени.

Может иметь форму DATETIME2(n), где n представляет количество цифр от 0 до 7 в дробной части секунд.

- **SMALLDATETIME**: хранит даты и время в диапазоне от 01/01/1900 до 06/06/2079, то есть ближайшие даты. Занимает от 4 байта.
- **DATETIMEOFFSET**: хранит даты и время в диапазоне от 0001-01-01 до 9999-12-31. Сохраняет детальную информацию о времени с точностью до 100 наносекунд. Занимает 10 байт.

Строковые типы данных

- **CHAR**: хранит строку длиной от 1 до 8 000 символов. На каждый символ выделяет по 1 байту. Не подходит для многих языков, так как хранит символы не в кодировке Unicode. Используется для хранения набора символов длинной n (n_{max} = 8000). Строка данного типа является строкой фиксированной длины. Если строка содержит символов меньше, чем n, то в конец строки добавляются пробелы. Если символов больше n, то часть конечных символов будет потеряна. Количество символов, которое может

хранить столбец, передается в скобках. Например, для столбца с типом CHAR(10) будет выделено 10 байт. И если мы сохраним в столбце строку менее 10 символов, то она будет дополнена пробелами.

- **VARCHAR**: хранит строку. На каждый символ выделяется 1 байт. Можно указать конкретную длину для столбца - от 1 до 8 000 символов, например, VARCHAR(10). Если строка должна иметь больше 8000 символов, то задается размер MAX, а на хранение строки может выделяться до 2 Гб: VARCHAR(MAX).

Не подходит для многих языков, так как хранит символы не в кодировке Unicode.

В отличие от типа CHAR если в столбец с типом VARCHAR(10) будет сохранена строка в 5 символов, то в столце будет сохранено именно пять символов.

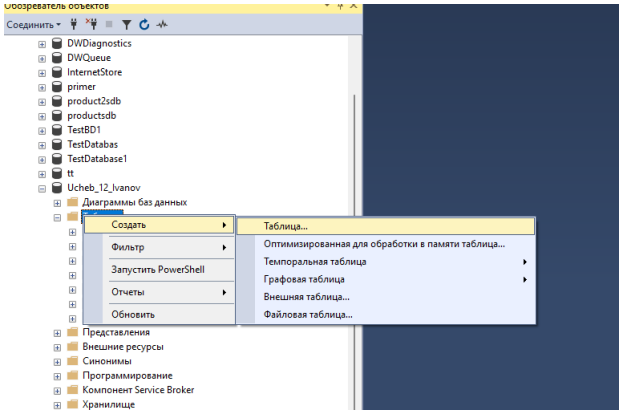
- **NCHAR**: хранит строку в кодировке Unicode длиной от 1 до 4 000 символов. На каждый символ выделяется 2 байта. Например, NCHAR(15)
- **NVARCHAR**: хранит строку в кодировке Unicode. На каждый символ выделяется 2 байта. Можно задать конкретный размер от 1 до 4 000 символов: . Если строка должна иметь больше 4000 символов, то задается размер MAX, а на хранение строки может выделяться до 2 Гб.

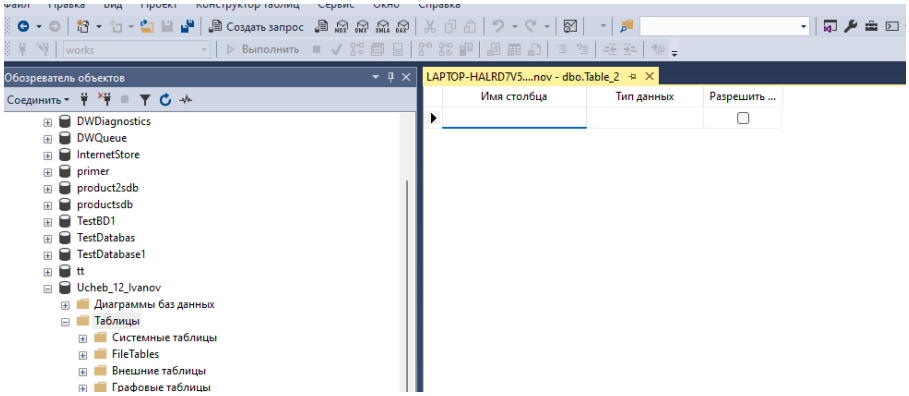
Примечание. Для использования русских символов (не ASCII кодировки) используются типы данных с приставкой "n" (nchar, nvarchar, ntext), которые кодируют символы двумя байтами. Иначе говоря, для работы с Unicode используются типы данных с "n".
Примечание. Для данных переменной длины используются типы данных с приставкой "var". Типы данных без приставки "var" имеют фиксированную длину области памяти, неиспользованная часть которой заполняется пробелами или нулями.

1.3 Использование конструктора таблиц в SQL Server Management Studio

В SSMS в обозревателе объектов ***подключитесь к экземпляру*** Компонент Database Engine , который содержит изменяемую базу данных. В обозревателе объектов ***разверните узел Базы данных,*** а затем базу данных, в которой будет размещена новая таблица.

В обозреватель объектов щелкните правой кнопкой мыши узел ***Таблицы*** базы данных и выберите ***Создать таблицу.***





Вводим имена столбцов, выберите типы данных и определите для каждого столбца, могут ли в нем присутствовать значения NULL.

Например,

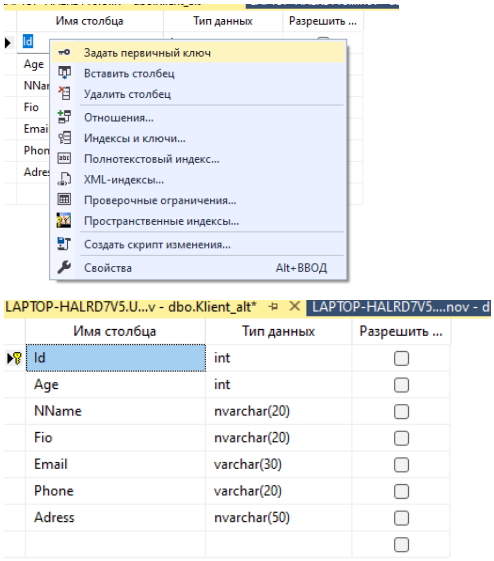
Имя столбца	Тип данных	Разрешить ...
Id	int	☐
Age	int	☐
NName	nvarchar(20)	☐
Fio	nvarchar(20)	☐
Email	varchar(30)	☐
Phone	varchar(20)	☐
Adress	nvarchar(50)	☐

Чтобы указать дополнительные свойства столбца, например идентификаторы или вычисляемые значения столбца, ***выберите столбец и на вкладке свойств столбца выберите соответствующие свойства. Примечание, описание свойств см. раздел 1.3***

Чтобы указать, что столбец **является столбцом первичного ключа**, щелкните его правой кнопкой мыши и выберите Задать первичный ключ.

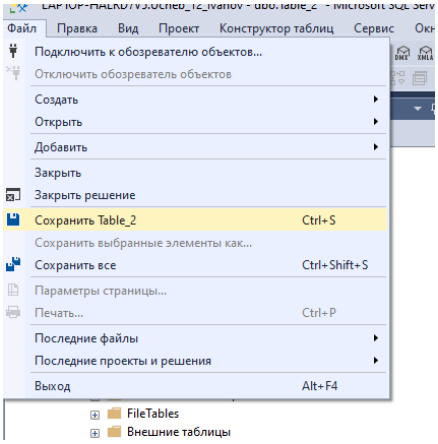
Первичный ключ **это одно или несколько полей в таблице. Он необходим для уникальной идентификации любой строки.** Первичный ключ накладывает некоторые ограничения. Все записи относящиеся к первичному ключу должны быть уникальны. Записи в полях относящихся к первичному ключу не могут быть пустыми. В каждой таблице может присутствовать только один первичный ключ.

«Первичный ключ — это значение или комбинация нескольких значений из таблицы, уникально определяющая каждую запись в этой таблице. стол. Если мы знаем это значение/комбинацию, мы можем легко найти связанную запись и получить доступ ко всем оставшимся значениям из этой записи».

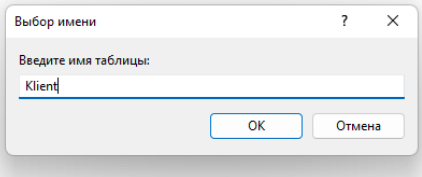


По умолчанию таблица содержится в схеме dbo. **Префикс dbo обозначает, что таблица является объектом БД (Data Base Object)**. В дальнейшем при работе с объектами БД префикс dbo можно опускать.

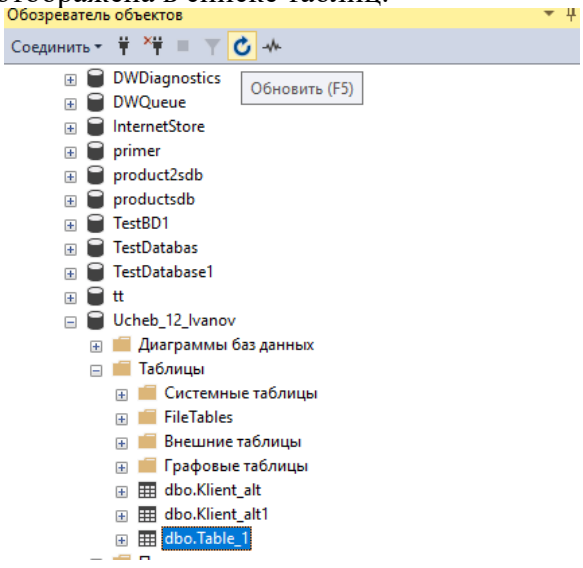
В меню Файл выберите команду **Сохранить имя_таблицы**.

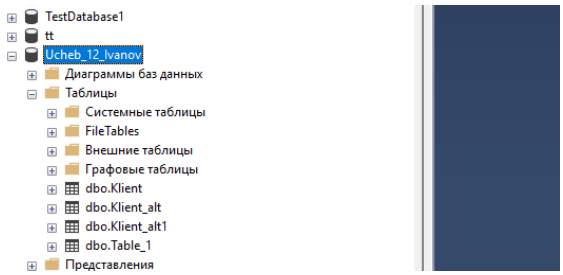


В диалоговом окне Выбор имени введите имя таблицы и нажмите кнопку ОК.



Чтобы просмотреть новую таблицу, в обозревателе объектов разверните узел Таблицы , а затем нажмите клавишу F5 , чтобы обновить список объектов. Новая таблица будет отображена в списке таблиц.





1.3 Свойства столбца таблицы (среда SQL Server Management Studio)

Эти свойства отображаются на нижней панели конструктора таблиц. Если не оговорено обратное, эти свойства можно изменять в окне «Свойства» при выборе столбца.

Общие сведения

Имя. Отображает имя выбранного столбца.

Разрешить значения NULL. Указывает, может ли столбец содержать значения NULL. Для изменения этого свойства щелкните флажок «Разрешить значения NULL», соответствующий столбцу на верхней панели конструктора таблиц.

Тип данных. Отображает тип данных выбранного столбца. Для редактирования свойства щелкните его значение, откройте раскрывающийся список и выберите другое значение.

Значение по умолчанию или привязка. Выводит значение по умолчанию для этого столбца, устанавливаемое, если никакого значения не было присвоено. Значение этого поля может быть либо значением ограничения SQL Server по умолчанию, либо именем глобального ограничения, к которому привязан столбец. Раскрывающийся список содержит все глобальные значения по умолчанию, определенные в базе данных. Для привязки столбца к глобальным значениям по умолчанию выберите значение из раскрывающегося списка. Или же, чтобы создать ограничение столбца по умолчанию, введите значение по умолчанию непосредственно в текстовом виде.

Длина. Указывает число символов, допустимых для символьных типов данных. Данное свойство доступно только для символьных типов данных/

Масштаб. Отображает максимальное количество цифр справа от десятичной запятой в значениях этого столбца. Для нечисловых типов данных в этом свойстве отображается 0.

Точность. Выводит максимальное число цифр в значениях этого столбца. Для нечисловых типов данных в этом свойстве отображается 0 .

Конструктор таблиц

Параметры сортировки. Отображает последовательность сортировки, которая SQL Server применяется по умолчанию к столбцу всякий раз, когда значения столбцов используются для сортировки строк результата запроса. Чтобы изменить параметры сортировки, выберите свойство, щелкните многоточие (…), которое отображается справа от значения свойства, чтобы открыть диалоговое окно Параметры сортировки .

Спецификация вычисляемого столбца. Выводит данные о вычисляемом столбце. Значение этого свойства совпадает со значением свойства потомка Формула и выводит эту формулу для данного вычисляемого столбца.

Примечание: Для изменения значения свойства Спецификация вычисляемого столбца необходимо развернуть его и изменить свойство потомка Формула.

Формула. Выводит формулу для данного вычисляемого столбца. Чтобы редактировать это свойство, введите в поле новую формулу.

Сохранен. Показывает, сохранены ли результаты вычислений по этой формуле. Если это свойство равно Нет , то сохраняется только формула, а значения вычисляются при каждом обращении к этому столбцу. Для редактирования свойства щелкните его значение, откройте раскрывающийся список и выберите другое значение.

Сжатый тип данных. Отображает сведения о типе данных поля в том же формате, что и в инструкции SQL CREATE TABLE. Например: поле, содержащее строку переменной длины с максимальной длиной 20 символов, будет представлено как «varchar(20)». Чтобы изменить это свойство, введите значение непосредственно в поле.

Описание. Выводит текст с описанием этого столбца. Чтобы изменить описание, выберите свойство , щелкните многоточие (…), которое отображается справа от значения свойства, и измените описание в диалоговом окне Свойство описания .

Детерминированное. Указывает, может ли тип данных для выбранного столбца быть определен точно.

Опубликован через службы DTS. Указывает, опубликован ли столбец через службы DTS.

Спецификация полного текста. Выводит данные о полнотекстовом индексе. Значение этого свойства совпадает со значением свойства потомка С полнотекстовым индексом и указывает, построен ли для данного столбца полнотекстовый индекс.

С полнотекстовым индексом. Указывает, построен ли для данного столбца полнотекстовый индекс. Этому свойству может быть присвоено значение Да только в том случае, если тип данных для данного столбца допустим для полнотекстового поиска и если таблица, к которой относится этот столбец, имеет соответствующий полнотекстовый индекс. Для редактирования свойства щелкните его значение, откройте раскрывающийся список и выберите другое значение.

Столбец полнотекстового типа. Выводит имя столбца, по которому построен полнотекстовый индекс для данного столбца. Это свойство можно задать, если значение свойства Тип данных для данного столбца равно либо image , либо varbinary. Указанный в этом свойстве столбец должен быть типа [n]char, [n]varchar или xml, и раскрывающийся список для этого свойства содержит только столбцы этих трех типов данных. Строки в столбце, указанном в этом свойстве, указывают тип документа соответствующих строк в столбце с возможностью полнотекстового поиска. Для редактирования свойства щелкните его значение, откройте раскрывающийся список и выберите другое значение.

Язык. Указывает язык средства разбиения по словам, использованного при индексировании этого столбца. Значение в этом свойстве — это на самом деле код локали для средства разбиения по словам. Дополнительные сведения о средствах разбиения по словам и кодах языка см. в разделе «Средства разбиения по словам и парадигматические модули». Для редактирования свойства щелкните его значение, откройте раскрывающийся список и выберите другое значение.

Статистическая семантика. Укажите, следует ли включить статистическое семантическое индексирование для выбранного столбца. Дополнительные сведения см. в разделе Семантический поиск (SQL Server).

Спецификация удостоверения

Выводит данные о том, будет ли данный столбец форсировать уникальность своих значений и, если будет, то как. Значение этого свойства показывает, является ли этот столбец столбцом идентификаторов и равным по значению свойству потомка Is Identity.

Спецификация идентификации необходимо развернуть его и изменить свойство потомка >**Is Identity** .СвойствоIs Identity показывает, является этот столбец столбцом раскрывающийся список и выберите другое значение.

СвойствоНачальное значение идентификатора выводит начальное значение, указанное при создании данного столбца идентификаторов. Это значение присваивается первой строке таблицы. Если оставить эту ячейку пустой, по умолчанию будет присвоено значение 1. Для изменения этого свойства введите новое значение непосредственно в поле.

СвойствоШаг приращения идентификатора выводит значение шага идентификатора, указанное при создании данного столбца идентификаторов. Это значение — шаг приращения, добавляемый к значению Начальное значение идентификатора для каждой следующей строки. Если оставить эту ячейку пустой, по умолчанию будет присвоено значение 1. Для изменения этого свойства введите новое значение непосредственно в поле.

Индексируемый.Указывает, может ли столбец быть проиндексирован. Например, недетерминированные вычисляемые столбцы не могут быть проиндексированы.

Опубликован слиянием. Указывает, опубликован ли столбец слиянием.

Не для репликации. Указывает, будут ли сохранены исходные значения идентификаторов во время репликации. Для редактирования свойства щелкните его значение, откройте раскрывающийся список и выберите другое значение.

Реплицировано. Указывает, реплицирован ли данный столбец в другое место.

RowGuid. Указывает, будет ли SQL Server использовать столбец в качестве глобального уникального идентификатора строк таблицы RowGuid. Задать значение Да можно только для уникального столбца идентификаторов. Для редактирования свойства щелкните его значение, откройте раскрывающийся список и выберите другое значение.

Размер. Указывает размер в байтах, который допускается типом данных столбца. Например, тип данных nchar может иметь длину 10 (количество символов), однако для работы с кодировкой Юникод его длина будет равняться 20.

1.4. Инструкция SELECT.

SELECT (Transact-SQL)

Возвращает строки из базы данных и позволяет делать выборку одной или нескольких строк или столбцов из одной или нескольких таблиц в SQL Server.

Полный синтаксис инструкции SELECT сложен, однако основные предложения можно кратко описать следующим образом:

[WITH { [XMLNAMESPACES ,][<common_table_expression>] }]
SELECT *выбранный_список* [INTO *новая_таблица*]
[FROM *источник_таблицы*] [WHERE *условие_поиска*]
[GROUP BY *выражение_группирования*]
[WINDOW *выражение_окна*]
[HAVING *условие_поиска*]
[ORDER BY *выражение_упорядочения* [ASC | DESC]]

Операторы UNION, EXCEPT и INTERSECT можно использовать между запросами, чтобы сравнить их результаты или объединить в один результирующий набор.

Порядок предложений в инструкции SELECT имеет значение. Любое из необязательных предложений может быть опущено; но если необязательные предложения используются, они должны следовать в определенном порядке.

Логический порядок обработки инструкции SELECT

Следующие действия демонстрируют логический порядок обработки или порядок привязки инструкции SELECT. Этот порядок определяет, когда объекты, определенные в одном шаге, становятся доступными для предложений в последующих шагах.

FROM	определяет имена используемых таблиц
JOIN	
GROUP BY	группирует строки, имеющие одинаковые значения в указанном столбце;
HAVING	фильтрует группы строк в соответствии с указанным условием
DISTINCT	
ON	
WHERE	фильтрует строки таблицы в соответствии с заданными условиями
WITH CUBE или WITH ROLLUP	
SELECT	форматирует выходные данные
ORDER BY	сортирует результаты выполнения запроса

Порядок предложений в запросе SELECT не может быть изменен.

Предложения SELECT и FROM являются обязательными, присутствие остальных зависит от контекста.

В предложении SELECT указывается список столбцов, которые должны быть возвращены запросом. Можно указать исходные элементы или вычисляемые поля во время выполнения запроса.

Конструкция DISTINCT | ALL исключает / разрешает вывод повторяющихся строк.

Конструкция **ALL используется по умолчанию**.

* означает вывод всех столбцов указанной таблицы.

В случае, если выборка производится из нескольких таблиц, перед символом звездочки может указываться имя таблицы.

SQL-запрос может содержать вычисляемые столбцы, значения которых могут определятся на основе значений данных, хранящихся в БД конструкции.

Вычисляемым столбцам следует давать название с помощью ключевого слова AS.

Вычисляемый столбец можно создать как: <Новое поле> = <выражение>

Если название столбца состоит из нескольких слов, разделенных пробелами, следует их записать в квадратных скобках: [].

Сортировка данных выполняется с помощью команды ORDER BY, которая добавляется в конец запроса, после чего перечисляется список столбцов. Для каждого столбца указывается тип сортировки ASC | DESC (ascending – **по возрастанию** | descending – по убыванию). **ASC – по умолчанию, можно не указывать**.

Конструкция **TOP <N>** позволяет выбрать определенное количество строк из таблицы. Дополнительный оператор **PERCENT** позволяет выбрать процентное количество строк из таблицы. Дополнительный оператор **WITH TIES** позволяет выбрать все строки с такими же свойствами.

Конструкция **OFFSET <N> ROWS** указывает число строк, которые необходимо пропустить, прежде чем будет начат возврат строк из выражения запроса.

Конструкция **FETCH NEXT <N> ROWS ONLY** указывает число строк, возвращаемых после обработки предложения OFFSET.

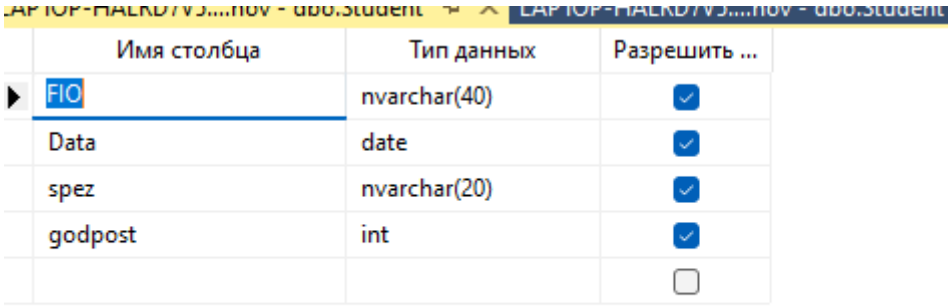
2. Практическая часть.

1. **Выполнить задания 1, 2.1-2.6, 3.1-3.12, отчет содержащий скрины прикрепить на уч. портал Лаб_1.1_ауд**
2. **Выполнить самостоятельную работу заданий пункта задания 4 отчет содержащий скрины прикрепить на уч. портал Лаб_1.1_сам**

Задание.

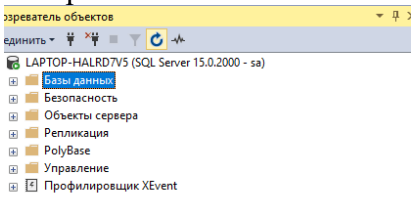
1. ***Создать средствами конструктора таблиц в SQL Server Management Studio следующую таблицу*** **Student:**

ФИО	Дата_рождения	Специальность	Год поступления
Александров Николай Николаевич	01.01.2000	Физика	2018
Бельский Василий Владимирович	02.05.2001	Математика	2019
Белопольский Андрей Иванович	21.10.2004	Информатика	2022
Петров Иван Петрович	17.07.2004	Физика	2022
Михайлов Николай Иванович	09.12.2005	математика	2023
Мироненко Светлана Владимировна	09.12.1998	Радиофизика	2016
Коваленко Владимир Иванович	09.12.2001	Информатика	2019
Виноградов Павел Гаврилович	09.12.2003	Химия	2021
Иванов Владимир Николаевич	09.12.1999	Радиофизика	2018
Семашкевич Виктория Михайловна	09.12.1998	Химия	2017

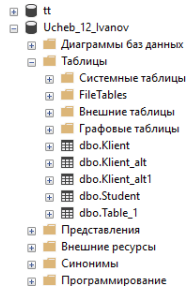


2. Заполните таблицу данными Student.

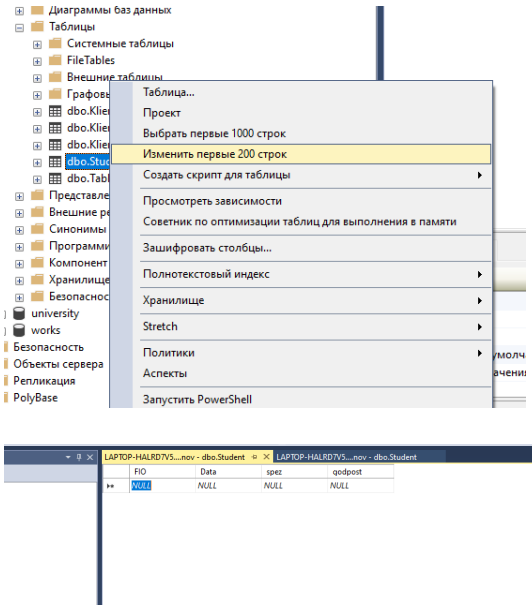
2.1 Для того, чтобы увидеть созданную таблицу в SSMS, необходимо, **обновить** обозреватель объектов.



В обозревателе объектов, открыть нужную базу, далее выбрать -**таблицы**.



2.2 Выделите, таблицу **Student**, в конт. Меню выберите **изменить первые 200 строк**

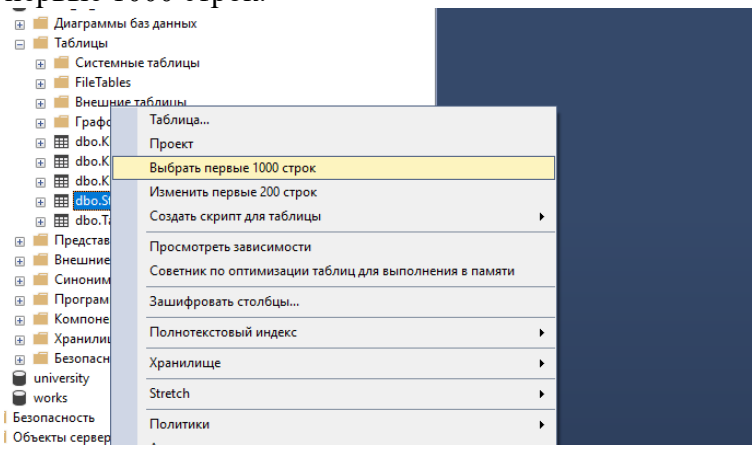


2.3. Введите информацию, согласно табл. **Student**, в результате получим:

FIO	Data	spez	qodpost
Александров Николай Николаевич	2001-01-01	Физика	2018
Бкльский Василий Владимирович	2001-05-02	математика	2019
Белопольский Андрей Иванович	2004-10-21	Информатика	2022
Петров Иван Петрович	2004-07-17	Физика	2022
Михайлов Николай Иванович	2005-12-09	математика	2023
Мироненко Светлана Владимировна	1998-12-09	Радиофизика	2016
Коваленко Владимир Иванович	2001-12-09	Информатика	2019
Виноградов Павел Гаврилович	2003-12-09	Химия	2021
Иванов Владимир Николаевич	1999-12-09	Радиофизика	2018
Семашкевич Виктория Михайловна	1998-12-09	Химия	2017
NULL	NULL	NULL	NULL

2.4 Сохраните введенные данные, закройте таблицу.

Чтобы посмотреть данные в таблице, выделите таблицу, и в конт. Меню выберите выбрать первые 1000 строк.



В результате получим, скрипт, и записи, которые содержаться в таблице **Student**

```
/****** Скрипт для команды SelectTopNRows из среды SSMS  ******/
SELECT TOP (1000) [FIO]
      ,[Data]
      ,[spez]
      ,[godpost]
  FROM [Ucheb_12_Ivanov].[dbo].[Student]
```

2 %

Результаты | Сообщения

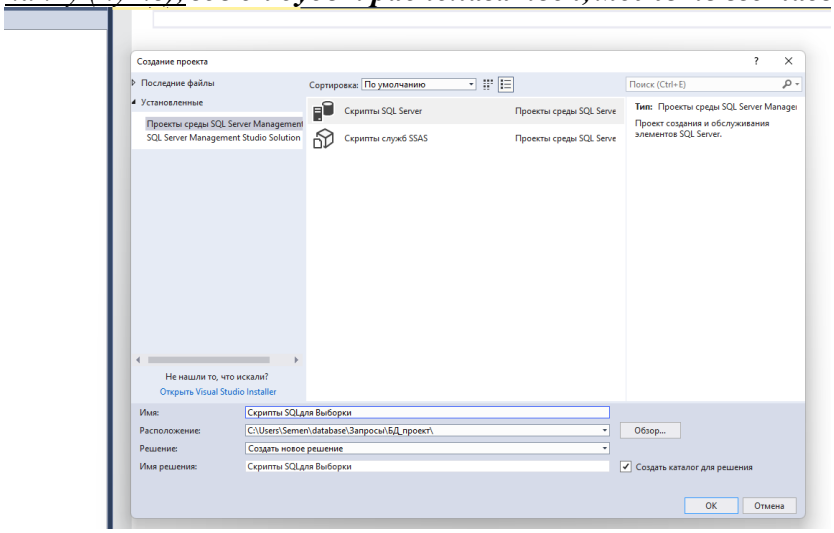
	FIO	Data	spez	godpost
1	Александров Николай Николаевич	2001-01-01	Физика	2018
2	Бкльский Василий Владимирович	2001-05-02	математика	2019
3	Белопольский Андрей Иванович	2004-10-21	Информатика	2022
4	Петров Иван Петрович	2004-07-17	Физика	2022
5	Михайлов Николай Иванович	2005-12-09	математика	2023
6	Мироненко Светлана Владимировна	1998-12-09	Радиофизика	2016
7	Коваленко Владимир Иванович	2001-12-09	Информатика	2019
8	Виноградов Павел Гаврилович	2003-12-09	Химия	2021
9	Иванов Владимир Николаевич	1999-12-09	Радиофизика	2018
10	Семашкевич Виктория Михайловна	1998-12-09	Химия	2017

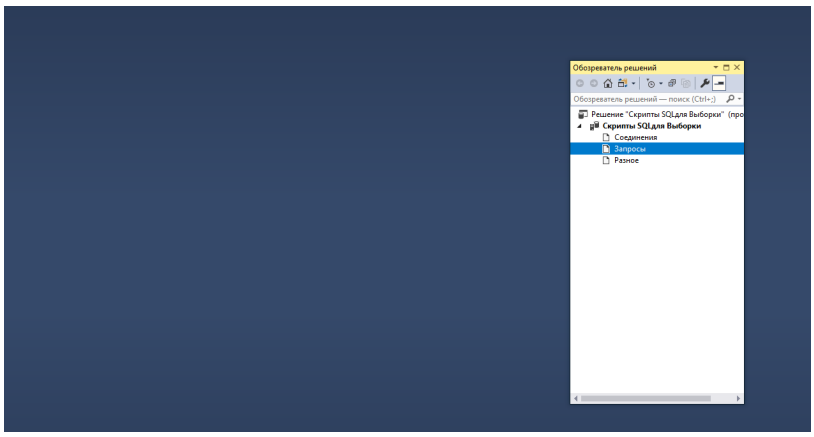
2.5 <u>Добавьте еще 7 записей</u> в таблицу, можно данные студентов своей группы.
2.6. Сохраните новые введенные данные, сохраните, **<u>и отобразите</u>** всю информацию таблицы **Student.**

3. <u>Команда Select</u>

Откройте свой проект, созданный в 1 лаб. Работе. **Файл -Открыть Решение или Проект.**

*Если не создавали проект, создайте**, см. лаб. Раб.1, Файл-Создать Проект,** <u>укажите папку (путь),</u> **<u>где он будет располагаться, можете его назвать «Скрипты для Выборки»</u>***

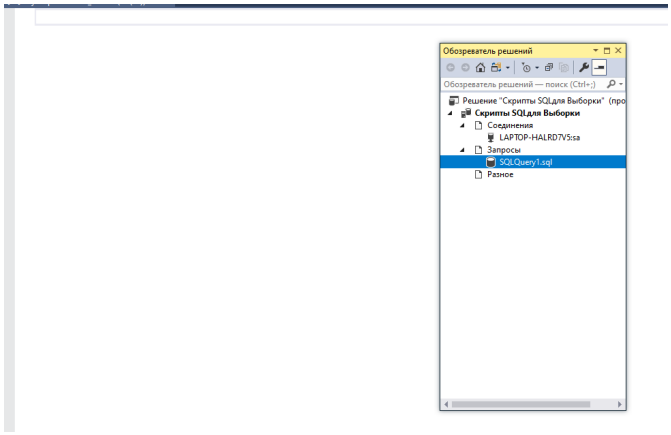




<u>Выполните следующие задания:</u>

<u>3.2 Вывести список всех СТУДЕНТОВ:</u>

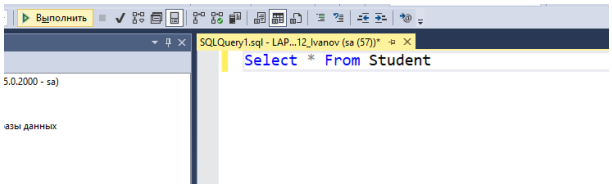
Выберите в меню **Проект-Создать запрос**



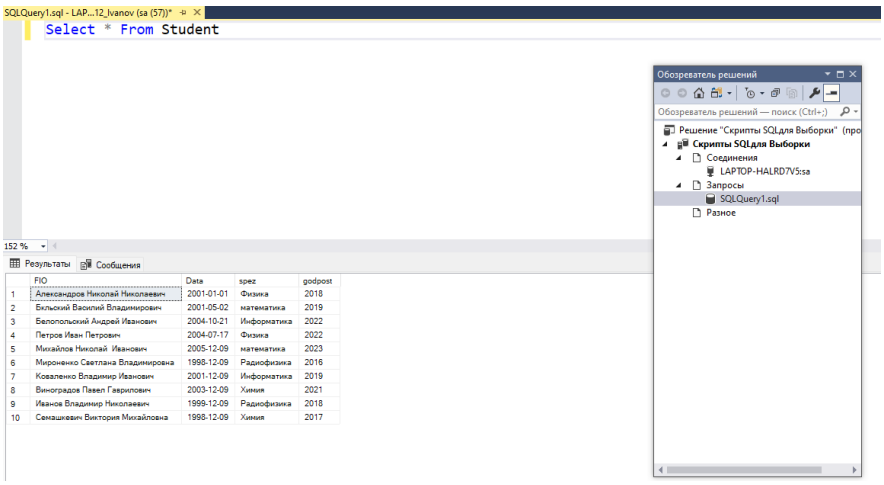
<u>Наберем следующий запрос</u>

```
Select * From Student
```

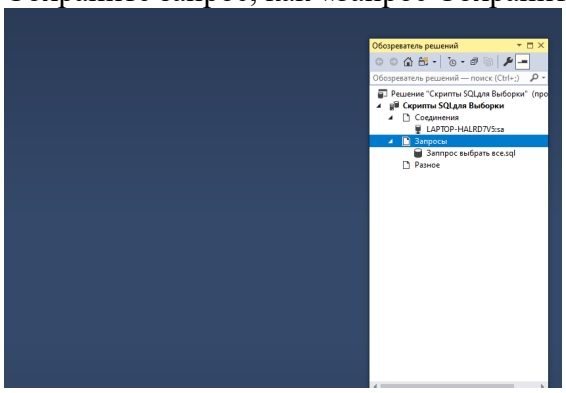
Выполним его



Получим:



Сохраните запрос, как «Запрос Сохранить все», он отобразиться в обозревателе решений



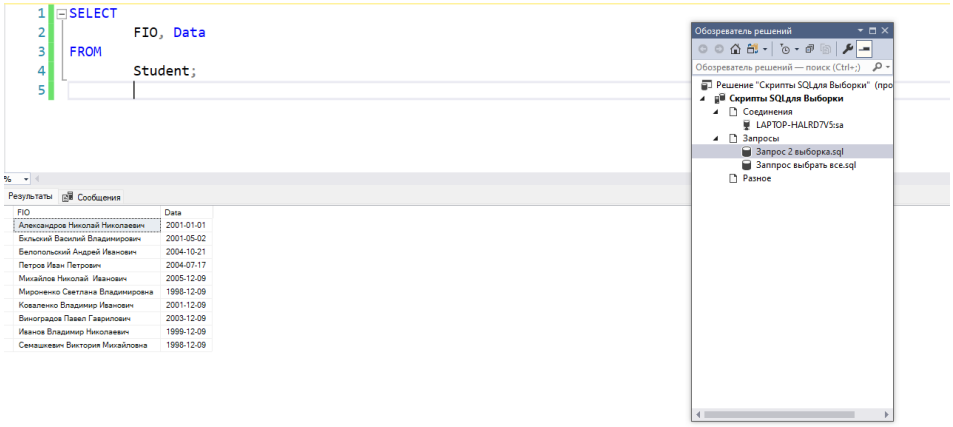
Примечание: Все следующие запросы будем сохранять в Проекте аналогичным образом.

3.3 Вывести ФИО и дату рождения всех студентов:

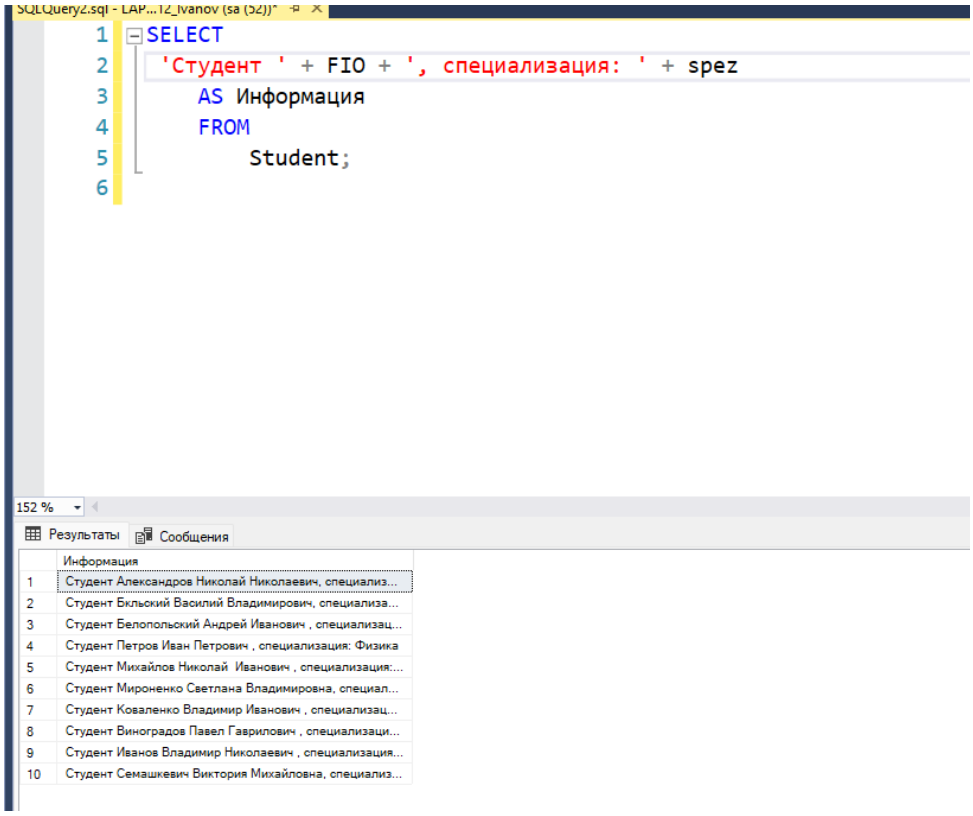
Создадим следующий запрос:

```
1  SELECT
2          FIO, Data
3  FROM
4          Student;
5
```

Выполним и сохраним запрос под именем «Запрос 2 выборка» получим:

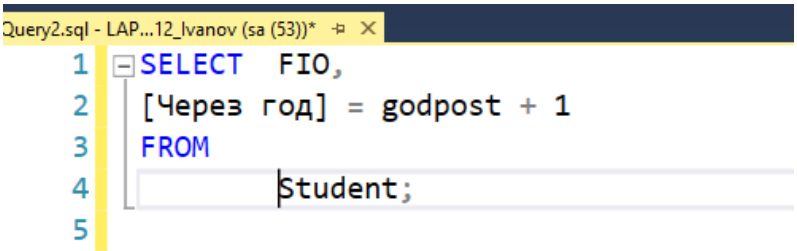


3.4: **Создайте вычисляемое поле «Информация», содержащее информацию об студентах в таком виде**: *«Студент Петров Петр Петрович, специализация: математика»:*

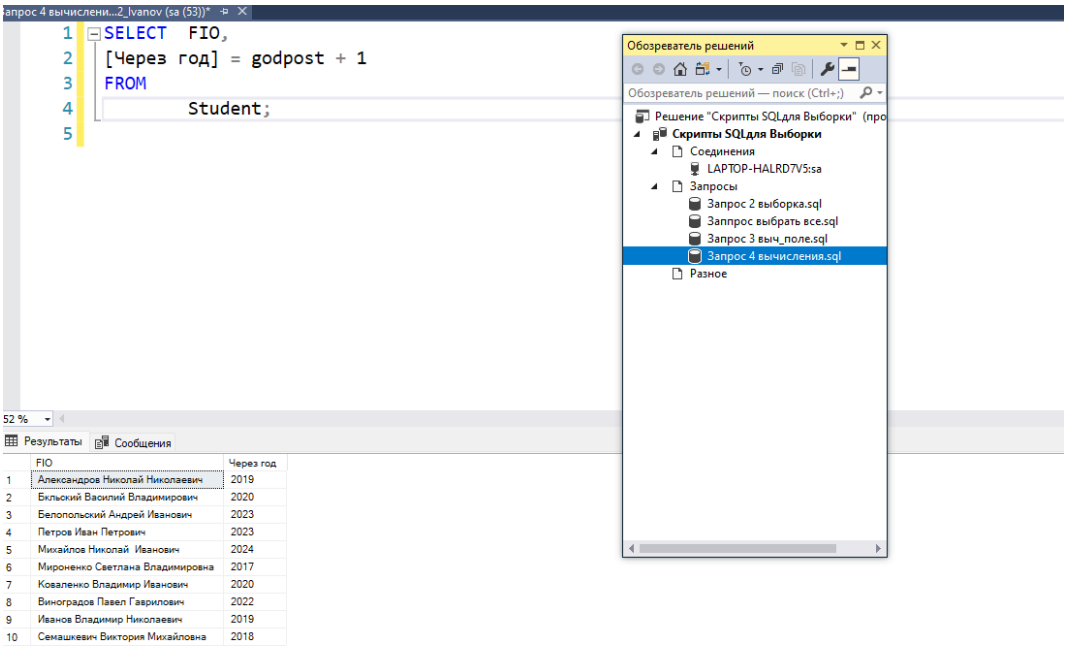


Сохраните Запрос **«Запрос 3 выч_поле»**

3.5 **Вывести FIO студентов и номер следующего года после поступления:**

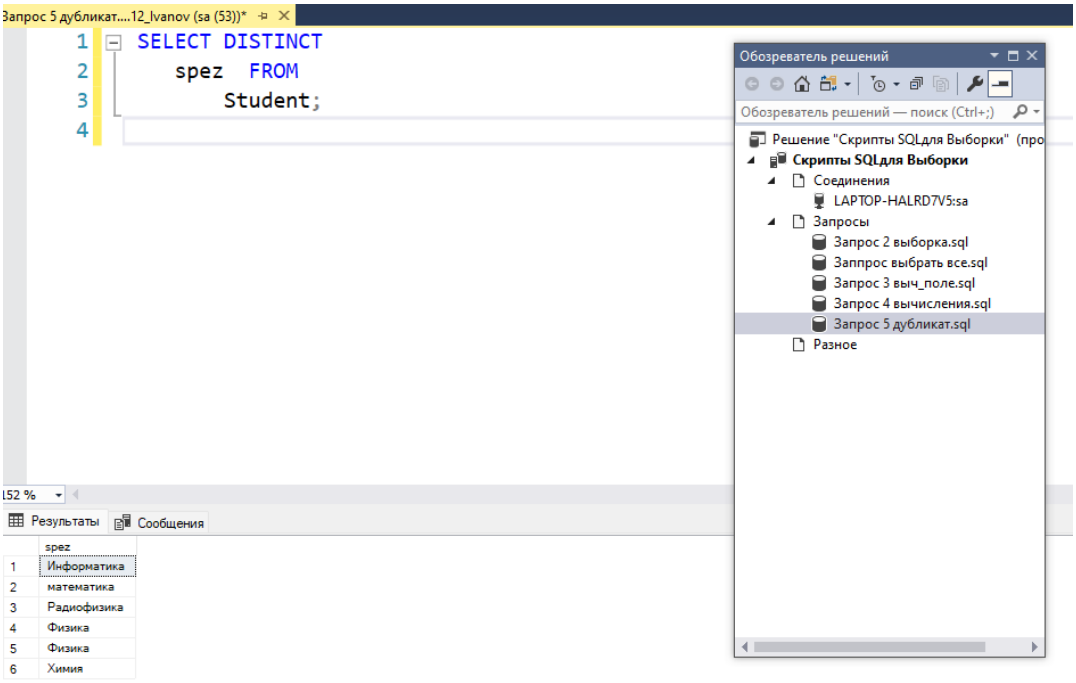


Выполнить и сохранить запрос под именем «Запрос 4 вычисления»



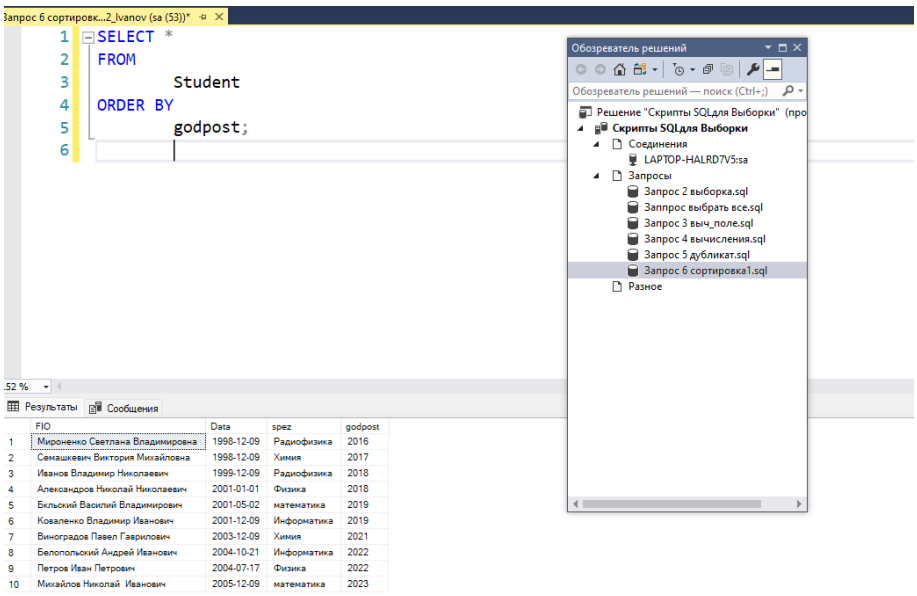
	FIO	Через год
1	Александров Николай Николаевич	2019
2	Бкльский Василий Владимирович	2020
3	Белопольский Андрей Иванович	2023
4	Петров Иван Петрович	2023
5	Михайлов Николай Иванович	2024
6	Мироненко Светлана Владимировна	2017
7	Коваленко Владимир Иванович	2020
8	Виноградов Павел Гаврилович	2022
9	Иванов Владимир Николаевич	2019
10	Семашкевич Виктория Михайловна	2018

3.6 **<u>Выведите список специализаций (spez), убрав дубликаты,</u>** сохраните запрос под именем «Запрос 5 дубликат»



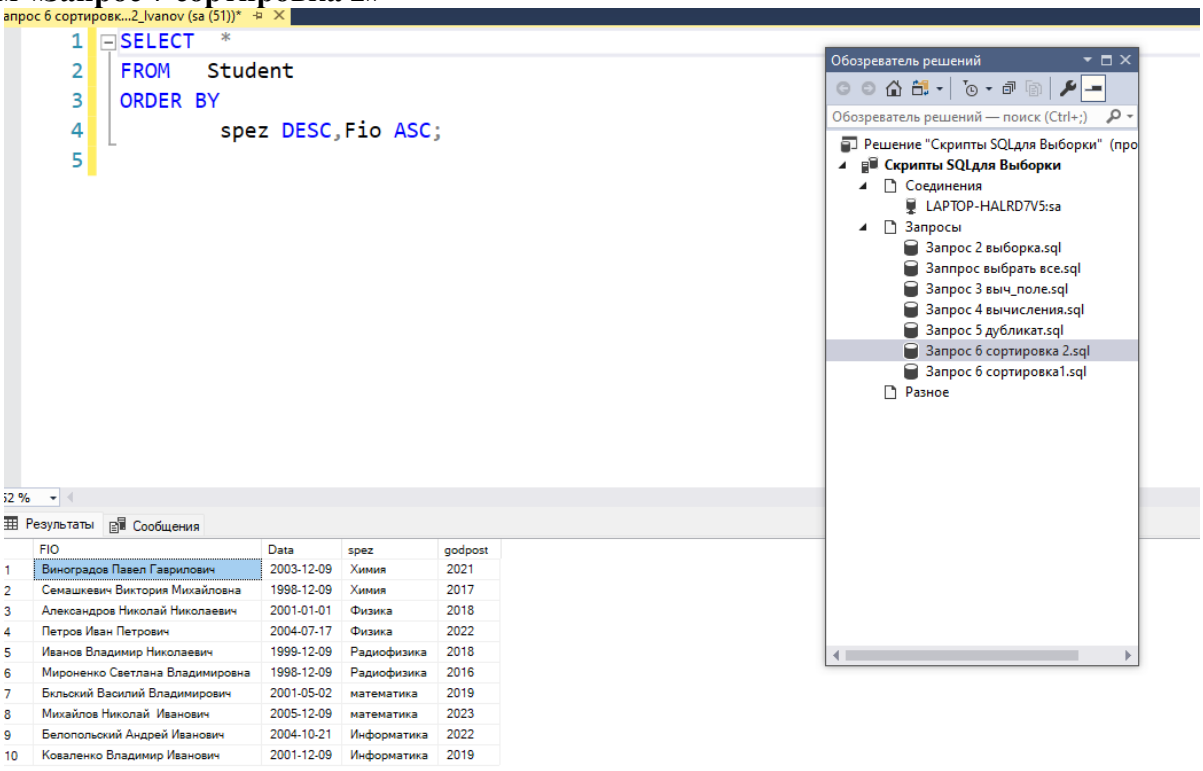
	spez
1	Информатика
2	математика
3	Радиофизика
4	Физика
5	Физика
6	Химия

3.7 **<u>Вывести список студентов, отсортированный по возрастанию года поступления,</u>** <u>сохранить «Запрос 6 сортировка»</u>



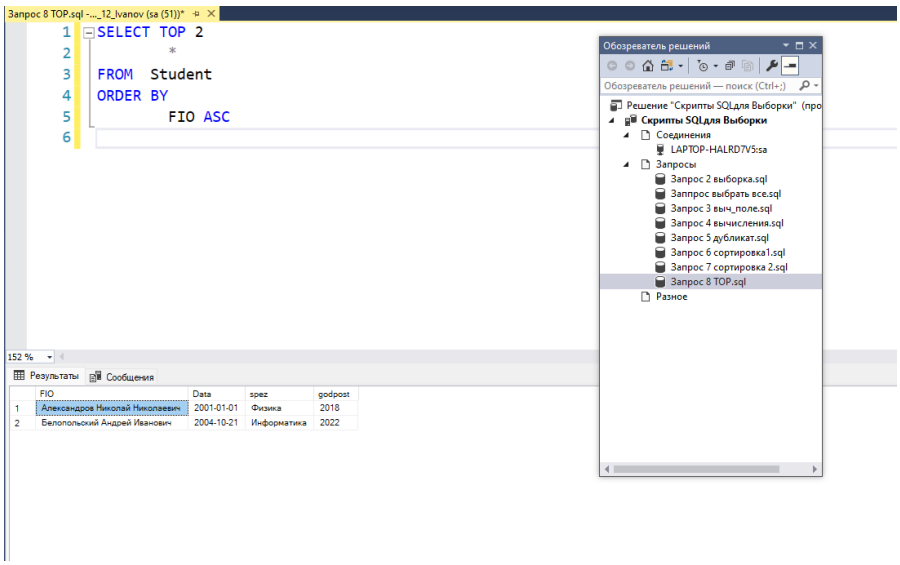
	FIO	Data	spez	godpost
1	Мироненко Светлана Владимировна	1998-12-09	Радиофизика	2016
2	Семашкевич Виктория Михайловна	1998-12-09	Химия	2017
3	Иванов Владимир Николаевич	1999-12-09	Радиофизика	2018
4	Александров Николай Николаевич	2001-01-01	Физика	2018
5	Бкльский Василий Владимирович	2001-05-02	математика	2019
6	Коваленко Владимир Иванович	2001-12-09	Информатика	2019
7	Виноградов Павел Гаврилович	2003-12-09	Химия	2021
8	Белопольский Андрей Иванович	2004-10-21	Информатика	2022
9	Петров Иван Петрович	2004-07-17	Физика	2022
10	Михайлов Николай Иванович	2005-12-09	математика	2023

3.8 **Вывести список студентов, отсортированный в обратном алфавитном порядке по полю «Spez» и в алфавитном порядке по полю «FIO», сохранить под именем «Запрос 7 сортировка 2»**

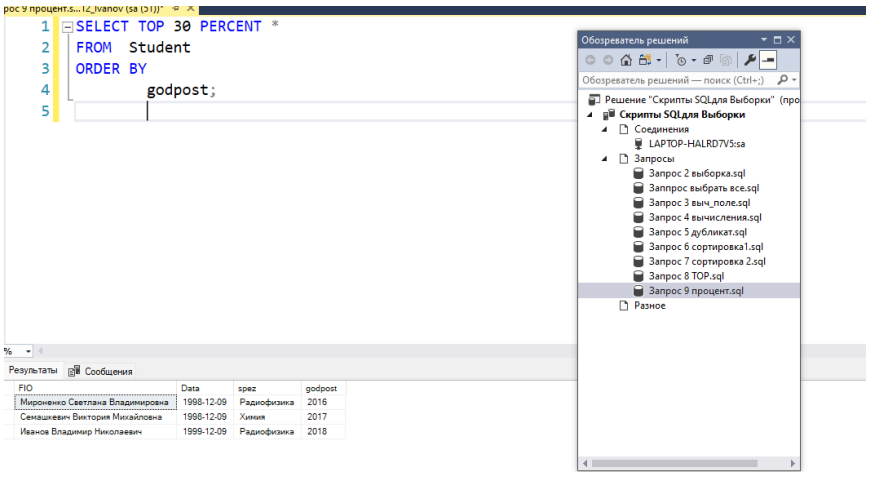


	FIO	Data	spez	godpost
1	Виноградов Павел Гаврилович	2003-12-09	Химия	2021
2	Семашкевич Виктория Михайловна	1998-12-09	Химия	2017
3	Александров Николай Николаевич	2001-01-01	Физика	2018
4	Петров Иван Петрович	2004-07-17	Физика	2022
5	Иванов Владимир Николаевич	1999-12-09	Радиофизика	2018
6	Мироненко Светлана Владимировна	1998-12-09	Радиофизика	2016
7	Бкльский Василий Владимирович	2001-05-02	математика	2019
8	Михайлов Николай Иванович	2005-12-09	математика	2023
9	Белопольский Андрей Иванович	2004-10-21	Информатика	2022
10	Коваленко Владимир Иванович	2001-12-09	Информатика	2019

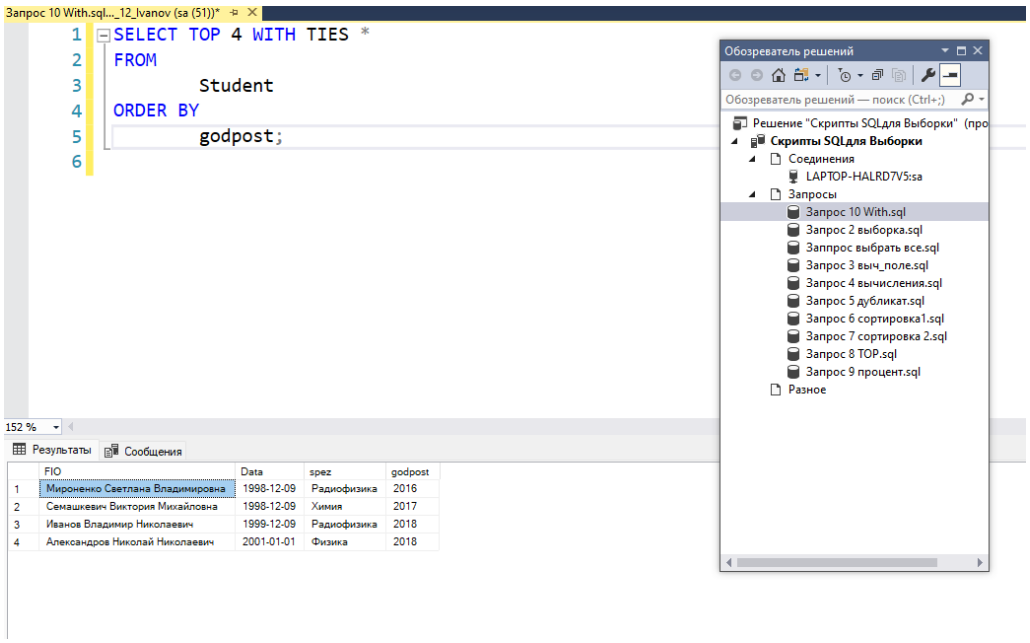
3.9 **Вывести <u>первые две строки</u> из списка студентов, отсортированного в алфавитном порядке по полю «ФИО»**, сохранить под именем «Запрос 8 TOP»



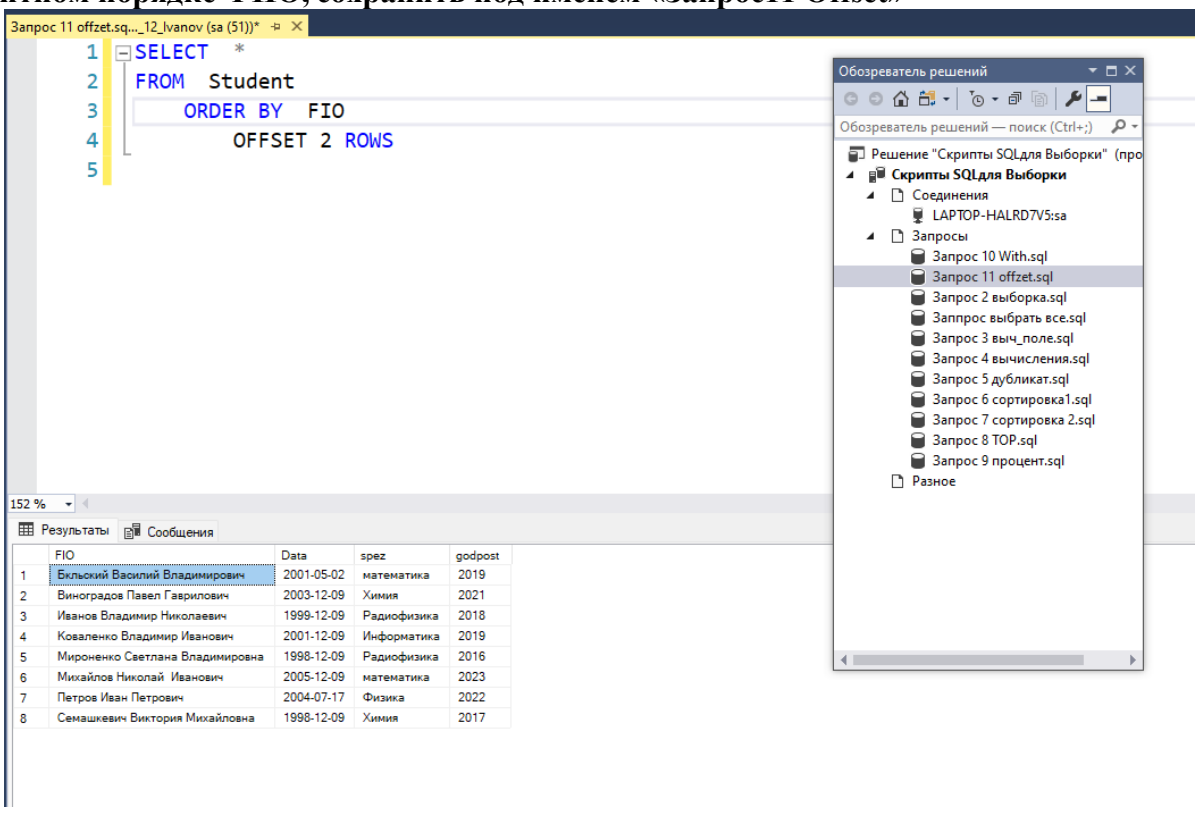
3.10: **<u>Вывести первые 30% строк из списка студентов, отсортированного по возрастанию года поступления,</u> сохранить под именем «Запрос 9 процент»**



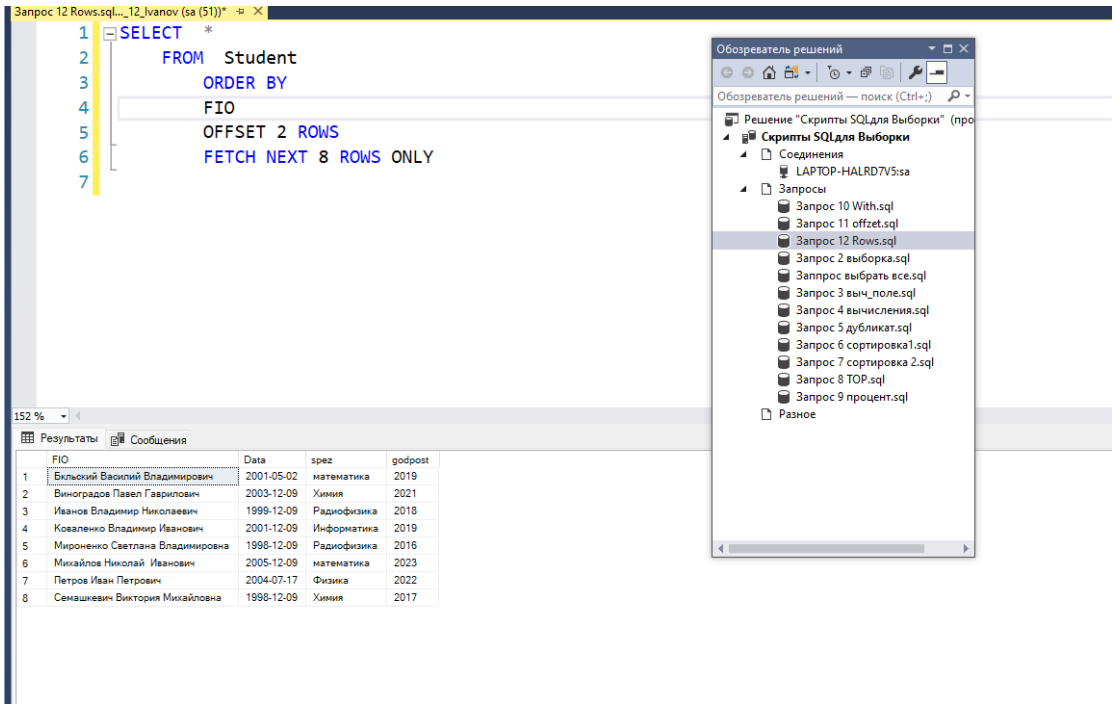
3.11 **Вывести из таблицы «Student, отсортированной по возрастанию года года поступления, список студентов, у которых год поступления – один из первых четырех в отсортированной таблице, сохранить под именем «Запрос10 With»**



3.12: **Вывести, начиная с третьего, список студентов, отсортированный в алфавитном порядке ФИО, сохранить под именем «Запрос11 Offset»**



3.13 **Вывести, начиная с третьего и до десятого, список студентов, отсортированный в алфавитном порядке ФИО, сохранить под именем «Запрос12 Rows»**



4. Самостоятельная работа.

1) Вывести ФИО, специализацию и дату рождения всех студентов.
2) Создать вычисляемое поле «О поступлении», которое содержит информацию об студентах в виде: «Петров Петр Петрович поступил в 2018».
3) Вывести ФИО студентов и вычисляемое поле «Через 5 лет после поступления».
4) Вывести список годов поступления, убрав дубликаты.
5) Вывести список студентов, отсортированный по убыванию даты рождения.
6) Вывести список студентов, отсортированный в обратном алфавитном порядке специализаций, по убыванию года поступления, и в алфавитном порядке ФИО.
7) Вывести первую строку из списка студентов, отсортированного в обратном алфавитном порядке ФИО.
8) Вывести фамилию студента, который раньше всех поступил.
9) Вывести первые 10% строк из списка студентов, отсортированного в алфавитном порядке ФИО.
10) Вывести из таблицы «Студенты», отсортированной по возрастанию года поступления, список студентов, у которых год поступления – один из первых пяти в отсортированной таблице.
11) Вывести, начиная с пятого, список студентов, отсортированный по возрастанию даты рождения.
12) Вывести 7 строку из списка студентов, отсортированного в алфавитном порядке ФИО.
13) Вывести с 5 по 9 строки из списка студентов, отсортированного в алфавитном порядке ФИО.