

## Домашнее задание №2

### Организация взаимодействия с базой данных

**Цель:** получение практических навыков физического проектирования базы данных в СУБД SQL Server.

**Задачи:** реализовать базу данных в СУБД SQL Server посредством SQL.

Таблица базы данных создается следующим образом:

```
CREATE TABLE Имя таблицы
(
    Имя столбца Тип данных [Свойства] [Декларативные ограничения],
    ...
    Имя столбца Тип данных [Свойства] [Декларативные ограничения]
)
```

Имя столбца и тип его данных определяются на этапе концептуального моделирования. Имя столбца может быть составным. В этом случае оно должно записываться внутри квадратных скобок.

Свойствами столбца являются:

- **NULL** – значения атрибута могут отсутствовать.
- **NOT NULL** – значения атрибута не могут отсутствовать.
- **IDENTITY**[(начальное значение, приращение) – значениями атрибута является последовательность чисел от начального значения с указанным приращением.

Декларативными ограничениями столбца могут быть:

- **PRIMARY KEY** – первичный ключ таблицы.
- **REFERENCES** [Владелец].[Имя таблицы (Имя столбца)] – ссылка на столбец другой таблицы, внешний ключ (**FOREIGN KEY**, см. ниже).
- **UNIQUE** – значения столбца могут быть уникальными.
- **DEFAULT** *выражение* – значение столбца по умолчанию.

- **CHECK** (логическое выражение) – значениями столбца могут быть только удовлетворяющие логическому выражению.

Помимо декларативных ограничений столбцов возможны и декларативные ограничения всей таблицы. Например, в случае составного первичного ключа, т.е. ключа, состоящего из нескольких столбцов. Декларативные ограничения таблицы описываются внутри оператора **CREATE TABLE** либо до, либо после описания ее столбцов следующим образом:

**CONSTRAINT** *Имя ограничения Декларативное ограничение таблицы*

Декларативным ограничением таблицы может быть любое из декларативных ограничений столбца. Аналогичным образом с использованием оператора **CONSTRAINT** может быть описано из любой из декларативных ограничений столбцов.

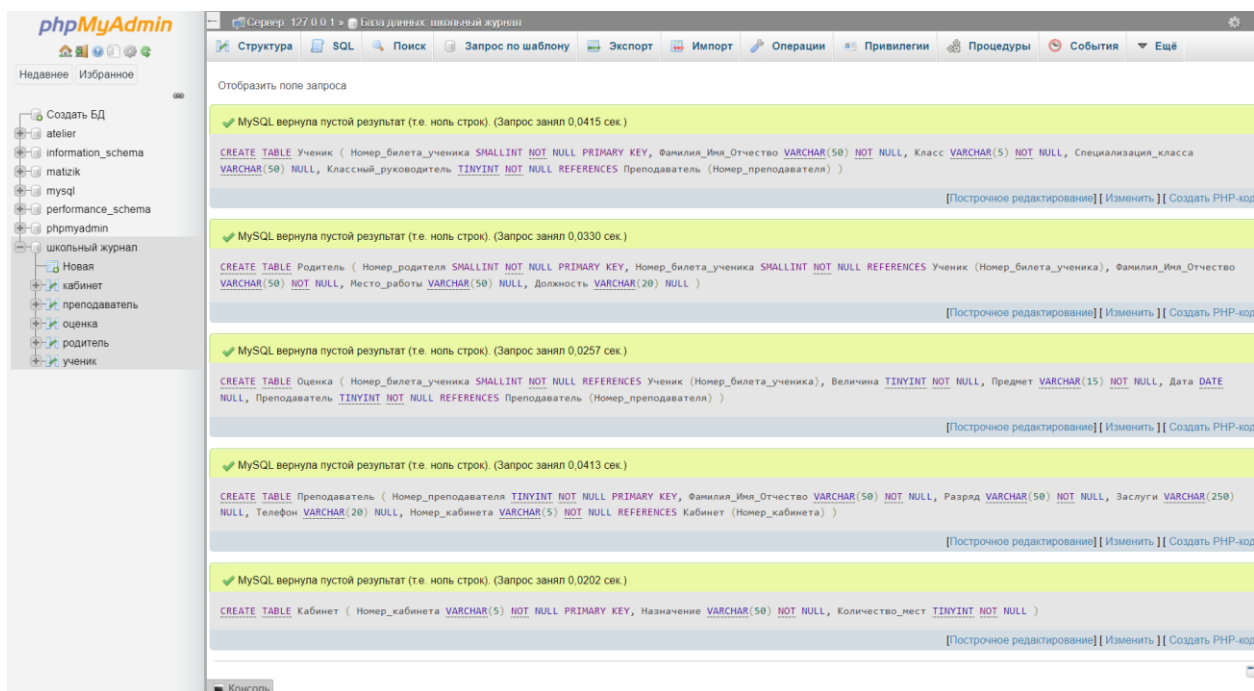
Столбец и таблица могут содержать несколько непротиворечивых ограничений декларативной целостности. Ниже создаются таблицы для предметной области «Школьный журнал».

```

Выполнить SQL-запрос(ы) к базе данных школьный журнал:
1 CREATE TABLE Ученик
2 (
3     Номер_билета_ученика SMALLINT NOT NULL PRIMARY KEY,
4     Фамилия_Имя_Отчество VARCHAR(50) NOT NULL,
5     Класс VARCHAR(5) NOT NULL,
6     Специализация_класса VARCHAR(50) NULL,
7     Классный_руководитель TINYINT NOT NULL REFERENCES Преподаватель (Номер_преподавателя)
8 );
9
10 CREATE TABLE Родитель
11 (
12     Номер_родителя SMALLINT NOT NULL PRIMARY KEY,
13     Номер_билета_ученика SMALLINT NOT NULL REFERENCES Ученик (Номер_билета_ученика),
14     Фамилия_Имя_Отчество VARCHAR(50) NOT NULL,
15     Место_работы VARCHAR(50) NULL,
16     Должность VARCHAR(20) NULL
17 );
18
19 CREATE TABLE Оценка
20 (
21     Номер_билета_ученика SMALLINT NOT NULL REFERENCES Ученик (Номер_билета_ученика),
22     Величина TINYINT NOT NULL,
23     Предмет VARCHAR(15) NOT NULL,
24     Дата DATE NULL,
25     Преподаватель TINYINT NOT NULL REFERENCES Преподаватель (Номер_преподавателя)
26 );
27
28 CREATE TABLE Преподаватель
29 (
30     Номер_преподавателя TINYINT NOT NULL PRIMARY KEY,
31     Фамилия_Имя_Отчество VARCHAR(50) NOT NULL,
32     Разряд VARCHAR(50) NOT NULL,
33     Заслуги VARCHAR(250) NULL,
34     Телефон VARCHAR(20) NULL,
35     Номер_кабинета VARCHAR(5) NOT NULL REFERENCES Кабинет (Номер_кабинета)
36 );
37
38 CREATE TABLE Кабинет
39 (
40     Номер_кабинета VARCHAR(5) NOT NULL PRIMARY KEY,
41     Назначение VARCHAR(50) NOT NULL,
42     Количество_мест TINYINT NOT NULL
43 )

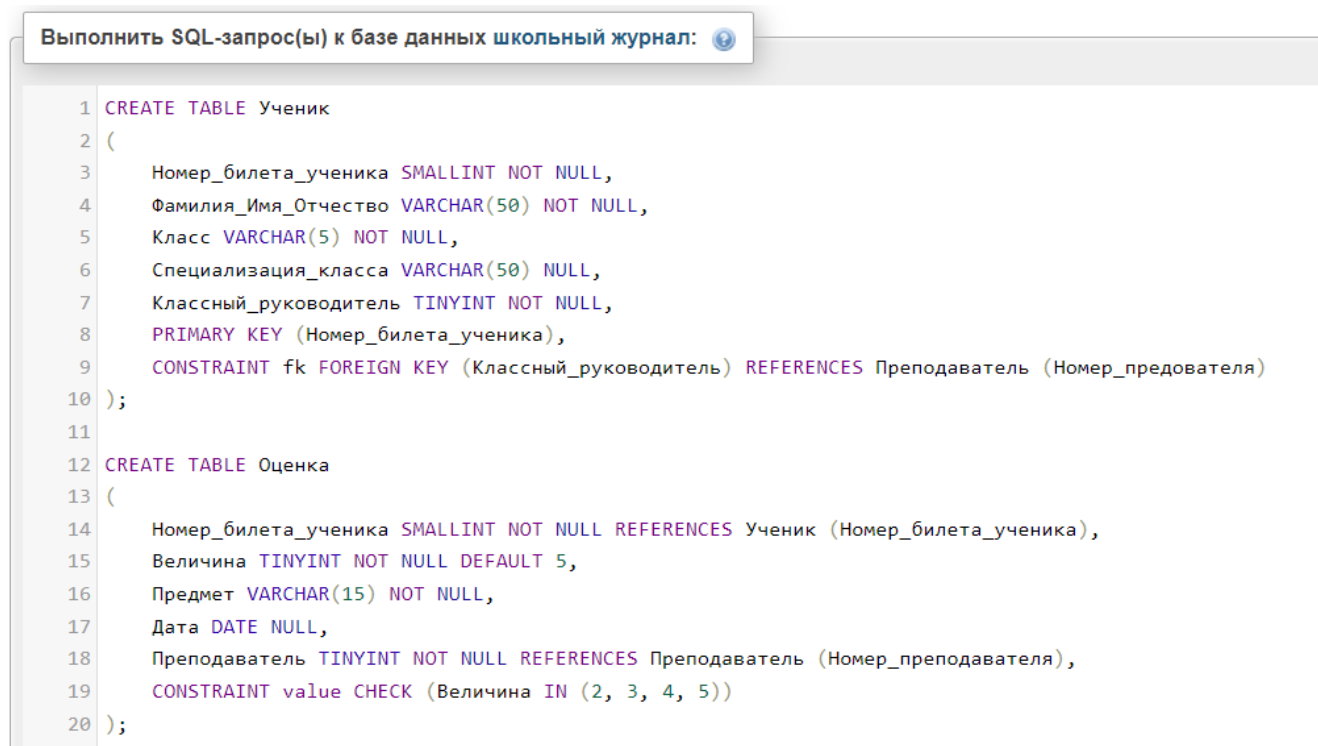
```

**Рисунок 1** - Создания таблиц предметной области «Школьный журнал»



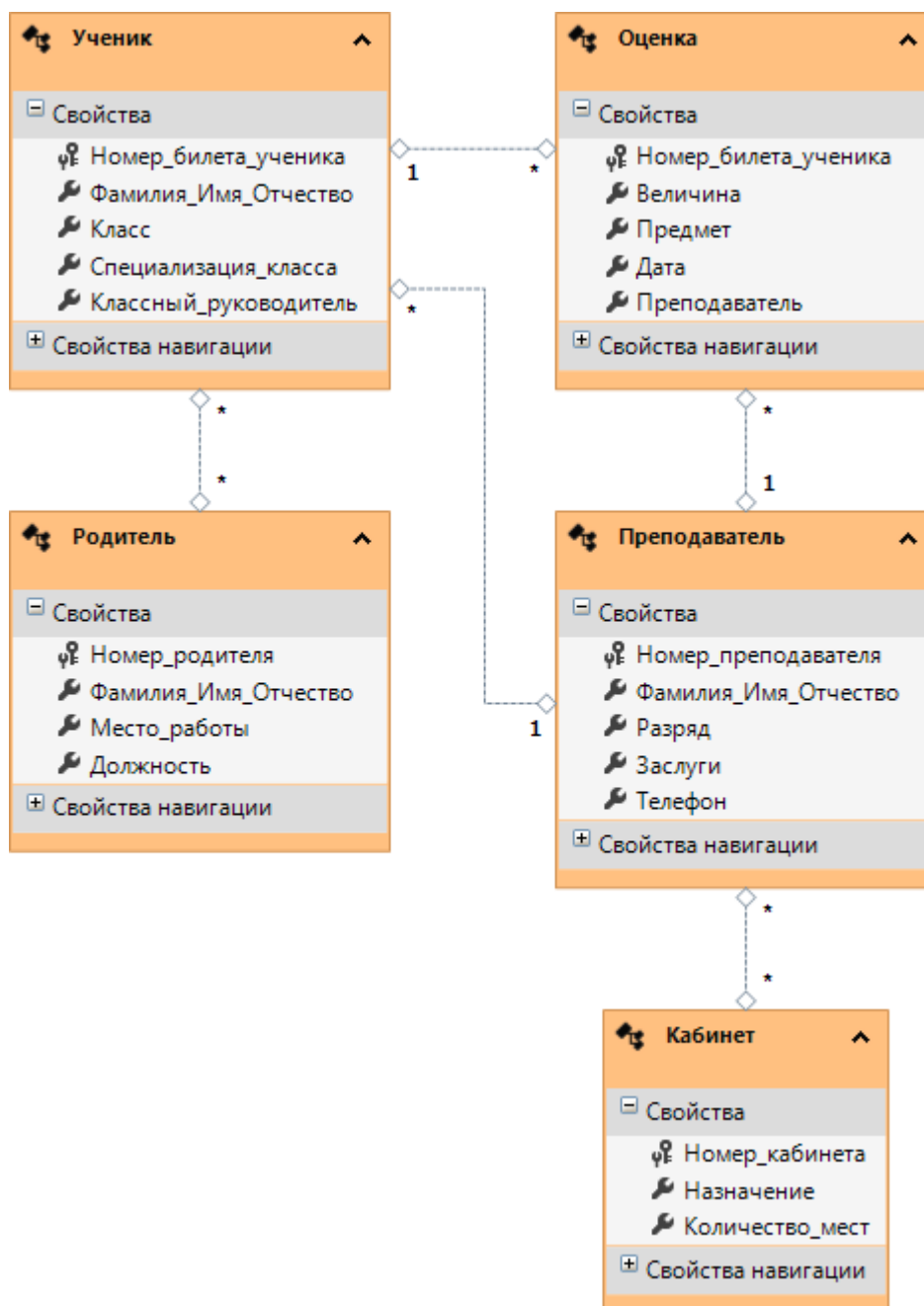
**Рисунок 2 - Результат выполнения кода**

Пример создания таблиц для отношений «Ученик» и «Оценка» с использованием декларативных ограничений таблицы:



**Рисунок 3 - Создание таблиц «Ученик» и «Оценка»**

На рис. 4 приведена физическая модель базы данных рассматриваемой предметной области.



**Рисунок 4** – Физическая модель базы данных предметной области «Школьный журнал»

Следует отметить, что большинство современных средств проектирования, в том числе и Microsoft® Visual Studio, не поддерживается вид связи между таблицами «многие-ко-многим». В рассматриваемой предметной области такой вид связи был выявлен между отношениями «Ученик» и «Родитель», «Преподаватель» и «Кабинет». Этот вид связи между таблицами может быть сведен к двум видам связи «один-ко-многим» в результате введения дополнительных таблиц, содержащих исключительно внешние ключи двух таблиц.

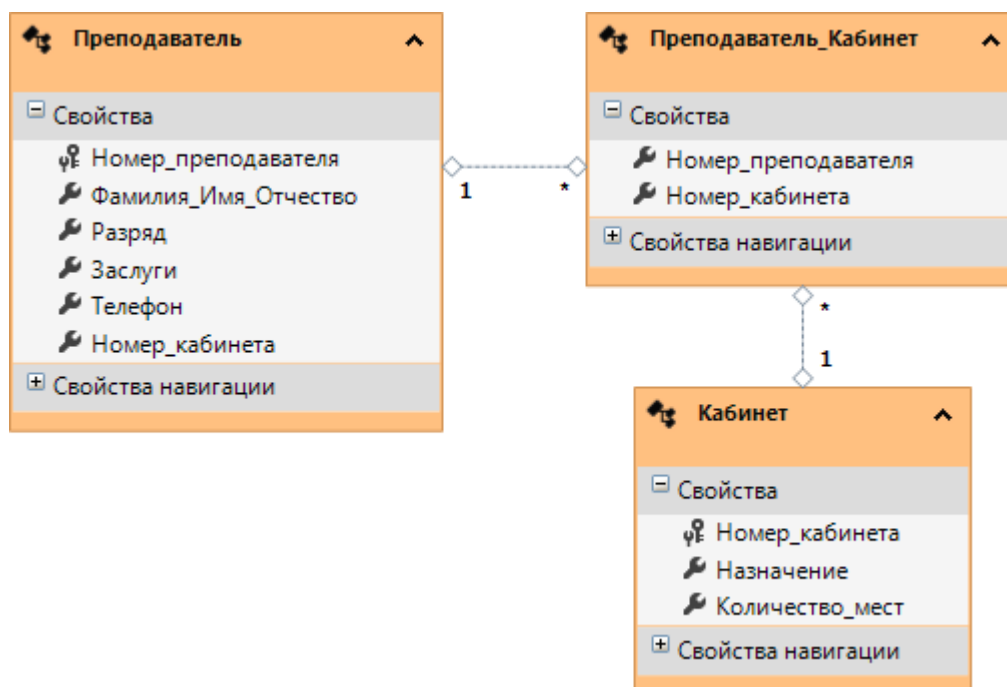
Выполнить SQL-запрос(ы) к базе данных школьный журнал: ?

```

1 CREATE TABLE Ученик_родитель
2 (
3     Номер_родителя SMALLINT NOT NULL REFERENCES Родитель (Номер_родителя),
4     Номер_билета_ученика SMALLINT NOT NULL REFERENCES Ученик (Номер_билета_ученика)
5 );
6 CREATE TABLE Преподаватель_кабинет
7 (
8     Номер_преподавателя TINYINT NOT NULL REFERENCES Преподаватель (Номер_преподавателя),
9     Номер_кабинета VARCHAR(5) NOT NULL REFERENCES Кабинет (Номер_кабинета)
10 )

```

**Рисунок 5** – Создание таблиц «Ученик\_родитель» и «Преподаватель\_кабинет»



**Рисунок 6** – Приведение вида связи «многие-ко-многим» к двум видам связи «один-ко-многим» для отношений «Преподаватель» и «Кабинет»

При описании связи между таблицами – ссылок из одной из них на первичный ключ или столбец с уникальными значениями другой можно указать действия, реализуемые СУБД при удалении или изменении такой связи. Действие, реализуемое при удалении или обновлении связи, позволяет реализовать ссылочную целостность данных и описывается после ключевых слов **ON DELETE** и **ON UPDATE** соответственно. В обоих случаях могут быть реализованы следующие действия:

**NO ACTION** – никакие действия не реализуются (значение по умолчанию).

- **CASCADE** – при удалении или изменении связи все записи в связанной таблице удаляются или обновляются.
- **SET NULL** – при удалении или изменении связи все записи в связанной таблице устанавливаются в **NULL**. При этом такие записи должны допускать нулевые значения.

- **SET DEFAULT** — при удалении или изменении связи все записи в связанной таблице устанавливаются в значения по умолчанию. При этом такие записи должны иметь такие значения.

Например, если ученик переводится в другую школу, то сохранение записей в школьном журнале об его успеваемости становится неактуальным, ненужным. Иными словами, удаление записи об ученике в таблице «Ученик» должно приводить и к удалению записей в связанной с ней таблице «Оценка». Как следствие, столбец таблицы «Оценка», ссылающийся на первичный ключ таблицы «Ученик», должен быть описан следующим образом:

```
Номер_билета_ученика SMALLINT NOT NULL REFERENCES Ученик
(Номер_билета_ученика) ON DELETE CASCADE
```

В случае необходимости все поименованные объекты таблицы базы данных могут быть изменены. Для этой цели используется оператор **ALTER TABLE**:

```
ALTER TABLE Имя таблицы
ADD Имя столбца Тип данных [Свойства] [Декларативные ограничения]
CHANGE Имя столбца [новое имя столбца] новый тип данных [новые свойства]
[новые декларативные ограничения]
DROP Имя столбца
ADD CONSTRAINT Имя ограничения Декларативное ограничение таблицы
DROP CONSTRAINT Имя ограничения
```

В одном операторе **ALTER TABLE** возможно только одно из следующих действий:

- добавление нового столбца (**ADD**),
- изменение существующего столбца (**CHANGE**),
- удаление существующего столбца (**DROP**),
- добавление нового ограничения (**ADD CONSTRAINT**),
- удаление существующего ограничения (**DROP CONSTRAINT**).

Например, изменение таблицы «Кабинет» с целью добавления нового столбца, описывающего состояние кабинета – «Отличное», «Хорошее», «Удовлетворительное» или «Плохое», следующее:

```
ALTER TABLE Кабинет
ADD Состояние NVARCHAR (20) NOT NULL DEFAULT N'Хорошее'
ALTER TABLE Кабинет
ADD CONSTRAINT condition CHECK (Состояние IN (N'Отличное', N'Хорошее',
N'Удовлетворительное', N'Плохое'))
```

Удаление таблицы реализуется следующим оператором:

```
DROP TABLE Имя таблицы
```

В заключении необходимо отметить следующее. В таблицах «Оценка», «Ученик\_Родитель» и «Преподаватель\_Кабинет» отсутствуют первичные ключи. Любой

SQL-запрос к этим таблицам будет отработан правильно. Несмотря на это, современные технологии объектно-реляционного проецирования требуют его наличия. Как следствие, все таблицы базы данных должны иметь первичные ключи, даже если их наличие избыточно.

Для организации взаимодействия с базами данных язык SQL предоставляет четыре основных оператора – **INSERT**, **UPDATE**, **DELETE** и **SELECT**. Эти операторы реализуют добавление, изменение, удаление и выборку данных соответственно.

Оператор **INSERT** имеет следующий формат:

```
INSERT INTO Имя таблицы [(Имя столбца, ..., Имя столбца)] VALUES (Значение, ..., Значение)
```

Если имена всех столбцов опущены, то данные добавляются во все столбцы таблицы. Если опускается значение какого-нибудь столбца, то оно либо может быть равно **NULL**, либо имеет значение по умолчанию. Часть этого оператора – ключевое слово **VALUES** и следующие за ним значения столбцов может быть заменена оператором **SELECT** (см. ниже). Добавление записей об учениках реализуется следующим образом:

**Выполнить SQL-запрос(ы) к базе данных школьный журнал:** ⓘ

```

1 INSERT INTO Ученик (Номер_билета_ученика, Фамилия_Имя_Отчество, Класс, Классный_Руководитель)
2   VALUES ('1', 'Петров Антон Сергеевич', '9А', '0');
3 INSERT INTO Ученик VALUES ('2', 'Иванов Андрей Иванович', '5Б', 'С математическим уклоном', '0')
```

Номер_билета_ученика	Фамилия_Имя_Отчество	Класс	Специализация_класса	Классный_руководитель
1	Петров Антон Сергеевич	9А	NULL	0
2	Иванов Андрей Иванович	5Б	С математическим уклоном	0

**Рисунок 7** – Команда **INSERT** и результат

Добавление данных об учениках будет реализовано СУБД только в том случае, если таблица «Преподаватель» содержит запись с данными о преподавателе, первичный ключ в которой равен 0.

Оператор **UPDATE**, реализующий изменение данных в таблице, имеет следующий формат:

```
UPDATE Имя таблицы SET (Имя столбца = Значение, ..., Имя столбца = Значение) [WHERE Условие]
```

Ключевое слово **WHERE** определяет условие выбора данных таблицы, подлежащих изменению (см. ниже оператор **SELECT**). Если это ключевое слово отсутствует, изменению подлежат все данные таблицы! Например, изменение записей об ученике

с номером ученического билета равном 1 при переводе его в класс 10А и сменен классного руководителя реализуется следующим образом:

Выполнить SQL-запрос(ы) к базе данных школьный журнал: ?

```

1 UPDATE Ученик
2 SET
3     Класс = '10А',
4     Классный_руководитель = '1'
5 WHERE
6     Номер_билета_ученика = '1'

```

Номер_билета_ученика	Фамилия_Имя_Отчество	Класс	Специализация_класса	Классный_руководитель
1	Петров Антон Сергеевич	10А	NULL	1
2	Иванов Андрей Иванович	5Б	С математическим уклоном	0

**Рисунок 8** – Команда UPDATE и результат

Оператор DELETE имеет самый простой формат:

**DELETE FROM** *Имя таблицы* [**WHERE** *Условие*]

Ключевое слово **WHERE** определяет условие выбора данных таблицы, подлежащих удалению (см. ниже оператор **SELECT**). Если это ключевое слово отсутствует, удаляются все данные таблицы! Например, удаление записи об ученике с номером ученического билета равном 1 реализуется следующим образом:

Выполнить SQL-запрос(ы) к базе данных школьный журнал: ?

```

1 DELETE FROM Ученик WHERE Номер_билета_ученика = '1'

```

Номер_билета_ученика	Фамилия_Имя_Отчество	Класс	Специализация_класса	Классный_руководитель
2	Иванов Андрей Иванович	5Б	С математическим уклоном	0

**Рисунок 9** – Команда DELETE и результат

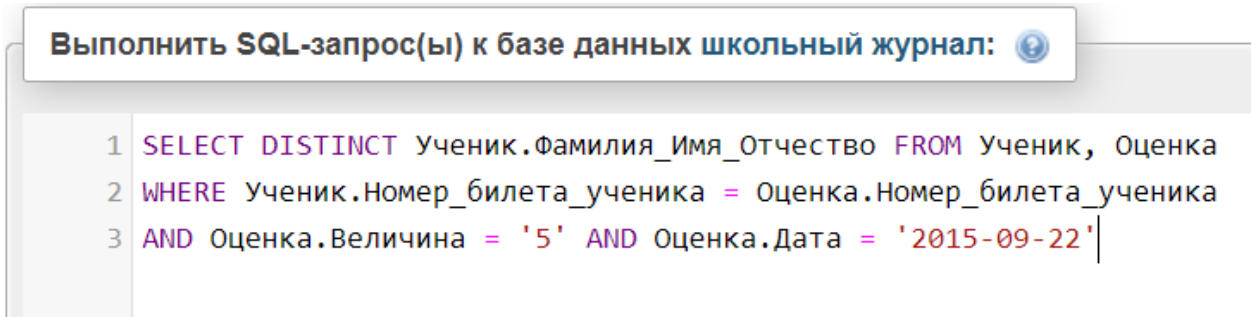
Оператор SELECT, реализующий выборку данных таблицы, имеет самый сложный формат:

**SELECT** \* или [**DISTINCT** или **ALL**] *Имя столбца*, ..., [**DISTINCT** или **ALL**] *Имя столбца*  
**FROM** *Имя таблицы* , ..., *Имя таблицы*  
 [**WHERE** *Условие*]  
 [**GROUP BY** *Имя столбца*, ..., *Имя столбца*]  
 [**HAVING** *Условие*]  
 [**ORDER BY** *Имя столбца* [**ASC** или **DESC**], ..., *Имя столбца* [**ASC** или **DESC**]]  
 [**INTO** *Имя таблицы* или **TEMP** *Имя таблицы*]

Непосредственно после ключевого слова **SELECT** перечисляются имена столбцов, из которых будут извлекаться данные. Имени столбца может предшествовать **DISTINCT** или **ALL**. В случае **DISTINCT** извлекаются неповторяющиеся данные столбца, в

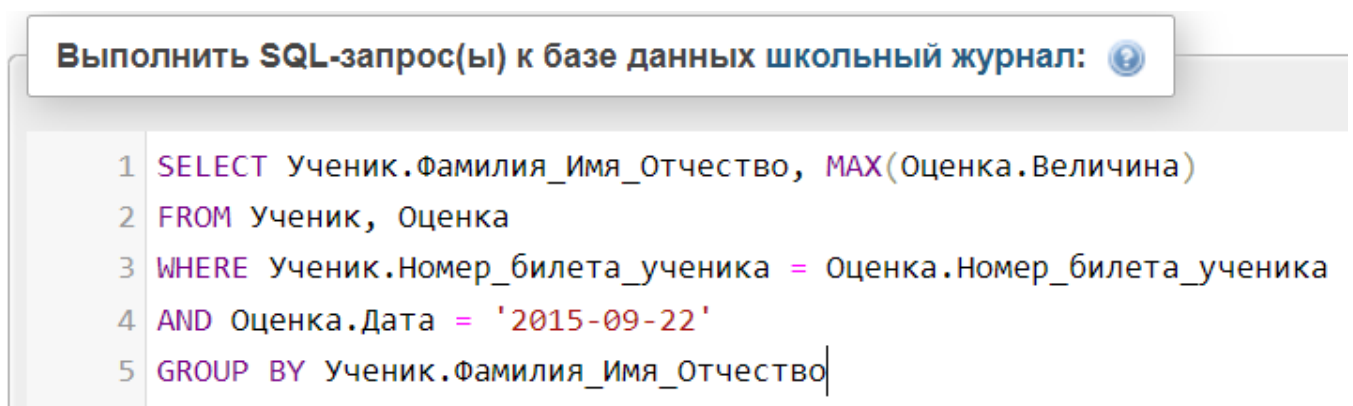


случае **ALL**– все. Значением по умолчанию является **ALL**. Данные этого запроса могут извлекаться из нескольких, как правило, связанных таблиц. Имена таблиц перечисляются после ключевого слова **FROM**. Ключевое слово **WHERE** определяет условие выбора данных таблицы (таблиц). Например, выбор имен всех учеников школы, получивших 22 сентября 2015 года хотя бы одну отличную оценку, реализуется следующим образом:



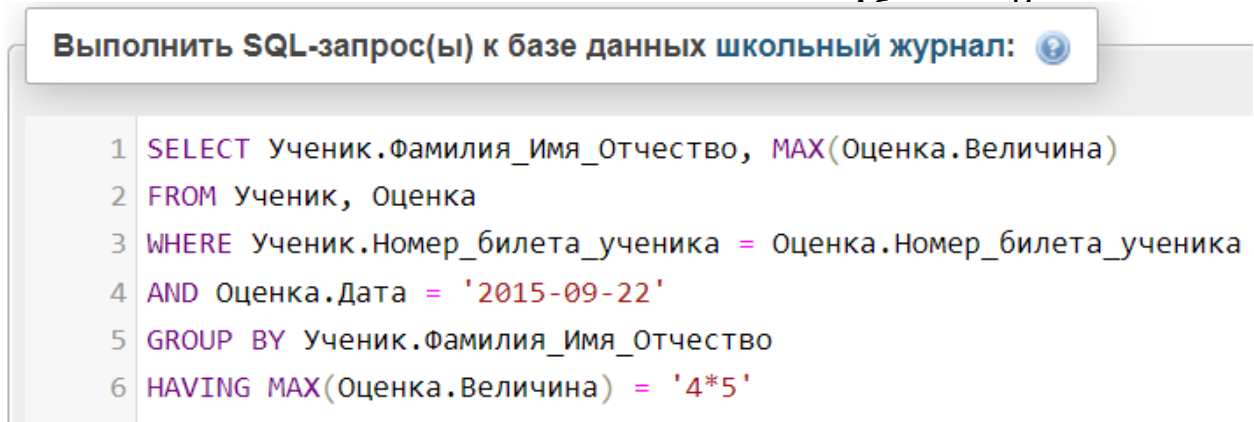
**Рисунок 10** – Команда SELECT

Ключевое слово **GROUP BY** используется совместно с агрегатными функциями Transact SQL для группировки результатов выборки по содержимому указанных столбцов таблицы (таблиц). Например, выборка всех учеников класса 10А, в соответствии с максимальной суммой полученных ими оценок 22 сентября 2015 года реализуется следующим образом:



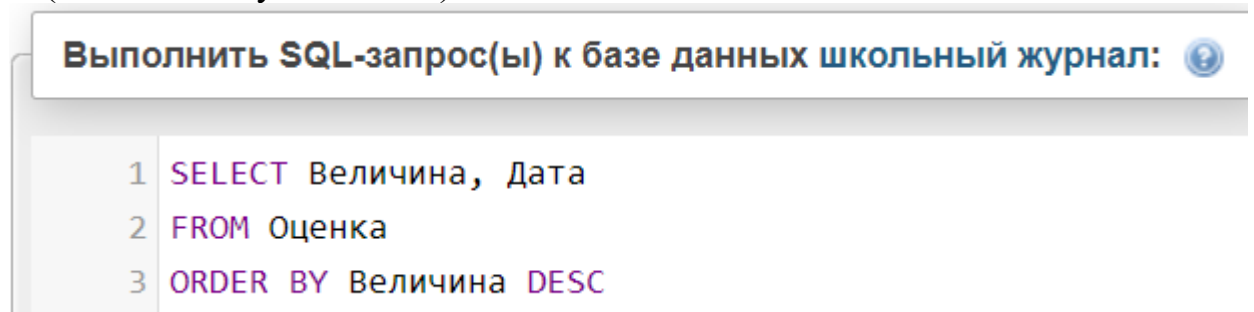
**Рисунок 11** – Команда SELECT с ключевым словом GROUP BY

Ключевое слово **HAVING** используется вместе с **GROUP BY** для того чтобы выбирать группы строк данных, удовлетворяющих указанному условию. Например, если в течение дня было 4 урока, то выборка всех отличников класса 10А, получивших 22 сентября 2015 по всем урокам отличные оценки реализуется следующим образом:

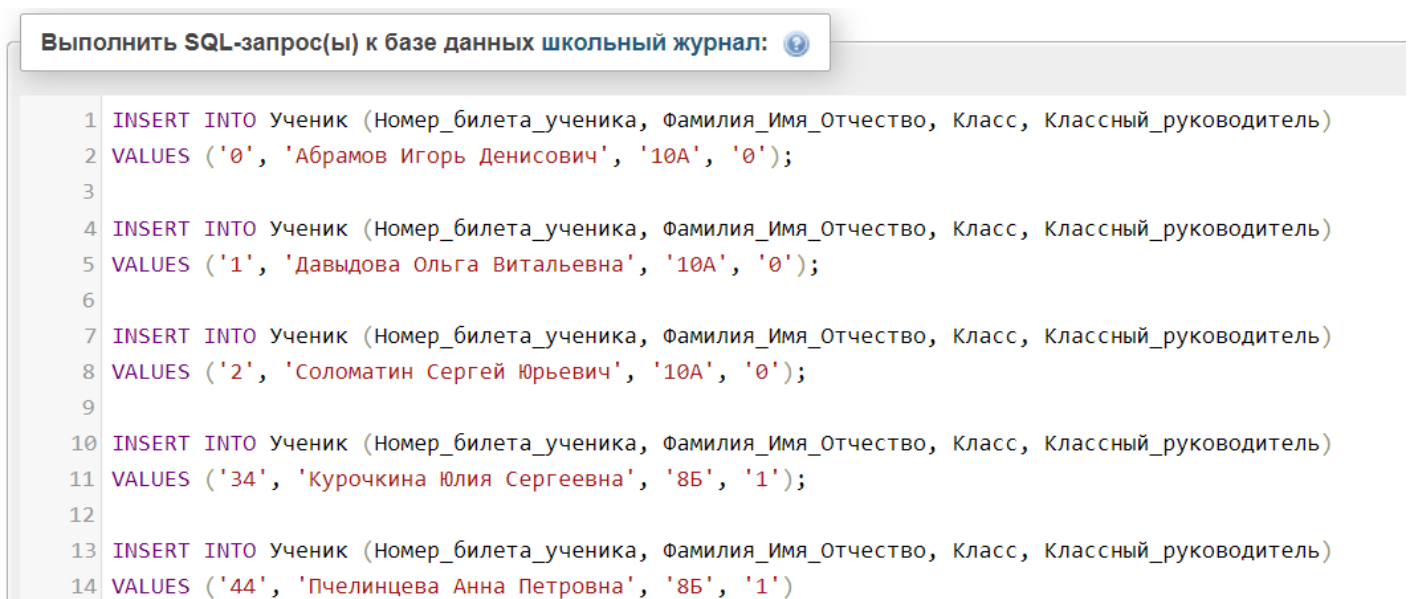


**Рисунок 12** – Команда SELECT с ключевым словом HAVING

Использование ключевого слова **ORDER BY** позволяет сортировать строки результирующей выборки по значениям указанных столбцов в восходящем порядке – **ASC** (значение по умолчанию) или нисходящем – **DESC**.



**Рисунок 13** – Команда SELECT с ключевым словом ORDER BY



**Рисунок 14** – Заполнение таблицы «Ученик»

Для выполнения этих команд необходимо в обозревателе серверов Microsoft® Visual Studio выделить базу данных, с которой мы реализуем взаимодействие, и в кон-

текстно-зависимом меню выбрать команду “Новый запрос”. Результат выполнения SQL-команд приведен на рисунке ниже.

+ Параметры

	Номер_билета_ученика	Фамилия_Имя_Отчество	Класс	Специализация_класса	Классный_руководитель
<input type="checkbox"/> Изменить  Копировать  Удалить	0	Абрамов Игорь Денисович	10А	NULL	0
<input type="checkbox"/> Изменить  Копировать  Удалить	1	Давыдова Ольга Витальевна	10А	NULL	0
<input type="checkbox"/> Изменить  Копировать  Удалить	2	Соломатин Сергей Юрьевич	10А	NULL	0
<input type="checkbox"/> Изменить  Копировать  Удалить	34	Курочкина Юлия Сергеевна	8Б	NULL	1
<input type="checkbox"/> Изменить  Копировать  Удалить	44	Пчелинцева Анна Петровна	8Б	NULL	1

↑ ☐ Отметить все С отмеченными: Изменить Копировать Удалить Экспорт

Рисунок 15 – Записи таблицы «Ученик»

Выполнить SQL-запрос(ы) к базе данных школьный журнал:

```

1 INSERT INTO Родитель (Номер_родителя, Фамилия_Имя_Отчество)
2 VALUES ('0', 'Абрамов Денис Сергеевич');
3
4 INSERT INTO Родитель (Номер_родителя, Фамилия_Имя_Отчество)
5 VALUES ('1', 'Абрамова Ольга Петровна');
6
7 INSERT INTO Родитель (Номер_родителя, Фамилия_Имя_Отчество)
8 VALUES ('2', 'Давыдов Виталий Игнатьевич');
9
10 INSERT INTO Родитель (Номер_родителя, Фамилия_Имя_Отчество)
11 VALUES ('3', 'Давыдова Анастасия Петровна');
12
13 INSERT INTO Родитель (Номер_родителя, Фамилия_Имя_Отчество)
14 VALUES ('4', 'Соломатин Юрий Владимирович');
15
16 INSERT INTO Родитель (Номер_родителя, Фамилия_Имя_Отчество)
17 VALUES ('5', 'Соломатина Виктоия Игоревна');
18
19 INSERT INTO Родитель (Номер_родителя, Фамилия_Имя_Отчество, Место_работы, Должность)
20 VALUES ('6', 'Курочкин юрий Владимирович', 'ЗАО "АвтоТрейд"', 'Генеральный директор');
21
22 INSERT INTO Родитель (Номер_родителя, Фамилия_Имя_Отчество)
23 VALUES ('7', 'Курочкина Людмила Петровна');
24
25 INSERT INTO Родитель (Номер_родителя, Фамилия_Имя_Отчество, Место_работы, Должность)
26 VALUES ('8', 'Пчелинцев Петр Анатольевич', 'ООО "Horns and hooves"', 'Юрист');
27
28 INSERT INTO Родитель (Номер_родителя, Фамилия_Имя_Отчество, Место_работы)
29 VALUES ('9', 'Пчелинцева-Пирогова Анастасия Борисовна', 'ООО "Horns and hooves"')|

```

Рисунок 16 – Заполнение таблицы «Родитель»

+ Параметры

		Номер_родителя	Номер_билета_ученика	Фамилия_Имя_Отчество	Место_работы	Должность
<input type="checkbox"/>	Изменить	0	0	Абрамов Денис Сергеевич	NULL	NULL
<input type="checkbox"/>	Изменить	1	0	Абрамова Ольга Петровна	NULL	NULL
<input type="checkbox"/>	Изменить	2	0	Давыдов Виталий Игнатьевич	NULL	NULL
<input type="checkbox"/>	Изменить	3	0	Давыдова Анастасия Петровна	NULL	NULL
<input type="checkbox"/>	Изменить	4	0	Соломатин Юрий Владимирович	NULL	NULL
<input type="checkbox"/>	Изменить	5	0	Соломатина Виктоия Игоревна	NULL	NULL
<input type="checkbox"/>	Изменить	6	0	Курочкин юрий Владимирович	ЗАО "АвтоТрейд"	Генеральный директор
<input type="checkbox"/>	Изменить	7	0	Курочкина Людмила Петровна	NULL	NULL
<input type="checkbox"/>	Изменить	8	0	Пчелинцев Петр Анатольевич	ООО "Horns and hooves"	Юрист
<input type="checkbox"/>	Изменить	9	0	Пчелинцева-Пирогова Анастасия Борисовна	ООО "Horns and hooves"	NULL

Рисунок 17 – Записи таблицы «Родитель»

Выполнить SQL-запрос(ы) к базе данных школьный журнал: ?

```

1 INSERT INTO Ученик_родитель VALUES ('0', '0');
2 INSERT INTO Ученик_родитель VALUES ('1', '0');
3 INSERT INTO Ученик_родитель VALUES ('2', '1');
4 INSERT INTO Ученик_родитель VALUES ('3', '1');
5 INSERT INTO Ученик_родитель VALUES ('4', '2');
6 INSERT INTO Ученик_родитель VALUES ('5', '2');
7 INSERT INTO Ученик_родитель VALUES ('6', '34');
8 INSERT INTO Ученик_родитель VALUES ('7', '34');
9 INSERT INTO Ученик_родитель VALUES ('8', '44');
10 INSERT INTO Ученик_родитель VALUES ('9', '44')

```

Рисунок 18 – Заполнение таблицы «Ученик\_родитель»

+ Параметры

Номер_родителя	Номер_билета_ученика
0	0
1	0
2	1
3	1
4	2
5	2
6	34
7	34
8	44
9	44

Рисунок 19 – Записи таблицы «Ученик\_Родитель»

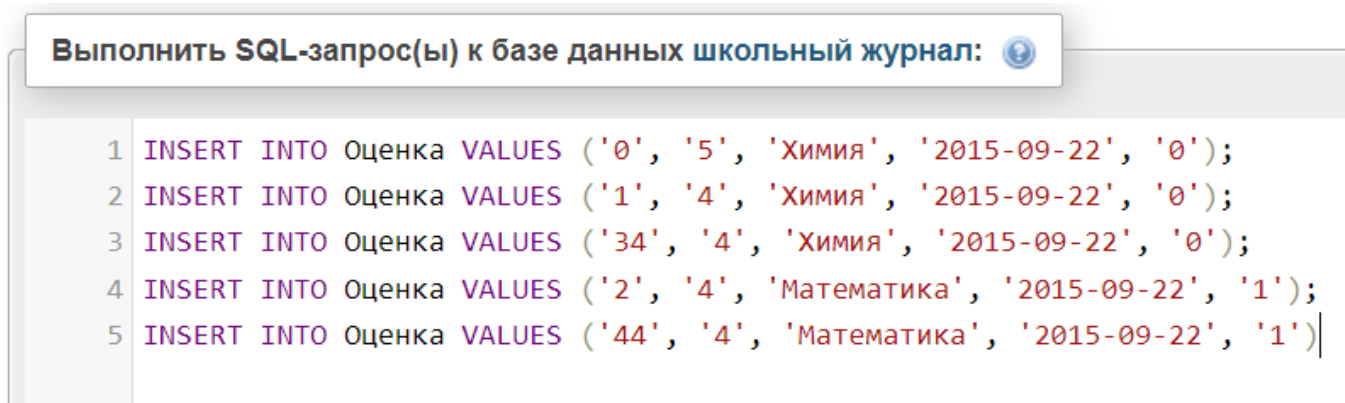


Рисунок 20 – Заполнение таблицы «Оценка»

+ Параметры

Номер_билета_ученика	Величина	Предмет	Дата	Преподаватель
0	5	Химия	2015-09-22	0
1	4	Химия	2015-09-22	0
34	4	Химия	2015-09-22	0
2	4	Математика	2015-09-22	1
44	4	Математика	2015-09-22	1

Рисунок 21 – Записи таблицы «Оценка»

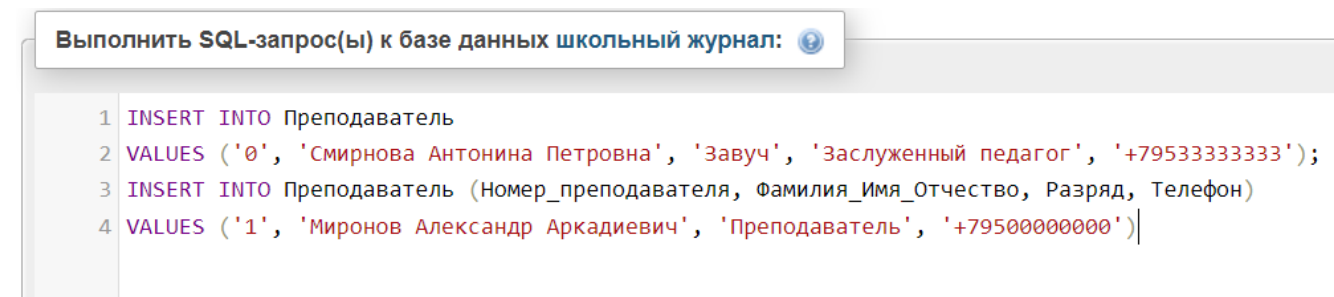








Рисунок 22 – Заполнение таблицы «Преподаватель»

+ Параметры

	Номер_преподавателя	Фамилия_Имя_Отчество	Разряд	Заслуги	Телефон
<input type="checkbox"/>  Изменить  Копировать  Удалить	0	Смирнова Антонина Петровна	Завуч	Заслуженный педагог	+7953333333
<input type="checkbox"/>  Изменить  Копировать  Удалить	1	Миронов Александр Аркадиевич	Преподаватель	NULL	+7950000000






 ☐ Отметить все    С отмеченными:  Изменить  Копировать  Удалить  Экспорт

Рисунок 23 – Записи таблицы «Преподаватель»

Выполнить SQL-запрос(ы) к базе данных школьный журнал: ?

```
1 INSERT INTO Кабинет VALUES ('201', 'Кабинет химии', '16');
2 INSERT INTO Кабинет VALUES ('317', 'Кабинет химии', '14');
3 INSERT INTO Кабинет VALUES ('215', 'Кабинет математики', '16');
4 INSERT INTO Кабинет VALUES ('120', 'Кабинет математики', '16');
5 INSERT INTO Кабинет VALUES ('120-1', 'Кабинет математики', '16');
```

Рисунок 24 – Заполнение таблицы «Кабинет»

+ Параметры

				Номер_кабинета	Назначение	Количество_мест
<input type="checkbox"/>		Изменить		Копировать		Удалить
	120	Кабинет математики	16			
<input type="checkbox"/>		Изменить		Копировать		Удалить
	120-1	Кабинет математики	16			
<input type="checkbox"/>		Изменить		Копировать		Удалить
	201	Кабинет химии	16			
<input type="checkbox"/>		Изменить		Копировать		Удалить
	215	Кабинет математики	16			
<input type="checkbox"/>		Изменить		Копировать		Удалить
	317	Кабинет химии	14			

☐ Отметить все

С отмеченными:

Изменить

Копировать

Удалить

Экспорт

Рисунок 25 – Записи таблицы «Кабинет»

Выполнить SQL-запрос(ы) к базе данных школьный журнал: ?

```
1 INSERT INTO Преподаватель_кабинет VALUES ('0', '201');
2 INSERT INTO Преподаватель_кабинет VALUES ('0', '317');
3 INSERT INTO Преподаватель_кабинет VALUES ('1', '215');
4 INSERT INTO Преподаватель_кабинет VALUES ('1', '120');
5 INSERT INTO Преподаватель_кабинет VALUES ('1', '120-1');
```

Рисунок 26 – Заполнение таблицы «Преподаватель\_кабинет»

+ Параметры

Номер_преподавателя	Номер_кабинета
0	201
0	317
1	215
1	120
1	120-1

Рисунок 27 – Записи таблицы «Преподаватель\_кабинет»



Пусть необходимо вывести для всех учеников классов 10А и 8Б получивших 22 сентября 2015 года любые оценки следующую информацию – информацию об ученике, предмет, по которому была получена оценка, фамилия имя отчество преподавателя. Иными словами, необходимо реализовать выборку связанной информации из нескольких таблиц.

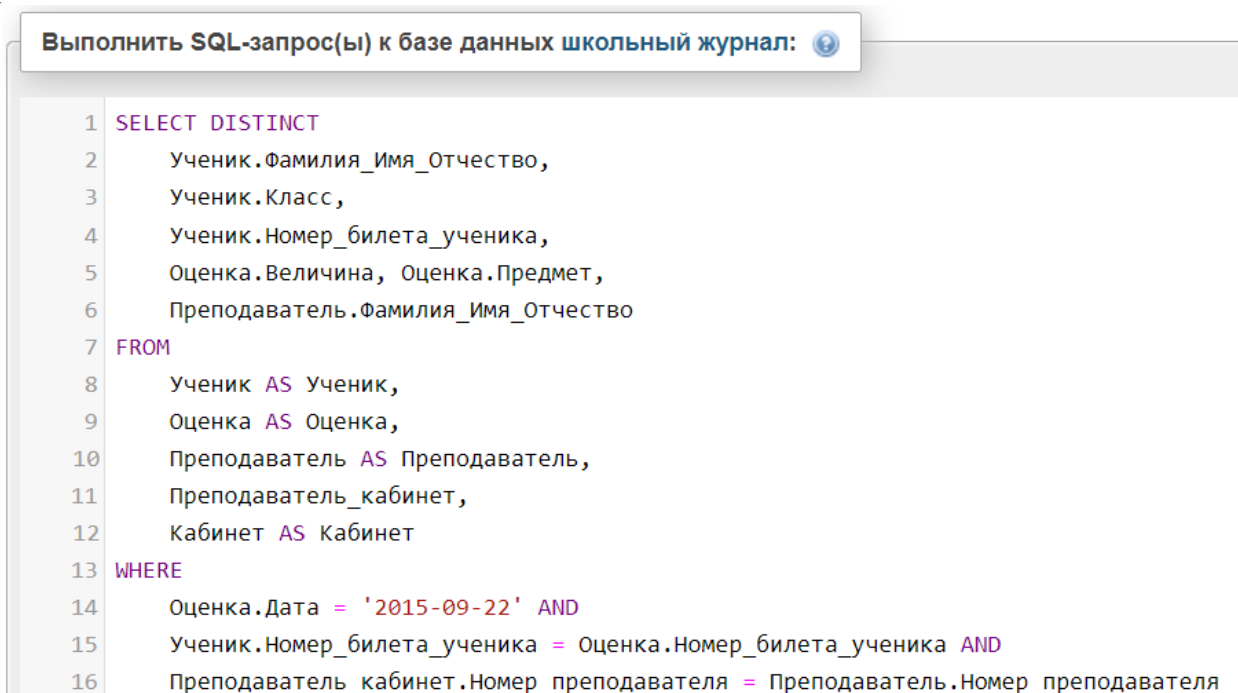


Рисунок 28 – Запрос выборки об учениках

+ Параметры

Фамилия_Имя_Отчество	Класс	Номер_билета_ученика	Величина	Предмет	Фамилия_Имя_Отчество
Абрамов Игорь Денисович	10А	0	5	Химия	Смирнова Антонина Петровна
Давыдова Ольга Витальевна	10А	1	4	Химия	Смирнова Антонина Петровна
Курочкина Юлия Сергеевна	8Б	34	4	Химия	Смирнова Антонина Петровна
Соломатин Сергей Юрьевич	10А	2	4	Математика	Смирнова Антонина Петровна
Пчелинцева Анна Петровна	8Б	44	4	Математика	Смирнова Антонина Петровна
Абрамов Игорь Денисович	10А	0	5	Химия	Миронов Александр Аркадиевич
Давыдова Ольга Витальевна	10А	1	4	Химия	Миронов Александр Аркадиевич
Курочкина Юлия Сергеевна	8Б	34	4	Химия	Миронов Александр Аркадиевич
Соломатин Сергей Юрьевич	10А	2	4	Математика	Миронов Александр Аркадиевич
Пчелинцева Анна Петровна	8Б	44	4	Математика	Миронов Александр Аркадиевич

Рисунок 29 – Выборка информации об учениках

Теперь предположим, что ученик 10А класса Абрамов Игорь Денисович переводится в другую школу, а у Пчелинцевой Анны Петровны изменяется номер билета ученика с 44 на 43. При наличии ключевых слов **ON UPDATE CASCADE** и **ON DELETE CASCADE** при описании связей между таблицами (внешних ключей) удаление ученика Абрамов Игорь Денисович из таблицы «Ученик» приведет к удалению записей о его оценках и о родителях из таблиц «Оценка» и «Родитель» соответственно. В этих же таблицах изменится и номер билета ученика для Пчелинцевой Анны Петровны.

Выполнить SQL-запрос(ы) к базе данных школьный журнал: ?

```

1 DELETE FROM Ученик
2     WHERE Фамилия_Имя_Отчество = 'Абрамов Игорь Денисович';
3 UPDATE Ученик SET Номер_билета_ученика = 43
4     WHERE Фамилия_Имя_Отчество = 'Пчелицина Анна Петровна'

```

Рисунок 30 – Запрос удаления данных об учениках

+ Параметры

Фамилия_Имя_Отчество	Класс	Номер_билета_ученика	Величина	Предмет	Фамилия_Имя_Отчество
Давыдова Ольга Витальевна	10А	1	4	Химия	Смирнова Антонина Петровна
Курочкина Юлия Сергеевна	8Б	34	4	Химия	Смирнова Антонина Петровна
Соломатин Сергей Юрьевич	10А	2	4	Математика	Смирнова Антонина Петровна
Пчелинцева Анна Петровна	8Б	44	4	Математика	Смирнова Антонина Петровна
Давыдова Ольга Витальевна	10А	1	4	Химия	Миронов Александр Аркадиевич
Курочкина Юлия Сергеевна	8Б	34	4	Химия	Миронов Александр Аркадиевич
Соломатин Сергей Юрьевич	10А	2	4	Математика	Миронов Александр Аркадиевич
Пчелинцева Анна Петровна	8Б	44	4	Математика	Миронов Александр Аркадиевич

Рисунок 31 – Выборка измененной информации об учениках

В рисунках 23 и 25 обращает на себя внимание тот факт, что записи об учениках выводятся дважды – по числу преподавателей, проводивших уроки с 10А и 8Б классами. Для исключения ненужных записей могут быть использованы так называемые соединения записей связанных таблиц. Различают внутреннее соединение, внешние соединения и перекрестное соединение. Внутреннее соединение записей таблиц реализует пересечение множеств всех записей связанных таблиц. Оно позволяет выбирать записи таблиц на основе значений связующих их столбцов. Внутреннее соединение следует за ключевым словом **FROM** и записывается следующим образом:

**INNER JOIN** Имя связанной таблицы **ON** Условие

Пример – выборка всех учеников классов 10А и 8Б получивших 22 сентября 2015 года оценки «хорошо» по химии.

Выполнить SQL-запрос(ы) к базе данных школьный журнал: ?

```

1 SELECT
2     Ученик.Фамилия_Имя_Отчество, Ученик.Класс
3 FROM
4     Ученик AS Ученик
5 INNER JOIN
6     Оценка AS Оценка
7 ON
8     Оценка.Дата = '2015-09-22' AND
9     Ученик.Номер_билета_ученика = Оценка.Номер_билета_ученика AND
10    Оценка.Предмет = 'Химия' AND
11    Оценка.Величина = 4 |

```



**Рисунок 32** – Внутреннее соединение таблиц «Ученик» и «Оценка»

+ Параметры

Фамилия_Имя_Отчество	Класс
Давыдова Ольга Витальевна	10А
Курочкина Юлия Сергеевна	8Б

**Рисунок 33** – Результат соединения

Внешние соединения бывают левыми, правыми и полным. Внешние соединения начинаются со следующих ключевых слов:

- **LEFT JOIN** (или **LEFT OUTER JOIN**) *Имя связанной таблицы ON Условие.*  
Результирующий набор левого внешнего соединения включает все строки из левой таблицы, заданной в предложении **LEFT JOIN**, а не только те, в которых соединяемые столбцы соответствуют друг другу. Если строка в левой таблице не имеет совпадающей строки в правой таблице, результирующий набор строк содержит значения **NULL** для всех столбцов списка выбора из правой таблицы.

+ Параметры

Фамилия_Имя_Отчество	Класс
Давыдова Ольга Витальевна	10А
Курочкина Юлия Сергеевна	8Б
Соломатин Сергей Юрьевич	10А
Пчелинцева Анна Петровна	8Б

**Рисунок 34** – Левое внешнее соединение таблиц «Ученик» и «Оценка»

- **RIGHT JOIN** (или **RIGHT OUTER JOIN**) *Имя связанной таблицы ON Условие.*

Правое внешнее соединение является обратным для левого внешнего соединения. Возвращаются все строки правой таблицы. Для левой таблицы возвращаются значения **NULL** каждый раз, когда строка правой таблицы не имеет совпадающей строки в левой таблице.

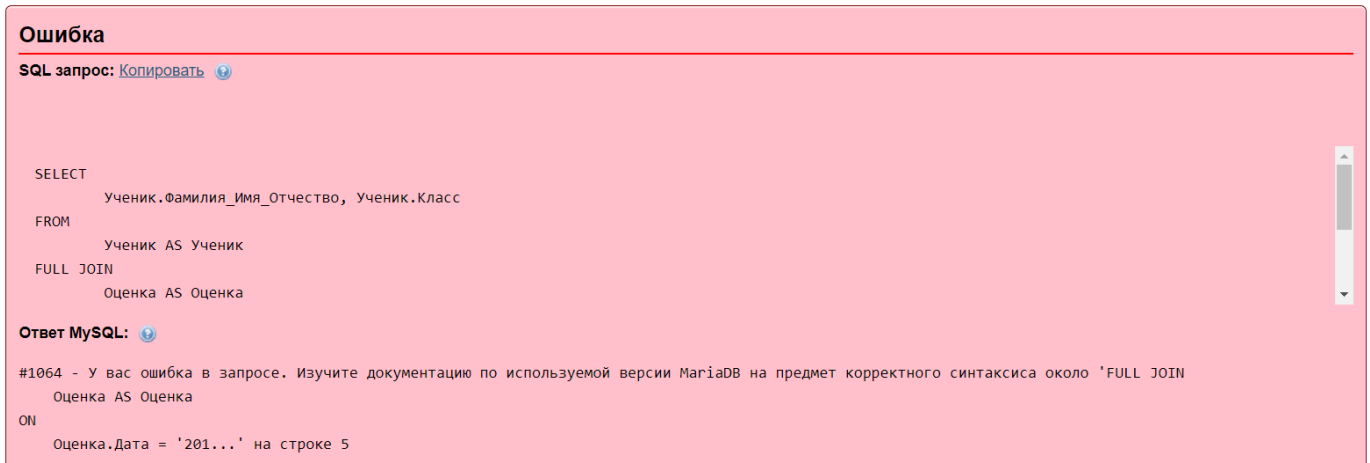
+ Параметры

Фамилия_Имя_Отчество	Класс
NULL	NULL
Давыдова Ольга Витальевна	10А
Курочкина Юлия Сергеевна	8Б
NULL	NULL
NULL	NULL

**Рисунок 34** – Правое внешнее соединение таблиц «Ученик» и «Оценка»

- **FULL JOIN** (или **FULL OUTER JOIN**) *Имя связанной таблицы* **ON** *Условие*.  
Полное внешнее соединение возвращает все строки из правой и левой таблицы. Каждый раз, когда строка не имеет соответствия в другой таблице, столбцы списка выбора другой таблицы содержат значения **NULL**. Если между таблицами имеется соответствие, вся строка результирующего набора содержит значения данных из базовых таблиц.

Однако в MySQL не поддерживается **FULL JOIN** и при попытке ввести полное внешнее соединение появляется ошибка.



**Рисунок 35** – Полное внешнее соединение таблиц «Ученик» и «Оценка»

Перекрестное соединение возвращает все строки из левой таблицы. Каждая строка из левой таблицы соединяется со всеми строками из правой таблицы. Перекрестные соединения называются также декартовым произведением. Такой тип соединения описывается следующим образом:

**CROSS JOIN** *Имя связанной таблицы*

удаление или модификацию. Для исключения подобных моментов служат ключевые слова **WITH CHECK OPTION** в определении представления. Фраза размещается в определении представления, и все команды модификации будут подвергаться проверке.