ПРАКТИЧЕСКАЯ РАБОТА по теме:

Основные конструкции языка SQL.

Каждый сайт в Интернете, любой проект, обрабатывающий значительный объем информации, вынужден хранить эту информацию в тех или иных базах данных (БД). Подавляющее большинство проектов информацию сохраняют в БД реляционного типа, делая записи в различных подобиях таблиц. Как внесение новых записей, так и обращение к имеющимся, осуществляется с благодаря использованию запросов, составляемых конструкциями SQL (structured query language) – непроцедурного декларативного языка структурированных запросов. В нашем случае это подразумевает, что, используя конструкции SQL, мы будем обращаться к БД, сообщая что нужно сделать с данными, но не указывая способ, как именно это нужно сделать.

Основываясь на указанных стандартах языка SQL, ряд организаций выпустили свои, расширенные версии стандартов указанного языка. Подобные версии иногда называют диалектами SQL.

Варианты спецификаций SQL разрабатываются компаниями и сообществами и служат, соответственно, для работы с разными СУБД (Системами Управления Базами Данных) – системами программ, заточенных под работу с продуктами из своей инфраструктуры.

Наиболее применяемые на сегодня СУБД, использующие свои стандарты (расширения) SQL:

MySQL – СУБД, принадлежащая компании Oracle.

PostgreSQL – свободная СУБД, поддерживаемая и развиваемая сообществом.

Microsoft SQL Server – СУБД, принадлежащая компании Microsoft. Применяет диалект Transact-SQL (T-SQL).

Благодаря тому, что диалекты SQL что создаются, специфицируются и используются разными организациями, имеют как общие черты, так и ряд отличий в возможностях расширений.

Общими чертами диалектов являются основные конструкции, применимые практически без отличий во многих реляционных БД. Основные отличия диалектов состоят в различиях использованных типов данных, количеством, реализацией и детальными возможностями команд. Разные диалекты применяют как разные наборы зарезервированных слов, так и разные наборы команд.

Здесь мы будем рассматривать запросы, применяя конструкции из спецификаций диалекта T-SQL.

Коснемся классификации SQL запросов.

Выделяют такие виды SQL запросов:

DDL (**Data Definition Language) -** язык определения данных. Задачей DDL запросов является создание БД и описание ее структуры. Запросами такого вида устанавливаются правила того, в каком виде различные данные будут размещаться в БД.

DML (Data Manipulation Language) - язык манипулирования данными. В число запросов этого типа входят различные команды, используя которые непосредственно производятся некоторые манипуляции с данными. DML-запросы нужны для добавления изменений в уже внесенные данные, для получения данных из БД, для их сохранения, для обновления различных записей и для их удаления из БД. В число элементов DML-обращений входит основная часть SQL операторов.

DCL (Data Control Language) - язык управления данными. Включает в себя запросы и команды, касающиеся разрешений, прав и других настроек СУБД.

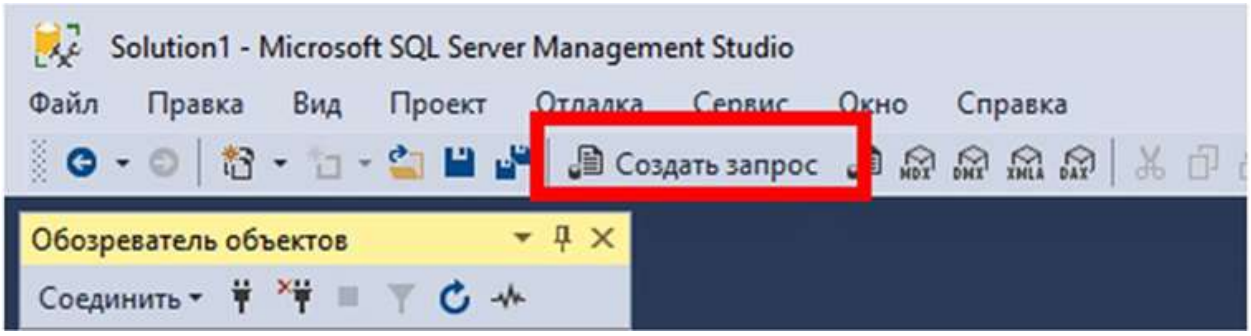
TCL (Transaction Control Language) - язык управления транзакциями. Конструкции такого типа применяют чтобы управлять изменениями, которые производятся с использованием DML запросов. Конструкции TCL позволяют нам производить объединение DML запросов в наборы транзакций.

Основные типы SQL запросов по их видам:

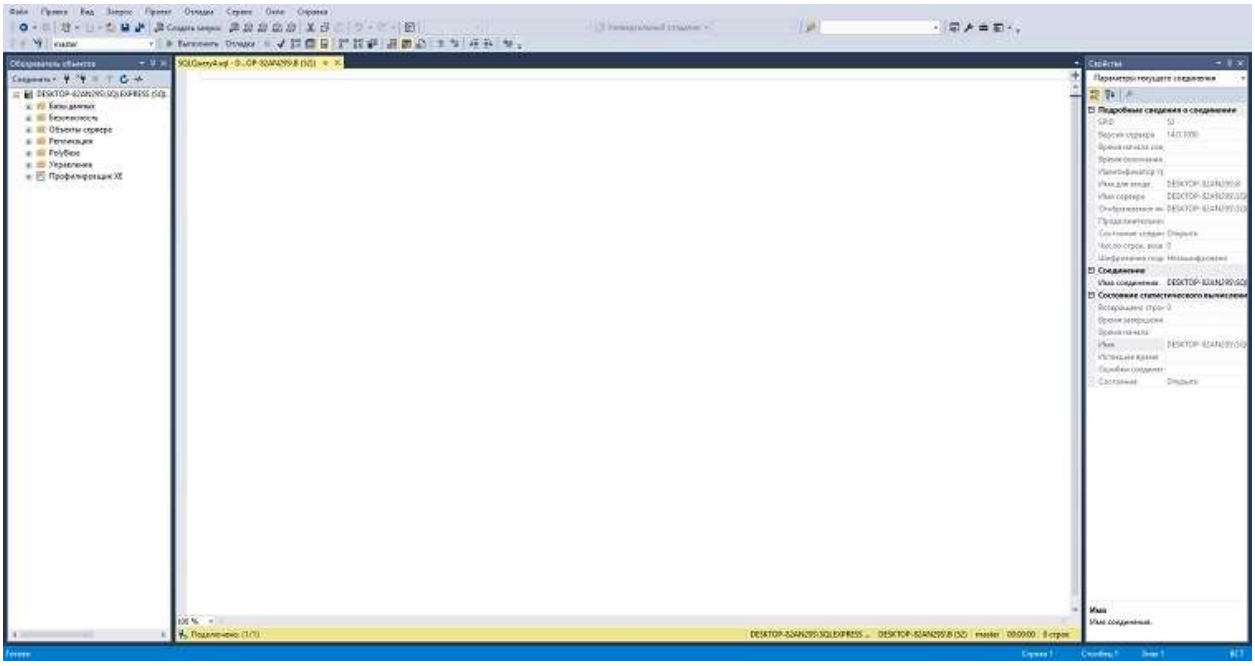
Создание и настройка базы данных

Нам нужна будет для примеров БД MS SQL Server 2017 и MS SQL Server Management Studio 2017.

Рассмотрим последовательность действий того, как создать SQL запрос. Воспользовавшись Management Studio, для начала создадим новый редактор скриптов. Чтобы это сделать, на стандартной панели инструментов выберем «Создать запрос». Или воспользуемся клавиатурной комбинацией Ctrl+N.



Нажимая кнопку «Создать запрос» в Management Studio, мы открываем тестовый редактор, используя который можно производить написание SQL запросов, сохранять их и запускать.

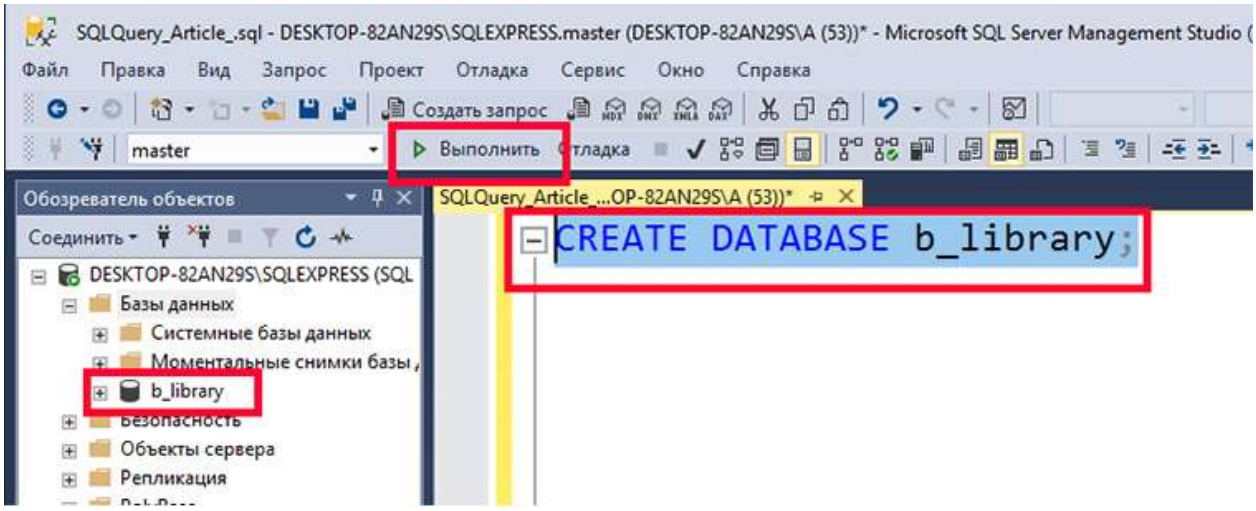


Используем для начала простые запросы SQL, благодаря которым можно создать и настроить новую БД, чтобы получить возможность в дальнейшем с ней работать.

Создадим новую БД с именем «b_library» для библиотеки книг. Чтобы это делать наберем в редакторе такой SQL запрос:

```
CREATE DATABASE b_library;
```

Далее выделим введенный текст и нажмем F5 или кнопку «Выполнить». У нас создастся БД «b_library».



Все дальнейшие манипуляции мы можем провести с этой созданной нами БД. Для этого сначала подключимся к этой базе:

```
USE b_library;
```

В БД «b_library» создадим таблицу авторов «**tAuthors**» с такими столбцами: AuthorId, AuthorFirstName, AuthorLastName, AuthorAge:

```
CREATE TABLE tAuthors (
AuthorId INT IDENTITY (1, 1) NOT NULL,
AuthorFirstName NVARCHAR (20) NOT NULL,
AuthorLastName NVARCHAR (20) NOT NULL,
AuthorAge INT NOT NULL
);
```

Заполним нашу таблицу таким авторами: Александр Пушкин, Сергей Есенин, Джек Лондон, Шота Руставели и Рабиндранат Тагор. Для этого используем такой SQL запрос:

```
INSERT tAuthors VALUES
('Александр', 'Пушкин', '37'),
('Сергей', 'Есенин', '30'),
('Джек', 'Лондон', '40'),
('Шота', 'Руставели', '44'),
('Рабиндранат', 'Тагор', '80');
```

Мы можем посмотреть в «**tAuthors**» записи, путем отправления в СУБД простого SQL запроса:

```
SELECT * FROM tAuthors;
```

Результаты | Сообщения

	AuthorId	AuthorFirstName	AuthorLastName	AuthorAge
1	1	Александр	Пушкин	37
2	2	Сергей	Есенин	30
3	3	Джек	Лондон	40
4	4	Шота	Руставели	44
5	5	Рабиндранат	Тагор	80

В нашей БД «**b_library**» мы создали первую таблицу «**tAuthors**», заполнили «**tAuthors**» авторами книг и теперь можем рассмотреть различные примеры SQL запросов, которыми мы сможем взаимодействовать с БД.

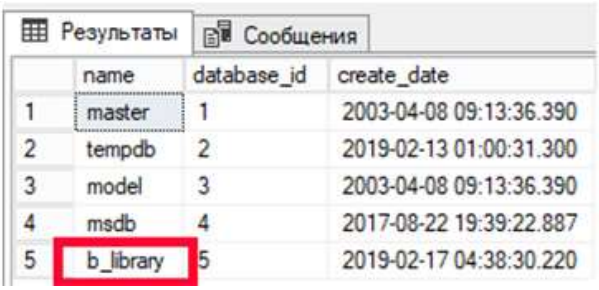
Примеры простых запросов SQL к базам данных.

Рассмотрим основные запросы SQL.

SELECT

1) Выведем все имеющиеся у нас БД:

```
SELECT name, database_id, create_date
FROM sys.databases;
```

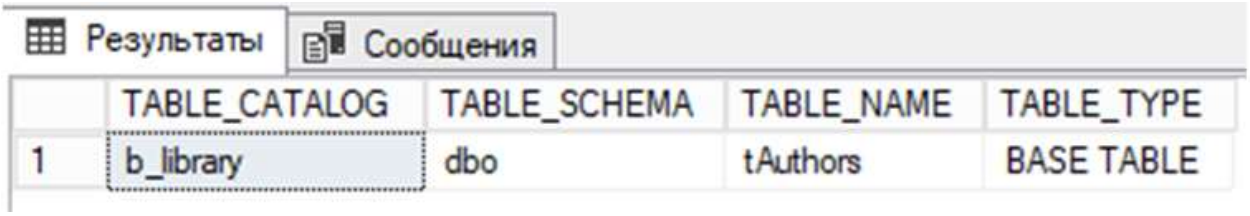


Результаты | Сообщения

	name	database_id	create_date
1	master	1	2003-04-08 09:13:36.390
2	tempdb	2	2019-02-13 01:00:31.300
3	model	3	2003-04-08 09:13:36.390
4	msdb	4	2017-08-22 19:39:22.887
5	b_library	5	2019-02-17 04:38:30.220

2) Выведем все таблицы в созданной нами ранее БД «**b_library**»:

```
SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_TYPE='BASE
TABLE'
```

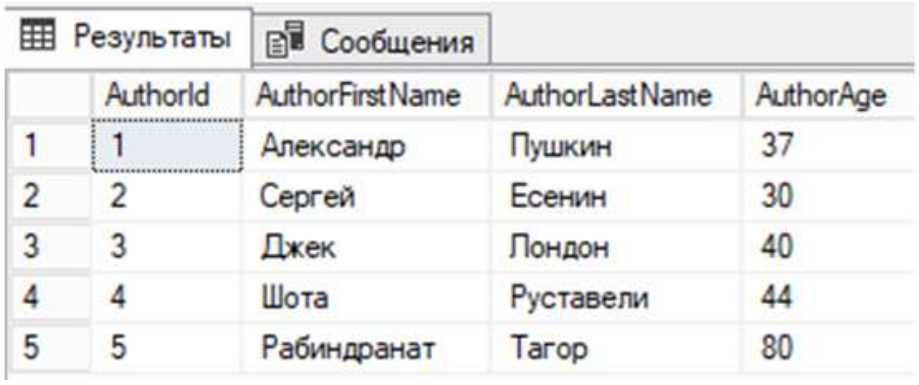


Результаты | Сообщения

	TABLE_CATALOG	TABLE_SCHEMA	TABLE_NAME	TABLE_TYPE
1	b_library	dbo	tAuthors	BASE TABLE

3) Выводим еще раз имеющиеся у нас записи по авторам книг из созданной выше «**tAuthors**»:

```
SELECT * FROM tAuthors;
```

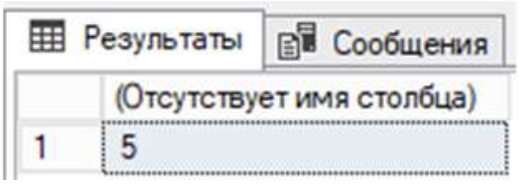


Результаты | Сообщения

	AuthorId	AuthorFirstName	AuthorLastName	AuthorAge
1	1	Александр	Пушкин	37
2	2	Сергей	Есенин	30
3	3	Джек	Лондон	40
4	4	Шота	Руставели	44
5	5	Рабиндранат	Тагор	80

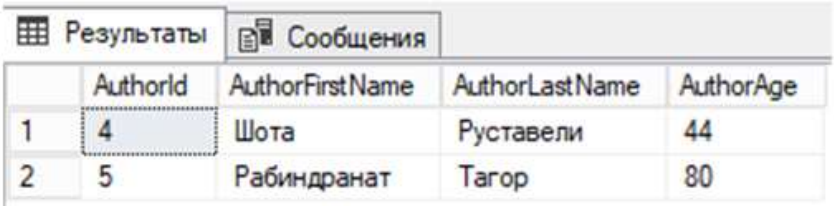
4) Выведем информацию о том, сколько у нас имеется записей строк в «**tAuthors**»:

```
SELECT count(*) FROM tAuthors;
```



5) Выведем из «**tAuthors»** две записи, начиная с четвертой. Используя ключевое слово OFFSET, пропустим первые три записи, а благодаря использованию ключевого слова FETCH – обозначим выборку только следующих 2 строк (ONLY):

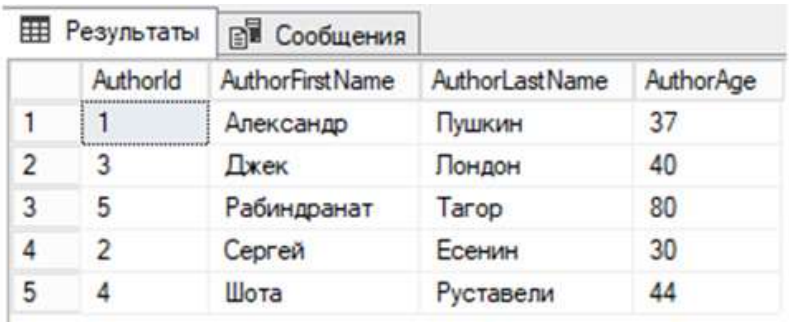
```
SELECT * FROM tAuthors
ORDER BY AuthorId
OFFSET 3 ROWS
FETCH NEXT 2 ROWS ONLY;
```



	AuthorId	AuthorFirstName	AuthorLastName	AuthorAge
1	4	Шота	Руставели	44
2	5	Рабиндранат	Тагор	80

6) Выведем из «tAuthors» все записи с сортировкой в алфавитном порядке по первой букве имени автора:

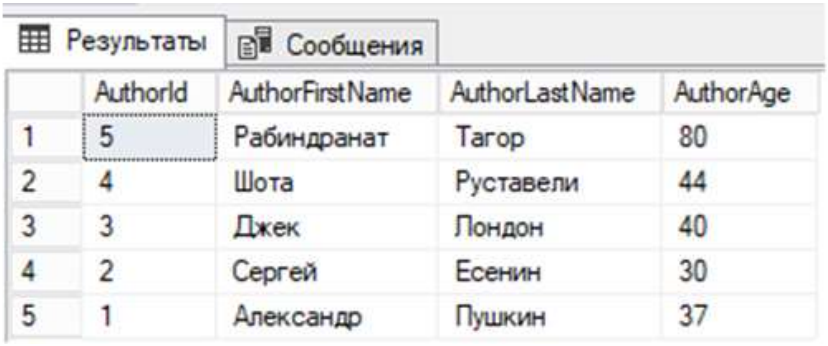
```
SELECT * FROM tAuthors ORDER BY AuthorFirstName;
```



	AuthorId	AuthorFirstName	AuthorLastName	AuthorAge
1	1	Александр	Пушкин	37
2	3	Джек	Лондон	40
3	5	Рабиндранат	Тагор	80
4	2	Сергей	Есенин	30
5	4	Шота	Руставели	44

7) Выведем из «**tAuthors»** данные, предварительно по AuthorId отсортировав их по убыванию:

```
SELECT * FROM tAuthors ORDER BY AuthorId DESC;
```

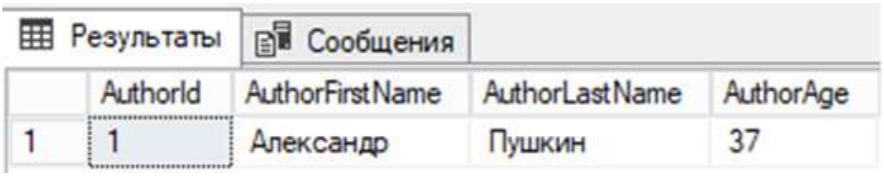


Результаты | Сообщения

	AuthorId	AuthorFirstName	AuthorLastName	AuthorAge
1	5	Рабиндранат	Тагор	80
2	4	Шота	Руставели	44
3	3	Джек	Лондон	40
4	2	Сергей	Есенин	30
5	1	Александр	Пушкин	37

8) Выберем записи из «**tAuthors»**, значение AuthorFirstName у которых соответствует имени «Александр»:

```
SELECT * FROM tAuthors WHERE AuthorFirstName='Александр';
```

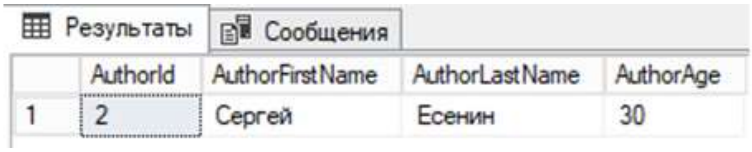


Результаты | Сообщения

	AuthorId	AuthorFirstName	AuthorLastName	AuthorAge
1	1	Александр	Пушкин	37

9) Выберем из «**tAuthors**» записи, где имя автора AuthorFirstName начинается с «се»:

```
SELECT * FROM tAuthors WHERE AuthorFirstName LIKE 'се%';
```

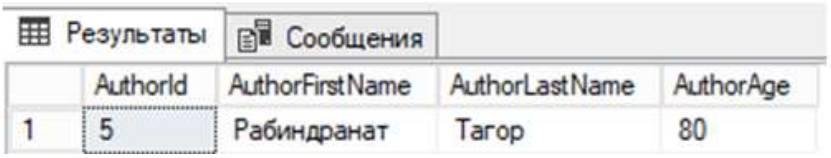


Результаты | Сообщения

	AuthorId	AuthorFirstName	AuthorLastName	AuthorAge
1	2	Сергей	Есенин	30

10) Выберем из «**tAuthors**» записи, в которых имя автора (AuthorFirstName) заканчивается на «ат»:

```
SELECT * FROM tAuthors WHERE AuthorFirstName LIKE '%ат' ORDER BY AuthorId;
```

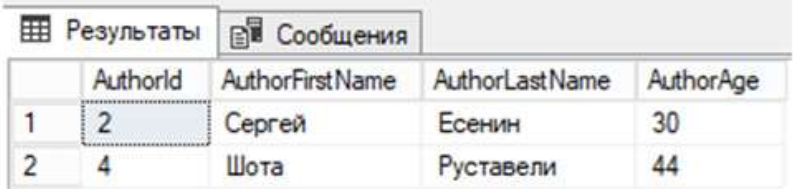


Результаты | Сообщения

	AuthorId	AuthorFirstName	AuthorLastName	AuthorAge
1	5	Рабиндранат	Тагор	80

11) Сделаем выборку всех строк из «tAuthors», значение AuthorId в которых равняется 2 или 4:

```
SELECT * FROM tAuthors WHERE AuthorId IN (2,4);
```

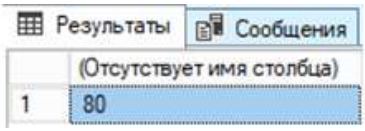


Результаты | Сообщения

	AuthorId	AuthorFirstName	AuthorLastName	AuthorAge
1	2	Сергей	Есенин	30
2	4	Шота	Руставели	44

12) Выберем в «tAuthors» такую запись AuthorAge, значение которой - наибольшее:

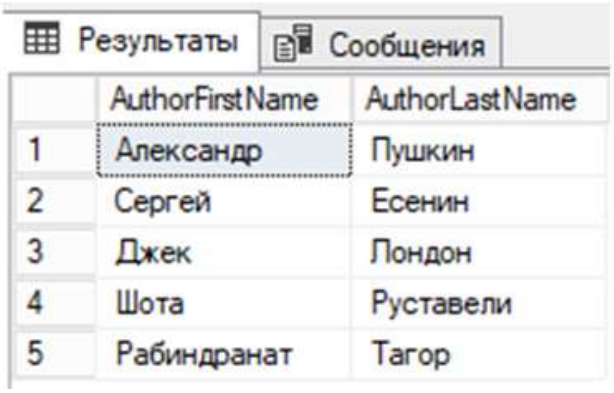
```
SELECT max(AuthorAge) FROM tAuthors;
```



	(Отсутствует имя столбца)
1	80

13) Проведем выборку из «tAuthors» по столбцам AuthorFirstName и AuthorLastName:

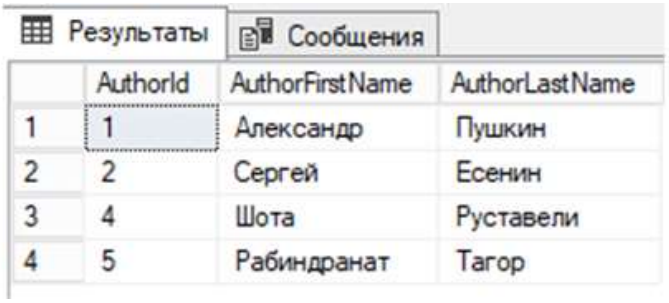
```
SELECT AuthorFirstName, AuthorLastName FROM tAuthors;
```



	AuthorFirstName	AuthorLastName
1	Александр	Пушкин
2	Сергей	Есенин
3	Джек	Лондон
4	Шота	Руставели
5	Рабиндранат	Тагор

14) Получим из «tAuthors» все строки, у которых AuthorId не равняется трем:

```
SELECT AuthorId, AuthorFirstName, AuthorLastName  FROM tAuthors WHERE AuthorId!='3';
```



	AuthorId	AuthorFirstName	AuthorLastName
1	1	Александр	Пушкин
2	2	Сергей	Есенин
3	4	Шота	Руставели
4	5	Рабиндранат	Тагор

INSERT

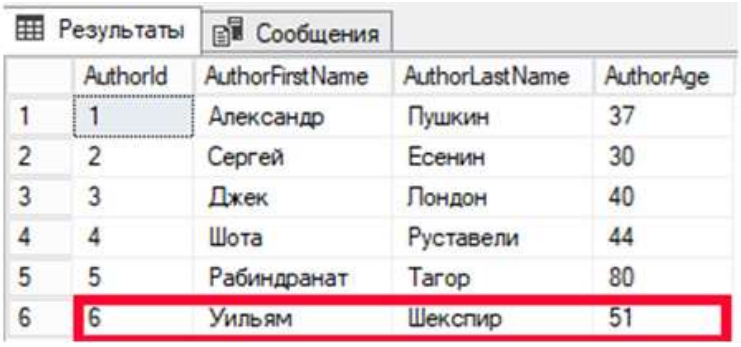
INSERT – это вид запроса SQL, при применении которого СУБД выполняет добавление новых записей в БД.

Добавим в «**tAuthors»** нового автора – Уильяма Шекспира, 51 год. Соответственно в поле AuthorFirstName добавится Уильям, в AuthorLastName добавится Шекспир, в AuthorAge – 51. В AuthorId, в нашем случае, автоматически добавится значение, инкрементированное от предыдущего на 1.

```
INSERT INTO tAuthors VALUES ('Уильям', 'Шекспир', '51');
```

Проверим:

```
SELECT * FROM tAuthors;
```



Результаты | Сообщения

	AuthorId	AuthorFirstName	AuthorLastName	AuthorAge
1	1	Александр	Пушкин	37
2	2	Сергей	Есенин	30
3	3	Джек	Лондон	40
4	4	Шота	Руставели	44
5	5	Рабиндранат	Тагор	80
6	6	Уильям	Шекспир	51

UPDATE

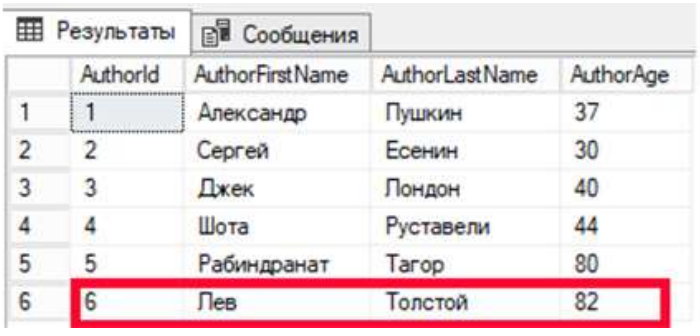
UPDATE – SQL запрос, позволяющий внести изменения или дописывать новую информацию в те записи, которые уже существуют.

Внесем корректировки в шестую запись (**AuthorId = 6**). Значения изменим для полей имени, фамилии и возраста автора.

```
UPDATE tAuthors SET AuthorFirstName = 'Лев', AuthorLastName='Толстой',
AuthorAge = '82' WHERE AuthorId = '6';
```

Затем, обратимся к БД, чтобы вывести все имеющиеся записи:

```
SELECT * FROM tAuthors;
```



Результаты | Сообщения

	AuthorId	AuthorFirstName	AuthorLastName	AuthorAge
1	1	Александр	Пушкин	37
2	2	Сергей	Есенин	30
3	3	Джек	Лондон	40
4	4	Шота	Руставели	44
5	5	Рабиндранат	Тагор	80
6	6	Лев	Толстой	82

Мы видим изменения информации в записи автора под номером 6.

DELETE

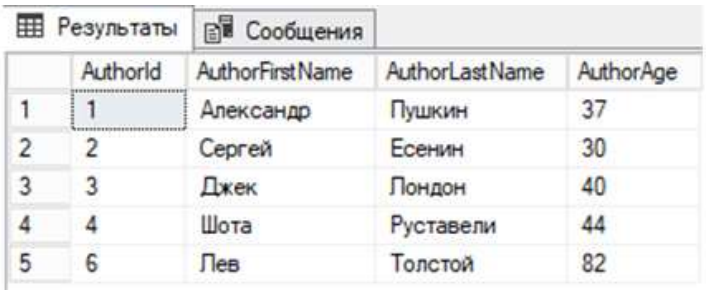
DELETE – SQL запрос, выполняя который в СУБД производится операция удаления определенной строки из таблицы в БД.

Обратимся к «**tAuthors**» с командой на удаление строки, где **AuthorId** = 5:

```
DELETE FROM tAuthors WHERE AuthorId = '5';
```

Чтобы увидеть изменения, снова обратимся к базе для вывода всех записей:

```
SELECT * FROM tAuthors;
```



Результаты | Сообщения

	AuthorId	AuthorFirstName	AuthorLastName	AuthorAge
1	1	Александр	Пушкин	37
2	2	Сергей	Есенин	30
3	3	Джек	Лондон	40
4	4	Шота	Руставели	44
5	6	Лев	Толстой	82

Мы видим, что запись автора под номером 5 теперь отсутствует в «tAuthors» и, соответственно, не выводится с другими записями.

DROP

DROP – ключевое слово в SQL, применяемое для удаления данных с помощью запроса. К примеру удаление некоторой таблицы из БД.

После рассмотрения ряда простых запросов к БД мы можем полностью удалить нашу таблицу «**tAuthors**» целиком, выполнив простой SQL запрос:

```
DROP TABLE tAuthors;
```

Задания:

1) Создать базу данных с помощью ЯП SQL и проделать все команды, которые были указаны в лекционном материале.

2) Предоставить отчёт со скриншотами на почту преподавателя.