

# Тема: Создание базы данных и таблиц БД

---

- 
- Каждая БД как минимум состоит из двух файлов, один для данных, другой для журнала транзакций.
-

---

Create Database <database name>

---

---

## Тип данных Краткое описание

---

<b><u>bigint</u></b>	Целочисленный тип данных, занимающий 8 байт
<b><u>float</u></b>	Нецелочисленный тип данных приблизительной точности
<b><u>ntext</u></b>	Текстовые данные Unicode длиной до 1 <u>Гбайта</u>
<b><u>int</u></b>	Целочисленный тип данных, занимающий 4 байта
<b><u>real</u></b>	Нецелочисленный тип данных приблизительной точности
<b><u>binary</u></b>	Двоичные данные фиксированной длины до 8000 байт
<b><u>smallint</u></b>	Целочисленный тип данных, занимающий 2 байта
<b><u>datetime</u></b>	Дата и время высокой точности (8-байтовый)
<b><u>varbinary</u></b>	Двоичные данные переменной длины до 8000 байт
<b><u>tinyint</u></b>	Целочисленный тип данных, занимающий 1 байт
<b><u>smalldatetime</u></b>	Дата и время низкой точности (4-байтовый)
<b><u>image</u></b>	Двоичные данные длиной до 2 Гбайт
<b><u>bit</u></b>	Один бит, принимает значение либо 0, либо 1
<b><u>char</u></b>	Символьные данные не Unicode фиксированной длины до 8000 символов
<b><u>decimal</u></b>	Нецелочисленный тип данных фиксированной точности

---

---

Тип данных	Краткое описание
<u>varchar</u>	Символьные данные не Unicode переменной длины до 8000 символов
<u>timestamp</u>	Временной штамп или версия строки
<u>numeric</u>	Нецелочисленный тип данных фиксированной точности
<u>text</u>	Текстовые данные не Unicode длиной до 2 Гбайт
<u>sql variant</u>	Тип данных, позволяющий хранить значения других типов данных
<u>money</u>	Денежный тип данных высокой точности (8-байтовый)
<u>nchar</u>	Символьные данные Unicode фиксированной длины до 4000 символов
<u>smallmoney</u>	Денежный тип данных низкой точности (4-байтовый)
<u>nvarchar</u>	Символьные данные Unicode переменной длины до 4000 символов
<u>uniqueidentifier</u>	Тип данных, предназначенный для хранения глобальных уникальных идентификаторов

---

# Создание таблицы:

---

```
CREATE TABLE <имя_таблицы>  
( <определение_столбца>,  
  <определение_столбца>,  
  <определение_столбца> )
```

---

В скобках указываются все предложения данной инструкции, определяющие отдельные элементы таблицы или ограничения целостности.

---

<определение\_столбца> ::=  
<имя\_столбца> <тип\_столбца> [ <ограничение\_столбца> ]

---

---

## Ограничения-

Null

первичный ключ

внешний ключ

уникальный столбец

проверочной ограничение Check

значение по умолчанию

---



с помощью параметра **CONSTRAINT** создается поименованное ограничение, так как модификация или удаление любого ограничения происходит по его имени.

---

Таблица может содержать ограничение **PRIMARY KEY** только для одного столбца или для группы столбцов.

**UNIQUE** может присутствовать любое количество раз. Null-значения допустимы.

---

<Ограничение\_столбца> состоит из:

---

[CONSTRAINT<имя\_ограничения>]-ключевое слово, после которого указывается название ограничения.

{[DEFAULT<выражение>]}- значение по умолчанию для столбца.

|[NULL]|[NOT NULL]- разрешающие или запрещающие null-значения

|[PRIMARY KEY|UNIQUE]- первичный ключ, уникальность значения.

| [FOREIGN KEY...REFERENCES <имя\_таблицы>]

[(*<имя\_столбца>* [,...,*n*])]- описывает внешний ключ.

|[ON DELETE {CASKADE|NO ACTION}]-действия при удалении: 1) удаляют и из зависимой таблицы; 2) игнорируют удаление и сообщение об ошибке.

|[ON UPDATE {CASKADE|NO ACTION}]- действия при изменении аналогичные действиям при удалении.

|[CHECK()]}-ограничение целостности, инициирующее контроль вводимых в столбец значений.

NO ACTION устанавливается по умолчанию.

---

Создать таблицу Студент с первичным ключом Код студента и данными: ФИО и группа.

---

```
CREATE TABLE Студент (код_студента  
    int PRIMARY KEY, фамилия CHAR  
    (15), имя CHAR (10), отчество CHAR  
    (15), группа CHAR (4))
```

---

Создать таблицу Предмет с первичным ключом  
Код предмета и название предмета, которое будет  
уникальным значением атрибута Преподаватель должно  
быть определено для каждого предмета.

---

```
CREATE TABLE предмет (код_предмета  
    INT CONSTRAINT PK_код_предмета  
    PRIMARY KEY (код_предмета),  
    название_предмета CHAR (20)  
    UNIQUE, преподаватель CHAR (30)  
    NOT NULL)
```

---

В следующем примере производится создание новой таблицы со свойством IDENTITY для получения автоматически увеличивающегося идентификационного номера.

- 
- CREATE TABLE new\_employees
  - ( id\_num int IDENTITY(1,1) PRIMARY KEY , fname varchar (20), lname varchar(30) );
-

# Задание

---

- 1 Создать БД со своей фамилией
  - 2 Создать таблицу Студент, указав PRIMARY KEY сразу после ключевого атрибута (вначале)
  - 3 Создать таблицу Студент1, указав PRIMARY KEY (в конце) после всех столбцов
  - 4 Создать таблицу Студент2, указав ограничение CONSTRAINT
  - Для PRIMARY KEY сразу после ключевого атрибута (вначале)
-

- 
- Создать таблицу Оценки с полями Код предмета, Код студента, оценка, дата сдачи. Они должны быть определены, при чем Код студента и Код предмета были внешними ключами, а оценку можно вводить от 2 до 5.
  - CREATE TABLE Оценки (код\_предмета INT CONSTRAINT FK\_KP FOREIGN KEY (код\_предмета) REFERENCES предмет (код\_предмета), код\_студента INT CONSTRAINT FK\_KC FOREIGN KEY (код\_студента) REFERENCES студент (код\_студента), оценка INT (1) CHECK ((оценка >= 2 AND оценка <= 5) дата\_сдачи DATE)
-