

# БАЗЫ ДАННЫХ

---

Лекция 3 Типы данных  
Ограничения целостности

# Пример

- Преподаватели пишут учебники по предметам
- Преподаватель – Предмет – Учебник

# Пример

- **Преподаватель**

- IdTeacher
- Name
- Birthday

- **Предмет**

- IdSubject
- SubjectName
- SubjectType

- **Авторы**

- IdBookAuthor
- **IdTeacher**
- **IdBook**

- **Учебник**

- IdBook
- BookName
- PublicationYear
- NumberOfPages
- **IdSubject**

# Пример

- Студент – Группа – Староста

# Пример

- **Студент**
- IdStudent
- StudentName
- Birthday
- Address
- **IdGroup**

- **Группа**
- IdGroup
- GroupNum
- Profession
- Faculty
- **Head**

# Типы данных

- Числовые
- Денежные
- Символьные
- Дата и время
- Прочие

# Числовые типы данных

- Точные
- Приближенные

# Точные числовые типы данных

Тип данных	Диапазон значений	Количество байт
tinyint	0 – 255	1
smallint	-32768 – 32768	2
int	$-2^{31} - (2^{31}-1)$	4
bigint	$-2^{63} - (2^{63}-1)$	8
bit	0 или 1	1
decimal(p, s) numeric(p, s) $1 \leq p \leq 38,$ $0 \leq s < p$	$(-10^{38}+1) - (10^{38}+1)$	5 – 17  p - максим. количество цифр (точность) s - количество цифр после точки (масштаб)



# Точность и масштаб

- Точность представляет собой количество десятичных знаков в числе
- Масштаб представляет собой количество десятичных знаков справа от десятичного разделителя
- Например:
- число 153,411
- точность 6
- масштаб 3

# Точность и масштаб

Операция	Точность результата	Масштаб результата
$e1 + e2$	$\max(s1, s2) + \max(p1-s1, p2-s2) + 1$	$\max(s1, s2)$
$e1 - e2$	$\max(s1, s2) + \max(p1-s1, p2-s2) + 1$	$\max(s1, s2)$
$e1 * e2$	$p1 + p2 + 1$	$s1 + s2$
$e1 / e2$	$p1 - s1 + s2 + \max(6, s1 + p2 + 1)$	$\max(6, s1 + p2 + 1)$
$e1 \% e2$	$\min(p1-s1, p2 -s2) + \max( s1,s2 )$	$\max(s1, s2)$

# Числовые типы данных

```
use TEMPDB
go
create table EXACT_NUMBER
(
  N_BIT          bit,
  N_TINYINT      tinyint,
  N_SMALLINT     smallint,
  N_INT          int,
  N_BIGINT       bigint,
  N_DECIMAL      decimal(10,3)
);
go
insert into EXACT_NUMBER values (1, 123, 12345, 1234567890)
insert into EXACT_NUMBER values (-1, 123, -12345, -1234567890)
insert into EXACT_NUMBER values (0, 123, 12345, 1234567890)
go
select * from EXACT_NUMBER;
go
```

N_BIT	N_TINYINT	N_SMALLINT	N_INT	N_BIGINT	N_DECIMAL
1	123	12345	1234567890	123456789012345678	1234567.890
1	123	-12345	-1234567890	-123456789012345678	-1234567.123
0	123	12345	1234567890	123456789012345678	1234567.123

# Приближенные числовые типы

## данных

Тип данных	Диапазон значений	Количество байт
float(p) $1 \leq p \leq 53$	$-1.79 \times 10^{308} -$ $-2,23 \times 10^{-308}$ 0, $2,23 \times 10^{-308} - 1.79 \times 10^{308}$	4 ( $p \leq 24$ )  8 ( $p > 24$ )
real float(24)	$-3.4 \times 10^{38} - -1,18 \times 10^{-38},$ 0, $1,18 \times 10^{-38} - 3.4 \times 10^{38}$	4  REAL является синонимом типа FLOAT(24).

# Приближенные числовые типы данных

```
use TEMPDB
```

```
go
```

```
create table APPROXIMATE_NUMBER
```

```
(
```

```
  N_FLOAT      float(53),
```

```
  N_REAL       real
```

```
);
```

```
go
```

```
insert into APPROXIMATE_NUMBER values (1.1,2.2);
```

```
insert into APPROXIMATE_NUMBER values (1.12345678901
```

```
insert into APPROXIMATE_NUMBER values (112.345678901
```

# Денежные типы данных

Тип данных	Диапазон	Хранение
money	От -922 337 203 685 477,5808 до 922 337 203 685 477,5807	8 байт
smallmoney	От -214 748,3648 до 214 748,3647	4 байта

# Символьные типы данных

Тип данных	Размер в символах	Количество байт
char(n)	1 – 8000	n
varchar(n)	1 – 4000	кол. символов + 2
varchar(max)	1 – (2 <sup>31</sup> -1)	кол. символов +2
nchar(n)	1 – 8000	2n
nvarchar(n)	1 – 4000	2 кол. символов + 2
nvarchar(max)	1 – (2 <sup>30</sup> -1)	2×кол. символов +2

# Символьные типы данных

```
use TEMPDB
go

create table CHARACTER_DATA
(
  C_CHAR          char(100),
  C_VARCHAR       varchar(200),
  C_VARCHARMAX    varchar(max),
  C_NCHAR         nchar(100),
  C_NVARCHAR      nvarchar(200),
  C_NVARCHARMAX   nvarchar(max)
);
go
```



# Типы данных для даты и времени

Тип данных	Диапазон, точность, формат	Количество байт
date	01.01.1753 – 31.12.9999, 1 день, YYYYMMDD	3
time(p) $0 \leq p \leq 7$	00:00:00.00000000 – 23:59:59.99999999, 100 нс, hh:mm:ss.nnnnnnnn	3 – 5
smalldatetime	01.01.1900 00:00 – 06.06.2079 23:59, 1 мин, YYYYMMDD hh:mm	4

# Типы данных для даты и времени

Тип данных	Диапазон, точность, формат	Количество байт
datetime	01.01.1753 00:00:00.000 –31.12.9999 23:59:59.999, 0.003 с, YYYYMMDD hh:mm:ss.nnn	8
datetime2(p) ) $0 \leq p \leq 7$	01.01.0001 00:00:00.00000000 –31.12.9999 23:59:59.99999999, 100 нс, YYYYMMDD hh:mm:ss.nnnnnnnn	6 – 8
datetime offset(p) $0 \leq p \leq 7$	01.01.0001.00:00:00:00000000 +00:00 – 31.12.9999.23:59:59:99999999 +23:59, 100 нс, YYYYMMDDhh:mm:ss:nnnnnnnn±hh:mm	8 – 10

# Типы данных для даты и времени

- SELECT

- `CAST('2017-05-08 12:35:29. 1234567 +12:15' AS time(7)) AS 'time',`
- `CAST('2017-05-08 12:35:29. 1234567 +12:15' AS date) AS 'date',`
- `CAST('2017-05-08 12:35:29.123' AS smalldatetime) AS 'smalldatetime',`
- `CAST('2017-05-08 12:35:29.123' AS datetime) AS 'datetime',`
- `CAST('2017-05-08 12:35:29. 1234567 +12:15' AS datetime2(7)) AS 'datetime2',`
- `CAST('2017-05-08 12:35:29.1234567 +12:15' AS datetimeoffset(7)) AS 'datetimeoffset';`

# Типы данных для даты и времени

Тип данных	Вывод
<b>time</b>	12:35:29. 1234567
<b>date</b>	2017-05-08
<b>smalldatetime</b>	2017-05-08 12:35:00
<b>datetime</b>	2017-05-08 12:35:29.123
<b>datetime2</b>	2017-05-08 12:35:29. 1234567
<b>datetimeoffset</b>	2017-05-08 12:35:29.1234567 +12:15

# Функции CAST и CONVERT

- CAST ( expression AS data\_type )
- CONVERT ( data\_type, expression[ , style ] )

Без века (гг)	С веком (гггг)	Стандартная схема	Стиль
-	<b>0</b> или <b>100</b>	default	мес дд гггг чч:мм
<b>1</b>	<b>101</b>	США	1 = мм/дд/гг 101 = мм/дд/гггг
<b>2</b>	<b>102</b>	ANSI	2 = гг.мм.дд 102 = гггг.мм.дд
<b>10</b>	<b>110</b>	США	10 = мм-дд-гг 110 = мм-дд-гггг
<b>12</b>	<b>112</b>	ISO	12 = ггммдд 112 = ггггммдд

# Функции CAST и CONVERT

```
select cast('-123.123' as numeric(7,3)), -- СИМВОЛЫ --> ЧИСЛО
       convert(numeric(7,3), '-123.123')
select cast(-123.123 as varchar(12)), -- ЧИСЛО --> СИМВОЛЫ
       convert(varchar(12), -123.123)
select cast(getdate() as char(30)), -- дата --> СИМВОЛЫ
       convert(char(30), getdate()),
       convert(char(30), getdate(), 112) -- 112 - стиль
select cast('20140107 22:00:01.123' as datetime), -- СИМВОЛЫ --> дата
       convert(datetime, '20140107 22:00:01.123')
```

# Двоичные типы данных

- Хранится последовательность битов
- Применяются для хранения изображений, звука, видео
- Можно хранить любые данные

# Двоичные типы данных

Тип данных	Размер в байтах	Количество байт
<b>binary(n)</b>	1 – 8000	n
<b>varbinary(n)</b>	1 – 8000	КОЛ. СИМВОЛОВ + 2
<b>varbinary(max)</b>	1 – (2 <sup>31</sup> -1)	КОЛ. СИМВОЛОВ +2



# Двоичные типы данных

```
use TEMPDB
go
create table BINARY_DATA
(
  B_BINARY          binary(400),
  B_VARBINARY       varbinary(400),
  B_VARBINARYM      varbinary(max)
);
go
insert into BINARY_DATA
  values(
    0x123456789012345,
    0x123456789012345,
```

# Прочие типы данных

- TIMESTAMP
- UNIQUEIDENTIFIER
- XML
- HIERARCHYID
- GEOGRAPHY, GEOMETRY
- FILESTREAM
- SQLVARIANT
- TEXT, NTEXT, IMAGE

# TIMESTAMP

- **ROWVERSION** - синоним **TIMESTAMP**
- Необходимо установить хронологию изменения данных
- занимает 8 байт
- Значения могут вводиться и изменяться **только сервером**

# TIMESTAMP

```
use TEMPDB
go
create table TIMESTAMP_DATA1
(
  ID      int,
         timestamp      -- имя по умолчанию TIME
);
go
create table TIMESTAMP_DATA2
(
  ID          int,
  T_ROWVERSION rowversion -- имя обязательно
);
go
insert into TIMESTAMP_DATA1 (ID)      values (1);
insert into TIMESTAMP_DATA2 (ID)      values (1);
```

ID	timestamp
1	0x0000000000000007EF

ID	T_ROWVERSION
1	0x0000000000000007F0

# UNIQUEIDENTIFIER

- 16-байтовый идентификатор GUID
- Главная особенность – способность генерировать уникальные значения, которые с очень малой вероятностью могут быть независимо получены еще раз.
- Могут быть получены при помощи встроенной функции **NEWID**.
- Могут быть преобразованы из строковой константы в формате xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx, где каждый символ x представляет шестнадцатеричную цифру в диапазоне 0–9 или a–f
- Пример:
- 8F1719F1-8B37-D821-B52D-00C04FC964FF

# UNIQUEIDENTIFIER

```
use TEMPDB
go
create table ID_DATA
(
  ID    uniqueidentifier,
  V     varchar(10)
);
go

insert into ID_DATA values(newid(), 'ABCD');

select *, datalength(ID) [datalength(ID)] from ID_DATA
```

ID	V	datalength(ID)
0E947E7A-CEDC-4253-8732-4A52A9024CF7	ABCD	16

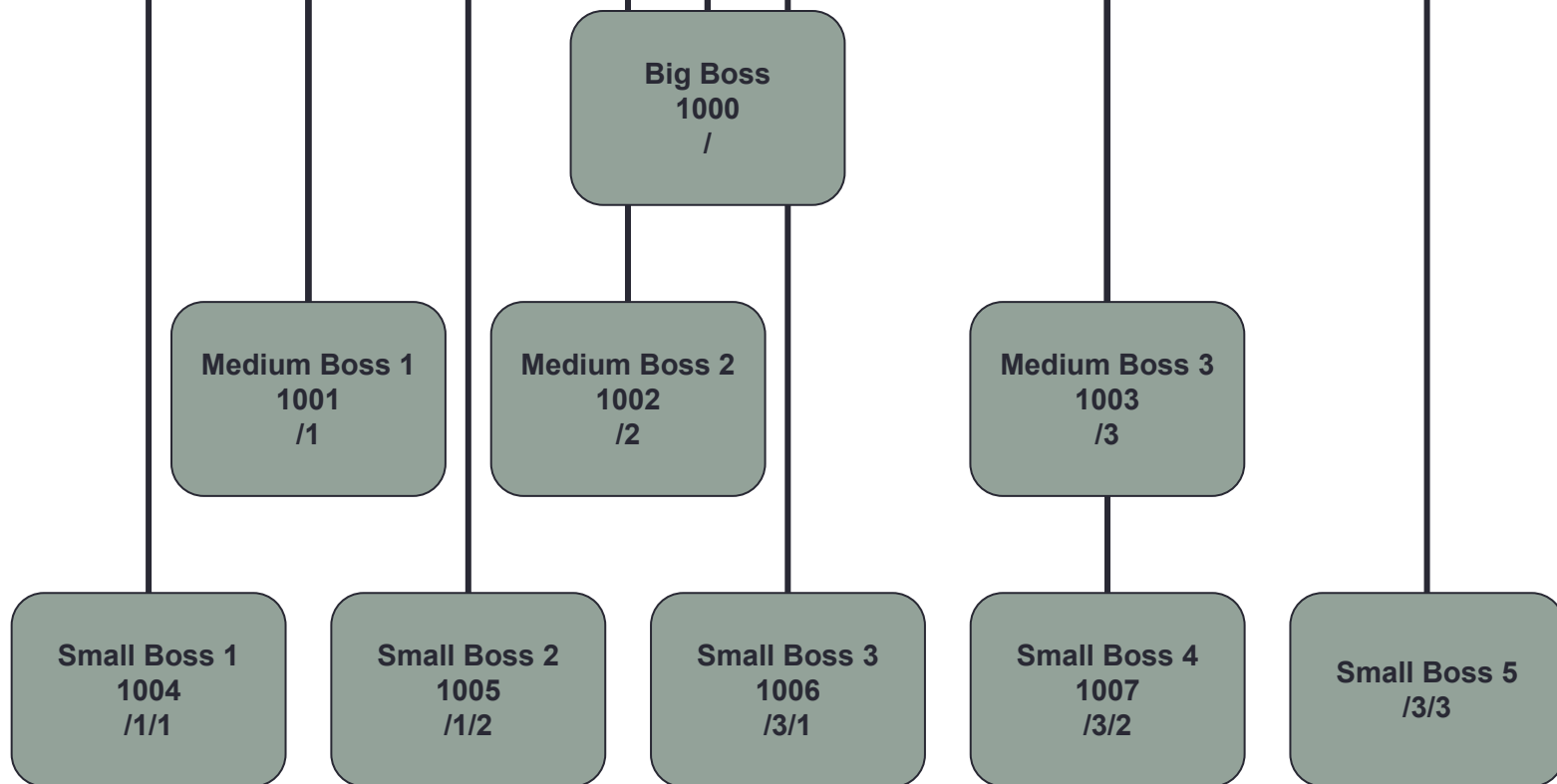
# XML

- Тип данных, в котором хранятся XML-данные

```
<?xml version="1.0"?>
<journal>
  <title>Very Useful Journal</title>
  <contacts>
    <address>sdsds</address>
    <tel>8-3232-121212</tel>
    <tel>8-3232-121212</tel>
    <email>j@j.ru</email>
    <url>www.j.ru</url>
  </contacts>
  <issues-list>
    <issue index="2">
      <title>XML today</title>
      <date>12.09.98</date>
      <about>XML</about>
      <home-url>www.j.ru/issues/</home-url>
      <articles>
        <article ID="3">
          <title>Issue overview</title>
          <url>/article1</url>
          <hotkeys>
            <hotkey>language</hotkey>
            <hotkey>marckup</hotkey> >
```

# HIERARCHYID

- Системный тип данных переменной длины
- Используется для представления положения в иерархии





# Пространственные типы

- GEOGRAPHY
- GEOMETRY
- **geography** хранит эллиптические данные, такие как координаты широты и долготы GPS
- **geometry** представляет данные в эвклидовом пространстве (плоской системе координат)

# FILESTREAM

- FILESTREAM размещает данные больших двоичных объектов (BLOB) типа **varbinary(max)** в файловой системе в виде файлов
- Можно вставлять, обновлять, запрашивать, выполнять поиск и выполнять резервное копирование данных FILESTREAM
- Следует использовать в следующих случаях:
  - средний размер сохраняемых объектов превышает 1 МБ;
  - важен быстрый доступ для чтения;

# SQLVARIANT

- Тип данных, хранящий значения различных типов данных
- Максимальная длина значения типа **sql\_variant** составляет 8016 байт
- Сюда включены структура и значение базового типа
- Максимальная длина значения соответствующего базового типа составляет 8 000 байт

# SQLVARIANT

- Типы значений, которые не могут храниться в типе данных **sql\_variant**

<b>varchar(max)</b>	<b>varbinary(max)</b>
<b>nvarchar(max)</b>	<b>xml</b>
<b>text</b>	<b>ntext</b>
<b>image</b>	<b>rowversion (timestamp)</b>
<b>sql_variant</b>	<b>geography</b>
<b>hierarchyid</b>	<b>geometry</b>
Определяемые пользователем типы	<b>datetimeoffset</b>

# Приоритет

- Тип данных с меньшим приоритетом будет преобразован в тип данных с большим приоритетом
- Если неявное преобразование не поддерживается, возвращается ошибка

# Приоритет

1. определяемые пользователем типы данных (высший приоритет);
2. `sql_variant`;
3. `xml`;
4. `datetimeoffset`;
5. `datetime2`;
6. `datetime`;
7. `smalldatetime`;
8. `date`;
9. `time`;
10. `float`;
11. `real`;
12. `decimal`;
13. `money`;
14. `smallmoney`;
15. `bigint`;
16. `int`;
17. `smallint`;
18. `tinyint`;
19. `bit`;
20. `ntext`;
21. `text`;
22. `image`;
23. `timestamp`;
24. `uniqueidentifier`;
25. `nvarchar` (включая `nvarchar(max)`);
26. `nchar`;
27. `varchar` (включая `varchar(max)`);
28. `char`;
29. `varbinary` (включая `varbinary(max)`);
30. `binary` (низший приоритет).

# Ограничения целостности

Условное обозначение	Действие ограничения целостности
<b>data type</b> тип данных	предотвращает появление в столбце значений, не соответствующих типу данных
<b>not null</b> запрет значений null	предотвращает появление в столбце значений null
<b>default</b> Значение по умолчанию	устанавливает значение в столбце по умолчанию при выполнении операции INSERT
<b>primary key</b> первичный ключ	предотвращает появление в столбце (группе столбцов) повторяющихся значений и пустого значения
<b>foreign key</b> внешний ключ	устанавливает связь между таблицей со столбцом, имеющим свойство foreign key (FK внешний ключ) и таблицей, имеющей столбец со свойством primary key (PK – первичный ключ); предотвращает несогласованные операции между PK и FK
<b>unique</b> уникальное значение	аналогично primary key, но допускает пустые значения и не может быть использован для связи с foreign key
<b>check</b> проверка значений	предотвращает появление в столбце значения, не удовлетворяющего логическому условию, записанному после check

# Ограничения целостности

- Для ограничений целостности
  - PRIMARY KEY
  - FOREIGN KEY
  - UNIQUE
  - CHECK
- может быть задано **имя**
- Если это имя не задано, при создании таблицы сервер назначает ограничениям собственные имена



# PRIMARY KEY

- Столбец или группа столбцов, имеющие уникальные значения для каждой строки, называется **ключом**
- **Create table FACULTY** -- факультет  
( **FACULTY char(10)**, -- идентификатор  
**FACULTY\_NAME varchar(50)** ); -- полное имя

```
insert into FACULTY (FACULTY, FACULTY_NAME )  
values ('ХТиТ', 'Химическая технология и техника');  
insert into FACULTY (FACULTY, FACULTY_NAME )  
values ('ХТиТ', 'Химическая технология и техника');  
select * from FACULTY;
```

FACULTY	FACULTY_NAME
ХТиТ	Химическая технология и техника
ХТиТ	Химическая технология и техника

# PRIMARY KEY

- **Create table FACULTY** --факультет
- **( FACULTY char(10) primary key,** --идентификатор
- **FACULTY\_NAME varchar(50)** --полное имя
- **);**

```
insert into FACULTY (FACULTY, FACULTY_NAME )
    values ('ХТиТ', 'Химическая технология и техника');
insert into FACULTY (FACULTY, FACULTY_NAME )
    values ('ХТиТ', 'Химическая технология и техника');
select * from FACULTY;
```

Msg 2627, Level 14, State 1, Line 4

Violation of PRIMARY KEY constraint 'PK\_FACULTY\_81E78829173876EA'. Cannot insert duplicate key in object 'dbo.FACULTY'.  
The statement has been terminated.

# PRIMARY KEY

**Create table FACULTY**

```
( FACULTY char(10)  
  constraint PK_FACULTY_FACULTY  
  primary key,  
  FACULTY_NAME varchar(50));
```

**Create table FACULTY**

```
( FACULTY char(10),  
  FACULTY_NAME varchar(50),  
  constraint PK_FACULTY_FACULTY  
  primary key (FACULTY));
```

# PRIMARY KEY

- **Create table SHEDULE\_TEACHER** -- расписание преподавателей
- ( **CLASSDATE** smalldatetime, -- дата и время занятий
- **TEACHER** char(10), -- преподаватель
- **SUBJECT** char(10), -- дисциплина
- **AUDITORIUM** char(10), -- аудитория
- **constraint PK\_ S\_ TEACHER primary key**
- **(CLASSDATE, TEACHER) );**

# NOT NULL

```
insert into FACULTY (FACULTY, FACULTY_NAME)
    values ('ИДиП', 'Издательское дело и полиграфия');
insert into FACULTY (FACULTY)
    values ('ХТиТ'); -- неявный ввод значения NULL
insert into FACULTY (FACULTY, FACULTY_NAME)
    values ('ЛХФ', NULL); -- явный ввод значения NULL
```

```
select * from FACULTY;
```

FACULTY	FACULTY_NAME
ИДиП	Издательское дело и полиграфия
ЛХФ	NULL
ХТиТ	NULL

```
update FACULTY set FACULTY_NAME = NULL where FACULTY = 'ИДиП';
select * from FACULTY;
```

# DEFAULT

```
create table FACULTY -- факультеты
(
    FACULTY      char(10),          -- идентификатор факульт
    FACULTY_NAME varchar(50) default '???' -- полное наименование
    constraint FACULTY_PK primary key(FACULTY)
);
go
```

```
insert into FACULTY (FACULTY, FACULTY_NAME)
    values ('ИДиП', 'Издательское дело и полиграфия');
insert into FACULTY (FACULTY)
    values ('ХТиТ'); -- неявный ввод значения default
insert into FACULTY (FACULTY, FACULTY_NAME)
    values ('ЛХФ', NULL); -- явный ввод значения NULL
insert into FACULTY (FACULTY, FACULTY_NAME )
    values ('ТТЛП', NULL); -- явный ввод значения NULL |
insert into FACULTY (FACULTY, FACULTY_NAME)
    values ('ИЭФ', default); -- явный ввод значения default

update FACULTY set FACULTY_NAME = NULL where FACULTY = 'ИДиП';
update FACULTY set FACULTY_NAME = default where FACULTY = 'ЛХФ';

select * from FACULTY;
```

FACULTY	FACULTY_NAME
ИДиП	NULL
ИЭФ	???
ЛХФ	???
ТТЛП	NULL
ХТиТ	???

# DEFAULT

```
create table FACULTY -- факультеты
(
    FACULTY      char(10),                -- идентификатор факультета
    FACULTY_NAME varchar(50) not null default '???' , -- полное наименование
    constraint FACULTY_PK primary key(FACULTY)
);
```

# FOREIGN KEY

- **Внешний ключ** – ограничение целостности, основанное на связи, установленной между двумя таблицами БД
- Виды связей:
  - 1:1 – каждому экземпляру **одной** таблицы соответствует в точности **один** экземпляр второй и **наоборот**
  - 1:n – может существовать экземпляр **одной** таблицы, который соответствует **нескольким** экземплярам другой, и **обратное не допускается**
  - m:n – экземпляр **одной** таблицы соответствует **нескольким** экземплярам другой таблицы и **наоборот**



# FOREIGN KEY

```
create table PULPIT -- кафедра
(
    PULPIT          char(20)          -- идентификатор кафедры
                    constraint PULPIT_PK primary key,
    PULPIT_NAME    varchar(100)      -- полное наименование кафедры
                    default '???' ,
    FACULTY        char(10)          -- идентификатор факультета
                    constraint PULPIT_FACULTY_FK foreign key
                    references FACULTY (FACULTY)
);
```

FACULTY	FACULTY_NAME
ИДиП	Издательское дело и полиграфия
ИЭФ	Инженерно-экономический факультет
ЛХФ	Лесохозяйственный факультет
ТОВ	Технология органических веществ
ТТЛП	Технология и техника лесной промышленности
ХТиТ	Химическая технология и техника

# FOREIGN KEY – INSERT

```
insert into PULPIT (PULPIT, PULPIT_NAME, FACULTY) -- OK
    values ('ИСиТ', 'Информационных систем и технологий ', 'ИДиП' );
insert into PULPIT (PULPIT, PULPIT_NAME, FACULTY) -- OK
    values ('ПОиСОИ', 'Полиграфического оборудования и систем обработки информации ', NULL);
insert into PULPIT (PULPIT, PULPIT_NAME, FACULTY) -- OK
    values ('ЛУ', 'Лесоустройства', 'ЛХФ');
insert into PULPIT (PULPIT, PULPIT_NAME, FACULTY) -- error 547
    values ('ЛВ', 'Лесоводства', 'XXXX');
insert into PULPIT (PULPIT, PULPIT_NAME, FACULTY) -- OK
    values ('ЛВ', 'Лесоводства', 'ЛХФ');
insert into PULPIT (PULPIT, PULPIT_NAME, FACULTY) -- OK
    values ('ТЛ', 'Транспорта леса', NULL);
select * from PULPIT order by FACULTY
```

# FOREIGN KEY – INSERT

```
(1 row(s) affected)
```

```
(1 row(s) affected)
```

```
(1 row(s) affected)
```

```
Msg 547, Level 16, State 0, Line 7
```

```
The INSERT statement conflicted with the FOREIGN KEY constraint "PULPIT_FACULTY_FK".  
The conflict occurred in database "BSTU", table "dbo.FACULTY", column 'FACULTY'.
```

```
The statement has been terminated.
```

```
(1 row(s) affected)
```

```
(1 row(s) affected)
```

```
(5 row(s) affected)
```

PULPIT	PULPIT_NAME	FACULTY
ПОиСОИ	Полиграфического оборудования и систем обработки...	NULL
ТЛ	Транспорта леса	NULL
ИСиТ	Информационных систем и технологий	ИДиП
ЛВ	Лесоводства	ЛХФ
ЛУ	Лесоустройства	ЛХФ

# FOREIGN KEY – UPDATE

```
update PULPIT set FACULTY = 'ИДиП' where PULPIT = 'ПОиСОИ'; -- OK
update PULPIT set FACULTY = 'ЛХФ' where FACULTY is NULL; -- OK
update PULPIT set FACULTY = NULL where PULPIT = 'ИСИТ'; -- OK
update PULPIT set FACULTY = 'XXX' where FACULTY = 'ЛХФ'; -- error 547
```

```
select * from PULPIT order by FACULTY
```

# FOREIGN KEY – UPDATE

```
(1 row(s) affected)
```

```
(1 row(s) affected)
```

```
(1 row(s) affected)
```

Msg 547, Level 16, State 0, Line 4

The UPDATE statement conflicted with the FOREIGN KEY constraint "PULPIT\_FACULTY\_FK".

The conflict occurred in database "BSTU", table "dbo.FACULTY", column 'FACULTY'.

The statement has been terminated.

PULPIT	PULPIT_NAME	FACULTY
ИСиТ	Информационных систем и технологий	NULL
ПОиСОИ	Полиграфического оборудования и систем обработки...	ИДиП
ТЛ	Транспорта леса	ЛХФ
ЛВ	Лесоводства	ЛХФ
ЛУ	Лесоустройства	ЛХФ

# FOREIGN KEY – DELETE

```
select count(*) 'количество строк до DELETE' from PULPIT;  
delete PULPIT; -- удаление всех строк PULPIT  
select count(*) 'количество строк после DELETE' from PULPIT;
```

количество строк до DELETE
5

---

количество строк после DELETE
0

# CHECK

```
-  
create table TEACHER  
(  
  TEACHER      char(10)  constraint TEACHER_PK  primary key,  
  TEACHER_NAME varchar(100),  
  GENDER       char(1)   constraint TEACHER_GENDER_CH check (GENDER in ('M', 'X')),  
  PULPIT       char(20)  constraint TEACHER_PULPIT_FK foreign key  
                    references PULPIT(PULPIT)  
);  
go
```

# CHECK

```
insert into TEACHER (TEACHER, TEACHER_NAME, PULPIT ) -- OK
values ('СМЛВ', 'Смелов Владимир Владиславович', 'ИСИТ');
insert into TEACHER (TEACHER, TEACHER_NAME, PULPIT, GENDER ) -- OK
values ('АКНВУ', 'Акунович Станислав Иванович', 'ИСИТ', 'м');
insert into TEACHER (TEACHER, TEACHER_NAME, PULPIT, GENDER ) -- OK
values ('БРС', 'Брусенцова Татьяна Палладьевна', 'ИСИТ', 'м');
insert into TEACHER (TEACHER, TEACHER_NAME, PULPIT, GENDER ) -- OK
values ('МРС', 'Мороз Елена Станиславовна', 'ИСИТ', 'ж');
insert into TEACHER (TEACHER, TEACHER_NAME, PULPIT, GENDER ) -- error 547
values ('РМНК', 'Романенко Дмитрий Михайлович', 'ИСИТ', 'х');

update TEACHER set GENDER = 'ж' where TEACHER = 'БРС'; -- OK
update TEACHER set GENDER = NULL where TEACHER = 'МРС'; -- OK
update TEACHER set GENDER = 'y' where TEACHER = 'СМЛВ'; -- error 547

select * from TEACHER
```



# CHECK

```
create table TEACHER
(
  TEACHER          char(10)  constraint TEACHER_PK  primary key,
  TEACHER_NAME    varchar(100),
  GENDER          char(1)
                 not null
                 default 'M'
                 constraint TEACHER_GENDER_CH check (GENDER in ('M', 'Ж')),
  PULPIT          char(20)  constraint TEACHER_PULPIT_FK foreign key
                 references PULPIT(PULPIT)
);
```

# UNIQUE

```
create table PROFESSION -- специальность
(
  PROFESSION      char(20)      -- специальность
                  constraint PROFESSION_PK primary key,
  FACULTY         char(10)      -- факультет
                  constraint PROFESSION_FACULTY_FK
                  foreign key references FACULTY(FACULTY),
  PROFESSION_NAME varchar(100) -- наименование специальности
                  constraint PROFESSION_NAME_UQ unique,
  QUALIFICATION   varchar(50)   -- квалификация
);
go
```

# UNIQUE

```
insert into PROFESSION(FACULTY, PROFESSION,PROFESSION_NAME, QUALIFICATION) --OK
  values ('ИдиП','1-40 01 02', 'Информационные системы и технологии',
         'инженер-программист-системотехник' );
insert into PROFESSION(FACULTY, PROFESSION,PROFESSION_NAME, QUALIFICATION) --OK
  values ('ИдиП','1-47 01 01', 'Издательское дело', 'редактор-технолог' );
insert into PROFESSION(FACULTY, PROFESSION,PROFESSION_NAME, QUALIFICATION) -- error 2627
  values ('ИдиП', '1-36 06 01', 'Издательское дело', 'инженер-электромеханик' );
insert into PROFESSION(FACULTY, PROFESSION,PROFESSION_NAME, QUALIFICATION) --OK
  values ('ХТиТ', '1-36 01 08', NULL, 'инженер-механик' );
insert into PROFESSION(FACULTY, PROFESSION,PROFESSION_NAME, QUALIFICATION) -- error 2627
  values ('ХТиТ', '1-36 07 01', NULL, 'инженер-механик' );
update PROFESSION set PROFESSION_NAME = 'Информационные системы и технологии' -- error 26
  where PROFESSION = '1-36 01 08';
select * from PROFESSION;
```

# IDENTITY

```
create table GROUPS -- учебные группы
(
  IDGROUP      int          -- идентификатор учебной группы
                identity(1,1)
                constraint GROUP_PK primary key,
  FACULTY      char(10)     -- факультет
                constraint GROUPS_FACULTY_FK foreign key
                references FACULTY(FACULTY),
  PROFESSION   char(20)     -- специальность
                constraint GROUPS_PROFESSION_FK foreign key
                references PROFESSION(PROFESSION),
  YEAR_FIRST   smallint     -- год поступления
                check (YEAR_FIRST<=YEAR(GETDATE()))),
);
go
```

# IDENTITY

```
insert into GROUPS (FACULTY, PROFESSION, YEAR_FIRST) -- OK
  values ('ИдиП', '1-40 01 02', 2013),
         ('ИдиП', '1-40 01 02', 2012),
         ('ИдиП', '1-40 01 02', 2011),
         ('ИдиП', '1-40 01 02', 2010),
         ('ИдиП', '1-47 01 01', 2013);

insert into GROUPS (IDGROUP, FACULTY, PROFESSION, YEAR_FIRST) -- error 544
  values (99, 'ИдиП', '1-47 01 01', 2013);

select * from GROUPS;
```

Вопросы?