

ПМ.02 Разработка и администрирование баз данных  
МДК.02.02 Технология разработки и защиты баз данных  
Тема 2.1 Проектирование и реализация баз данных

## Лекция 11

# Команды DDL и DML SQL

Говоров А.И., Говорова М.  
М.

Университет ИТМО  
2016-2017

# Создание таблиц

Формат:

```
CREATE TABLE <имя_таблицы>
```

```
(<определение_столбца> | [ограничение_таблицы  
][, ...])
```

# Определение столбца

Формат:

определение\_столбца::=

имя\_столбца имя\_домена | тип данных [(размер)]  
[ограничение\_столбца...]

Ограничения столбца:

- NULL | NOT NULL
- PRIMARY KEY
- UNIQUE
- CHECK
- FOREIGN KEY

# Ограничения целостности

- Контроль возможности использовать NULL-значения
- Ограничения для доменов атрибутов
- Категорная целостность
- Ссылочная целостность
- Ограничения предметной области

# Ограничения на объекты реляционной базы данных

№пп	Ограничение	Описание
1	<b>CHECK</b>	гарантирует, что значения находятся в границах специфицированного интервала, задаваемого предикатом
2	<b>DEFAULT</b>	помещает значение по умолчанию в столбец. Гарантирует, что столбец всегда имеет значение
3	<b>FOREIN KEY</b>	гарантирует, что значения существуют как значение в столбце первичного ключа другой таблицы. Обеспечивает процедуры удаления дочерних строк при удалении связанных с ней родительских
4	<b>NOT NULL</b>	гарантирует, что столбце всегда содержит значение
5	<b>PRIMARY KEY</b>	гарантирует, что столбец всегда содержит значение и оно уникально в таблице
6	<b>UNIQUE</b>	гарантирует, что значение будет уникальным в таблице

# Создание таблиц: примеры (1)

## Пример 1:

```
CREATE TABLE stud (name VARCHAR(27) PRIMARY KEY, country  
    VARCHAR(10) NOT NULL);
```

\*создание таблицы stud с 2-я полями символьного типа и главным индексом name

## Пример 2:

```
CREATE TABLE s1 (fio VARCHAR(50) NOT NULL,  
    discipline VARCHAR(50) NOT NULL,  
    rating SMALLINT NOT NULL,  
    PRIMARY KEY(fio, discipline));
```

\*создание таблицы s1 с 3-я полями и составным главным индексом (fio, discipline)

# Создание таблиц: примеры (2)

## Пример 1:

```
CREATE TABLE Orders (onum integer NOT NULL PRIMARY KEY, amt decimal, odate  
    date NOT NULL, cnum integer NOT NULL, snum integer NOT NULL,  
    UNIQUE (snum, cnum));
```

## Пример 2:

```
CREATE TABLE Salespeople (snum integer NOT NULL PRIMARY KEY, sname char(15)  
    CHECK (sname BETWEEN 'AA' AND 'MZ'), city char(15), comm decimal NOT  
    NULL DEFAULT = 0.1);
```

## Пример 3:

```
CREATE TABLE Orders (onum integer NOT NULL, amt decimal, odate date, cnum  
    integer NOT NULL, snum integer NOT NULL, CHECK ((cnum > snum) AND (onum  
    > cnum)));
```

# Базовое определение оператора CREATE TABLE

```
CREATE TABLE имя_таблицы
({имя_столбца тип_данных [NOT NULL] [UNIQUE]
  [DEFAULT значение_по_умолчанию]
  [CHECK (условие_проверки_на_допустимость)]
[,...]}
[PRIMARY KEY (список_столбцов),]
{[UNIQUE (список_столбцов), [...]}
{[FORING KEY (список_столбцов_внешних_ключей)
  REFERENCES имя_родительской_таблицы
  [(список_столбцов_ключей-кандидатов)]]}
{[MATCH PARTIAL | FULL ]
[ON UPDATE правило_ссылочной_целостности]
[ON DELETE правило_ссылочной_целостности] [...]}
{[CHECK (условие_проверки_на_допустимость)]}
```



# Создание таблиц: примеры (3)

## Пример 1:

```
CREATE TABLE Customers ( cnum integer NOT NULL PRIMARY KEY cname char(10), city char(10), snum integer, FOREIGN KEY (snum) REFERENCES Salespeople (snum));
```

\*Создание таблицы Заказчиков с полем snum определенным в качестве внешнего ключа ссылающегося на таблицу Продавцов.

## Пример 2:

```
CREATE TABLE Customers ( cnum integer NOT NULL PRIMARY KEY, cname char(10), city char(10), snum integer REFERENCES Salespeople (snum));
```

\*Вариант ограничения столбца ограничением FOREIGN KEY по-другому называется- ссылочное ограничение (REFERENCES), так как он фактически не содержит в себе слов FOREIGN KEY, а просто использует слово REFERENCES, и далее им родительского ключа. Customers.snum как внешний ключ, у которого родительский ключ - это Salespeople.snum. Это эквивалентно такому ограничению таблицы: FOREIGN KEY (snum) REFERENCES Salespeople (snum).

## Пример 3:

```
CREATE TABLE Cityorders (onum integer NOT NULL PRIMARY KEY, amt decimal, cnum integer, snum integer, city char (15), FOREIGN KEY (onum, amt, snum) REFERENCES Orders (onum, amt, snum), FOREIGN KEY (cnum, city) REFERENCES Customers (cnum, city) );
```

# Значения квалификаторов для ON UPDATE и ON DELETE

- **CASCADE** - сначала удаляется (изменяется) заданная строка в *родительской таблице*, а затем удаляются зависимые от нее строки;
- **RISTRICT** - строка может быть удалена, если никакие другие строки не зависят от нее, в противном случае удаления не происходит;
- **SET NULL** - для любого удаляемого первичного ключа соответствующее ему значение внешнего ключа дочерней строки принимает нуль-значение (внешний ключ не может быть задан как NOT NULL);
- **SET DEFAULT** - сначала удаляется (изменяется) заданная строка в *родительской таблице*, а затем внешним ключам зависимых от нее строк дочерней таблицы присваиваются значения по умолчанию (внешний ключ должен быть задан как DEFAULT с указанием значения по умолчанию);
- **NO ACTION** - удаление (изменение) заданной строки в *родительской таблице* отменяется: используется по

# Создание таблиц: примеры (4)

## Пример:

```
CREATE TABLE management
```

```
(managno INTEGER NOT NULL, empno INTeger, job  
INTEGER , PRIMARY KEY (MANAGNO),
```

```
FOREIGN KEY fnkey (EMPNO)
```

```
REFERENCES EMPLOYEE
```

```
ON DELETE CASCADE);
```

# Индексы

## Создание индекса

Формат:

```
CREATE [UNIQUE] INDEX имя_индекса  
ON имя_таблицы (имя_столбца1 [ASC] | [DESC] [,  
имя_столбца2]...);
```

Пример:

```
CREATE UNIQUE INDEX ndxmng ON MANAGEMENT(MANAGNO);
```

## Удаление индекса

Формат:

```
DROP INDEX имя_индекса;
```

Пример:

```
DROP INDEX ndxmng ;
```

# Удаление таблиц

Формат:

```
DROP TABLE <имя_таблицы>  
[RISTRICТ | CASCADE]
```

Пример:

```
DROP TABLE s1;
```

# Вставка данных

Формат:

```
INSERT INTO имя_таблицы [(поле_1[, поле_2[,  
...]])]  
VALUES (значение_1[, значение_2[, ...]]);
```

# Вставка данных: примеры (1)

## Пример 1:

```
INSERT INTO Salespeople VALUES (1001, 'Peel', 'London', 0.12);
```

## Пример 2:

```
INSERT INTO Salespeople VALUES (1001, 'Peel', NULL, 0.12);
```

## Пример 3:

```
INSERT INTO Customers (city, cname, cnum) VALUES ('London', 'Honman',  
2001);
```

## Пример 4:

```
INSERT INTO Salespeople (city, cname, comm, cnum) VALUES ('San Jose',  
'Blanco', NULL, 1100);
```

# Вставка данных: использование результата запроса

Формат:

```
INSERT INTO имя_таблицы [(поле_1[, поле_2[, ...]])]  
SELECT ...);
```

Правила:

- Таблица должна уже быть создана командой CREATE TABLE.
- Таблица-результат\_запроса должна иметь структуру, которая совпадает со структурой таблицы.



# Вставка данных: примеры (2)

## Пример 1:

INSERT INTO Londonstaff

SELECT \* FROM Salespeople WHERE city = 'London';

## Пример 2:

INSERT INTO Multicust

SELECT \* FROM Salespeople WHERE 1 < (SELECT COUNT (\*) FROM Customers WHERE Customers.snum = Salespeople.snum);

## Пример 3:

INSERT INTO Daytotals (date, total)

SELECT odate, SUM (amt) FROM Orders GROUP BY odate;

# Обновление записей

Формат:

UPDATE имя\_таблицы

SET имя\_столбца1=новое значение\_1[,  
имя\_столбца2=новое значение\_2[, ...]]

WHERE условие\_отбора;

# Редактирование данных: примеры

## Пример 1:

```
UPDATE Customers SET rating = 200;
```

## Пример 2:

```
UPDATE Customers SET rating = 200 WHERE snum = 1001;
```

## Пример 3:

```
UPDATE Salespeople SET sname = 'Gibson',city = 'Boston',comm =  
0.10 WHERE snum = 1004;
```

## Пример 4:

```
UPDATE Salespeople SET comm = comm * 2;
```

## Пример 5:

```
UPDATE customers SET rating = NULL WHERE city = 'London';
```

# Удаление записей

Формат:

```
DELETE FROM имя_таблицы  
[WHERE условие_отбора];
```

# Удаление данных: примеры (1)

## Пример 1:

```
DELETE FROM Salespeople;
```

## Пример 2:

```
DELETE FROM Salespeople WHERE snum = 1003;
```

## Пример 3:

```
DELETE FROM Salespeople WHERE city = 'London';
```

## Пример 4:

```
DELETE FROM Customers WHERE NOT EXISTS (SELECT * FROM  
Orders WHERE cnum = Customers.cnum);
```

# Практическое задание

1. Создать БД (без определения связи между таблицами), с использованием DB Browser for SQLite (<http://sqlitebrowser.org/>);.
2. Заполнить БД рабочими данными.
3. Выполнить (в браузере и вручную):
  - вставку полной и неполной записи в таблицу;
  - редактирование нескольких полей записи;
  - удаление записей.

# SQLite

1. Скачать SQLite-браузер  
<http://sqlitebrowser.org/>.
2. Неполное Руководство по SQLite для пользователей Windows. Перевод: А.Г. Пискунов. 21 августа 2014г.  
<http://agp1.hx0.ru/.SQLite.Allow.pdf>
3. Вводное руководство в SQLite (sql database sqlite php)  
[http://www.opennet.ru/base/dev/sqlite\\_guide.txt.html](http://www.opennet.ru/base/dev/sqlite_guide.txt.html)
4. Типы данных в SQLite версии 3  
<http://xbb.uz/db/Tipy-dannyh-v-SQLite-versii-3>

# Список источников

1. [6], с.292-303
2. [http://www.sql.ru/docs/sql/u\\_sql/](http://www.sql.ru/docs/sql/u_sql/) - Martin Gruber.  
Understanding SQL (главы 15-19)
3. <http://www.intuit.ru/studies/courses/1095/191/info> - ОСНОВЫ проектирования реляционных баз данных («Интуит») :  
<http://www.intuit.ru/studies/courses/1095/191/lecture/4981> -  
Лекция 8: Введение в структурированный язык запросов – SQL  
<http://www.intuit.ru/studies/courses/1095/191/lecture/4983> -  
Лекция 9: Создание объектов для хранения данных. Работа с ограничениями
4. [http://www.mstu.edu.ru/study/materials/zelenkov/ch\\_4\\_6\\_1.html](http://www.mstu.edu.ru/study/materials/zelenkov/ch_4_6_1.html)  
(ch\_4\_6\_2, ch\_4\_6\_3) – Язык SQL. DDL: Операторы создания  
схемы базы данных. DML: Команды модификации данных



**СПАСИБО ЗА  
ВНИМАНИЕ**