

АЛТАЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ИНСТИТУТ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ  
Кафедра информатики

# **Основы языка SQL**

## **Создание, модификация и удаление объектов баз данных**

Михеева Т.В., к.т.н.,  
доцент кафедры информатики

# SQL – Structured Query Language

- SQL – это структурированный язык запросов к реляционным базам данных (БД).
- SQL – декларативный язык, основанный на операциях реляционной алгебры.
- Стандарты SQL, определённые Американским национальным институтом стандартов (ANSI):
  - ✓ SQL-1 (SQL/89) – первый вариант стандарта.
  - ✓ **SQL-2 (SQL/92) – основной расширенный стандарт.**
  - ✓ SQL-3 (SQL:1999) – относится к объектно-реляционной модели данных.
  - ✓ SQL:2003, SQL:2008, SQL:2011, SQL:2016
- Подмножества языка SQL:
  - ✓ **DDL (Data Definition Language)** – команды создания/изменения/удаления объектов базы данных (*create/alter/drop*);
  - ✓ **DML (Data Manipulation Language)** – команды добавления/модификации/удаления данных (*insert/update/delete*), а также команда извлечения данных *select*;
  - ✓ **DCL (Data Control Language)** – команды управления данными (установка/снятие ограничений целостности). Входит в подмножество DDL.

## Преимущества языка SQL

- стандартность;
- независимость от конкретных СУБД;
- возможность переноса с одной вычислительной системы на другую;
- реляционная основа языка;
- возможность создания интерактивных запросов;
- возможность программного доступа к БД;
- обеспечение различного представления данных;
- возможность динамического изменения и расширения структуры БД;
- поддержка архитектуры клиент-сервер.

## Диалекты SQL

Несмотря на наличие международного стандарта, многие компании, занимающиеся разработкой СУБД (например, Oracle, Microsoft, MySQL, PostgresPro), вносят изменения в язык SQL, применяемый в разрабатываемой СУБД. Таким образом, появляются специфичные для каждой конкретной СУБД диалекты языка SQL.

Каждая СУБД имеет свой собственный “диалект” SQL, включающий, кроме основ SQL, команды управления (циклы, условия), функции и прочие средства:

- ORACLE – PL/SQL (Procedural Language/SQL)
- MS SQL Server – Transact SQL
- MySQL – SQL/PSM (SQL/Persistent Stored Module)
- PostgreSQL, Postgres, Postgres Pro – PostgreSQL
- и т.д.

## Типы команд SQL

- Реализация в SQL концепции операций, ориентированных на табличное представление данных, позволила создать компактный язык с небольшим набором предложений. Язык SQL может использоваться как для выполнения запросов к данным, так и для построения прикладных программ.
- Основные категории команд языка SQL предназначены для выполнения различных функций, включая построение объектов базы данных и манипулирование ими, начальную загрузку данных в таблицы, обновление и удаление существующей информации, выполнение запросов к базе данных, управление доступом к ней и ее общее администрирование.

## Типы команд SQL

Основные категории команд языка SQL:

- DDL – язык определения данных;
- DML – язык манипулирования данными;
- DQL – язык запросов ;
- DCL – язык управления данными;
- команды администрирования данных;
- команды управления транзакциями.

## Основные объекты баз данных

- Таблицы представляют собой совокупность каких-либо сведений об *объектах*, явлениях, процессах реального мира. Никакие другие *объекты* не хранят *данные*, но они могут обращаться к *данным* в таблице.
- Представлениями (просмотрами) называют виртуальные таблицы, содержимое которых определяется запросом
- Хранимые процедуры представляют собой группу команд SQL, объединенных в один модуль. Такая группа команд компилируется и выполняется как единое целое
- Триггерами называется специальный класс хранимых процедур, автоматически запускаемых при добавлении, изменении или удалении *данных* из таблицы

## Основные объекты баз данных

- **Функции** в языках программирования – это конструкции, содержащие часто исполняемый код. Функция выполняет какие-либо действия над *данными* и возвращает некоторое значение.
- **Индекс** – структура, связанная с таблицей или представлением и предназначенная для ускорения поиска информации в них.
- **Пользовательские типы данных** – это *типы данных*, которые создает пользователь на основе системных *типов данных*, когда в нескольких таблицах необходимо хранить однотипные значения; причем нужно гарантировать, что столбцы в таблице будут иметь одинаковый размер, *тип данных* и чувствительность к значениям NULL .
- **Ограничения целостности** – механизм, обеспечивающий автоматический контроль соответствия *данных* установленным условиям (или ограничениям).



## Определение структур базы данных (DDL)

- Язык определения данных (Data Definition Language, DDL) позволяет создавать и изменять структуру объектов базы данных, например, создавать и удалять таблицы. Основными командами языка DDL являются следующие:

**CREATE**

**ALTER**

**DROP**

# Определение структур базы данных (DDL)

- ❑ CREATE TABLE - создать таблицу
- ❑ ALTER TABLE - изменить таблицу
- ❑ DROP TABLE - удалить таблицу
- ❑ CREATE DOMAIN - создать домен
- ❑ ALTER DOMAIN - изменить домен
- ❑ DROP DOMAIN - удалить домен
- ❑ CREATE COLLATION - создать последовательность
- ❑ DROP COLLATION - удалить последовательность
- ❑ CREATE VIEW - создать представление
- ❑ DROP VIEW - удалить представление
- ❑ CREATE INDEX – создать индекс
- ❑ ALTER INDEX - изменить индекс
- ❑ DROP INDEX - удалить индекс

# Манипулирование данными (DML)

- Язык манипулирования данными (Data Manipulation Language, DML) используется для манипулирования информацией внутри объектов реляционной базы данных посредством трех основных команд:

**INSERT**

**UPDATE**

**DELETE**

INSERT – добавить строки в таблицу.

UPDATE – изменить строки в таблице.

DELETE – удалить строки в таблице.

## Выборка данных (DQL)

- Язык запросов DQL (Data Query Language) наиболее известен пользователям реляционной базы данных, несмотря на то, что он включает всего одну команду SELECT.
- Эта команда вместе со своими многочисленными опциями и предложениями используется для формирования запросов к реляционной базе данных.

## Язык управления данными (DCL)

- Команды управления данными позволяют управлять доступом к информации, находящейся внутри базы данных. Как правило, они используются для создания объектов, связанных с доступом к данным, а также служат для контроля над распределением привилегий между пользователями. Команды управления данными: GRANT, REVOKE.

GRANT - предоставить привилегии пользователю или приложению на манипулирование объектами.

REVOKE - отменить привилегии пользователя или приложения.

## Команды администрирования данных

- С помощью команд администрирования данных пользователь осуществляет контроль за выполняемыми действиями и анализирует операции базы данных
- Они также могут оказаться полезными при анализе производительности системы.
- Не следует путать администрирование данных с администрированием базы данных, которое представляет собой общее управление базой данных и подразумевает использование команд всех уровней.

## Команды управления транзакциями

- Существуют следующие команды, позволяющие управлять транзакциями базы данных: COMMIT, ROLLBACK, SAVEPOINT, SET TRANSACTION.

COMMIT – зафиксировать внесенные изменения.

ROLLBACK – откатить внесенные изменения.

SAVEPOINT – установить точку сохранения текущей транзакции.

SET TRANSACTION – установить характеристики текущей транзакции.

## Идентификаторы языка SQL

Стандарт SQL задает набор символов, который используется по умолчанию, – он включает строчные и прописные буквы латинского алфавита ( A-Z, a-z ), цифры ( 0-9 ) и символ подчеркивания ( \_ ). На формат идентификатора накладываются следующие ограничения:

- идентификатор может иметь длину до 128 символов;
- идентификатор должен начинаться с буквы;
- идентификатор не может содержать пробелы.

$$\langle \text{идентификатор} \rangle = \langle \text{буква} \rangle \{ \langle \text{буква} \rangle | \langle \text{цифра} \rangle \} [ , \dots n ]$$



## Типы данных языка SQL, определенные стандартом

Тип данных	Объявления
Символьный	CHAR   VARCHAR
Битовый	BIT   BIT VARYING
Точные числа	NUMERIC   DECIMAL   INTEGER   SMALLINT
Округленные числа	FLOAT   REAL   DOUBLE PRECISION
Дата/время	DATE   TIME   DATETIME
Интервал	INTERVAL

CHAR(size) - строка символов фиксированной длины размером size символов;

VARCHAR(size) - строка символов переменной длины максимальным размером до size символов;

# Арифметические операции

- Унарные: +, -
- Бинарные: +, -, \*, /
- Операции над строками: Сцепление строк ||
- Операции сравнения: = != <> ^= < > <= >=  
операнд BETWEEN нач\_значение AND кон\_значение  
операнд IN (список выражений | подзапрос)  
операнд NOT IN (список выражений | подзапрос)

# Арифметические операции

## □ Операции сравнения

операнд LIKE шаблон % \_

операнд IS [NOT] NULL

операция сравнения с квантором ANY

операция сравнения с квантором ALL

операция сравнения EXISTS

## □ Логические операции: NOT, AND, OR

## □ Операции над множествами: UNION ALL, UNION, INTERSECT, MINUS

## Особенности синтаксиса

- ✓ В командах SQL не различаются прописные и строчные буквы (кроме содержимого символьных строк)
- ✓ Каждая команда может занимать несколько строк и заканчивается символом ';'
- ✓ Символ и символьная строка заключаются в одинарные кавычки: 'A', '2', 'строка', 'другая строка'
- ✓ Однострочный комментарий начинается с символов '--'
- ✓ Многострочный (блочный) комментарий заключается в символы /\*...\*/

## Особенности синтаксиса

- ✓ квадратные скобки [] – применяются для выделения элементов массива;
- ✓ фигурные скобки {} – конструкции, заключенные в эти скобки, должны рассматриваться как целые синтаксические единицы;
- ✓ круглые скобки () – применяются для группировки выражений и повышения приоритета операций.

## Особенности синтаксиса

- ✓ точка с запятой (;) – завершающий элемент предложений SQL;
- ✓ запятая (,) – используется для разделения элементов списков;
- ✓ точка (.) – используется в числовых константах, а также для отделения имен таблицы и столбца.
- ✓ пробелы – могут вводиться для повышения наглядности между любыми синтаксическими конструкциями предложений SQL;
- ✓ звездочка (\*) - для обозначения "все" (все поля строки или составное значение) или может использоваться как аргумент некоторых агрегатных функций.

## Создание таблицы

□ **CREATE TABLE** имя\_таблицы (имя\_столбца тип\_данных [ограничения целостности] [ , ...n]);

```
CREATE TABLE Products
(
  prod_id  char(10),
  prod_name char(255),
  prod_price decimal(8,2),
  prod_desc text
);
```

## Ограничения целостности

- ✓ уникальность (значений атрибута или комбинации значений атрибутов):

**UNIQUE** (*имя\_атрибута1* [, *имя\_атрибута2*,...])

- ✓ обязательность / необязательность:

**NOT NULL / NULL**

- ✓ первичный ключ:

**PRIMARY KEY**(*имя\_атрибута1* [, *имя\_атрибута2*,...])

- ✓ внешний ключ:

**FOREIGN KEY**(*имя\_атрибута1* [, *имя\_атрибута2*,...]) **REFERENCES**  
*имя\_таблицы* [(*имя\_атрибута1* [, *имя\_атрибута2*,...])]

- ✓ условие на значение поля:

**CHECK** (*условие*)

Например: `check (salary >= 4500), check (date2 > date1)`



# Ограничения поля

[CONSTRAINT <имя\_ограничения>] тип\_ограничения

## □ PRIMARY KEY

[CONSTRAINT <имя\_ограничения>] PRIMARY KEY (имя\_поля [,<имя\_поля>...])

## □ UNIQUE

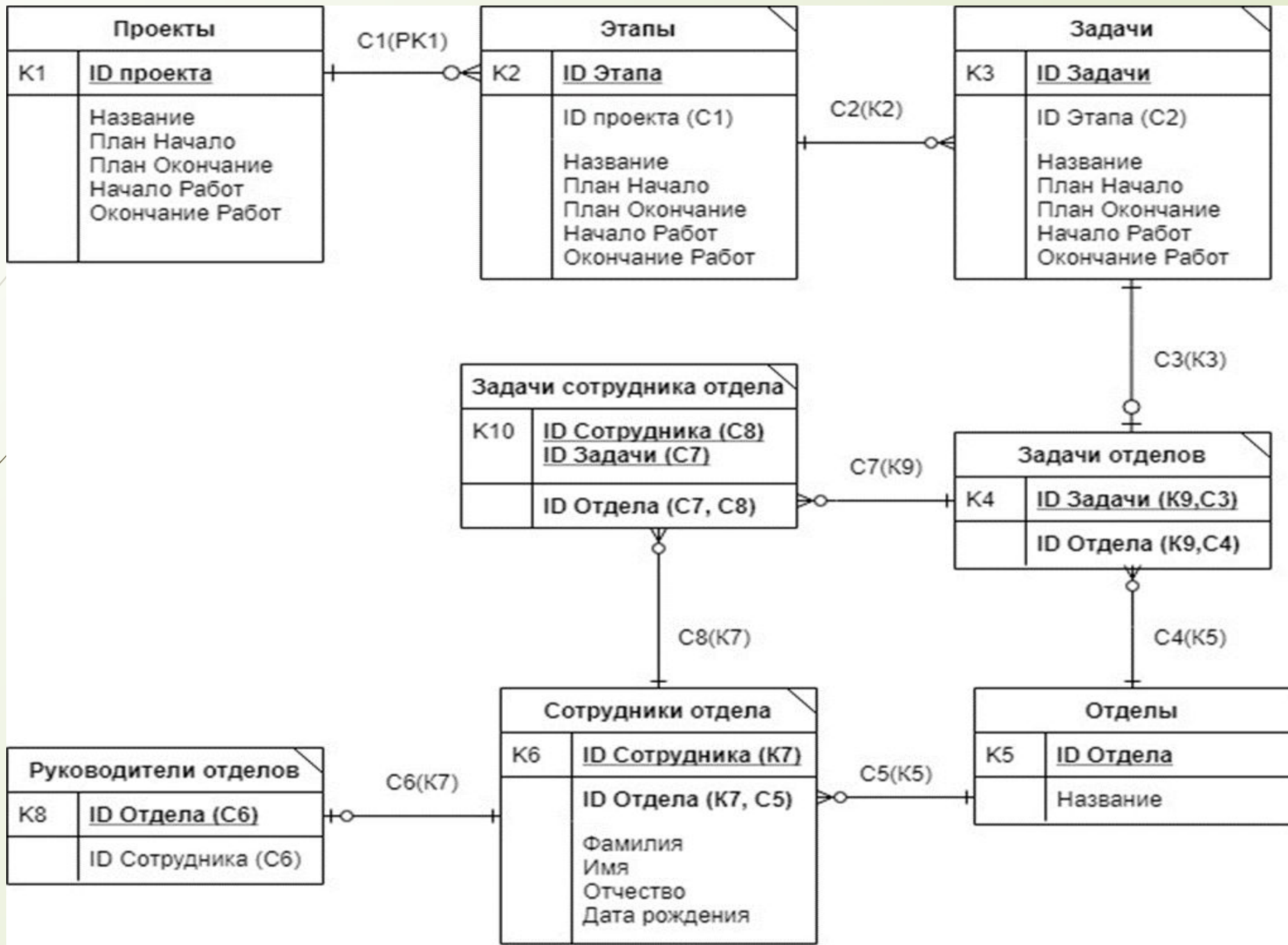
[CONSTRAINT <имя\_ограничения>] UNIQUE (имя\_поля [,<имя\_поля>...])

## □ CHECK (условие)

## □ REFERENCES

[CONSTRAINT <имя\_ограничения>] FOREIGN KEY (имя\_поля [,<имя\_поля>...])

REFERENCES имя\_таблицы (имя\_поля [,<имя\_поля>...])



Отделы	
K5	ID Отдела
	Название

### Отделы

Атрибут	Тип	Ключ	Описание
ID_отдела	Числовой, счетчик	K5	Идентификатор отдела
Название	Текст (30)		Название отдела

```

CREATE TABLE Отделы
(
  ID_Отдела SERIAL NOT NULL,
  Название VARCHAR(30) NOT NULL,
  CONSTRAINT K5 PRIMARY KEY (ID_Отдела)
);

```

```

CREATE TABLE "O

```

## Сотрудники

Атрибут	Тип	Ключ	Описание
ID_сотрудника	Числовой, счетчик	К6, К7	Идентификатор сотрудника
Фамилия	Текст (30)		Фамилия сотрудника
Имя	Текст (30)		Имя сотрудника
Отчество	Текст (30)		Отчество сотрудника
Дата рождения	Дата		Дата рождения сотрудника
ID_отдела	Числовой	К7, С5	Отдел, в котором работает

```
CREATE TABLE Сотрудники_отдела
```

```
(
```

```
ID_Сотрудника SERIAL NOT NULL,
```

```
Фамилия VARCHAR(30) NOT NULL,
```

```
Имя VARCHAR(30) NOT NULL,
```

```
Отчество VARCHAR(30) NULL,
```

```
Дата_рождения DATE NOT NULL,
```

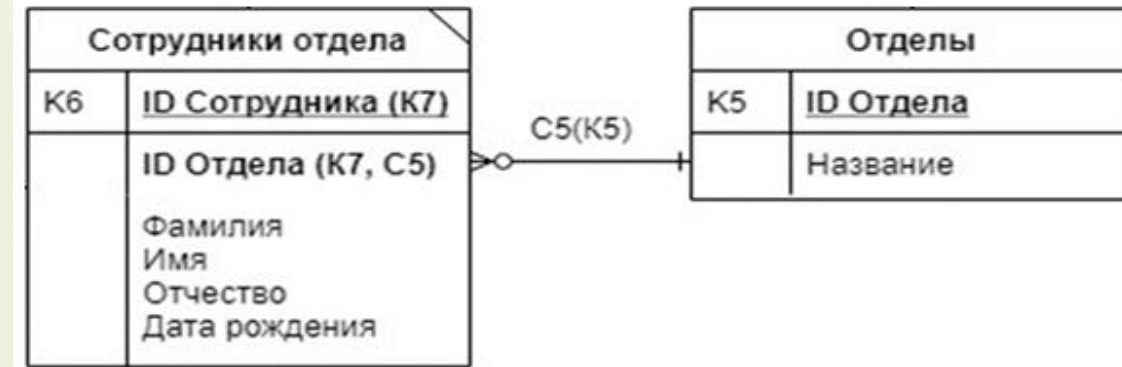
```
ID_Отдела INTEGER NOT NULL,
```

```
CONSTRAINT K6 PRIMARY KEY (ID_Сотрудника),
```

```
CONSTRAINT K7 UNIQUE (ID_Сотрудника, ID_Отдела),
```

```
CONSTRAINT C5 FOREIGN KEY (ID_Отдела) REFERENCES Отделы (ID_Отдела)
```

```
);
```



## Изменение структуры таблицы

- добавление поля в таблицу

**ALTER TABLE** имя\_таблицы **ADD** (<определение\_поля>[,<определение\_поля>...]);

- изменение определения поля

**ALTER TABLE** имя\_таблицы **MODIFY** [COLUMN] (<определение\_поля>[,<определение\_поля>...]);

- добавление нового ограничения

**ALTER TABLE** имя\_таблицы **ADD CONSTRAINT** <определение\_ограничения>;

# Изменение структуры таблицы

- удаление первичного ключа таблицы

```
ALTER TABLE имя_таблицы DROP PRIMARY KEY;
```

- переименование поля

```
ALTER TABLE имя_таблицы RENAME COLUMN старое_имя_поля TO новое_имя_поля;
```

# Изменение таблицы

Отделы	
K5	<u>ID Отдела</u>
	Название

```
CREATE TABLE "Отделы"  
(  
  "ID Отдела" SERIAL,  
  "Название" VARCHAR(30) NOT NULL,  
);  
ALTER TABLE "Отделы" ADD PRIMARY KEY ("ID Отдела");
```

## Удаление таблицы

- ❑ **DROP TABLE** имя\_таблицы [RESTRICT | CASCADE];
- ❑ **DROP TABLE** имя\_таблицы **IF EXISTS**;