Базы данных

Язык запросов SQL.
Введение

Пример таблицы реляционной БД

Табельный номер	*ФИО сотрудника*	*Должность*	*Оклад*	*Год рождения*	*Отдел*
023	Волкова Елена Павловна	секретарь	26000	1985	2
113	Белов Сергей Юрьевич	инженер	39800	1980	1
101	Рогов Сергей Михайлович	директор	62000	1972	2
056	Панина Анна Алексеевна	инженер-программист	41800	1978	1
...	...	...	...	...	...
098	Фролов Юрий Вадимович	начальник отдела	49200	1971	9

Термины. Свойства отношения

Отношение, таблица

первичный ключ

столбец

Табельный номер	*ФИО сотрудника*	*Должность*	*Оклад*	*Год рождения*
023	Волкова Елена Павловна	секретарь	26000	1985
113	Белов Сергей Юрьевич	инженер	39800	1980

описание (схема отношения)

строка, запись, кортеж

Отношение обладает двумя основными свойствами:

1. В отношении не должно быть одинаковых кортежей, т.к. это множество.
2. Порядок кортежей в отношении несущественен.

Организация связей между таблицами

Связь один-ко-многим: Отделы – Сотрудники

Таблица «Сотрудники»

Табельный номер	*ФИО сотрудника*	*Отдел*
023	Волкова Елена Павловна	2
113	Белов Сергей Юрьевич	1
101	Рогов Сергей Михайлович	2
056	Панина Анна Алексеевна	1
…	…	…
098	Фролов Юрий Вадимович	9

Таблица «Отделы»

Номер отдела	*Название отдела*
1	Информационный отдел
2	Администрация
3	Отдел кадров
…	…
9	Проектный отдел

«Номер отдела» – первичный ключ в таблице «Отделы»

«Отдел» – внешний ключ в таблице «Сотрудники» к таблице «Отделы»

Организация связей между таблицами

Связь многие-ко-многим: Проекты – Сотрудники

Таблица «Сотрудники»

ФИО	***Номер***
Волкова Е.П.	023
Белов С.Ю.	113
Рогов С.М.	101
Панина А.А.	056
Фролов Ю.В.	098
…	…

Таблица «Участие»

Участник	*Роль*	*Проект*
113	исполнитель	23/Н
101	руководитель	18-К
056	исполнитель	18-К
101	консультант	09/Р
098	руководитель	23/Н
…	…	…

Таблица «Проекты»

Шифр	*Название проекта*
23/Н	АИС "Налог"-2
18-К	ИПС "Жители"
09/Р	ГИС "Город"
…	…

В таблице «Участие»:
«Участник» – внешний ключ к таблице «Сотрудники»
«Проект» – внешний ключ к таблице «Проекты»

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, SQL/2003) – относится к объектно-реляционной модели данных.
- 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.

Работа с SQL

- 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 заключается в одинарные кавычки:

    'A', '2' , 'строка', 'другая строка'
  - ✓ Однострочный комментарий начинается с символов '--'.
  - ✓ Многострочный комментарий заключается в символы /* ... */.
- Запуск MySQL: клиент mysql запускается с использованием программы «Командная строка»:

mysql –uимя_пользователя -рпароль

mysql> use имя_базы_данных;

Команды DDL

CREATE – создание объекта.

ALTER – изменения структуры объекта.

DROP – удаление объекта.

Общий вид синтаксиса команд DDL:

create
alter
drop

} *тип_объекта имя_объекта [параметры];*

Создание таблиц

CREATE TABLE *имя_таблицы*
(*имя_поля* *тип_данных* [(*размер*)] [NOT NULL]
[DEFAULT *выражение*]
[*ограничения_целостности_поля*…]
.,..
[, *ограничения_целостности_таблицы* .,..]
)
[*параметры*];

ограничения_целостности (ОЦ):
[CONSTRAINT *имя_ОЦ*] *название_ОЦ* [*параметры*]

Типы данных MySQL

- Символьные типы:
  - ✓ **CHAR** [(длина)] – строка фиксированной длины.
    Длина по умолчанию – 1, максимальная длина 255 б.
    Строка дописывается до указанной длины пробелами.
  - ✓ **VARCHAR** (длина) – строка переменной длины.
    Максимальная длина 255 б. Хранятся только значащие символы.
- Числовые типы:

  NUMERIC [(точность[, масштаб])] – используется для представления чисел с заданной точностью.

  Масштаб по умолчанию – 0.
  numeric(4) – числа от -999 до 9999
  numeric(8,2) – числа от -99999.99 до 999999.99

  MySQL поддерживает все числовые типы данных языка SQL92 по стандартам ANSI/ISO. Они включают в себя типы точных числовых данных (NUMERIC, DECIMAL, INTEGER и SMALLINT) и типы приближенных числовых данных (FLOAT, REAL и DOUBLE PRECISION). Ключевое слово INT является синонимом для INTEGER, а ключевое слово DEC - синонимом для DECIMAL.

Числовые типы данных MySQL

Таблица 1 – Целочисленные типы данных

Тип	Диапазон	Память (байт)
TINYINT[(M)]	-127..128 или 0..255	1
BIT		1
BOOL		1
SMALLINT[(M)]	-32768..32767 или 0..65535	2
MEDIUMINT[(M)]	-8388608..8388607 или 0..16777215	3
INT[(M)]	$-2^{31}..2^{31} - 1$ или $0..2^{32} - 1$	4
INTEGER[(M)]		4
BIGINT[(M)]	$-2^{63}..2^{63} - 1$ или $0..2^{64} - 1$	8

Таблица 2 – Типы данных с плавающей запятой

Тип	Диапазон	Память (байт)
FLOAT(*точность*)	зависит от точности	различна
FLOAT[(M, D)]	±1.175494351E-38 ±3.402823466E+38	4
DOUBLE[(M, D)]	±1.7976931348623157E+308 ±2.2250738585072014E-308	8
DOUBLE PRECISION[(M, D)]	±1.7976931348623157E+308 ±2.2250738585072014E-308	8
REAL[(M, D)]	±1.7976931348623157E+308 ±2.2250738585072014E-308	8
DECIMAL[(M[, D])]	различный	M + 2
NUMERIC[(M, D)]	различный	M + 2
DEC[(M, D)]	различный	M + 2
FIXED[(M, D)]	различный	M + 2

Типы данных MySQL: дата и время

Тип	Описание
DATE	Дата в формате ГГГГ-ММ-ДД
TIME	Время в формате ЧЧ:ММ:СС
TIMESTAMP	Дата и время в формате timestamp, выводится в виде ГГГГММДДЧЧММСС
DATETIME	Дата и время в формате ГГГГ-ММ-ДД ЧЧ:ММ:СС

Величины DATETIME, DATE и TIMESTAMP задаются:

• Как строка в формате 'YYYY-MM-DD HH:MM:SS' ('YYYY-MM-DD') или в формате 'YY-MM-DD HH:MM:SS‘ ('YY-MM-DD'). Допускается ``облегченный'' синтаксис - можно использовать любой знак пунктуации в качестве разделительного между частями разделов даты или времени.

Например, величины '98-12-31 11:30:45', '98.12.31 11+30+45', '98/12/31 11*30*45' и '98@12@31 11^30^45' являются эквивалентными.

• Как строка без разделительных знаков в формате 'YYYYMMDDHHMMSS' ('YYYYMMDD') или в формате 'YYMMDDHHMMSS‘ ('YYMMDD').

• Как число в формате YYYYMMDDHHMMSS или в формате YYMMDDHHMMSS.

• Как результат выполнения функции, возвращающей дату (например, функции NOW() или CURRENT_DATE).

Недопустимые значения величин DATETIME, DATE или TIMESTAMP преобразуются в значение ``ноль'' соответствующего типа величин ('0000-00-00 00:00:00', '0000-00-00', или 00000000000000).

Типы данных MySQL: время

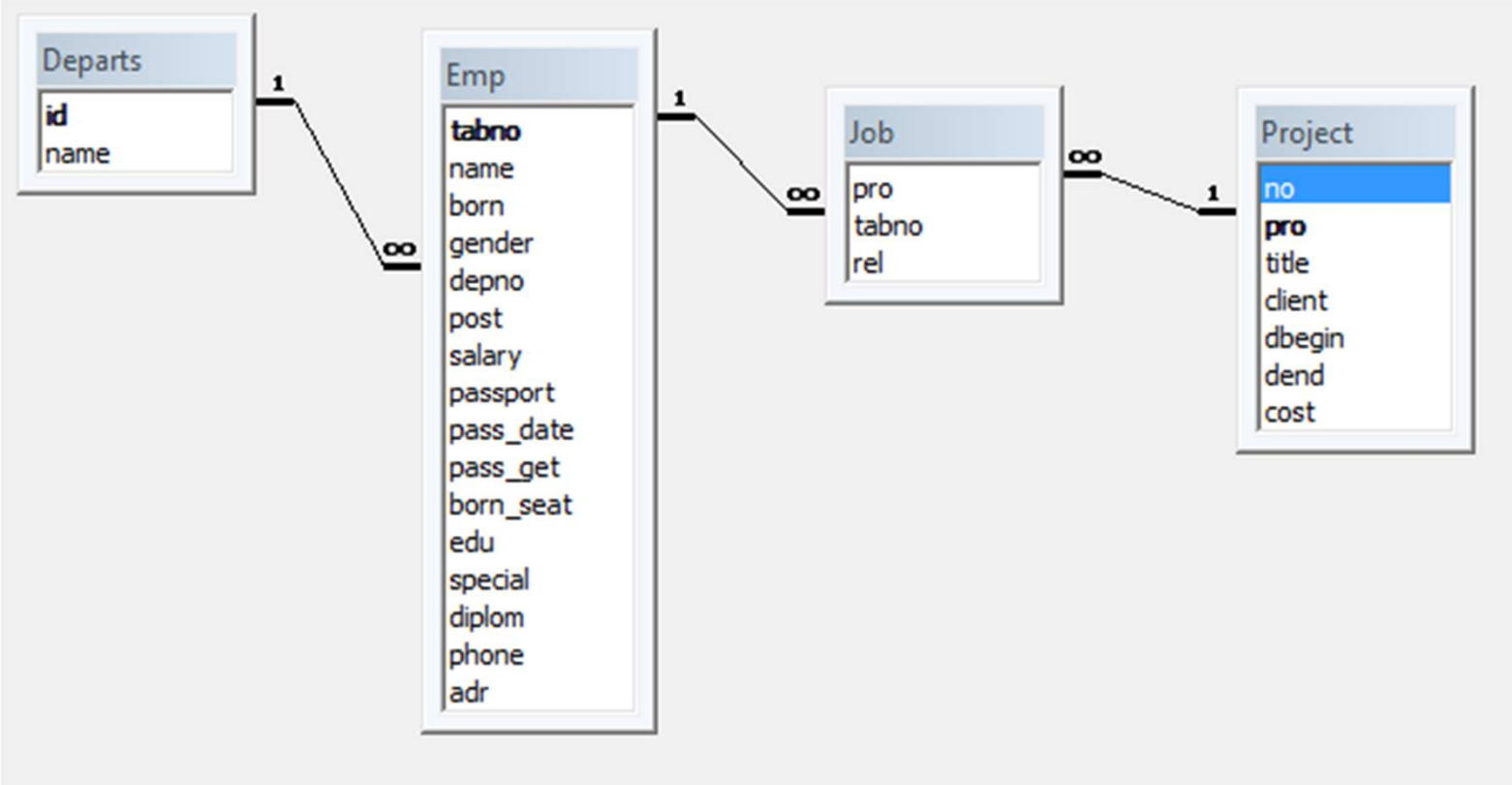
MySQL извлекает и выводит величины типа TIME в формате 'HH:MM:SS'.
Величины TIME могут изменяться в пределах от '-838:59:59' до '838:59:59'.
Величины TIME могут быть заданы в различных форматах:
Как строка в формате 'D HH:MM:SS' (следует учитывать, что MySQL пока не обеспечивает хранения дробной части величины в столбце рассматриваемого типа). Можно также использовать одно из следующих "облегченных" представлений: HH:MM:SS, HH:MM, D HH:MM, D HH или SS. Здесь D – это дни из интервала значений 0-33.
Как строка без разделителей в формате 'HHMMSS', при условии, что строка интерпретируется как дата.
Как число в формате HHMMSS, при условии, что строка интерпретируется как дата. Например, величина 101112 понимается как '10:11:12'.
Как результат выполнения функции, возвращающей величину, приемлемую в контексте типа данных типа TIME (например, CURRENT_TIME).
Примеры:
'101112' –> '10:11:12‘
'109712' –> '00:00:00'
'8:3:2' –> '08:03:02'
'1112' и 1112 –> '00:11:12‘ – крайние правые разряды – секунды!
'11:12' –> '11:12:00‘ – двоеточие означает, что крайние левые разряды – часы

Ограничения целостности

По стандарту ASNI/SQL поддерживаются следующие ограничения целостности:

- уникальность (значений атрибута или комбинации значений полей):
  UNIQUE (*имя_поля1* [, *имя_поля2*,...])
- обязательность / необязательность:
  NOT NULL / NULL
- первичный ключ:
  PRIMARY KEY(*имя_поля1* [, *имя_поля2*,...])
- внешний ключ:
  FOREIGN KEY(*имя_поля1* [, *имя_поля2*,...]) **REFERENCES** *имя_таблицы* [(*имя_поля1* [, *имя_поля2*,...])]
  [ON DELETE CASCADE | ON DELETE SET NULL]
- условие на значение поля:
  CHECK (*условие*)
  Например: check (salary>=4500), check (date2 > date1)

Пример БД: проектная организация



Departs – отделы,

Emp – сотрудники,

Project – проекты,

Job – участие в проектах.

Пример БД: проектная организация

Emp – сотрудники:

tabno – табельный номер сотрудника, первичный ключ;
name – ФИО сотрудника, обязательное поле;
born – дата рождения сотрудника, обязательное поле;
gender – пол сотрудника, обязательное поле;
depno – номер отдела, обязательное поле, внешний ключ;
post – должность сотрудника;
salary – оклад, больше МРОТ;
passport – серия и номер паспорта, уникальный обязательный атрибут;
pass_date – дата выдачи паспорта, обязательное поле;
pass_get – кем выдан паспорт, обязательное поле;
born_seat – место рождения сотрудника;
edu – образование сотрудника;
special – специальность по образованию;
diplom – номер диплома;
phone – телефоны сотрудника;
adr – адрес сотрудника;
edate – дата вступления в должность, обязательное поле:
chief – руководитель, внешний ключ на поле tabNo.

Пример БД: проектная организация

Departs – отделы:

did – номер отдела, первичный ключ;
name – название отдела, обязательное поле.

Project – проекты:

No – номер проекта, первичный ключ;
title – название проекта, обязательное поле;
pro – краткое название проекта, обязательное уникальное поле;
client – заказчик, обязательное поле;
dbegin – дата начала выполнения проекта, обязательное поле;
dend – дата завершения проекта, обязательное поле;
cost – стоимость проекта, обязательное поле.

Job – участие в проектах:

pro – краткое название проекта, внешний ключ;
tabNo – номер сотрудника, участвующего в проекте, внешний ключ;
rel – роль сотрудника в проекте; может принимать одно из трех значений:
'исполнитель', 'руководитель', 'консультант'.
Первичный ключ – комбинация полей **pro** и **tabNo**.

Создание таблиц БД проектной организации

Таблица «Отделы» (Depart):

```
create table depart (did numeric(4) constraint pk_depart  PRIMARY KEY,
                     name varchar(100) not null
) TYPE=INNODB ;
```

Таблица «Сотрудники» (Emp):

```
create table emp (    tabno numeric(6) constraint pk_emp PRIMARY KEY,
            name   varchar(50) not null,
            born     date not null,
            gender char(1)   not null,
            depno  numeric(4) not null  constraint fk_depart  REFERENCES depart,
            post     varchar(50) not null,
            salary  numeric(8,2) not null  check (salary > 4630),
            passport     char(10) not null   constraint passport_uniq UNIQUE,
            pass_date  date  not null,        pass_get varchar(100) not null,
            born_seat  varchar(100),          edu varchar(30),
            special       varchar(100),       diplom varchar(40),
            phone         varchar(30),        adr varchar(80),
            edate date  not null  default  current_date,
            chief  numeric(6) constraint fk_emp  REFERENCES  emp
) TYPE=INNODB ;
```

Создание таблиц БД проектной организации

Таблица «Проекты» (Project):

```
create table project (No numeric(5) constraint pk_project  primary key,
                      title   varchar(200) not null,
                      pro   varchar(15) not null constraint  pro_uniq unique,
                      client   varchar(100) not null,
                      dbegin date not null,
                      dend   date not null,
                      cost    numeric(9)
) TYPE=INNODB;
```

Таблица «Участие в проектах» (Job):

```
create table job (  pro      varchar(15)   not null  references  project (abbr),
                    tabNo  numeric(6)    not null  references  emp,
                    rel      varchar(20)    default  'исполнитель',
          primary key (tabno, pro),
          check ( rel  IN  ('исполнитель', 'руководитель', 'консультант') )
) TYPE=INNODB ;
```

Подмножество команд DML

- **INSERT** – добавление строк в таблицу.
  - ✓ Добавляет одну или несколько строк в указанную таблицу.
- **UPDATE** – изменение данных.
  - ✓ Изменяет значения одного или нескольких полей в записях указанной таблицы.
  - ✓ Можно указать условие, по которому выбираются обновляемые строки.
  - ✓ Если условие не указано, обновляются все строки таблицы.
  - ✓ Если ни одна строка не удовлетворяет условию, ни одна строка не будет обновлена.
- **DELETE** – удаление строк из таблицы.
  - ✓ Удаляет одну или несколько строк из таблицы.
  - ✓ Можно указать условие, по которому выбираются удаляемые строки.
  - ✓ Если условие не указано, удаляются все строки таблицы.
  - ✓ Если ни одна строка не удовлетворяет условию, ни одна строка не будет удалена.

Добавление данных

INSERT – добавление строк в таблицу:

INSERT INTO *имя_таблицы* [(*список_полей_таблицы*)]
{ VALUES (*список_выражений*) | *запрос* };

Примеры:

```
-- Добавить в таблицу "Отделы" новую запись (все поля):
  insert into  depart
     values(7, 'Договорной отдел');

-- Добавить в таблицу "Сотрудники" новую запись (не все поля):
  insert into emp (tabno, name, born, gender, depno, passport, pass_date_pass_get,
              post, salary, phone)
     values( 301, 'САВИН АНДРЕЙ ПАВЛОВИЧ', '1969.11.07',
              'М', 5, '4405092876', '1999.02.15', 'ОВД "Митино" г.Москвы',
              'программист', 58000, '(495)121-34-11');
```

Замечание: значение по умолчанию используется только тогда, когда значение поля не вводится в явном виде.

Изменение данных

UPDATE – изменение данных:

UPDATE *имя_таблицы*
SET *имя_поля1 = выражение1* [, *имя_поля2 = выражение2*,…]
[WHERE *условие*];

Примеры:

-- Изменить статус сотрудника Бобкова Л.П., табельный номер 74, по отношению к проекту 30."Система автоматизированного управления предприятием":

```
update job
        set rel = 'консультант'
        where tabno = 74 and pro = 30;
```

-- Перевести сотрудника Жаринова А.В., табельный номер 68, на должность ведущего программиста и повысить оклад на три тысячи рублей:

```
update emp
        set post = 'ведущий программист', salary = salary+8000
        where tabno = 68;
```

Удаление данных

DELETE – удаление строк из таблицы:

DELETE FROM *имя_таблицы*

[WHERE *условие*];

Примеры.

-- Удалить сведения о том, что сотрудник Афонасьев В.Н., табельный номер 147, участвует в проектах:

```
delete from job
        where tabno=147;
```

-- Удалить сведения о сотруднике Афонасьеве В.Н., табельный номер 147:

```
delete from emp
        where tabno = 147;
```