Go

Начальный курс по Go

NEW

new

Функция new создаёт переменную нужного типа, кладёт туда нулевое значение и возвращает адрес

```
var ptr *int = new(int)
```

CONSTRUCTORS

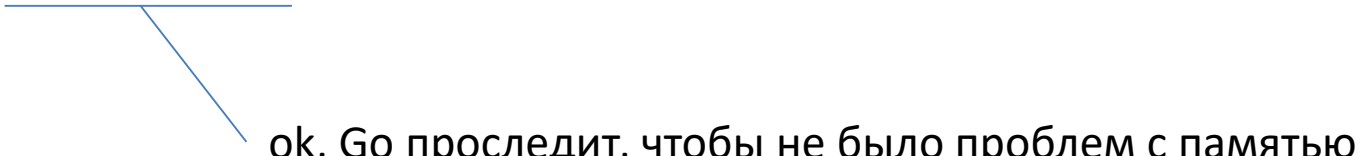
Constructors

Иногда необходимо выполнить некоторую логику перед созданием "объекта" нужного типа (т.е. простое нулевое значение или значение, созданное через литерал, недостаточно)

Constructor

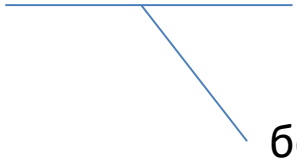
Специальная функция, которая создаёт нужный объект:

```
func NewStuff(args ...) *Stuff {
  // TODO: some checks
  return &Stuff{...}
}
```



Composite literals & new

&Stuff{} эквивалентно new(Stuff)



MAKE

make

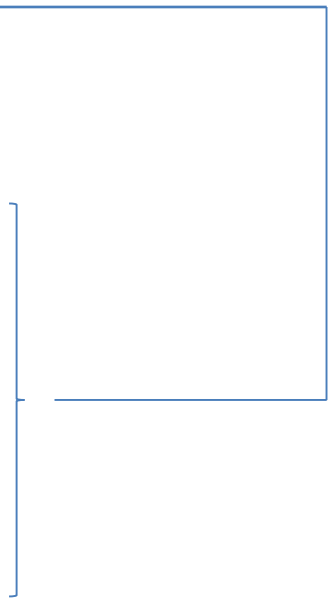
make в отличие от new создаёт инициализированные объекты и возвращает значение (не указатель)

Используется только для создания slice, map и channel

STRUCT COMPOSITION

Композиция

```
type account struct {
  id int64
  person
}
type person struct {
  name string
  age uint
  ...
}
```



account содержит person, но при этом:

1. позволяет обращаться account.name
2. реализует все интерфейсы person

Композиция

Композицию нужно использовать только тогда, когда один тип (или интерфейс) содержит все поля или методы другого типа (или интерфейса)

Композиция

```
type MFU interface {
  Printer
  Scanner
}
```

можно писать только интерфейсы

SLICES

Слайсы

```
var slice []int
```

nil - слайс

```
slice := []int{}

slice := make([]int, 5)

slice := make([]int, 5, 10)
```

Операции

Добавление в слайс (даже пустой)

```
slice = append(slice, 1)

slice = append(slice, another...)
```

Операции

Срезы

```
copy := slice[:]

sub := slice[2:4]

head := slice[:1]

tail := slice[len(slice) - 2:]
```

Операции

Важно: слайс указывает на массив (до тех пор пока Go не решит, что для этого слайса нужно создать новый массив, например, при добавлении в него элементов)

Операции

Желательно не изменять слайс (либо менять копию)

Копию создать можно через функцию copy

Трюки со слайсами

https://github.com/golang/go/wiki/SliceTricks

Нужно пройтись по всем

MAP

map

key-value хранилище:

```
var data map[string]int // неинициализированный
data := map[string]int{} // пустой
data := map[string]int{
  "key": value
}
```

map

```
data := make(map[string]int)

data := make(map[string]int, 10)
```

Операции

```
value := data[key]

value, ok := data[key]

data[key] = value // только для инициализированных

delete(data, key)
```

Операции

```
for key, value := range data {
    ...
}
```

Порядок не гарантирован

PACKAGE

Package

Единица инкапсуляции: один или несколько файлов с расширением .go (чаще всего определяется каталогом)

Package

Для использования имён (переменных, констант или функций) из другого пакета необходимо импортировать пакет (см. fmt)

Для использования доступны только имена, начинающиеся с большой буквы (exported), например, fmt.Println

SQL

СУБД

Почему бы всё не хранить в файлах, зачем нужны СУБД?

СУБД

- Структура информации
- Запросы
- Целостность
- Транзакции
- Журналирование
- Многопользовательский доступ

Структура

Поддержка определённой структуры информации

Запросы

Функциональность хранилища по извлечению, сохранению, изменению и удалению данных

CRUD

Целостность

Функциональность хранилища сохранению консистентности данных

Реляционная модель

Основные идеи заложены Эдгаром Коддом

Codd, F. A Relational Model of Data for Large Shared Data Banks

Реляционная модель

Id	Name	Gender	Partner	Salary
1	Vasya	M	Masha	50 000
2	Masha	F	Vasya	50 000
3	Petya	M		…
…	…	…		…

Реляционная модель

Тип данных - целое, строка, boolean и т.д.

Домен - допустимое множество значений

Атрибут - именованное вхождение домена в заголовок отношения

Кортеж - набор пар атрибут-значение (где каждый атрибут встречается только один раз)

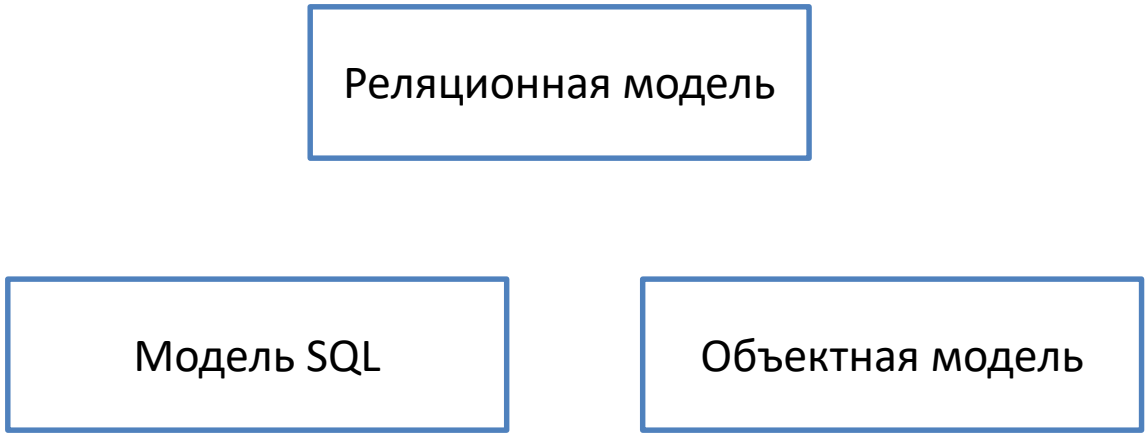
Отношение - множество кортежей

Relational & SQL

На самом деле реляционная модель данных

и SQL – это две разных модели данных

Реляционная модель



Составляющие

- Структурная (в SQL – таблицы)
- Манипуляционная (DML, DRL)
- Целостная

Реляционная алгебра

- Выборка - выбираем записи по условию
- Проекция - отсечение ненужных атрибутов
- Объединение - записи из обоих множеств
- Пересечение - есть в обоих выбранных множествах
- Разность - есть в одном, но нет в другом
- Произведение - декартово произведение
- Деление
- Соединение - соединение по значению атрибута (ключу)

http://citforum.ru/database/dblearn/dblearn04.shtml

"Главные" люди

- Кодд
- Дейт и Дарвен

Ограничения целостности

- Первичный ключ (сущностная целостность)
- Внешний ключ (ссылочная целостность)

Первичный ключ

Минимальное подмножество заголовков отношения уникальное для любого кортежа

Минимальность заключается в том, что если хотя бы один из заголовков удалить, то набор оставшихся не будет уникальным для всех кортежей

Если первичный ключ не объявлен явно, то он им является весь заголовок

Внешний ключ

Внешний ключ – ссылка на атрибут или группу атрибутов другого отношения (либо того же самого)*

В реляционной модели - на потенциальный ключ отношения (не обязательного другого)

Виды СУБД

Объектно-ориентированные СУБД

Реляционные СУБД (SQL)

Объектно-реляционные СУБД

Манифесты и истинно-реляционная модель

SQL

Набор таблиц, состоящих из строк

Упорядоченный набор столбцов

определённого типа данных

ER-модели

На самостоятельное изучение

SQL

SQLite

MySQL

PostgreSQL

MS SQL Server

Oracle

Firebird

SQL Standards

1986	SQL-86
1989	SQL-89
1992	SQL-92
1999	SQL:1999
2003	SQL:2003
2006	SQL:2006
2008	SQL:2008
2011	SQL:2011
2016	SQL:2016

SQL Dialects

У каждой базы данных свой диалект

(вариация) SQL

SQL

- Data definition
- Data retrieval
- Data manipulation
- Access control
- Data sharing
- Data integrity

SQLITE

SQLite

Встраиваемая база данных

http://sqlite.org

GoLand

Database -> Data Source добавить

SQLite

Типы данных

- NULL
- INTEGER
- REAL
- TEXT
- BLOB

http://sqlite.org/datatype3.html

NULL

- Ничему не равен, включая самого себя
- Столбец любого типа по умолчанию может содержать NULL

IS NULL

IS NOT NULL

DDL

```
CREATE TABLE <tbl> (
   <name> <type> <restrictions>
   [CONSTRAINT <name> <expr>]
);
```

На уровне колонки

[CONSTRAINT <name>] <restriction>

Restrictions:

NOT NULL

CHECK (<expr>)

DEFAULT <expr>

UNIQUE <params>

PRIMARY KEY <params>

REFERENCES <tbl> [<column>]

NOT NULL

В поле нельзя записать NULL

CHECK

Может использоваться как в столбце, так и в конце определения таблицы

Обычно принято ограничения, затрагивающие только конкретный столбец записывать в столбце, а относящиеся к нескольким столбцам в конце таблицы

NOT NULL

Эквивалентно CHECK (<col> IS NOT NULL)

DEFAULT

Позволяет указать DEFAULT значение

В DEFAULT выражении нельзя ссылаться на другие столбцы

DEFAULT вычисляется в момент записи

UNIQUE

Позволяет установить ограничение уникальности на столбец, и на несколько столбцов

PRIMARY KEY

Устанавливает ограничение первичного ключа на столбец или группу столбцов

ПЗ: Менеджеры

Создать таблицу менеджеры:

1. ID (первичный ключ + автоинкремент)
2. Имя
3. Логин (должен быть уникальный)

Создание таблиц

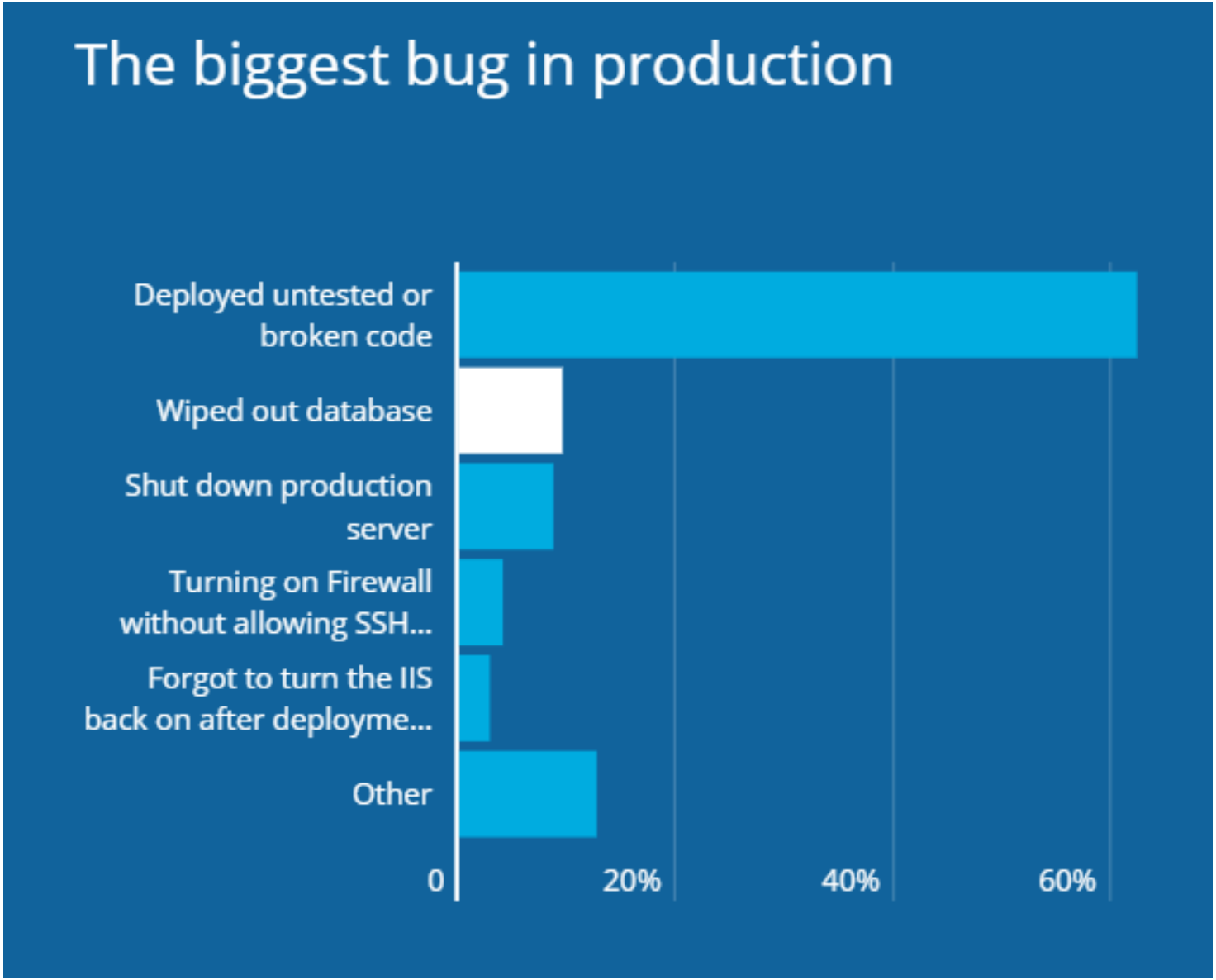
```
CREATE TABLE managers (
    id INTEGER AUTO INCREMENT PRIMARY KEY,
    name TEXT NOT NULL,
    login TEXT UNIQUE
);
```

DDL

Удаление таблицы:

DROP TABLE <tbl>;

Никогда не делайте на Production базе



Отчёт HackerRank 2019

Манипуляция данными

INSERT – вставка

UPDATE – обновление

DELETE – удаление

SELECT – выборка (Retrieval)

INSERT

```
INSERT INTO <tbl> VALUES (…);

INSERT INTO <tbl> (<col>, <col>) VALUES (…);

INSERT INTO <tbl> (<col>, <col>) VALUES
    (…),
    (…)
;
```

ДЗ

Продумать и описать две таблицы (в дополнение к менеджерам):

1. Товары (id, name, price, qty)
2. Продажи - столбцы и типы определить самим и согласовать с соседями по парте

Спасибо за внимание

Ильназ Гильязов

2020г.