



**ТИНЬКОФФ**

# БД: миграция как микросервис

Avito Backend United meetup #7: Долма, Ереван

# Dima Burmistrov

*#agile #clouds #python #infra*

*Строю облака*



**ТИНЬКОФФ**

*Руководитель направления  
разработки продукта HWaaS*

*Роботизирую датацентры*

# БД: миграция как микросервис

*История одной миграции*

# Agenda



БД: миграция как микросервис

*История одной миграции*

# Agenda

БД: миграция как микросервис

*История одной миграции*

# Agenda

БД: миграция как микросервис

История одной миграции

# Agenda

БД: миграция как микросервис

*История одной миграции*

# БД: миграция как микросервис

*История одной миграции*



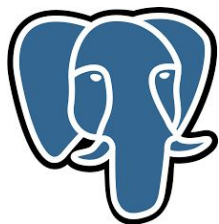




*Смена типа хранилища*



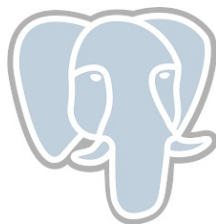
*Смена типа хранилища*



*Смена вендора*



*Смена типа хранилища*



*Смена вендора*



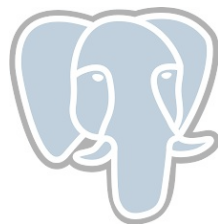
***UPDATE ...;***  
***ALTER TABLE ...;***

*Внутренние изменения*

## Цели



*Смена типа хранилища*



*Смена вендора*



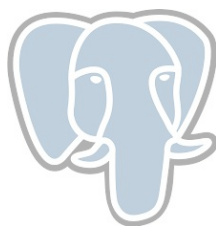
***UPDATE ...;***

***ALTER TABLE ...;***

*Внутренние изменения*



*Смена типа хранилища*



*Смена вендора*



***UPDATE ...;***

***ALTER TABLE ...;***

*Внутренние изменения*

## *Цели*

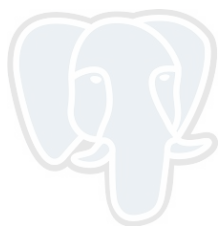
*функциональные*

*эксплуатационные*

*экономические*



*Смена типа хранилища*



*Смена вендора*



**UPDATE ...;**

**ALTER TABLE ...;**

*Внутренние изменения*

## Цели

*функциональные*

*эксплуатационные*

*экономические*

## Типовое решение

### Migration Code

```
-- +goose Up
CREATE TABLE post (
  id int NOT NULL,
  title text,
  body text,
  PRIMARY KEY(id)
);

-- +goose Down
DROP TABLE post;
```



## Типовое решение

Migration Code

Migration Tools

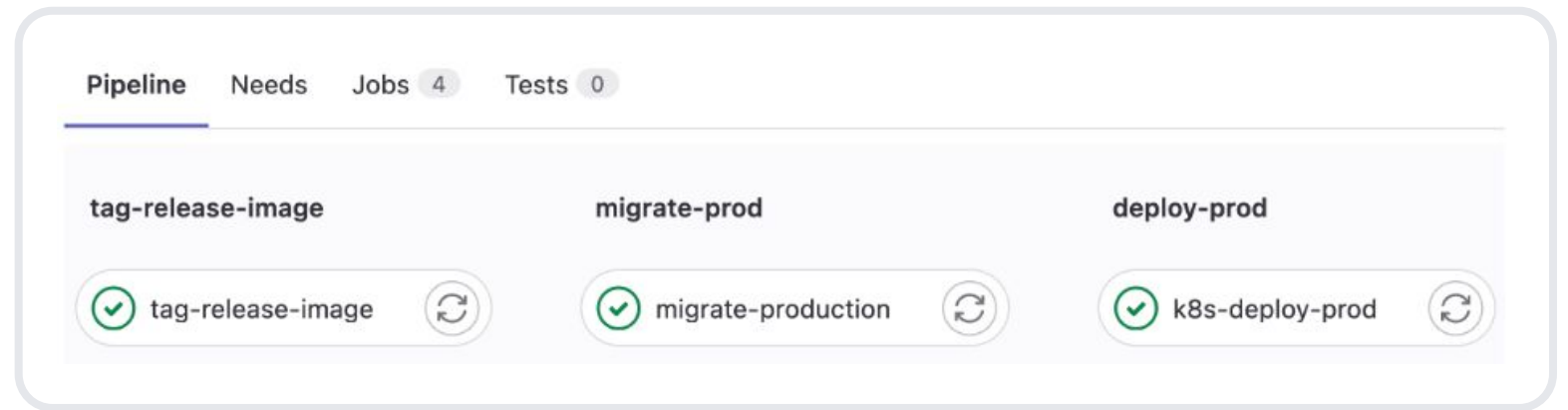
- [South](#) (Python/Django)
- [Alembic](#) (Python/SQLAlchemy)
- [Nomad](#) (Python)
- [Migrations](#) (PHP/Doctrine)
- [Phinx](#) (PHP)
- [migrate4j](#) (Java)
- [Active Record Migrations](#) (Ruby/ROR)
- [Goose](#) (SQL)
- [Agnostic](#)
- [migrate.sh/pgcli/...](#)

## Типовое решение

Migration Code

Migration Tools

CI/CD



## Типовое решение

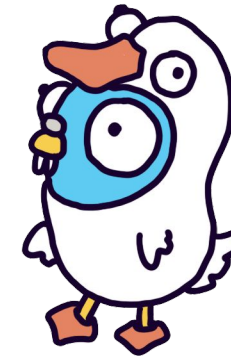
Migration Code

Migration Tools

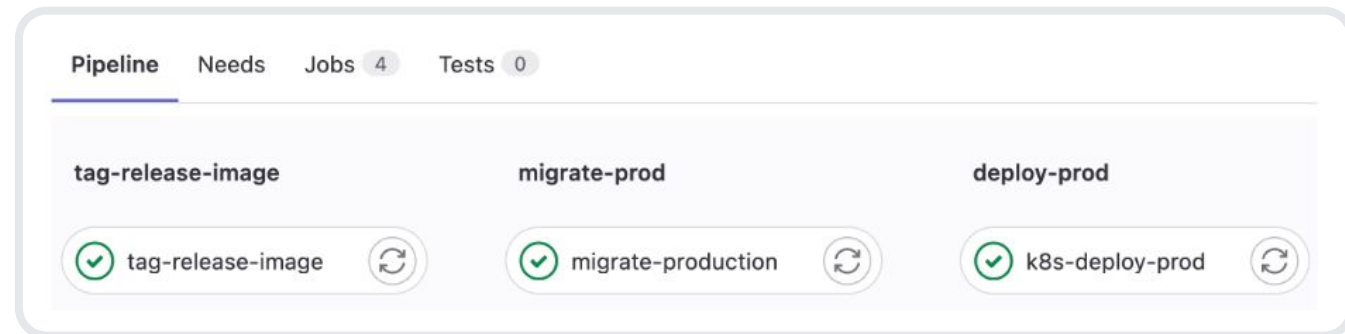
CI/CD

```
-- +goose Up
CREATE TABLE post (
  id int NOT NULL,
  title text,
  body text,
  PRIMARY KEY(id)
);

-- +goose Down
DROP TABLE post;
```



Goose

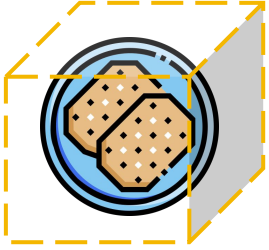




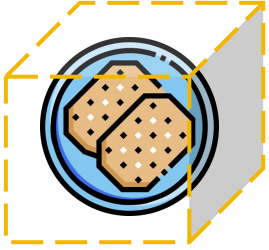
# БД: миграция как микросервис

## *История одной миграции*

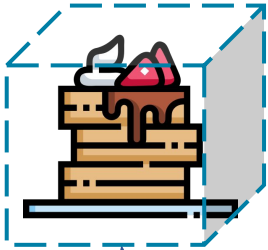
Legacy



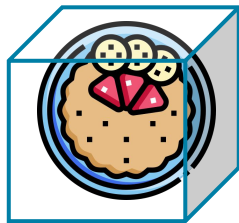
Legacy



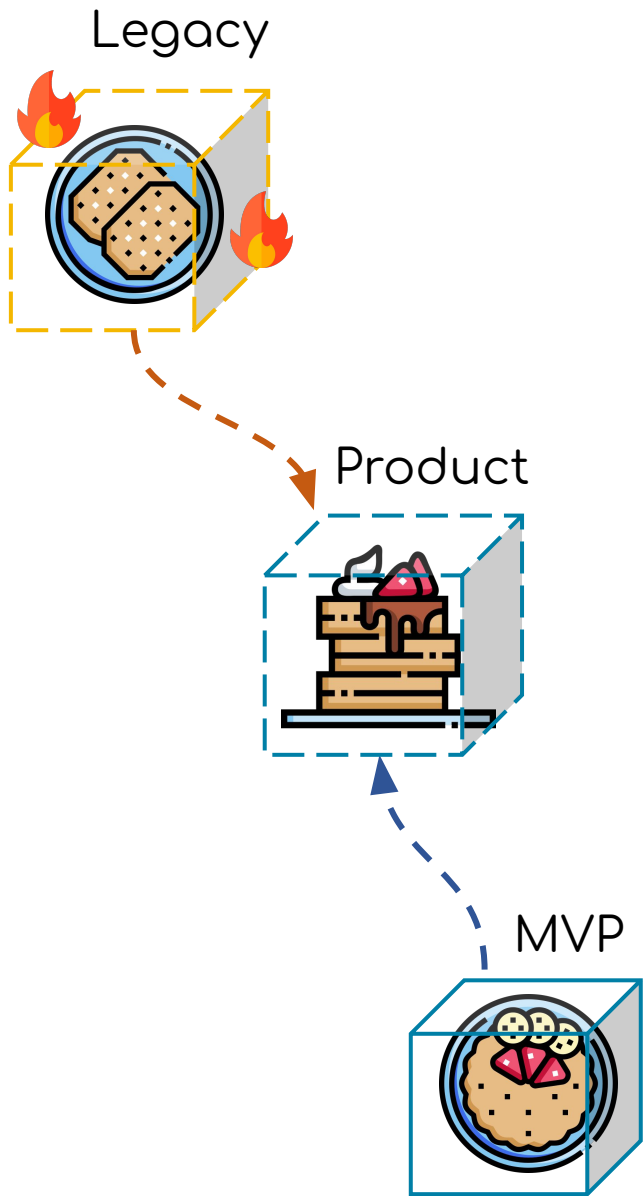
Product



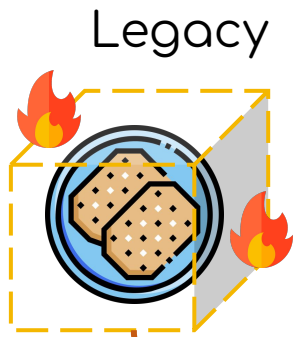
MVP



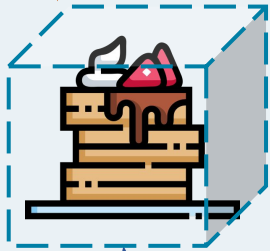
# #Agile



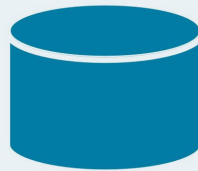
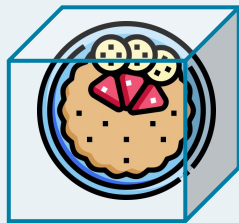
# #Agile



Product

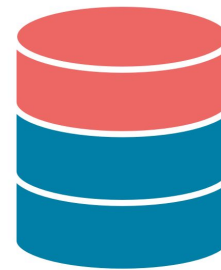
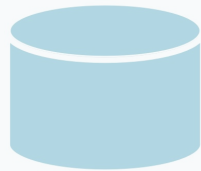
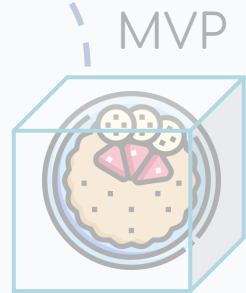
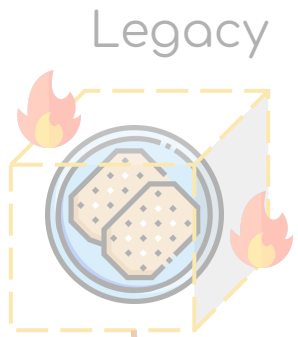


MVP



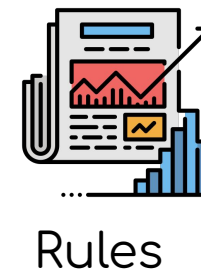
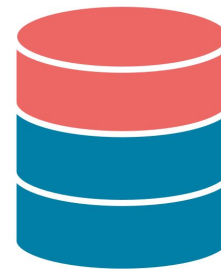
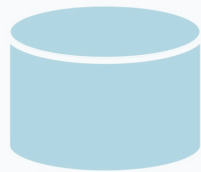
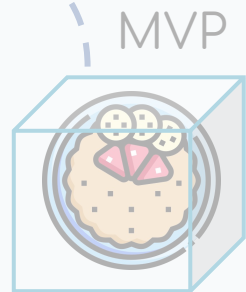
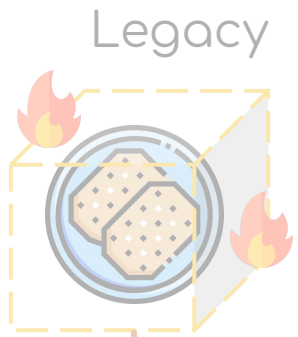


# #Agile



Rules

# #Agile

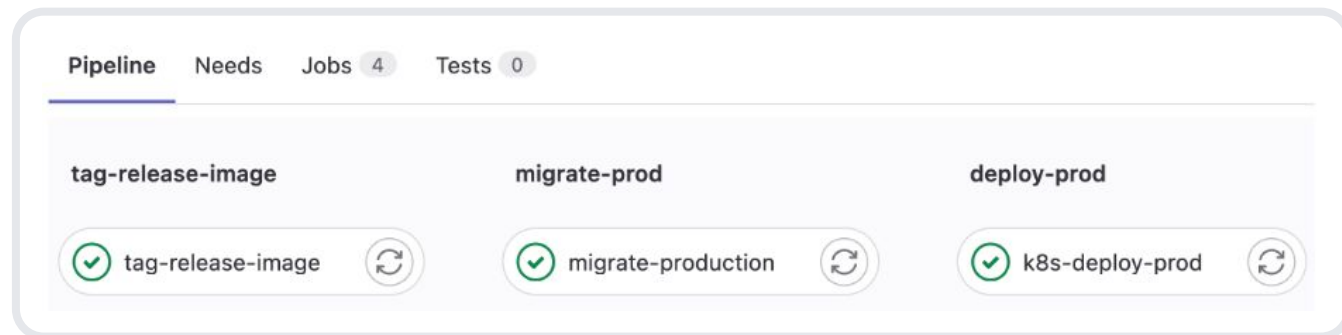


Migration

## Наше базовое решение

```
class Migration:
    def upgrade(self, session):
        stmt = """
CREATE TABLE post (
    id int NOT NULL,
    title text,
    body text,
    PRIMARY KEY(id));
"""
        session.execute(stmt)

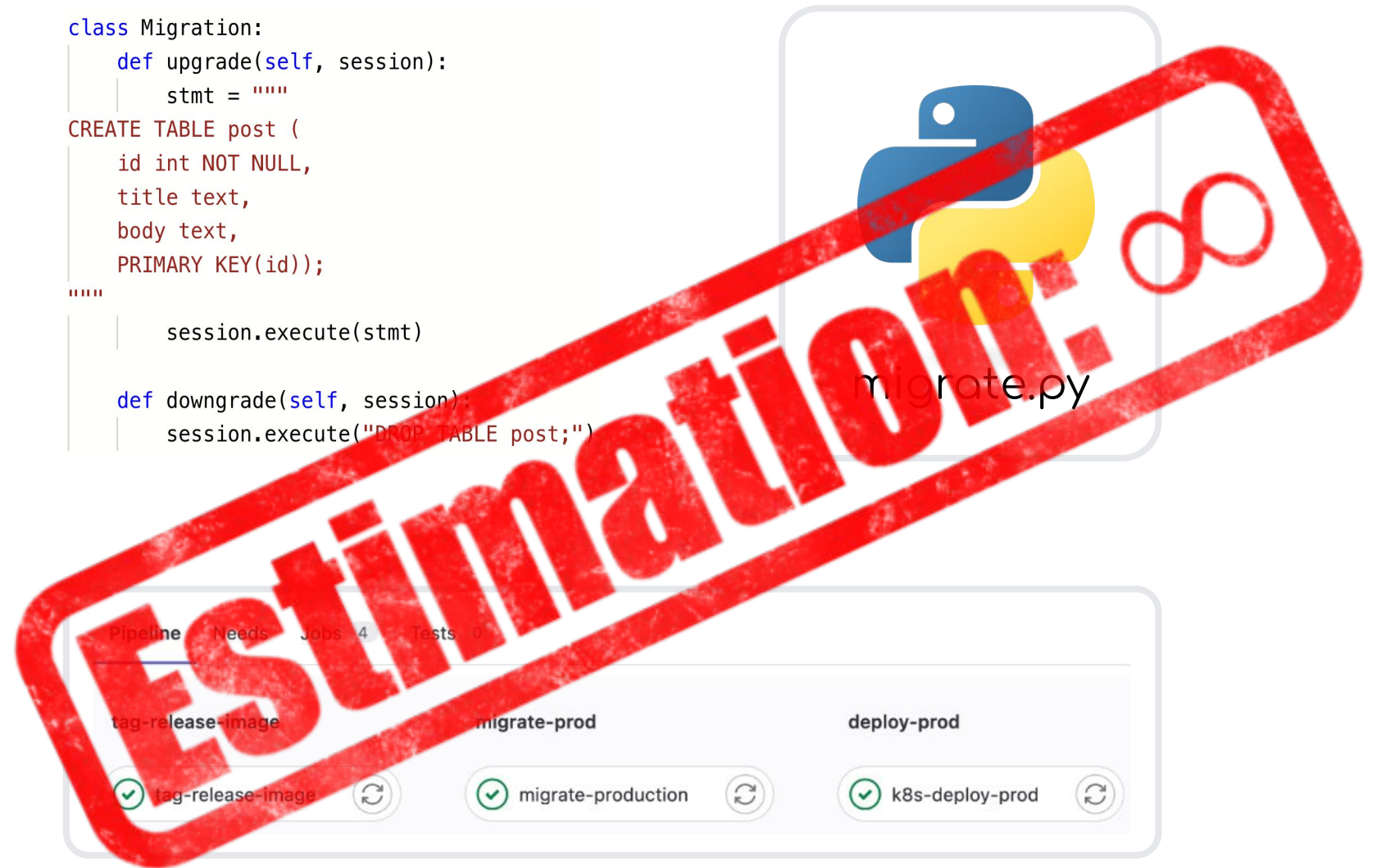
    def downgrade(self, session):
        session.execute("DROP TABLE post;")
```



## Наше базовое решение

```
class Migration:
    def upgrade(self, session):
        stmt = """
CREATE TABLE post (
    id int NOT NULL,
    title text,
    body text,
    PRIMARY KEY(id));
"""
        session.execute(stmt)

    def downgrade(self, session):
        session.execute("DROP TABLE post;")
```

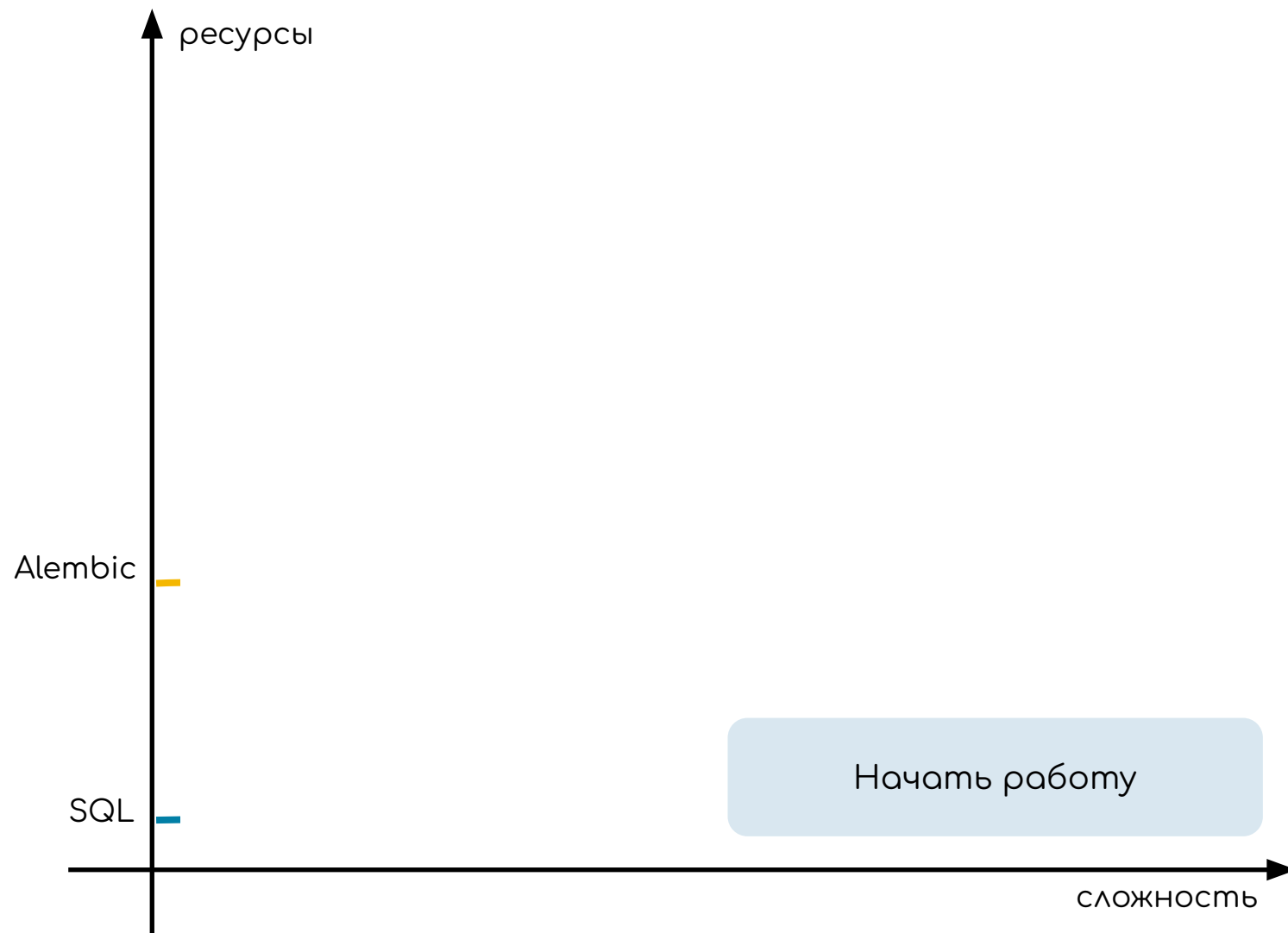


✓ Автоматизация



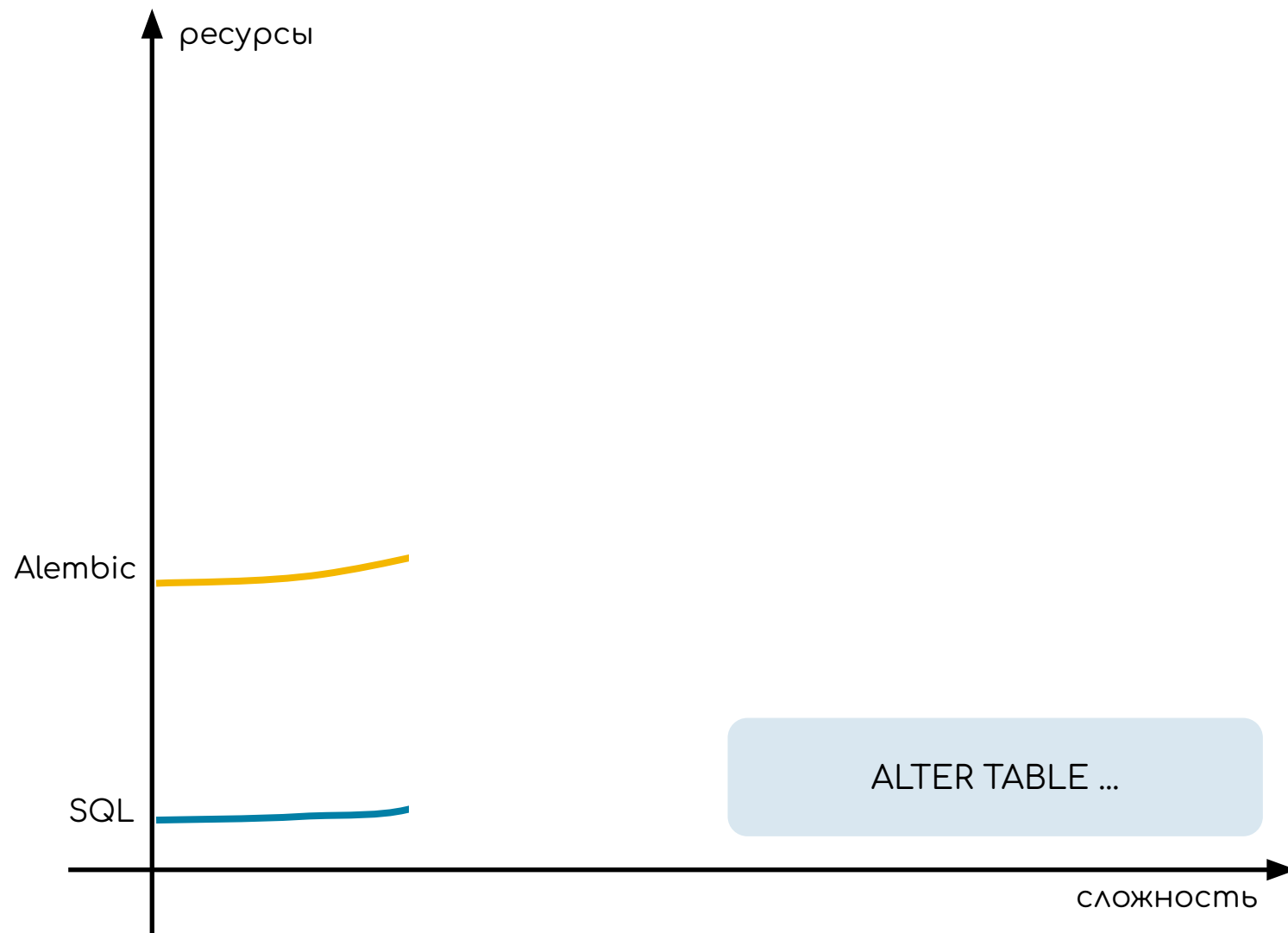
✓ Автоматизация

✗ Время на реализацию  $\Delta t$

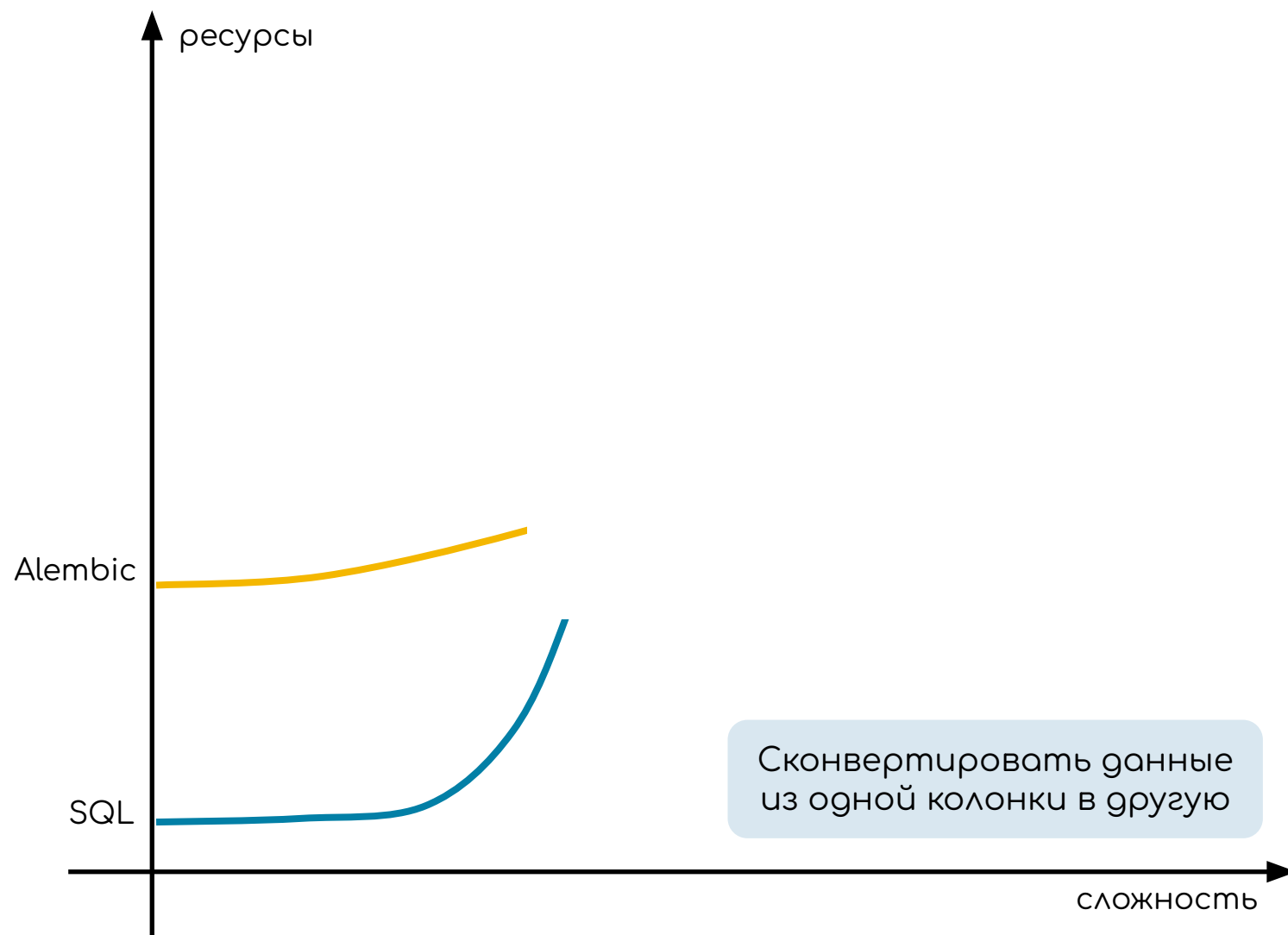


✓ Автоматизация

✗ Время на реализацию  $\Delta t$



- ✓ Автоматизация
- ✗ Время на реализацию  $\Delta t$



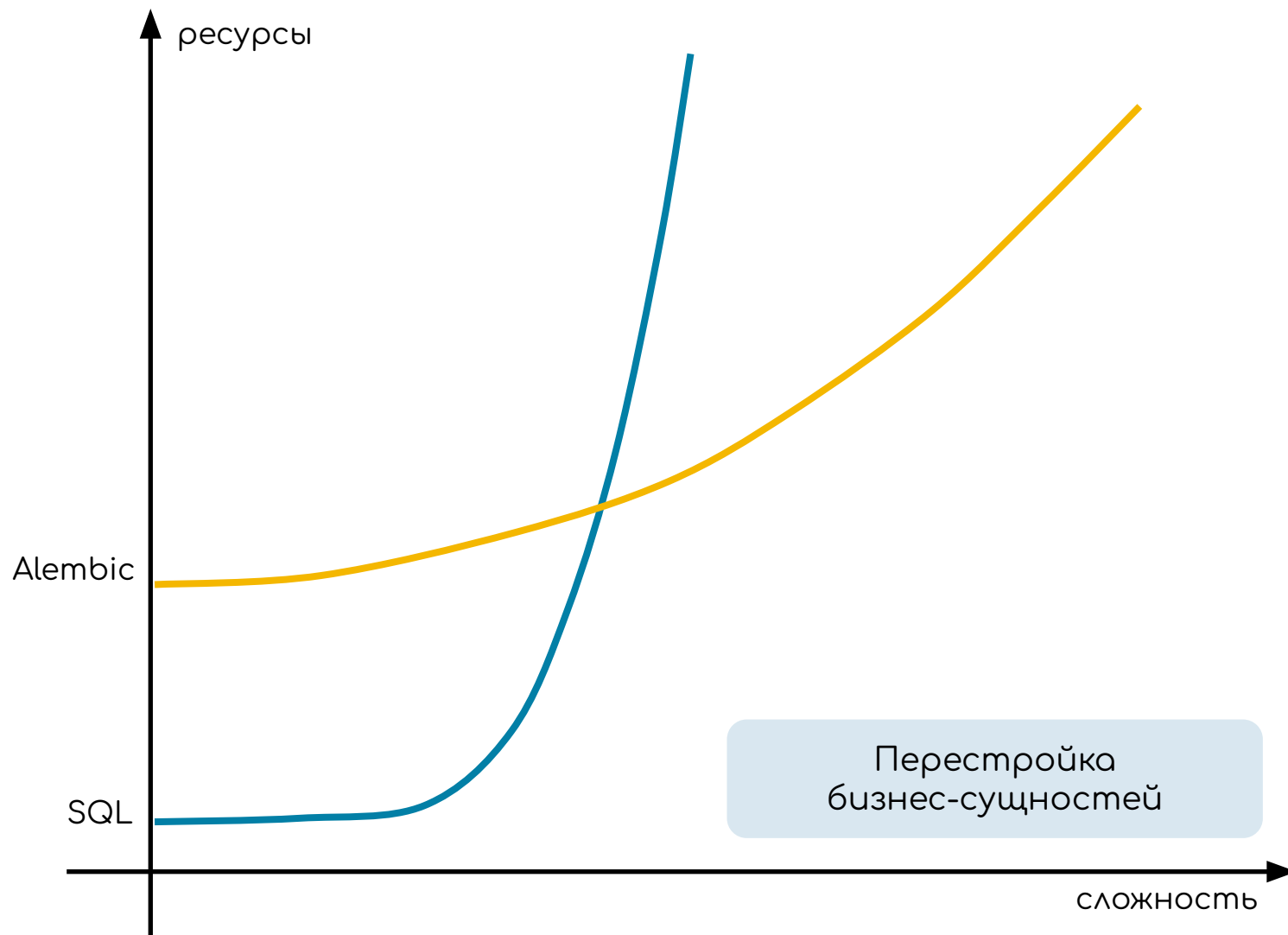


- ✓ Автоматизация
- ✗ Время на реализацию  $\Delta t$



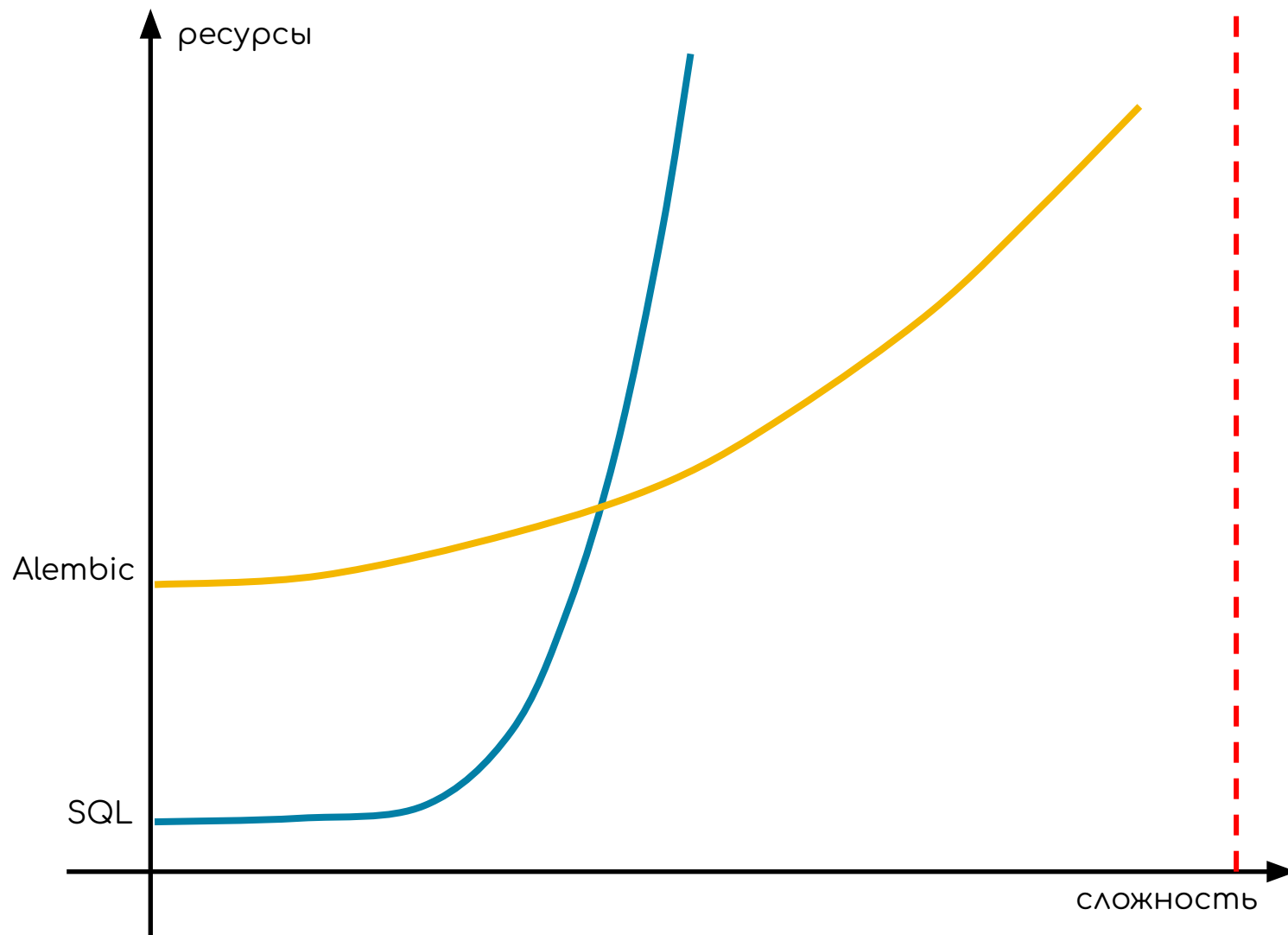
✓ Автоматизация

✗ Время на реализацию  $\Delta t$



- ✓ Автоматизация
- ✗ Время на реализацию  $\Delta t$

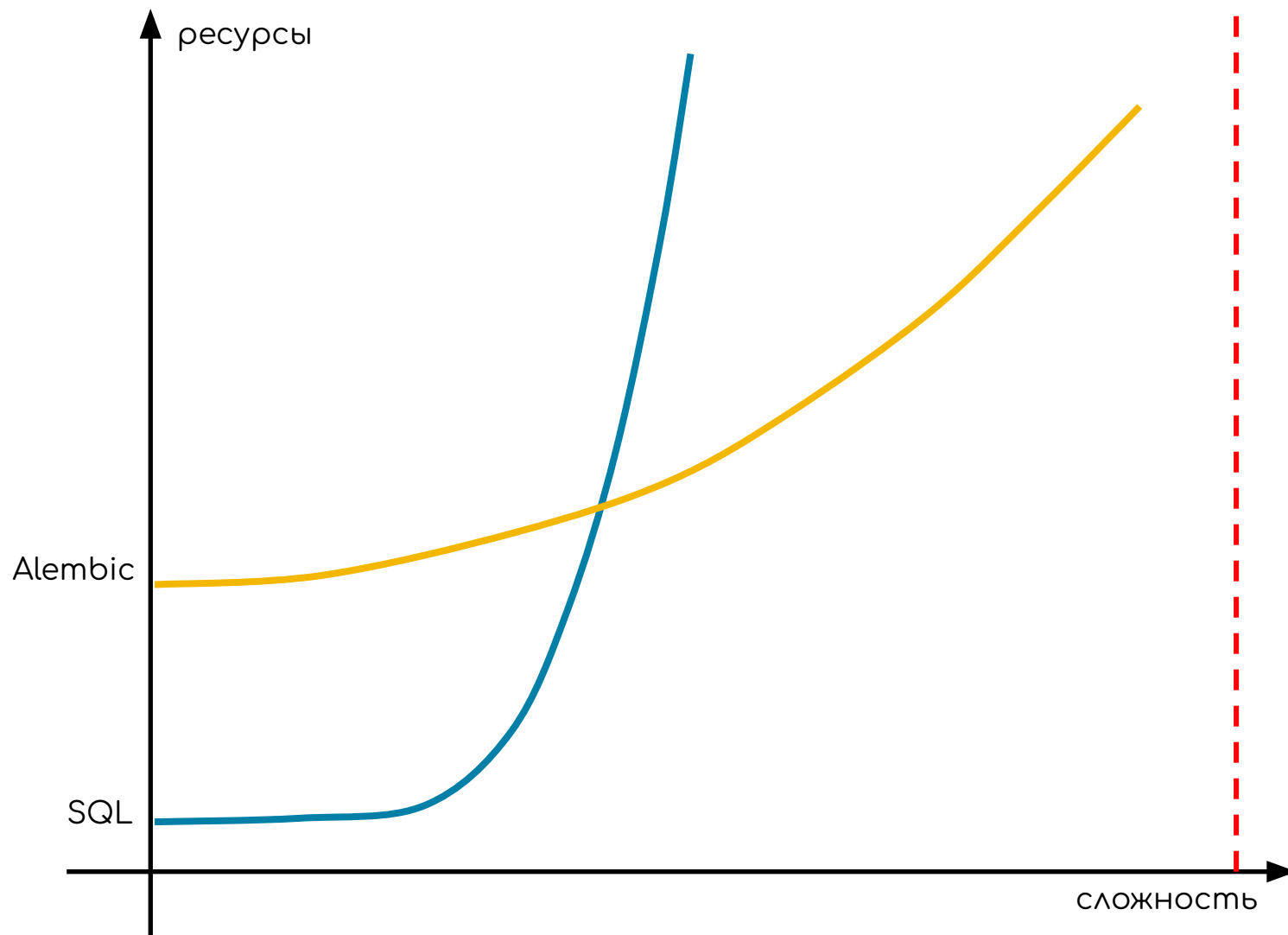
- ✗ Требования и ограничения
- ✗ Экспертиза
- ✗ Boilerplate code



✓ Автоматизация  
✗ Время на реализацию  $\Delta t$

✗ Требования и ограничения  
✗ Экспертиза  
✗ Boilerplate code

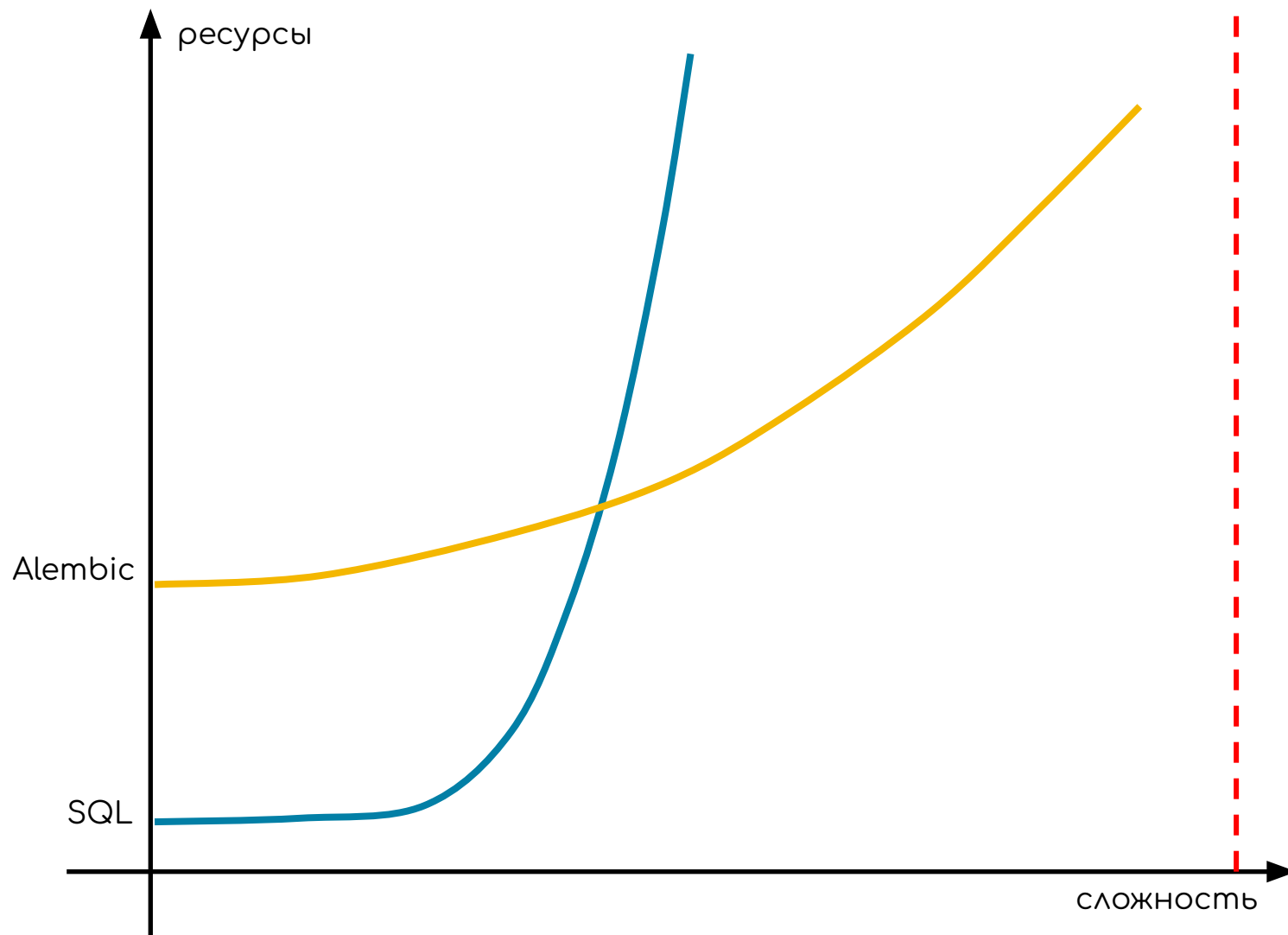
✗ Контроль за процессом  
✗ Управление этапами миграции  
✗ Понимание работы решения



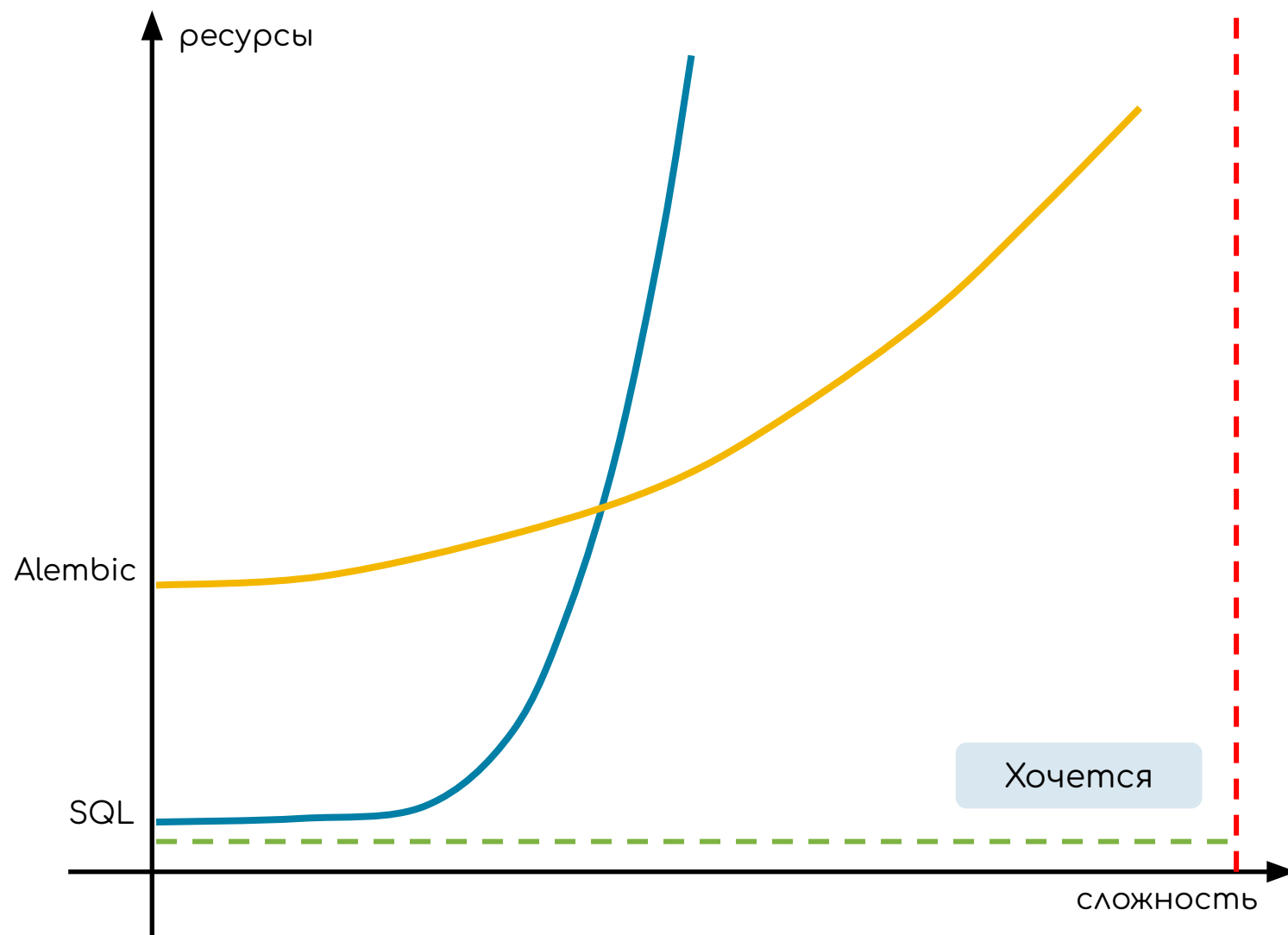
- ✓ Автоматизация
- ✗ Время на реализацию  $\Delta t$
- ✗ Сложность реализации
- ✗ Риски

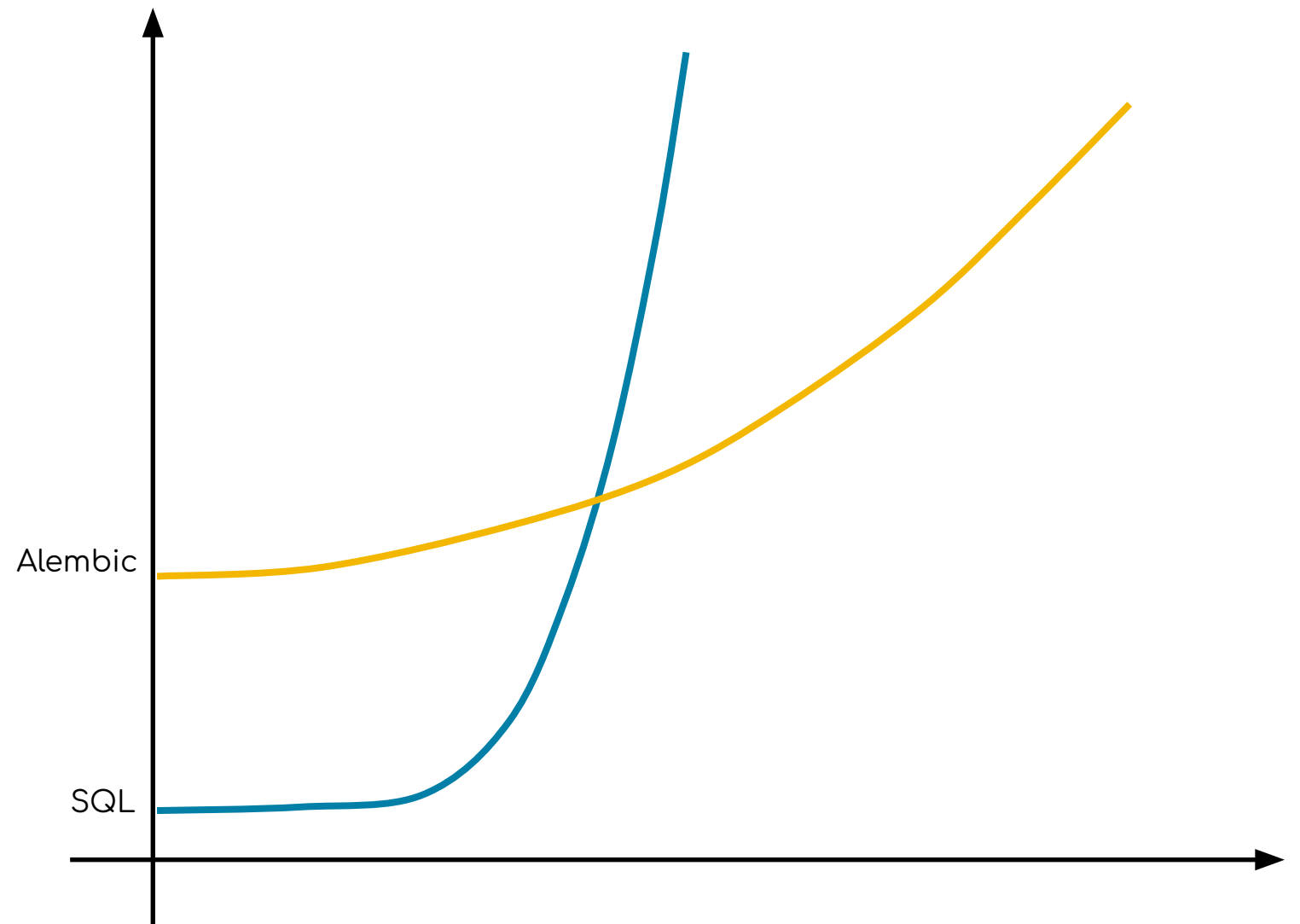
- ✗ Требования и ограничения
- ✗ Экспертиза
- ✗ Boilerplate code

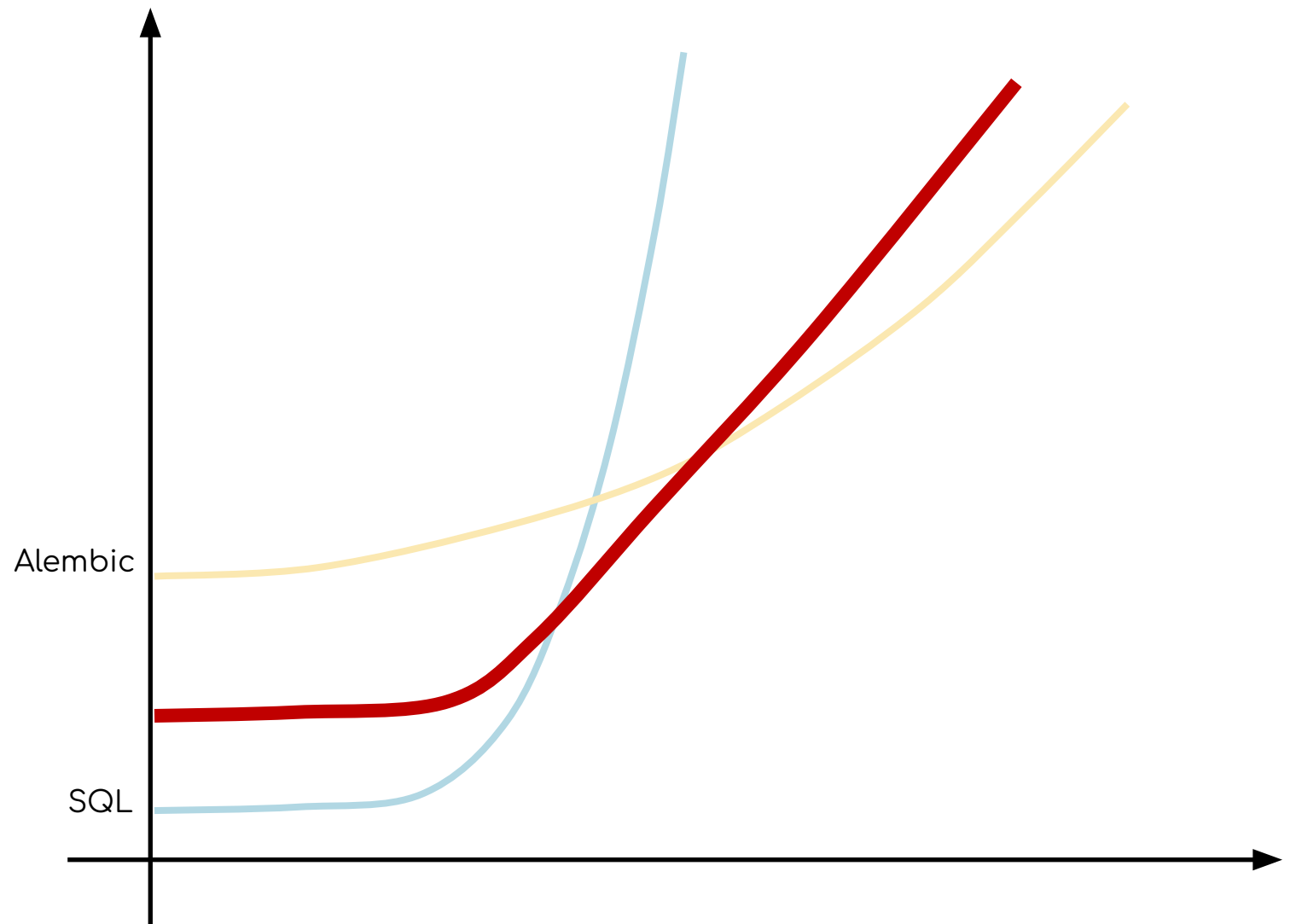
- ✗ Контроль за процессом
- ✗ Управление этапами миграции
- ✗ Понимание работы решения



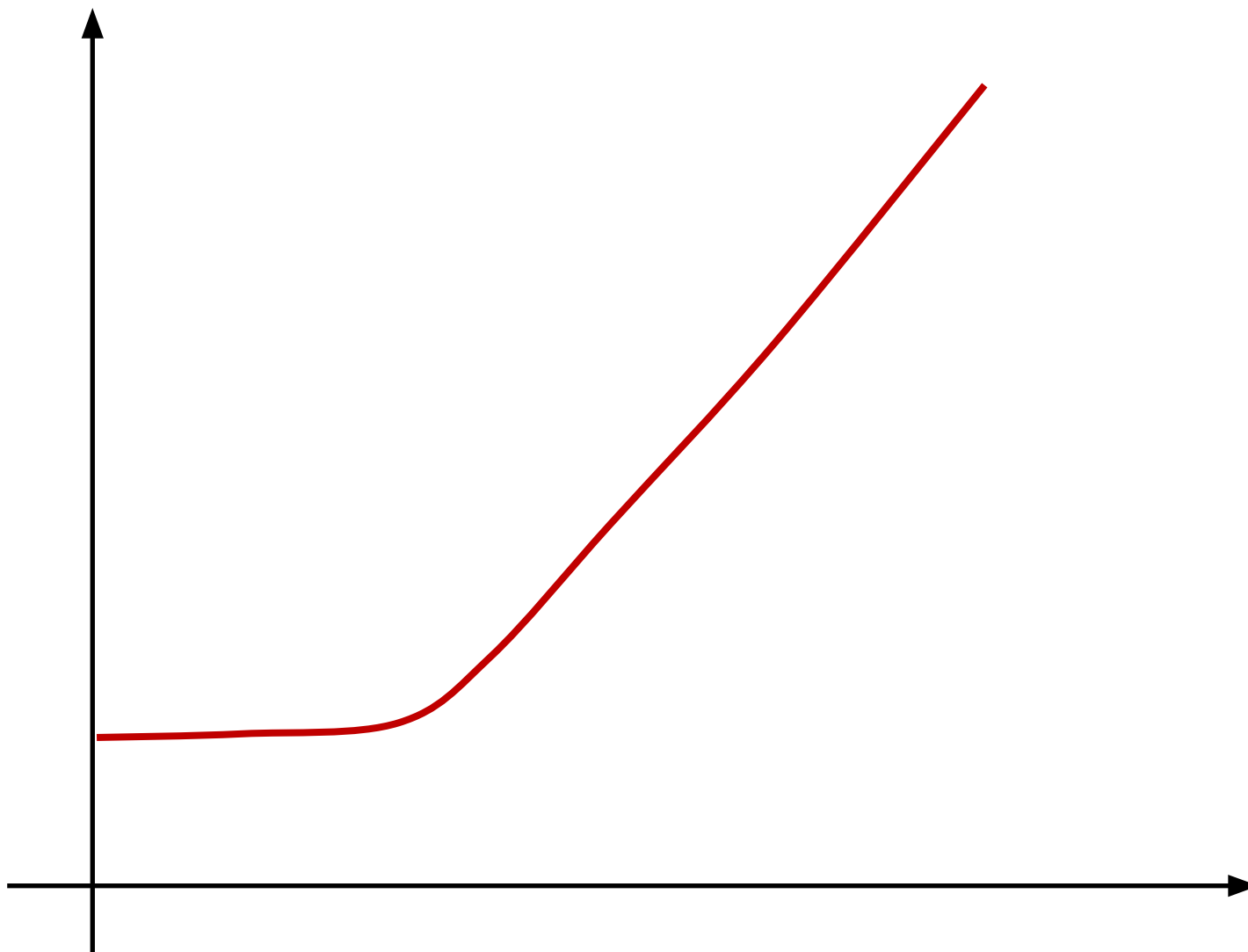
- ✓ Автоматизация
- ✗ Время на реализацию  $\Delta t$
- ✗ Сложность реализации
- ✗ Риски

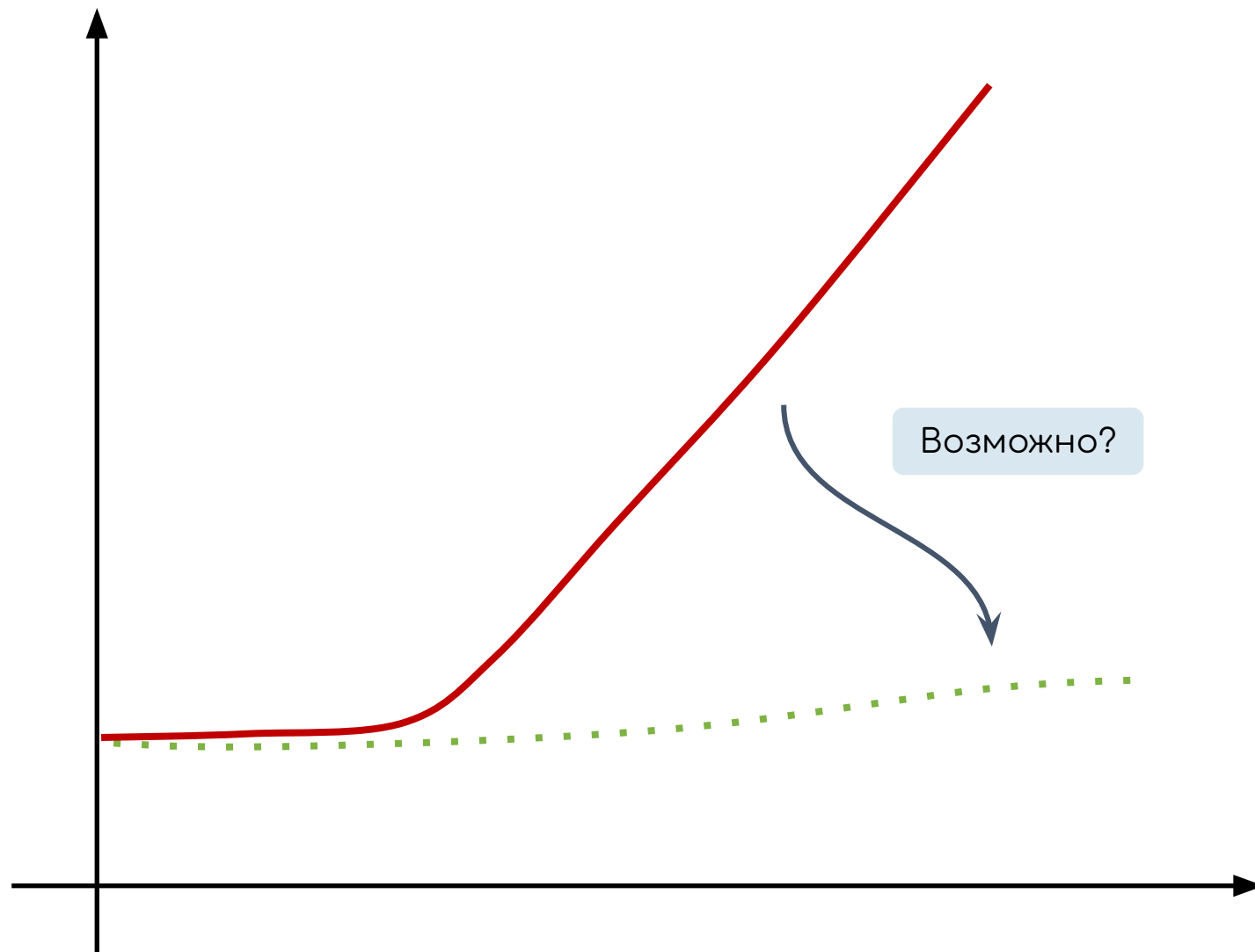














# Трудности

Нельзя тормозить разработку других фич



# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go



# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)



# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

**Нехватка времени и ресурсов**



# Трудности

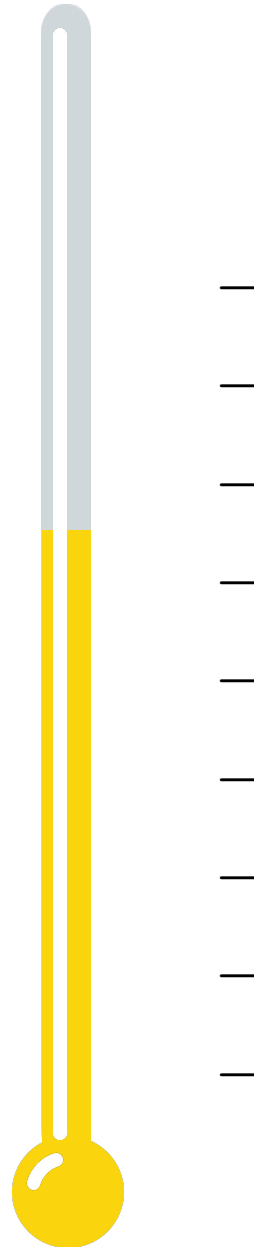
Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей





# Трудности

Нельзя тормозить разработку других фич

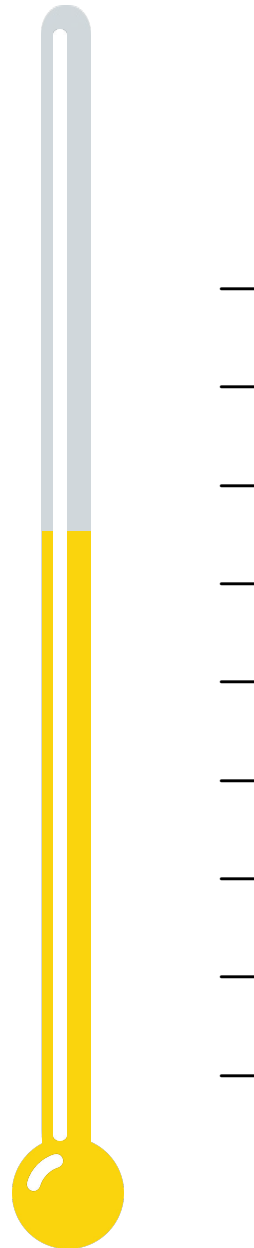
Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей

**Задача: сделать сложную миграцию БД**



# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей

Задача: сделать сложную миграцию БД

**Ограничения инструментария**



# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей

Задача: сделать сложную миграцию БД

Ограничения инструментария

**Без права на ошибку (прозрачность, надежность)**



# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей

Задача: сделать сложную миграцию БД

Ограничения инструментария

Без права на ошибку (*прозрачность, надежность*)

**Миграция – это всегда сложность и риск**



—

—

—

—

—

—

—

—

—

# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей

Задача: сделать сложную миграцию БД

Ограничения инструментария

Без права на ошибку (*прозрачность, надежность*)

Миграция – это всегда сложность и риск

# Решения

—

—

—

—

—

—

—

—

—



# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей

Задача: сделать сложную миграцию БД

Ограничения инструментария

Без права на ошибку (*прозрачность, надежность*)

**Миграция – это всегда сложность и риск**

# Решения

— Best Practices и понимание

—

—

—

—

—

—

—

—



database migrations best practices

Видео

Картинки

Database

Новости

Покупки

Результатов: примерно 3 230 000 (0,34 сек.)

#### fast fact:

- Determine the project's scope. ...
- Ensure that the migration plan is compatible with existin
- Establish a migration time frame. ...
- Validate and test data post-migration. ...
- Audit and document every step of the process.



Iron Mountain

<https://www.ironmountain.com/general-articles/data-...>

## Data Migration Best Practices: Get the Formats Right

database change management best practices



Картинки

Видео

Покупки

Новости

Книги

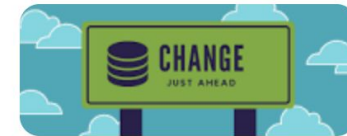
Карты

Авиабилеты

Результатов: примерно 486 000 000 (0,33 сек.)

### Top 10 Database Change Management Best Practices

- #1: Use a version control tool for all database changes.
- #2 Automate database deployments.
- #3 Implement a rollback plan.
- #4 Keep track of all changes made to the database.
- #5 Test all changes in a non-production environment before deploying to production.



Ещё • 22 янв. 2023 г.



DBmaestro

<https://www.dbmaestro.com> › blog › database-automation

### Top 10 Database Change Management Best Practices

## *Database Change Management*



### *Schema*

*структурное и контролируемое  
вековое*



## *Database Change Management*



### *Schema*

*структурное и контролируемое  
вековое*



### *Data*

*логическое и ситуативное  
эфемерное*

## Database Change Management



### *Schema*

*структурное и контролируемое  
вековое*



### *Data*

*логическое и ситуативное  
эфемерное*



### *Code*

*изменяемые требования  
направляющее*

## Migration



### **Schema**

*структурное и контролируемое  
вековое*

### **Data**

*логическое и ситуативное  
эфемерное*

### **Code**

*изменяемые требования  
направляющее*

## Migration



*внутренняя реорганизация  
данных  
и способов их хранения*

## *Migration*



*консистентная*

*внутренняя реорганизация  
данных  
и способов их хранения*

## *Migration*



*консистентная*

*атомарность*

*обратимость*

## Migration



консистентная

~~атомарность  
обратимость~~



## Migration



*консистентная*

~~атемарность  
обратимость~~

*сохранение сути  
упорядоченность*





## Best Practices



### Safe Database Migration Pattern Without Downtime

*Поддержать в коде приложения  
одновременную работу  
с несколькими*

- источниками данных*
- версиями схем данных*
- отличными копиями данных*

## Best Practices



## Общие

- храните в VCS (ex. Git)
- применяйте инкрементально
- используйте “инструменты вместо рук”
- создавайте план миграции
- информируйте участников

database migrations best practices

Видео

Картинки

Database

Новости

Покупки

Результатов: примерно 3 230 000 (0,34 сек.)

#### fast fact:

- Determine the project's scope. ...
- Ensure that the migration plan is compatible with existin
- Establish a migration time frame. ...
- Validate and test data post-migration. ...
- Audit and document every step of the process.



Iron Mountain

<https://www.ironmountain.com/general-articles/data-...>

## Data Migration Best Practices: Get the Formats Right

database change management best practices



Картинки

Видео

Покупки

Новости

Книги

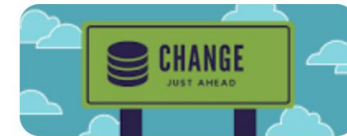
Карты

Авиабилеты

Результатов: примерно 486 000 000 (0,33 сек.)

### Top 10 Database Change Management Best Practices

- #1: Use a version control tool for all database changes.
- #2 Automate database deployments.
- #3 Implement a rollback plan.
- #4 Keep track of all changes made to the database.
- #5 Test all changes in a non-production environment before deploying to production.



Ещё • 22 янв. 2023 г.



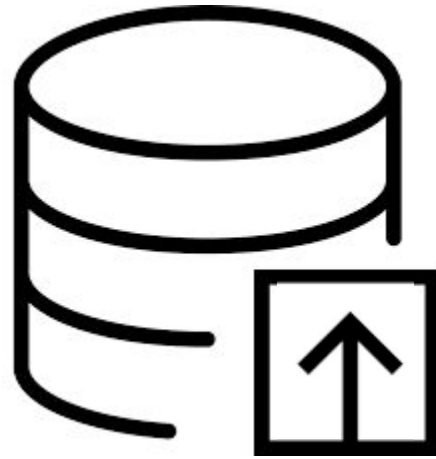
DBmaestro

<https://www.dbmaestro.com> › blog › database-automation

### Top 10 Database Change Management Best Practices



*делайте BACKUP !!!*



*проверяйте восстановление  
из backup-а*

# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей

Задача: сделать сложную миграцию БД

Ограничения инструментария

Без права на ошибку (*прозрачность, надежность*)

**Миграция – это всегда сложность и риск**

# Решения

— Best Practices и понимание

—

—

—

—

—

—

—

—



# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей

Задача: сделать сложную миграцию БД

Ограничения инструментария

**Без права на ошибку (прозрачность, надежность)**

Миграция – это всегда сложность и риск

# Решения

— Best Practices и понимание

— **План миграции и тестирование**

—

—

—

—

—

—

—



## *План миграции*

- *план тестирования*
- *план подготовки (+backup!)*
- *план применения*
- *план отката*
- *план восстановления*
- *план коммуникаций*



## Тестирование

### *план*

- *тестирования*
- *действий*

### *prod-like окружение*

- *сервис*
- *пользовательская нагрузка*

### *prod-like база*

- *объём*
- *реальный профиль данных*

### *prod-исполнитель*

# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей

Задача: сделать сложную миграцию БД

Ограничения инструментария

**Без права на ошибку (прозрачность, надежность)**

Миграция – это всегда сложность и риск

# Решения

— Best Practices и понимание

— **План миграции и тестирование**

—

—

—

—

—

—

—



# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей

Задача: сделать сложную миграцию БД

## Ограничения инструментария

Без права на ошибку (*прозрачность, надежность*)

Миграция – это всегда сложность и риск

# Решения

— Best Practices и понимание

— План миграции и тестирование

— **Подбор инструментария и реализация**

—

—

—

—

—

—



## MIGRATION

SCHEMA

DATA

MIGRATION

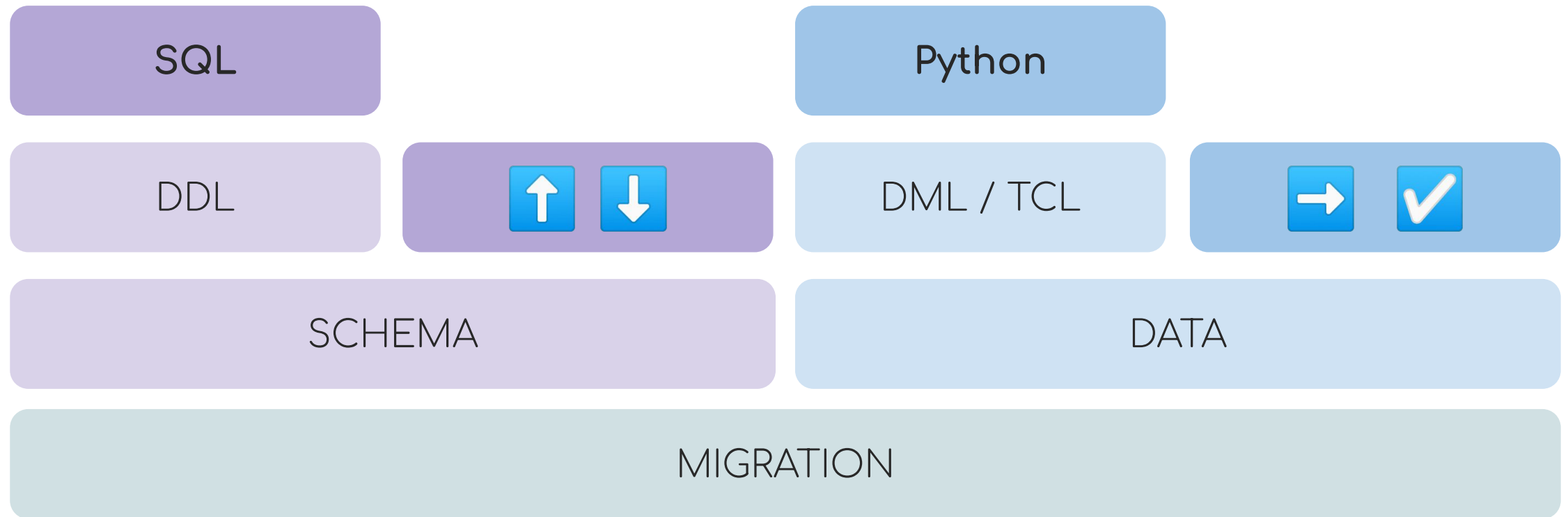
DDL

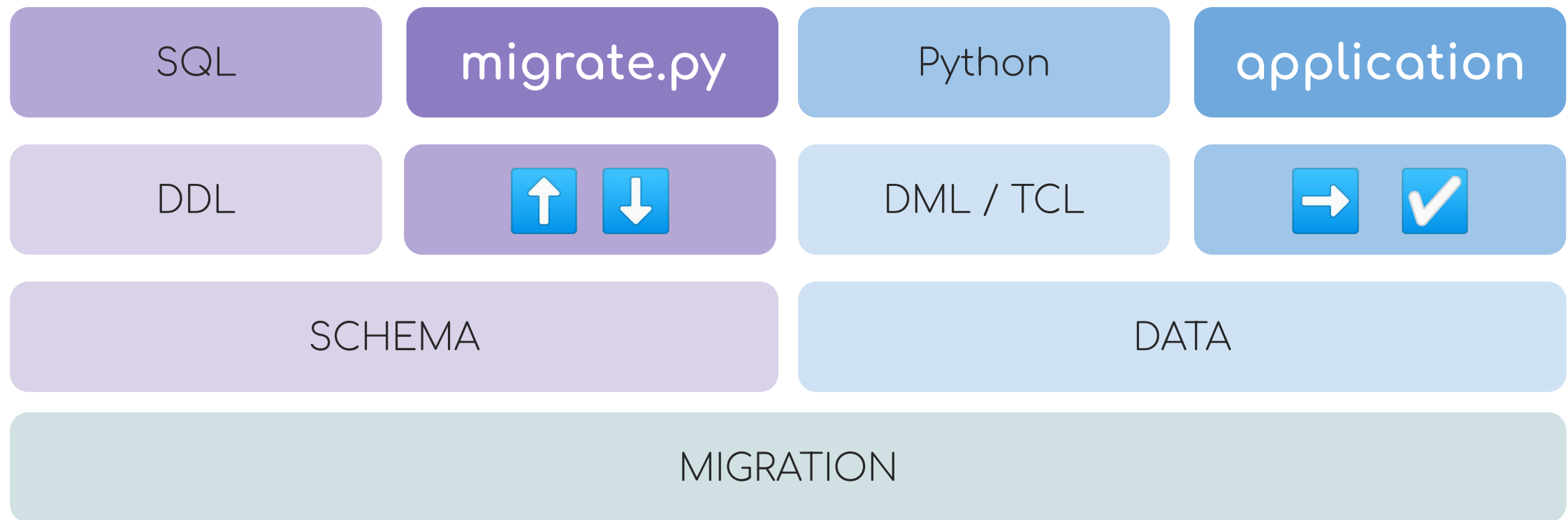
DML / TCL

SCHEMA

DATA

MIGRATION







# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей

Задача: сделать сложную миграцию БД

## Ограничения инструментария

Без права на ошибку (*прозрачность, надежность*)

Миграция – это всегда сложность и риск

# Решения

— Best Practices и понимание

— План миграции и тестирование

— **Подбор инструментария и реализация**

—

—

—

—

—

—



# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей

**Задача: сделать сложную миграцию БД**

Ограничения инструментария

Без права на ошибку (*прозрачность, надежность*)

Миграция – это всегда сложность и риск

# Решения

— Best Practices и понимание

— План миграции и тестирование

— Подбор инструментария и реализация

— **Задача: сделать несложный сервис**

—

—

—

—

—





# БД: миграция **как микросервис**

*История одной миграции*

# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

**Сложное изменение моделей сущностей**

Задача: сделать сложную миграцию БД

Ограничения инструментария

Без права на ошибку (*прозрачность, надежность*)

Миграция – это всегда сложность и риск

# Решения

— Best Practices и понимание

— План миграции и тестирование

— Подбор инструментария и реализация

— Задача: сделать несложный сервис

— **Основная кодовая база**

—

—

—

—



# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

## Нехватка времени и ресурсов

Сложное изменение моделей сущностей

Задача: сделать сложную миграцию БД

Ограничения инструментария

Без права на ошибку (*прозрачность, надежность*)

Миграция – это всегда сложность и риск

# Решения

— Best Practices и понимание

— План миграции и тестирование

— Подбор инструментария и реализация

— Задача: сделать несложный сервис

— Основная кодовая база

— **Готовые наработки для создания микросервисов**

—


—

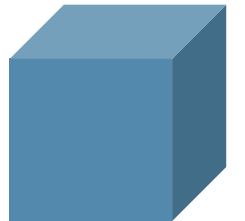
—



**Garson**

*Library that serves*

Библиотека для создания **daemon**-ов 



## Garson

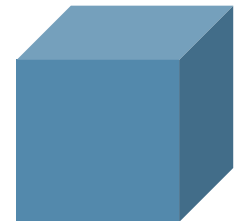
*Library that serves*

Библиотека для создания **daemon**-ов



предоставляет

- **entrypoint** приложения
- runtime ( [python-daemon](#) )
- абстракцию *Service*
- **main loop** *service*



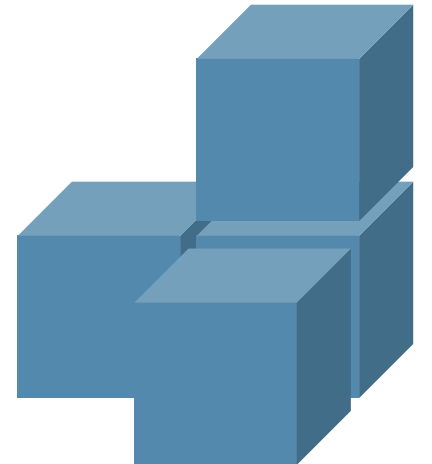
*Garson*

~~Framework~~

Extensible

Integrable

Manageable





*Garson*

~~Framework~~

Extensible

Integrable

Manageable

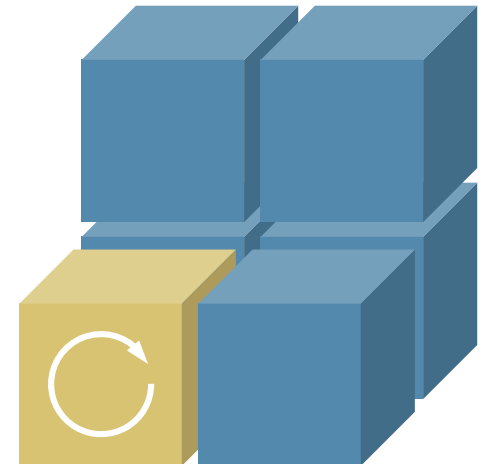
Batteries included



**Garson**

*Main Loop Service*

Iteration

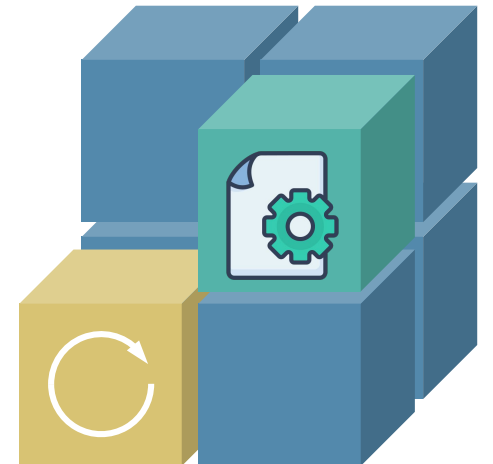


**Garson**

*Main Loop Service*

Iteration

Configuration



# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

## Нехватка времени и ресурсов

Сложное изменение моделей сущностей

Задача: сделать сложную миграцию БД

Ограничения инструментария

Без права на ошибку (*прозрачность, надежность*)

Миграция – это всегда сложность и риск

# Решения

— Best Practices и понимание

— План миграции и тестирование

— Подбор инструментария и реализация

— Задача: сделать несложный сервис

— Основная кодовая база

— **Готовые наработки для создания микросервисов**

—

—

—



# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

## Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей

Задача: сделать сложную миграцию БД

Ограничения инструментария

Без права на ошибку (*прозрачность, надежность*)

Миграция – это всегда сложность и риск

# Решения

— Best Practices и понимание

— План миграции и тестирование

— Подбор инструментария и реализация

— Задача: сделать несложный сервис

— Основная кодовая база

— Готовые наработки для создания микросервисов

— **Шаблон микросервиса для БД миграции**

—

—

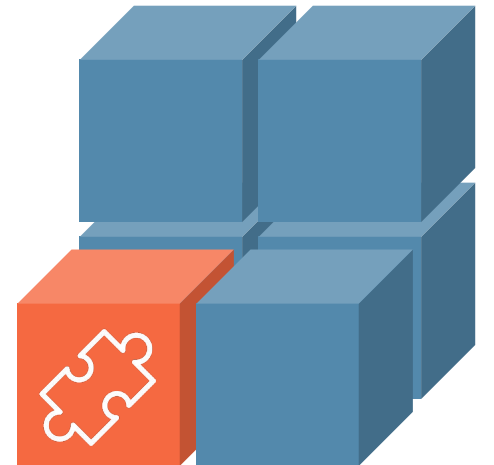


# *Migrator*

*Garson-based*

## Main Loop Service **Addon**

- “Panic Mode”
- Context: database, report
- API: complete(), fail()

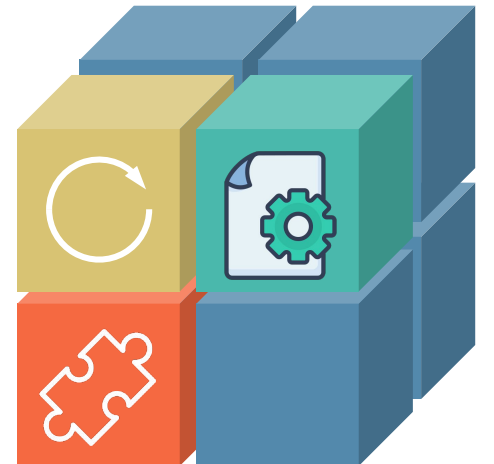


# *Migrator*

*Garson-based*

## Main Loop Service **Addon**

- “Panic Mode”
- Context: database, report
- API: `complete()`, `fail()`



# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

## Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей

Задача: сделать сложную миграцию БД

Ограничения инструментария

Без права на ошибку (*прозрачность, надежность*)

Миграция – это всегда сложность и риск

# Решения

— Best Practices и понимание

— План миграции и тестирование

— Подбор инструментария и реализация

— Задача: сделать несложный сервис

— Основная кодовая база

— Готовые наработки для создания микросервисов

— **Шаблон микросервиса для БД миграции**

—

—





# Трудности

Нельзя тормозить разработку других фич

**Учитывать новичков Junior / Go**

Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей

Задача: сделать сложную миграцию БД

Ограничения инструментария

Без права на ошибку (*прозрачность, надежность*)

Миграция – это всегда сложность и риск

# Решения

— Best Practices и понимание

— План миграции и тестирование

— Подбор инструментария и реализация

— Задача: сделать несложный сервис

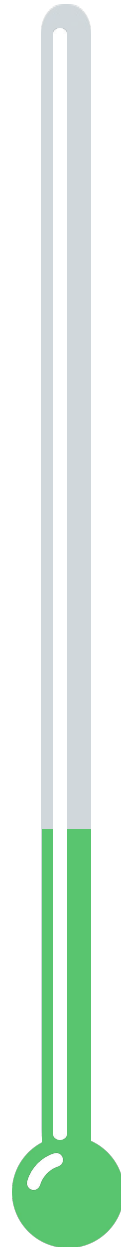
— Основная кодовая база

— Готовые наработки для создания микросервисов

— Шаблон микросервиса для БД миграции

— **Сборка микросервиса из готовых компонентов**

—



# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей

Задача: сделать сложную миграцию БД

Ограничения инструментария

Без права на ошибку (*прозрачность, надежность*)

Миграция – это всегда сложность и риск

# Решения

— Best Practices и понимание

— План миграции и тестирование

— Подбор инструментария и реализация

— Задача: сделать несложный сервис

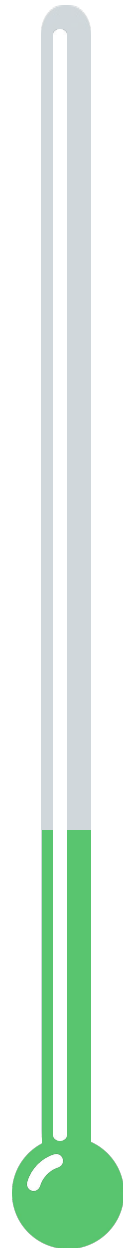
— Основная кодовая база

— Готовые наработки для создания микросервисов

— Шаблон микросервиса для БД миграции

— Сборка микросервиса из готовых компонентов

— **Задача для ~самостоятельной работы**



# Трудности

Нельзя тормозить разработку других фич

Учитывать новичков Junior / Go

Повторяемость решения (типовая задача)

Нехватка времени и ресурсов

Сложное изменение моделей сущностей

Задача: сделать сложную миграцию БД

Ограничения инструментария

Без права на ошибку (*прозрачность, надежность*)

Миграция – это всегда сложность и риск

# Решения

— Best Practices и понимание

— План миграции и тестирование

— Подбор инструментария и реализация

— Задача: сделать несложный сервис

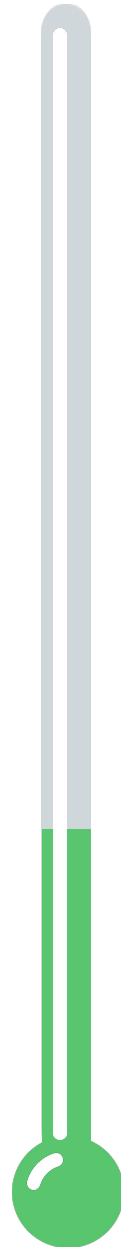
— Основная кодовая база

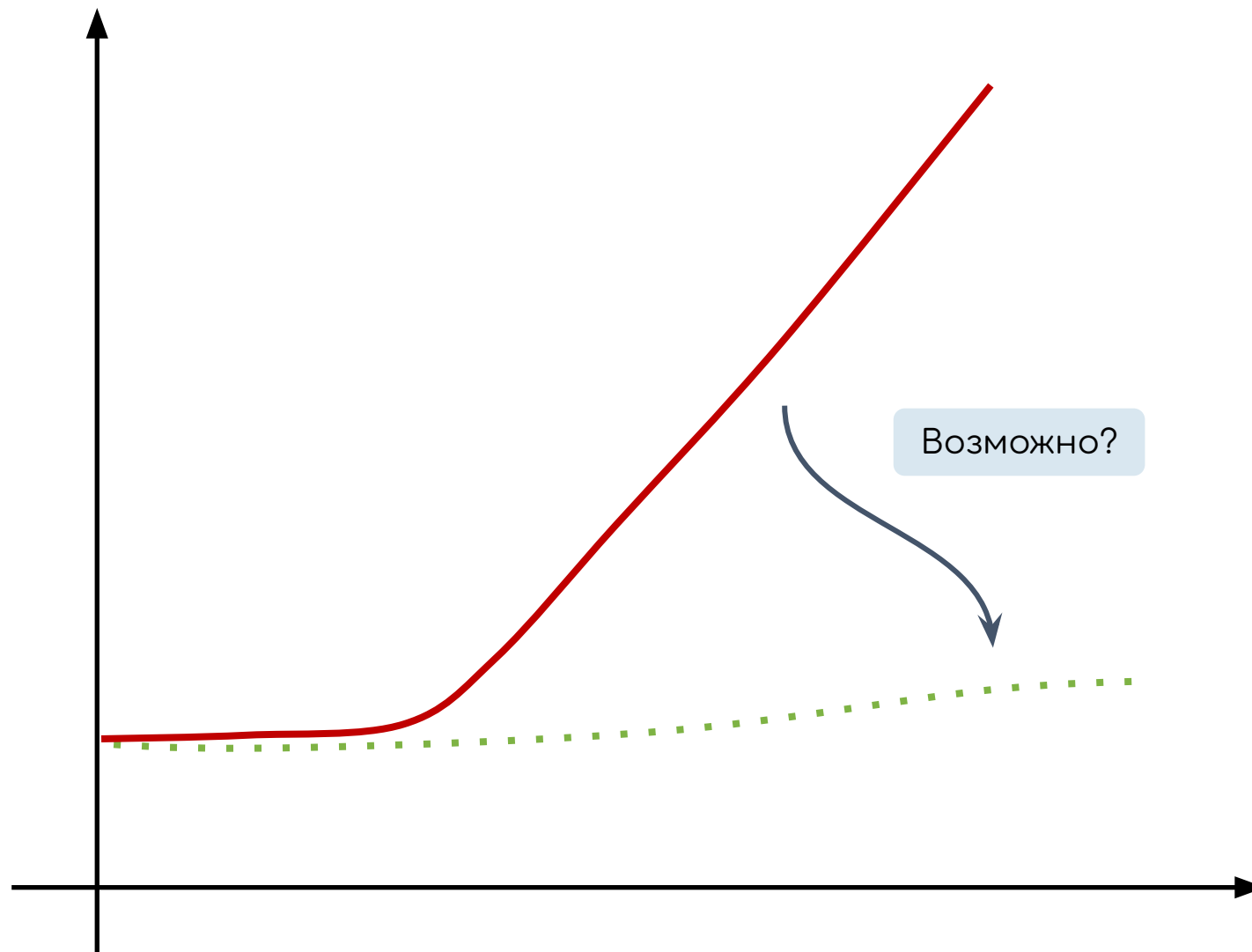
— Готовые наработки для создания микросервисов

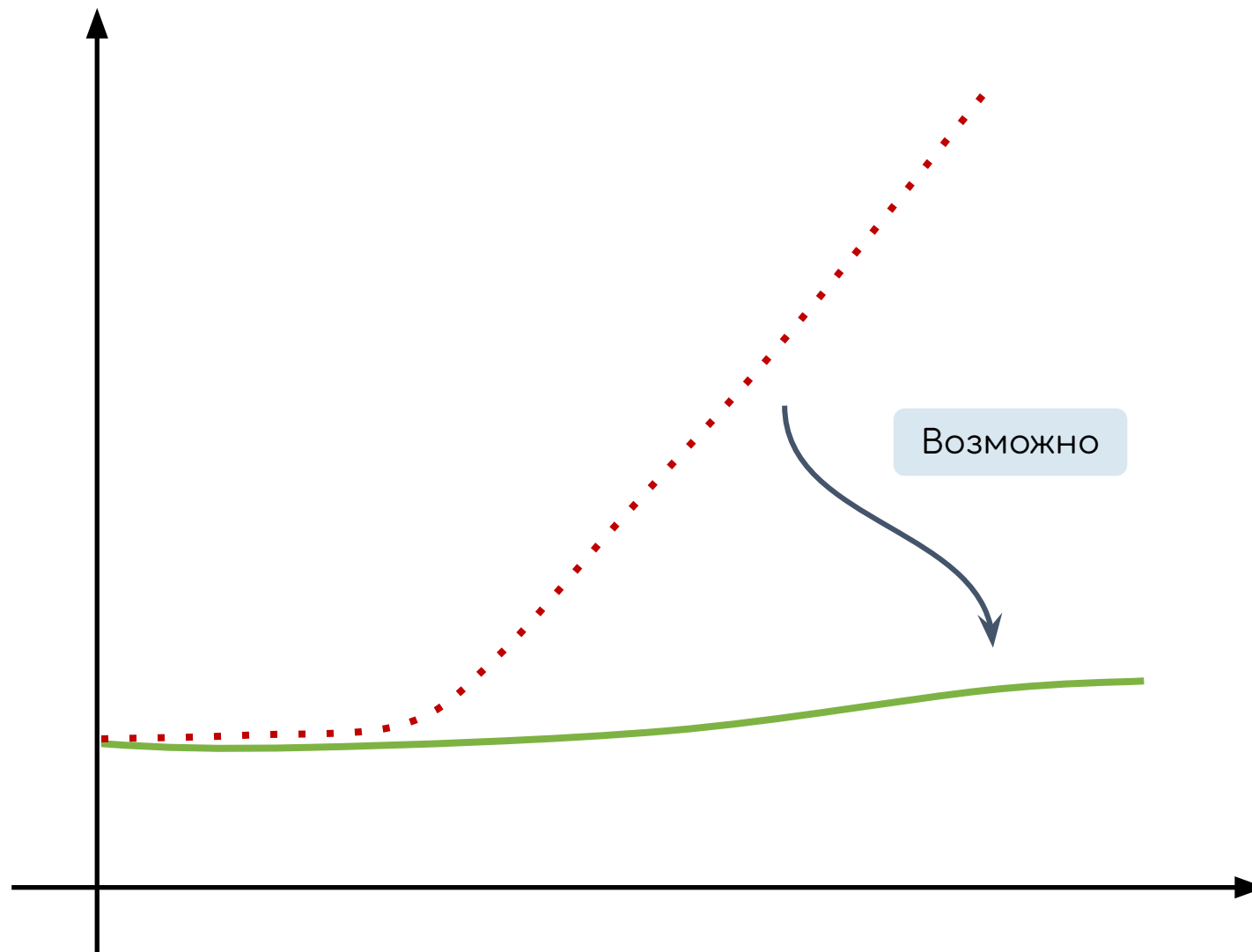
— Шаблон микросервиса для БД миграции

— Сборка микросервиса из готовых компонентов

— Задача для ~самостоятельной работы

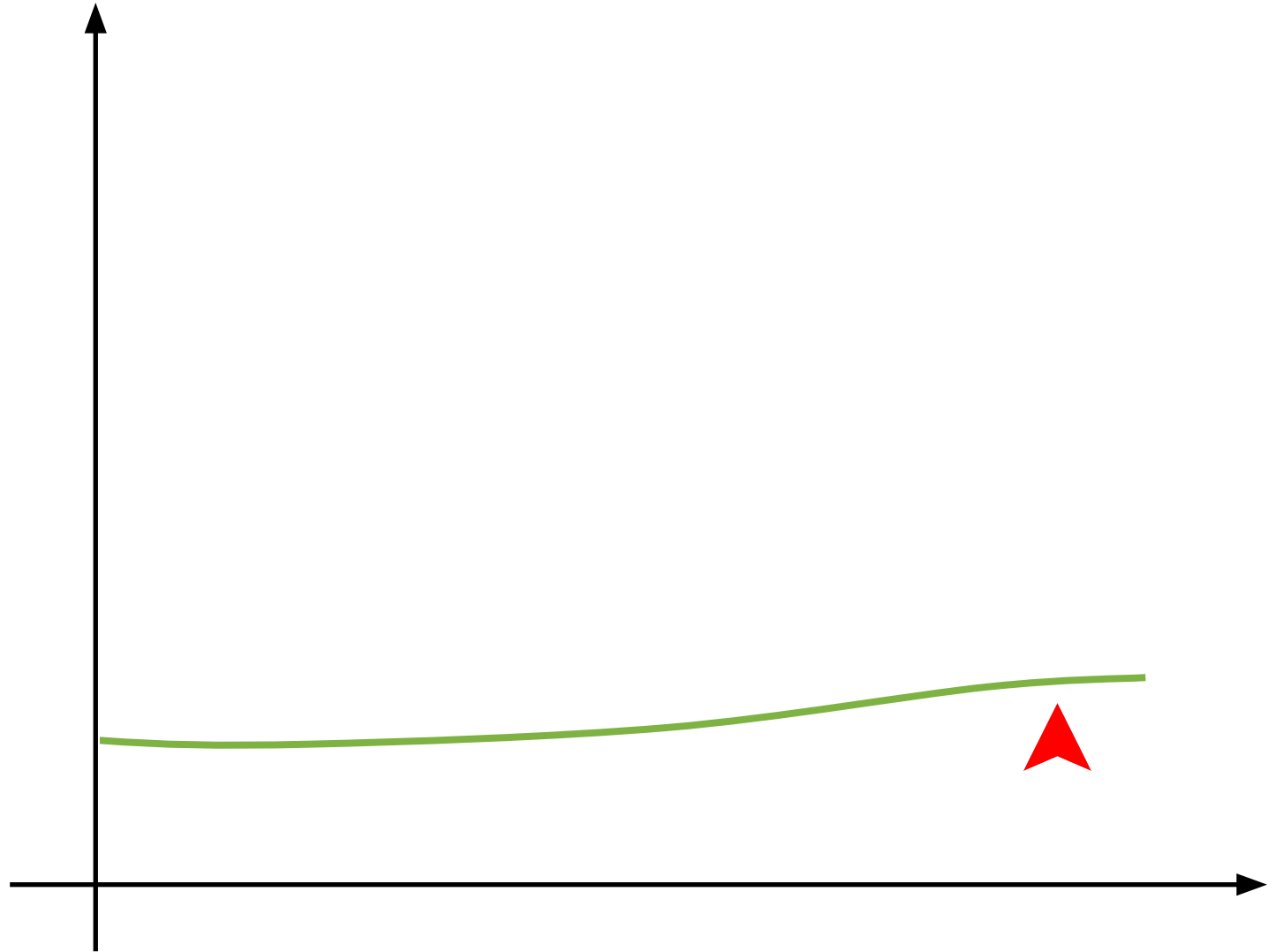






*PROFIT*

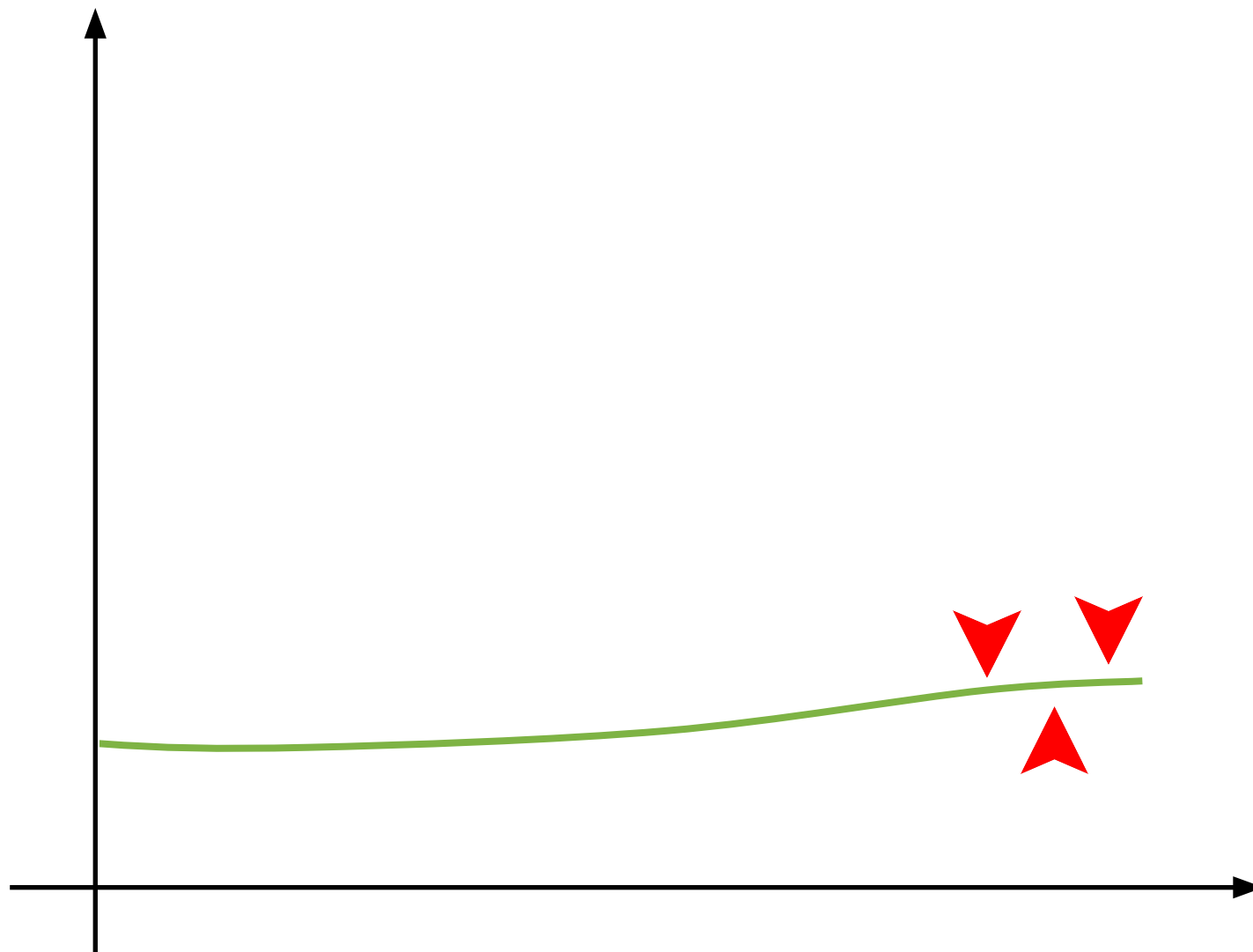
*миграция успешно проведена*



*PROFIT*

*миграция успешно проведена*

*даже 3 миграции*

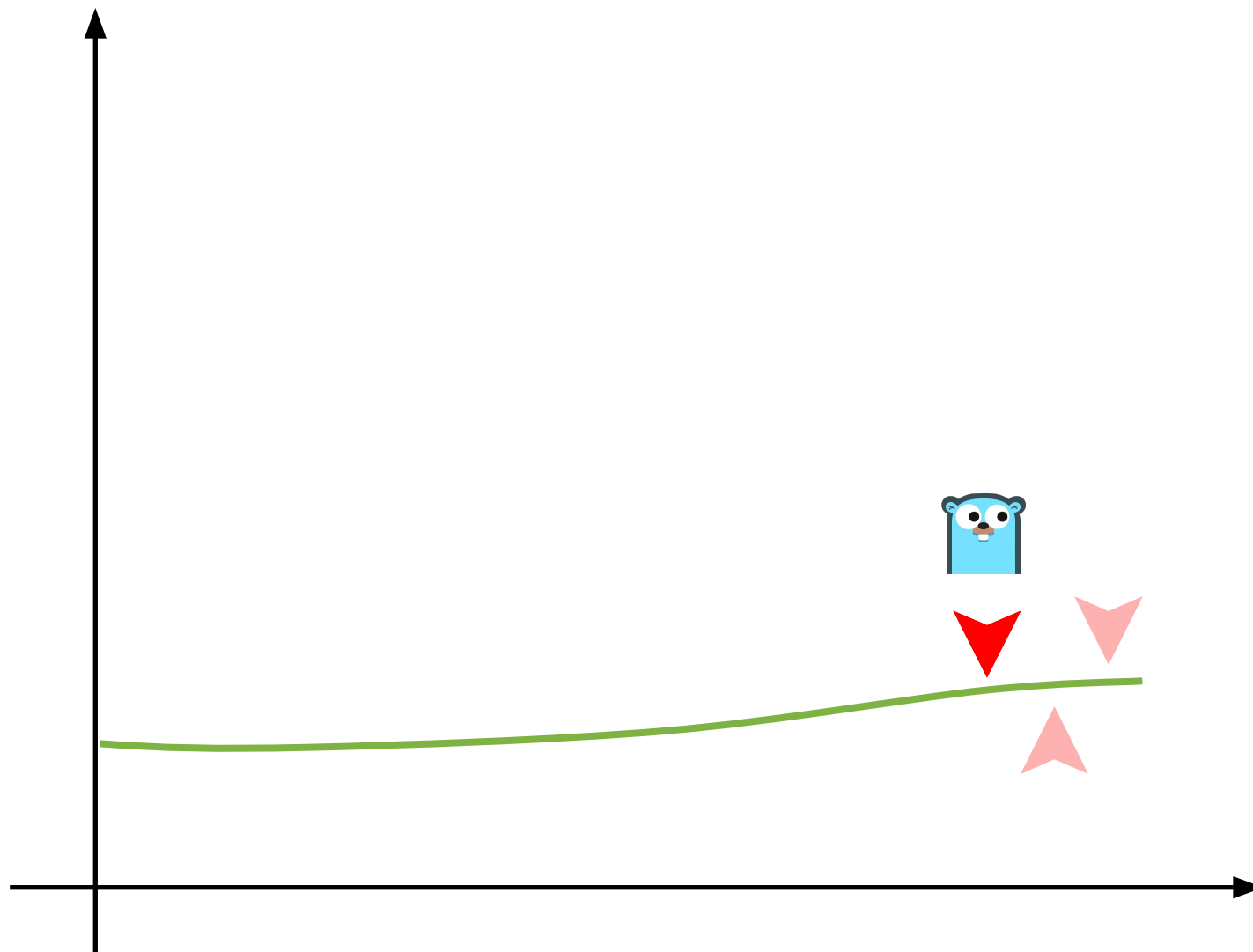


*PROFIT*

*миграция успешно проведена*

*даже 3 миграции*

*Go-шник*



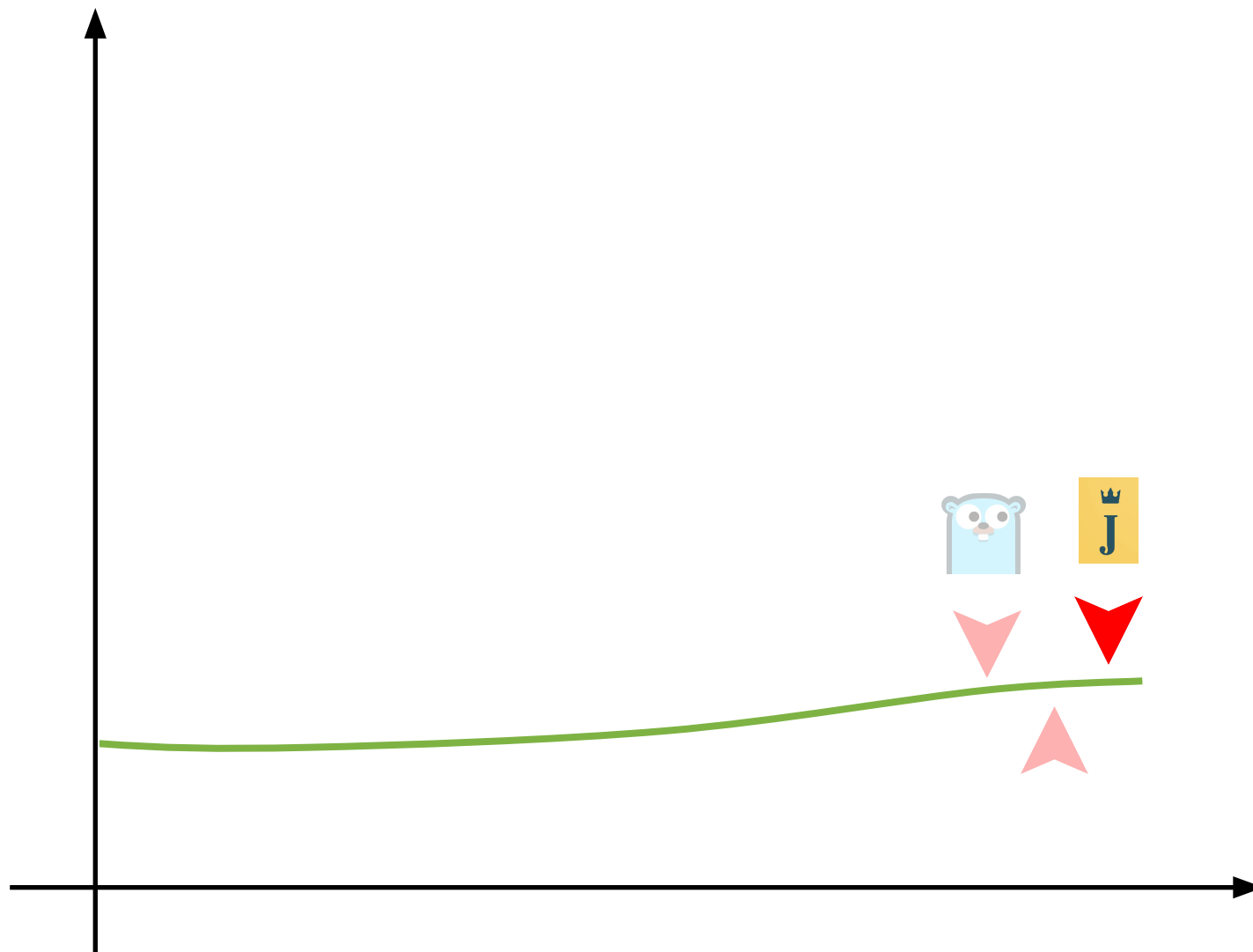


# PROFIT

миграция успешно проведена

даже 3 миграции

Go-шник и **Junior**



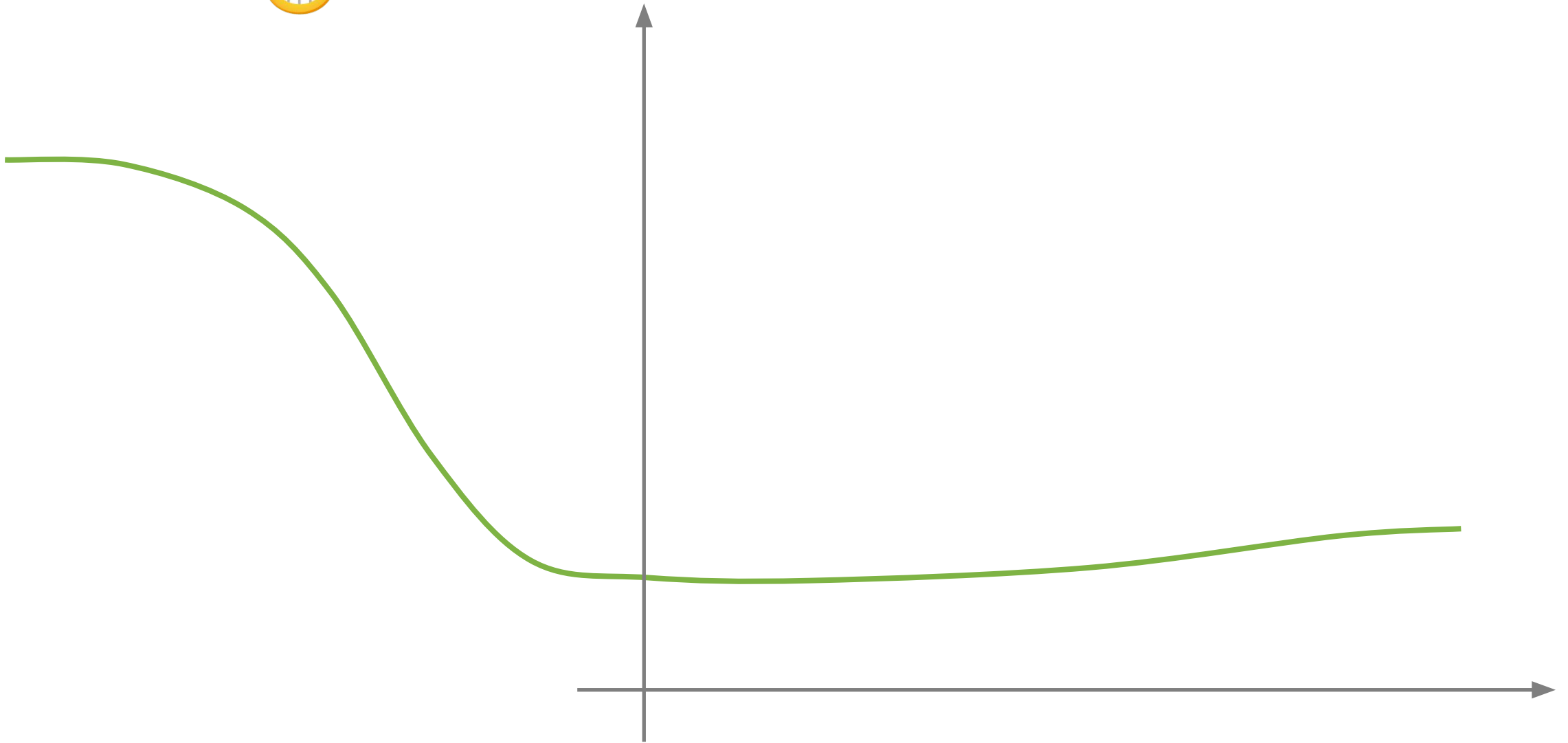
*Миграция – это больше, чем просто техническая задача, это бизнес-проект.  
(со всеми вытекающими требованиями и рисками)*

*Процессы вокруг БД – одни из ключевых в обеспечении стабильности  
и успеха бизнеса.*

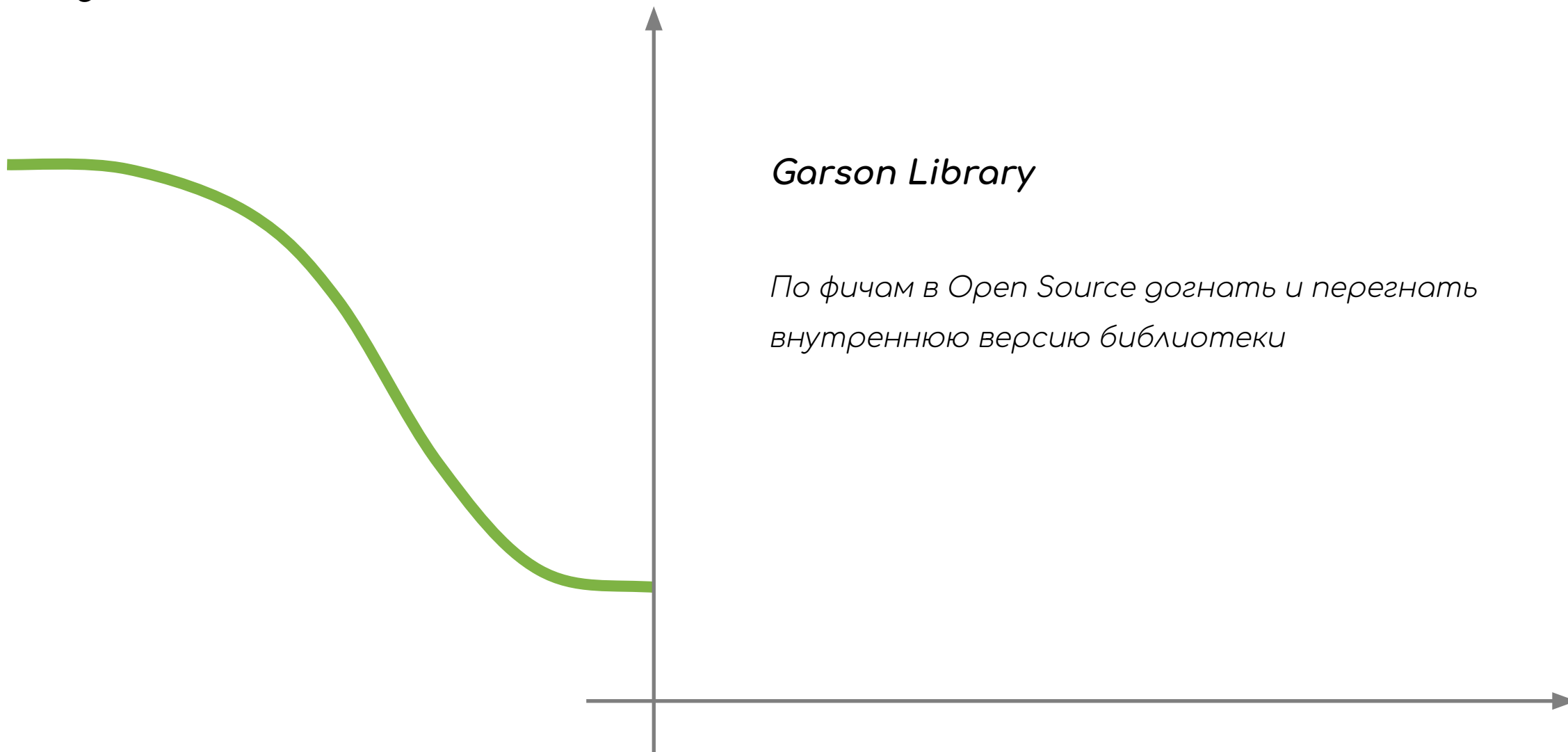
*Хорошая миграция – это проектно организованная миграция.*

*Что дальше?*

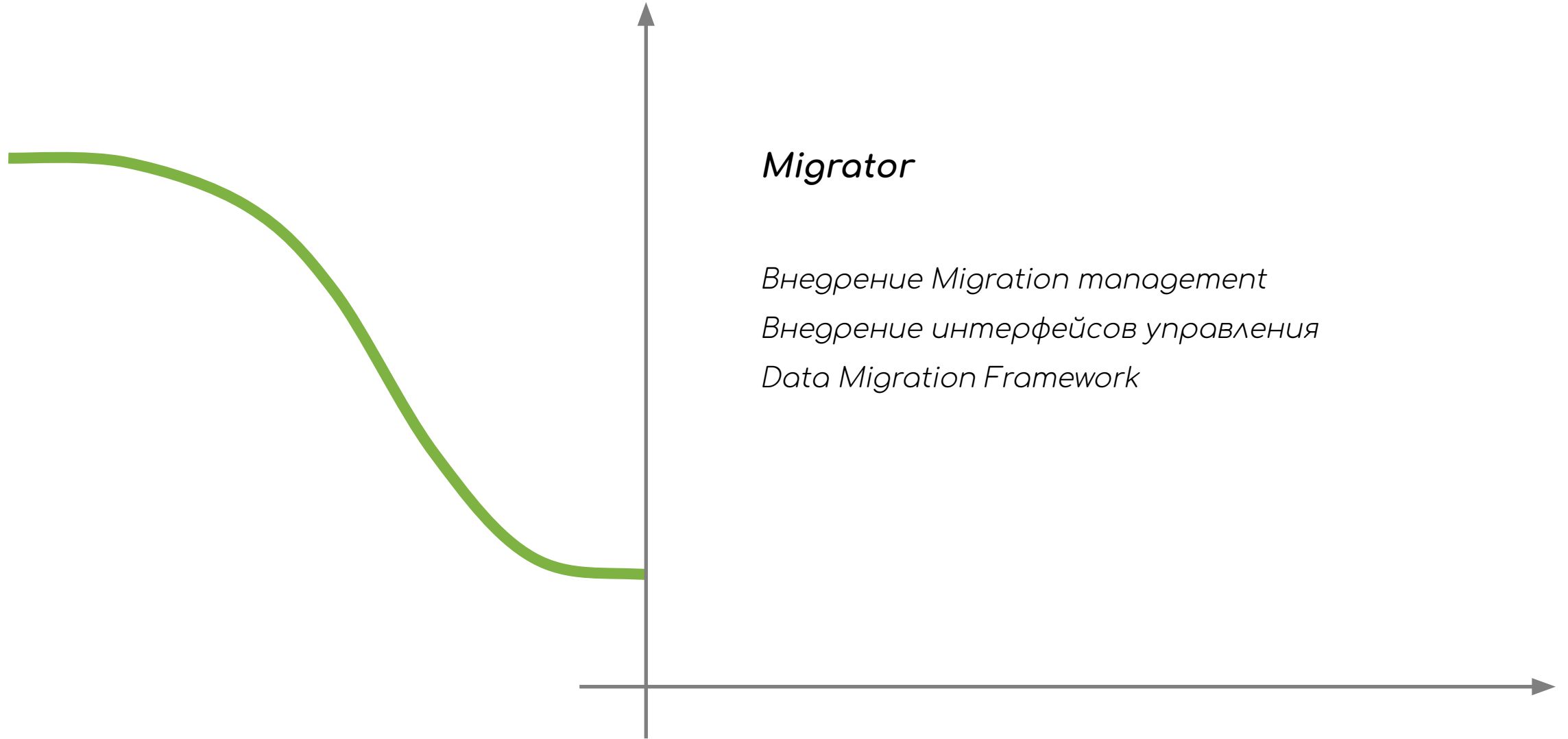
Что дальше? 😊



Что дальше?



Что дальше?



*Миграция – это нечто большее...*

- Бизнес-проект
- Бизнес-процесс
- Часть продукта



[Slides summary](#)



*Напишите мне, чтобы присоединиться  
к разработке Garson / Migrator*



Dima  
Burmistrov



[@pyctrl](#)



[dimaburmistrov](#)