
Distributed data in Django services
Pavel Lonkin, Team Leader
MORE AS A SERVICE

Hi, there!

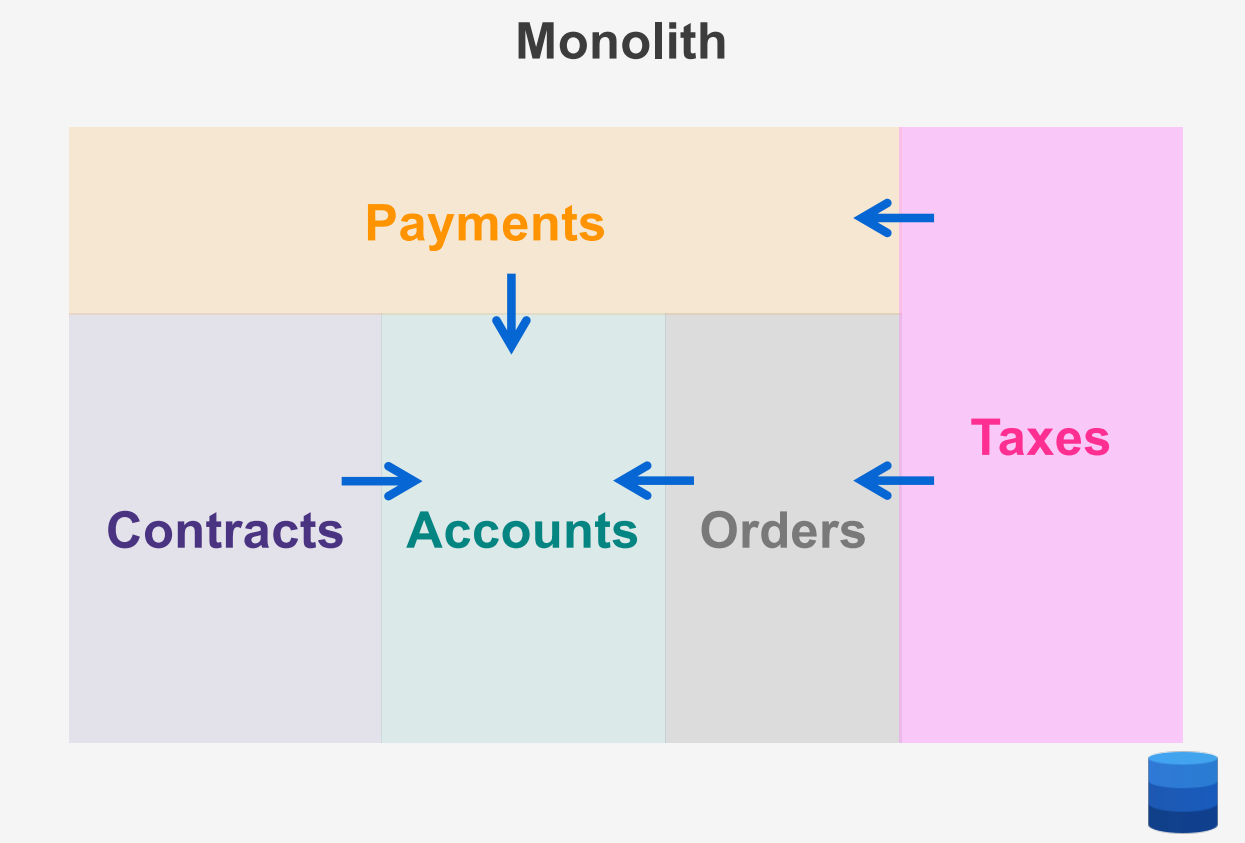
Pavel “Pasha” Lonkin

https://www.linkedin.com/in/plonkin

@pavel_lonkin

@d3rky

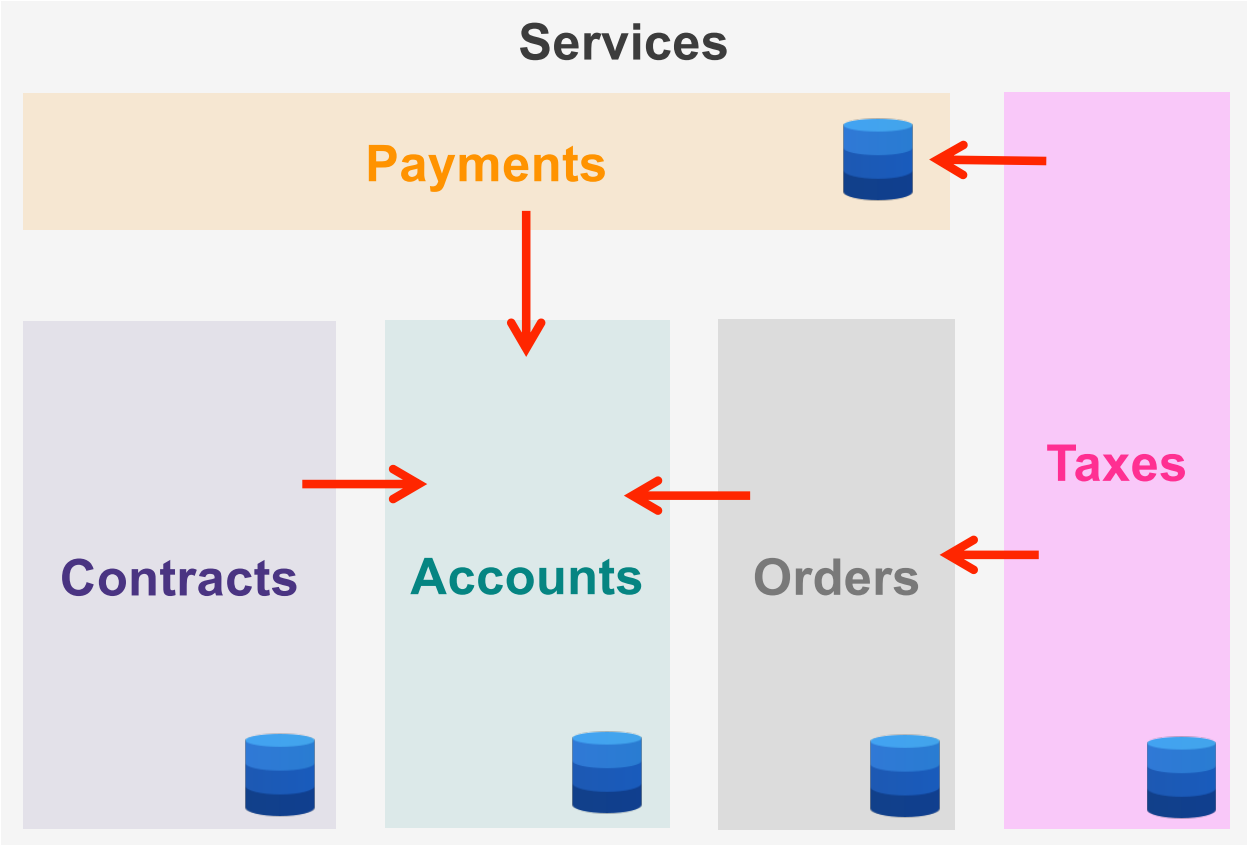
What’s the problem?



Example of API call

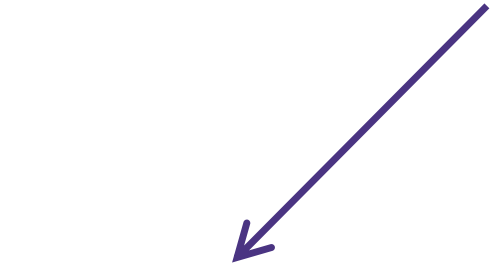
```
GET /payments

{
 "id": "P-001",
 ...
 "account": {
    "id": "AC-002",
    "name": "Pavel"
 }
 ...
}
```

What's the problem?

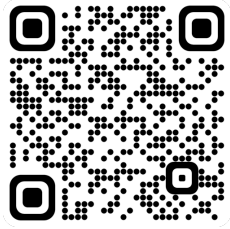


How to break the "ice"?
How to join data then?

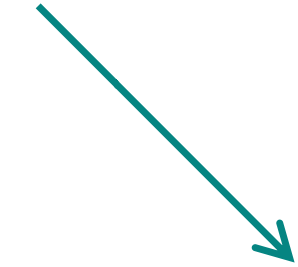
How to break the ”ice”?



API Composition



https://microservices.io/patterns/data/api-composition.html

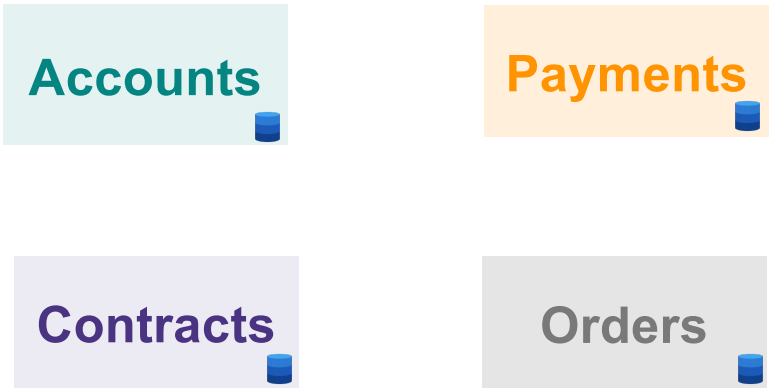


CQRS

(Command Query Responsibility Segregation)

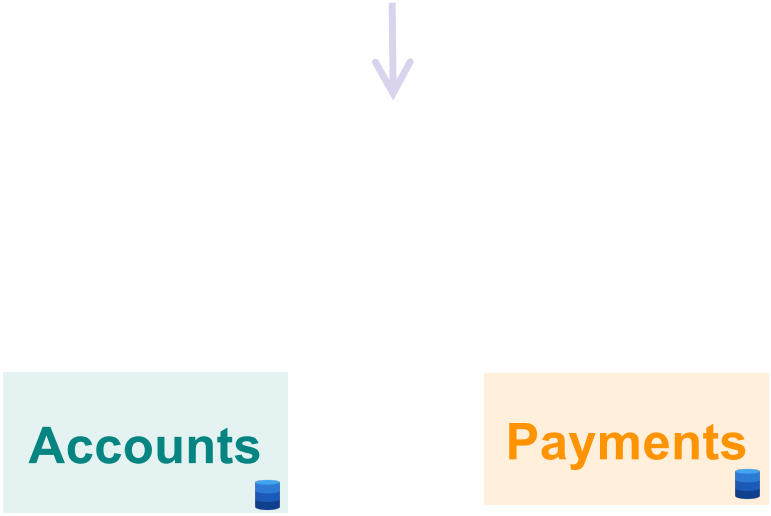
https://microservices.io/patterns/data/cqrs.html

API Composition

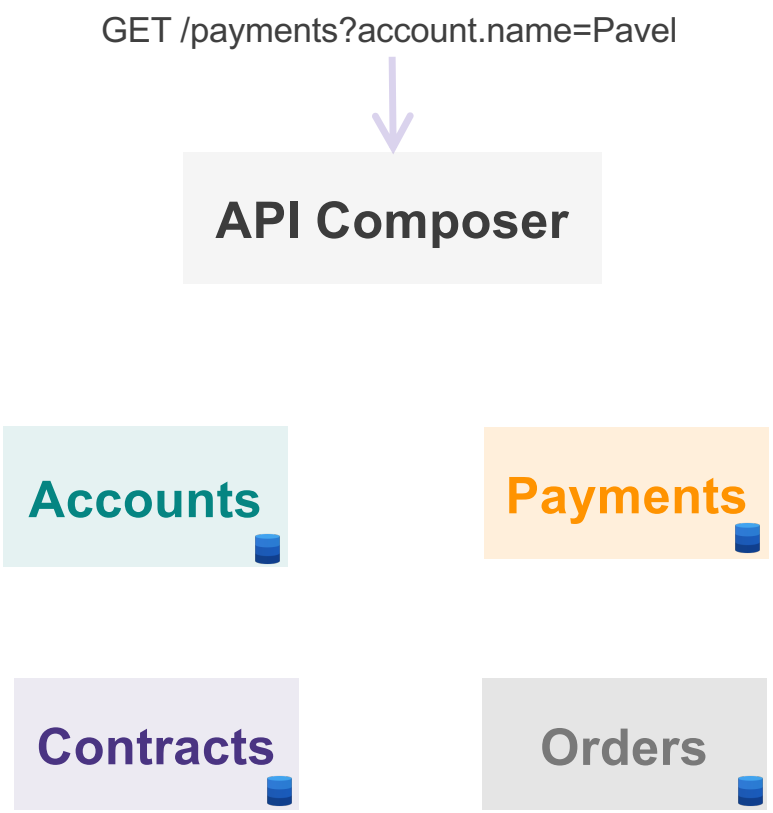


API Composition

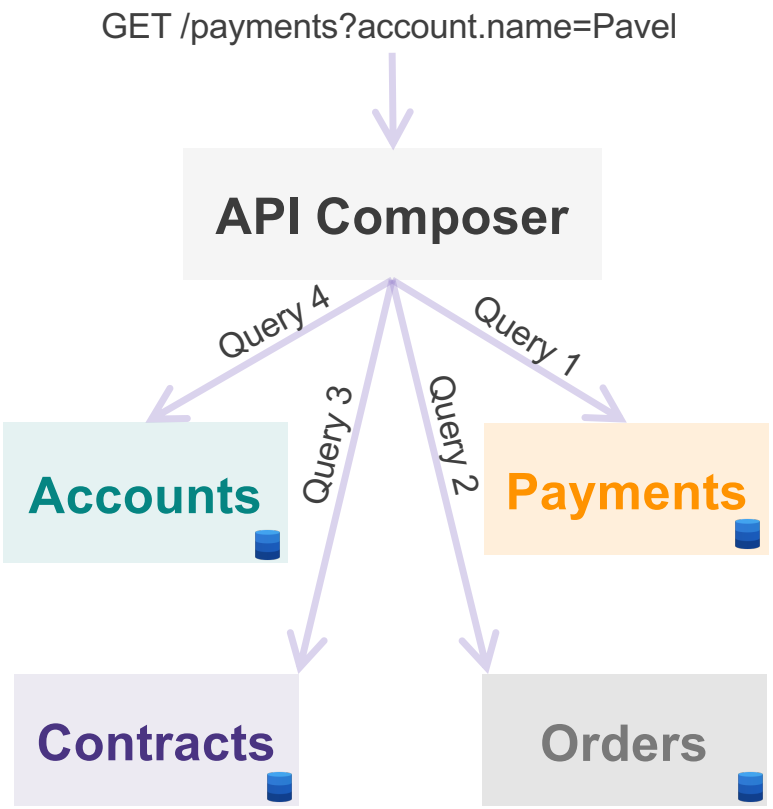
GET /payments?account.name=Pavel



API Composition



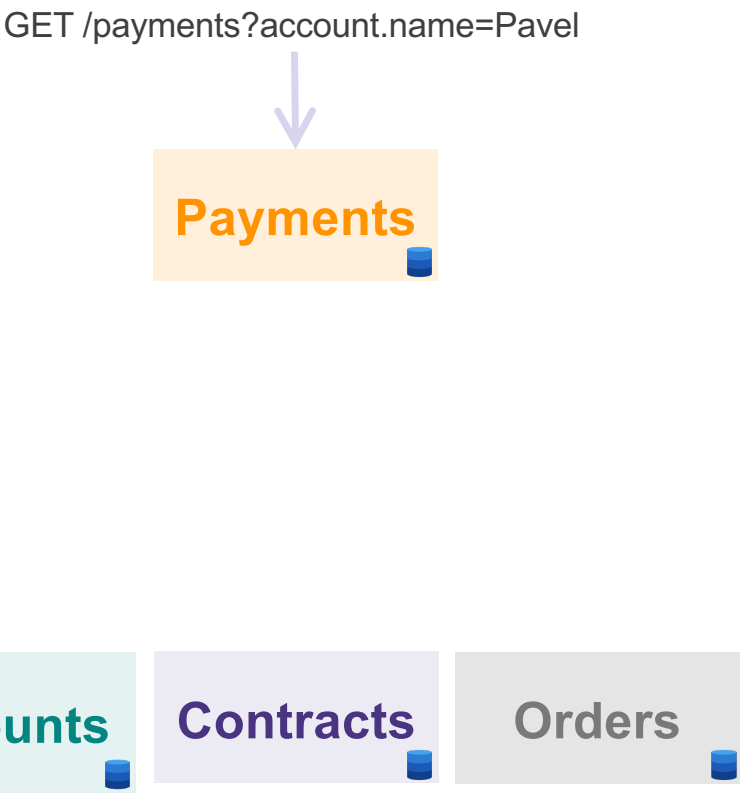
API Composition



- ✗ Inefficient
- ✗ In-memory joins of large data sets

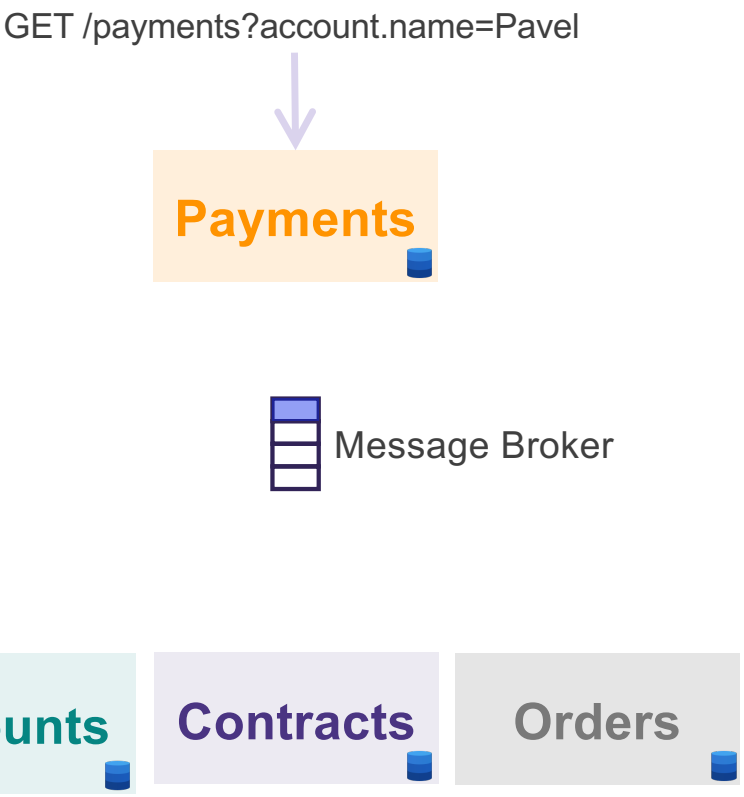
CQRS

(Command Query Responsibility Segregation)



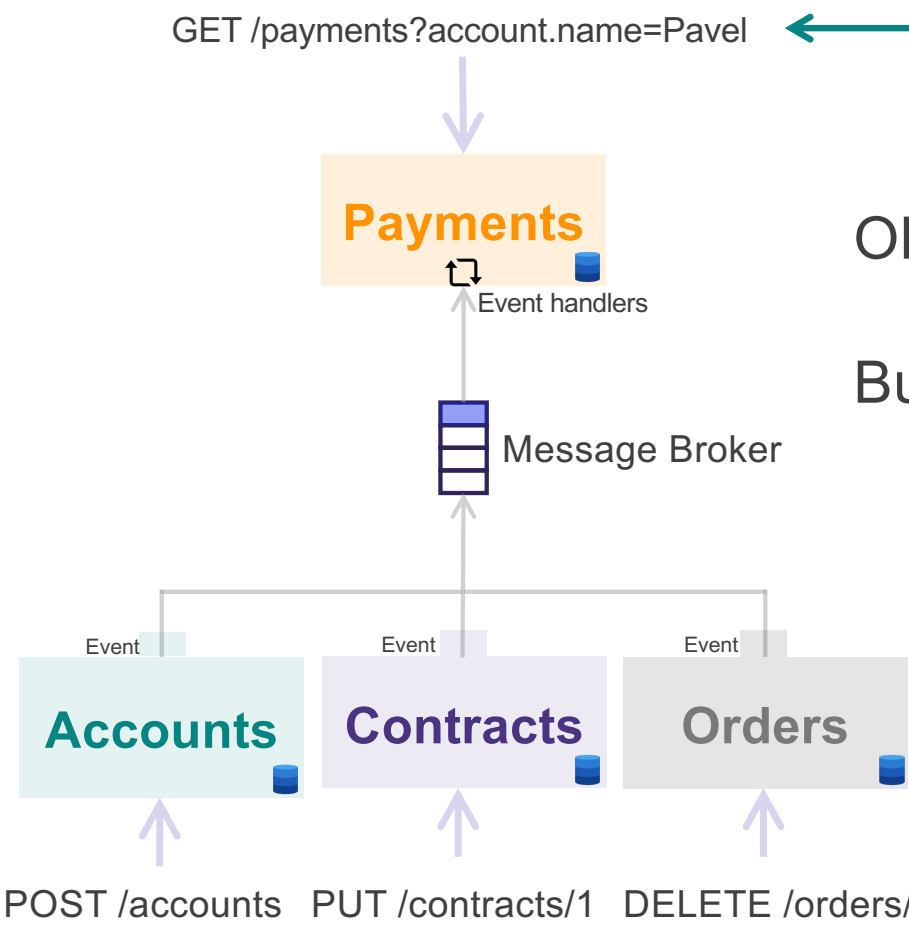
CQRS

(Command Query Responsibility Segregation)



CQRS

(Command Query Responsibility Segregation)

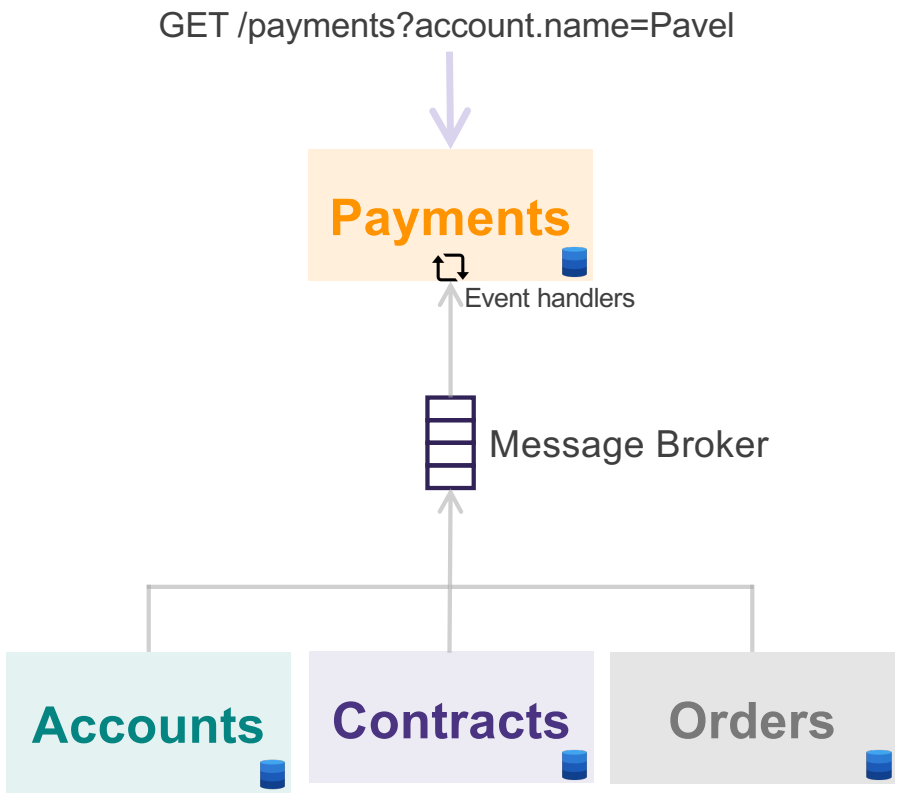


Ok, Pasha, that's cool.

But where are **queries** and **commands**?

CQRS

(Command Query Responsibility Segregation)



- ✓ Only needed data is on the destination side
- ✓ Low coherence
- ✓ Guaranteed consistency
- ✗ Replication lag or eventual consistency
- ✗ Code & data duplication

API Composition

✗ Inefficient

✗ In-memory joins of large data sets

CQRS ✓

(Command Query Responsibility Segregation)

✓ Only needed data is on the destination side

✓ Low coherence

✓ Guaranteed consistency

✗ Replication lag or eventual consistency

✗ Code & data duplication

What did we have at that moment?

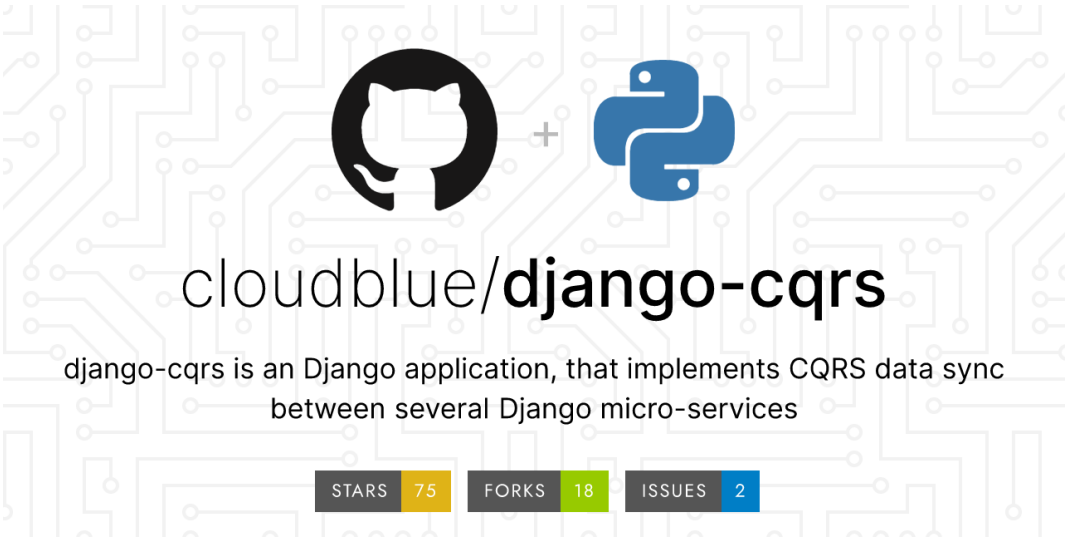
- 😄 We were young and full of energy
- 😄 Django
- 😄 Different databases (MySQL, Postgres)
- 😢 Nothing reasonable for CQRS pattern in Django

But…

We have a Team and Max!

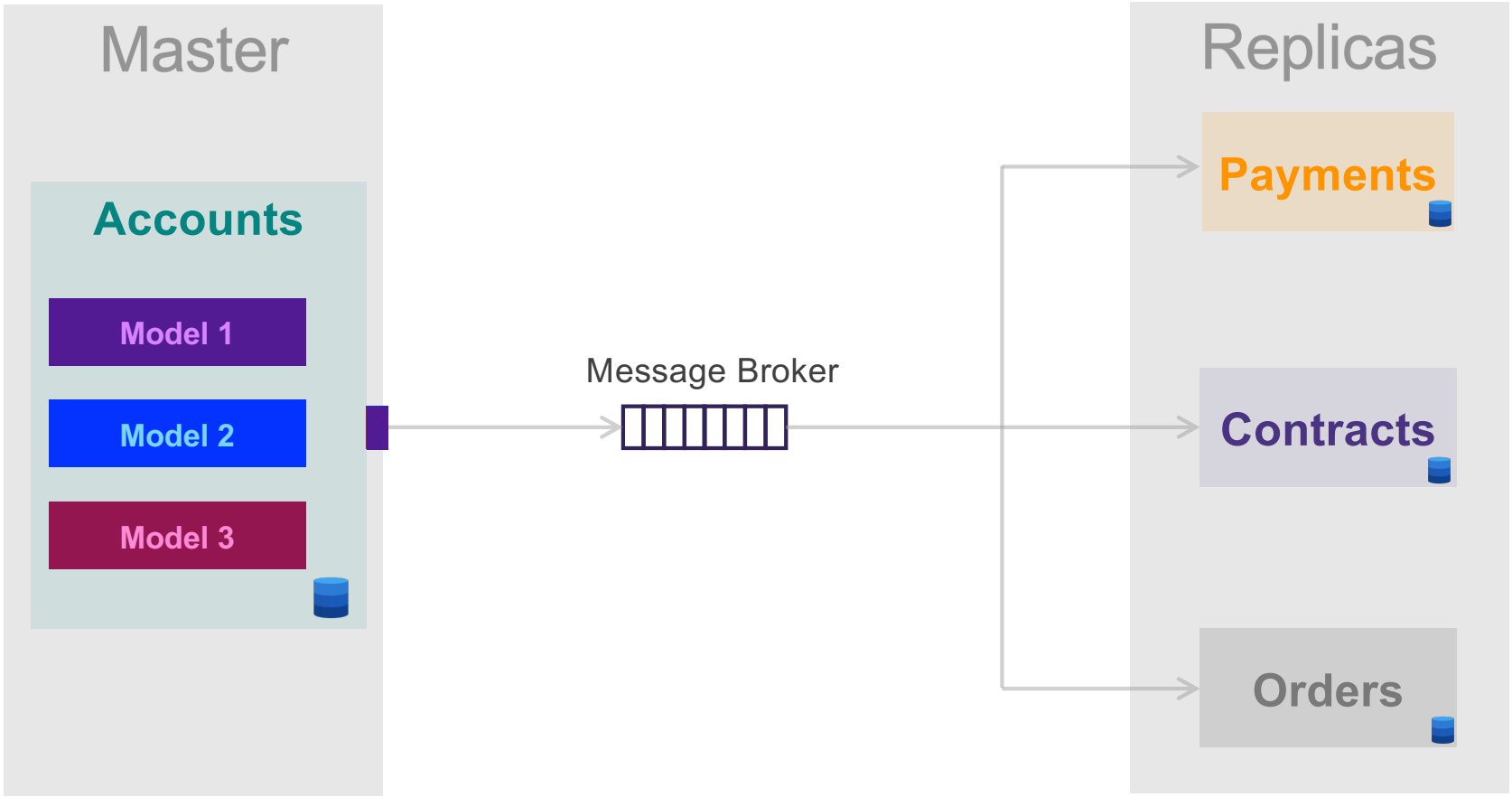
Maxim Kolyubyakin

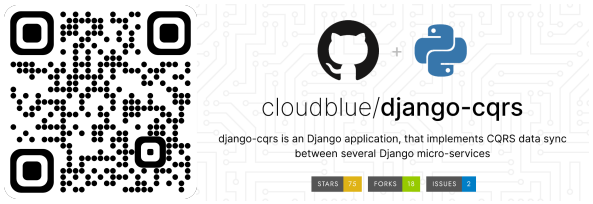
Django CQRS

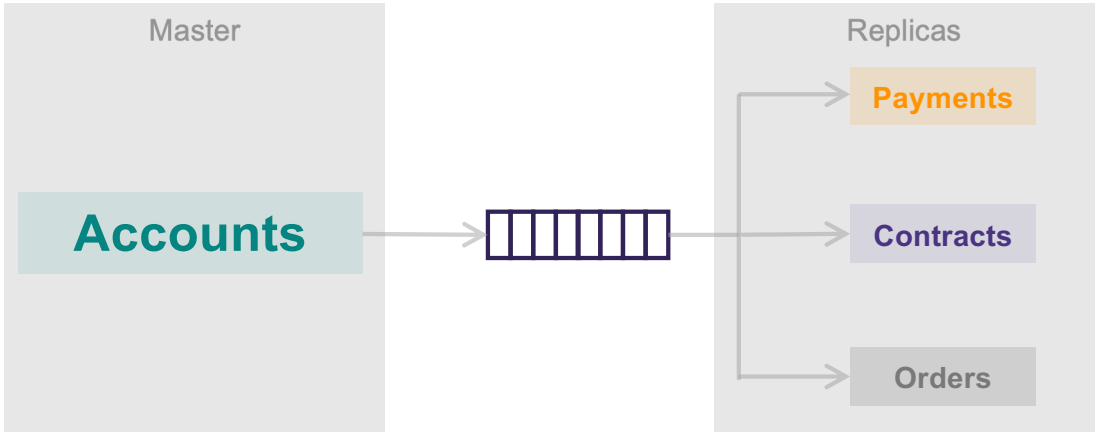

cloudblue/django-cqrs
django-cqrs is an Django application, that implements CQRS data sync between several Django micro-services
STARS 75
FORKS 18
ISSUES 2

cloudblue/**django-cqrs**

django-cqrs is an Django application, that implements CQRS data sync between several Django micro-services

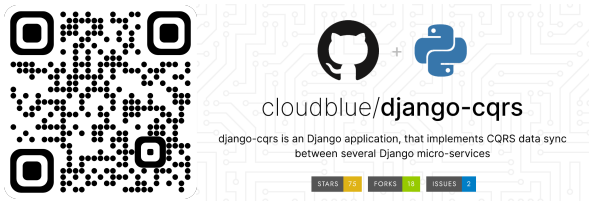






Key Features

- You need 5 steps to run it in your Django app
- You don’t need to dramatically change your code
- Supports different databases from out of the box
- Production ready



5 steps to run Django CQRS

1. Define Master Settings

Define Master Settings

```
# settings.py

CQRS = {
    'transport': 'dj_cqrs.transport.rabbit_mq.RabbitMQTransport',
    'host': RABBITMQ_HOST,
    'port': RABBITMQ_PORT,
    'user': RABBITMQ_USERNAME,
    'password': RABBITMQ_PASSWORD,
}
```

2. Update Master Models

Update Master Models

```
# models.py

from django.db import models
from dj_cqrs.mixins import MasterMixin, RawMasterMixin


class Account(MasterMixin, models.Model):
    CQRS_ID = 'account'
    CQRS_PRODUCE = True  # set this to False to prevent sending instances to
```

3. Define Replica Settings

Define Replica Settings

```
# settings.py

CQRS = {
    'transport': 'dj_cqrs.transport.RabbitMQTransport',
    'queue': 'account_replica',
    'host': RABBITMQ_HOST,
    'port': RABBITMQ_PORT,
    'user': RABBITMQ_USERNAME,
    'password': RABBITMQ_PASSWORD,
}
```

4. Define Replica Models

Define Replica Models

```
from django.db import models
from dj_cqrs.mixins import ReplicaMixin


class AccountRef(ReplicaMixin, models.Model):
    CQRS_ID = 'account'

    id = models.IntegerField(primary_key=True)
```

5. Run Replica Consumers

Run Replica Consumers

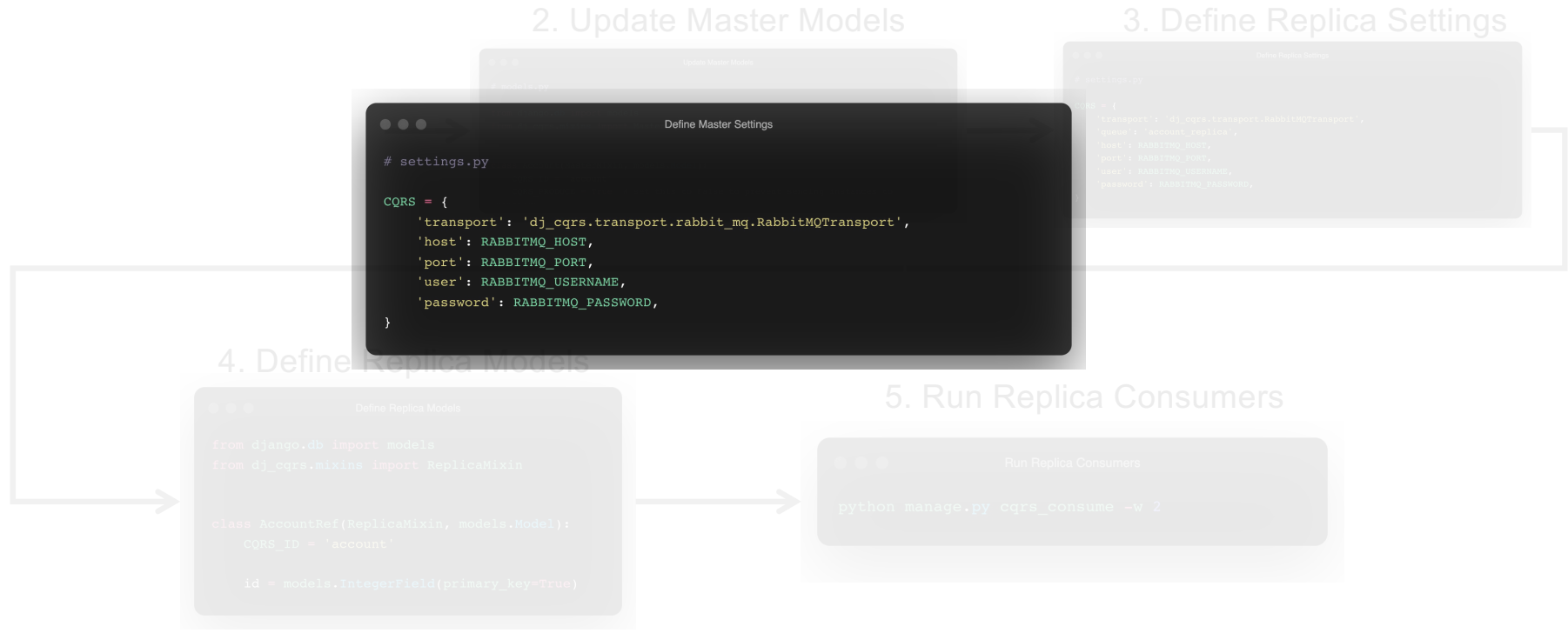
```
python manage.py cqrs_consume -w 2
```

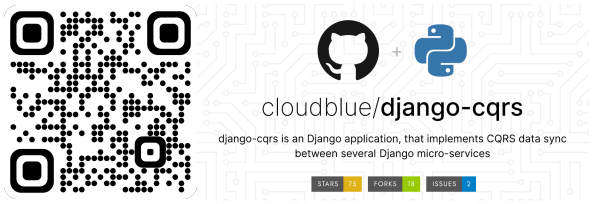
cloudblue/**django-cqrs**

django-cqrs is an Django application, that implements CQRS data sync between several Django micro-services

STARS 75 FORKS 18 ISSUES 2

5 steps to run Django CQRS





5 steps to run Django CQRS

1. Define Master Settings

Define Master Settings

```python
# settings.py

CQRS = {
    'transport': 'dj_cqrs.transport.rabbit_mq.RabbitMQTransport',
    'host': RABBITMQ_HOST,
    'port': RABBITMQ_PORT,
    'user': RABBITMQ_USERNAME,
    'password': RABBITMQ_PASSWORD,
}
```

2. Update Master Models

Update Master Models

```python
# models.py

from django.db import models
from dj_cqrs.mixins import MasterMixin, RawMasterMixin


class Account(MasterMixin, models.Model):
    CQRS_ID = 'account'
    CQRS_PRODUCE = True  # set this to False to prevent sending instances to
```

3. Define Replica Settings

Define Replica Settings

```python
# settings.py

CQRS = {
    'transport': 'dj_cqrs.transport.RabbitMQTransport',
    'queue': 'account_replica',
    'host': RABBITMQ_HOST,
    'port': RABBITMQ_PORT,
    'user': RABBITMQ_USERNAME,
    'password': RABBITMQ_PASSWORD,
}
```

4. Define Replica Models

Define Replica Models

```python
from django.db import models
from dj_cqrs.mixins import ReplicaMixin


class AccountRef(ReplicaMixin, models.Model):
    CQRS_ID = 'account'

    id = models.IntegerField(primary_key=True)
```

5. Run Replica Consumers

Run Replica Consumers

```
python manage.py cqrs_consume -w 2
```

cloudblue/**django-cqrs**

django-cqrs is an Django application, that implements CQRS data sync between several Django micro-services

STARS 75 FORKS 18 ISSUES 2

5 steps to run Django CQRS

1. Define Master Settings

3. Define Replica Settings

Update Master Models

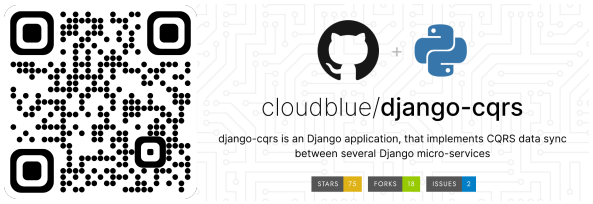
```python
# models.py

from django.db import models
from dj_cqrs.mixins import MasterMixin, RawMasterMixin


class Account(MasterMixin, models.Model):
    CQRS_ID = 'account'
    CQRS_PRODUCE = True  # set this to False to prevent sending instances to
```

4. Define Replica Models

```python
from django.db import models
from dj_cqrs.mixins import ReplicaMixin


class AccountRef(ReplicaMixin, models.Model):
    CQRS_ID = 'account'

    id = models.IntegerField(primary_key=True)
```

5. Run Replica Consumers

```
python manage.py cqrs_consume -w 2
```



5 steps to run Django CQRS

1. Define Master Settings

Define Master Settings

```python
# settings.py

CQRS = {
    'transport': 'dj_cqrs.transport.rabbit_mq.RabbitMQTransport',
    'host': RABBITMQ_HOST,
    'port': RABBITMQ_PORT,
    'user': RABBITMQ_USERNAME,
    'password': RABBITMQ_PASSWORD,
}
```

2. Update Master Models

Update Master Models

```python
# models.py

from django.db import models
from dj_cqrs.mixins import MasterMixin, RawMasterMixin


class Account(MasterMixin, models.Model):
    CQRS_ID = 'account'
    CQRS_PRODUCE = True  # set this to False to prevent sending instances to
```

3. Define Replica Settings

Define Replica Settings

```python
# settings.py

CQRS = {
    'transport': 'dj_cqrs.transport.RabbitMQTransport',
    'queue': 'account_replica',
    'host': RABBITMQ_HOST,
    'port': RABBITMQ_PORT,
    'user': RABBITMQ_USERNAME,
    'password': RABBITMQ_PASSWORD,
}
```

4. Define Replica Models

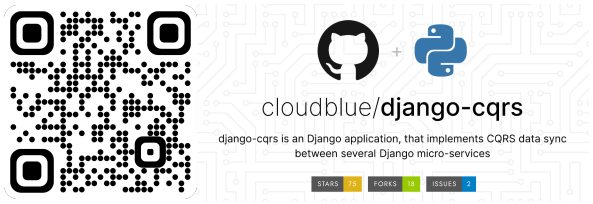
Define Replica Models

```python
from django.db import models
from dj_cqrs.mixins import ReplicaMixin


class AccountRef(ReplicaMixin, models.Model):
    CQRS_ID = 'account'

    id = models.IntegerField(primary_key=True)
```

5. Run Replica Consumers

Run Replica Consumers

```
python manage.py cqrs_consume -w 2
```



5 steps to run Django CQRS

1. Define Master Settings

2. Update Master Models

Define Replica Settings

```python
# settings.py

CQRS = {
    'transport': 'dj_cqrs.transport.RabbitMQTransport',
    'queue': 'account_replica',
    'host': RABBITMQ_HOST,
    'port': RABBITMQ_PORT,
    'user': RABBITMQ_USERNAME,
    'password': RABBITMQ_PASSWORD,
}
```

4. Define Replica Models

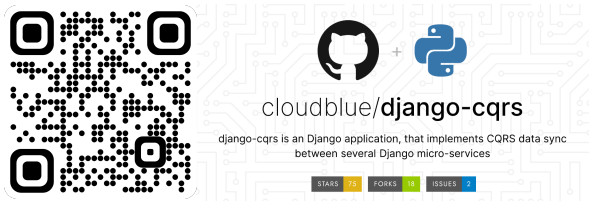
Define Replica Models

```python
from django.db import models
from dj_cqrs.mixins import ReplicaMixin


class AccountRef(ReplicaMixin, models.Model):
    CQRS_ID = 'account'

    id = models.IntegerField(primary_key=True)
```

5. Run Replica Consumers

Run Replica Consumers

```
python manage.py cqrs_consume -w 2
```



5 steps to run Django CQRS

1. Define Master Settings

Define Master Settings

```python
# settings.py

CQRS = {
    'transport': 'dj_cqrs.transport.rabbit_mq.RabbitMQTransport',
    'host': RABBITMQ_HOST,
    'port': RABBITMQ_PORT,
    'user': RABBITMQ_USERNAME,
    'password': RABBITMQ_PASSWORD,
}
```

2. Update Master Models

Update Master Models

```python
# models.py

from django.db import models
from dj_cqrs.mixins import MasterMixin, RawMasterMixin


class Account(MasterMixin, models.Model):
    CQRS_ID = 'account'
    CQRS_PRODUCE = True  # set this to False to prevent sending instances to
```

3. Define Replica Settings

Define Replica Settings

```python
# settings.py

CQRS = {
    'transport': 'dj_cqrs.transport.RabbitMQTransport',
    'queue': 'account_replica',
    'host': RABBITMQ_HOST,
    'port': RABBITMQ_PORT,
    'user': RABBITMQ_USERNAME,
    'password': RABBITMQ_PASSWORD,
}
```

4. Define Replica Models

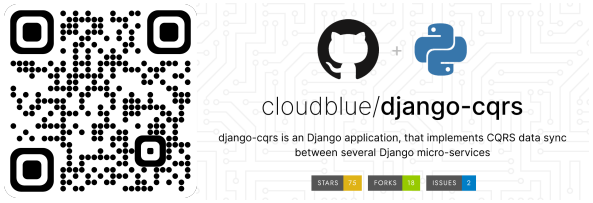
Define Replica Models

```python
from django.db import models
from dj_cqrs.mixins import ReplicaMixin


class AccountRef(ReplicaMixin, models.Model):
    CQRS_ID = 'account'

    id = models.IntegerField(primary_key=True)
```

5. Run Replica Consumers

Run Replica Consumers

```
python manage.py cqrs_consume -w 2
```



5 steps to run Django CQRS

1. Define Master Settings

2. Update Master Models

3. Define Replica Settings

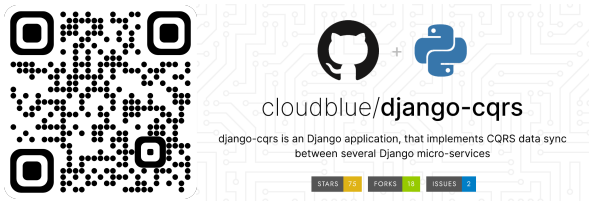
Define Replica Models

```python
from django.db import models
from dj_cqrs.mixins import ReplicaMixin


class AccountRef(ReplicaMixin, models.Model):
    CQRS_ID = 'account'

    id = models.IntegerField(primary_key=True)
```

5. Run Replica Consumers

Run Replica Consumers

```
python manage.py cqrs_consume -w 2
```



5 steps to run Django CQRS

1. Define Master Settings

Define Master Settings

```
# settings.py

CQRS = {
    'transport': 'dj_cqrs.transport.rabbit_mq.RabbitMQTransport',
    'host': RABBITMQ_HOST,
    'port': RABBITMQ_PORT,
    'user': RABBITMQ_USERNAME,
    'password': RABBITMQ_PASSWORD,
}
```

2. Update Master Models

Update Master Models

```
# models.py

from django.db import models
from dj_cqrs.mixins import MasterMixin, RawMasterMixin


class Account(MasterMixin, models.Model):
    CQRS_ID = 'account'
    CQRS_PRODUCE = True  # set this to False to prevent sending instances to
```

3. Define Replica Settings

Define Replica Settings

```
# settings.py

CQRS = {
    'transport': 'dj_cqrs.transport.RabbitMQTransport',
    'queue': 'account_replica',
    'host': RABBITMQ_HOST,
    'port': RABBITMQ_PORT,
    'user': RABBITMQ_USERNAME,
    'password': RABBITMQ_PASSWORD,
}
```

4. Define Replica Models

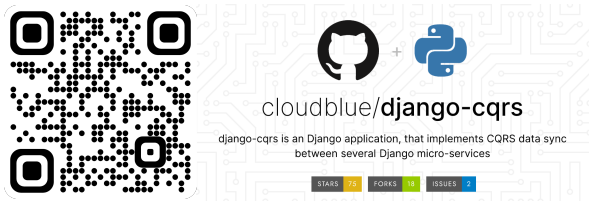
Define Replica Models

```
from django.db import models
from dj_cqrs.mixins import ReplicaMixin


class AccountRef(ReplicaMixin, models.Model):
    CQRS_ID = 'account'

    id = models.IntegerField(primary_key=True)
```

5. Run Replica Consumers

Run Replica Consumers

```
python manage.py cqrs_consume -w 2
```



5 steps to run Django CQRS

1. Define Master Settings

2. Update Master Models

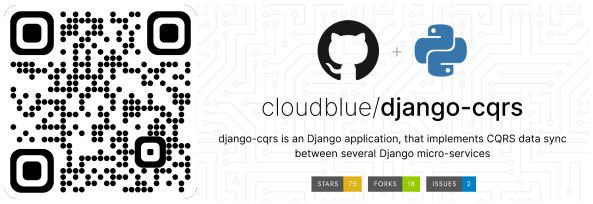
3. Define Replica Settings

Run Replica Consumers

```
python manage.py cqrs_consume -w 2
```

4. Define Replica Models

```
from django.db import models
from dj_cqrs.mixins import ReplicaMixin


class AccountRef(ReplicaMixin, models.Model):
    CQRS_ID = 'account'

    id = models.IntegerField(primary_key=True)
```



5 steps to run Django CQRS

1. Define Master Settings

Define Master Settings

```python
# settings.py

CQRS = {
    'transport': 'dj_cqrs.transport.rabbit_mq.RabbitMQTransport',
    'host': RABBITMQ_HOST,
    'port': RABBITMQ_PORT,
    'user': RABBITMQ_USERNAME,
    'password': RABBITMQ_PASSWORD,
}
```

2. Update Master Models

Update Master Models

```python
# models.py

from django.db import models
from dj_cqrs.mixins import MasterMixin, RawMasterMixin


class Account(MasterMixin, models.Model):
    CQRS_ID = 'account'
    CQRS_PRODUCE = True  # set this to False to prevent sending instances to
```

3. Define Replica Settings

Define Replica Settings

```python
# settings.py

CQRS = {
    'transport': 'dj_cqrs.transport.RabbitMQTransport',
    'queue': 'account_replica',
    'host': RABBITMQ_HOST,
    'port': RABBITMQ_PORT,
    'user': RABBITMQ_USERNAME,
    'password': RABBITMQ_PASSWORD,
}
```

4. Define Replica Models

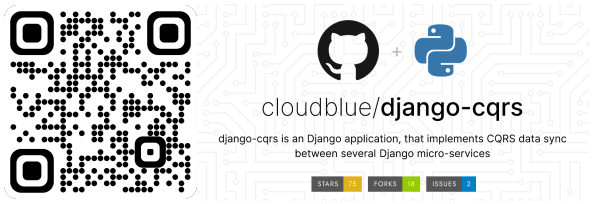
Define Replica Models

```python
from django.db import models
from dj_cqrs.mixins import ReplicaMixin


class AccountRef(ReplicaMixin, models.Model):
    CQRS_ID = 'account'

    id = models.IntegerField(primary_key=True)
```

5. Run Replica Consumers

Run Replica Consumers

```
python manage.py cqrs_consume -w 2
```



5 steps to run Django CQRS

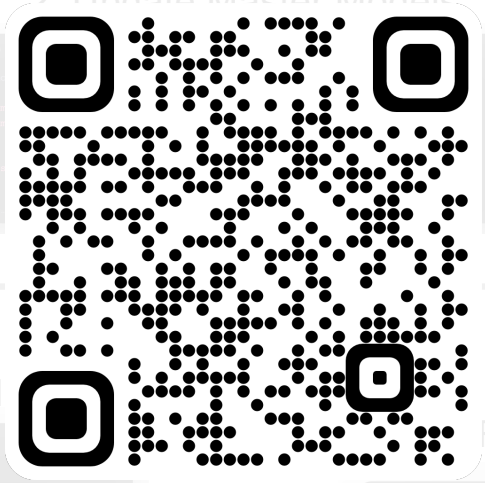
1. Define Master Settings

2. Update Master Models

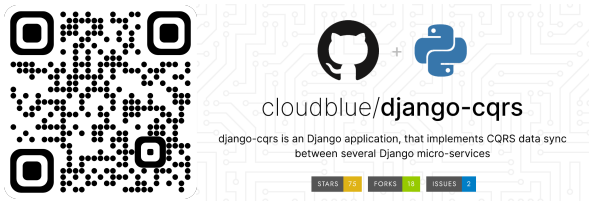
3. Define Replica Settings

4. Define Replica M

Run Replica Consumers



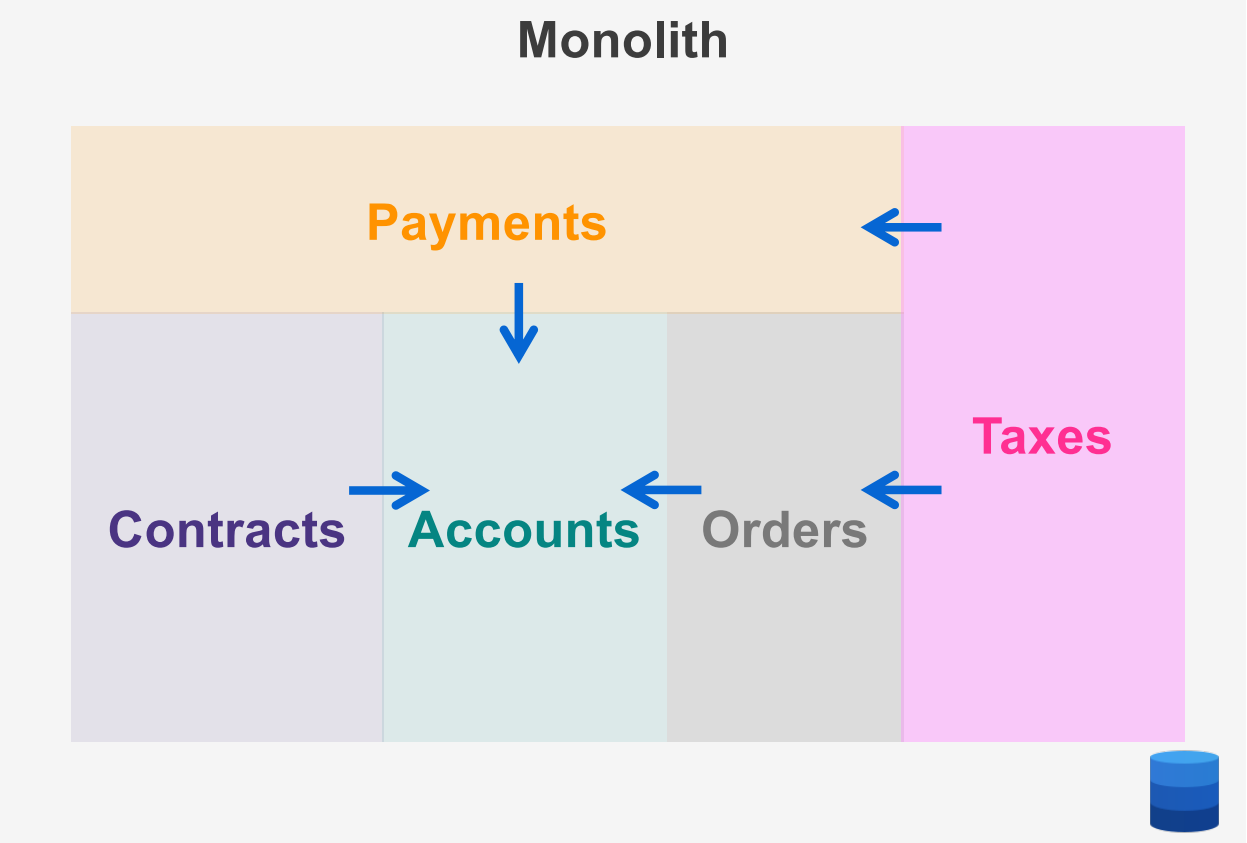
Demo example



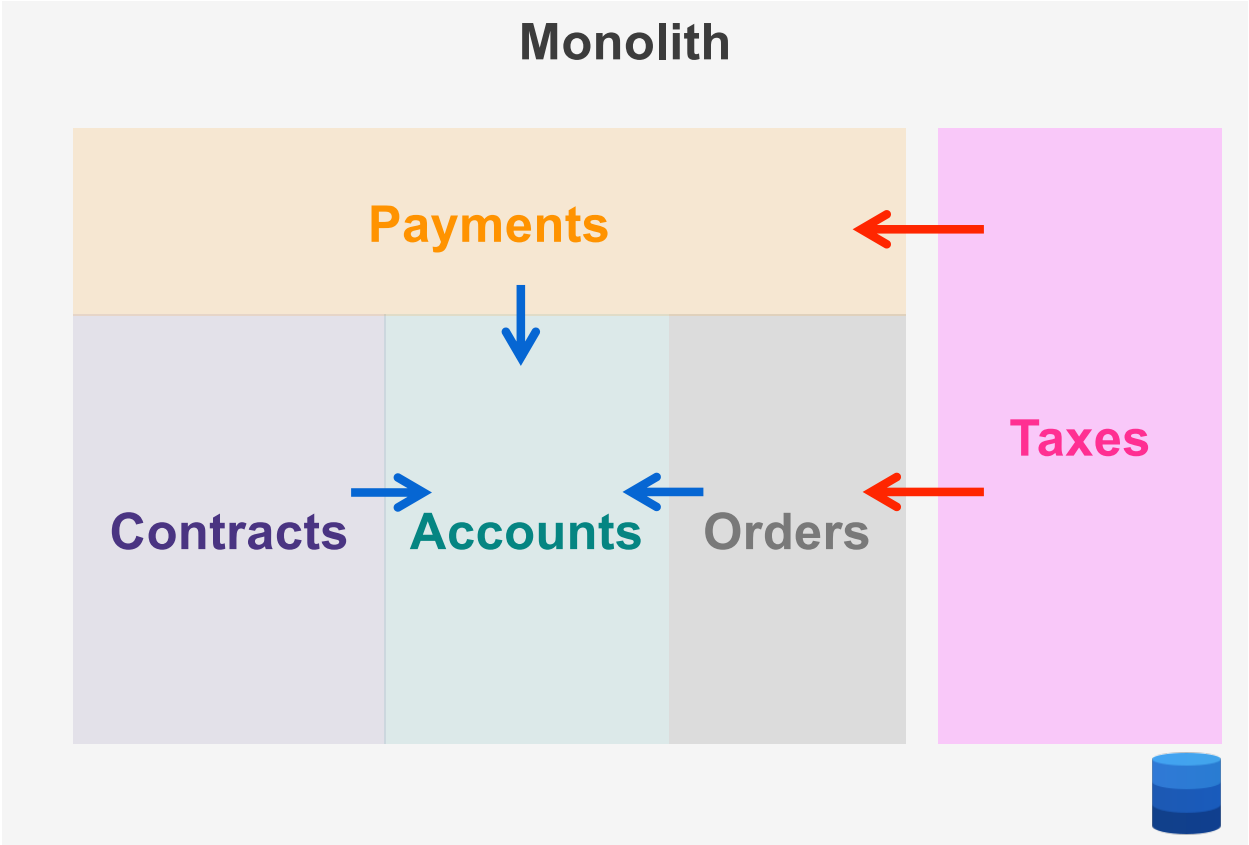
Other Features

- ✓ Retry and Dead Letters Queues
- ✓ Administration tools to sync, check diff between master and replicas, etc
- ✓ Support of multiple transports (you can use your own)

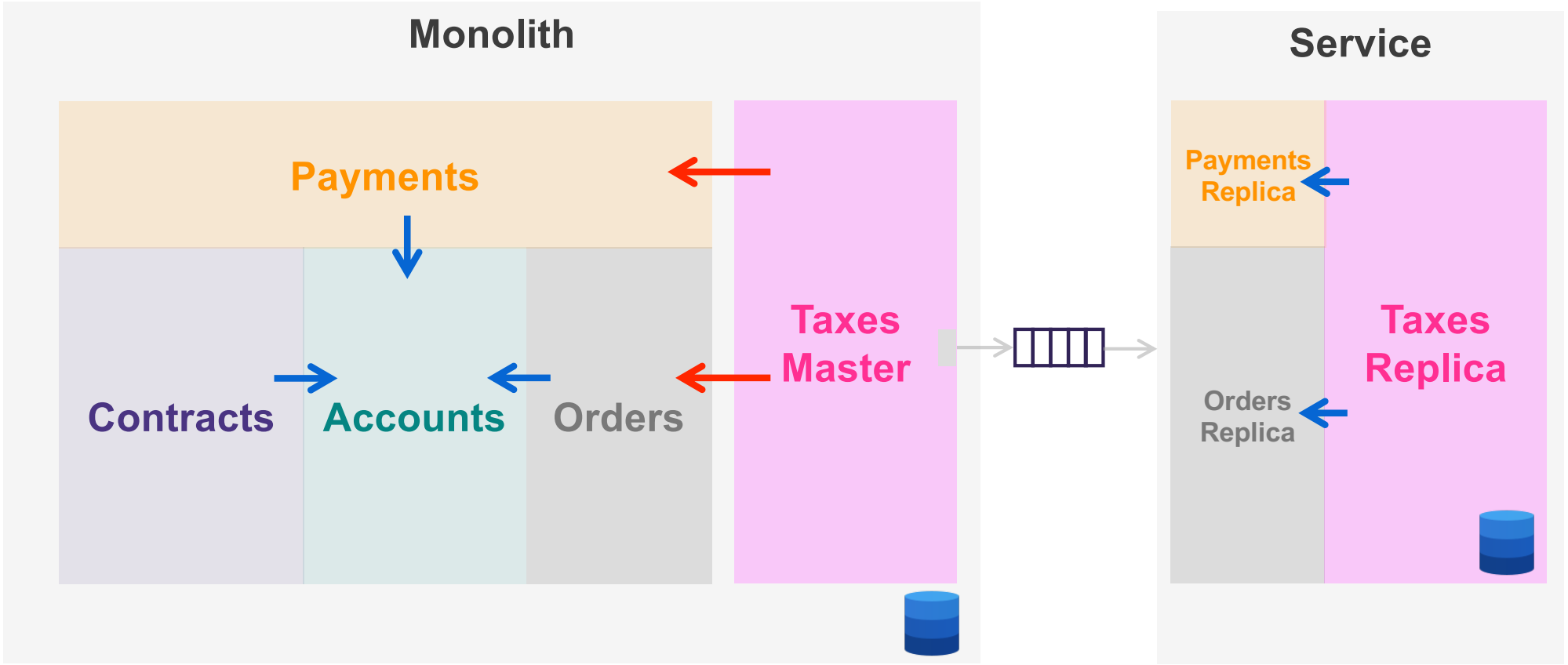
Bonus! Zero downtime data migrations



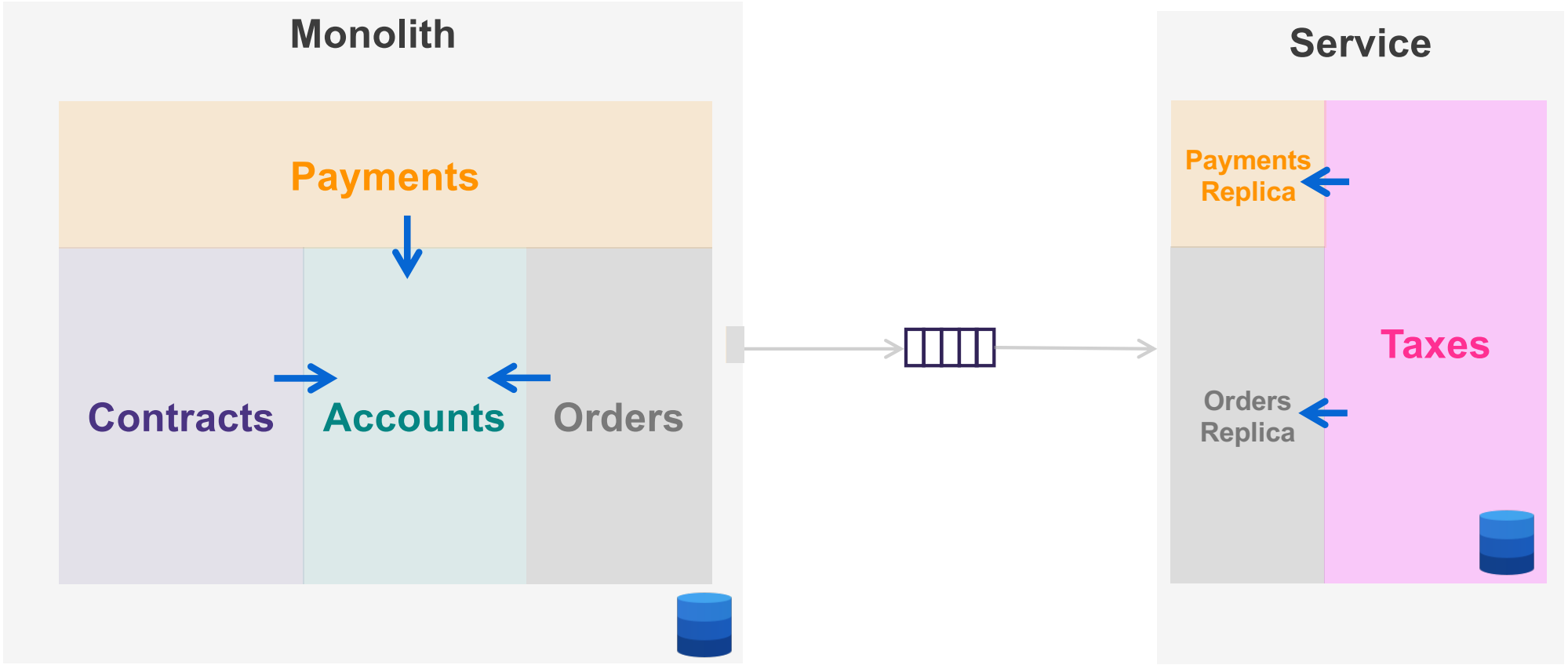
Bonus! Zero downtime data migrations

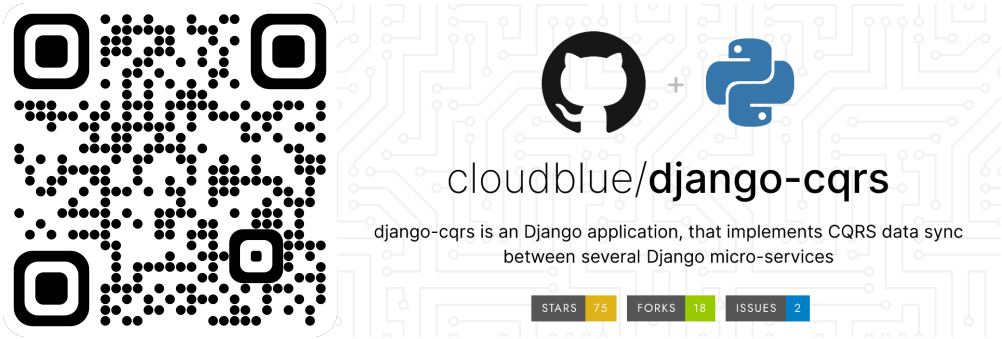


Bonus! Zero downtime data migrations

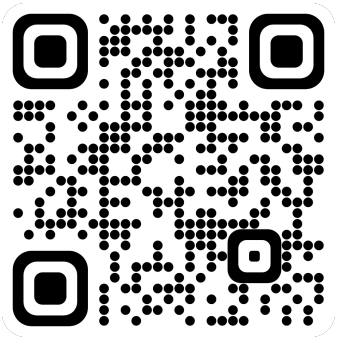


Bonus! Zero downtime data migrations





Careers



https://www.cloudblue.com/company/careers/

https://www.linkedin.com/in/plonkin

@pavel_lonkin

@d3rky