# PostgreSQL Partitioning

Управление данными и производительность с .NET и EF Core

# Илья Мжачев

Ведущий разработчик


Email ilya@mzhachev.ru

Telegram @mzhach


# О проекте

# О проекте

- Обрабатываем > 10 млн. звонков в день

# О проекте

- Обрабатываем > 10 млн. звонков в день
- Между звонками в среднем 5-10 секунд

# О проекте

- Обрабатываем > 10 млн. звонков в день
- Между звонками в среднем 5-10 секунд
- > 50% звонков обслуживается роботами

# Проблема

# Создание связки

Представитель
+79780000002

Телефония

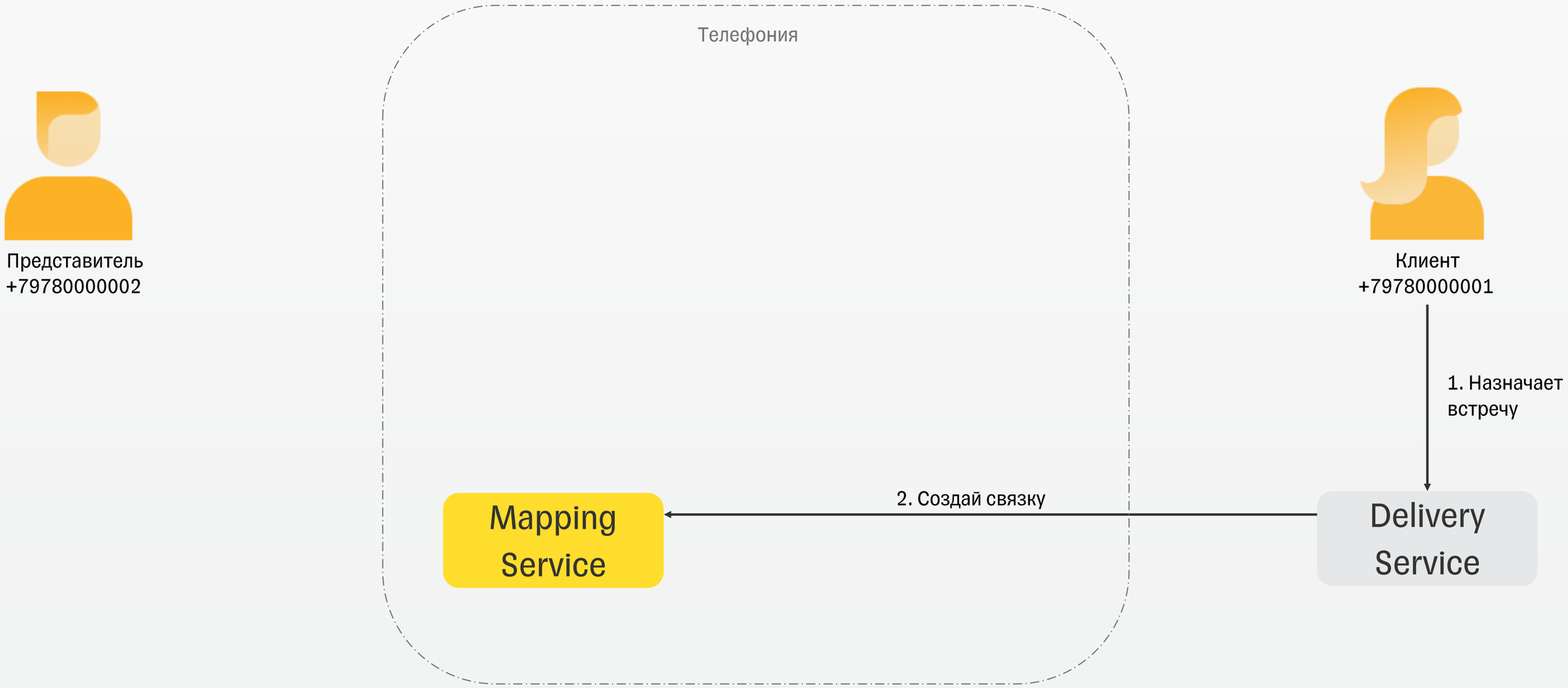
Mapping Service

Клиент
+79780000001

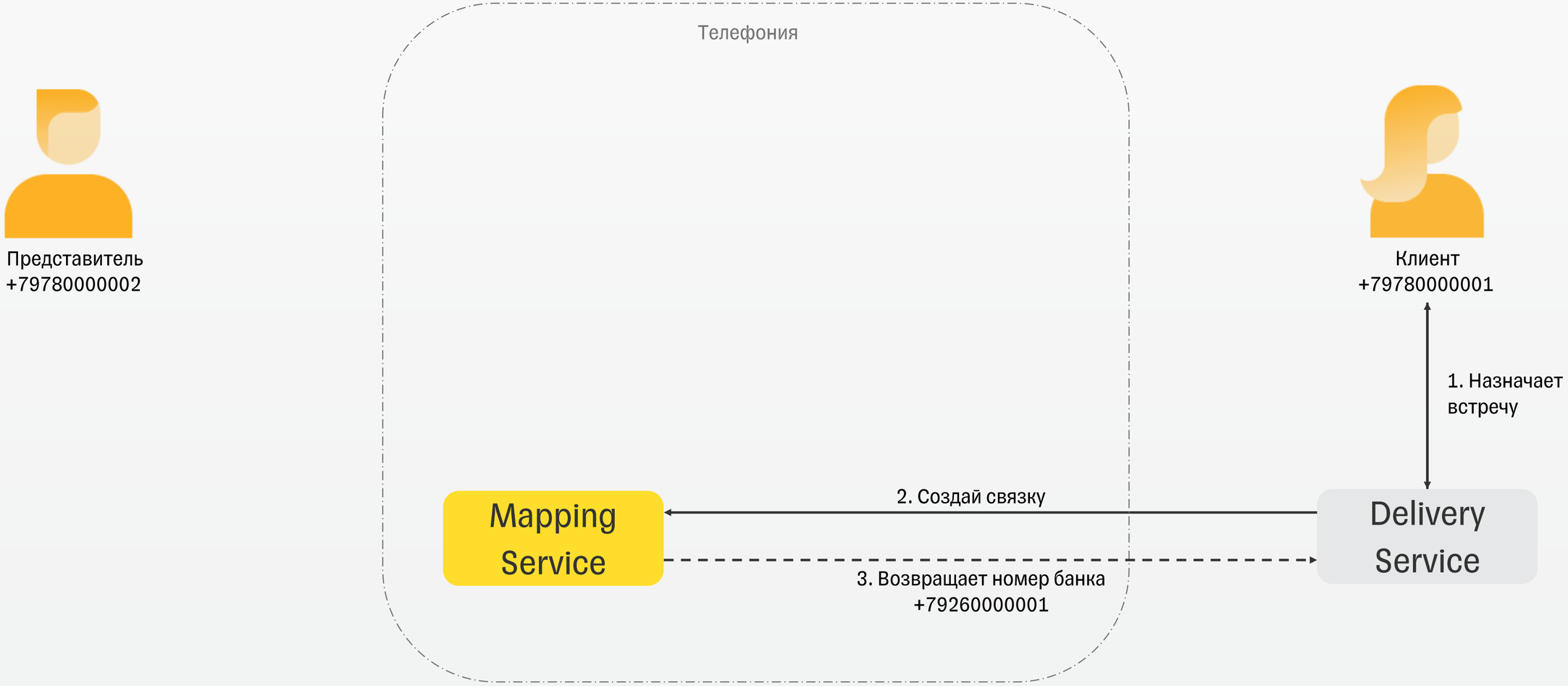
1. Назначает встречу

Delivery Service

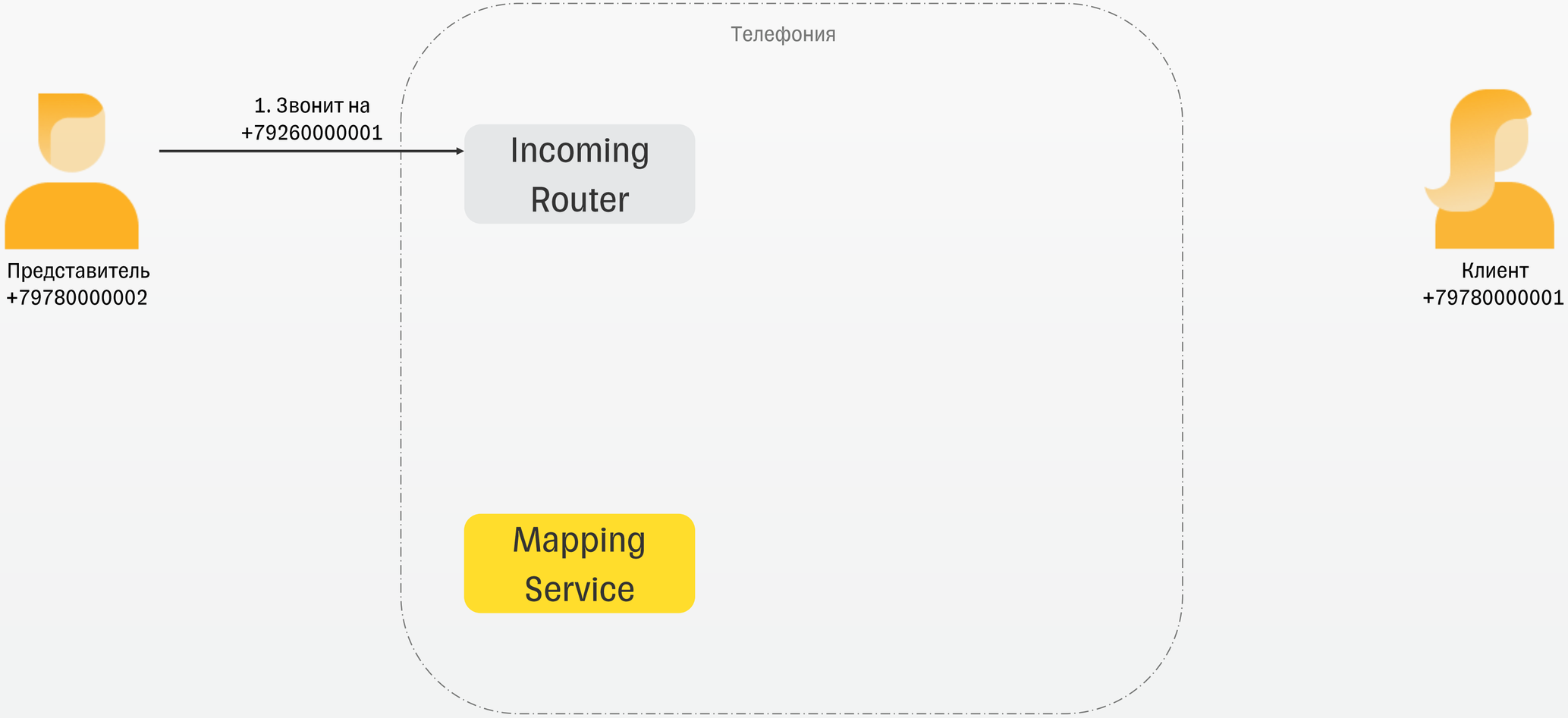
# Создание связки



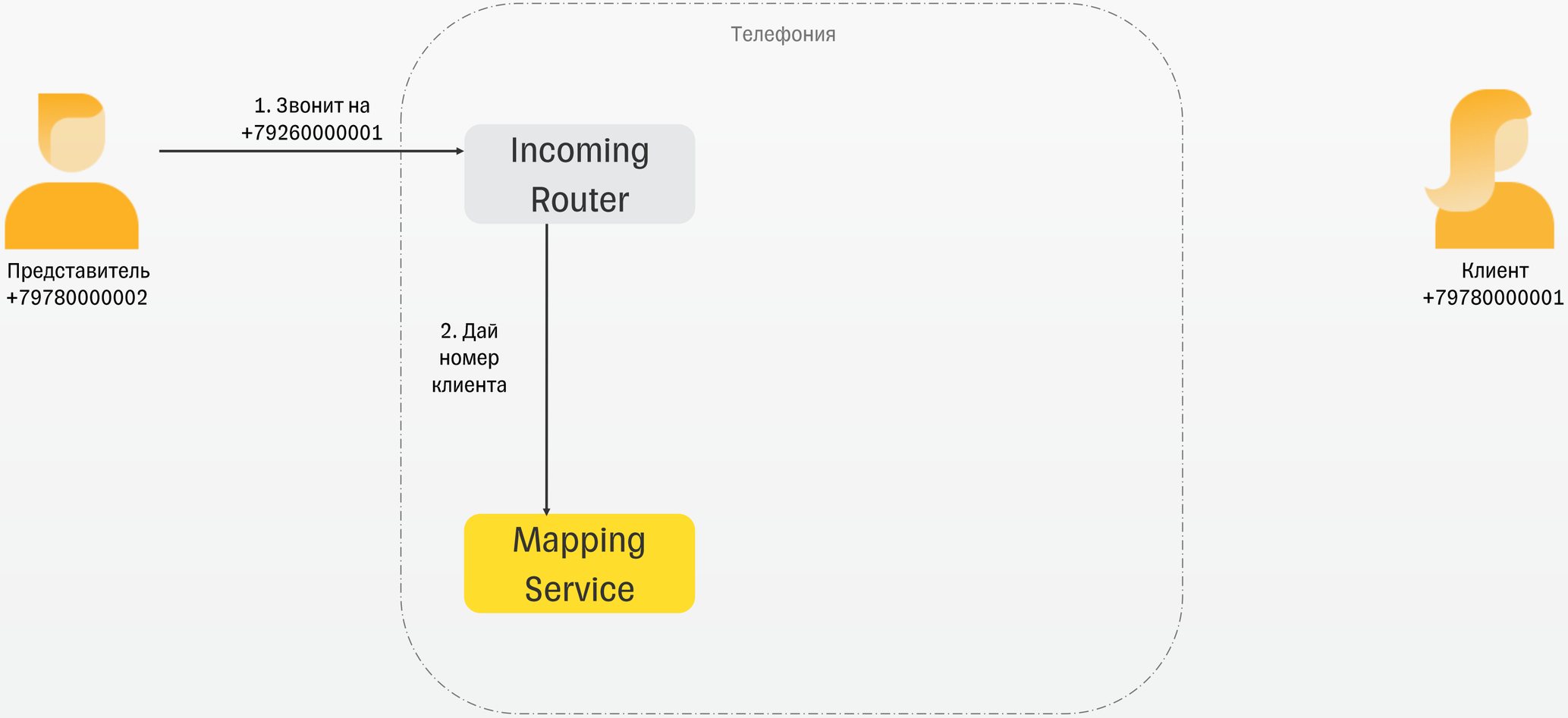
# Создание связки



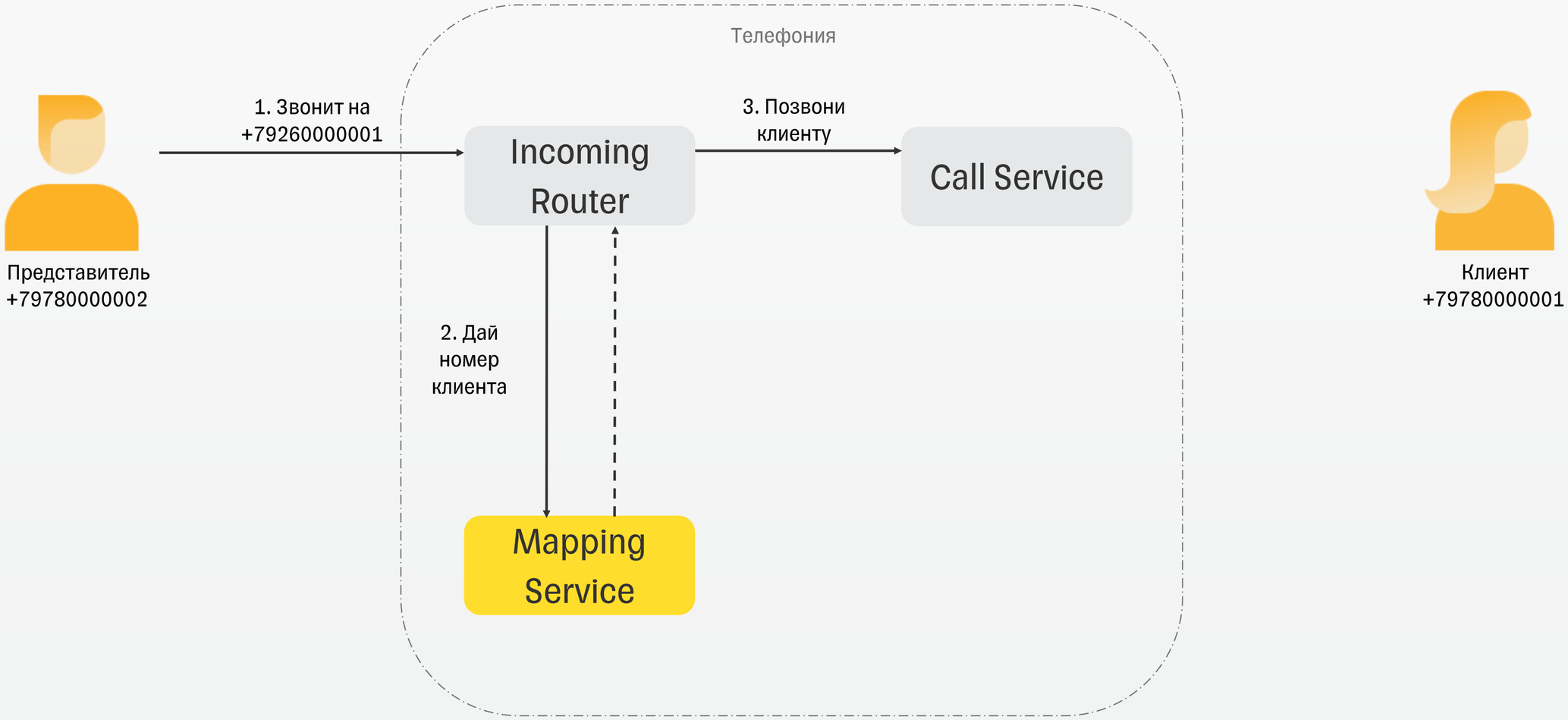
# Сокрытие номера клиента



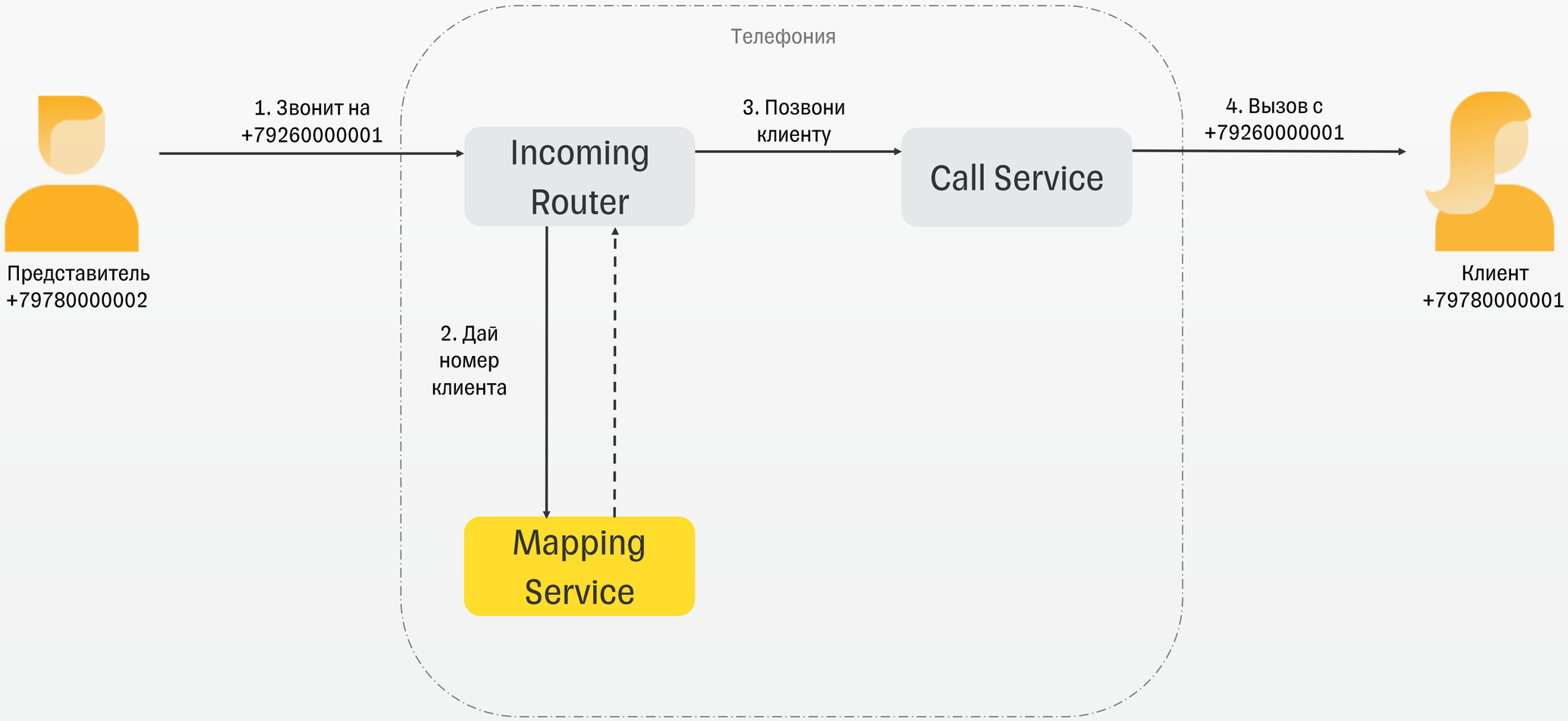
# Сокрытие номера клиента



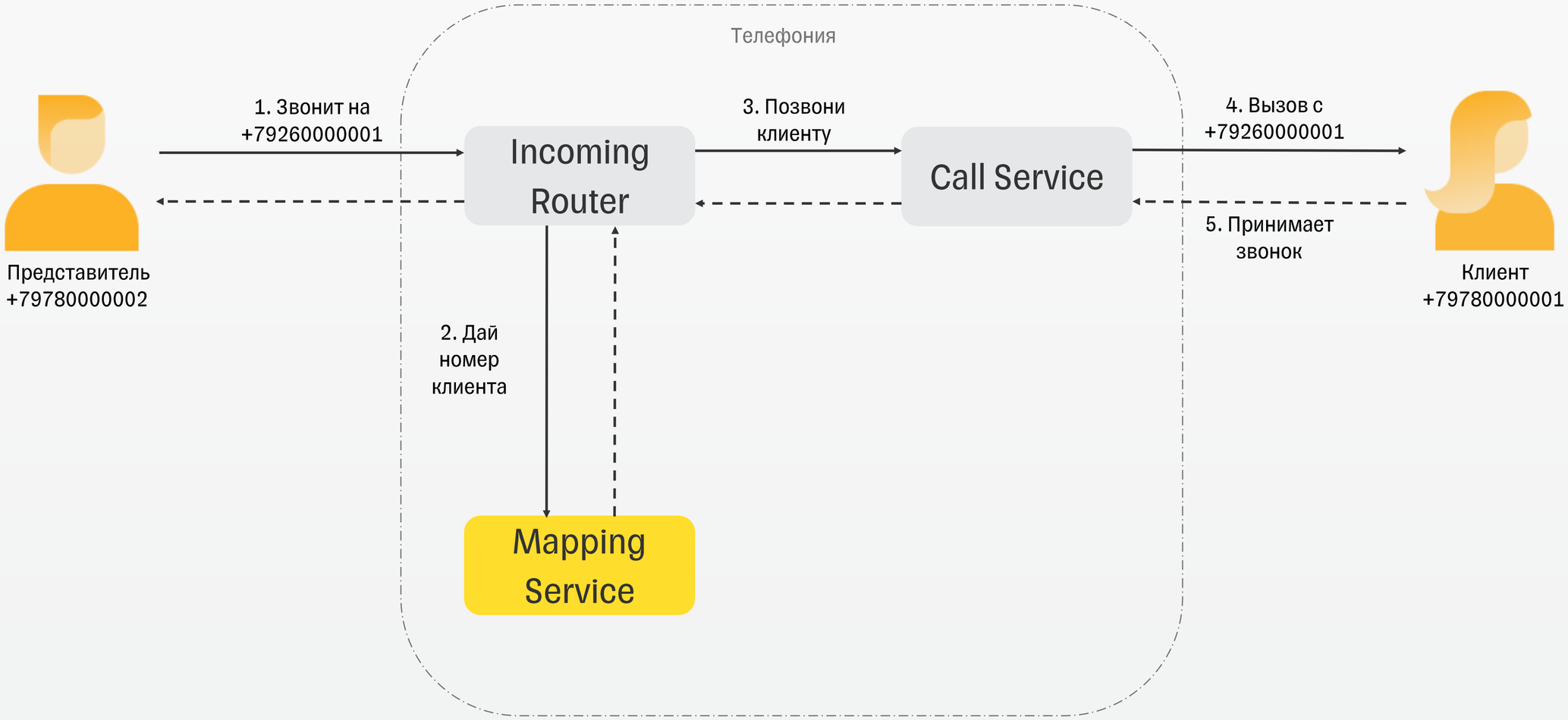
# Сокрытие номера клиента



# Сокрытие номера клиента



# Сокрытие номера клиента



# А что делать со связками встреч, которые уже прошли?

# Способы удаления данных из PostgreSQL

01

Через “DELETE”

# Минусы удаления через DELETE

### Дополнительная нагрузка

Запрос на большой таблице создает дополнительную нагрузку.

### Данные очищаются не сразу

Освобождение памяти произойдет только после VACUUM FULL.

# Способы удаления данных из PostgreSQL

01

Через “DELETE”

02

Через “TRUNCATE”

# Способы удаления данных из PostgreSQL

01

Через “DELETE”

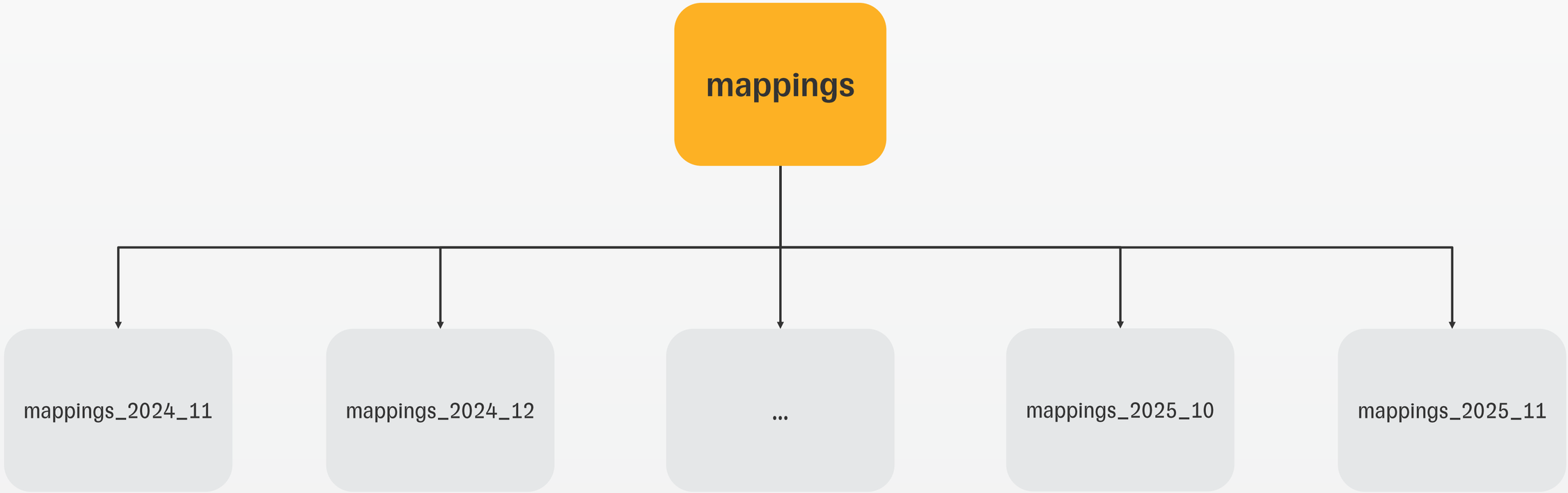
02

Через “TRUNCATE”

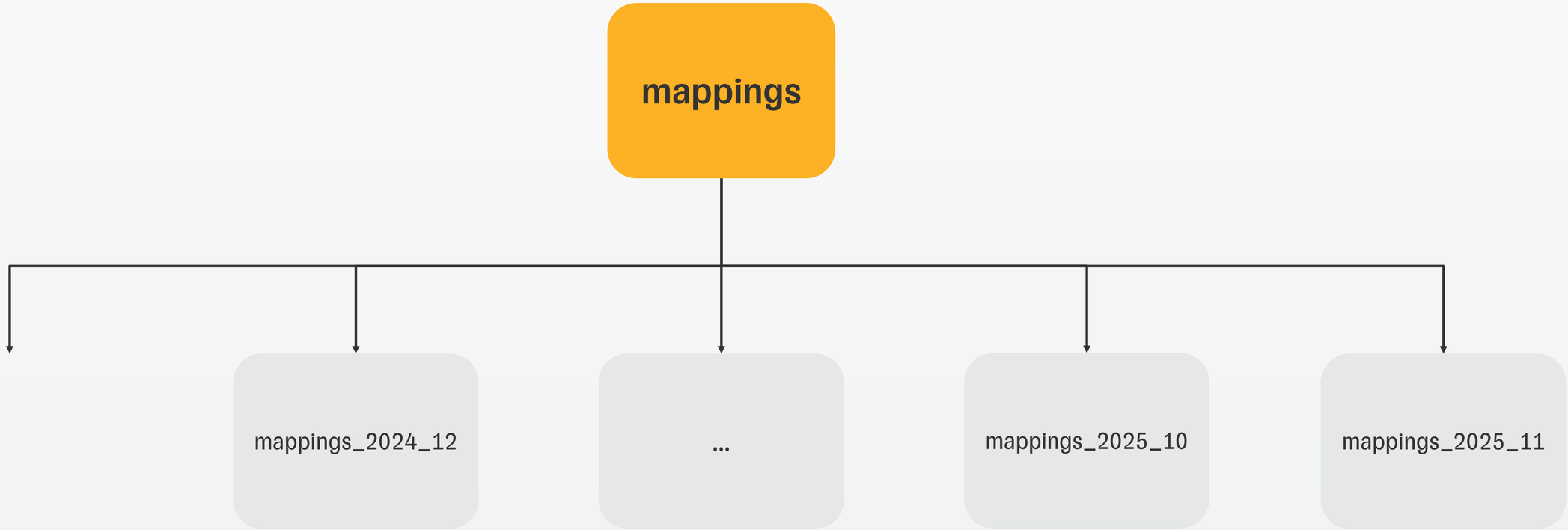
03

**Удалять партициями**

# Таблица связок



# Таблица связок



# Декларативное партиционирование

# Создание партицированной таблицы

- Нужно использовать конструкцию **PARTITION BY**;

create_partition_table.sql ×

```
CREATE TABLE mappings (
    id UUID NOT NULL,
    agent_number VARCHAR(20) NOT NULL,
    client_number VARCHAR(20) NOT NULL,
    virtual_number VARCHAR(20) NOT NULL,
    start_date TIMESTAMPTZ NOT NULL,
    end_date TIMESTAMPTZ
) PARTITION BY RANGE (end_date);
```

# Создание партицированной таблицы

- Нужно использовать конструкцию **PARTITION BY**;
- Указать форму партиционирования. Существует три формы: **RANGE**, **LIST**, **HASH**;

create_partition_table.sql ×

```
CREATE TABLE mappings (
    id UUID NOT NULL,
    agent_number VARCHAR(20) NOT NULL,
    client_number VARCHAR(20) NOT NULL,
    virtual_number VARCHAR(20) NOT NULL,
    start_date TIMESTAMPTZ NOT NULL,
    end_date TIMESTAMPTZ
) PARTITION BY RANGE (end_date);
```

# Создание партицированной таблицы

- Нужно использовать конструкцию **PARTITION BY**;
- Указать форму партиционирования. Существует три формы: **RANGE**, **LIST**, **HASH**;
- Указать ключ партиционирования – одну из колонок либо несколько или выражение.

create_partition_table.sql ×

```
CREATE TABLE mappings (
    id UUID NOT NULL,
    agent_number VARCHAR(20) NOT NULL,
    client_number VARCHAR(20) NOT NULL,
    virtual_number VARCHAR(20) NOT NULL,
    start_date TIMESTAMPTZ NOT NULL,
    end_date TIMESTAMPTZ
) PARTITION BY RANGE (end_date);
```

# Создание партиций

- Нужно использовать конструкцию **PARTITION OF** и указать имя партицированной таблицы к которой она будет прикреплена;

create_partitions.sql ×

```
CREATE TABLE mappings_2025_11
PARTITION OF mappings
FOR VALUES
    FROM ('2025-11-01 00:00:00+03')
    TO ('2025-12-01 00:00:00+03');

CREATE TABLE mappings_2025_12
PARTITION OF mappings
FOR VALUES
    FROM ('2025-12-01 00:00:00+03')
    TO (‘2026-01-01 00:00:00+03');
```

# Создание партиций

- Нужно использовать конструкцию **PARTITION OF** и указать имя партицированной таблицы к которой она будет прикреплена;
- Указать диапазон значений. **FROM** включительно

create_partitions.sql

```
CREATE TABLE mappings_2025_11
PARTITION OF mappings
FOR VALUES
  FROM ('2025-11-01 00:00:00+03')
  TO ('2025-12-01 00:00:00+03');

CREATE TABLE mappings_2025_12
PARTITION OF mappings
FOR VALUES
  FROM ('2025-12-01 00:00:00+03')
  TO ('2026-01-01 00:00:00+03');
```

# Создание партиций

- Нужно использовать конструкцию **PARTITION OF** и указать имя партицированной таблицы к которой она будет прикреплена;
- Указать диапазон значений. **FROM** включительно, **TO** не включительно;

create_partitions.sql

```
CREATE TABLE mappings_2025_11
PARTITION OF mappings
FOR VALUES
    FROM ('2025-11-01 00:00:00+03')
    TO ('2025-12-01 00:00:00+03');

CREATE TABLE mappings_2025_12
PARTITION OF mappings
FOR VALUES
    FROM ('2025-12-01 00:00:00+03')
    TO ('2026-01-01 00:00:00+03');
```

# Создание партиций

- Нужно использовать конструкцию **PARTITION OF** и указать имя партицированной таблицы к которой она будет прикреплена;
- Указать диапазон значений. **FROM** включительно, **TO** не включительно;
- Также можно создать партицию по умолчанию.

create_partitions.sql

```sql
CREATE TABLE mappings_2025_11
PARTITION OF mappings
FOR VALUES
    FROM ('2025-11-01 00:00:00+03')
    TO ('2025-12-01 00:00:00+03');

CREATE TABLE mappings_2025_12
PARTITION OF mappings
FOR VALUES
    FROM ('2025-12-01 00:00:00+03')
    TO (‘2026-01-01 00:00:00+03');

CREATE TABLE mappings_default
PARTITION OF mappings DEFAULT;
```

# DDL операции

ddl_operations.sql ×

# DDL операции

ddl_operations.sql ×

```sql
ALTER TABLE mappings
ADD COLUMN "comment" TEXT;
```

# DDL операции

ddl_operations.sql ×

```sql
ALTER TABLE mappings
ADD COLUMN "comment" TEXT;

CREATE INDEX mappings_agent_number_virtual_number_idx
ON mappings (agent_number, virtual_number);
```

# DDL операции

ddl_operations.sql ×

```sql
ALTER TABLE mappings
ADD COLUMN "comment" TEXT;

CREATE INDEX mappings_agent_number_virtual_number_idx
ON mappings (agent_number, virtual_number);

CREATE UNIQUE INDEX mappings_id_uq
ON mappings (id);
```

# DDL операции

ddl_operations.sql ×

```sql
ALTER TABLE mappings
ADD COLUMN "comment" TEXT;

CREATE INDEX mappings_agent_number_virtual_number_idx
ON mappings (agent_number, virtual_number);

CREATE UNIQUE INDEX mappings_id_uq
ON mappings (id);
```

# DDL операции

ddl_operations.sql ×

```sql
ALTER TABLE mappings
ADD COLUMN "comment" TEXT;

CREATE INDEX mappings_agent_number_virtual_number_idx
ON mappings (agent_number, virtual_number);

CREATE UNIQUE INDEX mappings_id_uq
ON mappings (id);

CREATE UNIQUE INDEX mappings_id_end_date_uq
ON mappings (id, end_date);
```

# DDL операции

ddl_operations.sql ×

```sql
ALTER TABLE mappings
ADD COLUMN "comment" TEXT;

CREATE INDEX mappings_agent_number_virtual_number_idx
ON mappings (agent_number, virtual_number);

CREATE UNIQUE INDEX mappings_id_uq
ON mappings (id);

CREATE UNIQUE INDEX mappings_id_end_date_uq
ON mappings (id, end_date);

CREATE UNIQUE INDEX mappings_2025_11_id_uq
ON mappings_2025_11 (id);
```

# Партицированная таблица – шаблон для партиций.

# Управление партициями

manage_partitions.sql ×

# Управление партициями

manage_partitions.sql ×

```
CREATE TABLE mappings_2025_10 (
    id UUID NOT NULL,
    agent_number VARCHAR(20) NOT NULL,
    client_number VARCHAR(20) NOT NULL,
    virtual_number VARCHAR(20) NOT NULL,
    start_date TIMESTAMPTZ NOT NULL,
    end_date TIMESTAMPTZ,
    “comment” TEXT
);
```

# Управление партициями

manage_partitions.sql ×

```sql
CREATE TABLE mappings_2025_10 (
    id UUID NOT NULL,
    agent_number VARCHAR(20) NOT NULL,
    client_number VARCHAR(20) NOT NULL,
    virtual_number VARCHAR(20) NOT NULL,
    start_date TIMESTAMPTZ NOT NULL,
    end_date TIMESTAMPTZ,
    “comment” TEXT
);

ALTER TABLE mappings ATTACH PARTITION mappings_2025_10
FOR VALUES FROM ('2025-10-01 00:00:00+03') TO ('2025-11-01 00:00:00+03');
```

# Управление партициями

manage_partitions.sql ×

```sql
CREATE TABLE mappings_2025_10 (
    id UUID NOT NULL,
    agent_number VARCHAR(20) NOT NULL,
    client_number VARCHAR(20) NOT NULL,
    virtual_number VARCHAR(20) NOT NULL,
    start_date TIMESTAMPTZ NOT NULL,
    end_date TIMESTAMPTZ,
    "comment" TEXT
);

ALTER TABLE mappings ATTACH PARTITION mappings_2025_10
FOR VALUES FROM ('2025-10-01 00:00:00+03') TO ('2025-11-01 00:00:00+03');

ALTER TABLE mappings
DETACH PARTITION mappings_2025_10;
```

# Управление партициями

manage_partitions.sql ×

```sql
CREATE TABLE mappings_2025_10 (
    id UUID NOT NULL,
    agent_number VARCHAR(20) NOT NULL,
    client_number VARCHAR(20) NOT NULL,
    virtual_number VARCHAR(20) NOT NULL,
    start_date TIMESTAMPTZ NOT NULL,
    end_date TIMESTAMPTZ,
    "comment" TEXT
);

ALTER TABLE mappings ATTACH PARTITION mappings_2025_10
FOR VALUES FROM ('2025-10-01 00:00:00+03') TO ('2025-11-01 00:00:00+03');

ALTER TABLE mappings
DETACH PARTITION mappings_2025_10;

DROP TABLE mappings_2025_10;
```

# Вставка данных

insert_data.sql ×

# Вставка данных

insert_data.sql ×

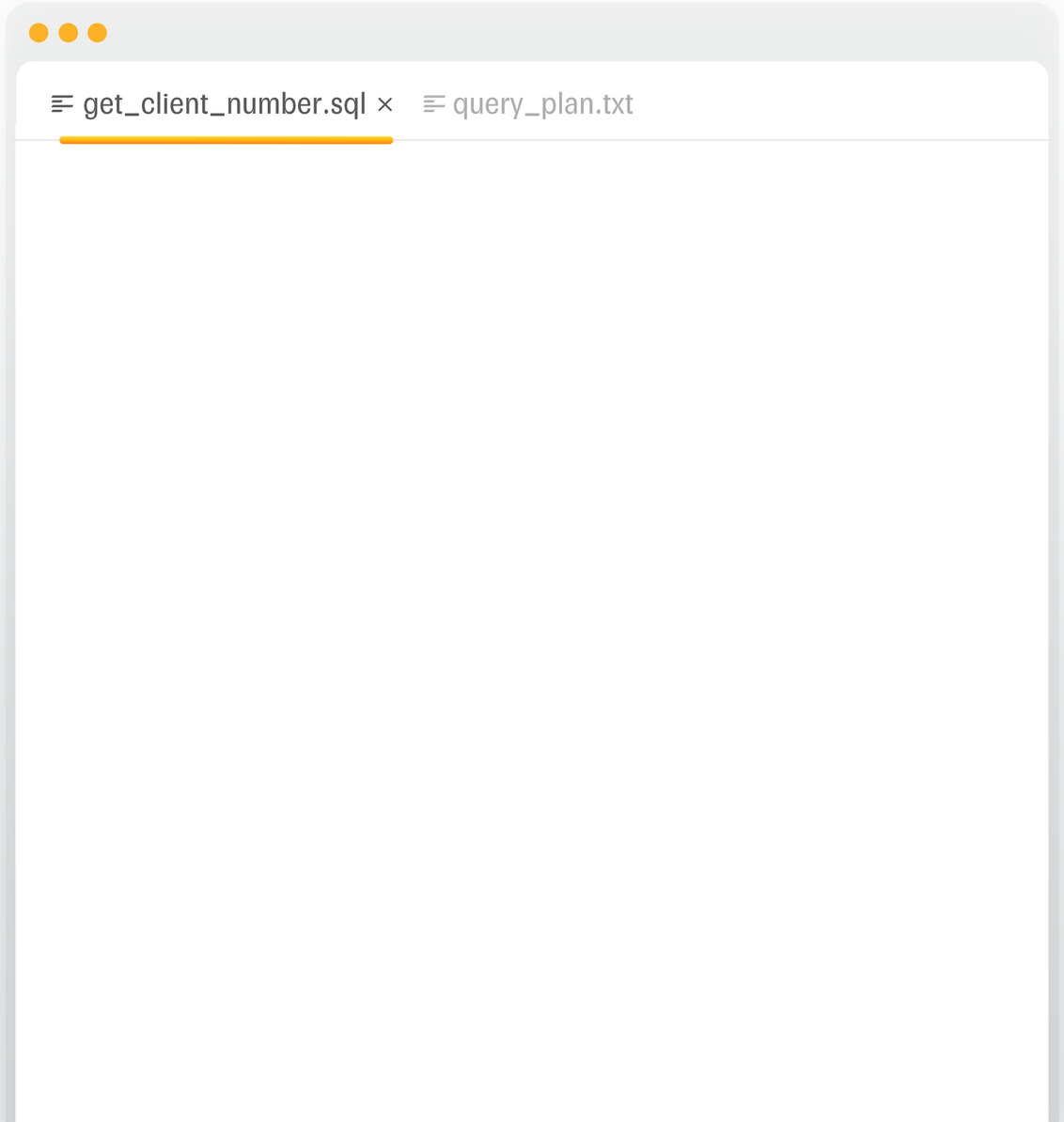
```
INSERT INTO mappings (..., end_date)
VALUES (..., '2025-11-21 19:00:00+03');
```

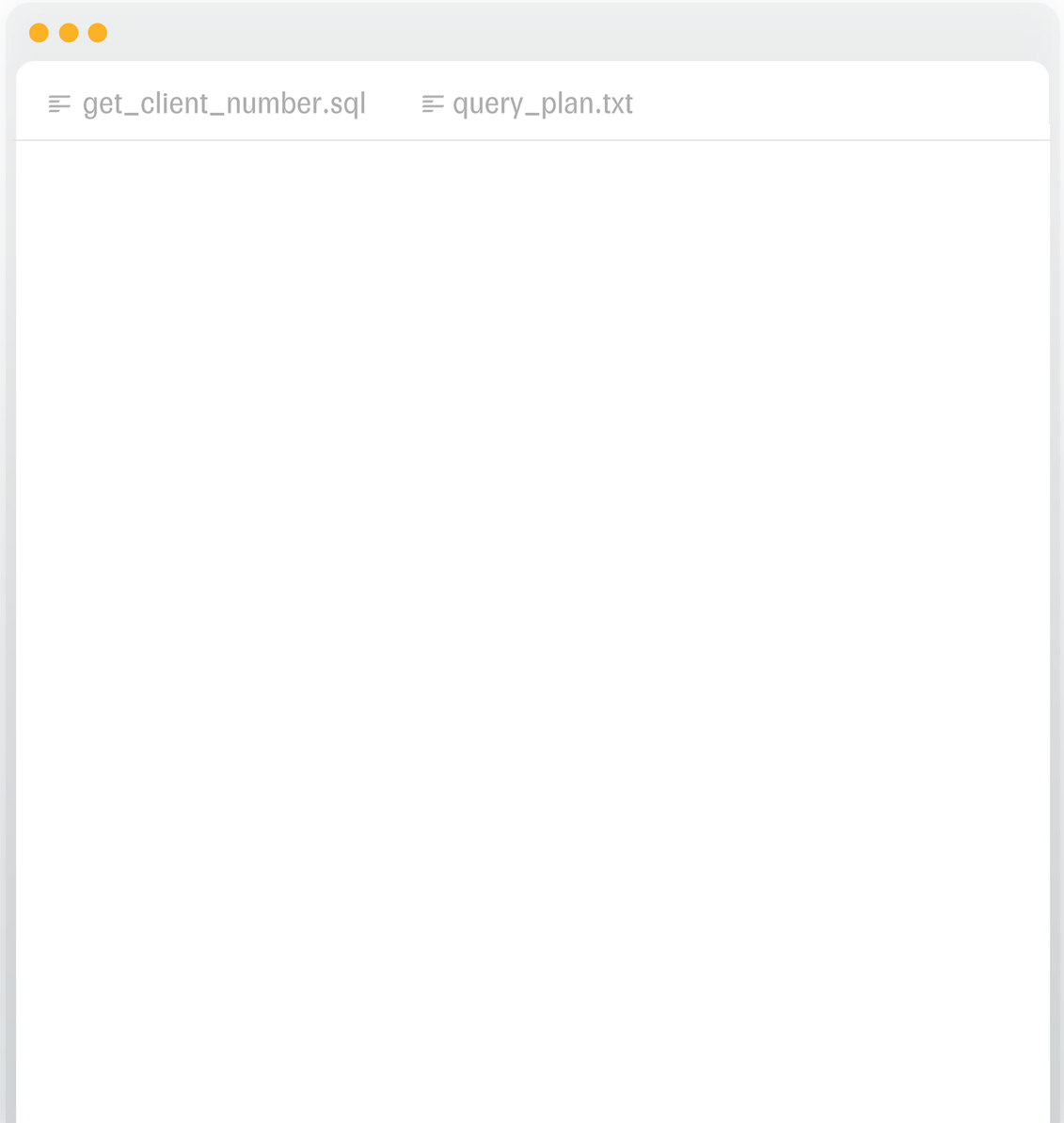
# Вставка данных

insert_data.sql ×

```sql
INSERT INTO mappings (..., end_date)
VALUES (..., '2025-11-21 19:00:00+03');

INSERT INTO mappings (..., end_date)
VALUES (..., '2077-01-01 19:00:00+03');
```

# Чтение данных



# Чтение данных

get_client_number.sql × query_plan.txt

```
SELECT
    client_number
FROM
    mappings
WHERE
    agent_number = '+79780000002'
    AND virtual_number = '+79260000001';
```

get_client_number.sql query_plan.txt

# Чтение данных

get_client_number.sql × query_plan.txt

```
SELECT
    client_number
FROM
    mappings
WHERE
    agent_number = '+79780000002'
    AND virtual_number = '+79260000001';
```



get_client_number.sql query_plan.txt ×

```
Append
  ->  Index Scan using mappings_2025_01_agent_number_virtual_number_idx on mappings_2025_01 mappings_1
        Index Cond: (((agent_number)::text = '+79780000002'::text) AND ((virtual_number)::text = '+79260000001'::text))
  ->  Index Scan using mappings_2025_02_agent_number_virtual_number_idx on mappings_2025_02 mappings_2
        Index Cond: (((agent_number)::text = '+79780000002'::text) AND ((virtual_number)::text = '+79260000001'::text))
  ->  Index Scan using mappings_2025_03_agent_number_virtual_number_idx on mappings_2025_03 mappings_3
        Index Cond: (((agent_number)::text = '+79780000002'::text) AND ((virtual_number)::text = '+79260000001'::text))
  ->  Index Scan using mappings_2025_04_agent_number_virtual_number_idx on mappings_2025_04 mappings_4
        Index Cond: (((agent_number)::text = '+79780000002'::text) AND ((virtual_number)::text = '+79260000001'::text))
  ->  Index Scan using mappings_2025_05_agent_number_virtual_number_idx on mappings_2025_05 mappings_5
        Index Cond: (((agent_number)::text = '+79780000002'::text) AND ((virtual_number)::text = '+79260000001'::text))
  ->  Index Scan using mappings_2025_06_agent_number_virtual_number_idx on mappings_2025_06 mappings_6
        Index Cond: (((agent_number)::text = '+79780000002'::text) AND ((virtual_number)::text = '+79260000001'::text))
  ->  Index Scan using mappings_2025_07_agent_number_virtual_number_idx on mappings_2025_07 mappings_7
        Index Cond: (((agent_number)::text = '+79780000002'::text) AND ((virtual_number)::text = '+79260000001'::text))
  ->  Index Scan using mappings_2025_08_agent_number_virtual_number_idx on mappings_2025_08 mappings_8
        Index Cond: (((agent_number)::text = '+79780000002'::text) AND ((virtual_number)::text = '+79260000001'::text))
  ->  Index Scan using mappings_2025_09_agent_number_virtual_number_idx on mappings_2025_09 mappings_9
        Index Cond: (((agent_number)::text = '+79780000002'::text) AND ((virtual_number)::text = '+79260000001'::text))
  ->  Index Scan using mappings_2025_10_agent_number_virtual_number_idx on mappings_2025_10 mappings_10
        Index Cond: (((agent_number)::text = '+79780000002'::text) AND ((virtual_number)::text = '+79260000001'::text))
  ->  Index Scan using mappings_2025_11_agent_number_virtual_number_idx on mappings_2025_11 mappings_11
        Index Cond: (((agent_number)::text = '+79780000002'::text) AND ((virtual_number)::text = '+79260000001'::text))
  ->  Index Scan using mappings_2025_12_agent_number_virtual_number_idx on mappings_2025_12 mappings_12
        Index Cond: (((agent_number)::text = '+79780000002'::text) AND ((virtual_number)::text = '+79260000001'::text))
  ->  Index Scan using mappings_default_agent_number_virtual_number_idx on mappings_default mappings_13
        Index Cond: (((agent_number)::text = '+79780000002'::text) AND ((virtual_number)::text = '+79260000001'::text))
```

# Чтение данных

get_client_number.sql × query_plan.txt

```sql
SELECT
    client_number
FROM
    mappings
WHERE
    agent_number = '+79780000002'
    AND virtual_number = '+79260000001'
    AND end_date >= '2025-11-01 00:00:00+03'
    AND end_date < '2025-12-01 00:00:00+03';
```



get_client_number.sql query_plan.txt ×

```
Index Scan using mappings_2025_11_agent_number_virtual_number_idx on mappings_2025_11 mappings
  Index Cond: (((agent_number)::text = '+79780000002'::text) AND ((virtual_number)::text = '+79260000001'::text))
  Filter: ((end_date >= '2025-11-01 00:00:00+03'::timestamp with time zone) AND (end_date < '2025-12-01
00:00:00+03'::timestamp with time zone))
```

# Обновление данных

update_mappings.sql ×

# Обновление данных

- Если обновить значение ключа партиционирования – то запись автоматически переедет в нужную партицию;

update_mappings.sql ×

```sql
UPDATE
    mappings
SET
    end_date = '2025-12-31 23:59:59+03'
WHERE
    id = 'a9e0bfa5-01c8-4385-9a6f-d7f6246deb02';
```

# Обновление данных

- Если обновить значение ключа партиционирования – то запись автоматически переедет в нужную партицию;
- И также лучше конкретизировать в какой партиции стоит искать запись.

update_mappings.sql ×

```
UPDATE
    mappings
SET
    end_date = '2025-12-31 23:59:59+03'
WHERE
    id = 'a9e0bfa5-01c8-4385-9a6f-d7f6246deb02’
   AND end_date >= '2025-11-01 00:00:00+03'
   AND end_date < '2025-12-01 00:00:00+03';
```

# Удаление данных

delete_mappings.sql ×

```
DELETE
FROM
    mappings
WHERE
    id = 'a9e0bfa5-01c8-4385-9a6f-d7f6246deb02’
   AND end_date >= '2025-11-01 00:00:00+03'
   AND end_date < '2025-12-01 00:00:00+03’;
```

# Партицированная таблица – роутер к данным.

# Что важно запомнить

01 Партицированная таблица – шаблон для партиций.

# Что важно запомнить

01 Партицированная таблица – шаблон для партиций.

02 **Партицированная таблица – роутер к данным.**

# Что важно запомнить

01 Партицированная таблица – шаблон для партиций.

02 Партицированная таблица – роутер к данным.

**03 Синтаксически запросы к партицированной таблице такие же.**

# Что важно запомнить

01 Партицированная таблица – шаблон для партиций.

02 Партицированная таблица – роутер к данным.

03 Синтаксически запросы к партицированной таблице такие же.

04 **PK и уникальные индексы требуют включения в них ключа партиционирования.**

# Что важно запомнить

01 Партицированная таблица – шаблон для партиций.

02 Партицированная таблица – роутер к данным.

03 Синтаксически запросы к партицированной таблице такие же.

04 PK и уникальные индексы требуют включения в них ключа партиционирования.

05 **Партиции могут иметь свои собственные индексы.**

# Что важно запомнить

01 Партицированная таблица – шаблон для партиций.

02 Партицированная таблица – роутер к данным.

03 Синтаксически запросы к партицированной таблице такие же.

04 PK и уникальные индексы требуют включения в них ключа партиционирования.

05 Партиции могут иметь свои собственные индексы.

06 **Думаем над тем какие партиции будут участвовать в запросе.**

# А как создавать партиции автоматически?

# Варианты автоматического создания партиций

## Средствами базы

Передать ответственность за партиции на базу данных.

## Логикой приложения

Передать ответственность за партиции на приложение.

# Автоматическое создание партиций базой данных

### Триггером

Перед каждой вставкой проверять, что нужная партиция существует

# Автоматическое создание партиций базой данных

### Триггером

Перед каждой вставкой проверять, что нужная партиция существует

### pg_partman

Интересно, но нет возможности установить данное расширение

# Автоматическое создание партиций базой данных

### Триггером

Перед каждой вставкой проверять, что нужная партиция существует

### pg_partman

Интересно, но нет возможности установить данное расширение

### pg_cron

Расширение можно установить. Нужно только написать процедуры.

# Процедура создания партиции

create_partition_procedure.sql ×

# Процедура создания партиции

create_partition_procedure.sql ×

```
CREATE OR REPLACE PROCEDURE create_mapping_partition(start_date DATE)
LANGUAGE plpgsql
AS $body$
DECLARE
  partition_name TEXT := format('mappings_%1$s', TO_CHAR(start_date, 'YYYY_MM'));
  full_partition_name TEXT := format('%1$s', partition_name);
  is_partition_exists BOOLEAN := FALSE;
  end_date DATE := start_date + INTERVAL '1 month';
BEGIN
  is_partition_exists := EXISTS (
    SELECT FROM pg_tables
    WHERE tablename = partition_name AND schemaname = 'public'
  );

  IF is_partition_exists THEN
    RAISE NOTICE 'Partition % has already exist', full_partition_name;
    RETURN;
  END IF;

  EXECUTE format('CREATE TABLE %1$s PARTITION OF %2$s FOR VALUES FROM (%3$L) TO (%4$L)', full_partition_name, table_name, start_date, end_date);
END;
$body$
```

# Процедура создания партиции

create_partition_procedure.sql ×

```
CREATE OR REPLACE PROCEDURE create_mapping_partition(start_date DATE)
LANGUAGE plpgsql
AS $body$
DECLARE
  partition_name TEXT := format('mappings_%1$s', TO_CHAR(start_date, 'YYYY_MM'));
  full_partition_name TEXT := format('%1$s', partition_name);
  is_partition_exists BOOLEAN := FALSE;
  end_date DATE := start_date + INTERVAL '1 month';
BEGIN
  is_partition_exists := EXISTS (
    SELECT FROM pg_tables
    WHERE tablename = partition_name AND schemaname = 'public'
  );

  IF is_partition_exists THEN
    RAISE NOTICE 'Partition % has already exist', full_partition_name;
    RETURN;
  END IF;

  EXECUTE format('CREATE TABLE %1$s PARTITION OF %2$s FOR VALUES FROM (%3$L) TO (%4$L)', full_partition_name, table_name, start_date, end_date);
END;
$body$
```

# Процедура создания партиции

create_partition_procedure.sql ×

```
CREATE OR REPLACE PROCEDURE create_mapping_partition(start_date DATE)
LANGUAGE plpgsql
AS $body$
DECLARE
  partition_name TEXT := format('mappings_%1$s', TO_CHAR(start_date, 'YYYY_MM'));
  full_partition_name TEXT := format('%1$s', partition_name);
  is_partition_exists BOOLEAN := FALSE;
  end_date DATE := start_date + INTERVAL '1 month';
BEGIN
  is_partition_exists := EXISTS (
    SELECT FROM pg_tables
    WHERE tablename = partition_name AND schemaname = 'public'
  );

  IF is_partition_exists THEN
    RAISE NOTICE 'Partition % has already exist', full_partition_name;
    RETURN;
  END IF;

  EXECUTE format('CREATE TABLE %1$s PARTITION OF %2$s FOR VALUES FROM (%3$L) TO (%4$L)', full_partition_name, table_name, start_date, end_date);
END;
$body$
```

# Процедура создания партиции

create_partition_procedure.sql ×

```
CREATE OR REPLACE PROCEDURE create_mapping_partition(start_date DATE)
LANGUAGE plpgsql
AS $body$
DECLARE
  partition_name TEXT := format('mappings_%1$s', TO_CHAR(start_date, 'YYYY_MM'));
  full_partition_name TEXT := format('%1$s', partition_name);
  is_partition_exists BOOLEAN := FALSE;
  end_date DATE := start_date + INTERVAL '1 month';
BEGIN
  is_partition_exists := EXISTS (
    SELECT FROM pg_tables
    WHERE tablename = partition_name AND schemaname = 'public'
  );

  IF is_partition_exists THEN
    RAISE NOTICE 'Partition % has already exist', full_partition_name;
    RETURN;
  END IF;

  EXECUTE format('CREATE TABLE %1$s PARTITION OF %2$s FOR VALUES FROM (%3$L) TO (%4$L)', full_partition_name, table_name, start_date, end_date);
END;
$body$
```

# Процедура создания партиции

create_partition_procedure.sql ×

```
CREATE OR REPLACE PROCEDURE create_mapping_partition(start_date DATE)
LANGUAGE plpgsql
AS $body$
DECLARE
  partition_name TEXT := format('mappings_%1$s', TO_CHAR(start_date, 'YYYY_MM'));
  full_partition_name TEXT := format('%1$s', partition_name);
  is_partition_exists BOOLEAN := FALSE;
  end_date DATE := start_date + INTERVAL '1 month';
BEGIN
  is_partition_exists := EXISTS (
    SELECT FROM pg_tables
    WHERE tablename = partition_name AND schemaname = 'public'
  );

  IF is_partition_exists THEN
    RAISE NOTICE 'Partition % has already exist', full_partition_name;
    RETURN;
  END IF;

  EXECUTE format('CREATE TABLE %1$s PARTITION OF %2$s FOR VALUES FROM (%3$L) TO (%4$L)', full_partition_name, table_name, start_date, end_date);
END;
$body$
```

# Процедура создания партиции

create_partition_procedure.sql ×

```
CREATE OR REPLACE PROCEDURE create_mapping_partition(start_date DATE)
LANGUAGE plpgsql
AS $body$
DECLARE
  partition_name TEXT := format('mappings_%1$s', TO_CHAR(start_date, 'YYYY_MM'));
  full_partition_name TEXT := format('%1$s', partition_name);
  is_partition_exists BOOLEAN := FALSE;
  end_date DATE := start_date + INTERVAL '1 month';
BEGIN
  is_partition_exists := EXISTS (
    SELECT FROM pg_tables
    WHERE tablename = partition_name AND schemaname = 'public'
  );

  IF is_partition_exists THEN
    RAISE NOTICE 'Partition % has already exist', full_partition_name;
    RETURN;
  END IF;

  EXECUTE format('CREATE TABLE %1$s PARTITION OF %2$s FOR VALUES FROM (%3$L) TO (%4$L)', full_partition_name, table_name, start_date, end_date);
END;
$body$
```

# Процедура создания партиции

create_partition_procedure.sql ×

```
CREATE OR REPLACE PROCEDURE create_mapping_partition(start_date DATE)
LANGUAGE plpgsql
AS $body$
DECLARE
  partition_name TEXT := format('mappings_%1$s', TO_CHAR(start_date, 'YYYY_MM'));
  full_partition_name TEXT := format('%1$s', partition_name);
  is_partition_exists BOOLEAN := FALSE;
  end_date DATE := start_date + INTERVAL '1 month';
BEGIN
  is_partition_exists := EXISTS (
    SELECT FROM pg_tables
    WHERE tablename = partition_name AND schemaname = 'public'
  );

  IF is_partition_exists THEN
    RAISE NOTICE 'Partition % has already exist', full_partition_name;
    RETURN;
  END IF;

  EXECUTE format('CREATE TABLE %1$s (LIKE %2$s INCLUDING DEFAULTS INCLUDING CONSTRAINTS)', full_partition_name, 'mappings');
  EXECUTE format('ALTER TABLE %1$s ADD CONSTRAINT end_date_check CHECK (end_date >= (%2$L) AND end_date < (%3$L))', full_partition_name, start_date, end_date);
  EXECUTE format('INSERT INTO %1$s SELECT * FROM %2$s WHERE end_date >= (%3$L) AND end_date < (%4$L)', full_partition_name, 'mappings_default', start_date, end_date);
  EXECUTE format('DELETE FROM %1$s WHERE end_date >= (%2$L) AND end_date < (%3$L)', 'mappings_default', start_date, end_date);
  EXECUTE format('ALTER TABLE %1$s ATTACH PARTITION %2$s FOR VALUES FROM (%3$L) TO (%4$L)', 'mappings', full_partition_name, start_date, end_date);
  EXECUTE format('ALTER TABLE %1$s DROP CONSTRAINT end_date_check', full_partition_name);
END;
$body$
```

# Процедура удаления партиции

drop_partition_procedure.sql ×

# Процедура удаления партиции

drop_partition_procedure.sql ×

```
CREATE OR REPLACE PROCEDURE drop_mapping_partitions (older_than INTERVAL)
LANGUAGE plpgsql
AS $body$
DECLARE
  min_date DATE := DATE_TRUNC('month', CURRENT_DATE - older_than)::DATE;
  partition_name VARCHAR(64);
  partition_date DATE;
  temprow RECORD;
BEGIN
  FOR temprow IN
    SELECT nsp_child.nspname || '.' || child.relname AS partition_name
    FROM pg_inherits
    JOIN pg_class parent ON pg_inherits.inhparent = parent.oid
    JOIN pg_class child ON pg_inherits.inhrelid = child.oid
    JOIN pg_namespace nsp_parent ON nsp_parent.oid = parent.relnamespace
    JOIN pg_namespace nsp_child ON nsp_child.oid = child.relnamespace
    WHERE parent.relname = 'mappings' AND nsp_parent.nspname = 'public' AND nsp_child.nspname = 'public' AND child.relname != 'mappings_default'
    ORDER BY partition_name
  LOOP
    partition_name := temprow.partition_name;
    partition_date := MAKE_DATE(SPLIT_PART(partition_name, '_', 2)::INT, SPLIT_PART(partition_name, '_', 3)::INT, 1);
    IF partition_date <= min_date THEN
      EXECUTE FORMAT('DROP TABLE %1$s', partition_name);
      RAISE NOTICE 'Partition deleted %', partition_name;
    END IF;
  END LOOP;
END;
$body$
```

# Процедура удаления партиции

drop_partition_procedure.sql ×

```
CREATE OR REPLACE PROCEDURE drop_mapping_partitions (older_than INTERVAL)
LANGUAGE plpgsql
AS $body$
DECLARE
  min_date DATE := DATE_TRUNC('month', CURRENT_DATE - older_than)::DATE;
  partition_name VARCHAR(64);
  partition_date DATE;
  temprow RECORD;
BEGIN
  FOR temprow IN
    SELECT nsp_child.nspname || '.' || child.relname AS partition_name
    FROM pg_inherits
    JOIN pg_class parent ON pg_inherits.inhparent = parent.oid
    JOIN pg_class child ON pg_inherits.inhrelid = child.oid
    JOIN pg_namespace nsp_parent ON nsp_parent.oid = parent.relnamespace
    JOIN pg_namespace nsp_child ON nsp_child.oid = child.relnamespace
    WHERE parent.relname = 'mappings' AND nsp_parent.nspname = 'public' AND nsp_child.nspname = 'public' AND child.relname != 'mappings_default'
    ORDER BY partition_name
  LOOP
    partition_name := temprow.partition_name;
    partition_date := MAKE_DATE(SPLIT_PART(partition_name, '_', 2)::INT, SPLIT_PART(partition_name, '_', 3)::INT, 1);
    IF partition_date <= min_date THEN
      EXECUTE FORMAT('DROP TABLE %1$s', partition_name);
      RAISE NOTICE 'Partition deleted %', partition_name;
    END IF;
  END LOOP;
END;
$body$
```

# Процедура удаления партиции

drop_partition_procedure.sql ×

```
CREATE OR REPLACE PROCEDURE drop_mapping_partitions (older_than INTERVAL)
LANGUAGE plpgsql
AS $body$
DECLARE
  min_date DATE := DATE_TRUNC('month', CURRENT_DATE - older_than)::DATE;
  partition_name VARCHAR(64);
  partition_date DATE;
  temprow RECORD;
BEGIN
  FOR temprow IN
    SELECT nsp_child.nspname || '.' || child.relname AS partition_name
    FROM pg_inherits
    JOIN pg_class parent ON pg_inherits.inhparent = parent.oid
    JOIN pg_class child ON pg_inherits.inhrelid = child.oid
    JOIN pg_namespace nsp_parent ON nsp_parent.oid = parent.relnamespace
    JOIN pg_namespace nsp_child ON nsp_child.oid = child.relnamespace
    WHERE parent.relname = 'mappings' AND nsp_parent.nspname = 'public' AND nsp_child.nspname = 'public' AND child.relname != 'mappings_default'
    ORDER BY partition_name
  LOOP
    partition_name := temprow.partition_name;
    partition_date := MAKE_DATE(SPLIT_PART(partition_name, '_', 2)::INT, SPLIT_PART(partition_name, '_', 3)::INT, 1);
    IF partition_date <= min_date THEN
      EXECUTE FORMAT('DROP TABLE %1$s', partition_name);
      RAISE NOTICE 'Partition deleted %', partition_name;
    END IF;
  END LOOP;
END;
$body$
```

# Процедура удаления партиции

drop_partition_procedure.sql ×

```
CREATE OR REPLACE PROCEDURE drop_mapping_partitions (older_than INTERVAL)
LANGUAGE plpgsql
AS $body$
DECLARE
  min_date DATE := DATE_TRUNC('month', CURRENT_DATE - older_than)::DATE;
  partition_name VARCHAR(64);
  partition_date DATE;
  temprow RECORD;
BEGIN
  FOR temprow IN
    SELECT nsp_child.nspname || '.' || child.relname AS partition_name
    FROM pg_inherits
    JOIN pg_class parent ON pg_inherits.inhparent = parent.oid
    JOIN pg_class child ON pg_inherits.inhrelid = child.oid
    JOIN pg_namespace nsp_parent ON nsp_parent.oid = parent.relnamespace
    JOIN pg_namespace nsp_child ON nsp_child.oid = child.relnamespace
    WHERE parent.relname = 'mappings' AND nsp_parent.nspname = 'public' AND nsp_child.nspname = 'public' AND child.relname != 'mappings_default'
    ORDER BY partition_name
  LOOP
    partition_name := temprow.partition_name;
    partition_date := MAKE_DATE(SPLIT_PART(partition_name, '_', 2)::INT, SPLIT_PART(partition_name, '_', 3)::INT, 1);
    IF partition_date <= min_date THEN
      EXECUTE FORMAT('DROP TABLE %1$s', partition_name);
      RAISE NOTICE 'Partition deleted %', partition_name;
    END IF;
  END LOOP;
END;
$body$
```

# Процедура удаления партиции

drop_partition_procedure.sql ×

```
CREATE OR REPLACE PROCEDURE drop_mapping_partitions (older_than INTERVAL)
LANGUAGE plpgsql
AS $body$
DECLARE
  min_date DATE := DATE_TRUNC('month', CURRENT_DATE - older_than)::DATE;
  partition_name VARCHAR(64);
  partition_date DATE;
  temprow RECORD;
BEGIN
  FOR temprow IN
    SELECT nsp_child.nspname || '.' || child.relname AS partition_name
    FROM pg_inherits
    JOIN pg_class parent ON pg_inherits.inhparent = parent.oid
    JOIN pg_class child ON pg_inherits.inhrelid = child.oid
    JOIN pg_namespace nsp_parent ON nsp_parent.oid = parent.relnamespace
    JOIN pg_namespace nsp_child ON nsp_child.oid = child.relnamespace
    WHERE parent.relname = 'mappings' AND nsp_parent.nspname = 'public' AND nsp_child.nspname = 'public' AND child.relname != 'mappings_default'
    ORDER BY partition_name
  LOOP
    partition_name := temprow.partition_name;
    partition_date := MAKE_DATE(SPLIT_PART(partition_name, '_', 2)::INT, SPLIT_PART(partition_name, '_', 3)::INT, 1);
    IF partition_date <= min_date THEN
      EXECUTE FORMAT('DROP TABLE %1$s', partition_name);
      RAISE NOTICE 'Partition deleted %', partition_name;
    END IF;
  END LOOP;
END;
$body$
```

# Планирование запуска процедур

schedule_partition_manage_jobs.sql ×

# Планирование запуска процедур

schedule_partition_manage_jobs.sql ×

```
INSERT INTO cron.job (schedule, command, nodename, nodeport, database, username, jobname)
VALUES (
  '0 0 * * *',
  $$CALL create_mapping_partition(DATE_TRUNC('month', CURRENT_DATE)::DATE)$$,
  '',
  INET_SERVER_PORT(),
  'myDatabase',
  CURRENT_USER,
  'create_mapping_partition');
```

# Планирование запуска процедур

schedule_partition_manage_jobs.sql ×

```
INSERT INTO cron.job (schedule, command, nodename, nodeport, database, username, jobname)
VALUES (
  '0 0 * * *',
  $$CALL create_mapping_partition(DATE_TRUNC('month', CURRENT_DATE)::DATE)$$,
  '',
  INET_SERVER_PORT(),
  'myDatabase',
  CURRENT_USER,
  'create_mapping_partition');

INSERT INTO cron.job (schedule, command, nodename, nodeport, database, username, jobname)
VALUES (
  '5 0 * * *',
  $$CALL drop_mapping_partitions('3 month'::INTERVAL)$$,
  '',
  INET_SERVER_PORT(),
  'myDatabase',
  CURRENT_USER,
  'drop_mapping_partitions');
```

# Нам не понравился результат

# Проблемы

- Процедуры сложно переиспользовать в других сервисах.

# Проблемы

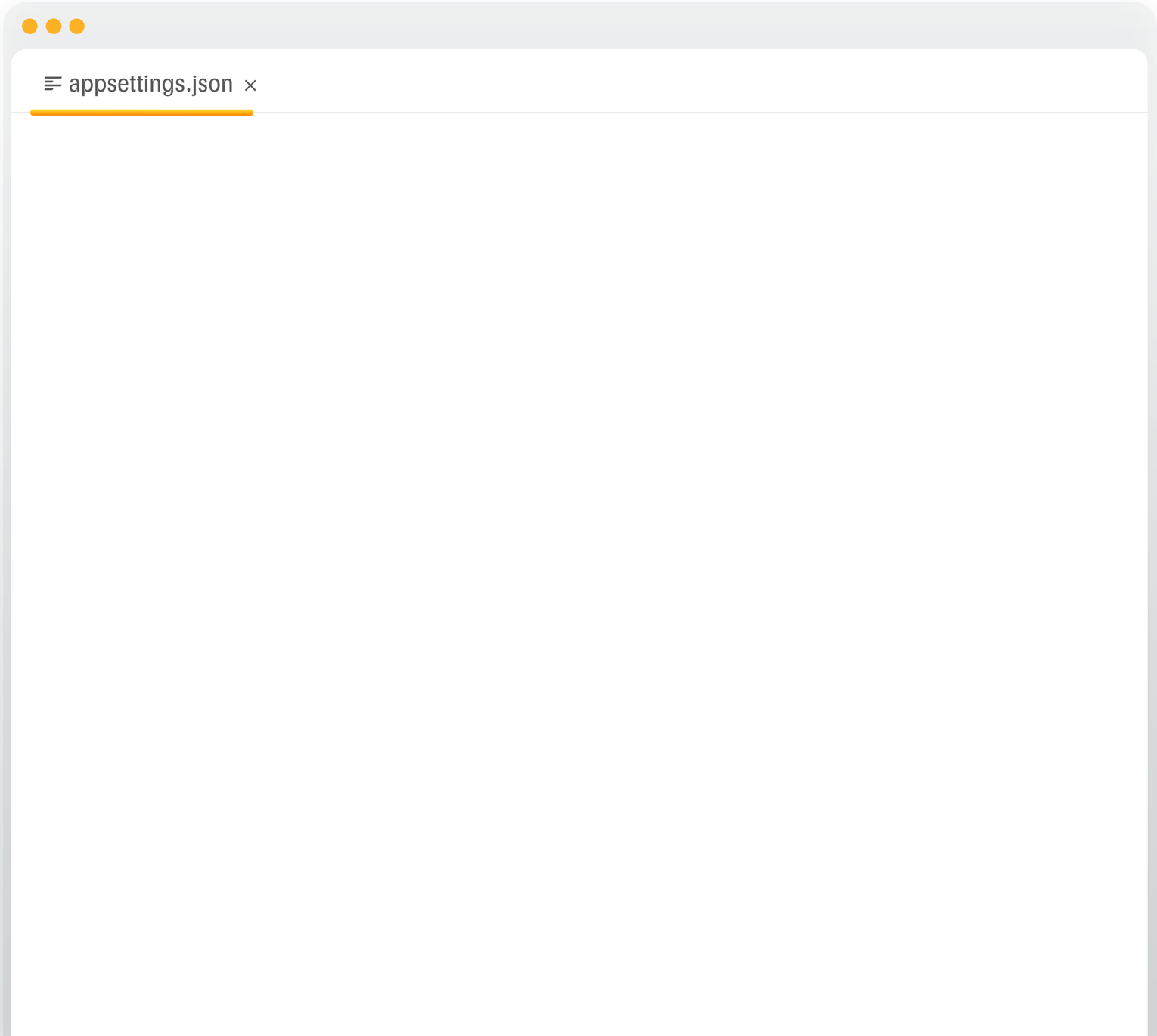
- Процедуры сложно переиспользовать в других сервисах.
- pg_cron устанавливается только в одну базу данных на одном сервере PostgreSQL.

# Проблемы

- Процедуры сложно переиспользовать в других сервисах.
- pg_cron устанавливается только в одну базу данных на одном сервере PostgreSQL.
- Отсутствует мониторинг. Если партиция не создалась, то об этом узнаем не сразу.

# Создаем PartitionManager

# Настройки менеджера партиций



# Настройки менеджера партиций

appsettings.json ×

```
{
  "PartitionManager": {
    "Tables": [
      {
        "Name": "mappings",
        "PartitionRange": "1 month",
        "CreateBefore": "7 days",
        "DropAfter": "1 month"
      }
    ]
  }
}
```

# Настройки менеджера партиций

- **Name** – название партиционированной таблицы;

appsettings.json ×

```
{
  "PartitionManager": {
    "Tables": [
      {
        "Name": "mappings",
        "PartitionRange": "1 month",
        "CreateBefore": "7 days",
        "DropAfter": "1 month"
      }
    ]
  }
}
```

# Настройки менеджера партиций

- **Name** – название партиционированной таблицы;
- **PartitionRange** – размер партиции;

appsettings.json ×

```
{
  "PartitionManager": {
    "Tables": [
      {
        "Name": "mappings",
        "PartitionRange": "1 month",
        "CreateBefore": "7 days",
        "DropAfter": "1 month"
      }
    ]
  }
}
```

# Настройки менеджера партиций

- **Name** – название партиционированной таблицы;
- **PartitionRange** – размер партиции;
- **CreateBefore** – за сколько дней **до** нужно создать партицию;

appsettings.json ×

```
{
  "PartitionManager": {
    "Tables": [
      {
        "Name": "mappings",
        "PartitionRange": "1 month",
        "CreateBefore": "7 days",
        "DropAfter": "1 month"
      }
    ]
  }
}
```

# Настройки менеджера партиций

- **Name** – название партиционированной таблицы;
- **PartitionRange** – размер партиции;
- **CreateBefore** – за сколько дней **до** нужно создать партицию;

appsettings.json

```
{
  "PartitionManager": {
    "Tables": [
      {
        "Name": "mappings",
        "PartitionRange": "1 month",
        "CreateBefore": "1 year",
        "DropAfter": "1 month"
      }
    ]
  }
}
```

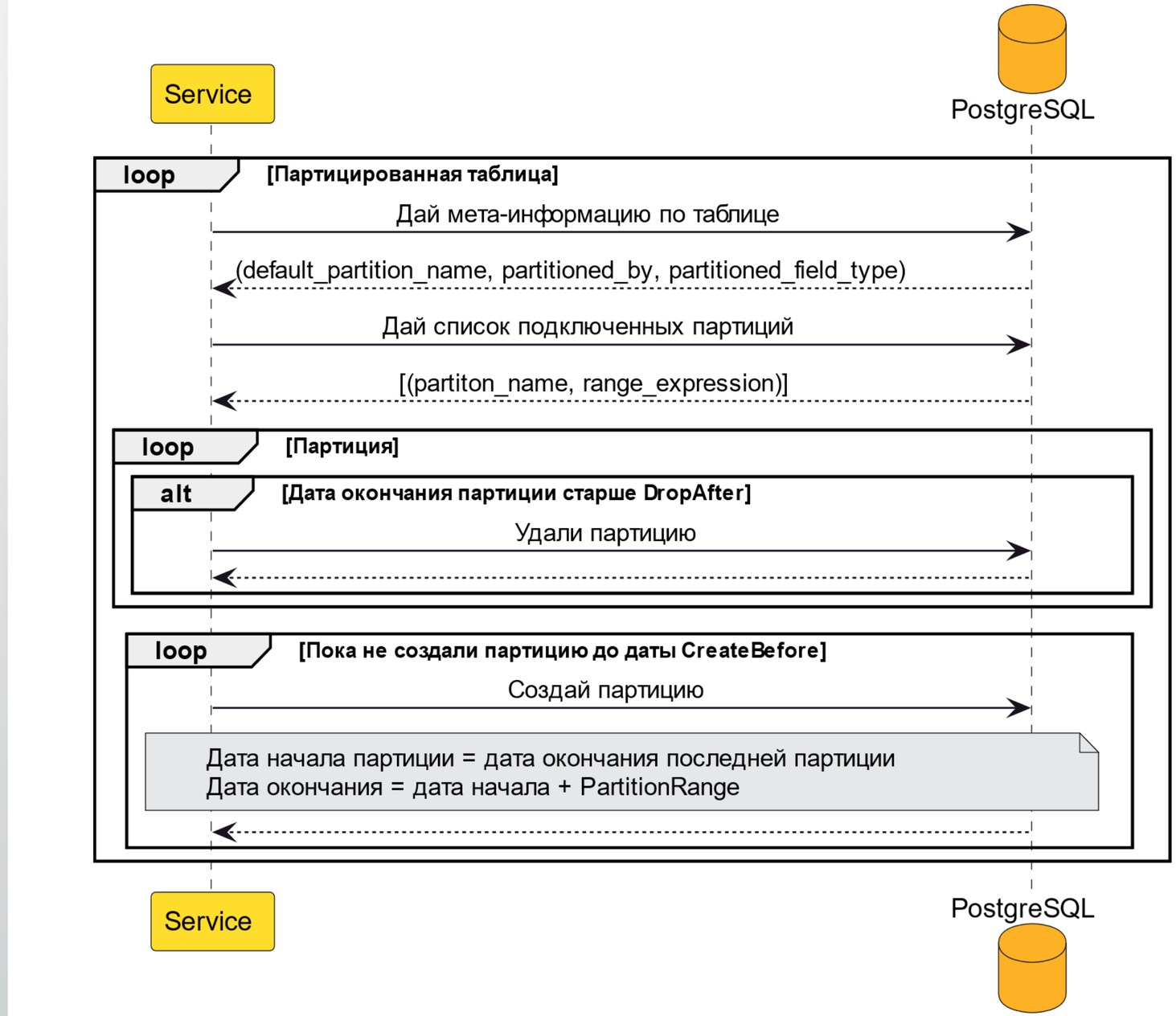
# Настройки менеджера партиций

- **Name** – название партиционированной таблицы;
- **PartitionRange** – размер партиции;
- **CreateBefore** – за сколько дней **ДО** нужно создать партицию;
- **DropAfter** – через сколько **ПОСЛЕ** нужно удалить партицию.

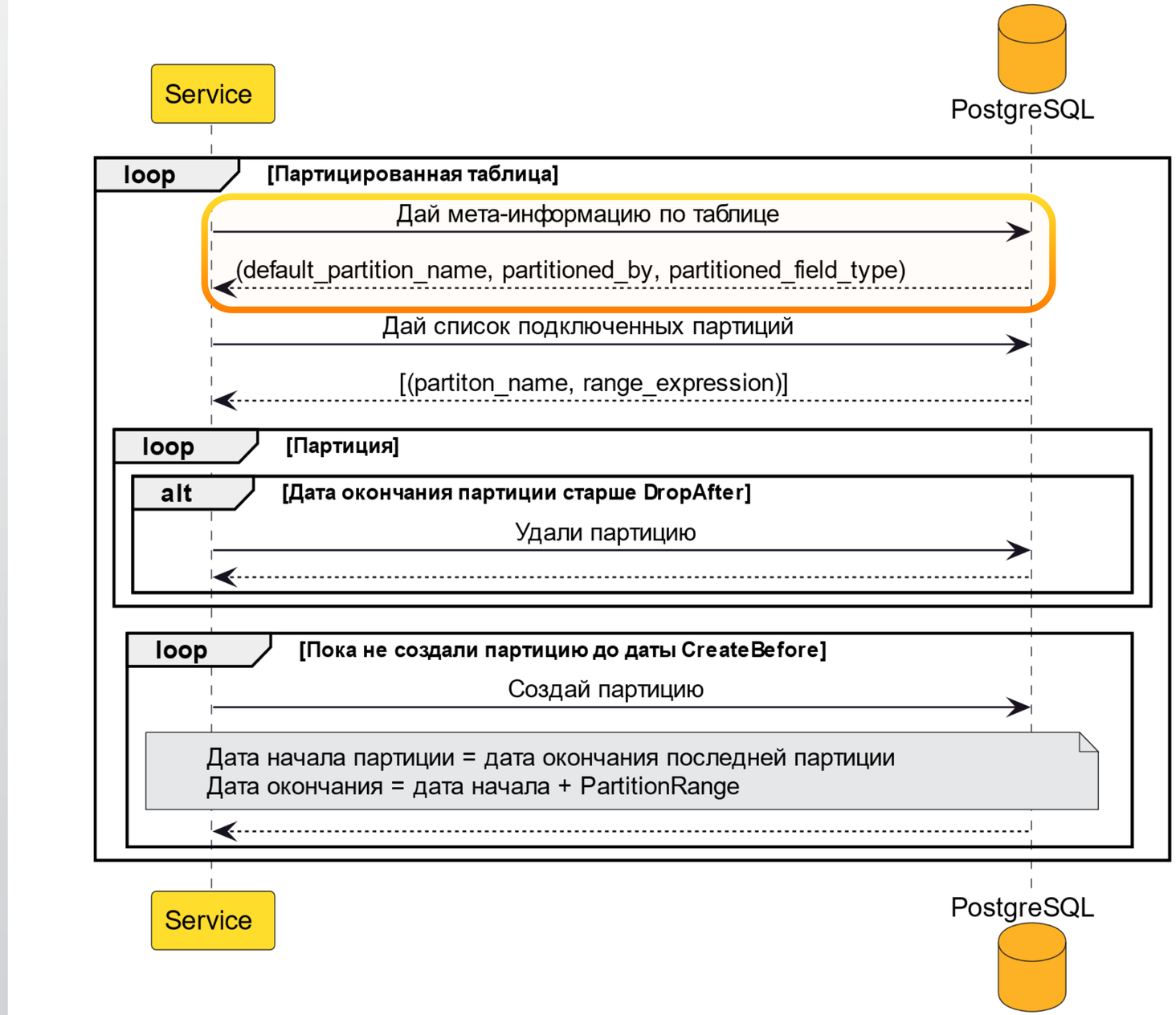
appsettings.json

```
{
  "PartitionManager": {
    "Tables": [
      {
        "Name": "mappings",
        "PartitionRange": "1 month",
        "CreateBefore": "1 year",
        "DropAfter": "1 month"
      }
    ]
  }
}
```

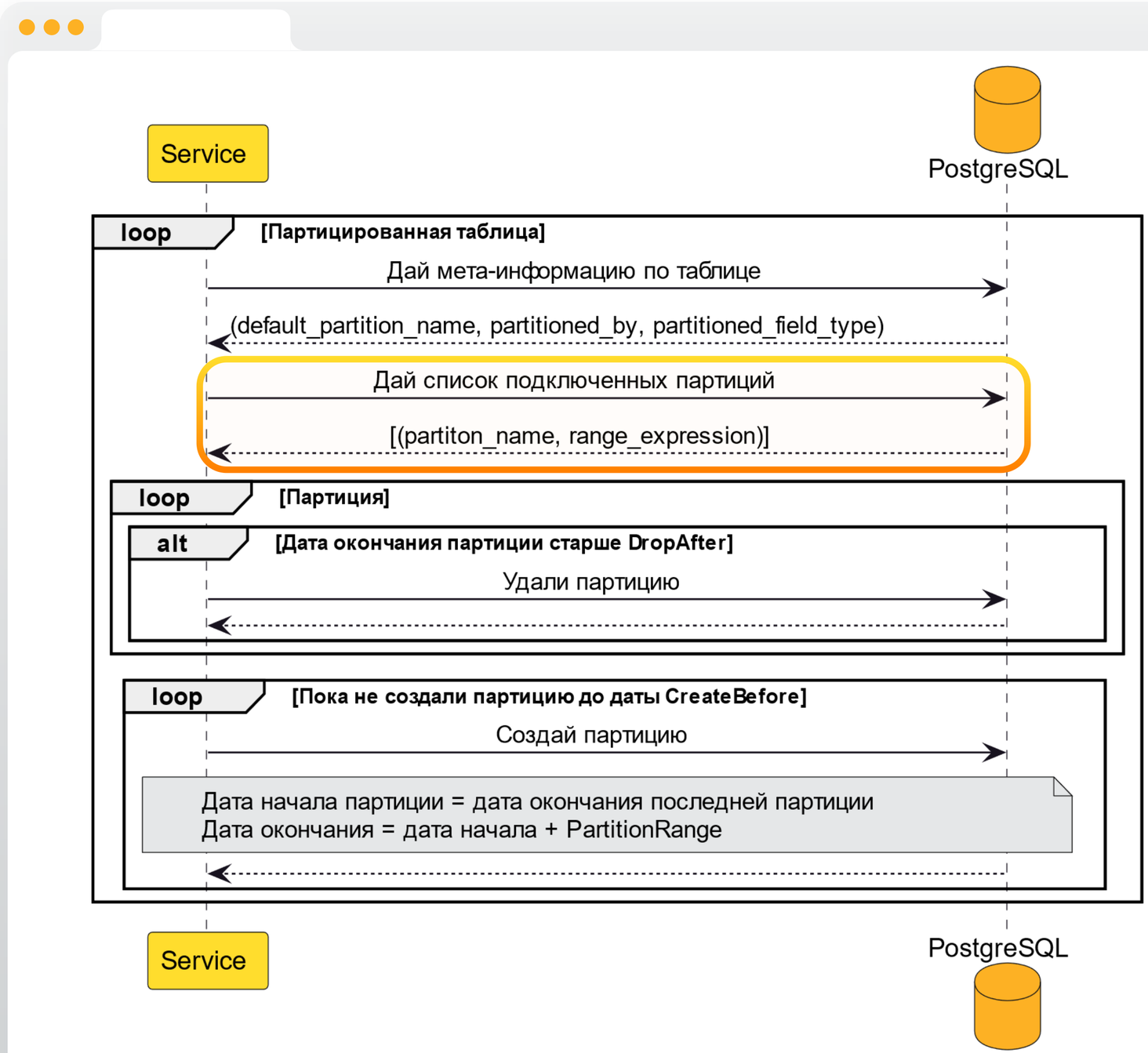
# Алгоритм работы



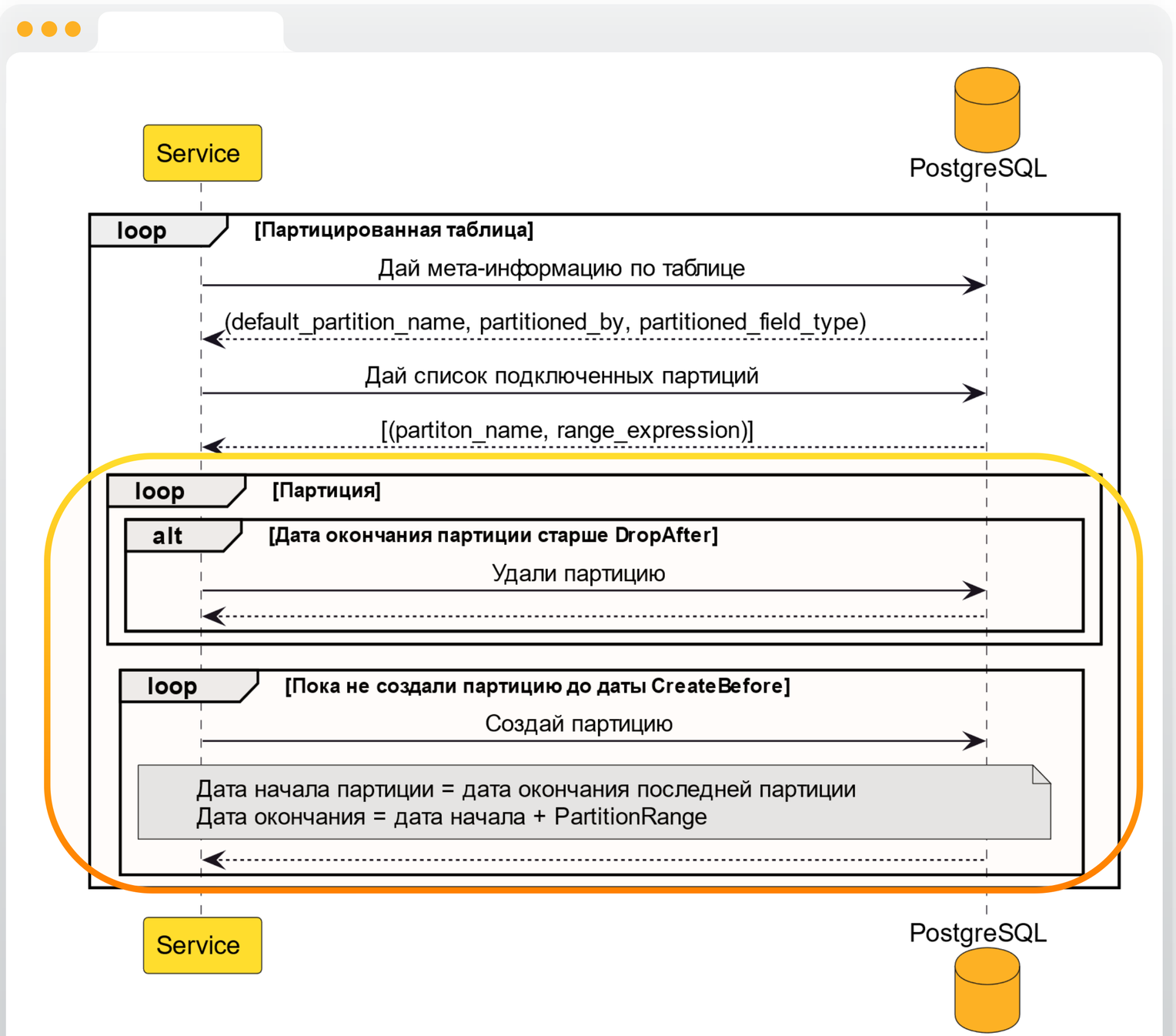
# Алгоритм работы



# Алгоритм работы



# Алгоритм работы



# Запрос для получения меты

get_partitioned_table_meta.sql ×

# Запрос для получения меты

get_partitioned_table_meta.sql ×

```
SELECT
  defpart.relname AS default_partition_name,
  pa.attname AS partitioned_by,
  attr_type.typname AS partioned_by_field_type
FROM pg_partitioned_table pt
JOIN pg_catalog.pg_class parent_class ON pt.partrelid = parent_class.oid
JOIN pg_catalog.pg_namespace parent_nsp ON parent_class.relnamespace = parent_nsp.oid
JOIN pg_catalog.pg_attribute pa ON pt.partrelid = pa.attrelid AND pa.attnum = ANY (pt.partattrs)
JOIN pg_catalog.pg_type attr_type ON pa.atttypid = attr_type.oid
LEFT JOIN pg_catalog.pg_class defpart ON pt.partdefid = defpart.oid
WHERE
  parent_class.relname = 'mappings'
  AND parent_nsp.nspname = 'public'
  AND pt.partstrat = 'r'
```

# Запрос для получения меты

get_partitioned_table_meta.sql ×

```
SELECT
    defpart.relname AS default_partition_name,
    pa.attname AS partitioned_by,
    attr_type.typname AS partioned_by_field_type
FROM pg_partitioned_table pt
JOIN pg_catalog.pg_class parent_class ON pt.partrelid = parent_class.oid
JOIN pg_catalog.pg_namespace parent_nsp ON parent_class.relnamespace = parent_nsp.oid
JOIN pg_catalog.pg_attribute pa ON pt.partrelid = pa.attrelid AND pa.attnum = ANY (pt.partattrs)
JOIN pg_catalog.pg_type attr_type ON pa.atttypid = attr_type.oid
LEFT JOIN pg_catalog.pg_class defpart ON pt.partdefid = defpart.oid
WHERE
    parent_class.relname = 'mappings'
    AND parent_nsp.nspname = 'public'
    AND pt.partstrat = 'r'
```

# Запрос для получения партиций

get_partitions.sql ×

```
SELECT
  child_class.relname AS partition_name,
  pe.part_expr AS range_expression
FROM pg_catalog.pg_partitioned_table pt
JOIN pg_catalog.pg_class parent_class ON pt.partrelid = parent_class.oid
JOIN pg_catalog.pg_namespace parent_nsp ON parent_class.relnamespace = parent_nsp.oid
JOIN pg_catalog.pg_inherits i ON pt.partrelid = i.inhparent
JOIN pg_catalog.pg_class child_class ON i.inhrelid = child_class.oid
LEFT JOIN LATERAL (
  SELECT pg_catalog.pg_get_expr(child_class.relpartbound, child_class.oid) AS part_expr
) AS pe ON TRUE
WHERE
  parent_class.relname = 'mappings'
  AND parent_nsp.nspname = 'public'
  AND pt.partdefid != child_class.oid
```

# Запрос для получения партиций

get_partitions.sql ×

```
SELECT
    child_class.relname AS partition_name,
    pe.part_expr AS range_expression
FROM pg_catalog.pg_partitioned_table pt
JOIN pg_catalog.pg_class parent_class ON pt.partrelid = parent_class.oid
JOIN pg_catalog.pg_namespace parent_nsp ON parent_class.relnamespace = parent_nsp.oid
JOIN pg_catalog.pg_inherits i ON pt.partrelid = i.inhparent
JOIN pg_catalog.pg_class child_class ON i.inhrelid = child_class.oid
LEFT JOIN LATERAL (
    SELECT pg_catalog.pg_get_expr(child_class.relpartbound, child_class.oid) AS part_expr
) AS pe ON TRUE
WHERE
    parent_class.relname = 'mappings'
    AND parent_nsp.nspname = 'public'
    AND pt.partdefid != child_class.oid
```

# Итого

# Проблемы

- Процедуры сложно переиспользовать в других сервисах.
- pg_cron устанавливается только в одну базу данных на одном сервере PostgreSQL.
- Отсутствует мониторинг. Если партиция не создалась, то об этом узнаем не сразу.

# Проблемы

- ~~Процедуры сложно переиспользовать в других сервисах.~~
- pg_cron устанавливается только в одну базу данных на одном сервере PostgreSQL.
- Отсутствует мониторинг. Если партиция не создалась, то об этом узнаем не сразу.

# Проблемы

- ~~Процедуры сложно переиспользовать в других сервисах.~~
- ~~pg_cron устанавливается только в одну базу данных на одном сервере PostgreSQL.~~
- Отсутствует мониторинг. Если партиция не создалась, то об этом узнаем не сразу.

# Проблемы

- ~~Процедуры сложно переиспользовать в других сервисах.~~
- ~~pg_cron устанавливается только в одну базу данных на одном сервере PostgreSQL.~~
- ~~Отсутствует мониторинг. Если партиция не создалась, то об этом узнаем не сразу.~~

# Как подобрать размер партиции?

# Объем данных

## Меньше – лучше

Идеально, когда партиция полностью помещается в буфер сервера

# Объем данных

## Меньше – лучше

Идеально, когда партиция полностью помещается в буфер сервера

VS

# Количество партиций

## Больше – хуже

Если есть запросы, которые ходят сразу во множество партиций, то партиций должно быть меньше

# Партицированные таблицы и EF Core

# Модель

Mappings.cs ×

```
public class Mapping
{
    public Guid Id { get; init; }
    public required string AgentNumber { get; set; }
    public required string ClientNumber { get; set; }
    public required string VirtualNumber { get; init; }
    public DateTimeOffset StartDate { get; init; }
    public DateTimeOffset? EndDate { get; set; }
}
```

# Конфигурация модели

DbContext.cs ×

```
protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    var mapping = modelBuilder.Entity<Mapping>();

    mapping.ToTable("mappings");
    mapping.HasKey(c => c.Id);
    mapping.Property(x => x.Id).HasColumnName("id");
    mapping.Property(x => x.AgentNumber).HasColumnName("agent_number").HasMaxLength(20);
    mapping.Property(x => x.ClientNumber).HasColumnName("client_number").HasMaxLength(20);
    mapping.Property(x => x.VirtualNumber).HasColumnName("virtual_number").HasMaxLength(20);
    mapping.Property(x => x.StartDate).HasColumnName("start_date");
    mapping.Property(x => x.EndDate).HasColumnName("end_date");

    mapping.HasIndex(x => x.Id);
    mapping.HasIndex(x => new { x.AgentNumber, x.VirtualNumber });
}
```

# Конфигурация модели

DbContext.cs ×

```
protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    var mapping = modelBuilder.Entity<Mapping>();

    mapping.ToTable("mappings");
    mapping.HasKey(c => c.Id);
    mapping.Property(x => x.Id).HasColumnName("id");
    mapping.Property(x => x.AgentNumber).HasColumnName("agent_number").HasMaxLength(20);
    mapping.Property(x => x.ClientNumber).HasColumnName("client_number").HasMaxLength(20);
    mapping.Property(x => x.VirtualNumber).HasColumnName("virtual_number").HasMaxLength(20);
    mapping.Property(x => x.StartDate).HasColumnName("start_date");
    mapping.Property(x => x.EndDate).HasColumnName("end_date");

    mapping.HasIndex(x => x.Id);
    mapping.HasIndex(x => new { x.AgentNumber, x.VirtualNumber });
}
```

# Конфигурация модели

DbContext.cs ×

```
protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    var mapping = modelBuilder.Entity<Mapping>();

    mapping.ToTable("mappings");
    mapping.HasNoKey();
    mapping.Property(x => x.Id).HasColumnName("id");
    mapping.Property(x => x.AgentNumber).HasColumnName("agent_number").HasMaxLength(20);
    mapping.Property(x => x.ClientNumber).HasColumnName("client_number").HasMaxLength(20);
    mapping.Property(x => x.VirtualNumber).HasColumnName("virtual_number").HasMaxLength(20);
    mapping.Property(x => x.StartDate).HasColumnName("start_date");
    mapping.Property(x => x.EndDate).HasColumnName("end_date");

    mapping.HasIndex(x => x.Id);
    mapping.HasIndex(x => new { x.AgentNumber, x.VirtualNumber });
}
```

# Операции над моделью

В EF модель без первичного ключа не может отслеживать изменения.

**Select**

**Insert**

**Update**

**Delete**

# Операции над моделью

В EF модель без первичного ключа не может отслеживать изменения.

**Select**

**Insert**



**Update**

**Delete**

# Операции над моделью

В EF модель без первичного ключа не может отслеживать изменения.

**Select**

**Insert**



**Update**



**Delete**

# Операции над моделью

В EF модель без первичного ключа не может отслеживать изменения.

**Select**

**Insert**



**Update**



**Delete**

# Конфигурация модели

DbContext.cs ×

```
protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    var mapping = modelBuilder.Entity<Mapping>();

    mapping.ToTable("mappings");
    mapping.HasKey(c => c.Id);
    mapping.Property(x => x.Id).HasColumnName("id");
    mapping.Property(x => x.AgentNumber).HasColumnName("agent_number").HasMaxLength(20);
    mapping.Property(x => x.ClientNumber).HasColumnName("client_number").HasMaxLength(20);
    mapping.Property(x => x.VirtualNumber).HasColumnName("virtual_number").HasMaxLength(20);
    mapping.Property(x => x.StartDate).HasColumnName("start_date");
    mapping.Property(x => x.EndDate).HasColumnName("end_date");

    mapping.HasIndex(x => x.Id);
    mapping.HasIndex(x => new { x.AgentNumber, x.VirtualNumber });
}
```

# Миграция

InitialMigration.cs ×

```
protected override void Up(MigrationBuilder migrationBuilder)
{
    migrationBuilder.CreateTable(
      name: "mappings",
      columns: table => new
      {
        id = table.Column<Guid>(type: "uuid", nullable: false),
        agent_number = table.Column<string>(type: "character varying(20)", maxLength: 20, nullable: false),
        client_number = table.Column<string>(type: "character varying(20)", maxLength: 20, nullable: false),
        virtual_number = table.Column<string>(type: "character varying(20)", maxLength: 20, nullable: false),
        start_date = table.Column<DateTimeOffset>(type: "timestamp with time zone", nullable: false),
        end_date = table.Column<DateTimeOffset>(type: "timestamp with time zone", nullable: true)
      },
      constraints: table =>
      {
      });

    migrationBuilder.CreateIndex(
      name: "IX_mappings_agent_number_virtual_number",
      table: "mappings",
      columns: new[] { "agent_number", "virtual_number" });

    migrationBuilder.CreateIndex(
      name: "IX_mappings_id",
      table: "mappings",
      column: "id");
}
```

# Миграция

InitialMigration.cs ×

```
protected override void Up(MigrationBuilder migrationBuilder)
{
    migrationBuilder.CreateTable(
        name: "mappings",
        columns: table => new
        {
            id = table.Column<Guid>(type: "uuid", nullable: false),
            agent_number = table.Column<string>(type: "character varying(20)", maxLength: 20, nullable: false),
            client_number = table.Column<string>(type: "character varying(20)", maxLength: 20, nullable: false),
            virtual_number = table.Column<string>(type: "character varying(20)", maxLength: 20, nullable: false),
            start_date = table.Column<DateTimeOffset>(type: "timestamp with time zone", nullable: false),
            end_date = table.Column<DateTimeOffset>(type: "timestamp with time zone", nullable: true)
        },
        constraints: table =>
        {
        });

    migrationBuilder.CreateIndex(
        name: "IX_mappings_agent_number_virtual_number",
        table: "mappings",
        columns: new[] { "agent_number", "virtual_number" });

    migrationBuilder.CreateIndex(
        name: "IX_mappings_id",
        table: "mappings",
        column: "id");
}
```

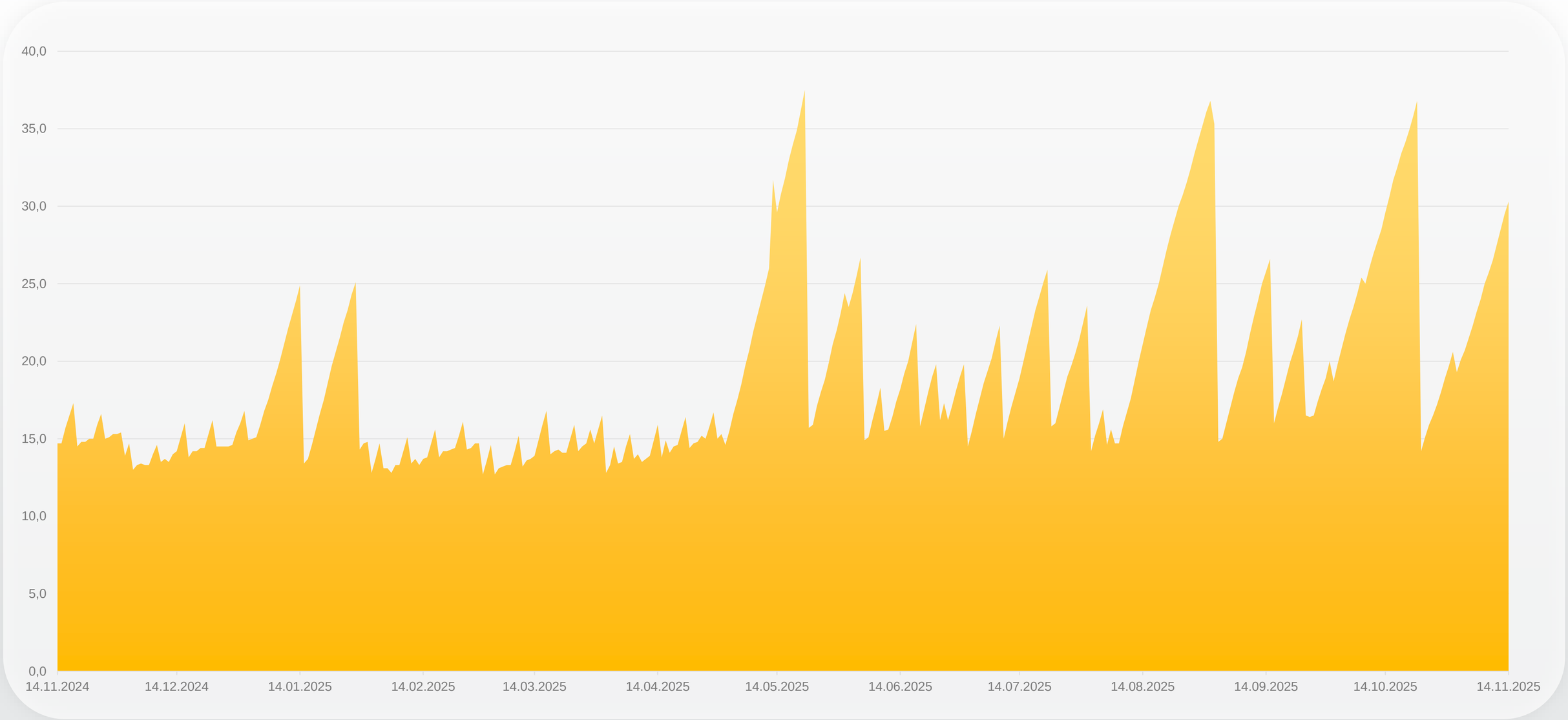
# Миграция

InitialMigration.cs ×

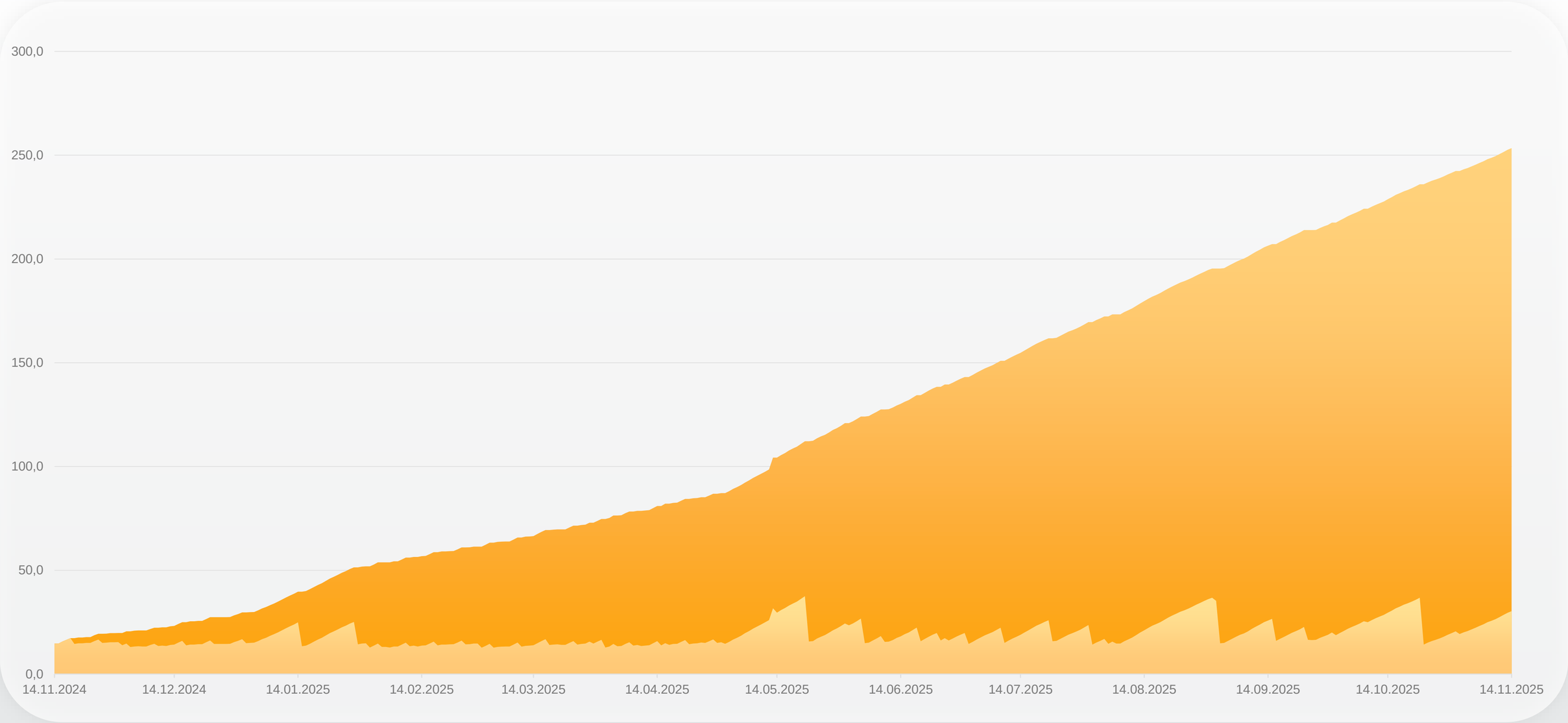
```
protected override void Up(MigrationBuilder migrationBuilder)
{
    migrationBuilder.Sql("""
                CREATE TABLE mappings (
                  id UUID NOT NULL,
                  agent_number VARCHAR(20) NOT NULL,
                  client_number VARCHAR(20) NOT NULL,
                  virtual_number VARCHAR(20) NOT NULL,
                  start_date TIMESTAMPTZ NOT NULL,
                  end_date TIMESTAMPTZ
                ) PARTITION BY RANGE (end_date)
                """);

    migrationBuilder.CreateIndex(
      name: "IX_mappings_agent_number_virtual_number",
      table: "mappings",
      columns: new[] { "agent_number", "virtual_number" });

    migrationBuilder.CreateIndex(
      name: "IX_mappings_id",
      table: "mappings",
      column: "id");

    migrationBuilder.Sql("CREATE TABLE mapping_defaults PARTITION OF mappings DEFAULT");
}
```

# Результаты на графиках

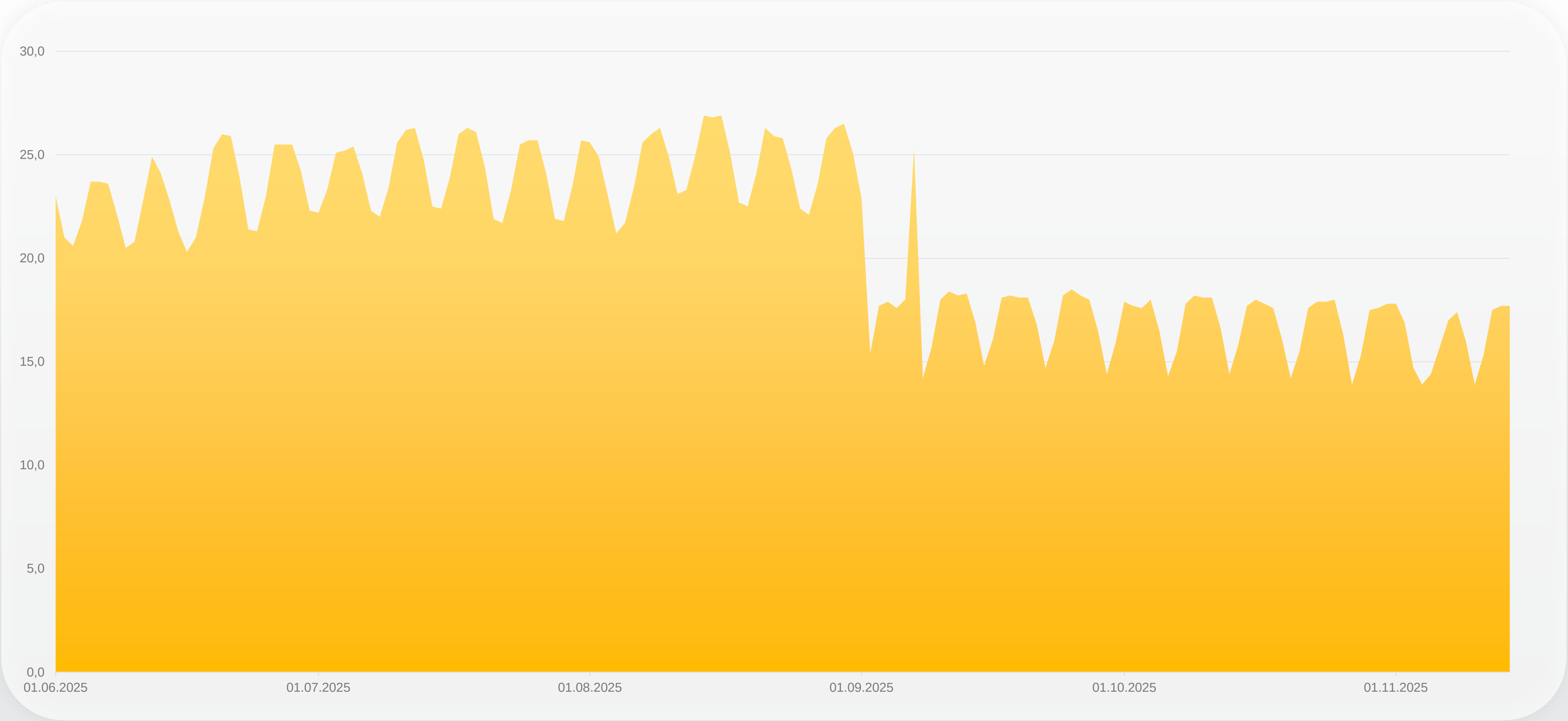
# Размер БД связок



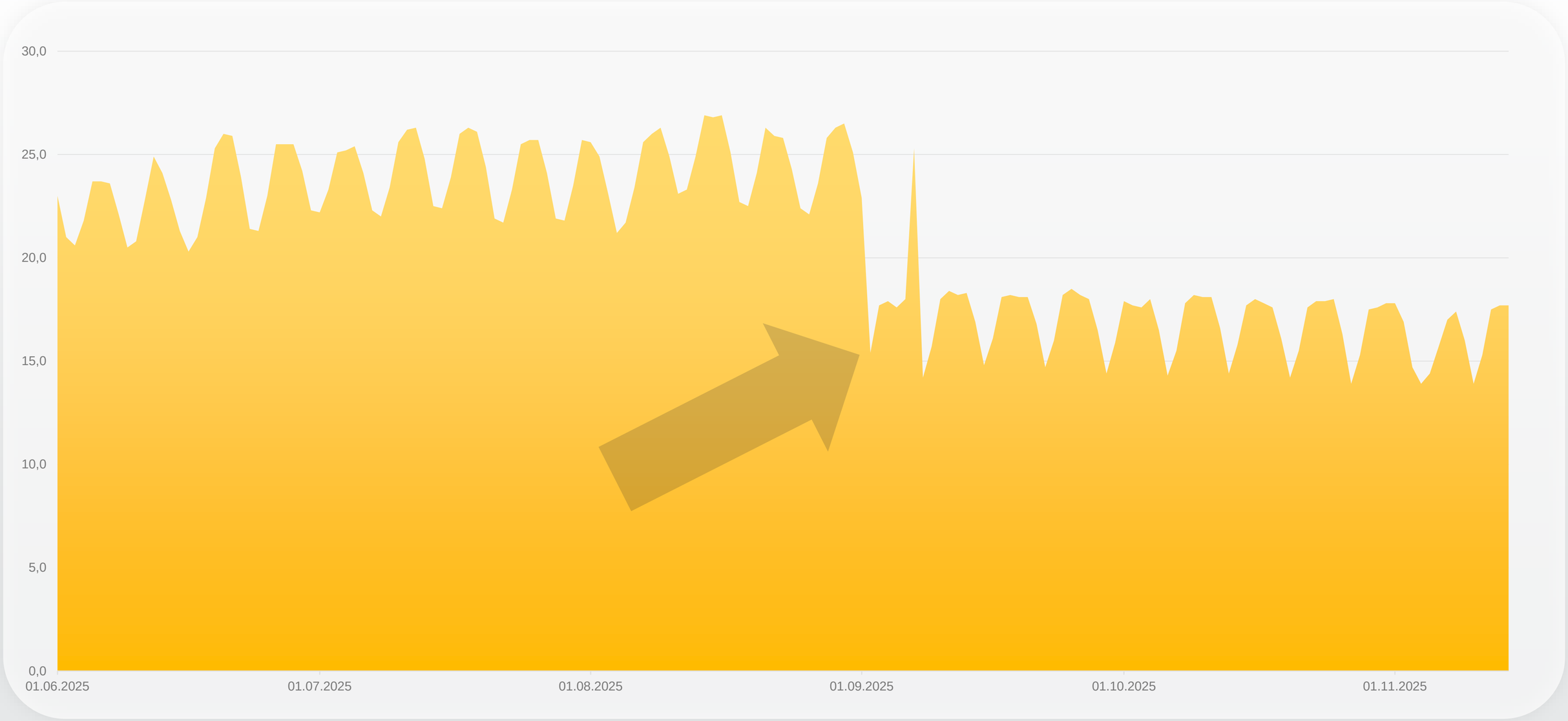
# Размер БД связок



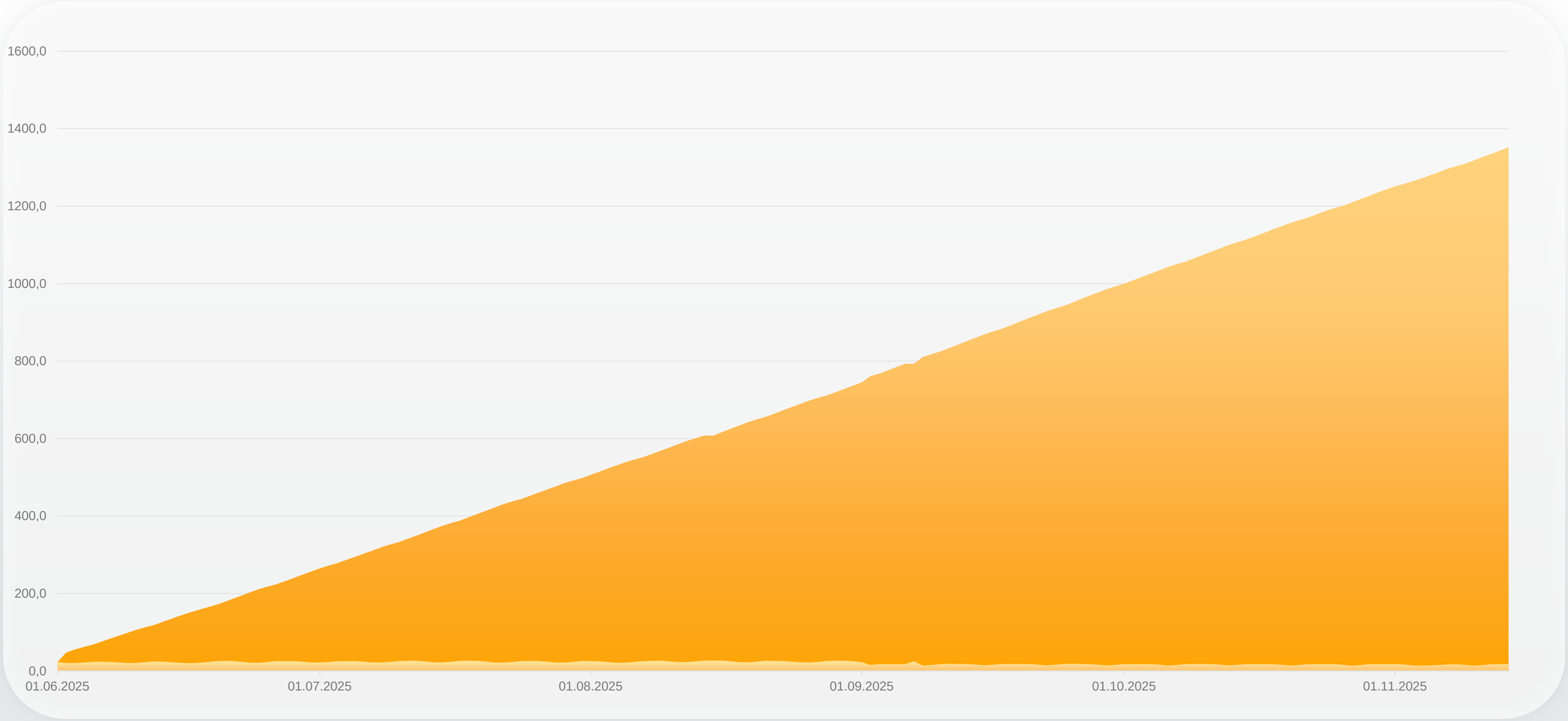
# Размер БД состояния звонков



# Размер БД состояния звонков



# Размер БД состояния звонков



.NET

EF Core

DbContext

Продам

SQL

PostgreSQL

Development

Гараж

C#

Planning

LINQ

Npgsql

# Спасибо за внимание!

Ваши вопросы