

# Validação: Relação TPS → QPS no Matcher

**Objetivo:** Validar se o entendimento das operações por transação está correto para dimensionamento de infraestrutura.

**Para:** Time de desenvolvimento do Matcher **De:** Time de Pricing **Data:** Janeiro 2026

## 1. Contexto

Estamos construindo o modelo de pricing do Matcher baseado no custo de infraestrutura. Para isso, precisamos entender quantas operações de banco de dados (QPS) são geradas para cada transação processada (TPS).

**Pergunta central:** Se um cliente processa X transações por segundo, quantas queries/ operações isso gera nos componentes de infra?

## 2. Meu Entendimento (baseado na leitura do código)

### 2.1 Operações de INGESTION (por transação)

Analisando `internal/ingestion/services/command/use_case.go` :

#	Operação	Componente	Código/Arquivo
1	Verificar duplicata	Redis	<code>dedupe_service.go</code> → <code>SETNX</code>
2	Verificar se existe	PostgreSQL	<code>transaction.postgresql.go</code> → <code>SELECT EXISTS</code>
3	Inserir transação	PostgreSQL	<code>transaction.postgresql.go</code> → <code>INSERT</code>

**Subtotal Ingestion:** 1 Redis + 2 PostgreSQL = **3 operações por transação**

## 2.2 Operações de MATCHING (por match realizado)

Analisando o fluxo de matching:

#	Operação	Componente	Razão
1	Buscar transações unmatched	PostgreSQL	<code>SELECT</code> (1x por batch, não por txn)
2	Adquirir lock txn A	Redis	<code>SETNX</code> para evitar race condition
3	Adquirir lock txn B	Redis	<code>SETNX</code> para evitar race condition
4	Inserir match_group	PostgreSQL	<code>INSERT</code>
5	Inserir match_item A	PostgreSQL	<code>INSERT</code>
6	Inserir match_item B	PostgreSQL	<code>INSERT</code>
7	Atualizar status txn A	PostgreSQL	<code>UPDATE</code>
8	Atualizar status txn B	PostgreSQL	<code>UPDATE</code>
9	Liberar lock txn A	Redis	<code>DEL</code>
10	Liberar lock txn B	Redis	<code>DEL</code>

**Subtotal Matching (por match 1:1):** 4 Redis + 5 PostgreSQL = **9 operações por match**

## 2.3 RabbitMQ

Operação	Frequência
Publicar evento <code>ingestion.completed</code>	1 por JOB (batch)
Consumir evento	1 por JOB

**Conclusão:** RabbitMQ tem uso muito baixo (não escala com TPS)

## 3. Fórmula Proposta

Assumindo **match 1:1** e **100% das transações resultam em match** (pior caso para sizing):

Para cada 1 TPS do cliente:

Redis:

- Ingestion: 1 op/txn
- Matching: 4 ops/match (assumindo 1 match por 2 txns = 2 ops/txn)
- Total Redis: ~3 ops por transação

PostgreSQL:

- Ingestion: 2 ops/txn
- Matching: 5 ops/match (assumindo 1 match por 2 txns = 2.5 ops/txn)
- Total PostgreSQL: ~4.5 ops por transação

TOTAL: ~7.5 operações por transação

**Arredondando para cima (margem de segurança):**

```
1 TPS ≈ 5 Redis ops + 6 PostgreSQL ops |  
1 TPS ≈ 11 QPS total na infraestrutura |
```

---

## 4. Perguntas para Validação

---

### Sobre Ingestion:

1. **O fluxo de dedupe sempre passa pelo Redis?** Ou existe cache em memória antes?
2. **O `SELECT EXISTS` é executado para toda transação?** Ou só quando o Redis retorna que não é duplicata?
3. **Os `INSERTs` são feitos em batch ou um por um?** Se batch, qual o tamanho típico?

### Sobre Matching:

1. **O matching é síncrono ou assíncrono?** Se assíncrono, o volume de operações pode ser "espalhado" no tempo?
2. **O lock distribuído no Redis é sempre usado?** Ou só quando há múltiplos workers?
3. **Os `INSERTs` de `match_group` e `match_items` são feitos em uma única transação SQL?** Ou são queries separadas?
4. **O `UPDATE` de status das transações é feito individualmente ou em batch?**

### Sobre Match Rate:

1. **Para sizing, devo assumir 100% match rate?** Ou existe um cenário típico diferente?
2. **Matches N:M (uma transação matchando com várias) são comuns?** Isso aumentaria as operações.

### Sobre Outros Componentes:

1. **Existem outras operações que não identifiquei?** (audit log, métricas, etc.)
2. **O RabbitMQ realmente só é usado 1x por job?** Ou existem outros eventos?

## 5. Tabela para Preencher (se possível)

Se puder validar/corrigir:

Operação	Minha Estimativa	Valor Real	Notas
Redis ops por transação (ingestion)	1		
PostgreSQL ops por transação (ingestion)	2		
Redis ops por match	4		
PostgreSQL ops por match	5		
RabbitMQ msgs por job	1		
Batch size típico de ingestion	1000?		
Match rate típico	80%?		

## 6. Por que isso importa?

Com essa validação, conseguimos:

1. **Dimensionar infraestrutura corretamente** - Saber qual instância de RDS/ElastiCache usar para cada volume de cliente
2. **Precificar por TPS** - Se sabemos que 1 TPS = X QPS, conseguimos calcular o custo de infra

3. **Evitar sub/super-dimensionamento** - Não queremos infra cara demais nem gargalos

---

## 7. Exemplo de Uso

---

Se a fórmula estiver correta:

Cliente faz	TPS equivalente	QPS na infra	Instância PostgreSQL sugerida
1M txns/mês	~1.2 TPS pico	~13 QPS	db.t3.medium (suporta ~30 QPS)
5M txns/mês	~6 TPS pico	~66 QPS	db.t3.large (suporta ~100 QPS)
20M txns/mês	~24 TPS pico	~264 QPS	db.r6g.large + replica

---

**Aguardo feedback!**

Qualquer correção no entendimento ajuda a refinar o modelo de pricing.