Propriedade dos dados

Defina quem decide, quem grava e quem consulta

Pré-requisitos

Conheça os contextos de Reservas e Pagamentos, identificadores estáveis e operações SQL básicas. O laboratório usa Python 3 com SQLite 3.24 ou posterior. Ele simula entregas de eventos dentro do mesmo processo.

Escolha a fonte de autoridade

O serviço de Reservas decide se a vaga está ocupada e mantém o estado oficial do agendamento. Pagamentos registra a cobrança e suas tentativas. Um painel pode exibir os dois estados, mas uma remarcação deve ser solicitada a Reservas, onde a disponibilidade e as transições podem ser verificadas.

Proteja a fronteira de escrita [1]

Conceda à aplicação proprietária os privilégios de escrita sobre suas tabelas, inclusive quando contextos compartilham a infraestrutura do banco. Outros contextos usam interfaces acordadas. Isso permite alterar o esquema interno sem exigir que consumidores conheçam colunas e regras de persistência.

Escolha como fornecer leituras

Uma consulta direta à API proprietária pode ser adequada quando a leitura precisa refletir o estado atual. Uma projeção local alimentada por eventos reduz dependências durante a leitura e permite compor um painel, com atraso entre a escrita e a exibição. Defina como medir esse atraso e como recompor a projeção após falhas.

Envie o que o consumidor precisa

Para o painel, envie identificador da reserva, estado e versão. A versão é crescente por reserva e atribuída pelo proprietário. O exemplo publica o estado completo que a projeção precisa. Se o evento trouxer apenas uma operação incremental, saltar versões pode perder efeitos e será preciso ordenar, reconciliar ou reler a origem.

Propriedade dos dados

Exercício resolvido • Projete o estado da reserva

Enunciado e dados de partida

Reservas publica o estado da mesma R45 nas versões 1, 2 e 3. O painel recebe primeiro a confirmação v2, depois a pendência v1, uma repetição da v2 e por fim o cancelamento v3. Faça a cópia convergir para o estado mais recente. O produtor garante que a mesma versão da mesma reserva tem sempre o mesmo conteúdo.

1. Comece o arquivo painel.py

```
import sqlite3

db = sqlite3.connect(':memory:')
db.execute("""
CREATE TABLE painel (
    reserva_id TEXT PRIMARY KEY,
    estado TEXT NOT NULL,
    versao INTEGER NOT NULL
)
""")

def projetar(evento):
    db.execute("""
        INSERT INTO painel VALUES (?, ?, ?)
        ON CONFLICT(reserva_id) DO UPDATE SET
            estado = excluded.estado,
            versao = excluded.versao
        WHERE excluded.versao > painel.versao
    """, evento)

def ler():
    return db.execute(
        "SELECT estado, versao FROM painel WHERE reserva_id = 'R45'"
    ).fetchone()
```

Entenda a escrita condicional [2]

A chave reserva_id permite uma linha por reserva. Se a primeira entrega for v2, o painel cria a linha já confirmada. Em entregas seguintes, excluded representa a linha candidata. O predicado aplica apenas versões maiores que a atual, sem uma leitura separada antes da escrita.

Propriedade dos dados

Execute e revise a convergência

2. Acrescente as entregas ao arquivo

```
eventos = [
    (('R45', 'confirmada', 2), ('confirmada', 2)),
    (('R45', 'pendente', 1), ('confirmada', 2)),
    (('R45', 'confirmada', 2), ('confirmada', 2)),
    (('R45', 'cancelada', 3), ('cancelada', 3)),
]
for evento, esperado in eventos:
    projetar(evento)
    assert ler() == esperado, (evento, ler())
    print(*ler())
```

3. Execute python painel.py

A saída esperada é ‘confirmada 2’ três vezes e ‘cancelada 3’ uma vez. A repetição mantém o estado e a versão. A versão pertence a R45 e só é comparável com outras versões da mesma reserva.

Caso de erro • Aceite toda atualização

Remova o WHERE da cláusula DO UPDATE. Na segunda entrega, a v1 sobrescreve a v2 e a segunda asserção falha. Restaure a condição. O painel tem sua própria persistência, mas atualizar a cópia não altera a vaga na origem.

Critérios para conferir

Teste a ordem indicada, confira a saída por etapa e identifique a fronteira de escrita do painel. Em produção, uma projeção precisa de um modo de recompor dados, como reler um snapshot da origem e consumir atualizações posteriores. Se o usuário realmente for confirmar uma reserva, a decisão final deve consultar a operação proprietária.

Referências primárias

[1] AWS Prescriptive Guidance • Banco por serviço e acesso por interfaces

[2] PostgreSQL • INSERT, ON CONFLICT e atualização condicional

Propriedade dos dados

Replique informações para leitura sem dividir a escrita

Escolha a fonte de cada informação

Um painel pode consultar uma projeção que combina o horário da reserva e o estado financeiro. A projeção é derivada de eventos e pode atrasar. As operações que alteram a reserva continuam no serviço responsável por ela.

Exercício adicional

O painel de suporte quer corrigir o horário editando sua projeção. Onde a alteração deve ocorrer?

Resolução

Envie a solicitação ao serviço de reservas para ele validar a disponibilidade e atualizar sua fonte de dados. O painel atualiza a projeção após o evento. Se o evento atrasar, mostre o horário da última atualização.

Ao assistir ao vídeo

Identifique como o acesso ao dado muda quando ele passa a ser controlado por um único serviço.

Vídeo complementar

Database Per Service (em inglês)