PROCESSO DE PENSAMENTO

Primeiramente fiz a leitura e entendimento do que se tratava o JSON. Minha primeira etapa foi analisar o conteúdo do arquivo em questão (ERP.json) e entender seu propósito: representar a resposta de um endpoint de ERP para restaurar, simulando um pedido completo. O JSON segue um padrão bem estruturado com entidades hierárquicas. Identifiquei que o JSON contém:

- Informações gerais da operação;

- Uma lista de pedidos;

- Itens do pedido;

- Dados complementares como menuItem e taxes.

A partir disso, fiz um mapeamento conceitual dos dados.

Segundo lugar, identifiquei as entidades principais que com base na estrutura do JSON e no contexto de operações de restaurante, destaquei as entidades que representam ativos importantes para análise de dados e integração com sistemas relacionais:

- GuestCheck -> representa o pedido principal;

- DetailLine -> cada item ou ação dentro de um pedido;

- MenuItem -> item de cardpápio vendido;

- Tax -> impostos cobrados.

Além disso, considerei possíveis futuras entidades (discount, serviceCharge, etc.), sugeridas na própria descrição do desafio.

Depois precisei tomar algumas decisões de MODELAGEM, segui com uma abordagem relacional e normalizada. Algumas das principais decisões:

- Chaves primárias reais dos dados ERP (guestCheckId e guestCheckLineItemId) foram utilizadas para rastreabilidade;

- Separação em tabelas distintas de impostos e itens de menu evita redundância e melhora manutenção;

- Uso de campos temporais em UTC e Local para garantir flexibilidade em análises temporais;

- A inclusão de campos financeiros foi feita com decimal, pensando na precisão exigida por aplicações reais de restaurantes e auditoria fiscal.

Sobre as preocupações consideradas...

1. Primeiro ponto é a ESCALABILIDADE, o modelo foi construído de forma a permitir expansão futura. Como o ERP pode enviar diferentes tipos de itens (discount,

serviceCharge), deixei o modelo preparado para generalizações ou subtipagem de DetailLine.

2. Consistência temporal também foi algo que pensei, com dados de data e hora aparecem com versões UTC e local. Em contexto de redes de restaurantes com múltiplas localidades e fusos horários, manter ambos registros pode evitar problemas em análises e relatórios.
3. Performance, apesar de ter priorizado a normalização para evitar duplicação, tive atenção a performance de leitura, já que é comum que pedidos completos precisem ser carregados com todos os detalhes.

Agora como consequência das minhas escolhas

Decisão	Benefícios	Riscos/Penalidades
Normalização das entidades	Flexibilidade, manutenção, reutilização de dados	Mais joins em consultas
Chaves estrangeiras entre as tabelas	Integridade referencial garantida	Maior custo de inserção/transação
Armazenar UTC e Local	Precisão em relatórios e histórico	Aumento do volume de dados armazenados
Uso de tipos fortes (DECIMAL, TIMESTAMP)	Correção fiscal, precisão financeira	Exige mais cuidado na conversão de dados em pipelines de ingestão
Separação de MenuItem de DetailLine	Clareza de propósito e organização para possíveis outros tipos	Consultas um pouco mais complexas