**DATABASE CACHING**

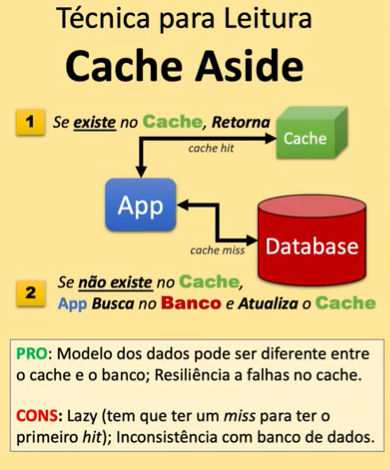
Para arquitetar o Cache, é necessário compreender a forma que se trabalha com cache de dados para evitar problemas como lentidão por esgotamento e necessidade de atualização da informação muito rápida.

Mas o que determina a sua decisão é:

* Demanda mais por leitura e quer reutilizar as leituras;
* Velocidade na escrita;
* Gerenciamento da atualização dos dados no cache;
* Ou deixar o sistema de cache gerenciar automaticamente.

Existem 5 técnicas mas a 5ª é uma variação:

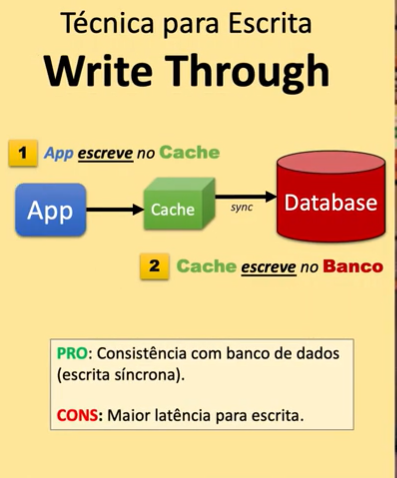
**Cache Aside** – recomendado para processamento de leitura intensiva em dados que se repetem muito, como por exemplo, para perfil do usuário, sessão usuário, informações de produto, departamento, categoria, coisas que executam todo momento e toda hora. Essa arquitetura permite que a estrutura e atributos das entidades do cache sejam diferentes do banco de dados, a aplicação que controla o cache do banco de dados conforme sua necessidade.



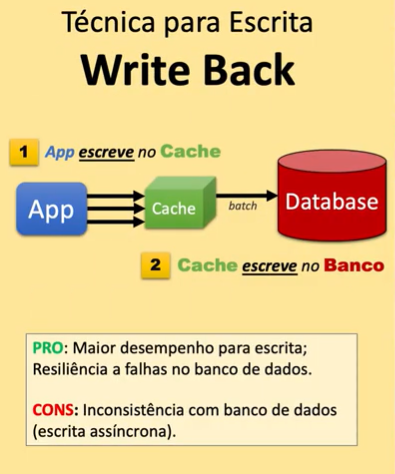
**Read Through** - O Cache fica entre a aplicação e o banco de dados e se o dado existe o cache já resolve para a aplicação e caso não exista, o próprio cache vai no banco de dados buscar a informação, devolve para aplicação e armazena no cache para rapidez no próximo atendimento. Neste caso a estrutura das entidades e seus atributos tem que ser idênticos ao banco de dados. Recomendado para processamento de leitura intensiva em dados que se repetem muito, como por exemplo, para perfil do usuário, sessão usuário, informações de produto, departamento, categoria, coisas que executam todo momento e toda hora.



**Write Through** – Arquitetura de escrita do dado onde a aplicação escreve primeiro no Cache e imediatamente vai pro Banco de Dados sendo este Cache o responsável por levar os dados para o banco. Neste caso a estrutura das entidades e seus atributos tem que ser idênticos ao banco de dados. A vantagem dessa atividade é a consistência do Banco de Dados com o Cache para a leitura, porém a desvantagem está na escrita que tem que aguardar a latência do banco de dados a partir da necessidade de consistência da operação entre cache e banco. Se você tem uma aplicação que tem Throughput muito alto para escrita, talvez essa não seja uma boa opção.



**Write Back/Write Behind** – A aplicação escreve no cache e este propaga os dados de forma assíncrona em batch para o banco de dados. Neste caso a estrutura das entidades e seus atributos tem que ser idênticos ao banco de dados. Existe uma inconsistência eventual nesta operação, mas se você tem uma atividade que demanda escrita intensa este possa ser o melhor caminho.



**Write Around** – É a combinação de técnicas anteriores onde a aplicação sempre escreve direto no banco de dados sem passar pelo cache. A leitura pode ser Read Through onde o próprio cache busca no banco quando não existir em cache ou Cache Aside onde a aplicação busca no banco caso não exista no cache. Esta técnica é bem utilizada em cenários Read Mostly onde a escrita não é tão crítica, mas a leitura é bem crítica e precisa estar em cache para evitar o trafego de rede.

