1. **O que é?** (Xavier e Tarzan)
   * Definir os conceitos fundamentais relacionados ao tema, dando exemplos concretos logo após cada definição sempre que possível.
2. **Para que serve?** (Isac e Lucca)
   * Explicar em que situações se pode usar a solução analisada.
     + 1. Grande volume de dados
       2. Muitas operações de escrita
3. **Quais as implicações?** (Felipe e Capa)
   * Mostrar vantagens e desvantagens da(s) solução(ões) analisada(s), impactos na tecnologia de bancos de dados.
   * Quando possível apresentar medidas numéricas em gráficos e ou tabelas.

O resultado do trabalho pode ser uma apresentação gravada e/ou ao vivo ou um documento texto. Pode-se combinar apresentação em parte gravada e em parte ao vivo.

**Todos os membros do grupo precisam colaborar com o desenvolvimento do trabalho e exercitar a habilidade de trabalhar em conjunto, dividindo as tarefas e integrando os resultados. O professor pode endereçar perguntas a qualquer membro do grupo para avaliar suas contribuições e rendimento.**

**Cada grupo terá 20 minutos para apresentar seu trabalho, dentro de aula síncrona, ficando outros 10 minutos para perguntas e discussões com o professor e o restante da turma.**

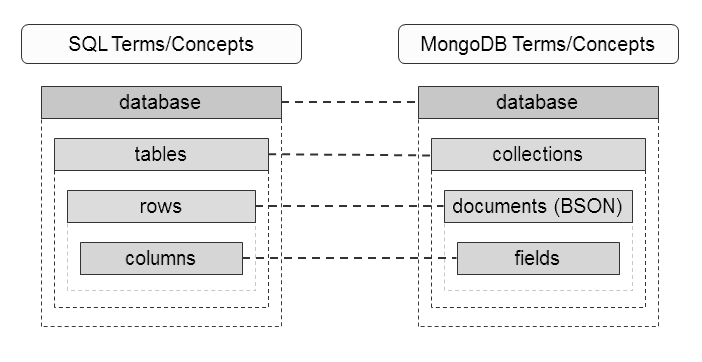
NoSQL *(Not Only SQL)* é o termo utilizado para banco de dados n**ão relacionais** de alto desempenho, onde geralmente não é utilizado o SQL como linguagem de consulta. O NoSQL foi criado para ter uma performance melhor e uma escalabilidade mais horizontal para suprir necessidades onde os bancos relacionais não são eficazes. No geral, temos 4 tipos de bancos de dados NoSQL:

* **Documento** – Os dados são armazenados como documentos. Os documentos podem ser descritos como dados no formato de chave-valor, como por exemplo, o padrão JSON. Um exemplo de banco de dados neste formato é o MongoDB;
* **Colunas** – Os dados são armazenados em linhas particulares de tabela no disco, podendo suportar várias linhas e colunas. Também permitem sub-colunas. Um banco de dados dessa família, por exemplo, é o Cassandra;
* **Grafos** – Os dados são armazenados na forma de grafos (vértices e arestas). O Neo4j é um banco que utiliza grafos;
* **Chave-valor** – Esta família de bancos NoSQL é a que aguenta mais carga de dados, pois o conceito dele é que um determinado valor seja acessado através de uma chave identificadora única. Um exemplo é o banco de dados Riak.

NOSQL -> nao-relacional

SQL -> relacional

**Resumindo:** o conceito de modelo relacional (SQL) se baseia no fato de que todos os dados sejam guardados em tabelas. Ao modelo não-relacional (NoSQL) não se aplica o conceito de *schema*: uma chave de valor é que é utilizada para recuperar valores, conjunto de colunas ou documentos.



Por exemplo, no caso de um fórum, cada post tem inúmeros replies. Será que fará sentido ter várias tabelas, uma a armazenar os posts em si e outra a armazenar todos os comentários, sendo que o servidor irá ter de percorrer as chaves estrangeiras de todos os replies existentes de todos os posts para os associar à chave principal do post em questão? Neste caso, a informação encontra-se “espalhada” entre as diversas tabelas, quando esta informação (post e replies) irá sempre ser apresentada em conjunto pela página web

muito usado em **Big Data**

**Just as transaction rates have grown out of recognition over the last decade, the volumes of data that are being stored also have increased massively. O’Reilly has cleverly called this the “industrial revolution of data.”**

**NoSQL databases have evolved to meet the scaling demands of modern Web 2.0 applications.**

Implicacoes

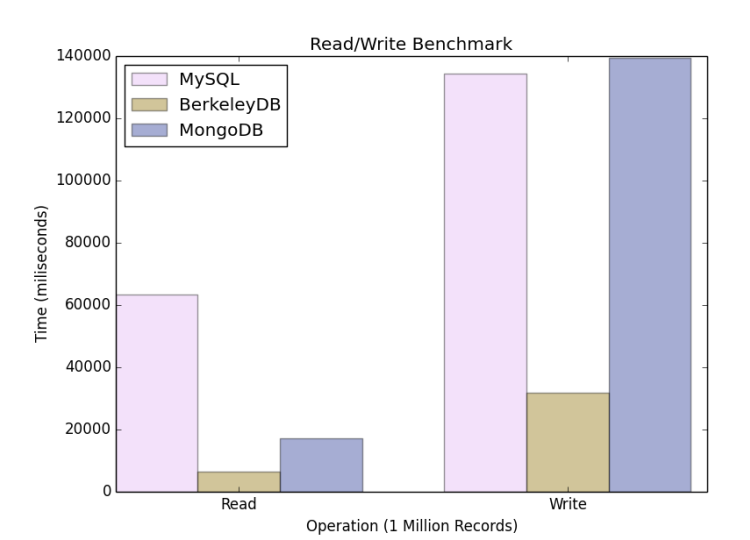
* Vantagens
  + mais performatico
  + definicao dos objetos é flexível (pode adicionar ou remover campos sem precisar alterar todos os ja existentes)
  + manutençao mais simples -> não é necessario pessoa DBA
* Desvantagens
  + grande parte dos sistemas NoSQL é open-source, então nao tem muito suporte técnico
  + Mais recente que o sistema relacional, mais difícil de achar respostas para implementaçoes -> falta **maturidade**
  + O modo como é implementado, visando facilitar o manuseio de grandes volumes de dados, acaba dificultando as analises para analytis e business inteligence

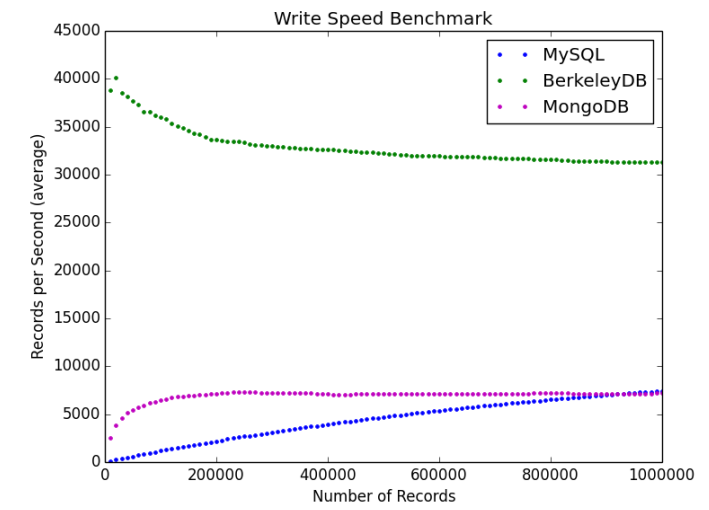
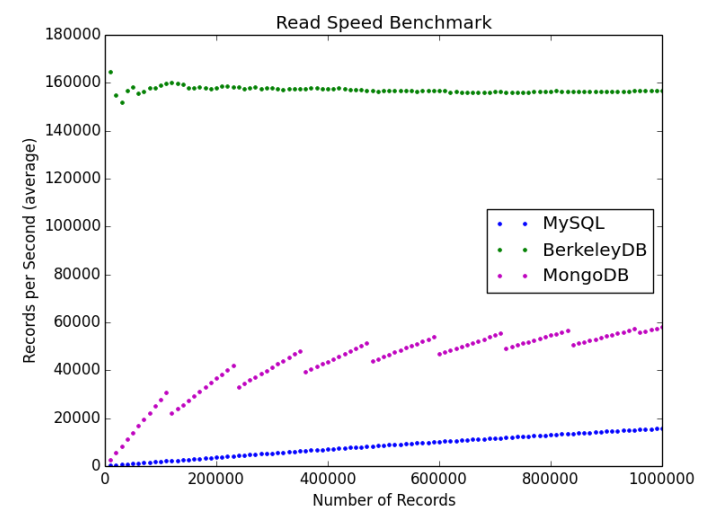
**Is NoSQL faster than SQL?**

In general, NoSQL is not faster than SQL just as SQL is not faster than NoSQL. For those that didn’t get that statement, it means that speed as a factor for SQL and NoSQL databases depends on the context.

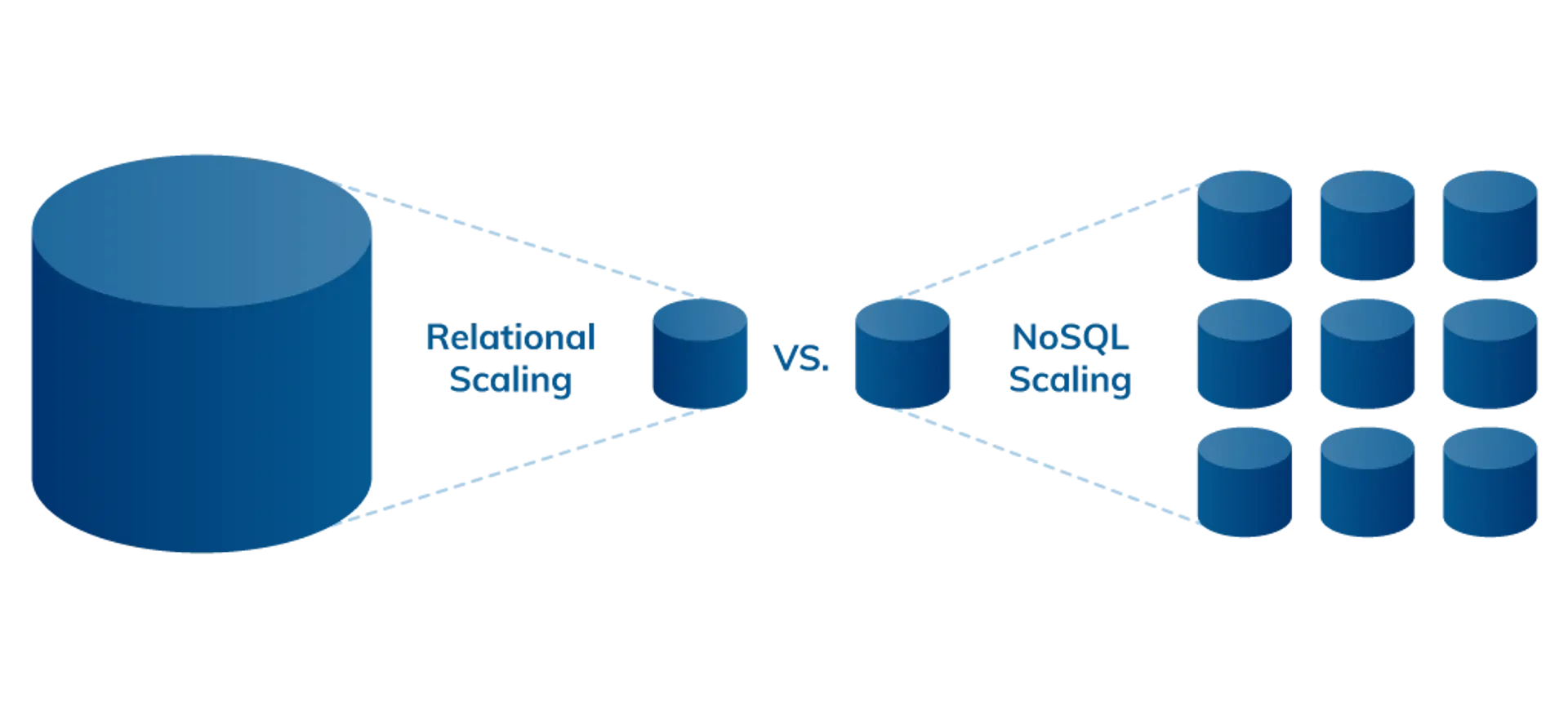
SQL databases are normalized databases where the data is broken down into various logical tables to avoid data redundancy and data duplication. In this scenario, SQL databases are faster than their NoSQL counterparts for joins, queries, updates, etc.

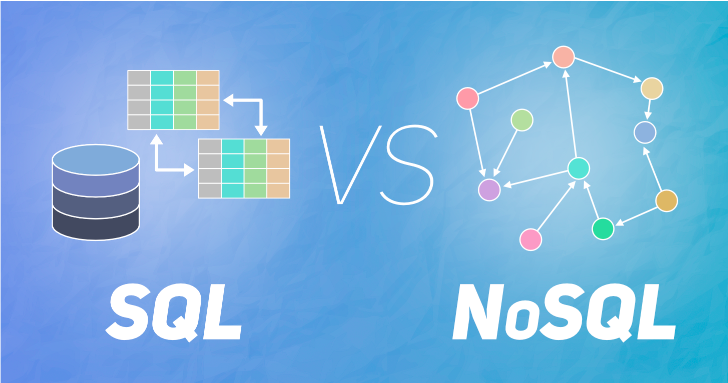
On the other hand, NoSQL databases are specifically designed for unstructured data which can be document-oriented, column-oriented, graph-based, etc. In this case, a particular data entity is stored together and not partitioned. So performing read or write operations on a single data entity is faster for NoSQL databases as compared to SQL databases.

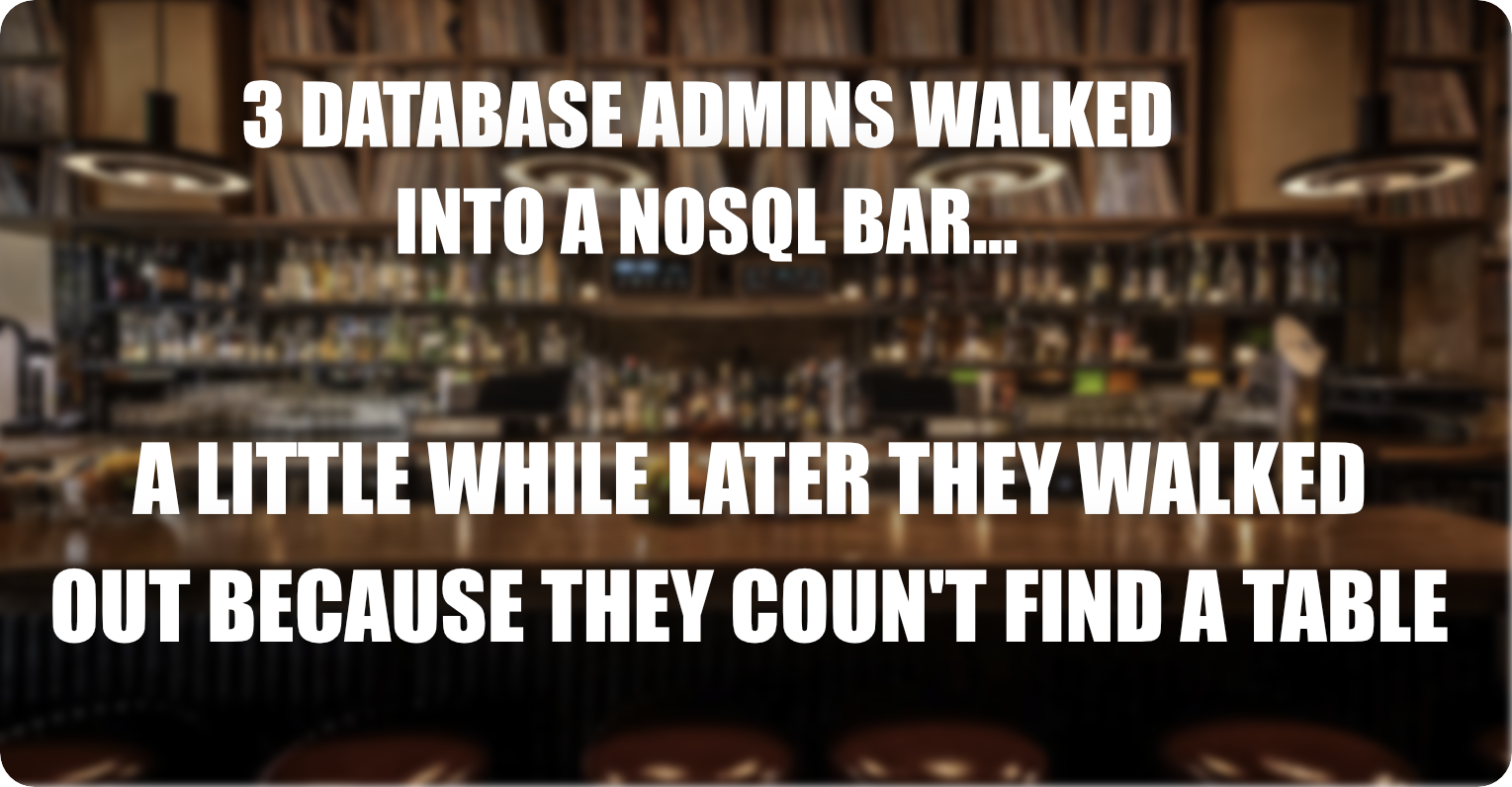




https://www.cs.rochester.edu/courses/261/fall2017/termpaper/submissions/06/Paper.pdf







Referencias

<https://www.treinaweb.com.br/blog/sql-vs-nosql-qual-usar>

<https://sirobotica.wordpress.com/2014/10/23/nosql-o-que-e-vantagens-e-limitacoes/>

<https://www.techrepublic.com/article/10-things-you-should-know-about-nosql-databases/>

<https://www.datastax.com/nosql>

<https://en.wikipedia.org/wiki/NoSQL>

<https://www.mongodb.com/nosql-explained>

https://www.ignoredbydinosaurs.com/posts/210-explaining-non-relational-databases-my-mom

<https://www.scalyr.com/blog/sql-vs-nosql-performance/>

<https://www.devmedia.com.br/banco-de-dados-nosql-um-novo-paradigma-revista-sql-magazine-102/25918>

<https://www.devmedia.com.br/introducao-ao-mongodb/30792>

https://www.devmedia.com.br/postgresql-tutorial/33025

<https://inside.contabilizei.com.br/quando-utilizar-nosql-sql-9c9255cd5f12>

Incompatibilidade de impedância:

<https://www.linkedin.com/pulse/voc%C3%AA-sabe-o-que-%C3%A9-incompatibilidade-de-imped%C3%A2ncia-banco-lima/?originalSubdomain=pt>

**Quando usar NoSQL?**

Apesar de ter um bom desempenho e funcionar bem frente a sistemas com alta escalabilidade, o NoSQL não é indicado para todos os casos. Em sistemas onde o modo de estruturação dos dados é relevante, o modelo relacional é recomendado. Por exemplo, imagine uma situação em que seja necessário fazer o levantamento do número de clientes para cada filial de uma determinada empresa. Se houver uma tabela específica para os dados de clientes, essa contagem poderá ser feita de maneira muito mais fácil e rápida. O NoSQL teria mais dificuldade para tal, tendo em vista que os dados destes clientes estão espalhados na memória.

Porém, há também diversos exemplos onde bancos não relacionais se saem muito bem. Uma das características que dá vantagem ao NoSQL é o fato de não necessitar da criação de um esquema prévio. Isso torna o paradigma atraente para projetos urgentes que não possuam muito tempo para serem desenvolvidos e que não tenham a duplicidade de informação, causada pelo NoSQL, como um problema crítico.

Outro aspecto que também beneficia este paradigma frente ao modelo relacional é a incompatibilidade de impedância que pode ser causada por este último quando usado para armazenar dados de um sistema orientado a objetos. O termo “incompatibilidade de impedância” vem da engenharia elétrica e se refere à diferença de resistência entre componentes de um circuito. O fato de linguagens orientadas a objetos utilizarem frequentemente referências a atributos de diferentes tipos dentro de um objeto, algo que não é possível em um banco de dados relacional, pode gerar um alto grau de processamento e latência nas buscas por estes valores quando necessário. Um exemplo pode ser visto na Figura X. O NoSQL, por sua vez, permite que os atributos relacionados a este objeto sejam armazenados em conjunto com o restante dos dados referentes a este em memória, acelerando posteriores procedimentos de busca e resultando num considerável ganho de desempenho.

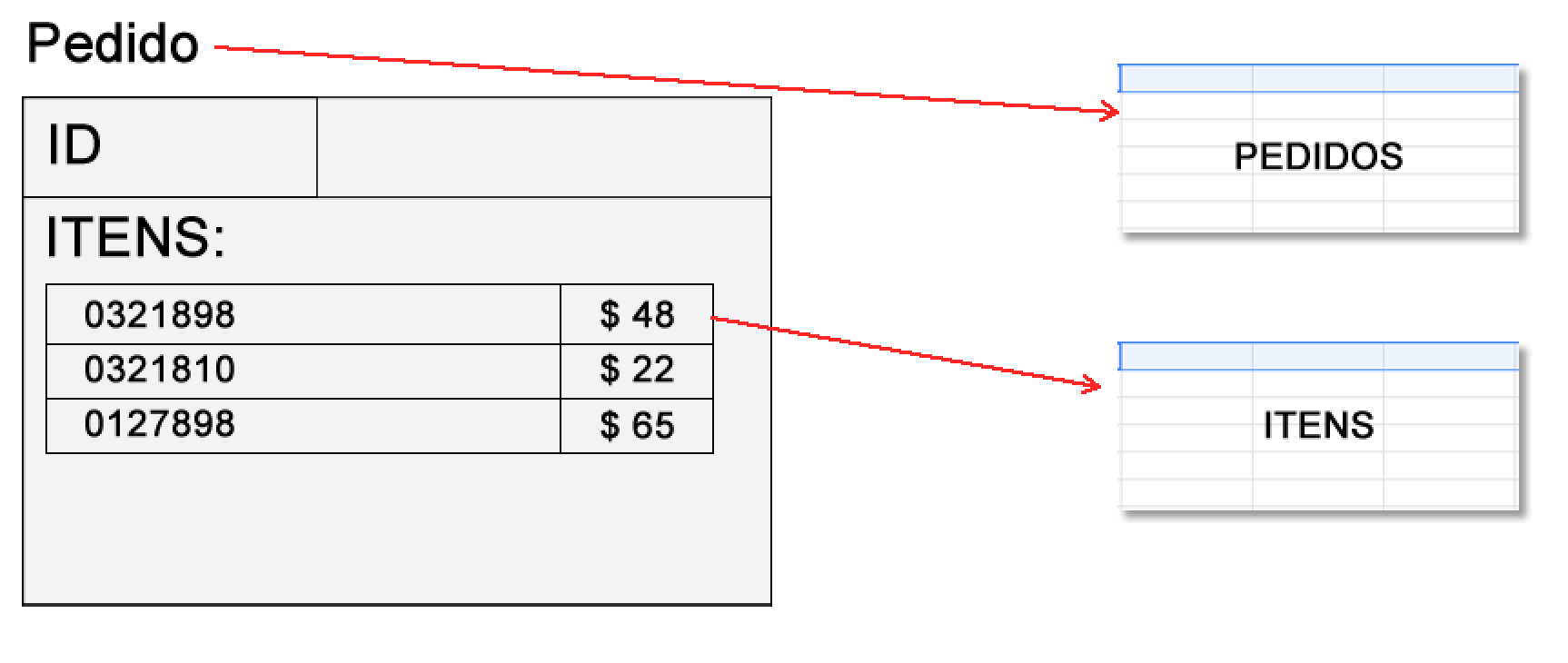


Figura X: Exemplo de incompatibilidade de impedância causada por um banco de dados relacional em conjunto com uma linguagem de programação orientada a objetos.