Instituto Federal de Educação, Ciência e Tecnologia do Piauí-IFPI
Curso de Tecnologia em Análise e Desenvolvimento de Sistemas
Campus Central- Prof.º Ricardo Ramos
Disciplina de Introdução à Computação
Alunos: Vinícius Gomes e Tatiana Sousa

TIPOS DE BANCO DE DADOS

Teresina, Piauí
28 de maio de 2019

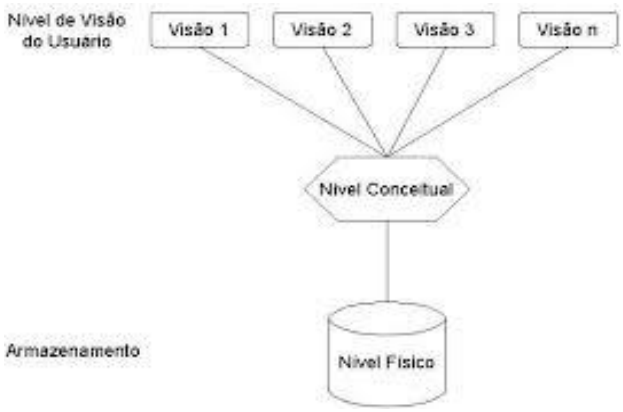
Conceito:

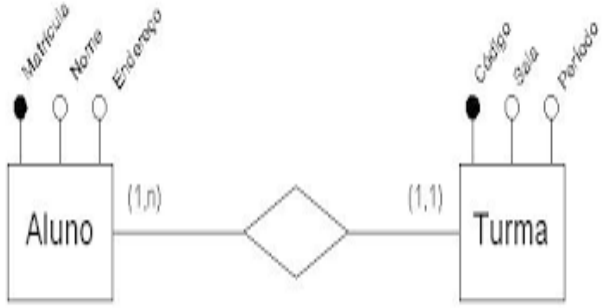
O termo **Banco de Dados** tem duas aplicações distintas. Alguns falam que banco de dados é o mesmo que **SGBD**, (Sistema Gerenciador de Banco de Dados), ou seja, um programa para gerenciar dados. O termo também é utilizado para definir uma **Base de dados** (grupo de dados agrupados por um SGBD).

Para criar a base de dados o SGBD utiliza uma **linguagem**. A mais utilizada atualmente é o **SQL**, (**Structured Query Language**).

Existem vários SGBDs no mercado. Alguns são pagos, outros gratuitos. Para criar um banco de dados não é preciso aprender a linguagem SQL, existem programas que criam uma interface gráfica, gerando um código em SQL automaticamente.

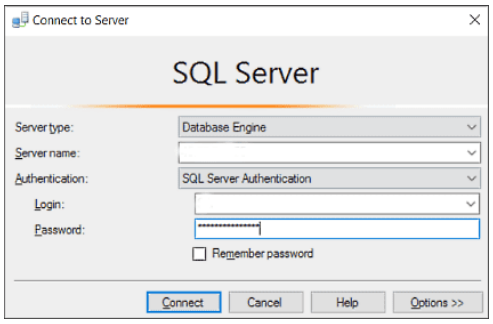
Imagens:

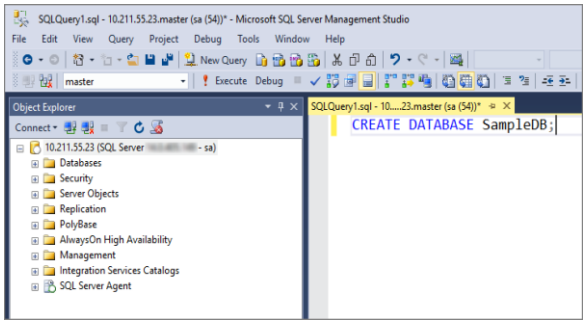




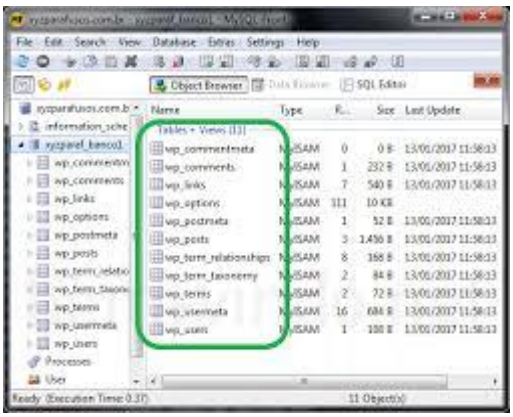
Alguns tipos de SGBD:

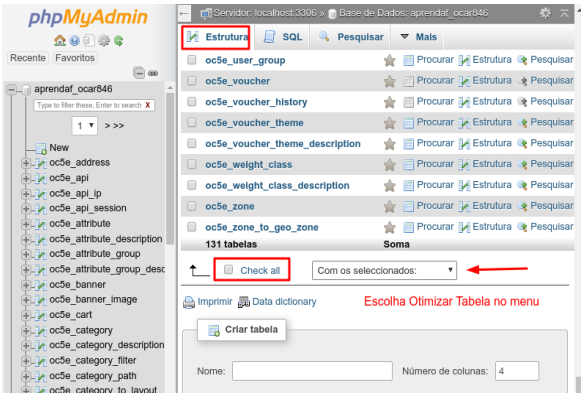
- **SQLServer**: Um dos maiores SGBD do mundo, sob licença da Microsoft, tem versões pagas e gratuitas.



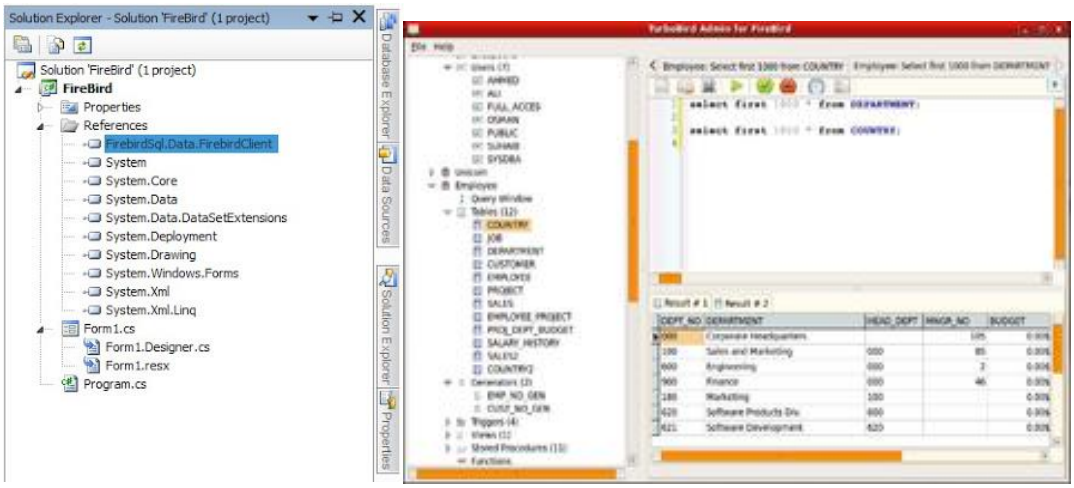


- **MySQL**: O MySQL é um software livre, com código fonte aberto e uso gratuito.



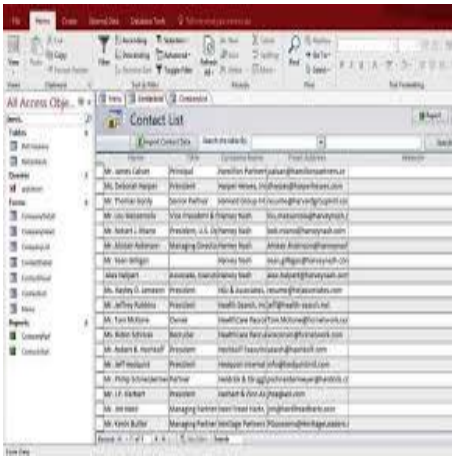


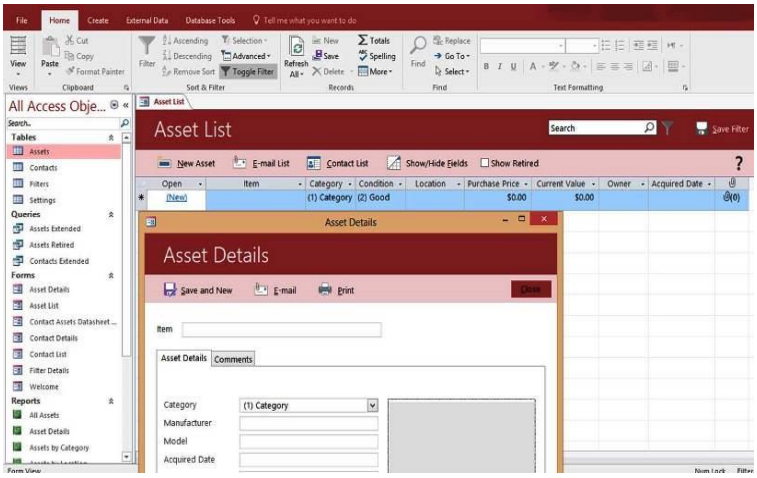
- **FirebirdSQL**: Roda na maioria dos sistemas Unix, e tem código fonte aberto.



- **mSQL**: Criado pela Hughes Technologies Pty Ltd., é um software que trabalha mais com o uso eficiente da memória, e é um sistema pequeno. Sua licença é altamente controlada pela empresa dona do produto. Trabalhado em **Linguagem C**.

- **Microsoft Access**: **é um SGBD da Microsoft que acompanha o pacote Office**. É muito utilizado para a aprendizagem e tem poucas atribuições profissionais, devido a sua limitação muito grande em armazenamento.





Para armazenar um dado, é necessário criar tabelas, dentro das tabelas são criadas colunas, onde as informações são armazenadas. Para os dados da base de dados ficarem organizados, devem ser criadas tabelas que não misturem informações.

Os **comando**s variam de um **SGBD** para o outro. Embora a linguagem seja a mesma, o comando para declarar um tipo de campo varia entre os Bancos de dados.

Para quem está começando na área de armazenamento de dados, o Access é o mais indicado, pois tem uma interface amigável o que facilita sua utilização.

Aplicações de Banco de Dados:

Sistemas Gerenciadores de Bancos de dados são usados em muitas aplicações, enquanto atravessando virtualmente a gama inteira de software de computador. Os Sistemas Gerenciadores de Bancos de dados são o método preferido de armazenamento/recuperação de dados/informações para aplicações multiusuárias grandes onde a coordenação entre muitos usuários é necessária. Até mesmo usuários individuais os acham conveniente, entretanto, muitos programas de correio eletrônico e organizadores pessoais estão baseados em tecnologia de banco de dados **standard.**
Bancos de dados ou bases de **dados** são conjuntos de arquivos relacionados entre si que se relacionam de forma a criar algum sentido (Informação) e dar mais eficiência durante uma pesquisa ou estudo.

Exemplos de **SGBDs**:

- **Oracle;**
- **SQL Server;**
- **DB2;**
- **PostgreSQL;**
- **MySQL;**
- **O próprio Access ou Paradox;**
- **Entre outros...**

Por último, temos que conceituar um sistema de **banco de dados** como o conjunto de quatro componentes básicos: **dados**, **hardware**, **software** e **usuários**.

- ***Sistema de Banco de dados Relacionais***

Um **Sistema Gerenciador de Banco de Dados Relacional** (SGBDR) é um software que controla o armazenamento, recuperação, exclusão, segurança e integridade dos dados em um banco de dados. Um banco de dados relacional armazena dados em tabelas.Os dados de uma simples "instância" de uma tabela são armazenados como uma linha.

- ***Sistema de Banco de Dados orientado a Objetos:***

Um **banco de dados orientado a objetos** é um banco de dados em que cada informação é armazenada na forma de objetos, ou seja, utiliza a estrutura de dados denominada orientação a objetos, a qual permeia as linguagens mais modernas. Começou a ser comercialmente viável em 1980. O gerenciador do banco de dados para um orientado a objeto é referenciado por vários como **ODBMS** ou **OODBMS**.

Existem dois fatores principais que levam à adoção da tecnologia de banco de dados orientados a objetos. A primeira, é que, em um banco de dados relacional, se torna difícil de manipular com dados complexos (esta dificuldade se dá pois o modelo relacional se baseia menos no senso comum relativo ao modelo de dados necessário ao projeto e mais nas contingências práticas do armazenamento eletrônico). O segundo fator é que os dados são geralmente manipulados pela aplicação escrita usando linguagens de programação orientada a objetos, como C++, C#, Java, Python ou Delphi (Object Pascal), e o código precisa ser traduzido entre a representação do dado e as tuplas da tabela relacional, o que além de ser uma operação tediosa de ser escrita, consome tempo. Esta perda entre os modelos usados para representar a informação na aplicação e no banco de dados é também chamada de "perda por resistência".

Vantagens e Desvantagens em se utilizar este B.D:

Benchmarks entre ODBMS's e relacionais DBMS's têm mostrado que ODBMS podem ser claramente superiores para certos tipos de tarefas. A principal razão para isto é que várias operações são feitas utilizando interfaces navegacionais ao invés das relacionais, e o acesso navegacional é geralmente implementado de forma muito eficiente por ponteiros[4].

Críticos das tecnologias baseadas em Bancos de Dados Navegacionais, como os ODBMS, sugerem que as técnicas baseadas em ponteiros são otimizadas para "rotas de pesquisa" ou pontos de vista muito específicos. Entretanto, para o propósito de consultas gerais a mesma informação, técnicas baseadas em ponteiros tenderão a ser mais lentas e mais difíceis de se formular do que as relacionais. Desta maneira, a abordagem navegacional parece simplificar para usos dos específicos conhecidos às custas do uso geral, ignorando usos futuros.

Outra coisa que trabalha contra os ODBMS parece ser a perda da interoperabilidade com um grande número de ferramentas/características que são tidas como certas no mundo SQL, incluindo a indústria de padrões de conectividade, ferramentas de relatório, ferramentas de OLAP e backup, e padrões de recuperação. Adicionalmente, banco de dados orientado a objetos perdem o fundamento formal matemático, ao contrário do modelo relacional, e isto às vezes conduz a fraqueza na sustentação da consulta. Entretanto esta objeção é descartada pelo fato que alguns ODBMS's suportam totalmente o SQL em adição ao acesso navegacional (Objectivity/SQL++). Mas, o uso eficaz pode requerer acordos para manter ambos os paradigmas sincronizados.

De fato, há uma tensão intrínseca entre a noção de encapsulamento, que esconde os dados e somente os disponibiliza através de uma interface de métodos publicados, e o pressuposto de muitas tecnologias de bancos de dados, de que estes dados podem ser acessados por consultas baseadas em seu conteúdo ao invés de caminhos predefinidos. O pensamento centrado em bancos de dados tende a ver o mundo através de forma declarativa e dirigida a uma visão de atributos, enquanto a OOP tenta ver o mundo através um ponto de vista comportamental. Esta é uma das várias "perdas por resistência" que envolvem OOP e banco de dados

Embora alguns afirmem que a tecnologia de banco de dados orientado a objetos fracassou, os argumentos essenciais em seu favor permanecem válidos, e as tentativas de integrar as funcionalidades de bancos de dados mais próxima as linguagens de programação orientadas a objeto continuam tanto nas comunidades de pesquisa quanto nas industriais.

Exemplos:

- ODMG:

  O ODMG (*Object Database Management Group*) era um consórcio de vendedores de banco de dados orientados a objetos e mapeadores objeto-relacionais, membros da comunidade acadêmica, e parceiros interessados. A meta era criar um conjunto de especificações que permitiriam a portabilidade das aplicações que armazenam objetos em sistemas de gerenciamento de banco de dados. Foram publicadas várias versões desta especificação. O último *release* foi a ODMG 3.0. Em 2001, a maioria dos principais vendedores de banco de dados orientado a objetos e mapeadores de objeto-relacionais reivindicaram a conformidade com a ODMG Java Language Binding. A conformidade

com os demais componentes da especificação foi variada.[5] Em 2001, o ODMG Java Language Binding foi submetido para o Java Community Process como base para a especificação Java Data Objects. As companhias membros do ODMG decidiram então concentrar esforços na especificação do Java Data Objects. Como resultado, a ODMG se dissolveu em 2001.

- ODL:

A Linguagem de Definição de Objeto (ODL) é a linguagem de especificação que define a interface para os tipos de objeto em conformidade com o Modelo de Objeto ODMG. Frequentemente abreviado pela sigla ODL.O objetivo desta linguagem é definir a estrutura de um diagrama de relação de entidade. A ODL é projetada para dar suporte ás construções semânticas do modelo de objeto ODMG e é independente de qualquer linguagem de programação em particular. Seu uso principal é para criar especificações de objeto, ou seja, classes e interfaces. Logo, a ODL não é uma linguagem de programação completa.Por exemplo, um usuário pode especificar um esquema de banco de dados na ODL independentemente de qualquer linguagem especifica para indicar como as construções ODL podem ser mapeadas em construções nas linguagens de programação especificas.

```
Type Date Tuple (year, day, month) Type year, day, month integer
```

```
interface Manager { keys id; attribute String name; attribute String
phone; relationShip Set<Employee> employees inverse Employee::manager}
```

```
interface Employee { keys id; attribute String name; attribute Date
Start_Date; relationShip Manager manager inverse Manager::employees }
```

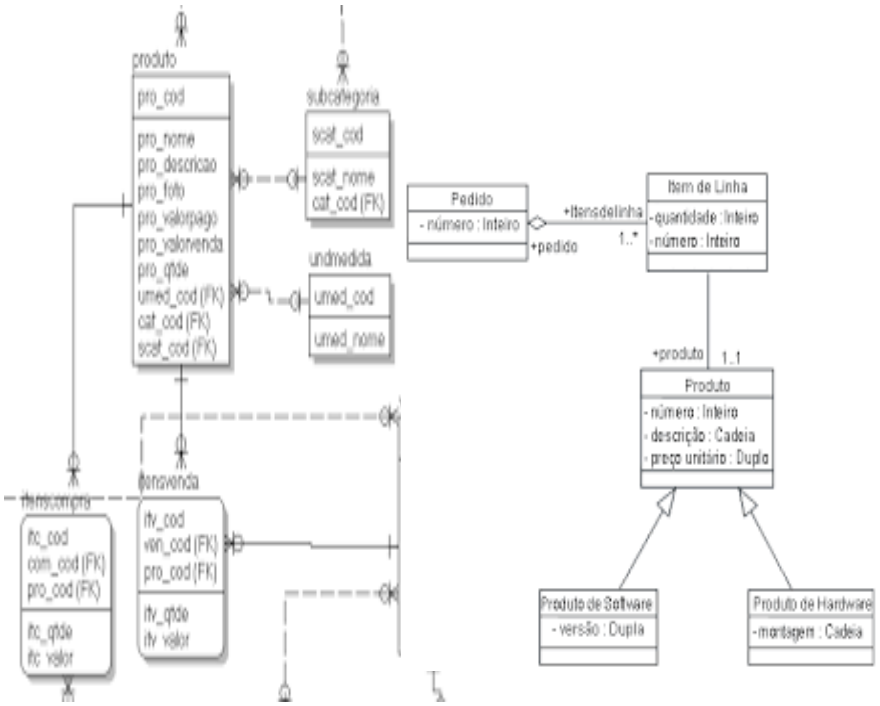
- OQL:

O Object Query Language (OQL) é um padrão de linguagem de consulta para bancos de dados orientados a objetos modelados a partir do SQL. OQL foi desenvolvido pelo Object Data Management Group (ODMG). Devido à sua complexidade geral, ninguém se aventurou a implementar o OQL completo. O OQL influenciou o design de algumas das novas linguagens de consulta, como JDOQL e EJB QL, mas elas não podem ser consideradas como diferentes tipos de OQL. A linguagem de consulta de objeto OQL é a linguagem proposta para o modelo de objeto ODMG. Ela foi projetada para trabalhar de perto com as linguagens de programação para as quais um binding ODMG é definido, como C++ e JAVA. Logo, uma consulta OQL embutida em uma dessas linguagens de programação pode retornar objetos que combinam com o sistema de tipos dessa linguagem. Além disso, as implementações de operações de classe em um esquema ODMG podem ter seu código escrito nessas linguagens de programação. A sintaxe OQL para consultas é semelhante a sintaxe da linguagem de consulta do padrão relacional, SQL, com recursos adicionais para os conceitos ODMG, como identidade, polimorfismo e relacionamentos.

Consulta Simples:

O exemplo a seguir ilustra como alguém pode recuperar a velocidade da CPU de todos os computadores com mais de 64MB de RAM de um banco de dados fictício :

```
SELECT pc.cpuspeed
    FROM PCs pc
WHERE pc.ram > 64;
```



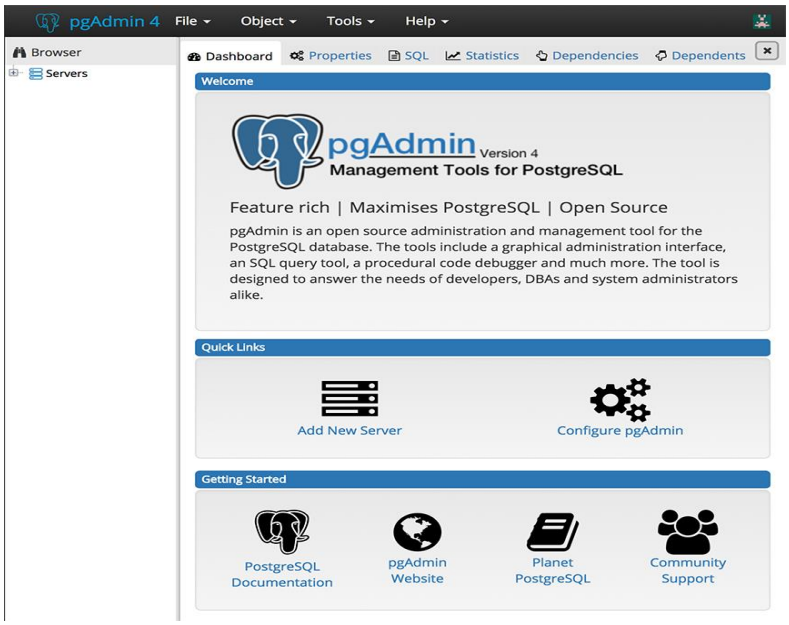
- ***Banco de Dados Objeto Relacional (BDOR)***

Esse tipo de banco de dados utiliza as tabelas do modelo relacional, mas nelas são armazenados objetos, com seus atributos e comportamento. É um banco que permite a manipulação de dados complexos, mesmo com desempenho inferior ao relacional. Utiliza os conceitos de supertabelas, supertipos, herança, reutilização de código, encapsulamento, controle de identidade de objetos (OID), referência a objetos, consultas avançadas e alta proteção dos dados.

O BDOR pode ser utilizado em aplicações de armazenamento de imagens (obtidas por satélite ou de alguma outra forma digital), projetos de arquitetura, dados sobre o espaço (regiões geográficas, criação de mapas), sistemas de informações geoespaciais, entre outros.

Exemplos:

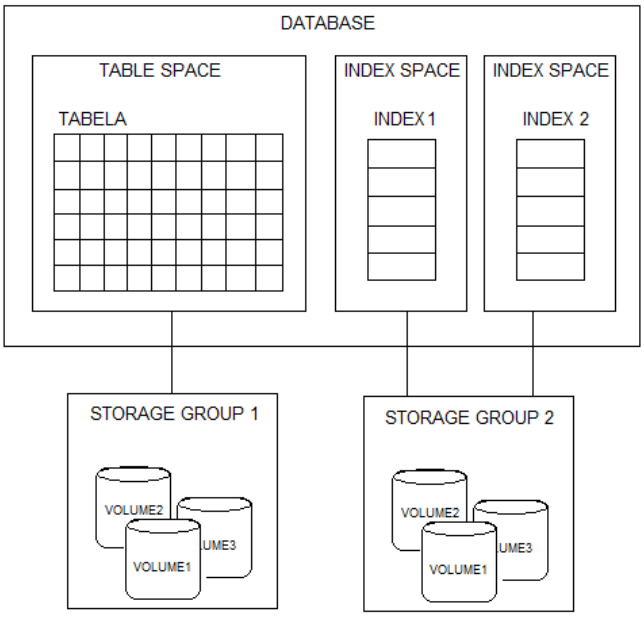
1) PostgreSQL: está sob a licença *PostgreSQL Licence*, que permite o livre uso, modificação e distribuição. Ele tem vários recursos, com destaque ao suporte a operações assíncronas e busca de textos avançada.



SITE: https://www.postgresql.org/

2) DB2: produzido pela IBM, é o banco de dados preferencial para soluções corporativas e robustas, porque manipula tanto cargas de trabalho de transação de alto volume quanto cargas de trabalho com uso intenso de dados.

Estrutura de armazenamento de dados:



SITE: https://www.ibm.com/developerworks/br/downloads/im/db2/index.html

- ***Banco de Dados NoSQL***

Os bancos de dados NoSQL não são necessariamente parecidos entre si, apenas são classificados dessa maneira para se diferenciarem dos bancos relacionais. Eles são

baseados em documentos e permitem armazenar e recuperar dados em formatos diferentes de tabelas. Esse tipo de banco de dados é mais flexível e redimensionável: é possível adicionar novos dados sem a necessidade de defini-los com antecedência no esquema do banco de dados, permitindo o processamento rápido de grandes volumes de dados não estruturados, semiestruturados e estruturados.

Exemplos:

1) MongoDB: é um banco de dados que armazena documentos no estilo JSON, assim, o desenvolvedor pode carregar para o banco de dados qualquer documento que tenha a sintaxe exigida pelo JSON, sem se preocupar com a estrutura de dados em si:

Ele também não usa o conceito de tabelas, mas de coleções. É de código aberto.
SITE: https://www.mongodb.com/

2) Elasticsearch: realiza buscas e análise de dados em grandes volumes, permitindo indexar documentos e realizar buscas em (quase) tempo real utilizando o princípio do Índice Invertido. Esse banco de dados suporta um grande volume de dados sem perder performance e também fornece uma API para realizar análises sobre os dados recuperados como resultado da busca. É de código aberto.
Armazenamento em indexação invertida:

ID	Text
1	Baseball is played during summer months.
2	Summer is the time for picnics here.
3	Months later we found out why.
4	Why is summer so hot here

↑ Sample document data

Dictionary and posting lists →

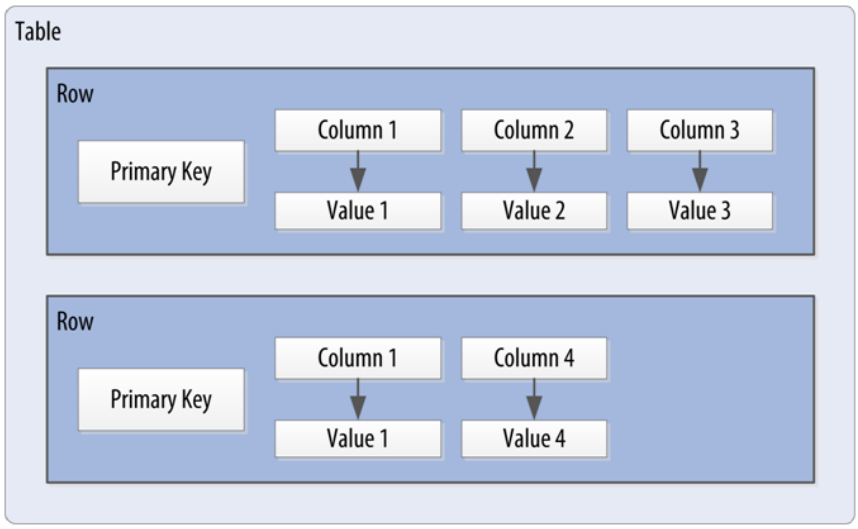
Term	Freq	Document ids
baseball	1	[1]
during	1	[1]
found	1	[3]
here	2	[2], [4]
hot	1	[4]
is	3	[1], [2], [4]
months	2	[1], [3]
summer	3	[1], [2], [4]
the	1	[2]
why	2	[3], [4]

SITE: https://www.elastic.co/pt/

3) Redis: é um banco de dados de código aberto e oferece um conjunto de estruturas versáteis de dados na memória que permite a fácil criação de várias aplicações personalizadas. Por conta da sua velocidade e facilidade de uso, o Redis é uma

escolha em alta demanda para aplicações web e móveis, como também de jogos, tecnologia de anúncios e IoT, que exigem o melhor desempenho do mercado.
SITE: https://redis.io/

4) Cassandra: é um banco de dados de código aberto e colunar, ou seja, armazena os dados em colunas, criando um relacionamento através de um id. É altamente escalável e distribuído e, como não tem transações em determinados momentos, a leitura dos dados pode não ter consistência.
Representação de uma tabela:



SITE: http://cassandra.apache.org/

Referências Bibliográficas

DUARTE, Eber, M. **SQL e Programação de Banco de Dados. Disponível em:** <http://www.criarweb.com/artigos/667.php> Acessado em: 26 de Nov. 2010.

BANCO DE DADOS : INTRODUÇÃO disponível em:<http://www.macoratti.net/banco.htm> Acessado em: 26 de Nov. 2010

Disponível conceito de Banco de Dados e exemplos em https://www.infoescola.com/informatica/banco-de-dados/

Disponível aplicação de Banco de Dados em https://pt.wikipedia.org/wiki/Banco_de_dados

AMAZON WEB SERVICES. **O que é o Redis?**, c2019. Disponível em: <https://aws.amazon.com/pt/elasticache/what-is-redis/>. Acesso em: 26 de maio de 2019.

BOSCARINO, Aylan de Sousa. **O que é Elasticseach?** Conheça o Elastiseach e porque ele é uma poderosa ferramenta para realização de buscas e análise de grandes volumes de dados. **Devmedia**, 2018. Disponível em: <https://www.devmedia.com.br/o-que-e-elasticsearch/40207>. Acesso em: 26 de maio de 2019.

RACKSPACE. **Biblioteca da Rackspace na nuvem**: o que é um banco de dados NoSQL?, c2019. Disponível em: <https://www.rackspace.com/pt-br/library/what-is-a-nosql-database>. Acesso em: 26 de maio de 2019.

CRIVELINI, Wagner. **Começando a entender os bancos de dados NoSQL:** o MongoDB. **Academia UOL meu negócio**, c2018 - 2019. Disponível em: <https://meunegocio.uol.com.br/academia/tecnologia/comecando-a-entender-os-bancos-de-dados-nosql-o-mongodb.html#rmcl>. Acesso em: 26 de maio de 2019.

VASCONCELOS, Rafael Oliveira. **Artigo** – Comparativo entre banco de dados orientado a objetos (BDOO) e bancos de dados objeto relacional (BDOR) .: parte 2. **Blog sobre tecnologia**, 2009. Disponível em: <https://rafaeloliveirav.wordpress.com/2009/06/19/artigo-comparativo-entre-banco-de-dados-orientado-a-objetos-bdoo-e-bancos-de-dados-objeto-relacional-bdor-parte-2/>. Acesso em: 25 de maio de 2018.

MASSINO, Eduardo Galvani; ROLAND, Carlos Eduardo de França. Banco de dados objeto-relacional para aplicações web. **Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica**, Franca, São Paulo, 2015.

MOTA, Fernando Jorge. SGBDs – o que são, o que fazem e alguns fatos importantes sobre eles. **Dicas e ferramentas para facilitar o seu dia-a-dia como desenvolvedor** [s.d.]. Disponível em: <https://fjorgemota.com/sgbds-sistemas-gerenciadores-de-

bancos-de-dados-o-que-sao-como-vivem-e-onde-podem-ser-encontrados/>. Acesso em: 25 de maio de 2019.

SABINO, Alexandre. Apache Cassandra: confira absolutamente tudo a respeito. **NSTECH**, 2018. Disponível em: <https://medium.com/nstech/apache-cassandra-8250e9f30942>. Acesso em: 27 de maio de 2019.