



# MODELAGEM DE DADOS

## Aula 1 – Bancos de Dados e SGBDs

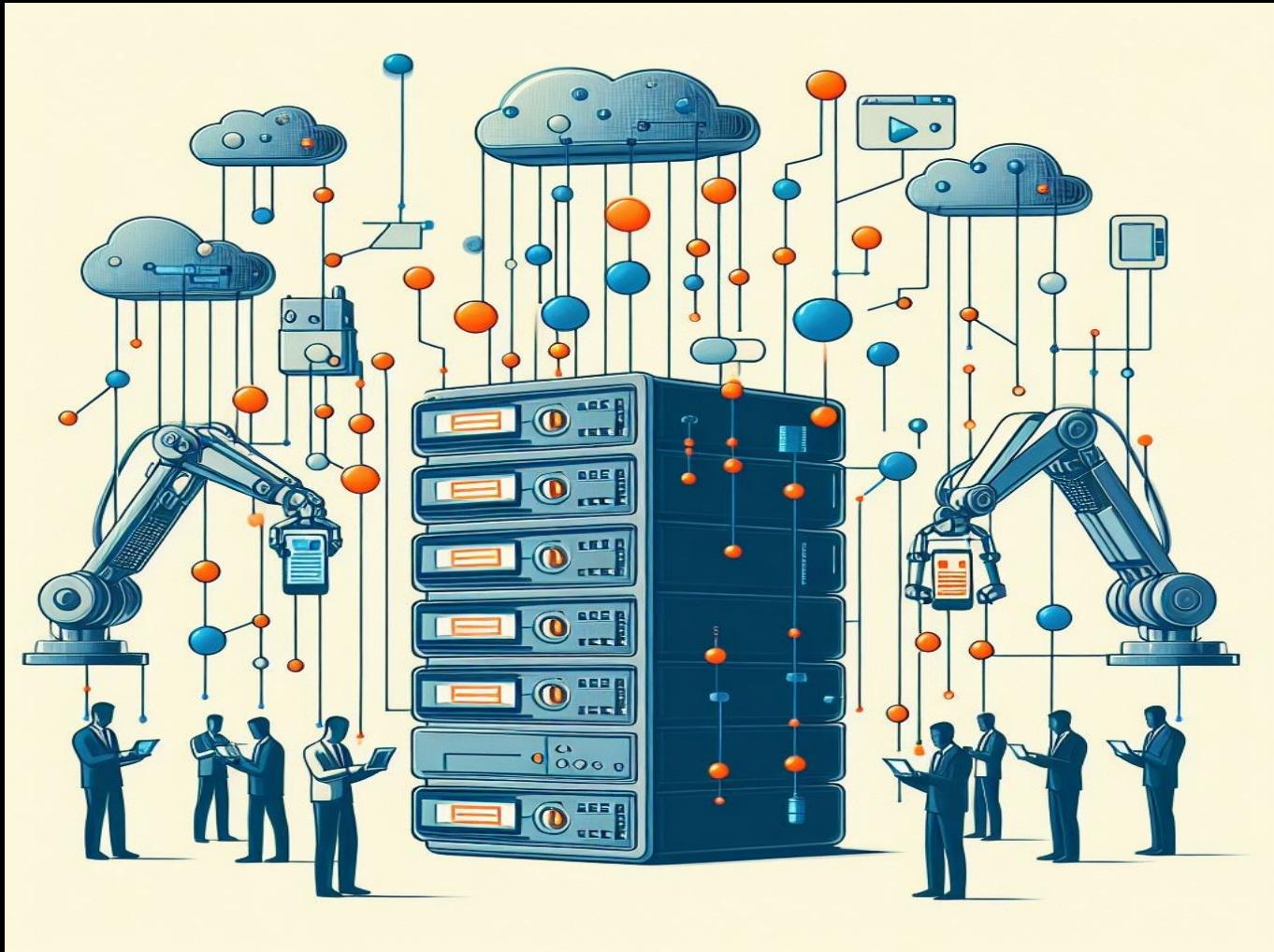
**Curso de Ciência da Computação**

Dr. Rodrigo Xavier de Almeida Leão  
Cientista de Dados

# Bancos de Dados

Coleção de dados relacionados que podem ser inseridos, atualizados, e recuperados e que possuem um significado implícito.

## Quando interagimos com Bancos de Dados?



# Quando interagimos com Banco de Dados?

**Cadastro em um site:** nome, endereço de e-mail, senha, entre outros dados pessoais.

**Realização de uma compra online:** informações de pagamento, como número do cartão de crédito, são inseridas em um banco de dados para processamento seguro da transação.

**Reserva de um voo:** informações de viagem, como destino, datas e número de passageiros.

**Consulta médica:** Durante uma consulta médica, o médico registra suas informações médicas, histórico de saúde e prescrições em um sistema de gerenciamento de registros médicos eletrônicos (EMR).

# Quando interagimos com Banco de Dados?

**Consulta de informações em um sistema de biblioteca:** informações sobre os livros disponíveis, como título, autor, número de cópias disponíveis e localização.

**Acesso a aplicativos de redes sociais:** informações sobre seus amigos, publicações, curtidas e outras atividades.

**Acesso a serviços bancários online:** Ao verificar o saldo da sua conta, transferir dinheiro ou pagar contas por meio de um aplicativo ou site bancário, você está interagindo com um banco de dados que armazena suas informações financeiras.

# Volume de Dados

Como?

- Evitar redundância e inconsistência
- Garantir a segurança e integridade
- Garantir a Interoperabilidade



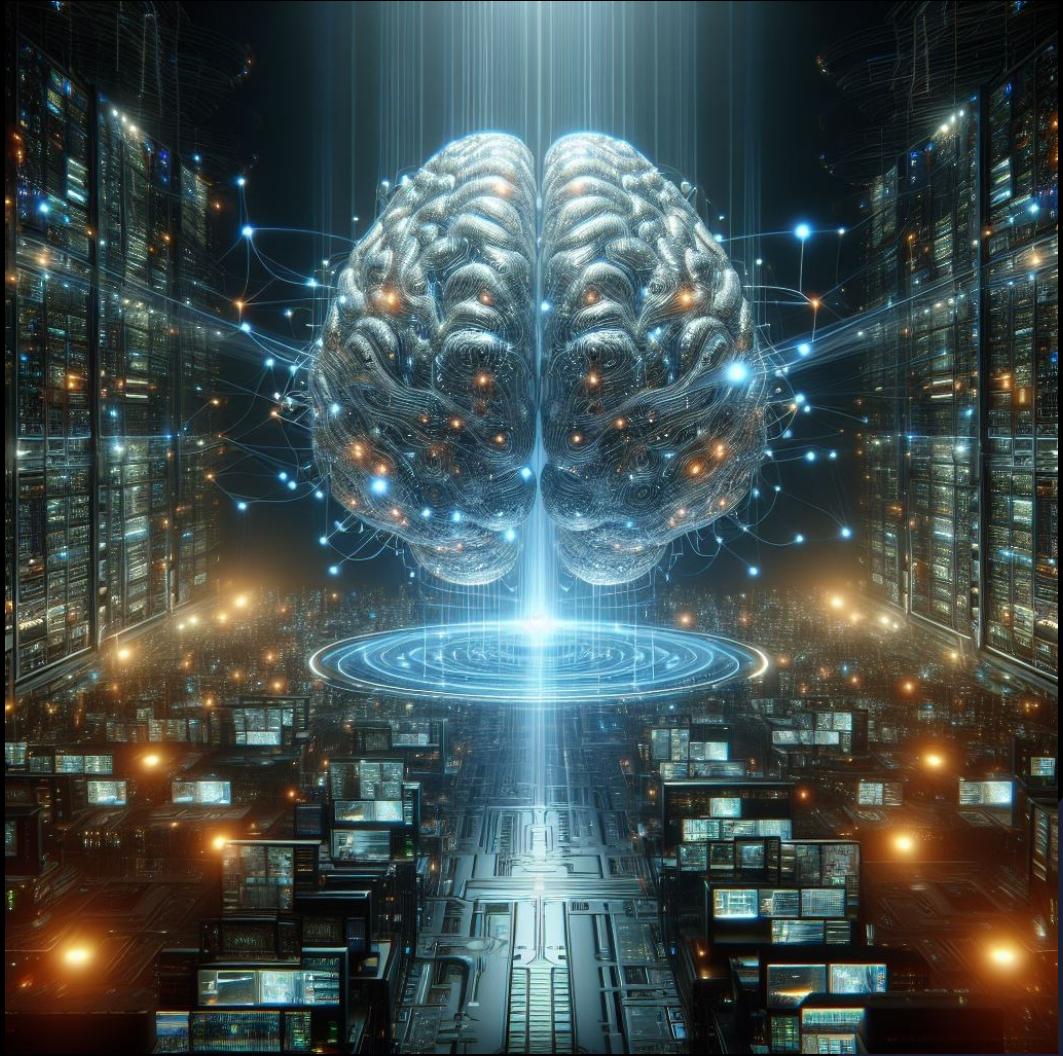
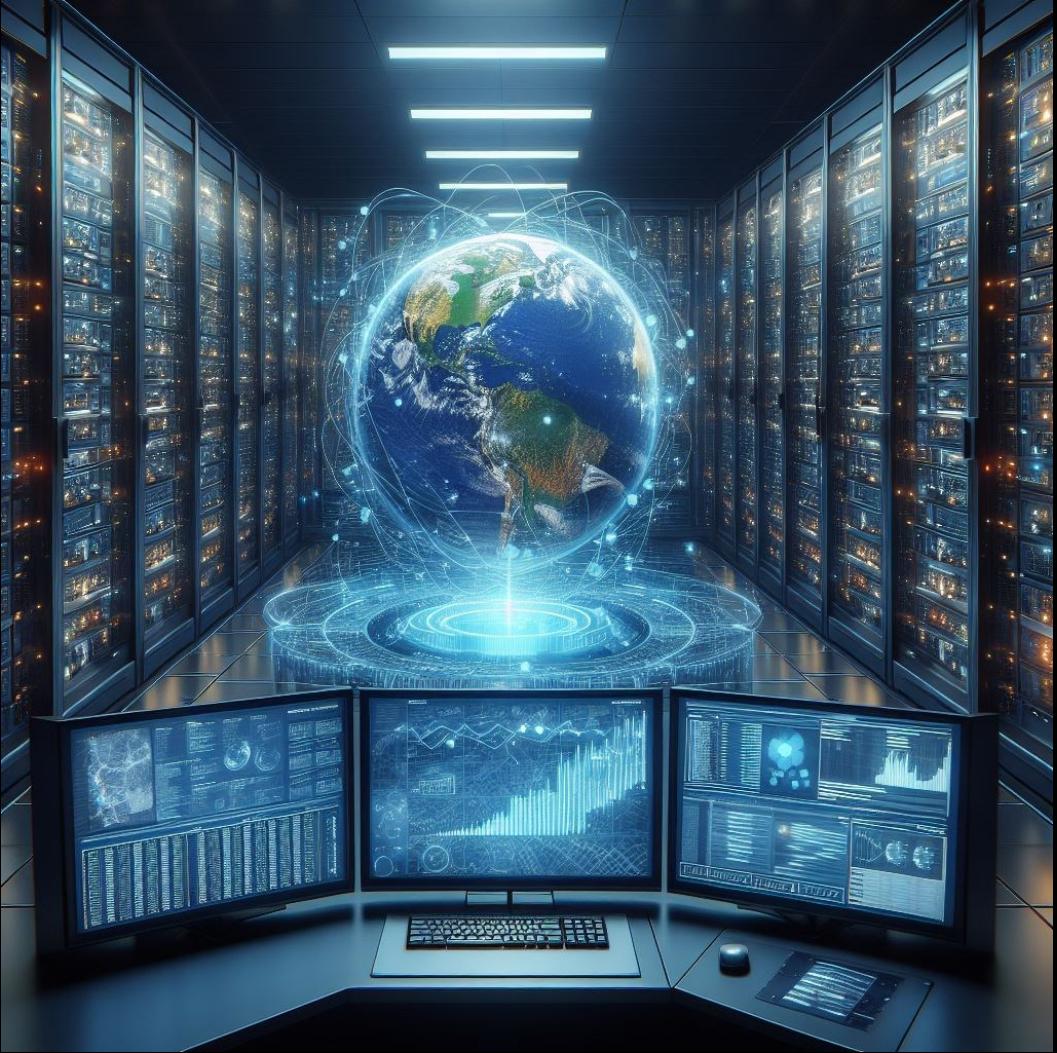
# BIG DATA



# BIG DATA



# BIG DATA



# Sistemas Gerenciadores de Bancos de Dados (SGDB)

- Quando projetamos um banco de dados, precisamos saber quais serão as aplicações que utilizarão o banco projetado.
- É necessário um dimensionamento a fim de poder indicar o SGBD mais apropriado para o cliente que deseja o software.
- Analisar as necessidades do cliente, levantar os requisitos do software e projetar soluções para cada um.

# Sistemas Gerenciadores de Bancos de Dados (SGDB)

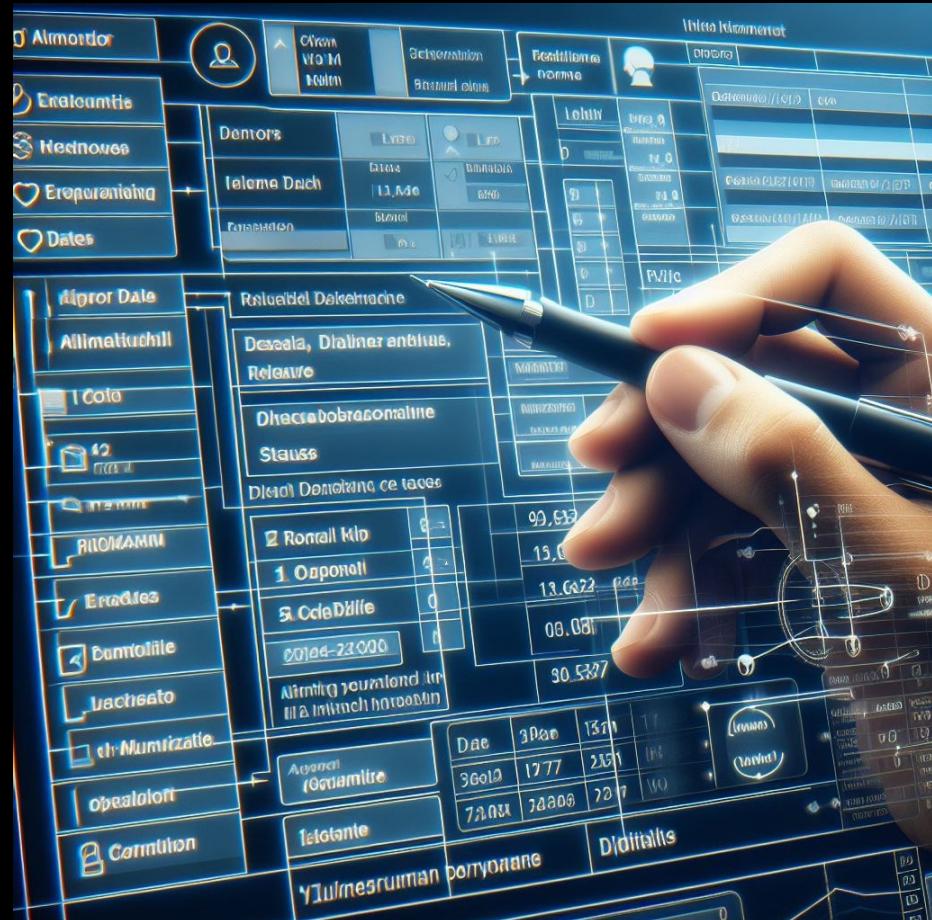
- Primeiramente, você irá até a secretaria e lá passará os seus dados pessoais, informações acadêmicas, entre muitas outras informações, que deverão ser cadastradas em um banco de dados.
- Para gerar boletos de pagamento, você não precisará informar seus dados novamente, pois o departamento financeiro já possui suas informações no banco de dados.
- São dois sistemas diferentes usando o mesmo banco de dados ou também, podemos dizer, usando a mesma base de dados.

# Sistemas Gerenciadores de Bancos de Dados (SGDB)

- O SGBD tem por objetivo facilitar a vida do programador ou analista, deixando livre para pensar na modelagem e não ficar pensando em questões técnicas de armazenamento de dados.
- Os Sistemas Gerenciadores de Bancos de Dados (SGBDs) são softwares projetados para facilitar o gerenciamento, armazenamento, recuperação e manipulação de grandes volumes de dados.

# Sistemas Gerenciadores de Bancos de Dados (SGDB)

- Eles oferecem uma interface entre os usuários e os bancos de dados subjacentes, permitindo que os usuários realizem operações como inserção, consulta, atualização e exclusão de dados de forma eficiente e segura

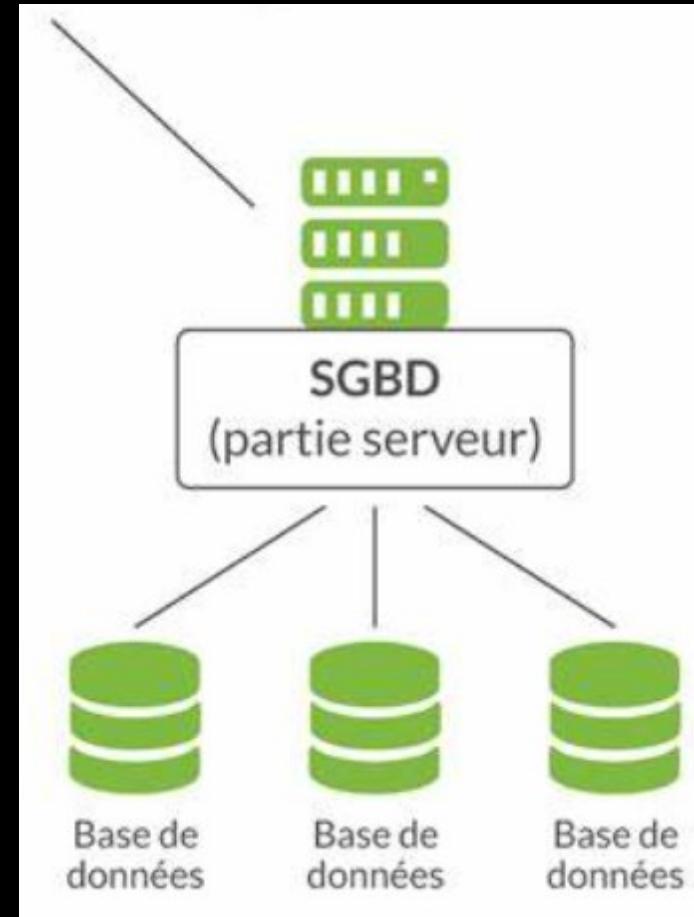


# SGBD's

- O Sistema Gerenciador de Banco de Dados é constituído por um conjunto de dados associados a um conjunto de programas para acesso a esses dados.
- O SGBD tem como finalidade a garantia de que as informações que foram inseridas no banco de dados estejam seguras, protegendo de ataques indevidos quanto ao seu acesso ou problemas ocasionados por erros de software ou hardware.

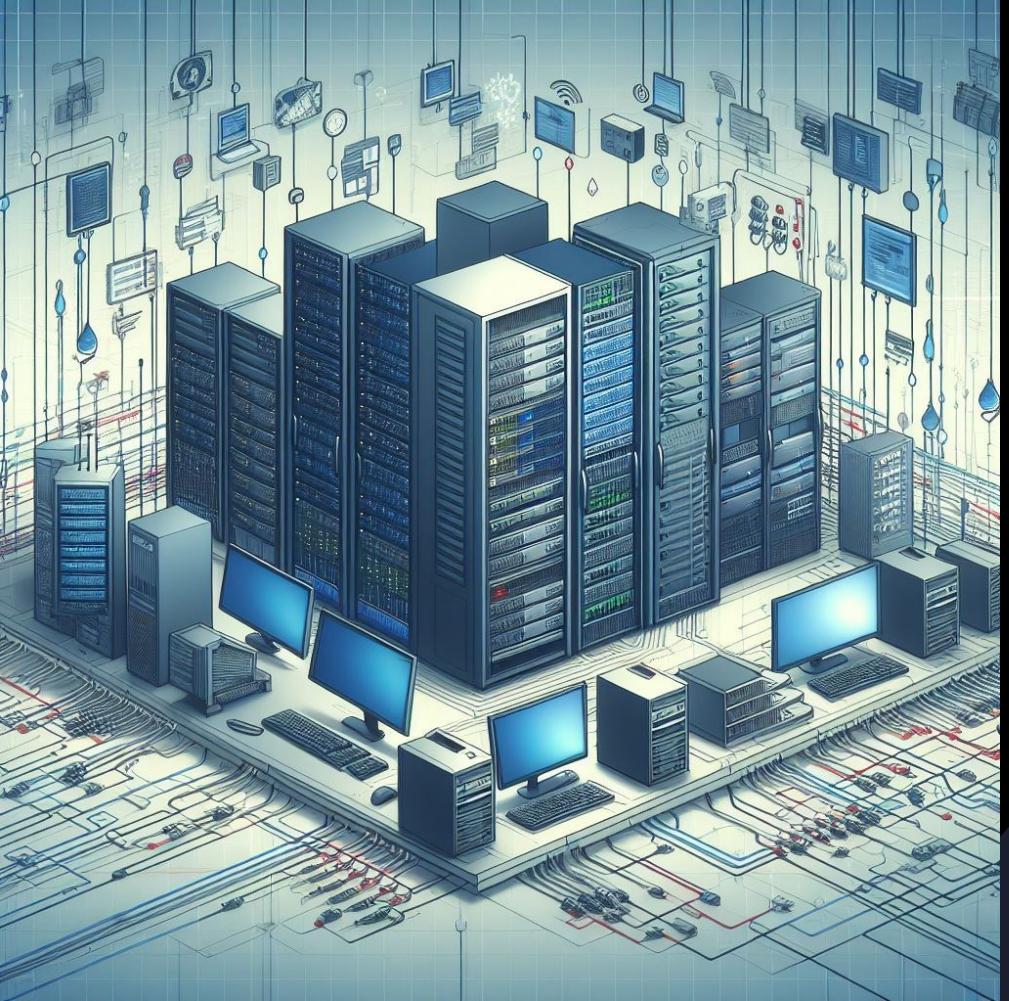
# SGBD's

- O SGBD pode ser distribuído por diversos computadores, no mesmo local ou até em locais diferentes (espaços, cidades, países).
- Caso o SGBD esteja em locais físicos diferentes, cada um passa a receber o nome de nó, e uma operação realizada no banco de dados pode ser executada em um ou em mais nós.

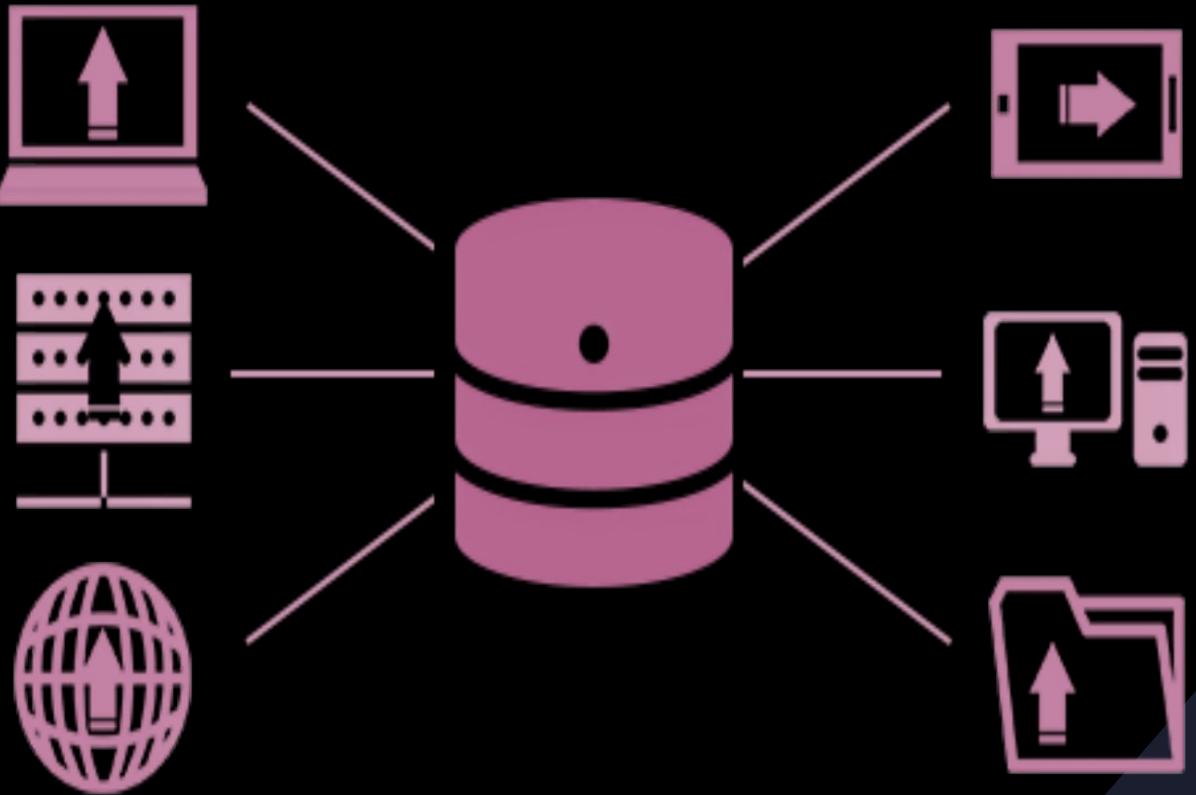


# Aplicação

- Quando nos referenciamos ao termo aplicação, estamos nos referindo aos softwares que estarão se beneficiando dos dados inseridos em um banco de dados.

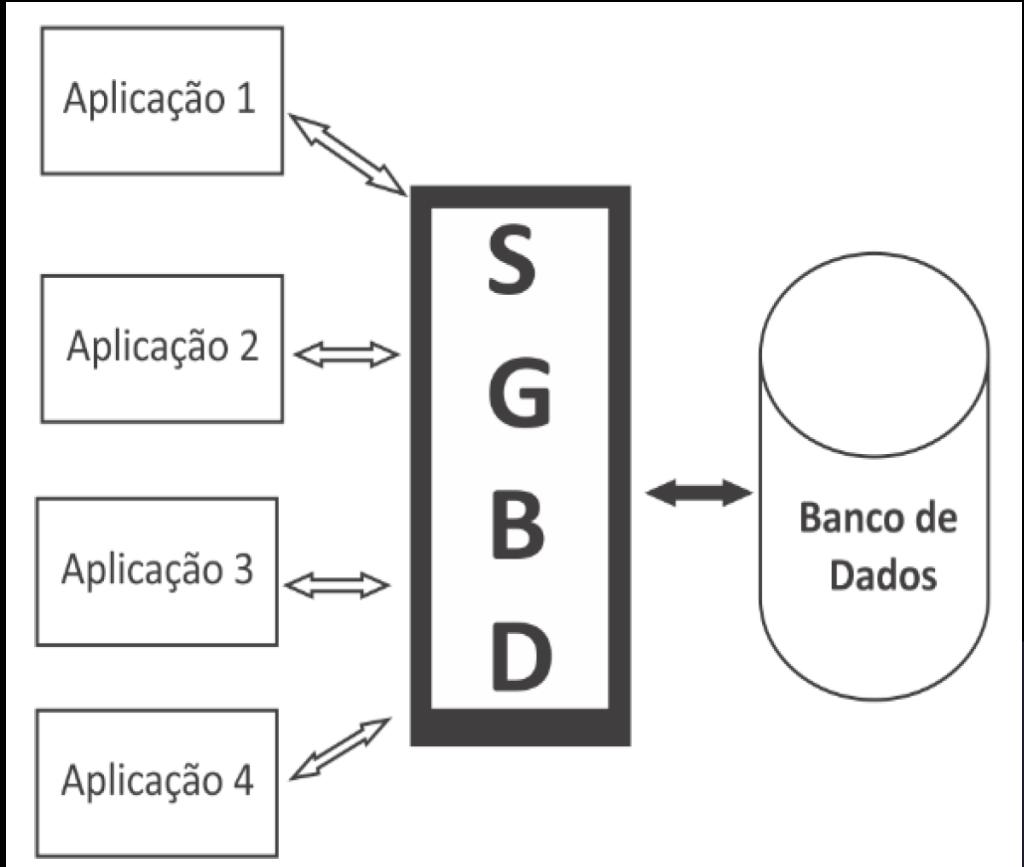


# Aplicação



# Aplicação

- O acesso ao banco de dados por diversas aplicações necessita de regras específicas para garantir tanto a **segurança** quanto a **integridade** das informações inseridas.
- Os SGBD executam essas garantias.



# CONTROLE DE OCORRÊNCIAS

- Dois vendedores acessam simultaneamente o registro e vendem a mesma geladeira para seus clientes.
- Para este tipo de evento damos o nome de controle de concorrência, uma das finalidades essenciais de um SGBD.
- São técnicas utilizadas para garantir a propriedade de isolamento de transações que estão sendo executadas ao mesmo tempo.

# OUTRAS FUNÇÕES DO SGBD's

- Permitir aos seus usuários a pesquisa em um banco de dados para recuperar uma determinada informação, alterar e gerar relatórios das informações.
- Proteção e a recuperação dos dados quando houver problemas de hardware ou software.
- Controle de acesso.
- Administração da redundância e a restrição de integridade.

# TRANSAÇÃO

- É um processo ou um determinado programa que pode incluir vários bancos de dados ou somente uma parte do banco de dados, realizando atividades como consultas, alterações e até exclusão de informações da base de dados.
- É uma consequência da efetivação de um programa (ou uma rotina) que acessa e possivelmente atualiza vários itens de dados.
- A transação é o resultado da execução de um programa de usuário.

# LOG DE TRANSAÇÃO

- Para recuperar-se de uma transação com falhas, possui um log para registrar todas as operações realizadas em dados.
- Funciona como um histórico das modificações.
- Caso haja erro, log permite a recuperação dos dados para que eles voltem ao estado inicial.

# LOG DE TRANSAÇÃO



# REQUISITOS DE UM SGBD

- Atomicity - Atomicidade
- Consistency - Consistência
- Isolation - Isolamento
- Durability - Durabilidade

# ATOMICIDADE

- Isso garante que todas as operações em uma transação sejam executadas com sucesso ou que nenhuma delas seja executada.
- Uma transação é uma unidade atômica de trabalho na qual todas as operações são realizadas com sucesso ou nenhuma delas é realizada.
- Se uma parte da transação falhar, todas as operações da transação serão revertidas para garantir a consistência dos dados

- Suponha que estamos aumentando os salários dos funcionários (este aumento é uma alteração em uma tabela e, uma transação) e que durante a atualização faltou luz. Somente uma parte dos funcionários receberá o aumento caso não haja a verificação de atomicidade.
- A ideia por trás da garantia de atomicidade é que o sistema de banco de dados mantenha um registro (em disco) dos antigos valores de quaisquer dados a serem alterados. Caso haja algum problema durante a realização da transação, o SGBD reestabelece os dados antigos, como se nunca tivessem sidos modificados.

# CONSISTÊNCIA

- Este princípio garante que o banco de dados permaneça em um estado consistente antes e depois da execução de uma transação.
- Isso significa que todas as restrições de integridade referencial, regras de negócio e restrições de integridade do banco de dados devem ser preservadas durante e após a execução da transação.
- Assim que a transação for finalizada, todos os dados devem estar íntegros. Um exemplo seria a soma de dois valores. Após uma transação, os valores iniciais não podem ser alterados.

# ISOLAMENTO

- O princípio de isolamento garante que as transações em execução sejam isoladas umas das outras, de modo que uma transação não possa interferir no resultado de outra transação em execução concorrentemente.
- Isso é alcançado através de técnicas de controle de concorrência, como bloqueio de dados, para evitar problemas como leituras sujas, leituras não repetíveis e escritas fantasmas.

# ISOLAMENTO

- Somente após o término de uma transação, ela estará liberada para receber outras.
- Mesmo asseguradas as propriedades de atomicidade e consistência para cada transação, a intercalação das operações de várias transações concorrentes pode resultar em inconsistências (erros nos resultados e/ou nos dados).
- Alterações feitas por transações simultâneas precisam ser isoladas das alterações feitas por qualquer outra transação simultânea.

# DURABILIDADE

- Este princípio garante que os efeitos de uma transação persistam no banco de dados mesmo em caso de falha do sistema, como uma queda de energia.
- Uma vez que uma transação é confirmada (*commit*), suas alterações são permanentemente gravadas no banco de dados e não podem ser desfeitas, mesmo em caso de falha posterior do sistema.

# DURABILIDADE

- É a certeza de que após uma transação ser realizada com sucesso, os resultados fiquem gravados no banco de dados, mesmo se algum problema tenha ocorrido, como a queda do sistema.
- A durabilidade ou persistência (como também é conhecida) em um meio de armazenamento confiável e seguro é um dos requisitos mais importantes de um Sistema Gerenciador de Banco de Dados.

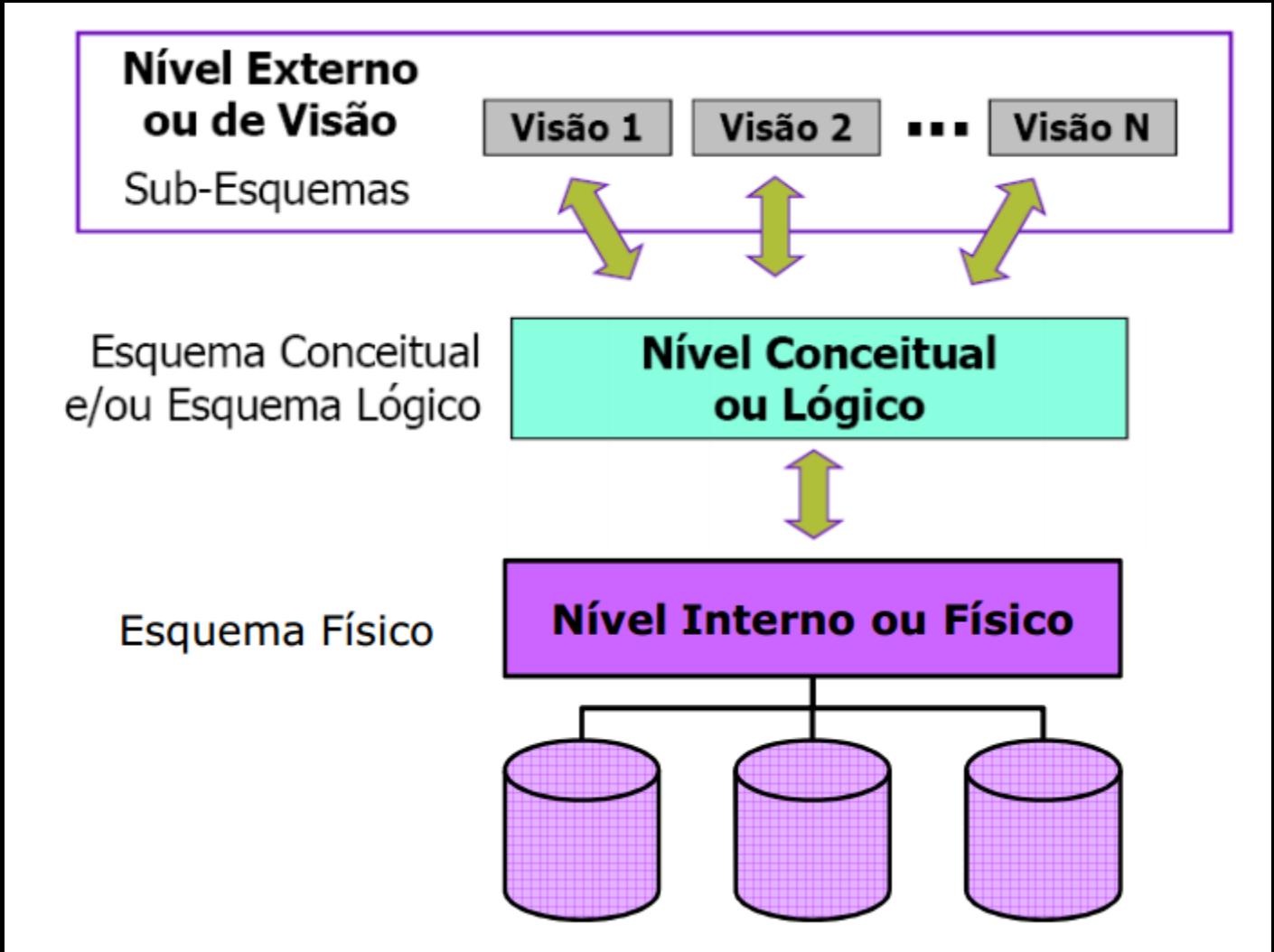
# ABSTRAÇÃO E VISÃO

- SGBD oferece aos usuários uma representação conceitual de dados, omitindo vários detalhes, por exemplo, como os dados realmente são guardados ou como as transações são realizadas no banco de dados.
- Essa representação de modelo de dados é informalmente conhecida como abstração de dados.
- Uma visão (ou view) pode ser uma parte de uma base de dados, podendo ser resultantes de pesquisas que retornam parte das informações armazenadas.

# ABSTRAÇÃO E VISÃO

- Uma visão (ou view) pode ser considerada como uma tabela virtual ou
- uma consulta armazenada, permitindo mais do que somente visualizar
- os dados, implementar algumas restrições.
- Um professor só tem acesso aos dados de seus alunos matriculados na sua disciplina. O docente, ao fazer uma pesquisa, não precisa ver as notas dos alunos em outras disciplinas e muito menos os dados pessoais ou financeiros dos alunos.

# ABSTRAÇÃO E VISÃO



phpMyAdmin

Banco de dados: **bd\_tutorial** | Tabelas: **103** | Último backup: **Não há backup**

**Filtros:** Tabelas | Estrutura | SQL | Procurar | Consulta | Exportar | Importar | Operações | Rótulas | Rotinas | Eventos | Gatilhos

**Linhas:** 1.555 | **Tipo:** Início | **Cotação:** Sobrecarga

**Ação:** Visualizar | Estrutura | Preencher | Inserir | Limpar | Eliminar

**Linhas:** 1.555 | **Tipo:** Util | **Cotação:** Sobrecarga

**Tables**

Table	Structure	Insert	Delete			
jos_ac_adl	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_ac_profiles	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_ac_stats	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_ac_storage	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_banner	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_bannerclick	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_categories	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_ckfields	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_ckforms	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_ckprofilefields	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_ckprofiles	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_components	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_contact_details	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_content	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_content_frontpage	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_content_rating	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_core_act_aro	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_core_act_aro_group	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_core_act_aro_groups	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_core_act_aro_map	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_core_act_aro_sadic	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_core_log_items	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_core_log_searches	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_core_menus	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_core_plugins	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_core_sessions	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_core_stats	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_core_templates_menu	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_newsfeeds	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_plugins	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_polt_menu	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_session	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_stats_agents	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
jos_templates_menu	Visualizar	Estrutura	Preencher	Inserir	Limpar	Eliminar
<b>103 tabelas</b>	<b>Soma</b>	<b>1.555</b>	<b>MyISAM</b>	<b>latin1_swedish_ci</b>	<b>55</b>	
<input type="checkbox"/> Marcar todos <input type="checkbox"/> Com marcados: <b>▼</b>						
<a href="#">Imprimir view</a> <a href="#">Dicionário de dados</a>						

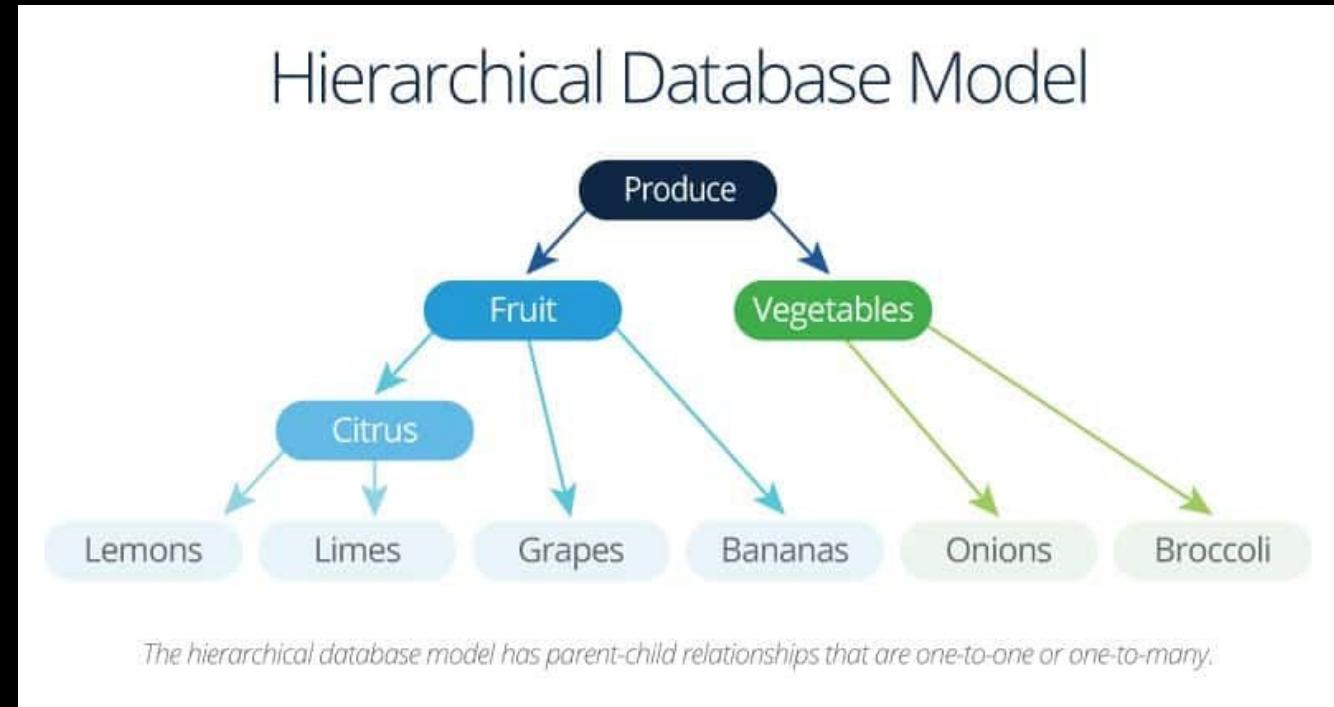
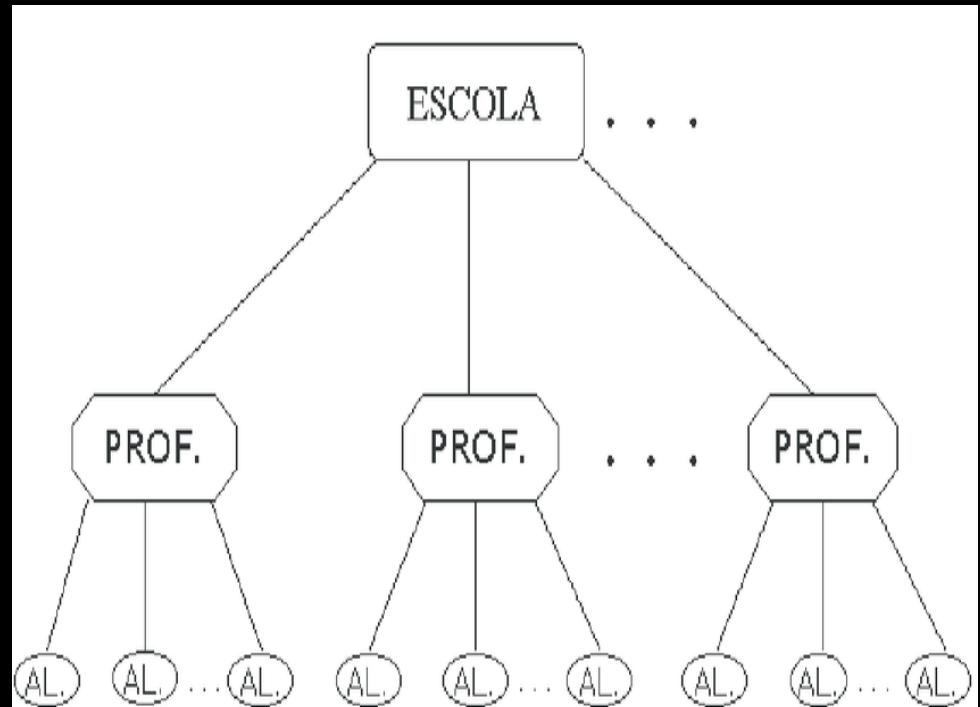
# CLASSIFICAÇÃO DE BANCOS DE DADOS

- As primeiras aplicações de banco de dados mantinham as informações de grandes empresas, como multinacionais, hospitais, universidades e bancos. Grande parte dos sistemas que utilizava os bancos de dados desta época, usava computadores de grande porte: os mainframes.
- Esses computadores eram muitos caros e possuíam somente uma interface para a linguagem de programação, e cada sistema desenvolvido levava muito tempo para ser programado, pois todas as transações realizadas no banco de dados eram programadas, testadas e depuradas.

# Hierárquicos

- Os dados são organizados em uma estrutura hierárquica semelhante a uma árvore, com registros pai e filhos. Cada registro pai pode ter vários registros filhos, mas cada registro filho só pode ter um pai.
- Caso fosse necessária a adição de uma nova informação (tabela ou campo), o banco de dados em sua totalidade precisaria ser reorganizado ou redefinido.
- Foi popular nas décadas de 1960 e 1970

# CLASSIFICAÇÃO DE BANCOS DE DADOS



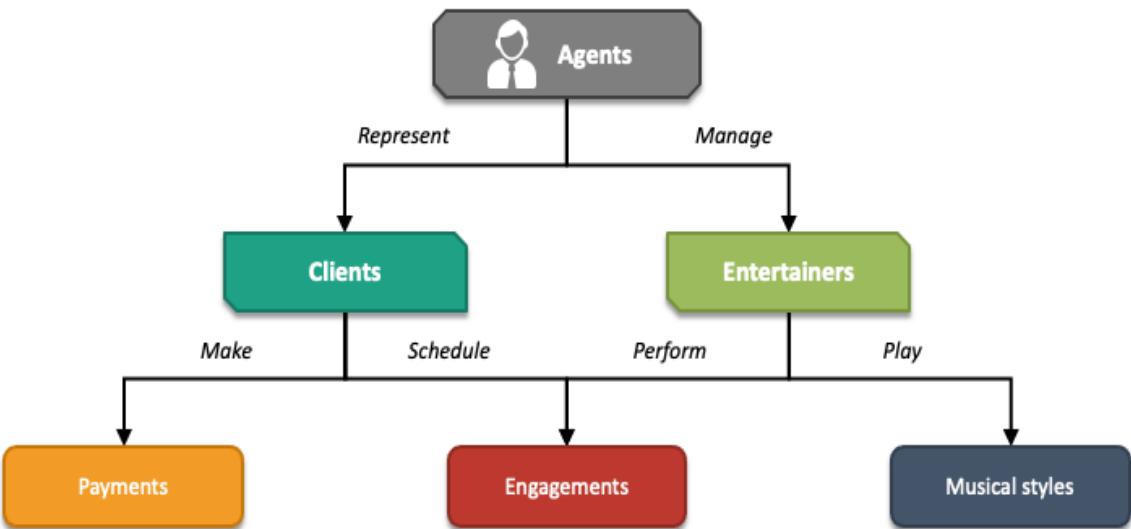
# REDE

- É uma extensão do modelo hierárquico, permitindo que um registro filho tenha mais de um pai.
- Isso é alcançado através do uso de estruturas de dados chamadas conjuntos de registros (sets).
- O modelo de rede foi desenvolvido para superar algumas limitações do modelo hierárquico, mas acabou sendo substituído pelo modelo relacional.

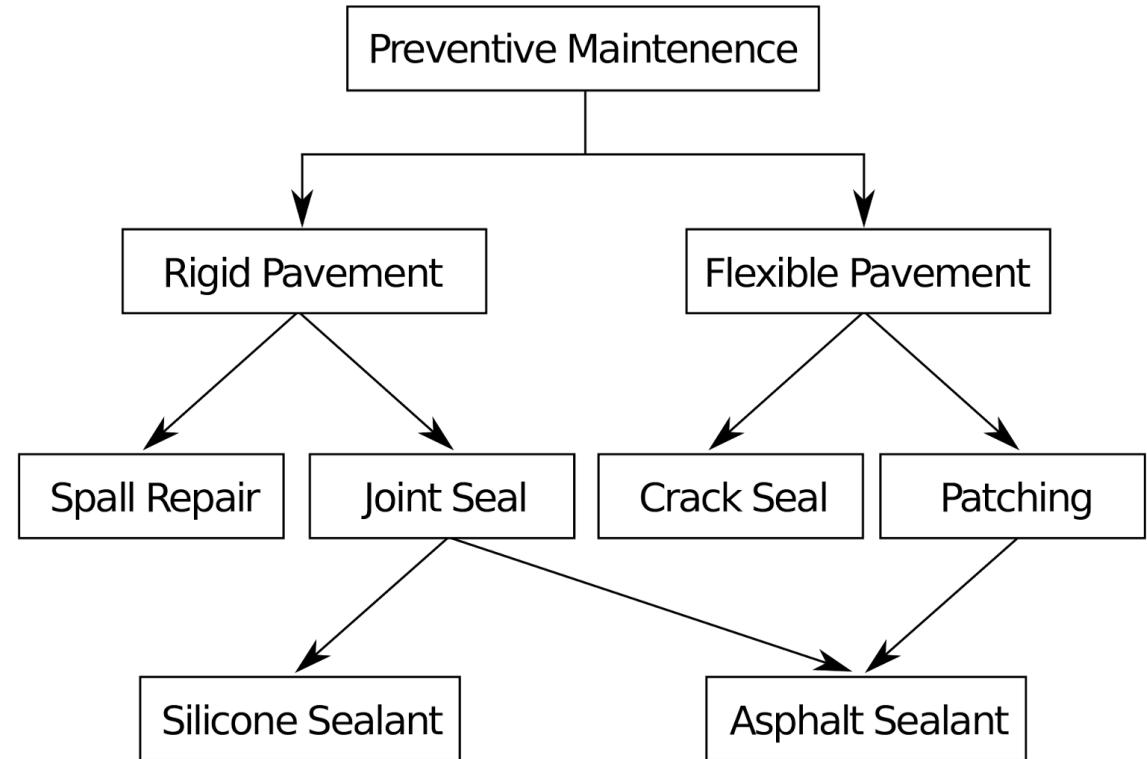
# CLASSIFICAÇÃO DE BANCOS DE DADOS

## NETWORK DATABASE MODEL

Enter your sub headline here



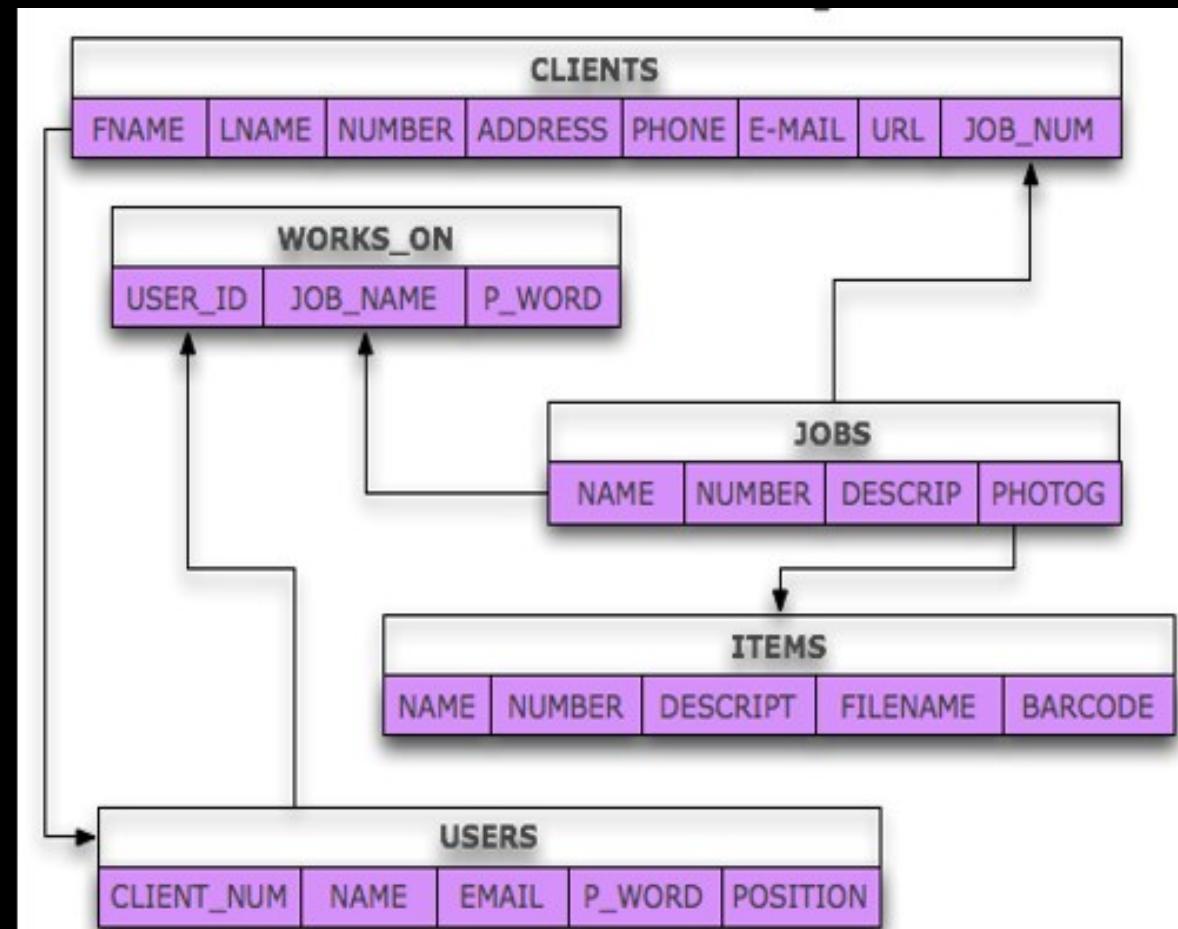
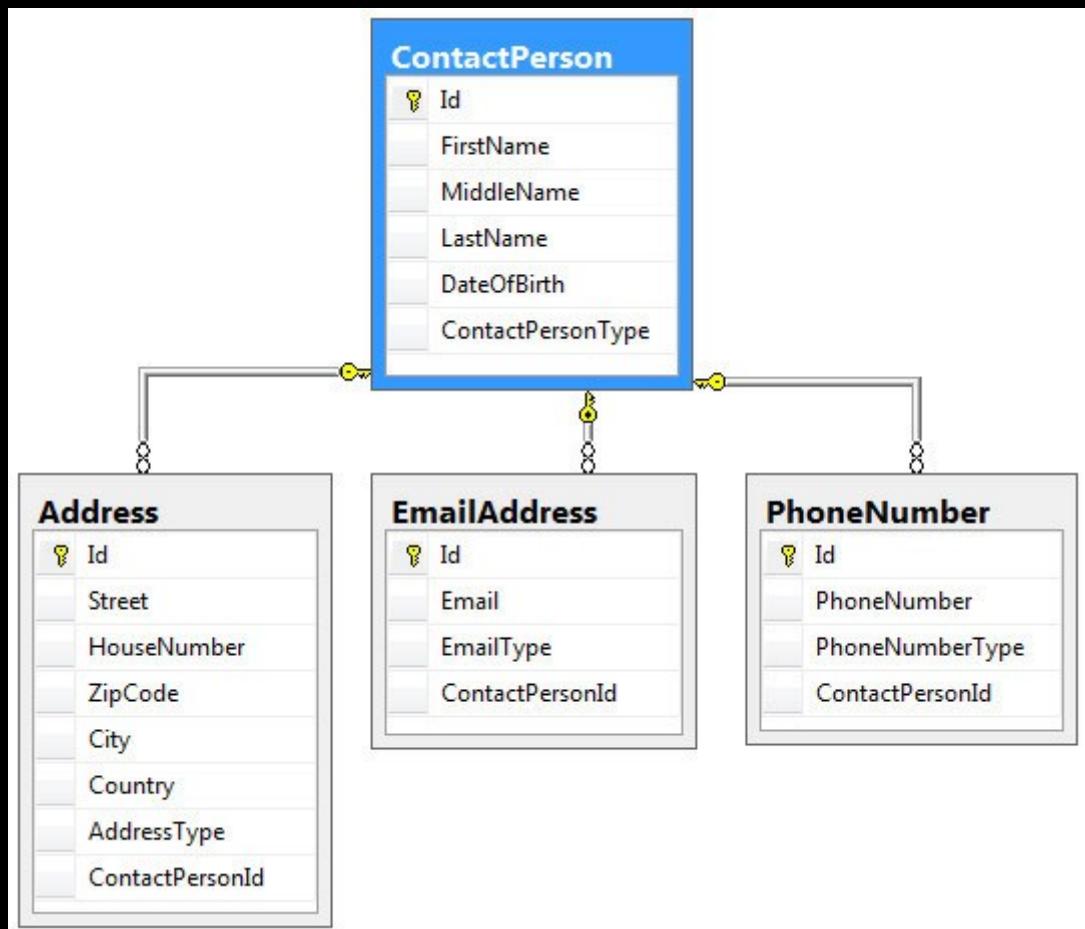
## Network Model



# RELACIONAL

- Os bancos de dados relacionais começaram a surgir comercialmente a partir de 1980.
- os dados são organizados em tabelas com linhas e colunas.
- Cada tabela representa uma entidade e cada linha representa uma instância dessa entidade.
- As relações entre as entidades são representadas por meio de chaves estrangeiras.

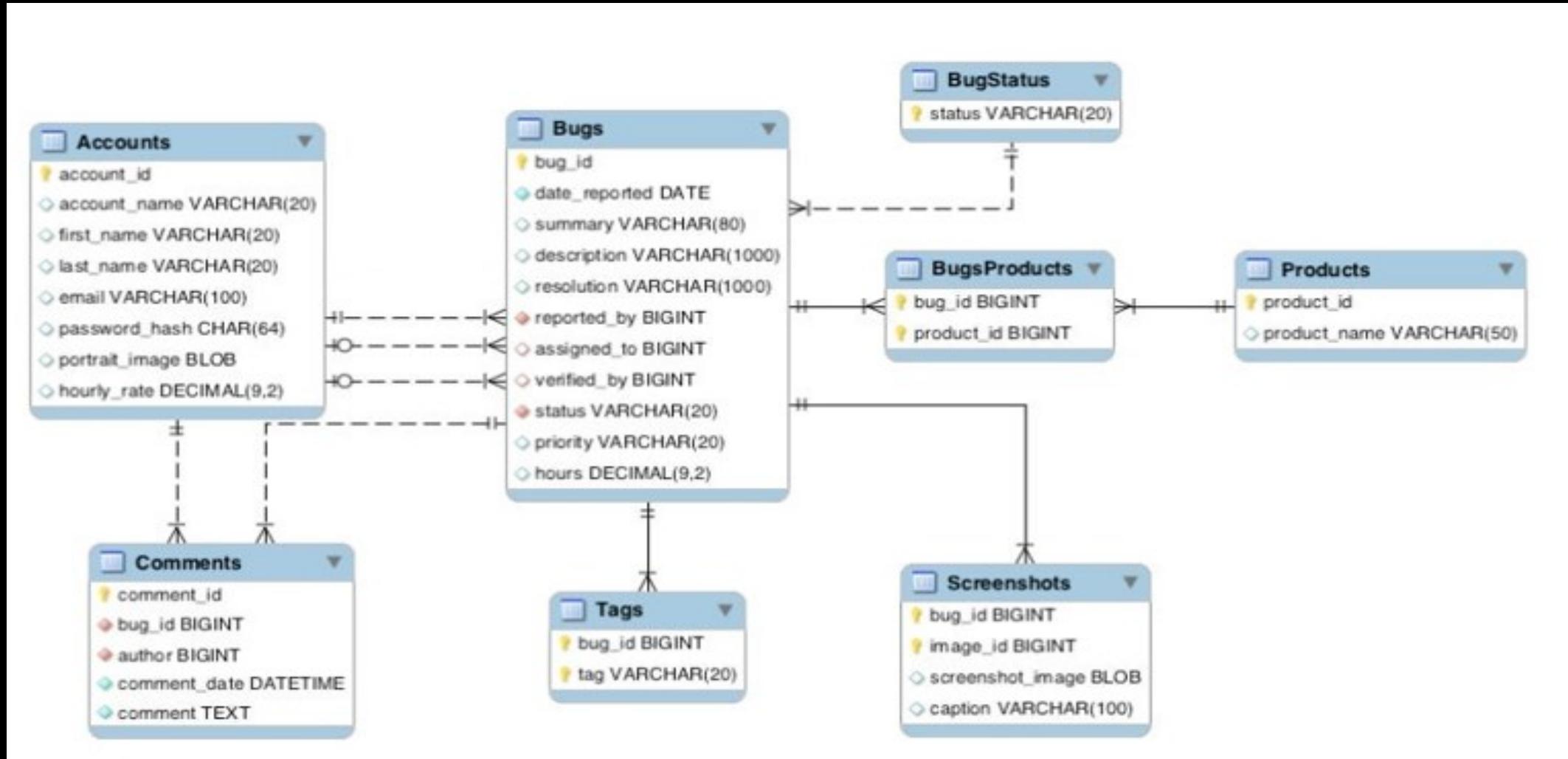
# RELACIONAL



# ORIENTADO A OBJETO

- Neste modelo, os dados são representados como objetos, que podem conter dados, bem como métodos para manipular esses dados.
- Este modelo é mais complexo do que o modelo relacional e é adequado para situações em que a estrutura dos dados é altamente complexa e variável, como em sistemas de engenharia, modelagem de simulação e desenvolvimento de software orientado a objetos

# ORIENTADO A OBJETO



# XML

- O XML (Extensible Markup Language) é uma linguagem de marcação que permite a representação de dados de forma estruturada e legível por máquina.
- Ele é frequentemente utilizado para a troca de dados entre sistemas diferentes devido à sua flexibilidade e capacidade de representar uma ampla variedade de tipos de dados.

# XML

- **Estrutura Hierárquica:** O XML organiza os dados em uma estrutura hierárquica de tags aninhadas, o que facilita a representação de dados complexos e relacionados de forma organizada. Essa estrutura hierárquica pode ser facilmente mapeada para a estrutura de dados de diferentes sistemas de banco de dados.
- **Independência de Plataforma:** O XML é independente de plataforma e linguagem de programação, o que significa que pode ser facilmente lido e processado por diferentes sistemas.

# XML

- **Flexibilidade:** O XML é altamente flexível e extensível, permitindo a definição de esquemas personalizados para representar tipos de dados específicos. Isso permite que diferentes sistemas definam suas próprias estruturas de dados e regras de validação, enquanto ainda podem trocar dados entre si usando XML.

# XML

- **Interoperabilidade:** Como o XML é amplamente suportado por uma variedade de tecnologias e ferramentas, ele facilita a interoperabilidade entre sistemas heterogêneos. Os sistemas podem trocar dados em formato XML e usar tecnologias como parsers XML para processar esses dados.
- **Padrões de Intercâmbio de Dados:** Existem muitos padrões e especificações que utilizam XML para facilitar a troca de dados entre sistemas, tais como SOAP (Simple Object Access Protocol) para serviços web, XML-RPC (XML Remote Procedure Call), e outros.

# XML

```
<book>
    <title> Harry Potter and the Philosopher's Stone </title>
    <author> J.K. Rowling </author>
    <year> 1997 </year>
</book>
```

# JSON

- O JSON (JavaScript Object Notation) é um formato de intercâmbio de dados leve, baseado em texto, que é amplamente utilizado para transmitir dados estruturados entre um servidor e um cliente, e entre diferentes sistemas.
- Ele foi inspirado na sintaxe de objetos literais do JavaScript, mas é independente de linguagem e pode ser facilmente interpretado e gerado por uma variedade de linguagens de programação

# JSON

- O JSON utiliza uma sintaxe simples e legível, composta por pares chave-valor, separados por dois pontos (:), e separados por vírgulas (,).
- Os valores podem ser strings, números, booleanos, arrays, objetos ou null.

```
{
```

```
"book": {
```

```
    "title": "Harry Potter and the Philosopher's Stone",
```

```
    "author": "J.K. Rowling",
```

```
    "year": 1997
```

```
}
```

```
}
```

# XML vs JSON

- No exemplo XML, cada campo (como título, autor e ano) é envolvido por tags de abertura e fechamento (<title>, </title>, <author>, </author>, <year>, </year>), o que aumenta a quantidade de caracteres necessários para representar os dados.
- Por outro lado, no exemplo JSON, os dados são representados de forma mais concisa usando uma estrutura de pares chave-valor dentro de objetos e arrays.

# XML Verbose

- O XML é **mais verbose** em comparação com o JSON, ou seja, o XML tende a exigir mais caracteres para representar a mesma informação em comparação com o JSON.
- Isso ocorre devido à natureza da sintaxe do XML, que requer tags de abertura e fechamento para cada elemento, enquanto o JSON usa uma estrutura de pares chave-valor mais compacta

# SGBD's não Freeware

- **Oracle Database:** Desenvolvido pela Oracle Corporation, o Oracle Database é um dos SGBDs mais poderosos e amplamente utilizados em empresas. Ele oferece uma ampla gama de recursos para gerenciamento de dados em grande escala, alta disponibilidade e segurança avançada. O Oracle Database é licenciado comercialmente e requer uma taxa de licenciamento para uso em ambientes de produção.

# SGBD's não Freeware

- **Microsoft SQL Server:** Desenvolvido pela Microsoft, o SQL Server é um SGBD relacional amplamente utilizado em ambientes corporativos. Ele oferece recursos avançados de gerenciamento, segurança e análise de dados. O SQL Server está disponível em várias edições, incluindo uma edição gratuita (SQL Server Express), mas as edições comerciais requerem uma licença paga.

# SGBD's não Freeware

- **IBM Db2:** O IBM Db2 é um SGBD relacional desenvolvido pela IBM, oferecendo suporte a ambientes de mainframe, servidor e nuvem. Ele fornece recursos avançados de gerenciamento de dados, escalabilidade e desempenho. O Db2 está disponível em várias edições, algumas das quais requerem uma licença paga.

# SGBD's não Freeware

- **SAP HANA:** Desenvolvido pela SAP, o SAP HANA é um SGBD em memória que oferece recursos avançados para análise em tempo real e processamento de dados de alta velocidade. É amplamente utilizado em empresas para suportar aplicativos empresariais críticos. O SAP HANA geralmente requer uma licença paga.

# SGBD's não Freeware

- **SAP HANA:** Desenvolvido pela SAP, o SAP HANA é um SGBD em memória que oferece recursos avançados para análise em tempo real e processamento de dados de alta velocidade. É amplamente utilizado em empresas para suportar aplicativos empresariais críticos. O SAP HANA geralmente requer uma licença paga.

# SGBD's Freeware

- **MySQL:** É um dos SGBDs mais populares do mundo, conhecido por sua confiabilidade, desempenho e facilidade de uso. O MySQL é distribuído sob a licença GNU GPL (General Public License) e está disponível gratuitamente para uso.
- **PostgreSQL:** É um poderoso SGBD relacional de código aberto que oferece recursos avançados, como suporte a transações ACID, integridade referencial e extensibilidade. O PostgreSQL é distribuído sob uma licença de software livre e está disponível gratuitamente para uso.

# SGBD's Freeware

- **SQLite:** É um SGBD embutido de código aberto que não requer um servidor separado, tornando-o ideal para aplicativos móveis e pequenos projetos. O SQLite é de domínio público e pode ser usado gratuitamente para qualquer propósito, comercial ou não comercial.
- **MariaDB:** É um fork do MySQL desenvolvido pela comunidade após a aquisição do MySQL pela Oracle Corporation. O MariaDB é distribuído sob a licença GPL e é uma alternativa de código aberto ao MySQL.

# SGBD's Freeware

- **Firebird:** É um SGBD relacional de código aberto que oferece suporte a transações ACID e é adequado para aplicativos embarcados, de desktop e de servidor. O Firebird é distribuído sob a licença de Interbase Public License (IPL) e pode ser usado gratuitamente.

# Principais Características dos SGBD's

- Permite inclusão, exclusão, seleção, ordenação e junção de registros de entidades.
- Possibilita a cópia e a exclusão de entidades.
- Estabelece relações entre as entidades e a criação de chaves.
- Permite a importação ou exportação de dados entre outras bases de dados.
- Possibilita a alteração da estrutura de campos e entidades.
- Permite consultas e relatórios da base de dados.
- Possibilita a criação de usuários com permissão de acesso individualizados.