Curso: ADS – Análise e Desenvolvimento de Sistema

Aluno: Murilo C. Ferreira

Ano atualização: 2024/2

Nome da Trilha: OT5 – Linguagem SQL: VIEW e Diagrama Entidade

Relacionamento.

**FICHAMENTO:**

Para o desenvolvimento deste fichamento , você deverá trazer além dos conceitos exemplos práticos de como utilizar o DER , WORKBENCH e VIEW.

**DER**

* **O que é?**

Para LucidChart (2024), o DER é uma representação gráfica da estrutura lógica de um banco de dados, descrevendo as entidades (tabelas), seus atributos (colunas) e os relacionamentos entre elas.

* **Objetos que compõem o DER;**

**Entidades**: Representam tabelas no banco de dados.

**Atributos**: São os campos ou colunas das tabelas.

**Relacionamentos**: Mostram como as entidades se conectam entre si (1:1, 1:N, N:M).

**Cardinalidade**: Define o número de ocorrências em um relacionamento.

**Chaves** **Primárias** (PK): Identificam unicamente cada registro de uma tabela.

**Chaves Estrangeiras** (FK): Estabelecem relacionamentos entre tabelas.

* **Passos para construção do DER:**

1. **Identificar as entidades**: Quais são os elementos principais do sistema?
2. **Definir os atributos**: Quais informações cada entidade deve armazenar?
3. **Estabelecer relacionamentos**: Como as entidades estão conectadas?
4. **Definir cardinalidade**: Determinar as regras de associação entre as entidades.
5. **Validar e revisar**: Certificar-se de que o DER reflete corretamente o modelo lógico.

**WORKBENCH**

* **Como criar um DER no Workbench;**

LucidChart (2024) ensina que, para criar um modelo de dados no MySQL Workbench, é necessário iniciar abrindo o software e selecionando a opção de criar um novo modelo de dados, acessível pelo menu File e, em seguida, a opção New Model. Após essa etapa, deve-se acessar a aba EER Diagram, que permite começar a construção do diagrama. Nesse ambiente, utiliza-se as ferramentas disponíveis para adicionar entidades (tabelas), seus atributos e os relacionamentos entre as entidades. Em seguida, é importante configurar os tipos de dados e as restrições para cada atributo, garantindo a integridade e consistência do modelo. Por fim, é possível gerar o script SQL correspondente, que implementará a estrutura do banco de dados no ambiente relacional.

* **Principais tipos de dados (MySQL).**

Segundo os autores, o MySQL pode comportar dados Numéricos (INT, BIGINT, DECIMAL, FLOAT), Texto (VARCHAR,CHAR,TEXT), Datas (DATE, DATETIME, TIMESTAMP) ou Booleanos (BOOLEAN ou TINYINT).

**VIEW**

* **O que é *View* em banco de dados?**

Os autores definem que uma *View* é uma tabela virtual baseada em uma consulta SQL. Ela não armazena dados diretamente, mas exibe os dados obtidos pela consulta subjacente.

* **Para que serve;**

Segundo LucidChart(2024), facilitar consultas complexas, encapsulando lógica SQL em uma estrutura reutilizável, restringir o acesso aos dados sensíveis, exibindo apenas as informações necessárias; e fornecer abstração para simplificar a interação com o banco de dados.

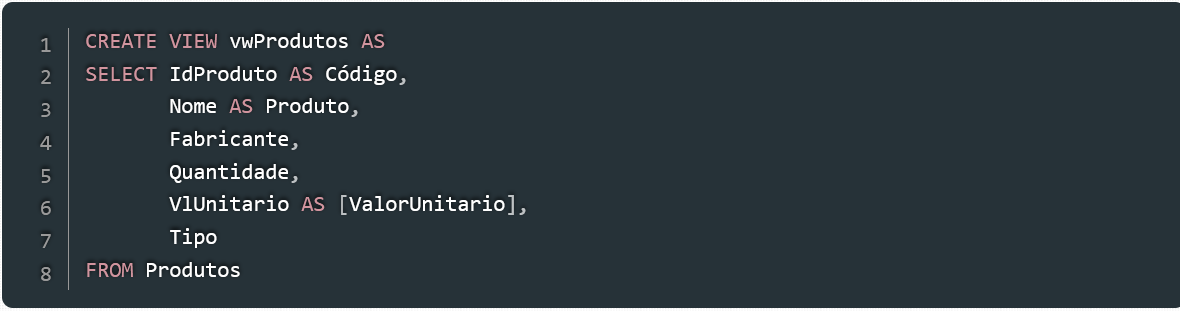
* **Como funciona**

A view é criada com o comando CREATE VIEW, especificando a consulta SQL que define seus dados. Quando a view é usada, o banco executa a consulta associada e retorna os dados em tempo real.

* **Qual a segurança em banco de dados que a view nos proporciona;**

As views em banco de dados oferecem importantes mecanismos de segurança e gerenciamento de informações. Elas possibilitam o controle de acesso, permitindo que usuários visualizem apenas colunas ou linhas específicas, sem acesso direto às tabelas subjacentes, o que protege dados sensíveis. Além disso, as *views* podem ser configuradas como read-only, garantindo proteção contra alterações indesejadas nos dados originais. Outro benefício relevante é a capacidade de ocultar a complexidade do banco de dados, abstraindo colunas sensíveis ou lógica de negócios elaborada, o que simplifica a interação dos usuários com os dados e reforça a segurança da aplicação

**Exemplo de criação e uso de View:**



*Fonte: DevMedia (2024)*

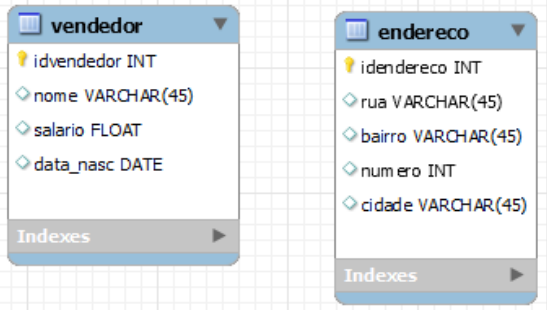
**Referências**

LUCIDCHART. **O que é um diagrama entidade relacionamento?** 2024. Disponível em: https://www.lucidchart.com/pages/pt/o-que-e-diagrama-entidade-relacionamento. Acesso em: 21 nov. 2024.

DEVMEDIA. **Conceitos e criação de views no SQL Server**. 2024. Disponível em: https://www.devmedia.com.br/conceitos-e-criacao-de-views-no-sql-server/22390. Acesso em: 21 nov. 2024.

**Atividade**

1. **Observe as tabelas abaixo com a seguinte situação e apresente a solução na validação:**

****

**A. Neste momento você deverá implementar a tabela vendedor e endereço analisando a seguinte situação.**

**B. Essas duas tabelas se relacionam:**

**Um vendedor possui apenas um endereço e um endereço pertence somente a um vendedor.**

**E agora? Onde a chave estrangeira acontece?**

**Na tabela vendedor ou na tabela endereço?**

Na tabela vendedor, a chave estrangeira é idendereco, e na tabela endereço é idvendedor.

**Faça o relacionamento de acordo com o que acha ideal, e discutiremos na validação.**

**Análise: a nossa solução está focada em um pequeno software, com poucos registros. Mas na expansão do mercado, que modificações poderiam ser implementadas para melhores resultados, segurança, consistência dos dados? Pense e traga para discussão na validação, assim como outras modificações, ideias que tiver.**

CONTINUANDO…

Agora que criou o relacionamento podemos desenvolver a segunda etapa desta Ot que é a implementação das Views com as seguintes questões:

1. Existem usuários que utilizam o sistema do supermercado, que não podem ter acesso

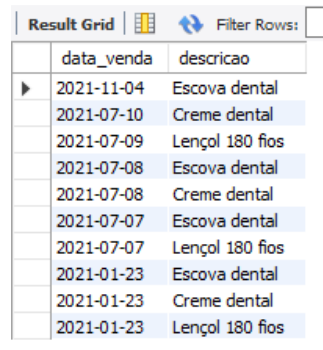
ao salário dos vendedores, como você resolveria essa situação?

Eu criaria uma terceira tabela com os dados do usuário e o nível de permissões e filtraria os dados com base nela.

2. Crie uma View que apresente a data da vendas e a descrição dos produtos que foram

vendidos por dia e ordene pela data mais recente.

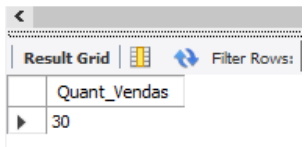
Resultado:



SELECT distinct v.data\_venda, p.descricao from vendas v JOIN vendas\_has\_produto vp ON v.idvenda=vp.idvenda JOIN produto p ON p.idproduto=vp.idproduto order by data\_venda desc;

3. Crie uma View que apresenta a quantidade de vendas que foram realizadas.

Resultado:



select SUM(quantidade) as Quant\_Vendas from vendas\_has\_produto;

4. Imagine alguma outra situação onde seja interessante criar uma view, dentro da

proposta em que estamos trabalhando e faça-a.

Gerar um relatório que exiba os seguintes dados: nome do produto, categoria do produto, quantidade vendida, data da venda.

select p.descricao, c.descricao as Categoria, vp.quantidade,v.data\_venda from produto p JOIN categoria c on p.idcategoria=c.idcategoria JOIN vendas\_has\_produto vp ON vp.idproduto=p.idproduto JOIN vendas v on v.idvenda=vp.idvenda;