

Centro Universitário Maurício de Nassau

Database Application – Prof.: Filipe Nascimento

Código Da Turma: GRA0790103DNA

Curso: Ciência da Computação

Aluno: Túlio Mendes – 01633581

Modelagem Dimensional

1. a) Casos de uso do star schema e do snowflake;

Os esquemas em estrela (star schema) e floco de neve (snowflake schema) são dois tipos comuns de modelagem de dados usados em data warehouses e sistemas de Business Intelligence (BI). Cada um tem seus próprios casos de uso, vantagens e desvantagens. Abaixo estão descrições detalhadas de cada um, incluindo os casos de uso mais adequados para cada esquema.

➤ Star Schema

- **Descrição:**

O star schema é uma abordagem de modelagem de dados onde uma tabela de fatos central está diretamente conectada a várias tabelas dimensionais. As tabelas dimensionais são altamente desnormalizadas, ou seja, contém dados redundantes para facilitar a consulta.

- **Tabela de Fatos:** Contém os dados quantitativos que representam os eventos de negócio, como vendas, transações, etc.;
- **Tabelas Dimensionais:** Contêm os atributos descritivos relacionados aos dados da tabela de fatos, como tempo, produto, cliente, etc

- **Casos de Uso:**

1. **Consultas Simples e Rápidas:** Adequado para ambientes onde a rapidez das consultas é crucial, como dashboards de BI, relatórios ad-hoc e ferramentas de visualização de dados. Devido à desnormalização, as consultas podem ser executadas rapidamente, com menos junções entre tabelas.
2. **Usuários Finais Não Técnicos:** Ideal para ambientes onde os usuários finais não são técnicos e precisam de um esquema de dados fácil de entender e navegar. A estrutura simples do star schema facilita a criação de consultas sem exigir conhecimento profundo de SQL.
3. **Análises OLAP (Online Analytical Processing):** Muito usado em sistemas OLAP, onde a agregação rápida de grandes volumes de dados é necessária para análises multidimensionais.
4. **Implementações de Data Mart:** Comumente utilizado em data marts, que são subconjuntos específicos de um data warehouse, focados em uma área de negócio particular, como vendas ou finanças.

Vantagens:

- ✓ Simplicidade e clareza na modelagem.
- ✓ Desempenho rápido em consultas devido à redução de junções.
- ✓ Facilidade de uso para usuários finais.

Desvantagens:

- ❖ Pode levar à redundância de dados.
- ❖ Menos eficiente em termos de armazenamento devido à desnormalização.

➤ **Snowflake Schema**

- **Descrição:**

O snowflake schema é uma variação do star schema onde as tabelas dimensionais são normalizadas em várias tabelas relacionadas. Isso significa que os dados são organizados de forma a minimizar a redundância, seguindo as regras de normalização.

- **Tabela de Fatos:** Semelhante ao star schema, contém os dados quantitativos dos eventos de negócio;
- **Tabelas Dimensionais:** São normalizadas em várias tabelas relacionadas, reduzindo a redundância e otimizando o armazenamento.

- **Casos de Uso**

1. **Minimização de Redundância:** Adequado para situações onde a minimização da redundância de dados é importante, como em grande data warehouses com muitas dimensões complexas;
2. **Armazenamento Eficiente:** Ideal quando o armazenamento é uma preocupação significativa, pois a normalização reduz o espaço necessário para armazenar os dados;
3. **Ambientes com Mudanças Frequentes nos Dados:** Útil em cenários onde as dimensões mudam frequentemente e a atualização de dados precisa ser feita de forma eficiente, sem a necessidade de replicar dados redundantes;
4. **Complexidade de Consultas:** Adequado para ambientes onde as consultas são mais complexas e os usuários são tecnicamente capacitados para escrever junções complexas.

Vantagens:

- ✓ Redução da redundância de dados.
- ✓ Menor espaço de armazenamento necessário.
- ✓ Melhor manutenção e atualização de dados dimensionais.

Desvantagens:

- ❖ Consultas mais complexas devido ao maior número de junções.
- ❖ Pode ser mais difícil para usuários finais não técnicos compreenderem a estrutura dos dados.

Resumo das Diferenças e Escolhas:

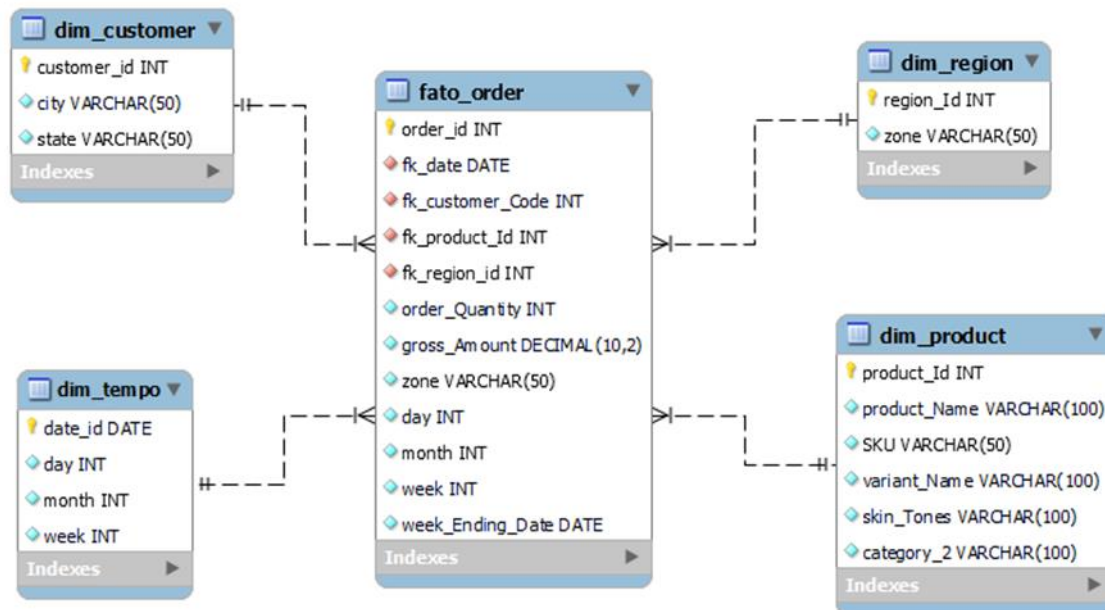
➤ **Star Schema:**

- **Quando usar:** Quando a rapidez das consultas e a simplicidade são mais importantes do que a minimização da redundância.
- **Ambientes típicos:** Dashboards, relatórios rápidos, usuários finais não técnicos.

➤ **Snowflake Schema:**

- **Quando usar:** Quando a minimização da redundância e a eficiência de armazenamento são mais importantes do que a simplicidade das consultas.
- **Ambientes típicos:** Grande data warehouses, ambientes de BI complexos, usuários técnicos.

b) implementar um exemplo de modelo star schema;



Este é um exemplo básico de um modelo de esquema estelar em MySQL, composto por uma tabela de fatos (Fato_Order) representando pedidos, e quatro tabelas de dimensão (Dim_Customer, Dim_Product, Dim_Date e Dim_Region) fornecendo contexto detalhado. A tabela de fatos inclui campos como data, quantidade de pedido e montante bruto, conectados às dimensões por chaves estrangeiras. As tabelas de dimensão contêm informações sobre clientes, produtos, datas e regiões. Relacionamentos entre a tabela de fatos e as dimensões são estabelecidos para análises eficientes e flexíveis. O esquema permite análises detalhadas de pedidos, incluindo agregações temporais, por cliente, produto e região, facilitando insights valiosos sobre o negócio.

O esquema estelar tem como finalidade principal facilitar análises de dados complexas e eficientes, especialmente em ambientes de data warehousing e business intelligence. Ele é projetado para fornecer uma estrutura otimizada para armazenar grandes volumes de dados transacionais e históricos, permitindo análises detalhadas e agregações em várias dimensões. Com este modelo, é possível realizar análises temporais, segmentações de clientes, análises de produtos e geográficas de forma rápida e eficiente. Isso ajuda as organizações a tomarem decisões informadas e estratégicas com base nos dados disponíveis, impulsionando o desempenho e a competitividade do negócio.

2. a) Cubos OLAP – Definições, Arquitetura e Implementação.

Definições:

Cubos OLAP (Online Analytical Processing) são estruturas de dados multidimensionais usadas para análise de dados complexos em sistemas de Business Intelligence (BI). Eles permitem consultas rápidas e eficientes para análises multidimensionais, facilitando a agregação de dados e o cálculo de métricas em diferentes dimensões (ex.: tempo, localização, produto).

Arquitetura:

- I. **Dimensões:** As dimensões representam as categorias pelos quais os dados serão analisados, como tempo, produto, cliente e localização geográfica. Cada dimensão pode ter várias hierarquias e níveis.
- II. **Tabela de Fatos:** A tabela de fatos contém as métricas ou medidas que serão analisadas, como vendas, lucro, quantidade de produtos vendidos, etc. Cada linha da tabela de fatos contém chaves estrangeiras que se relacionam com as dimensões.
- III. **Cubo:** O cubo OLAP é uma estrutura multidimensional que combina todas as dimensões com a tabela de fatos. Cada célula do cubo contém os valores das métricas para uma combinação específica das dimensões.

Implementação:

- I. **Modelagem de Dados:** O processo de implementação de um cubo OLAP começa com a modelagem de dados, onde identificamos as dimensões relevantes e as métricas que serão analisadas. Em seguida, projetamos a estrutura da tabela de fatos e das tabelas de dimensão.
- II. **ETL (Extract, Transform, Load):** Os dados são extraídos das fontes de dados, transformados para se adequarem ao modelo de dados do cubo OLAP e carregados na tabela de fatos e nas tabelas de dimensão.
- III. **Construção do Cubo:** Uma vez que os dados estão carregados nas tabelas de fatos e dimensão, o cubo OLAP é construído usando ferramentas OLAP. Isso envolve a agregação dos dados em diferentes níveis de granularidade e a pré-cálculo de agregações para consultas rápidas.
- IV. **Consulta e Análise:** Os usuários podem então acessar o cubo OLAP usando ferramentas de análise OLAP e realizar consultas multidimensionais para obter insights sobre os dados. Eles podem explorar os dados em diferentes perspectivas, realizar drill-downs, drill-ups e drill-across para analisar os dados em detalhes.

b) Exemplo de cubo implementado (mysql);

1. Criação das Tabelas:

- Começamos criando as tabelas necessárias para o nosso esquema estrela. Isso inclui as tabelas de fatos (fato_order) e dimensões (dim_customer, dim_product, dim_date e dim_region). As tabelas de dimensão armazenam informações contextuais sobre os dados factuais, enquanto a tabela de fatos contém as métricas ou medidas que queremos analisar;
- Cada tabela é projetada com as colunas apropriadas para armazenar os dados relevantes, como datas, códigos de cliente, nomes de produtos, quantidades de pedidos, valores brutos, etc.;
- As chaves estrangeiras são definidas nas tabelas de fatos para estabelecer relacionamentos com as tabelas de dimensão.

```
-- Tabela de Fatos (Fato_Order)
CREATE TABLE fato_order (
    order_id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    date DATE,
    customer_Code INT,
    product_Id INT,
    order_Quantity INT,
    gross_Amount DECIMAL(10, 2),
    day INT,
    month INT,
    week INT,
    week_Ending_Date DATE,
    zone VARCHAR(50),
    FOREIGN KEY (Customer_Code) REFERENCES dim_customer(Customer_Code),
    FOREIGN KEY (Product_Id) REFERENCES dim_product(Product_Id)
);
```

2. Populando as Tabelas:

- Após a criação das tabelas, inserimos dados fictícios nelas para simular um ambiente de produção. Isso é feito por meio de comandos INSERT INTO, onde inserimos linhas de dados nas tabelas;
- Por exemplo, na tabela dim_customer, podemos inserir informações sobre diferentes clientes, como seus códigos, cidades e estados. Da mesma forma, na tabela fato_order, podemos inserir detalhes de pedidos, como datas, códigos de clientes, IDs de produtos, quantidades e valores brutos.

```
-- Populando Dim_Customer
INSERT INTO dim_customer (customer_code, city, state)
VALUES
    (1, 'New York', 'NY'),
    (2, 'Los Angeles', 'CA'),
    (3, 'Chicago', 'IL');
```

3. Criação da View do Cubo OLAP:

- Em seguida, criamos uma view que representa o cubo OLAP. Essa view, chamada de Cubo_OLAP, combina os dados das tabelas de dimensão e da tabela de fatos para fornecer uma visão multidimensional dos dados;
- Na view, selecionamos colunas relevantes de cada tabela e realizamos junções (JOIN) para combinar os dados de acordo com as chaves estrangeiras. Isso nos permite analisar os dados de vendas de várias perspectivas, como datas, clientes, produtos e regiões.

```
-- Criação da Tabela Virtual (View) representando o cubo OLAP;
-- Para obter o valor do produto e o cliente que o comprou.
• CREATE OR REPLACE VIEW Valor_Produto_Cliente AS
SELECT
    f.customer_Code,      -- Código do cliente
    c.city,               -- Cidade do cliente
    c.state,              -- Estado do cliente
    p.product_Name,       -- Nome do produto
    f.gross_Amount AS Valor_do_Produto -- Valor bruto do produto
FROM
    fato_order f
JOIN
    dim_customer c ON f.customer_Code = c.customer_code -- Junção com a dimensão do cliente
JOIN
    dim_product p ON f.product_Id = p.product_Id;      -- Junção com a dimensão do produto
```





4. Consulta na View do Cubo OLAP:

- Por fim, realizamos consultas na view `Cubo_OLAP` para extrair informações agregadas dos dados de vendas;
- Essas consultas podem incluir funções de agregação, como `SUM`, para calcular totais ou métricas agregadas. Podemos realizar consultas para analisar as vendas ao longo do tempo, por região, por cliente, por produto, entre outros aspectos;
- As consultas na view do cubo OLAP nos permitem obter insights valiosos sobre o desempenho das vendas e ajudam na tomada de decisões estratégicas no negócio.

Consulta:

```
-- Consulta para obter informações sobre quem comprou qual produto e quanto pagou por ele
• SELECT *
FROM Valor_Produto_Cliente
LIMIT 10; -- Limitando o resultado a 10 linhas para manter a saída concisa
```

Obtemos o resultado:

Result Grid			Filter Rows:	<input type="text"/>	Export:		Wrap Cell Content:	
	customer_Code	city	state	product_Name	Valor_do_Produto			
▶	1	New York	NY	Product A	100.00			
	2	Los Angeles	CA	Product B	50.00			
	3	Chicago	IL	Product C	80.00			