**Desafio - Processamento de Dados Simplificado com Power BI**

Em um projeto desafiador, embarcamos na tarefa de importar dados de um banco de dados MySQL hospedado na plataforma Azure para o Power BI. Nosso objetivo principal foi efetuar uma série de etapas cruciais de transformação de dados, resultando na criação de um conjunto de dados pronto para análise. Ao longo dessa jornada, adotamos um conjunto de ações estratégicas para atingir esse objetivo.

Primeiramente, estabelecemos uma conexão sólida entre o banco de dados MySQL na Azure e a ferramenta de visualização de dados Power BI. Esse passo crítico permitiu a transferência segura e eficaz de dados do ambiente de armazenamento na nuvem para a plataforma de análise.

A partir desse ponto, iniciamos as operações de transformação de dados. Isso envolveu a limpeza de informações irrelevantes ou inconsistentes, bem como a união de tabelas, quando necessário, para criar uma representação coesa e coerente dos dados. Além disso, aplicamos filtros, agregações e cálculos para aprimorar a qualidade e a utilidade das informações.

Com os dados agora refinados e alinhados às necessidades de análise, procedemos à criação de relatórios e painéis no Power BI. Esses elementos visuais foram projetados para oferecer insights claros e acionáveis a partir do conjunto de dados tratado. Utilizamos recursos de visualização avançada para destacar tendências, identificar padrões e comunicar de forma eficaz as descobertas aos stakeholders.

Em resumo, este projeto representou uma jornada desafiadora e gratificante de importação, transformação e análise de dados. Ao superar as complexidades inerentes a esse processo, conseguimos disponibilizar um conjunto de dados pronto para uso, capacitando a tomada de decisões informadas e eficazes com base em informações confiáveis e relevantes.

Parte superior do formulário

**1. Criação de uma Instância na Azure para MySQL**

1. Primeiro, criamos uma instância na plataforma Azure para hospedar nosso banco de dados MySQL. Isso envolveu a configuração de um ambiente seguro e escalável para os dados.

**2. Criação do Banco de Dados**

1. Em seguida, criamos um banco de dados com base em um arquivo disponibilizado no GitHub. Essa etapa envolveu a criação de tabelas e a inserção inicial de dados.

Passo 3: Integração do Power BI com MySQL na Azure

1. Configuramos uma conexão direta entre o Power BI Desktop e o banco de dados MySQL hospedado na Azure. Isso nos permitiu importar os dados diretamente no Power BI para análise.

**4. Verificação de Problemas na Base de Dados**

1. Após a importação dos dados, realizamos uma análise detalhada para identificar problemas que precisavam ser abordados antes da análise. Aqui estão algumas das diretrizes seguidas:

- \*\*Verificação dos Cabeçalhos e Tipos de Dados:\*\*

- Examinamos os cabeçalhos das tabelas e os tipos de dados para garantir que estavam corretos. Isso incluiu a correção de erros de digitação e a definição de tipos de dados apropriados.

- \*\*Modificação dos Valores Monetários:\*\*

- Para aumentar a precisão, modificamos os valores monetários para o tipo "decimal fixo".

- \*\*Verificação de Nulos e Remoção:\*\*

- Identificamos valores nulos em uma coluna e decidimos como tratá-lo. Foi decidido deixá-lo como estava, dado que se trata de somente um indivíduo que pelos dados conclui-se que é a figura de mais alto escalão da empresa.

- \*\*Identificação de Funcionários Sem Gerente:\*\*

- Utilizamos o Power Query no Power BI para criar uma nova coluna personalizada chamada "FuncionarioSemGerente". Aqui estão as etapas realizadas:

1. Clique na guia "Adicionar Coluna" (Add Column) na parte superior.

2. Selecione "Coluna Personalizada" (Custom Column) no menu suspenso.

3. Na caixa de diálogo "Coluna Personalizada", demos um nome à nova coluna ("FuncionarioSemGerente").

4. Na fórmula de coluna personalizada, usamos a função if para verificar se a coluna "Super\_ssn" é nula e atribuímos "Sim" ou "Não" com base nessa verificação.

**5. Verificar se há algum departamento sem gerente**

Nesta etapa, verificamos se há departamentos na tabela "azure\_company departament" que não possuem um gerente atribuído. A análise foi realizada visualmente, examinando os dados da coluna "azure\_company.employee(Mgr\_ssn)".

\*\*Resultado:\*\*

Após uma verificação visual dos dados na coluna "azure\_company.employee(Mgr\_ssn)", não foi encontrado nenhum valor nulo. Isso indica que todos os departamentos possuem um gerente atribuído na tabela "azure\_company departament".

Portanto, não há departamentos sem gerentes na tabela de departamentos.

\*Observação:\* Esta análise foi feita visualmente, examinando os dados disponíveis na tabela "azure\_company departament".

**6. Preenchimento de Departamentos Sem Gerentes**

Nesta etapa, verificamos se existem departamentos sem gerentes na base de dados. A análise revelou que não havia departamentos sem gerentes, portanto, não foi necessário preencher lacunas ou tomar medidas adicionais.

Esta etapa foi concluída com sucesso, uma vez que não havia departamentos sem gerentes na base de dados.

**7. Verificação do Número de Horas dos Projetos**

Nesta etapa, verificamos o número de horas atribuídas a cada projeto na tabela "azure\_company works\_on". Para isso, realizamos as seguintes etapas:

1. Acessamos a tabela "azure\_company works\_on", que contém as informações sobre as horas trabalhadas em projetos.

2. Utilizamos uma função de agregação, como "Somar" (Sum), para calcular o número total de horas por projeto.

3. Criamos uma medida calculada no Power BI para mostrar esses totais de horas.

\*\*Resultado:\*\*

Concluímos a verificação do número de horas atribuídas a cada projeto na tabela "azure\_company works\_on". Isso nos fornece informações valiosas sobre o esforço dedicado a cada projeto.

**8. Separar Colunas Complexas**

O desafio foi separar colunas complexas em algumas tabelas do banco de dados e torná-las mais acessíveis para análise no Power BI. Aqui estão os detalhes dessa etapa:

1. Na tabela "azure\_company employee", algumas colunas, como "azure\_company.departament(Ssn)", continham valores do tipo "table" em todas as linhas. No entanto, devido à natureza dos dados e à falta de uma estrutura consistente, não foi possível realizar a expansão dessas colunas. Portanto, elas permaneceram não separadas.

2. Da mesma forma, na tabela "azure\_company project", a expansão das colunas complexas resultou em valores nulos, e não foi possível criar uma estrutura útil a partir delas. Essas colunas também permaneceram não separadas.

3. As outras tabelas, porém, resultaram em colunas complexas que puderam ser separadas e, assim sendo, gerou melhor panorama para análise dos dados dispostos.

Este passo ressalta a importância de avaliar a estrutura dos dados e decidir se a separação de colunas complexas é realmente necessária com base na consistência dos dados e nos critérios de relevância.

**9. Mesclar Consultas Employee e Departament**

Neste passo, realizamos a mesclagem das consultas "employee" e "departament" para criar uma nova tabela chamada "Employee Department" que inclui o nome dos departamentos associados aos colaboradores. Esta etapa é fundamental para enriquecer nossos dados e permitir análises mais detalhadas.

Aqui está um resumo do que foi feito:

1. Utilizamos o Power Query Editor no Power BI Desktop para realizar a mesclagem das consultas.

2. Configuramos a mesclagem com base na tabela "employee," pois queremos incluir informações de departamento para cada colaborador.

3. Fizemos a mesclagem usando o relacionamento entre as tabelas "employee" (usando a coluna "Ssn") e "departament" (usando a coluna "Mgr\_ssn").

A nova tabela "Employee Department Association" agora inclui todos os campos da tabela "employee" e o campo adicional "DepartmentName," que contém o nome do departamento associado a cada colaborador.

**10. Remoção de Colunas Desnecessárias**

Nesta etapa do projeto, focamos na otimização da nossa estrutura de dados para torná-la mais simples e direcionada para a proposta da análise no Power BI. Embora tenhamos extraído informações valiosas ao mesclar as tabelas "employee" e "departament," é importante garantir que nossa tabela final esteja enxuta e contenha apenas as informações realmente relevantes para nosso objetivo.

Aqui estão os destaques deste passo:

- \*\*Remoção de Colunas Não Relevantes:\*\* Identificamos e eliminamos colunas que não são diretamente relevantes para nossa análise. Isso nos ajuda a manter nossa tabela mais limpa e a focar nos dados essenciais.

- Ao remover as colunas não relevantes, garantimos que nosso conjunto de dados seja mais fácil de usar e que as visualizações e análises subsequentes sejam mais precisas e diretas. É uma prática recomendada para manter o foco em nossos objetivos analíticos.

**11. Realize a junção dos colaboradores e respectivos nomes dos gerentes**

- Primeiro foi realizada uma consulta SQL por meio do shell da Azure, cujo comando foi:

(SELECT e1.Super\_ssn, CONCAT(e1.Fname, ' ', COALESCE(e1.Minit, ''), ' ', e1.Lname) AS EmployeeName,

CONCAT(e2.Fname, ' ', COALESCE(e2.Minit, ''), ' ', e2.Lname) AS ManagerName

FROM azure\_company.employee AS e1

LEFT JOIN azure\_company.employee AS e2 ON e1.Super\_ssn = e2.Ssn

WHERE e1.Super\_ssn IS NOT NULL

AND e1.Super\_ssn <> ''

ORDER BY e1.Super\_ssn, EmployeeName;)

- Após a obtenção da consulta importada, foi feito no PowerQuery uma Nova Coluna separando gerentes de Colaboradores por meio da lógica:

= if [EmployeeName] = "John B Smith" or [EmployeeName] = "Franklin T Wong" or [EmployeeName] = "Ahmad V Jabbar" then "Manager" else "Collaborator"

**12. Mescle as colunas de Nome e Sobrenome para ter apenas uma coluna definindo os nomes dos colaboradores**

1. As colunas Fname, Minit e Lname foram unificadas (mescladas) e se transformaram na coluna Name.

**13. Mesclagem de Tabelas para Criar a Tabela Department By Local**

-- Neste passo, foi realizada a mesclagem das tabelas "azure\_company departament" e "azure\_company dept\_locations" com o objetivo de criar uma nova tabela chamada "Department By Local". Essa mesclagem teve como finalidade criar combinações únicas entre departamentos e localizações, o que será útil na criação de um modelo estrela em um módulo futuro.

-- Aqui estão os passos resumidos:

\*\*Mesclagem de Tabelas\*\*:

1. Utilizando uma ferramenta ou linguagem adequada (como SQL, Power Query, ou outra), você mesclou as tabelas "azure\_company departament" e "azure\_company dept\_locations" com base em algum campo comum, como um identificador único de departamento ou localização. Essa mesclagem resultou em uma nova tabela chamada "Department By Local".

\*\*Objetivo Cumprido\*\*:

2. A criação da tabela "Department By Local" permitirá que você tenha combinações únicas de departamentos e localizações em seu conjunto de dados. Isso é fundamental para a criação de um modelo estrela em um módulo futuro, onde você provavelmente vinculará outras tabelas de fatos e dimensões a essa tabela para análises mais avançadas e complexas.

-- Este é um passo importante na preparação de dados para análises posteriores e na criação de modelos de dados mais robustos.

**14. Explique por que, neste caso supracitado, podemos apenas utilizar o mesclar e não o atribuir**

1. No caso supracitado no desafio, inerente a este passo, só podemos mesclar devido à necessidade de uma operação de mesclagem (join), pois é apropriada quando se deseja combinar dados de diferentes fontes ou tabelas com base em uma chave comum. Nesse caso, é o vínculo dos dados dos gerentes e colaboradores à tabela que contém as informações sobre "store", "Dnumber" e "Mgr\_ssn". A chave comum, no entanto, é o "Mgr\_ssn" (número de Seguro Social do gerente) que aparece em ambas as tabelas. Isso permite a relação de cada funcionário (colaborador) com seu respectivo gerente, bem como com as informações da loja e do departamento em que trabalham.

\*\*Explicação:\*\*

\* A operação de mesclagem (join) é uma operação relacional que combina duas ou mais tabelas com base em uma chave comum.

\* No caso deste desafio, a chave comum é o "Mgr\_ssn" (número de Seguro Social do gerente).

\* A mesclagem dos dados dos gerentes e colaboradores à tabela que contém as informações sobre "store", "Dnumber" e "Mgr\_ssn" permite a relação de cada funcionário (colaborador) com seu respectivo gerente, bem como com as informações da loja e do departamento em que trabalham.

\*\*Exemplo:\*\*

-- Tabela de gerentes(

SELECT

Ssn,

Fname,

Minit,

Lname,

Title

FROM azure\_company.manager)

-- Tabela de colaboradores

(SELECT

Ssn,

Fname,

Minit,

Lname,

HireDate

FROM azure\_company.collaborator)

-- Mesclagem das duas tabelas

SELECT

c.Ssn,

c.Fname,

c.Minit,

c.Lname,

c.HireDate,

mgr.Store,

mgr.Dnumber

FROM azure\_company.collaborator AS c

LEFT JOIN azure\_company.manager AS mgr

ON c.Ssn = mgr.Mgr\_ssn"

Este código SQL mescla a tabela de gerentes e a tabela de colaboradores com base na chave comum "Mgr\_ssn". O resultado é uma nova tabela que contém os dados de todos os funcionários, incluindo os gerentes e os colaboradores.

\*\*Observação:\*\*

\* No caso deste desafio, foi usado o tipo de mesclagem LEFT JOIN. Este tipo de mesclagem retorna todos os registros da tabela da esquerda, mesmo que não haja uma correspondência na tabela da direita. Isso é necessário porque queremos garantir que todos os colaboradores estejam incluídos na nova tabela, mesmo que não tenham um gerente associado.

Espero que isso ajude!

**15. Agrupamento de Dados por Gerente**

\*\*Objetivo:\*\* Saber quantos colaboradores existem por gerente.

\*\*Passos:\*\*

1. Agrupamento da tabela "Staff - Mgr" por meio da coluna Mgr-Staff.

3. Geração do número de gerente e de staff.

4. Visualização dos resultados.

**16. Eliminação de Tabelas Desnecessárias**

\*\*Objetivo:\*\* Remover tabelas extras desnecessárias.

\*\*Passos:\*\*

1. Identificação de tabelas desnecessárias.

2. Remoção das tabelas extras no Power BI.