

Chapter

2

Sistema para Controle de Eventos Acadêmicos

Eduardo Vitor dos Santos Silva e Silva, Elis Marcela de Souza Alcantara,
Otávio Novais de Oliveira, Saulo Matos Pereira Gomes

Abstract

This work presents the development of an academic event management system based on a relational database implemented in PostgreSQL. The proposed approach is centered on the database, using advanced DBMS features for data storage, validation, processing, and report generation. In Marco 2, functionalities such as user authentication, enrollment and cancellation management, and analytical, strategic, and operational dashboards were implemented. All features were developed exclusively through SQL commands, intermediate and advanced queries, materialized views, stored procedures, and transactional control. Tests with large volumes of synthetic data demonstrated satisfactory performance and effective indexing strategies. The results indicate that the adopted approach is suitable for systems that integrate transactional operations with data analysis.

Resumo

Este trabalho apresenta o desenvolvimento de um sistema para controle de eventos acadêmicos baseado em um banco de dados relacional em PostgreSQL. A proposta adotou uma abordagem centrada no banco de dados, utilizando recursos avançados do SGBD para armazenamento, validação, processamento e geração de relatórios. No Marco 2, foram implementadas funcionalidades como autenticação de usuários, controle de inscrições e cancelamentos e dashboards analíticos, estratégicos e operacionais. Todas as funcionalidades foram desenvolvidas exclusivamente por meio de comandos SQL, consultas intermediárias e avançadas, materialized views, stored procedures e controle transacional. Testes com grandes volumes de dados sintéticos demonstraram bom desempenho e eficiência das estratégias de indexação. Os resultados indicam que a abordagem adotada é adequada para sistemas que integram operações transacionais e análise de dados.

2.1. INTRODUÇÃO

A organização de eventos científicos e acadêmicos envolve diversas etapas, como gestão de usuários, controle de inscrições, acompanhamento de pagamentos, registro de presença e emissão de certificados. Em muitos casos, esses processos são executados de forma manual ou descentralizada, o que aumenta a probabilidade de erros, inconsistências e retrabalho. Dessa forma, a adoção de um sistema de informação estruturado torna-se essencial para garantir eficiência, confiabilidade e integridade dos dados envolvidos na realização de eventos acadêmicos.

No contexto da disciplina MATA60 — Banco de Dados, o grupo optou pelo projeto [PRJ9] Sistema para Controle de Eventos Acadêmicos, cujo objetivo é projetar e implementar um banco de dados relacional capaz de apoiar integralmente essas atividades. O sistema desenvolvido contempla funcionalidades essenciais, tais como cadastro de usuários, gerenciamento de eventos (registros), realização de inscrições, controle financeiro por meio de pagamentos associados às inscrições e emissão de certificados para participantes.

Este relatório apresenta a segunda etapa do projeto da disciplina, dando continuidade ao trabalho desenvolvido na primeira etapa. Na fase inicial, realizou-se a estruturação básica do sistema para controle de eventos acadêmicos, com a definição do modelo conceitual e lógico, a implantação das tabelas no PostgreSQL, o preenchimento com dados de teste e os ajustes iniciais de desempenho por meio de indexação.

Nesta segunda etapa, o foco desloca-se para tornar o banco de dados efetivamente útil na operação do sistema. Enquanto a etapa anterior preocupou-se com a forma de armazenar as informações, esta fase dedica-se a como utilizá-las de modo inteligente e aplicado. O objetivo é oferecer suporte concreto às atividades cotidianas, como emissão de certificados, controle de pagamentos e geração de relatórios estratégicos.

Para isso, são utilizados recursos avançados do PostgreSQL. Criaram-se materialized views, que funcionam como relatórios pré-calculados com carregamento rápido; desenvolveram-se stored procedures, que automatizam tarefas repetitivas, como validação de login ou cancelamento de inscrições; e implementaram-se transações, garantindo que

operações críticas como uma inscrição acompanhada de pagamento, sejam completadas com integridade, sem riscos de inconsistência

Todo o sistema foi construído utilizando apenas comandos SQL. Isso significa que todas as interfaces que um usuário encontraria, desde telas de cadastro e login até inscrições e visualização de dashboards, correspondem a consultas e rotinas executadas diretamente no banco de dados. Não são empregadas outras linguagens de programação nem camadas externas; o PostgreSQL, por si só, entrega todas as funcionalidades necessárias

As consultas desenvolvidas enquadram-se nos níveis intermediário e avançado. Elas combinam dados de múltiplas tabelas, realizam cálculos, agregam resultados e aplicam regras de negócio, como validar se um usuário pode inscrever-se em uma atividade somente quando já inscrito no evento principal. Cada consulta está diretamente associada a um requisito do sistema, assegurando que toda funcionalidade planejada seja implementada.

Dessa forma, esta etapa consolida a evolução do projeto: parte-se de um banco de dados bem estruturado e chega-se a um sistema operacional, capaz de suportar tanto tarefas cotidianas quanto análises estratégicas, utilizando apenas recursos nativos do PostgreSQL e estabelecendo uma base sólida para a avaliação funcional e analítica do sistema como um todo.

2.2.DESCRITIVO DO PROJETO

O projeto tem como objetivo o desenvolvimento de um sistema de informação para gerenciamento de eventos acadêmicos, utilizando um banco de dados relacional implementado no SGBD PostgreSQL. O sistema permite o cadastro e a validação de usuários, eventos e inscrições, além da geração de informações analíticas a partir dos dados armazenados.

No Marco 1, o trabalho concentrou-se na construção da base do sistema. Nessa etapa, foi realizada a ampliação do estudo de caso proposto, com a definição dos requisitos e do minimundo do projeto. Em seguida, foram elaboradas as modelagens conceitual e lógica do banco de dados, a implantação física no PostgreSQL, a população das tabelas com dados sintéticos e a criação de planos de indexação para melhorar o desempenho das consultas.

Já no Marco 2, o foco passou a ser o uso mais avançado do banco de dados, indo além do simples armazenamento de informações. Nesta etapa, o banco de dados é utilizado como o principal componente do sistema de informação, sendo responsável por executar regras de negócio, controlar operações e fornecer suporte a análises dos dados.

Para isso, foram utilizados recursos avançados do PostgreSQL, como materialized views, stored procedures e transações. Esses recursos permitem organizar melhor as consultas, centralizar a lógica das operações e garantir a consistência dos dados durante a execução de comandos que envolvem múltiplas etapas.

Os principais artefatos do sistema, como telas de cadastro e validação, além de dashboards analíticos estratégicos e operacionais, foram implementados exclusivamente por meio de comandos SQL, considerando apenas a conexão direta ao banco de dados. Não foi utilizado nenhum tipo de linguagem de programação externa ou camada adicional de back-end, conforme as restrições estabelecidas pela disciplina.

Dessa forma, o Marco 2 representa uma evolução natural do projeto, pois transforma o banco de dados em um elemento ativo do sistema de informação, capaz de atender tanto às operações do dia a dia quanto às demandas analíticas, mantendo aderência aos requisitos definidos e às diretrizes técnicas do trabalho. Os artefatos do sistema e seus respectivos comandos SQL são detalhados na seção seguinte deste relatório.

2.2.1 AJUSTES REALIZADOS NA ENTREGA

Durante o desenvolvimento do Marco 2, foram realizados ajustes em relação à Entrega 1, motivados principalmente pelo amadurecimento do projeto e pelas novas exigências apresentadas nesta etapa. Esses ajustes não alteraram a estrutura central do banco de dados, mas ampliaram a forma como ele é utilizado dentro do sistema de informação, adequando-o às novas demandas funcionais e analíticas.

No Marco 1, o foco esteve na modelagem, implantação, população e otimização do banco de dados, com ênfase na criação de consultas intermediárias e avançadas para avaliação de desempenho. Já no Marco 2, tornou-se necessário reorganizar e estender parte das consultas e rotinas SQL, de modo a atender aos artefatos do sistema definidos no enunciado, como telas de cadastro, validação e dashboards analíticos.

Como parte desses ajustes, foram realizadas alterações estruturais pontuais no esquema do banco de dados. Na tabela TB_Registro, foi adicionada a coluna VL_ValorEvento, responsável por armazenar o valor de inscrição associado a cada evento, permitindo o suporte a consultas analíticas relacionadas a custos e arrecadação.

Na tabela TB_Usuario, foi acrescentada a coluna DS_Senha, com o objetivo de possibilitar o controle de autenticação dos usuários e o acesso às funcionalidades do sistema. Complementarmente, foi criada a tabela TB_RecuperacaoSenha, destinada a apoiar a implementação da funcionalidade de recuperação de senha, disponibilizada na Tela 1 do sistema.

Além disso, foi criada a tabela TB_AuditoriaCancelamento, cuja finalidade é registrar o log das solicitações de cancelamento de inscrições em eventos. Essa tabela dá suporte à funcionalidade de cancelamento implementada na Tela 2, permitindo manter um histórico das operações realizadas e garantindo maior rastreabilidade das ações dos usuários.

2.2.2 ARQUITETURA DO SISTEMA

A arquitetura do Sistema para Controle de Eventos Acadêmicos foi concebida com o objetivo de transformar o banco de dados em um componente ativo e central do sistema de informação, capaz de suportar tanto operações cotidianas quanto demandas analíticas

estratégicas. Para isso, foram utilizados recursos avançados do PostgreSQL, garantindo que toda a lógica de negócio, validação, processamento e geração de relatórios fosse executada diretamente no banco de dados, sem a necessidade de camadas externas de back-end ou linguagens de programação adicionais.

2.2.2.1 Requisitos Do Sistema

A tabela abaixo resume os principais requisitos funcionais atendidos pelo sistema:

Tabela 2.1. Requisitos do sistema

ID	Requisitos	Descrições
R1	Cadastro de usuários	Registro e atualização dos dados dos usuários.
R2	Cadastro de eventos	Criação e gerenciamento de eventos acadêmicos.
R3	Submissão de atividades	Permitir o envio de trabalhos ou atividades vinculadas aos eventos
R4	Inscrição em eventos	Registrar usuários como participantes de eventos.
R5	Processamento de pagamentos	Associar pagamentos às inscrições realizadas.
R6	Participação em atividades	Vincular usuários às atividades do evento e registrar seus papéis.
R7	Emissão de certificados	Emitir certificados de participação/presentes.
R8	Consultas administrativas	Permitir consultas a eventos, inscrições, atividades e usuário.
R9	Integridade dos dados	Garantir consistência e segurança por meio de constraints, chaves e regras de validação
R10	Tipos de inscrição	Modalidades como aluno/profissional/isento.
R11	Código do gateway	Registro do identificador da transação digital.
R12	Validação pública de certificados	Código único para autenticação de certificado.
R13	Registro de presença	Obrigatório para emissão do certificado.
R14	Registro de papéis	Autor, apresentador, ouvinte por atividade.

2.2.2.2 Modelo Conceitual (MER) para o Marco 2 — diagrama Peter Chen

O diagrama entidade-relacionamento (Figura 1.1) foi revisado no Marco 2 para refletir as adições de colunas e tabelas descritas na seção 2.2.1, mantendo a coerência com a arquitetura centrada no banco de dados. As principais entidades e

relacionamentos permanecem estáveis, com acréscimos focados em autenticação, auditoria e análise financeira.

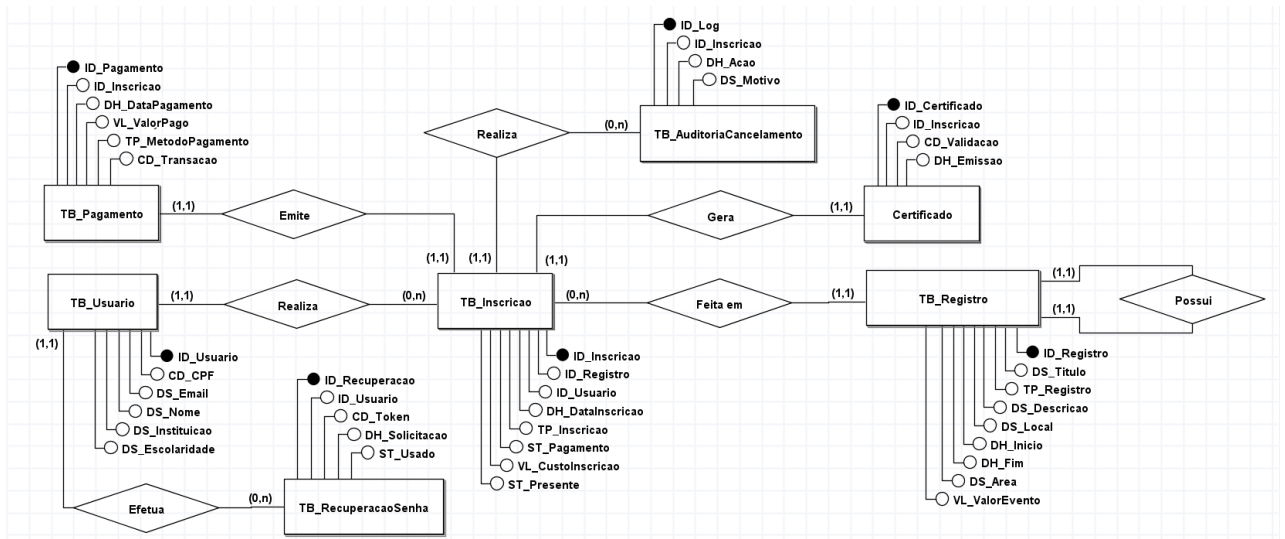


Figura 2.1 - Modelo Conceitual atualizado para o Marco 2

Esta arquitetura assegura que o sistema não seja apenas um repositório de dados, mas sim um sistema operacional completo, capaz de atender com eficiência, segurança e inteligência às demandas de gestão de eventos acadêmicos.

2.3. ARTEFATOS DO SISTEMA

2.3.1 Tela 1: Autenticação e Gestão de Credenciais

Esta interface atua como o ponto de entrada seguro do sistema, integrando as funções de login, solicitação de recuperação e redefinição de credenciais. O fluxo foi projetado para atender aos requisitos de Cadastro de usuários (R1) e Integridade dos dados (R9), garantindo que o acesso seja concedido apenas a portadores de credenciais válidas.

Na primeira etapa (Login), o sistema prioriza a performance e a segurança. A validação das credenciais não consulta a tabela principal diretamente, mas sim uma Materialized View (VM_LoginUsuarios), que contém apenas os dados essenciais (CPF, Hash de Senha e Nome) e possui índice único para busca rápida. A lógica de verificação é encapsulada na Stored Procedure SP_LoginUsuario_Login, que sanitiza o CPF (removendo caracteres não numéricos) antes de validar o acesso.

Caso o usuário necessite recuperar o acesso, o sistema aciona um fluxo seguro gerido pela procedure SP_LoginUsuario_Recuperar. Esta rotina verifica a correspondência entre CPF e E-mail e gera um token criptográfico com validade estrita de 30 minutos, armazenado na tabela TB_RecuperacaoSenha. Para garantir a consistência, a procedure invalida automaticamente quaisquer tokens antigos não utilizados.

Por fim, a etapa de atualização de senha (SP_LoginUsuario_Atualizar) exige o token válido. O sistema verifica se o código pertence ao usuário, se não expirou e se ainda não foi utilizado. Após a validação, a senha é alterada na tabela principal e a Materialized View é atualizada (REFRESH) imediatamente, garantindo que o novo login funcione instantaneamente sem comprometer a integridade do banco de dados.

The image displays three distinct user interface forms stacked vertically. The first form, titled 'LOGIN NO SISTEMA', contains input fields for 'CPF' and 'Senha' (password), an 'ENTRAR' button, and a link '[Esqueci minha senha]'. The second form, titled 'RECUPERAÇÃO DE SENHA', includes input fields for 'CPF' and 'E-mail', and an 'ENVIAR POR E-MAIL' button. The third form, titled 'ATUALIZAR SENHA', features input fields for 'CPF', 'Nova Senha', and 'Token', along with an 'ATUALIZAR' button. All forms have a clean, modern design with blue headers and buttons.

Figura 2.2 - Representação dos processos de Autenticação e Gestão de Credenciais na Tela 1

2.3.2 Tela 2: Gestão de Inscrições e Cancelamento Seguro

Esta interface foi projetada para centralizar o acompanhamento do ciclo de vida das participações do usuário, permitindo a visualização consolidada de inscrições em eventos ativos e históricos. O desenvolvimento desta tela atende primariamente ao requisito R4 (Inscrição em eventos), oferecendo ao usuário controle sobre seus registros.

Para garantir alta performance na recuperação de dados consolidados, unificando informações de eventos, locais, datas e status financeiro, o sistema utiliza uma Materialized View. Este artefato otimiza as Consultas administrativas (R8), reduzindo a carga de processamento em tempo real e permitindo que o usuário verifique rapidamente o status do Processamento de pagamentos (R5) e seus Tipos de inscrição (R10).

Além da consulta, a tela implementa a funcionalidade crítica de cancelamento de inscrição. Esta operação é regida por uma Stored Procedure transacional que aplica regras de negócio estritas, permitindo o cancelamento apenas se solicitado com, no mínimo, 48 horas de antecedência. Ao encapsular essa lógica no banco de dados, o sistema garante a Integridade dos dados (R9), prevenindo inconsistências financeiras. Complementarmente, a segurança é reforçada por uma tabela de auditoria dedicada, que registra automaticamente a autoria, o motivo e o *timestamp* da operação, assegurando a preservação histórica e a rastreabilidade dos fatos.

2.3.3 Dashboard 1: Gráficos Analíticos Estratégicos

O Dashboard 1 foi concebido para oferecer uma visão estratégica consolidada do desempenho do sistema de controle de eventos acadêmicos, atendendo principalmente aos requisitos de consultas administrativas (R8) e análises estratégicas de gestão. Este painel reúne quatro indicadores fundamentais, pré-calculados e organizados em uma Materialized View denominada `mv_dashboard_estrategico_vetores`.

O primeiro gráfico, “Novos Usuários Pagantes por Mês”, mede o crescimento real da base de usuários que realizaram pelo menos um pagamento, agrupando-os por mês e apresentando os resultados em formato de barras, atendendo ao requisito de processamento de pagamentos (R5) e ao monitoramento da base de usuários cadastrados (R1). O segundo indicador, “Ticket Médio Mensal”, calcula o valor médio pago por inscrição em cada mês, utilizando dados da tabela de pagamentos e sendo exibido como um gráfico de linha para facilitar a identificação de tendências ao longo do tempo, reforçando o acompanhamento financeiro associado ao controle de pagamentos (R5).

O terceiro gráfico, “Distribuição Percentual da Receita por Método de Pagamento”, mostra a participação de cada meio de pagamento na receita total, utilizando barras e incluindo tooltips com valores absolutos por método, o que atende ao requisito de registro do código do gateway (R11) e oferece insights sobre os métodos de pagamento preferidos pelos usuários. Por fim, o quarto indicador, “Taxa de Conversão por Evento”, avalia a eficácia de conversão de inscrições em receita, calculando a proporção entre pagantes e inscritos por evento e também sendo visualizado em barras, com tooltips

indicando o número total de pagantes, diretamente relacionado ao requisito de inscrição em eventos (R4) e à qualidade do evento (R14).

Todas essas consultas foram desenvolvidas com SQL avançado, envolvendo agregações, múltiplos JOINS, CTEs e funções de janela, e são atualizadas por meio de uma Stored Procedure específica, garantindo desempenho e consistência nos dados apresentados, além de assegurar a integridade dos dados (R9) por meio de consultas validadas e atualizações controladas.

2.3.4 Dashboard 2: Gráficos Analíticos Operacionais

O Dashboard Operacional foi projetado para fornecer à equipe de gestão uma visão tática e imediata sobre a saúde dos eventos. O desenvolvimento deste painel atende diretamente ao requisito R8 (Consultas Administrativas), centralizando indicadores vitais em uma única interface.

A arquitetura de dados baseia-se em Materialized Views para garantir performance de leitura, isolando o processamento pesado de agregações e janelas temporais da base transacional, o que contribui para a Integridade e consistência dos dados (R9) ao evitar bloqueios em tabelas de escrita.

Abaixo, apresentamos a descrição técnica, funcional e os requisitos atendidos pelos 6 gráficos implementados:

Gráfico 1 – Evolução Temporal com Média Móvel (Gráfico de Linha): Este gráfico é a principal ferramenta para análise de tendência de crescimento, monitorando diretamente o fluxo de Inscrição em eventos (R4). Diferente de uma contagem simples, ele implementa uma análise estatística de Média Móvel de 7 dias (utilizando Window Functions).

Seu objetivo é suavizar oscilações diárias naturais (como quedas em fins de semana) para revelar a tendência real de adesão ao evento. Ele permite identificar se uma campanha de marketing surtiu efeito imediato ou se o interesse pelo evento está esfriando, permitindo ações corretivas rápidas antes da data de realização.

Gráfico 2 – Funil de Conversão Operacional (Gráfico de Barras/Funil): Representa a eficiência do fluxo do usuário através das etapas críticas: Inscritos → Pagantes → Certificados. A construção dos dados utiliza Common Table Expressions (*CTEs*) para agregar múltiplas tabelas e transformá-las em um formato unificado. Requisitos Atendidos: R4 (Inscrição), R5 (Pagamentos), R13 (Registro de Presença) e R7 (Emissão de Certificados).

Com esse gráfico, visamos identificar gargalos operacionais no ciclo de vida do participante. Uma alta quebra entre Inscritos e Pagantes pode indicar problemas no preço ou no gateway de pagamento. Já uma alta quebra entre Pagantes e Certificados (No-Show) indica falhas na comunicação pós-venda ou no engajamento durante o evento.

Gráfico 3 – Ocupação por Modalidade (Gráfico de Rosca/Pizza): Permite segmentar o público total entre as modalidades oferecidas (ex: Presencial vs. Online) para fornecer uma visão clara da distribuição logística dos participantes, validando o requisito de Tipos de inscrição (R10). Essa análise orienta decisões de infraestrutura física. Se a maioria for presencial, prioriza-se o espaço físico e *coffee-break*; se for online, prioriza-se a largura de banda e servidores de *streaming*, otimizando o orçamento.

Gráfico 4 – Status Financeiro: Confirmado vs. Pendente (Gráfico de Barras Empilhadas): Cruza o volume de inscrições com a tabela de pagamentos para categorizar a receita em "Realizada" (Pago) e "Prevista" (Pendente), atendendo ao núcleo do requisito de Processamento de pagamentos (R5). Permite monitorar o fluxo de caixa imediato e a taxa de inadimplência potencial, possibilitando à equipe financeira visualizar quanto da receita estimada já está garantida em caixa. Um alto volume de "Pendentes" próximo à data do evento aciona a necessidade de réguas de cobrança e lembretes de vencimento.

Gráfico 5 – Engajamento Institucional (Gráfico de Barras Horizontais): Ranking das instituições (Universidades ou Empresas) que possuem o maior número de participantes vinculados, baseando-se nos dados coletados no Cadastro de usuários (R1). Tem por objetivo identificar a origem corporativa ou acadêmica do público, fundamental para

estratégias B2B. Identificar as "Top Instituições" permite criar parcerias estratégicas, oferecer descontos para grupos ou focar a publicidade em nichos que já demonstraram alta aderência.

Gráfico 6 – Demanda por Atividades Específicas (Gráfico de Colunas): Aprofunda a visão do evento principal para suas sub-atividades (workshops, minicursos, palestras), utilizando o auto-relacionamento da tabela de registros (Cadastro de eventos - R2) e a intenção de participação (Participação em atividades - R6). Medir a popularidade de temas específicos dentro do evento macro. Essencial para a gestão de espaço físico (alocação de salas). Atividades com alta demanda podem ser movidas para auditórios maiores, enquanto temas de nicho são realocados para salas menores, evitando superlotação ou espaços ociosos.

2.4. ROTINAS IMPLEMENTADAS

2.4.1 Tela 1: Autenticação e Gestão de Credenciais

As rotinas da Tela 1 foram implementadas utilizando exclusivamente comandos SQL no PostgreSQL, conforme as restrições da disciplina. Foram desenvolvidas três Stored Procedures principais que encapsulam toda a lógica de negócio do processo de login e recuperação de senha, além de uma Materialized View otimizada para consultas de autenticação.

A primeira rotina, SP_LoginUsuario_Login, gerencia o processo de login convencional. Quando um usuário insere seu CPF e senha na interface e clica em "Entrar", esta procedure é acionada. Ela começa normalizando o CPF, removendo quaisquer caracteres não numéricos, e então consulta a Materialized View VM_LoginUsuarios para verificar se as credenciais correspondem a um usuário válido. Caso positivo, retorna uma mensagem de sucesso; caso contrário, informa se o problema foi CPF não encontrado ou senha incorreta. Com isso fizemos testes para essa etapa e pode ser observada na Figura 2.3 e 2.4.

```
-- 1: Exibe login do usuário
SELECT * FROM public.vm_loginusuarios
WHERE CD_CPF = '00000000001'

-- 2: Realiza o login
CALL public.sp_loginusuario_login(
    '00000000001',
    'novasenha123'
)
```

Data Output Messages Notifications					
	id_usuario integer	cd_cpf character varying	ds_senha character varying	ds_email character varying	ds_nome character varying
1	1	00000000001	novasenha123	usuario1@exemplo.c...	Nome Ana Lima 1

Figura 2.3 - Informações do usuário 1

Data Output	Messages	Notifications
1	p_resultado character varying	Login OK - Usuário: Nome Ana Lim...

Figura 2.4 - Sucesso no Login

Para a recuperação de senha, implementamos a `SP_LoginUsuario_Recuperar`. Esta rotina é executada quando o usuário, após clicar em "Esqueci minha senha", preenche seu CPF e e-mail e solicita a geração de um token. A procedure valida se o CPF e e-mail correspondem a um registro existente, gera um token único usando uma combinação de CPF, timestamp atual e um valor aleatório criptografado em MD5, invalida tokens anteriores não utilizados do mesmo usuário e armazena o novo token na tabela `TB_RecuperacaoSenha`. Em ambiente de desenvolvimento, o token é exibido na própria tela para fins de demonstração.

```
-- 3: Solicita alteração de senha
CALL public.sp_loginusuario_recuperar(
    '000000000001',
    'usuario1@exemplo.com'
)

-- 4: Busca o Token Gerado
SELECT *
FROM TB_RecuperacaoSenha rs
JOIN TB_Usuario u ON rs.ID_Usuario = u.ID_Usuario
WHERE u.CD_CPF = '000000000001'
ORDER BY DH_Solicitacao DESC;
```

Figura 2.5 - Token Gerado

Data Output

Messages

Notifications

</

Figura 2.6 - Visualizar se o Token ainda é Válido.

A terceira rotina, SP_LoginUsuario_Atualizar, completa o ciclo de recuperação. Quando o usuário recebe o token e insere seu CPF, nova senha e o token na seção de atualização, esta procedure verifica três condições críticas: se o token pertence

ao usuário, se não foi utilizado anteriormente e se foi gerado há menos de 30 minutos. Se todas as validações forem bem-sucedidas, a senha é atualizada na tabela TB_Usuario, o token é marcado como usado e a Materialized View VM_LoginUsuarios é atualizada para refletir imediatamente a mudança.

```
-- 5: Altera senha com token
CALL public.sp_loginusuario_atualizar(
    '000000000001',
    'novasenha0401',
    'c86766587d37971668e107838840fc5b'
)

-- 6: Exibe login do usuário com nova senha
SELECT * FROM public.vm_loginusuarios
WHERE CD_CPF = '000000000001'
```










Data Output		Messages	Notifications
        			
	p_resultado character varying		
1	Senha atualizada com sucesso (via tok...		

Figura 2.7 - Senha Atualizada

Data Output

Messages

Notifications

≡+

▼

▼

SQL

	<div><div>Id_usuario</div><div>integer</div><div>🔒</div></div>	<div><div>cd_cpf</div><div>character varying</div><div>🔒</div></div>	<div><div>ds_senha</div><div>character varying</div><div>🔒</div></div>	<div><div>ds_email</div><div>character varying</div><div>🔒</div></div>	<div><div>ds_nome</div><div>character varying</div><div>🔒</div></div>
1	1	00000000001	novasenha0401	usuario1@exemplo.c...	Nome Ana Lima 1

Figura 2.8 - O usuário com a nova senha salva no banco de dados

A Materialized View `VM_LoginUsuarios` serve como componente de otimização, pré-computando os dados necessários para autenticação (`ID_Usuario`, `CPF`, `senha`, `e-mail` e `nome`) apenas para usuários com credenciais válidas. Ela possui um índice único no campo `CPF` que acelera significativamente as consultas de login. Esta view é atualizada automaticamente sempre que uma senha é modificada através da procedure de atualização, garantindo consistência imediata entre os dados operacionais e a camada de otimização.

A demonstração prática dessas rotinas foi realizada através da execução sequencial dos comandos SQL de teste, conforme ilustrado nas imagens que acompanham este relatório. Primeiro, testou-se o login com credenciais conhecidas, onde foi possível observar a consulta à Materialized View e o retorno das informações do usuário. Em seguida, simulou-se a perda da senha solicitando um token de recuperação, processo que gerou um código único e o registrou na tabela dedicada. Com o token gerado, executou-se a atualização da senha utilizando o token válido, operação que incluiu múltiplas validações e a modificação segura da credencial. Por fim, testou-se novamente o login com a nova senha, confirmando que todo o fluxo foi executado com sucesso. Todas essas operações foram realizadas diretamente no psql, sem necessidade de interfaces gráficas ou camadas intermediárias, demonstrando visualmente como o banco de dados por si só pode prover funcionalidades completas de autenticação.

A implementação completa da Tela 1, incluindo todos os scripts SQL das Materialized Views, Stored Procedures e tabelas de suporte, está disponível no repositório público do projeto no GitHub. O arquivo principal `5_tela_login.sql` contém o código DDL para criação da estrutura e das rotinas, enquanto o arquivo `rotina_de_teste_da_tela_de_login.sql` apresenta a sequência de testes que valida todo o fluxo de autenticação. Esses scripts podem ser acessados, executados e adaptados por qualquer interessado, seguindo o guia passo a passo também disponível no repositório, permitindo a reprodução integral do ambiente de desenvolvimento e a verificação dos resultados apresentados neste relatório.

2.4.2 Tela 2: Gestão de Inscrições e Cancelamento Seguro

Dado que a estrutura do banco de dados foi criada e alimentada segundo os scripts disponibilizados no Github, responsáveis por criar a tabela `TB_AuditoriaCancelamento`, a Materialized View `VM_MinhasInscicoes` e a Stored Procedure `SP_RealizarCancelamentoSeguro`, podemos executar os testes de funcionalidade da Tela 2, realizando alguns casos de testes para a simulação de uso do sistema.

Cenário A – O Usuário entra na tela "Minhas Inscrições": Desse modo, todas as inscrições de eventos referentes ao usuário logado no sistema é exibida.

```
SELECT * FROM VM_MinhasInscricoes
WHERE ID_Usuario = 42
```

Cenário B – Tentativa de Fraude: Um usuário mal intencionado (ID 999) tenta chamar a API para cancelar a inscrição (ID 150) que pertence ao usuário 42.

```
-- Usuário 999 tentando cancelar inscrição 150
CALL sp_RealizarCancelamentoSeguro(150, 999, 'Tentativa de Hack');

-- RESULTADO ESPERADO NO OUTPUT:
-- ERROR: Acesso negado: Inscrição de outro usuário.
```

Cenário C – O usuário tenta cancelar uma inscrição que pertence a ele: Nesse caso, se as condições de cancelamento forem atendidas, o cancelamento deve ser efetuado e o Log registrado na tabela TB_AuditoriaCancelamento.

```
-- 1. Para teste, buscamos o ID de Inscrição de Usuário em um
Evento que ainda não ocorreu
REFRESH MATERIALIZED VIEW VM_MinhasInscricoes;
SELECT
    mi.ID_Inscricao,
    mi.ID_Usuario,
    mi.Nome_Evento,
    mi.Data_Inicio,
    mi.Status_Atual
FROM public.VM_MinhasInscricoes mi
WHERE
    mi.ID_Evento = (
        SELECT ID_Registro
        FROM TB_REGISTRO
        WHERE TP_Registro = 'Evento'
            AND DH_Inicio > CURRENT_TIMESTAMP
        ORDER BY DH_Inicio ASC
        LIMIT 1
    )
    AND mi.Status_Atual = 'Pago'
LIMIT 1;

-- Suponha que a query acima retornou:
-- ID_Inscricao = 150
```

```
-- ID_Usuario = 42
-- Nome_Evento = 'Feira de Hardware e Robótica 2026'
```

Com os IDs em mãos, podemos simular a chamada de cancelamento via Stored Procedure:

```
-- 2. Solicita cancelamento de um evento específico
CALL public.sp_realizarcancelamentoseguro(
    '150', -- id inscrição
    '42', -- id usuário
    'motivos de saúde'
);
-- RESULTADO ESPERADO NO OUTPUT:
-- NOTICE: Inscrição 150 cancelada com sucesso.
```

Cenário D – Verificação de Integridade e Auditoria

```
-- 3. Verificar se o status mudou na tabela original
SELECT ID_Inscricao, ST_Pagamento, ST_Presente
FROM TB_Inscricao
WHERE ID_Inscricao = 150;

-- 4. Consultar LOG de modificação
SELECT * FROM TB_AuditoriaCancelamento
WHERE ID_Inscricao = 150; -- id inscrição
```

2.4.3 Dashboard 1: Gráficos Analíticos Estratégicos

A implementação do Dashboard Estratégico foi pensada para ser prática, eficiente e totalmente integrada ao banco de dados. Tudo começa com a Materialized View chamada `mv_dashboard_estrategico_vetores`. Essa view funciona como uma “tabela especial” que já armazena os dados dos quatro gráficos prontos para uso, em um formato organizado e padronizado. Cada linha da view representa um ponto de um gráfico, com colunas que indicam o nome do gráfico, o tipo de visualização sugerido (como barras ou linhas), o que vai no eixo X (por exemplo, o mês ou o nome do evento), o valor principal no eixo Y, um valor secundário para detalhes extras e um rótulo explicando esse valor secundário. Essa padronização facilita muito o trabalho de quem vai criar os relatórios ou dashboards visuais, pois os dados já vêm formatados e consistentes.

Data Output Messages Graph Visualiser x Notifications							SQL		Show
	nome_grafico text	tipo_grafico_sugerido text	eixo_x text	eixo_y_valor numeric	valor_secundario numeric	rotulo_secundario text			
1	S1_Novos_Usuarios_Pagantes_Por...	bar	2025-12	2748	2748	novos_usuarios			
2	S1_Novos_Usuarios_Pagantes_Por...	bar	2026-01	103	103	novos_usuarios			
3	S2_Ticket_Medio_Mensal	line	2025-12	128.80	3741	qtd_pagamentos			
4	S2_Ticket_Medio_Mensal	line	2026-01	130.59	217	qtd_pagamentos			
5	S3_Distribuicao_Receita_Por_Meto...	bar	Boleto	7.36	37540.00	receita_total_meto...			
6	S3_Distribuicao_Receita_Por_Meto...	bar	Cartão de Crédito	42.74	218080.00	receita_total_meto...			
7	S3_Distribuicao_Receita_Por_Meto...	bar	PIX	49.90	254570.00	receita_total_meto...			
8	S4_Taxa_Conversao_Evento	bar	Congresso de Tecnologia 2025	61.29	801	qtd_pagantes			
9	S4_Taxa_Conversao_Evento	bar	Feira de Hardware e Robótica	59.53	784	qtd_pagantes			
10	S4_Taxa_Conversao_Evento	bar	Semana Acadêmica de IA	60.68	798	qtd_pagantes			
11	S4_Taxa_Conversao_Evento	bar	Simpósio de Redes e Segurança	58.34	759	qtd_pagantes			
12	S4_Taxa_Conversao_Evento	bar	Workshop de Lógica de Programaç...	60.90	816	qtd_pagantes			

Figura 2.9 - “Tabela especial” que já armazena os dados dos quatro gráficos

Para que os gráficos não fiquem desatualizados, criamos uma Stored Procedure chamada `sp_refresh_dashboard_estrategico`. Essa procedure é como um botão de “atualizar” que, quando acionado, recalcula todos os dados da Materialized View com as informações mais recentes do sistema. Ela pode ser programada para rodar automaticamente em horários específicos (por exemplo, toda madrugada) ou ser executada manualmente sempre que necessário, garantindo que os gestores sempre vejam informações atualizadas.

Além da view principal, criamos quatro views dedicadas, uma para cada gráfico: `vw_grafico_s1_novos_usuarios_pagantes_mes`, `vw_grafico_s2_ticket_medio_mensal`, `vw_grafico_s3_receita_pct_por_metodo` e `vw_grafico_s4_taxa_conversao_evento`. Essas views simplificam ainda mais o acesso aos dados, pois permitem que cada gráfico seja consultado de forma isolada, direta e sem complicação. Se uma ferramenta de BI ou uma página web precisar exibir apenas o gráfico de ticket médio, por exemplo, basta consultar a view correspondente, não é necessário filtrar ou tratar dados manualmente.

Antes de considerar o dashboard pronto, realizamos testes completos para validar sua precisão e funcionamento. Primeiro, executamos a procedure de atualização para garantir que a Materialized View fosse preenchida corretamente. Em seguida, consultamos a view principal para verificar se todos os quatro gráficos estavam presentes e com dados consistentes.

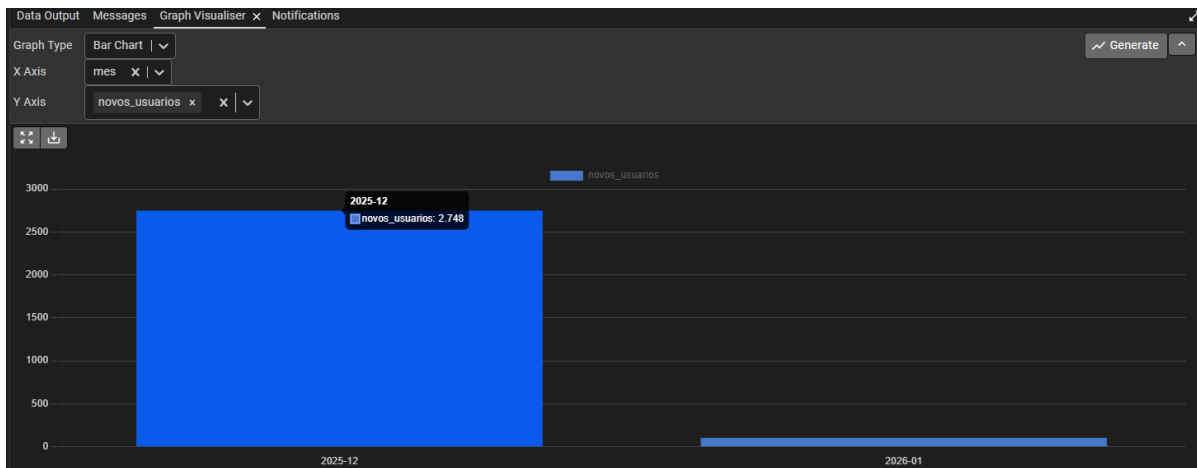


Figura 2.10 - gráfico do vw_grafico_s1_novos_usuarios_pagantes_mes

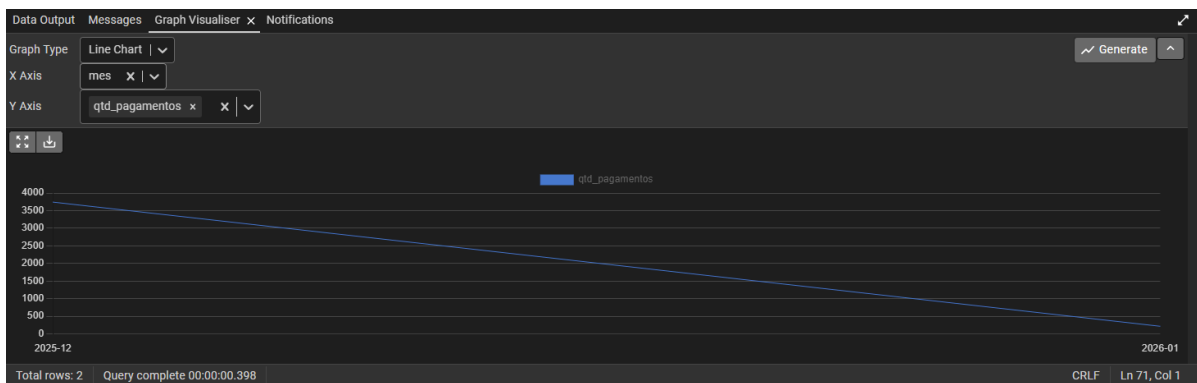


Figura 2.11 - gráfico do vw_grafico_s2_ticket_medio_mensal



Figura 2.12 - gráfico do vw_grafico_s3_receita_pct_por_metodo



Figura 2.13 - gráfico do vw_grafico_s4_taxa_conversao_evento

Uma das grandes vantagens dessa abordagem é que toda a inteligência do dashboard, cálculos, agregações, formatação fica concentrada no próprio banco de dados. Isso significa que não dependemos de programas externos, planilhas ou camadas intermediárias para gerar os indicadores. O resultado é um sistema mais rápido, confiável e fácil de manter, já que qualquer ajuste na lógica dos gráficos pode ser feito diretamente nas views e procedures, sem impactar outras partes do sistema.

2.4.4 Dashboard 2: Gráficos Analíticos Operacionais

Nesta seção, simulamos a rotina de um Gestor de Eventos utilizando o Dashboard 2 para monitorar o andamento do "Congresso de Tecnologia". O objetivo é validar se os dados retornados pelas Materialized Views permitem a tomada de decisões táticas rápidas (requisito R8).

Como o dashboard se baseia em dados analíticos, o primeiro passo é garantir que as visualizações reflitam o estado mais atual do banco de dados transacional. Para isso, o sistema executa a Stored Procedure de orquestração, que atualiza todas as Materialized Views necessárias para o Dashboard.

```
-- O sistema aciona a atualização dos dados (Refresh)
CALL sp_Atualizar_Dashboard_Analitico();
-- Saída esperada: "NOTICE: Dashboard atualizado com sucesso em:
2025-10-25 14:00:00"
```

Após a atualização, o gestor consulta as Views específicas para responder a perguntas críticas de negócio.

A Materialized View `VM_Dash2_Grafico1_TendenciaInscricoes` foi desenvolvida para processar a análise temporal de adesão. Sua função primária é calcular a média móvel com janela de 7 dias, visando suavizar oscilações diárias e eliminar ruídos estatísticos. A estrutura de dados resultante encontra-se normalizada para alimentar diretamente um Gráfico de Linha (*Line Chart*), onde o Eixo X representa a escala cronológica e o Eixo Y projeta duas séries simultâneas: o volume absoluto de inscrições diárias e a curva de tendência calculada.

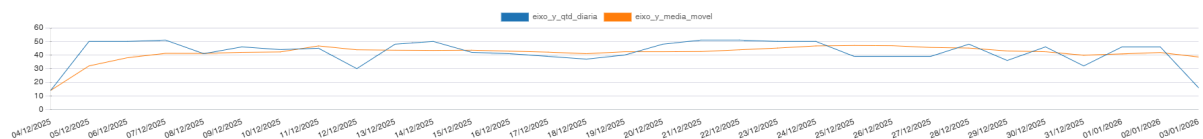


Figura 2.14 - gráfico de linha para a tendência de inscrições em um determinado evento

A materialized view `VM_Dash2_Grafico2_FunilConversao` é responsável por consolidar o ciclo de vida do participante, unificando dados de tabelas distintas (inscrição, pagamento e certificação) através do uso de *Common Table Expressions* (CTEs) e operadores de conjunto (`UNION ALL`). O retorno da query foi estruturado para alimentar um gráfico de Barras (simulando um Funil), onde o eixo X categoriza as etapas sequenciais da jornada e o eixo Y representa o volume absoluto de usuários remanescentes, permitindo a identificação visual imediata de gargalos de conversão.

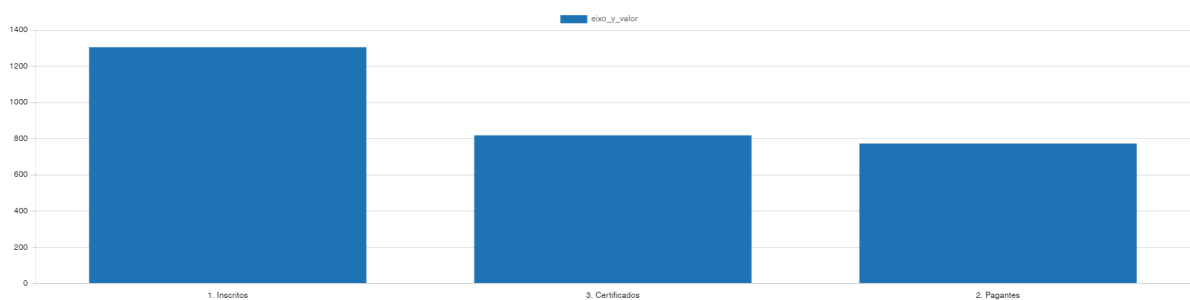


Figura 2.15 - gráfico de barras para o ciclo de vida do participante

A materialized view `VM_Dash2_Grafico3_OcupacaoModalidade` segmenta o volume total de inscritos baseando-se no tipo de participação registrada (Presencial ou Online). O retorno da query agrupa os totais por categoria e está formatado para alimentar um gráfico de Rosca ou Pizza (Pie Chart), onde as fatias representam

proporcionalmente a distribuição logística do público, permitindo a análise imediata da demanda por infraestrutura física versus digital.

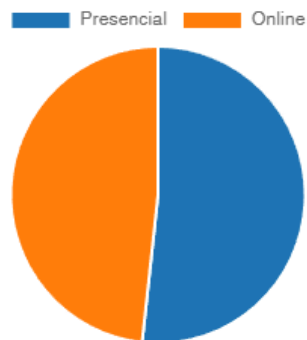


Figura 2.16 - gráfico de pizza para a modalidade dos eventos

A materialized view VM_Dash2_Grafico4_StatusFinanceiro realiza o cruzamento entre inscrições e pagamentos utilizando lógica condicional (CASE WHEN) para categorizar cada registro como "Pago" ou "Pendente". A estrutura de dados resultante foi desenhada para compor um gráfico de Barras Empilhadas (Stacked Bar Chart), onde o eixo X agrupa os eventos e as séries de dados (cores da pilha) distinguem a receita confirmada da prevista, facilitando a visualização da saúde do fluxo de caixa.

A materialized view vm_dash2_grafico5_topinstituicoes quantifica o engajamento dos participantes agrupando-os por sua organização de origem (universidade ou empresa), conforme declarado no cadastro de usuários. O retorno da query é ordenado de forma decrescente para alimentar um gráfico de Barras Horizontais, onde o eixo Y lista as instituições e o eixo X mede o volume de participantes, gerando um ranking claro dos principais parceiros estratégicos do evento.

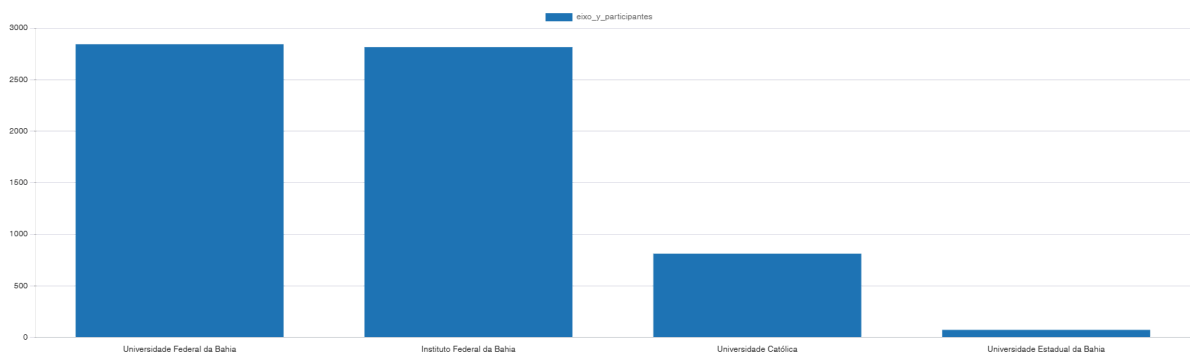


Figura 2.17 - gráfico de barras para a representatividade das instituições

A materialized view `vm_dash2_grafico6_demandaatividades` explora o auto-relacionamento da tabela de registros para isolar e contabilizar as inscrições em sub-eventos (como workshops e palestras). O retorno da query fornece a granularidade necessária para um gráfico de Colunas (Bar Chart vertical), onde o eixo X apresenta os títulos das atividades e o eixo Y exibe a demanda de público, servindo como indicador principal para a gestão de capacidade e alocação de salas.

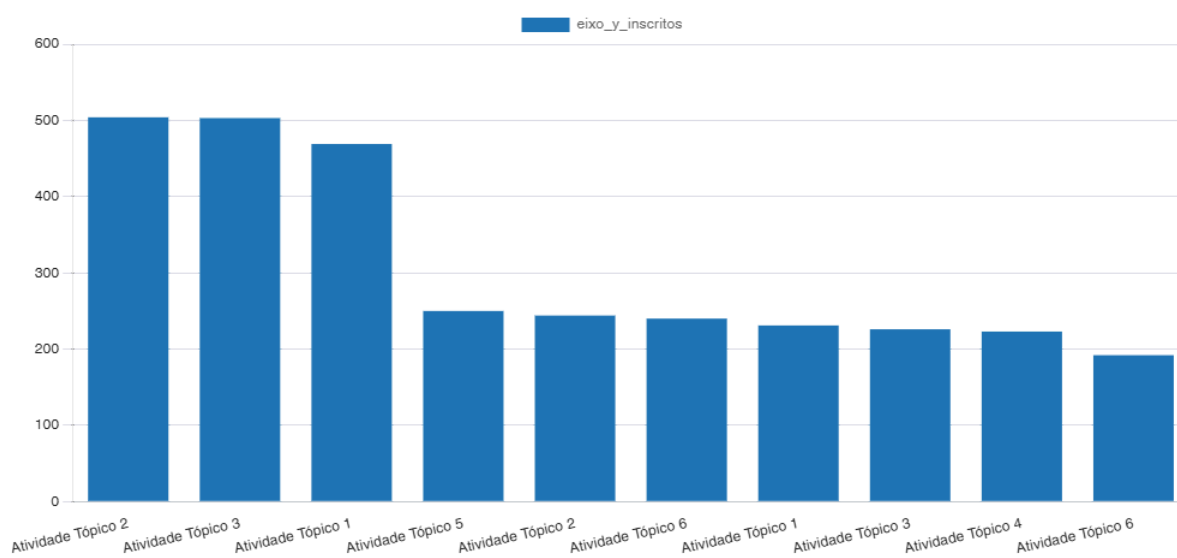


Figura 2.18 - gráfico de barras para analisar a demanda das atividades

2.5.CONCLUSÃO

O desenvolvimento do sistema para controle de eventos acadêmicos evidenciou o potencial do banco de dados relacional como elemento central na construção de sistemas de informação completos e eficientes. Ao longo do Marco 2, foi possível demonstrar que o PostgreSQL pode ir além da função tradicional de armazenamento de dados, assumindo responsabilidades relacionadas à lógica de negócio, validação de regras, processamento das operações e geração de informações analíticas, reduzindo a dependência de camadas externas de programação.

A utilização de recursos avançados do SGBD, como consultas intermediárias e avançadas, materialized views, stored procedures, índices e controle transacional, permitiu a implementação de funcionalidades essenciais do sistema, incluindo autenticação de usuários, gerenciamento de inscrições e cancelamentos e a construção de dashboards analíticos, estratégicos e operacionais. Essa abordagem garantiu consistência transacional, integridade dos dados e desempenho adequado, mesmo em cenários com grande volume de informações, além de facilitar a manutenção e evolução do sistema.

Os testes realizados com os registros sintéticos possibilitaram avaliar o comportamento do banco de dados sob carga e comprovar a eficiência das estratégias de indexação adotadas. Os resultados obtidos indicaram que o sistema é capaz de responder de forma satisfatória a consultas complexas e análises em tempo hábil, atendendo aos requisitos estabelecidos para o projeto e validando as decisões de modelagem e implementação adotadas ao longo do desenvolvimento.

Além de atender ao problema proposto de gestão de eventos acadêmicos, a solução desenvolvida apresenta características que a tornam aplicável a outros domínios que demandam integração entre operações transacionais e análise de dados. A arquitetura centrada no banco de dados demonstrou ser uma alternativa viável e eficiente para sistemas que exigem confiabilidade, desempenho e capacidade analítica, reforçando a importância do banco de dados como componente estratégico em sistemas de informação modernos.

Por fim, o projeto contribuiu para consolidar os conceitos estudados na disciplina, permitindo a aplicação prática de técnicas avançadas de banco de dados em um cenário

realista. O Marco 2 cumpriu seu papel ao demonstrar a evolução do sistema para além do modelo conceitual e lógico, evidenciando a capacidade do banco de dados de sustentar tanto as operações do dia a dia quanto a geração de informações relevantes para apoio à tomada de decisão.

2.6 ANEXOS

Os anexos deste relatório incluem todos os códigos, scripts SQL, modelos de dados e tutoriais necessários para a reprodução completa do projeto. Para facilitar o acesso, organização e versionamento, todo o material foi disponibilizado em um repositório público no GitHub. Nele, encontram-se o script de criação do banco de dados (DDL), o script de população com dados sintéticos, as consultas intermediárias e avançadas, além dos blocos PL/pgSQL utilizados para automatizar a geração dos registros.

O repositório também contém um guia passo a passo para execução do projeto, incluindo instruções para instalação do PostgreSQL, carregamento das tabelas, execução dos índices e realização dos testes de desempenho. Dessa forma, qualquer usuário pode reproduzir integralmente o ambiente utilizado, verificar os resultados apresentados neste relatório e adaptar o projeto conforme suas próprias necessidades.

O link para o repositório GitHub :

https://github.com/Eduardo-vitors/Projeto_Banco_De_Dados

2.7 REFERÊNCIAS

MICROSOFT. Conference Management Toolkit (CMT). Disponível em: <https://cmt3.research.microsoft.com/>. Acesso em: 21/10/2025.

POSTGRESQL GLOBAL DEVELOPMENT GROUP. PostgreSQL Documentation: System Catalogs. Versão 16. Disponível em: <https://www.postgresql.org/docs/current/catalogs.html>. Acesso em: 30 de outubro. 2025.

DATE, C. J. Introdução a Sistemas de Banco de Dados. 8. ed. Rio de Janeiro: LTC, 2004.

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. Sistemas de Banco de Dados. 6. ed. São Paulo: Pearson, 2020.

ELMASRI, R.; NAVATHE, S. B. Fundamentos de Sistemas de Banco de Dados. 7. ed. São Paulo: Pearson, 2019.

CORONATO, A. C.; ALMEIDA, E. PostgreSQL: Banco de Dados Livre para Aplicações Corporativas. Rio de Janeiro: Ciência Moderna, 2019.

ANSI/ISO. Information Technology – Database Languages – SQL: Information Schema and Definition Schema (SQL/Schema). ISO/IEC 9075-11:2016.