



UNIVERSIDADE CATÓLICA DE
PERNAMBUCO

Projeto 2: Eureka!

Projeto de um banco de dados para um evento benéfico realizado pela
Prefeitura do Recife

Guilherme Melo, Marina

**Cantarelli, Rennan Carneiro,
Vinicius Romariz.**

Ciência da Computação

Prof. Lucas Rodolfo Celestino de Farias



Ideia

A Prefeitura do Recife organizou um evento benéfico que ofereceu consultas médicas e uma praça de alimentação para a comunidade. O objetivo foi proporcionar serviços de saúde e alimentação de qualidade. Funcionários da recepção trabalharam em turnos para garantir um atendimento eficiente, e houve um fluxo significativo de pedidos para consultas.

As mesas de atendimento foram bem organizadas, com controle das ocupações, refletindo o sucesso do evento. A praça de alimentação também atraiu muitos visitantes, oferecendo refeições e lanches gratuitos. Em suma, o evento foi um grande sucesso, promovendo um impacto positivo na qualidade de vida da população.

Objetivos do nossa banco de dados

- 1. Organização de Dados:** Centraliza informações sobre participantes, consultas e programação, facilitando a gestão.
- 2. Gerenciamento de Consultas:** Rastreia dados dos participantes para um controle eficaz.
- 3. Acompanhamento de Atendimento:** Monitora a demanda por consultas e uso da praça de alimentação em tempo real.
- 4. Análise de Dados:** Avalia horários de movimento e eficiência dos serviços para melhorias futuras.
- 5. Comunicação Eficiente:** Facilita a comunicação com participantes e colaboradores.
- 6. Relatórios e Transparência:** Gera relatórios sobre o desempenho do evento, promovendo confiança na organização.

Pergunta 1

Qual o total arrecadado com as inscrições dos eventos realizados?

```
SELECT COUNT(*) * 50 AS Total_Arrecadado
FROM Inscricao
WHERE Status = 'Confirmada';
```

- Esse código SQL calcula o total arrecadado com inscrições confirmadas, multiplicando o número de inscrições confirmadas por 50. Ele conta todas as inscrições com `Status = 'Confirmada'` na tabela `Inscricao` e retorna o valor total arrecadado.

```
mysql> SELECT COUNT(*) * 50 AS Total_Arrecadado
-> FROM Inscricao
-> WHERE Status = 'Confirmada';
+-----+
| Total_Arrecadado |
+-----+
|          150      |
+-----+
```

Pergunta 2

Quantas consultas foram realizadas por cada médico durante o evento?

```
SELECT m.ID_Medico, COUNT(c.ID_Consulta) AS Total_Consultas
FROM Medico m
LEFT JOIN Consulta c ON m.ID_Medico = c.ID_Medico
GROUP BY m.ID_Medico;
```

- Esse código SQL conta o total de consultas realizadas por cada médico. Ele faz uma junção entre as tabelas `Medico` e `Consulta`, agrupando os dados por ID de médico e retornando o número de consultas associadas a cada um.

```
mysql> SELECT m.ID_Medico, COUNT(c.ID_Consulta) AS Total_Consultas
-> FROM Medico m
-> LEFT JOIN Consulta c ON m.ID_Medico = c.ID_Medico
-> GROUP BY m.ID_Medico;
+-----+-----+
| ID_Medico | Total_Consultas |
+-----+-----+
| 1         | 1               |
| 2         | 1               |
| 3         | 1               |
| 4         | 1               |
| 5         | 1               |
+-----+-----+
```

Pergunta 3

Qual é o horário de maior movimento de pedidos no restaurante?

```
mysql> SELECT Hora, COUNT(*) AS Total_Pedidos
-> FROM Pedido
-> GROUP BY Hora;
+-----+-----+
| Hora      | Total_Pedidos |
+-----+-----+
| 12:00:00   |           1 |
| 13:00:00   |           1 |
| 14:00:00   |           1 |
| 18:00:00   |           1 |
| 20:00:00   |           1 |
+-----+-----+
```

```
SELECT      Hora,      COUNT(*)      AS
Total_Pedidos
FROM Pedido
GROUP BY Hora;
```

- Esse código SQL conta o número total de pedidos feitos em cada horário. Ele agrupa os dados da tabela `Pedido` pela coluna 'Hora' e retorna o total de pedidos para cada horário específico.

Pergunta 4

Quais funcionários trabalharam na recepção durante o evento e em quais turnos?

```
SELECT f.ID_Funcionario, f.Turno, r.Nome_receptionista
FROM Funcionario f
JOIN Recepcao r ON f.ID_Funcionario = r.ID_Funcionario;
```

- Esse código SQL recupera o ID do funcionário, seu turno e o nome do receptionista para cada funcionário que trabalhou na recepção. Ele faz uma junção entre as tabelas `Funcionario` e `Recepcao` com base no ID do funcionário e retorna essas informações.

```
mysql> SELECT f.ID_Funcionario, f.Turno, r.Nome_receptionista
-> FROM Funcionario f
-> JOIN Recepcao r ON f.ID_Funcionario = r.ID_Funcionario;
+-----+-----+-----+
| ID_Funcionario | Turno      | Nome_receptionista |
+-----+-----+-----+
| 1 | Matutino | Ana
| 2 | Vespertino | Bruno
| 3 | Noturno | Carlos
| 4 | Matutino | Diana
| 5 | Vespertino | Eduardo
+-----+-----+-----+
```

Pergunta 5

Quais funcionários
trabalharam na
recepção durante o
evento e em quais
turnos?

```
mysql> SELECT Status, COUNT(*) AS Total_Mesas
    -> FROM Mesa
    -> WHERE Status IN ('Ocupada', 'Reservada')
    -> GROUP BY Status;
+-----+-----+
| Status | Total_Mesas |
+-----+-----+
| Ocupada | 2 |
| Reservada | 1 |
+-----+-----+
```

```
SELECT Status, COUNT(*) AS Total_Mesas
FROM Mesa
WHERE Status IN ('Ocupada', 'Reservada')
GROUP BY Status;
```

- Esse código SQL conta o número total de mesas que estão com o status "Ocupada" ou "Reservada". Ele filtra a tabela 'Mesa' para esses dois status, agrupa os resultados pelo status e retorna a quantidade de mesas para cada um.

Pergunta 6

Quantos pacientes participaram do evento e quais especialidades médicas foram mais procuradas?

```
mysql> SELECT COUNT(DISTINCT p.ID_Paciente) AS Total_Pacientes,
->           m.Especializacao
->      FROM Paciente p
-> JOIN Consulta c ON p.ID_Paciente = c.ID_Paciente
-> JOIN Medico m ON c.ID_Medico = m.ID_Medico
-> GROUP BY m.Especializacao;
+-----+-----+
| Total_Pacientes | Especializacao |
+-----+-----+
|           1 | Cardiologista |
|           1 | Clínico Geral |
|           1 | Dermatologista |
|           1 | Oftalmologista |
|           1 | Pediatra       |
+-----+-----+
```

```
SELECT COUNT(DISTINCT p.ID_Paciente) AS Total_Pacientes,
m.Especializacao
FROM Paciente p
JOIN Consulta c ON p.ID_Paciente =
c.ID_Paciente
JOIN Medico m ON c.ID_Medico = m.ID_Medico
GROUP BY m.Especializacao;
```

- Esse código SQL conta o número total de pacientes distintos que realizaram consultas em cada especialização médica. Ele agrupa os resultados pela coluna Especializacao da tabela Medico, retornando a quantidade de pacientes atendidos para cada especialização.

Pergunta 7

Qual foi a capacidade de cada local utilizado no evento e quantas pessoas participaram em cada um?

```
mysql> SELECT l.Nome, l.Capacidade, COUNT(i.ID_Participante) AS Participantes
    -> FROM Lugar l
    -> LEFT JOIN Evento e ON l.ID_Lugar = e.ID_Lugar
    -> LEFT JOIN Inscricao i ON e.ID_Evento = i.ID_Evento
    -> GROUP BY l.ID_Lugar;
+-----+-----+-----+
| Nome           | Capacidade | Participantes |
+-----+-----+-----+
| Auditório Central |      200 |          1 |
| Sala de Reuniões |       50 |          1 |
| Teatro Municipal |      300 |          1 |
| Centro de Convenções |  500 |          1 |
| Espaço Gourmet |     150 |          1 |
+-----+-----+-----+
```

```
SELECT l.Nome, l.Capacidade,
COUNT(i.ID_Participante) AS Participantes
FROM Lugar l
LEFT JOIN Evento e ON l.ID_Lugar = e.ID_Lugar
LEFT JOIN Inscricao i ON e.ID_Evento =
i.ID_Evento
GROUP BY l.ID_Lugar;
```

- Esse código SQL conta o número total de participantes inscritos em eventos para cada local. Ele agrupa os resultados pelo ID_Lugar e retorna o nome do local, sua capacidade e a quantidade de participantes inscritos em eventos realizados nesse local.

Pergunta 8

Quais foram os pedidos com maior frequência de consumo?

```
mysql> SELECT i.ID_Item, COUNT(*) AS Total_Vendas
-> FROM Itens i
-> JOIN Pedido p ON i.ID_Pedido = p.ID_Pedido
-> GROUP BY i.ID_Item
-> ORDER BY Total_Vendas DESC;
+-----+-----+
| ID_Item | Total_Vendas |
+-----+-----+
| 3       | 1           |
| 4       | 1           |
| 5       | 1           |
| 6       | 1           |
| 8       | 1           |
| 7       | 1           |
| 9       | 1           |
| 1       | 1           |
| 2       | 1           |
+-----+-----+
```

```
SELECT i.ID_Item, COUNT(*) AS Total_Vendas
FROM Itens i
JOIN Pedido p ON i.ID_Pedido =
p.ID_Pedido
GROUP BY i.ID_Item
ORDER BY Total_Vendas DESC;
```

- Esse código SQL conta o número total de vendas para cada item. Ele agrupa os resultados pelo ID_Item e retorna o identificador do item e a quantidade total de vendas para cada um, ordenando os itens do mais vendido para o menos vendido.

Pergunta 9

Qual foi o número total de itens servidos no restaurante, e quais foram os ingredientes mais utilizados?

```
mysql> SELECT COUNT(*) AS Total_Itens_Servidos
      -> FROM Itens;
+-----+
| Total_Itens_Servidos |
+-----+
|                  9 |
+-----+
```

```
mysql> SELECT Ingredientes, COUNT(*) AS Total_Utilizados
      -> FROM Itens
      -> GROUP BY Ingredientes
      -> ORDER BY Total_Utilizados DESC;
+-----+-----+
| Ingredientes | Total_Utilizados |
+-----+-----+
| Arroz, Feijão |                 1 |
| Picanha       |                 1 |
| Salada        |                 1 |
| Pizza          |                 1 |
| Coca-Cola     |                 1 |
| Churrasco     |                 1 |
| Sanduíche     |                 1 |
| Sopa           |                 1 |
| Bolo           |                 1 |
```

```
SELECT COUNT(*) AS Total_Itens_Servidos
FROM Itens;
```

```
SELECT Ingredientes, COUNT(*) AS Total_Utilizados
FROM Itens
GROUP BY Ingredientes
ORDER BY Total_Utilizados DESC;
```

- Esse código SQL conta o número total de itens servidos.
- O segundo código conta a quantidade de vezes que cada ingrediente foi utilizado. Ele agrupa os resultados pela coluna Ingredientes e retorna a quantidade total de utilizações para cada ingrediente, ordenando do mais utilizado para o menos utilizado.

Pergunta 10

Qual foi o número total de inscrições no evento e quantas foram realizadas por tipo de evento (seminários, palestras, consultas médicas)?

```
mysql> SELECT COUNT(*) AS Total_Inscricoes  
      -> FROM Inscricao;
```

Total_Inscricoes
5

```
mysql> SELECT e.Tipo_evento, COUNT(i.ID_Inscricao) AS Total_Inscricoes  
      -> FROM Evento e  
      -> LEFT JOIN Inscricao i ON e.ID_Evento = i.ID_Evento  
      -> GROUP BY e.Tipo_evento;
```

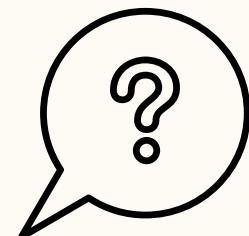
Evento	Total_Inscricoes
Congresso de Saúde	1
Feira de Empreendedorismo	1
Palestra	1
Show	1
Workshop	1

```
SELECT COUNT(*) AS Total_Inscricoes  
FROM Inscricao;
```

```
SELECT e.Tipo_evento,  
COUNT(i.ID_Inscricao) AS  
Total_Inscricoes  
FROM Evento e  
LEFT JOIN Inscricao i ON e.ID_Evento =  
i.ID_Evento  
GROUP BY e.Tipo_evento;
```

- Esse código SQL conta o número total de inscrições. Ele retorna a quantidade total de todas as inscrições registradas na tabela Inscricao.
- O segundo código SQL conta a quantidade de inscrições para cada tipo de evento. Ele agrupa os resultados pela coluna Tipo_evento da tabela Evento e retorna a quantidade de inscrições para cada tipo de evento.

Nosso toque final...



Automação de Relatórios

Geração automática de relatórios semanais com métricas importantes.

Foi implementada uma view, que irá gerar relatórios semanais de comparecimento ao evento. Assim, será possível gerar um relatório a qualquer momento para obter uma visão geral da participação nos eventos na última semana e acompanhar o andamento do programa em geral.



Trigger para Alerta de Alta Demanda em Especialidades Médicas

Torna o monitoramento das especialidades dinâmico e cria um histórico que pode ser consultado facilmente.

Trigger que insere um alerta em uma tabela de "Alertas", sempre que o número de consultas em uma especialidade médica ultrapassa um limite definido. É útil para monitorar demandas e gerar relatórios de especialidades com alta procura. Faz com que o monitoramento das especialidades seja dinâmico, criando um histórico que pode ser consultado facilmente.

Link do nosso repositório no GitHub

- <https://github.com/GuiMelo012x/Projeto-2-Eureka.git>

Obrigado !!

Agradeçemos a sua presença! :)