

TABELA USUÁRIOS

```
CREATE TABLE usuarios (  
id int NOT NULL AUTO_INCREMENT,  
nome varchar(100) NOT NULL,  
email varchar(150) NOT NULL,  
telefone varchar(20) DEFAULT NULL,  
permissoes enum('admin','voluntario') NOT NULL DEFAULT 'voluntario',  
status enum('pendente','aprovado') NOT NULL DEFAULT 'pendente',  
criado_em timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,  
atualizado_em timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON  
UPDATE CURRENT_TIMESTAMP,  
senha_hash varchar(255) NOT NULL, PRIMARY KEY (id) )
```

```
SELECT email FROM USUARIOS WHERE email =  
'carla.mendes@example.com';
```

```
CREATE UNIQUE INDEX idx_usuarios_email ON usuarios (email);
```

ANTES DO ÍNDICE

The screenshot shows a database query execution tool with the following components:

- Query Builder:** Displays the SQL query: `SELECT email FROM USUARIOS WHERE email = 'carla.mendes@example.com';`
- Execution Plan:** A table showing the operations performed by the database engine.

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				1
TABLE ACCESS	USUARIOS	FULL		1

The execution plan indicates a full table scan on the 'USUARIOS' table, which is inefficient for this query. The tool also shows a 'Filter Predicates' section with the condition `EMAIL='carla.mendes@example.com'` and an 'Other XML' section with metadata.

DEPOIS DO ÍNDICE

The screenshot shows a database query editor with two tabs: 'Planilha' and 'Query Builder'. The 'Query Builder' tab is active, displaying a SQL query:

```
SELECT email FROM USUARIOS WHERE email = 'carla.mendes@example.com';  
  
CREATE UNIQUE INDEX idx_usuarios_email ON usuarios (email);
```

Below the query, the 'Plano de Explicação' (Execution Plan) tab is visible, showing the execution plan for the SQL query. The plan is as follows:

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				1
INDEX	IDX_USUARIOS_EMAIL	UNIQUE SCAN		1

The execution plan also includes a tree view showing the query structure:

- SELECT STATEMENT
 - INDEX
 - Access Predicates
 - EMAIL='carla.mendes@example.com'
 - Other XML
 - {info}
 - info type="db_version"
 - 11.2.0.2
 - info type="parse schema"

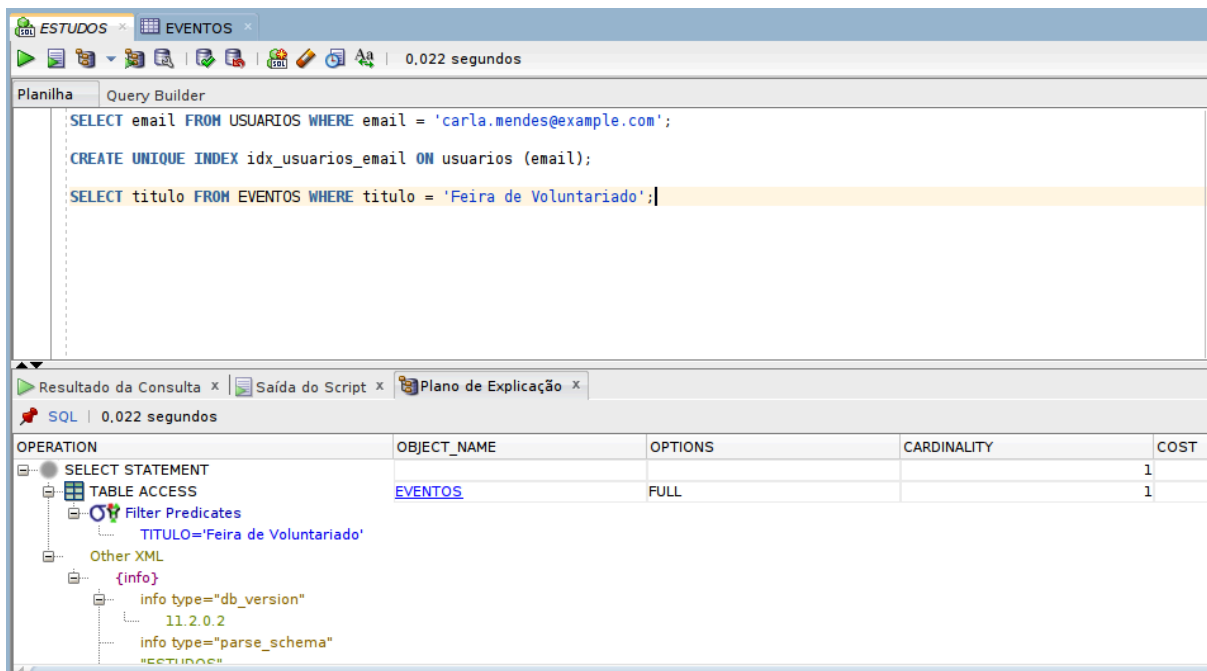
TABELA EVENTOS

```
CREATE TABLE eventos (  
  id int NOT NULL AUTO_INCREMENT,  
  titulo varchar(150) NOT NULL,  
  descricao text,  
  data_evento date NOT NULL,  
  horario time DEFAULT NULL,  
  local varchar(200) DEFAULT NULL,  
  status enum('planejado','finalizado','cancelado') NOT NULL DEFAULT  
  'planejado',  
  criado_em timestamp NULL DEFAULT CURRENT_TIMESTAMP,  
  atualizado_em timestamp NULL DEFAULT CURRENT_TIMESTAMP ON  
  UPDATE CURRENT_TIMESTAMP, PRIMARY KEY (id) )
```

```
SELECT titulo FROM EVENTOS WHERE titulo = 'Feira de Voluntariado';
```

```
CREATE INDEX idx_eventos_titulo ON eventos (titulo);
```

ANTES DO ÍNDICE



The screenshot shows the SQL Developer interface with a query script in the 'Query Builder' tab. The script contains three SQL statements: a SELECT query on the USUARIOS table, a CREATE INDEX statement, and another SELECT query on the EVENTOS table. The execution plan for the second SELECT query is displayed in the 'Plano de Explicação' tab, showing a full table scan on the EVENTOS table.

```
SELECT email FROM USUARIOS WHERE email = 'carla.mendes@example.com';

CREATE UNIQUE INDEX idx_usuarios_email ON usuarios (email);

SELECT titulo FROM EVENTOS WHERE titulo = 'Feira de Voluntariado';
```

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				1
TABLE ACCESS	EVENTOS	FULL		1

Additional details from the execution plan:

- Filter Predicates: TITULO='Feira de Voluntariado'
- Other XML: {info}
- info type="db_version": 11.2.0.2
- info type="parse_schema": "ESTUDOS"

DEPOIS DO ÍNDICE

The screenshot shows a database query editor with the following SQL commands:

```

SELECT email FROM USUARIOS WHERE email = 'carla.mendes@example.com';

CREATE UNIQUE INDEX idx_usuarios_email ON usuarios (email);

SELECT titulo FROM EVENTOS WHERE titulo = 'Feira de Voluntariado';

CREATE INDEX idx_eventos_titulo ON eventos (titulo);

```

Below the SQL editor, the execution plan is displayed for the second query (CREATE INDEX). The plan shows the following operations:

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
CREATE INDEX STATEMENT				82
INDEX BUILD	IDX_EVENTOS_TITULO	NON UNIQUE		
INDEX	IDX_EVENTOS_TITULO	CREATE INDEX		82
INDEX	IDX_EVENTOS_TITULO	FAST FULL SCAN		82

Additional information shown in the plan includes:

- Other XML: {info}
- info type="db_version": 11.2.0.2
- info type="parse_schema": "ESTUDOS"

Por que criei esses índices:

Eu criei o índice no campo **email** da tabela **usuarios** e no campo **titulo** da tabela **eventos** porque são colunas que aparecem com frequência nas cláusulas **WHERE** das consultas do sistema.

No caso de **usuarios.email**, o sistema precisa autenticar e localizar usuários pelo e-mail a todo momento (login, recuperação de senha, etc.). Sem um índice, o banco faria uma varredura completa na tabela a cada login. Com o índice, os valores de e-mail ficam organizados em uma estrutura B-tree, o que permite que o SGBD encontre rapidamente o registro correspondente, reduzindo bastante o tempo de resposta.

No caso de **eventos.titulo**, a aplicação também precisa buscar eventos específicos pelo título ou mostrar detalhes de um evento filtrado por título. O índice neste campo evita “full table scans” na tabela de eventos e acelera essas consultas.

Esses índices afetam positivamente o desempenho porque diminuem o número de páginas lidas do disco/memória para atender a cada **SELECT**. Ou seja, eles reduzem a carga do servidor e tornam as consultas muito mais escaláveis conforme as tabelas crescem.

Por outro lado, eu também sei que índices têm custo: cada vez que um **INSERT**, **UPDATE** ou **DELETE** é feito, o índice precisa ser atualizado. Por isso, escolhi criar índices apenas nas colunas realmente usadas para filtro/pesquisa frequente, onde o ganho de leitura supera o custo adicional de escrita.

Em resumo, os índices ajudam a tornar as operações de leitura muito mais rápidas e eficientes para a API e o back-end, ao mesmo tempo em que mantêm o modelo de dados consistente e pronto para crescer.