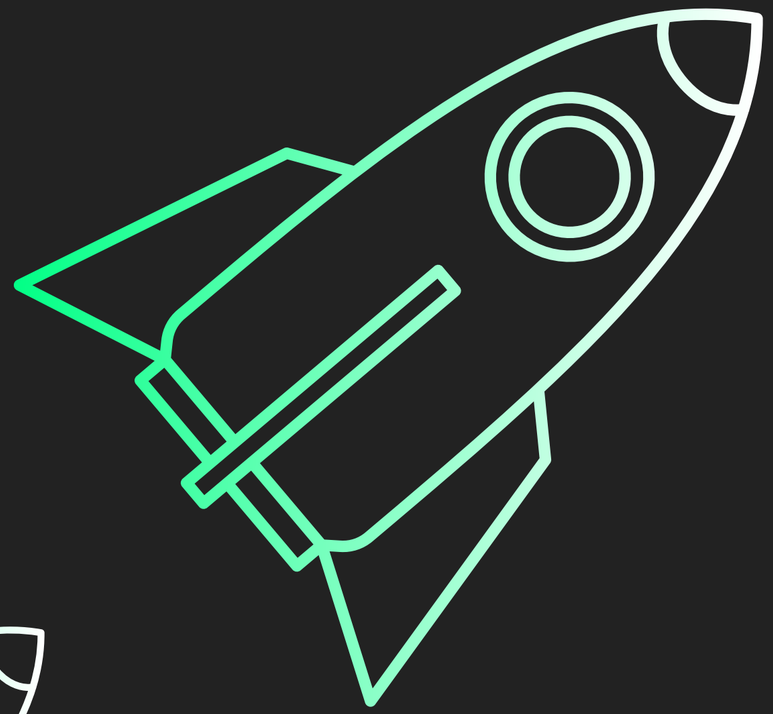
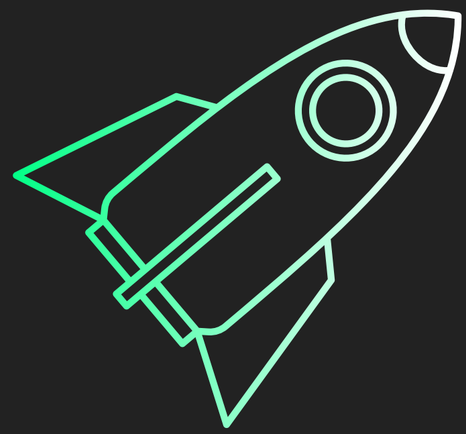
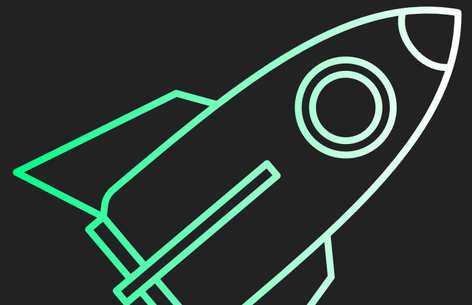
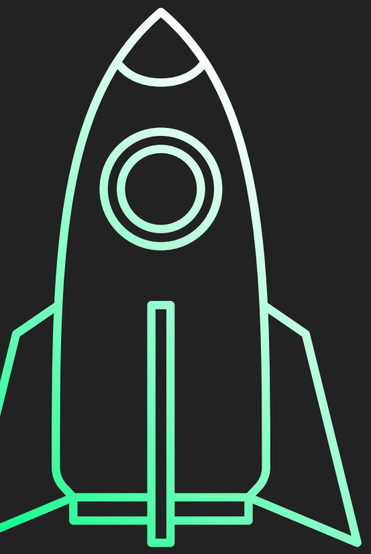
# BANCO DE DADOS AVANÇADO



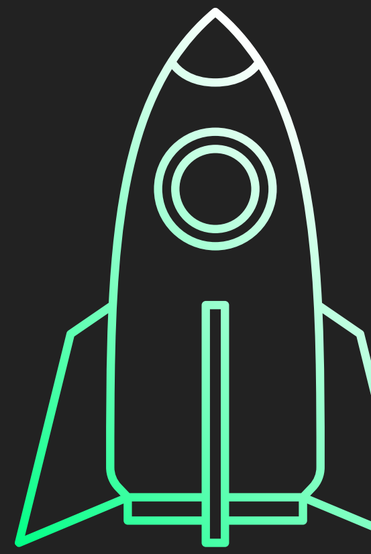




PROJETO DB



# Tabelas



# Entidades

1 USUÁRIO

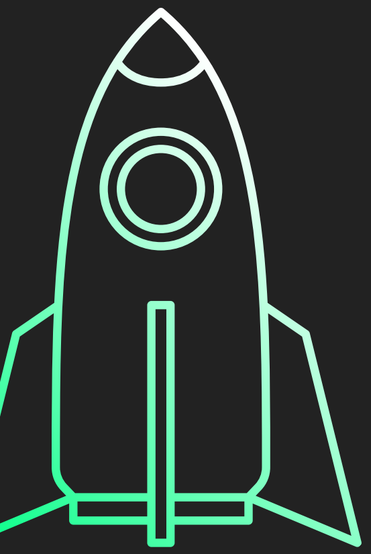
1 PRATOS

1 PEDIDOS

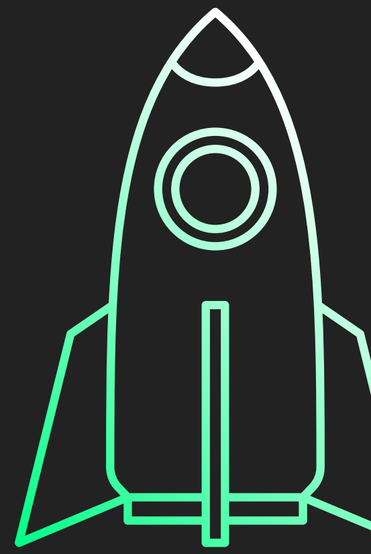
2 RESTAURANTES

1 ENTREGADORES

1 ITENS_PEDIDO



# SubConsultas



# SUBCONSULTAS

```sql
SELECT restaurantes.nome, precos.media_preco
FROM (
    SELECT restaurante_id, AVG(preco)::numeric(10,2) AS media_preco
    FROM pratos
    GROUP BY restaurante_id
) AS precos
JOIN restaurantes ON restaurantes.id = precos.restaurante_id;
```

Data Output | Messages | Notifications

Showing rows: 1 to

| | nome character varying (100) | media_preco numeric (10,2) |
|---|---|---|
| 1 | Pizzaria Bella | 32.50 |
| 2 | Hamburgueria Top | 20.00 |
| 3 | Sushi House | 19.00 |

# SUBCONSULTAS

```
61
62  SELECT usuarios.nome,
63         (SELECT COUNT(*) FROM pedidos
64         WHERE pedidos.usuario_id = usuarios.id) AS total_pedidos
65  FROM usuarios;
```

Data Output | Messages | Notifications

SQL | Showing row

| | nome<br>character varying (100) | total_pedidos<br>bigint |
|---|---|---|
| 1 | Carlos Silva | 2 |
| 2 | Maria Oliveira | 2 |
| 3 | João Souza | 1 |

# SUBCONSULTAS

```sql
SELECT restaurantes.nome
FROM restaurantes
WHERE EXISTS (
    SELECT 1
    FROM pratos
    WHERE pratos.restaurante_id = restaurantes.id
      AND pratos.disponivel = TRUE
);
```

Data Output | Messages | Notifications

SQL

| | nome<br>character varying (100) |
|---|---|
| 1 | Pizzaria Bella |
| 2 | Hamburgueria Top |
| 3 | Sushi House |

# SUBCONSULTAS

```sql
SELECT restaurantes.nome
FROM restaurantes
WHERE NOT EXISTS (
    SELECT 1
    FROM pratos
    WHERE pratos.restaurante_id = restaurantes.id
);
```

Data Output | Messages | Notifications

SQL

| nome<br>character varying (100) |
| --- |

# SUBCONSULTAS

```
75
76  SELECT restaurantes.nome
77  FROM restaurantes
78  WHERE NOT EXISTS (
79      SELECT 1
80      FROM pratos
81      WHERE pratos.restaurante_id = restaurantes.id
82  );
```

Data Output | Messages | Notifications

SQL

| nome<br>character varying (100) |
| --- |

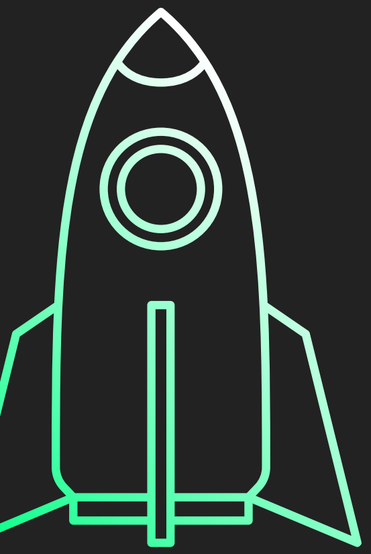
# SUBCONSULTAS

```
83
84  SELECT usuarios.nome, usuarios.email
85  FROM usuarios
86  WHERE usuarios.id IN (
87      SELECT pedidos.usuario_id
88      FROM pedidos
89      GROUP BY pedidos.usuario_id
90      HAVING COUNT(*) > 1
91  );
```

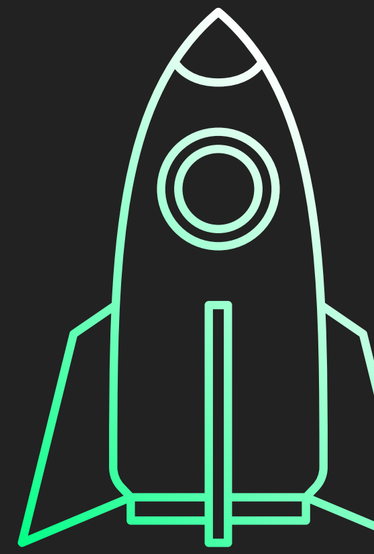
Data Output | Messages | Notifications

SQL

| | nome<br>character varying (100) | email<br>character varying (100) |
|---|---|---|
| 1 | Carlos Silva | carlos.silva@email.com |
| 2 | Maria Oliveira | maria.oliveira@email.com |



# Joins



# JOINS

```sql
92
93  SELECT pedidos.id, usuarios.nome AS cliente, restaurantes.nome AS restaurante
94  FROM pedidos
95  INNER JOIN usuarios ON pedidos.usuario_id = usuarios.id
96  INNER JOIN restaurantes ON pedidos.restaurante_id = restaurantes.id;
97
```

Data Output | Messages | Notifications

Showing rows: 1 to 5 | Pag

| | id integer | cliente character varying (100) | restaurante character varying (100) |
|---|---|---|---|
| 1 | 1 | Carlos Silva | Pizzaria Bella |
| 2 | 2 | Maria Oliveira | Hamburgueria Top |
| 3 | 3 | Carlos Silva | Sushi House |
| 4 | 4 | João Souza | Pizzaria Bella |
| 5 | 5 | Maria Oliveira | Sushi House |

# JOINS

```
98  SELECT restaurantes.nome AS restaurante, pratos.nome AS prato
99  FROM restaurantes
100 LEFT JOIN pratos ON pratos.restaurante_id = restaurantes.id;
```

Data Output | Messages | Notifications

Showing row

| | restaurante character varying (100) | prato character varying (100) |
|---|---|---|
| 1 | Pizzaria Bella | Pizza Margherita |
| 2 | Pizzaria Bella | Pizza Calabresa |
| 3 | Hamburgueria Top | Hambúrguer Clássico |
| 4 | Hamburgueria Top | Milkshake Chocolate |
| 5 | Sushi House | Sushi Salmão |
| 6 | Sushi House | Temaki Atum |

# JOINS

```
101
102 SELECT pratos.nome AS prato, restaurantes.nome AS restaurante
103 FROM pratos
104 RIGHT JOIN restaurantes ON pratos.restaurante_id = restaurantes.id;
105
```

Data Output | Messages | Notifications

SQL | Showing rows: 1 to

| | prato character varying (100) | restaurante character varying (100) |
|---|---|---|
| 1 | Pizza Margherita | Pizzaria Bella |
| 2 | Pizza Calabresa | Pizzaria Bella |
| 3 | Hambúrguer Clássico | Hamburgueria Top |
| 4 | Milkshake Chocolate | Hamburgueria Top |
| 5 | Sushi Salmão | Sushi House |
| 6 | Temaki Atum | Sushi House |

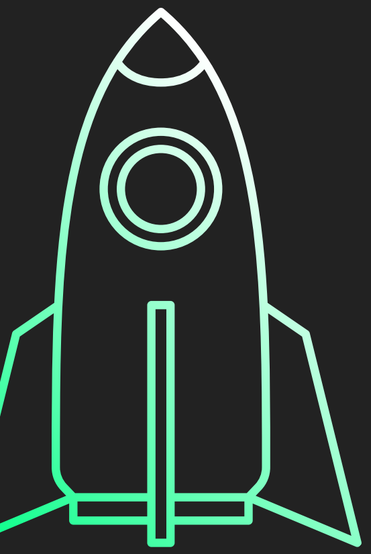
# JOINS

```sql
SELECT restaurantes.nome, pedidos_aggregados.total_pedidos
FROM restaurantes
JOIN (
    SELECT pedidos.restaurante_id, COUNT(*) AS total_pedidos
    FROM pedidos
    GROUP BY pedidos.restaurante_id
) AS pedidos_aggregados
ON restaurantes.id = pedidos_aggregados.restaurante_id;
```

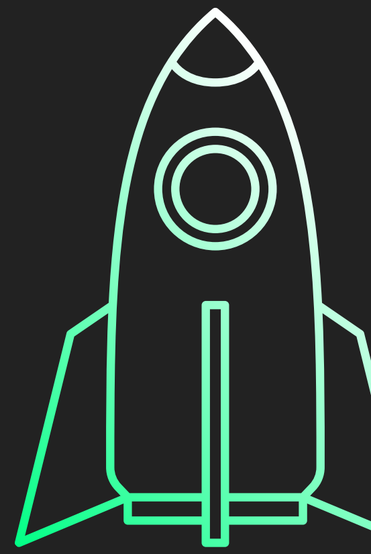
Data Output | Messages | Notifications

Showing rows

| | nome character varying (100) | total_pedidos bigint |
|---|---|---|
| 1 | Pizzaria Bella | 2 |
| 2 | Hamburgueria Top | 1 |
| 3 | Sushi House | 2 |



# Procedure



# PROCEDURES

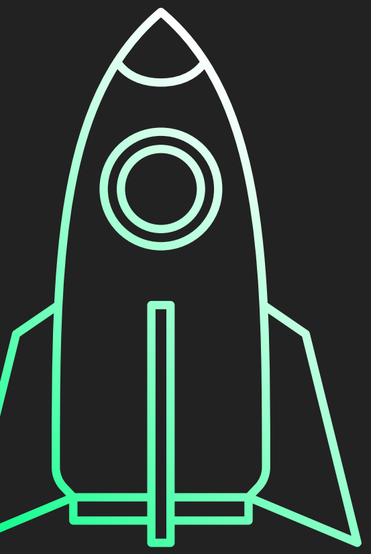
```
CREATE OR REPLACE PROCEDURE listar_pedidos()
LANGUAGE plpgsql
AS $$
BEGIN
    RETURN QUERY
    SELECT pedidos.id AS pedido_id, usuarios.nome AS nome_usuario, restaurantes.nome AS no
    FROM pedidos
    JOIN usuarios ON pedidos.usuario_id = usuarios.id
    JOIN restaurantes ON pedidos.restaurante_id = restaurantes.id;
END;
$$;

CALL listar_pedidos();
```

# PROCEDURES

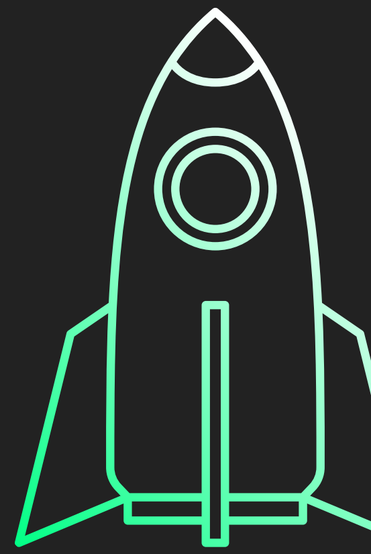
```
CREATE OR REPLACE PROCEDURE buscar_pedidos(usuario_id_param INTEGER)
LANGUAGE plpgsql
AS $$
BEGIN
    RETURN QUERY
    SELECT id, usuario_id, status
    FROM pedidos
    WHERE usuario_id = usuario_id_param;
END;
$$;
```

# PROCEDURES

```
CREATE OR REPLACE PROCEDURE atualizar_status_pedido(pedido_id_param INTEGER, novo_status_p
LANGUAGE plpgsql
AS $$
BEGIN
    UPDATE pedidos
    SET status = novo_status_param
    WHERE id = pedido_id_param;
END;
$$;
```



# Functions



# FUNCTIONS

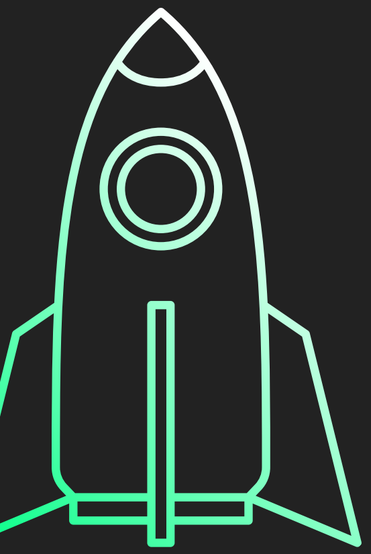
```sql
CREATE OR REPLACE FUNCTION obter_preco_prato(prato_id_param INTEGER)
RETURNS NUMERIC
LANGUAGE sql
AS $$
    SELECT preco FROM pratos WHERE id = prato_id_param;
$$;
```

# FUNCTIONS

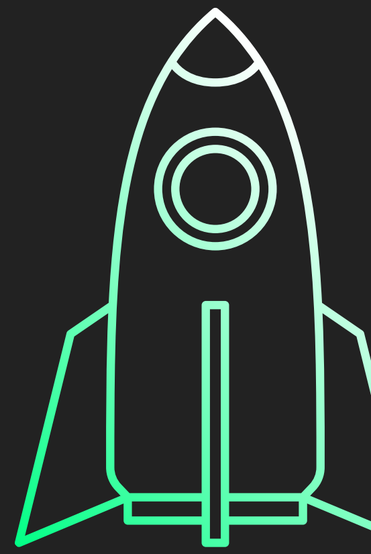
```sql
CREATE OR REPLACE FUNCTION contar_pedidos_usuario(usuario_id_param INTEGER)
RETURNS INTEGER
LANGUAGE sql
AS $$
    SELECT COUNT(*) FROM pedidos WHERE usuario_id = usuario_id_param;
$$;
```

# FUNCTIONS

```sql
CREATE OR REPLACE FUNCTION listar_pratos_disponiveis()
RETURNS TABLE(id INTEGER, nome VARCHAR, preco NUMERIC)
LANGUAGE sql
AS $$
    SELECT id, nome, preco FROM pratos WHERE disponivel = TRUE;
$$;
```



# Triggers



# TRIGGERS

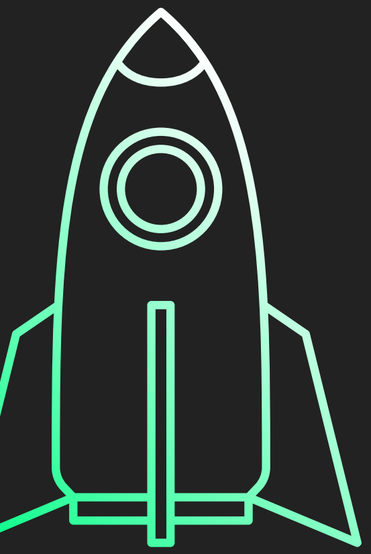
```sql
CREATE OR REPLACE FUNCTION funcao_trigger_auditar_pedidos()
RETURNS TRIGGER
LANGUAGE plpgsql
AS $$
BEGIN
    INSERT INTO registro_pedidos (pedido_id, operacao, data_hora)
    VALUES (NEW.id, TG_OP, now());
    RETURN NEW;
END;
$$;

CREATE TRIGGER trigger_auditar_pedidos
AFTER INSERT OR UPDATE OR DELETE ON pedidos
FOR EACH ROW EXECUTE FUNCTION funcao_trigger_auditar_pedidos();
```

# TRIGGERS

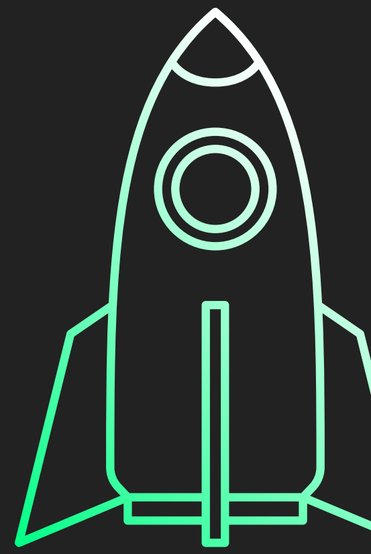
```
CREATE OR REPLACE FUNCTION funcao_trigger_deletar_itens_do_pedido()
RETURNS TRIGGER
LANGUAGE plpgsql
AS $$
BEGIN
    DELETE FROM itens_pedido WHERE pedido_id = OLD.id;
    RETURN OLD;
END;
$$;

CREATE TRIGGER trigger_deletar_itens_apos_exclusao_pedido
AFTER DELETE ON pedidos
FOR EACH ROW EXECUTE FUNCTION funcao_trigger_deletar_itens_do_pedido();
```

# TRIGGERS

```sql
CREATE OR REPLACE FUNCTION funcao_trigger_bloquear_exclusao_entregador_com_pedidos_ativos()
RETURNS TRIGGER
LANGUAGE plpgsql
AS $$
BEGIN
    IF EXISTS (
        SELECT 1 FROM pedidos
        WHERE entregador_id = OLD.id AND status <> 'finalizado'
    ) THEN
        RAISE EXCEPTION 'Não é permitido excluir entregador com pedidos ativos.';
    END IF;
    RETURN OLD;
END;
$$;

CREATE TRIGGER trigger_bloquear_exclusao_entregador_com_pedidos_ativos
BEFORE DELETE ON entregadores
FOR EACH ROW EXECUTE FUNCTION funcao_trigger_bloquear_exclusao_entregador_com_pedidos_ativo
```



# índicis



# INDEXES

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

CREATE INDEX idx_pratos_restaurante_id ON pratos(restaurante_id);

CREATE INDEX idx_pedidos_restaurante_status ON pedidos(restaurante_id, status);

CREATE INDEX idx_pedidos_usuario_id ON pedidos(usuario_id);

CREATE INDEX idx_itens_pedido_pedido_id ON itens_pedido(pedido_id);
```

# MUITO OBRIGADO!