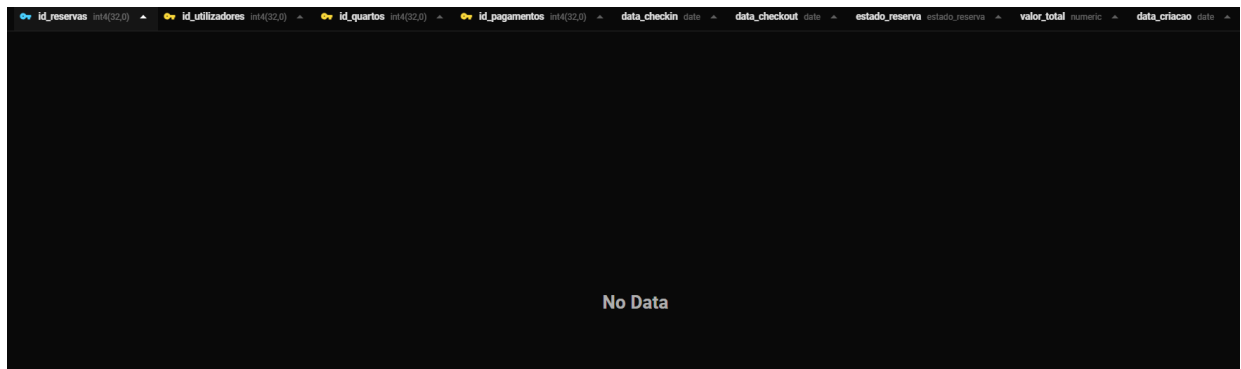


## Exercicio 1:

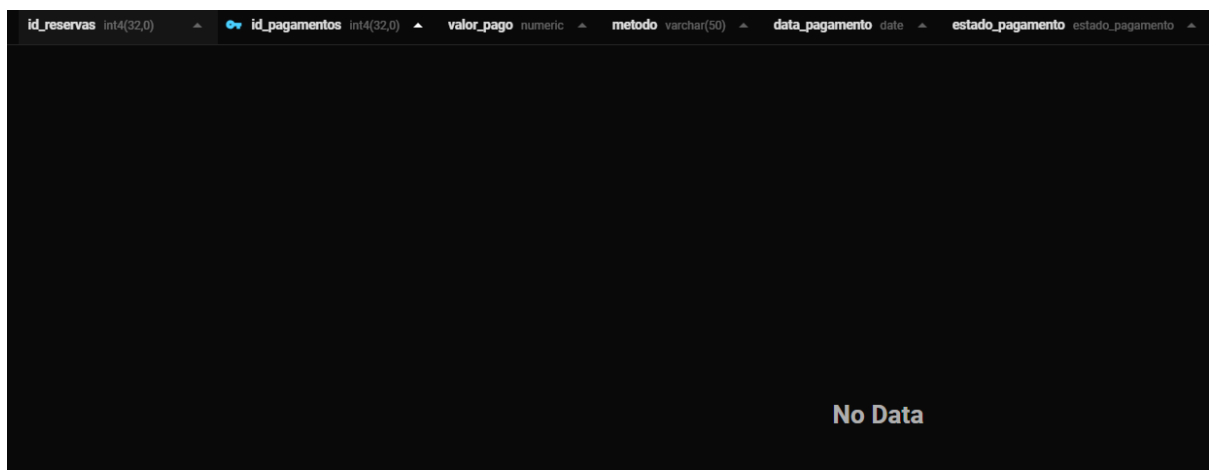
Antes da criação dos procedimentos.



The screenshot shows a database interface with a table named 'reservas'. The table has the following columns: id\_reservas (int4(32,0)), id\_utilizadores (int4(32,0)), id\_quartos (int4(32,0)), id\_pagamentos (int4(32,0)), data\_checkin (date), data\_checkout (date), estado\_reserva (estado\_reserva), valor\_total (numeric), and data\_criacao (date). The table is currently empty, displaying 'No Data'.

id_reservas	id_utilizadores	id_quartos	id_pagamentos	data_checkin	data_checkout	estado_reserva	valor_total	data_criacao
No Data								

*Figura 1 - tabela reservas*



The screenshot shows a database interface with a table named 'pagamentos'. The table has the following columns: id\_reservas (int4(32,0)), id\_pagamentos (int4(32,0)), valor\_pago (numeric), metodo (varchar(50)), data\_pagamento (date), and estado\_pagamento (estado\_pagamento). The table is currently empty, displaying 'No Data'.

id_reservas	id_pagamentos	valor_pago	metodo	data_pagamento	estado_pagamento
No Data					

*Figura 2 - tabela pagamentos*

Código da criação dos procedimentos:

```
CREATE OR REPLACE PROCEDURE inserir_reserva (r_id_cliente INT, r_id_quarto INT,
r_data_checkin DATE, r_data_checkout DATE)
LANGUAGE plpgsql
AS $$
DECLARE
    r_id_reserva INT; -- variavel para guardar o id da reserva para depois associar ao
    pagamento
BEGIN
    -- Inserir reserva com estado 'pendente'
    INSERT INTO RESERVAS (ID_UTILIZADORES, ID_QUARTOS, DATA_CHECKIN, DATA_CHECKOUT,
    ESTADO_RESERVA, VALOR_TOTAL, DATA_CRIACAO)
    VALUES (r_id_cliente, r_id_quarto , r_data_checkin, r_data_checkout, 'pendente',
    (SELECT preco FROM QUARTOS WHERE ID_QUARTOS = r_id_quarto) ,now()) RETURNING ID_RESERVAS
    INTO r_id_reserva;
    call inserir_pagamento(r_id_reserva, 'multibanco');

END;
$$;

CREATE OR REPLACE PROCEDURE inserir_pagamento(p_id_reserva INT, p_metodo VARCHAR(50))
LANGUAGE plpgsql
AS $$
DECLARE
    p_id_pagamento INT; -- variavel para guardar o id do pagamento para depois associar a
    reserva
BEGIN
    INSERT INTO PAGAMENTOS (ID_RESERVAS, VALOR_PAGO, METODO, DATA_PAGAMENTO,
    ESTADO_PAGAMENTO)
    VALUES (p_id_reserva, (SELECT valor_total FROM RESERVAS WHERE ID_RESERVAS =
    p_id_reserva), p_metodo, now(), 'pago') RETURNING ID_PAGAMENTOS INTO p_id_pagamento;
    UPDATE RESERVAS SET ESTADO_RESERVA = 'confirmada', ID_PAGAMENTOS = p_id_pagamento
    WHERE ID_RESERVAS = p_id_reserva;
```

Chamada dos procedimentos para inserir reservas e pagamentos:

```
CALL inserir_reserva(3, 1, '2021-06-01', '2021-06-05');
CALL inserir_reserva(3, 2, '2021-06-01', '2021-06-05');
```

Pós execução:

id_pagamentos	id_utilizadores	id_quartos	id_reservas	data_checkin	data_checkout	estado_reserva	valor_total	data_criacao
1	3	1	1	2021-06-01	2021-06-05	confirmada	50.00	2025-03-18
2	3	2	2	2021-06-01	2021-06-05	confirmada	50.00	2025-03-18

Figura 4 - pós procedimento reservas

id_pagamentos	id_reservas	valor_pago	metodo	data_pagamento	estado_pagamento
1	1	50.00	multibanco	2025-03-18	pago
2	2	50.00	multibanco	2025-03-18	pago

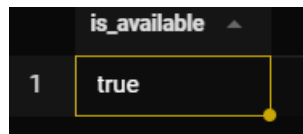
Figura 3 - pós procedimento tabela pagamentos

## Exercicio 2:

```
create or replace function is_available(num_quarto int, data date)
returns boolean as $result$
declare
    result boolean;
begin
    select exists(
        select quartos.id_quartos
        from quartos, reservas
        where quartos.id_quartos = reservas.id_quartos
            and data between reservas.data_checkin and
reservas.data_checkout
            and quartos.numero = num_quarto
    ) into result;
    return not result;
end;
$result$ language plpgsql;
```

Execução:

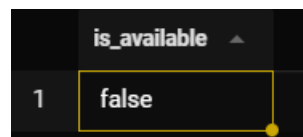
```
SELECT is_available(101, '2025-02-04')
```



	is_available
1	true

Figura 5 - pós função com um valor válido

```
SELECT is_available(101, '2021-06-03')
```



	is_available
1	false

Figura 6 - pós função com um valor inválido

Exercicio 3:

Codigo:

```
create or replace function status_trigger_func()
returns trigger as $status_trigger$
begin
    if new.estado_reserva = 'confirmada' then
        update quartos
            set estado_quarto = 'ocupado'
            where id_quartos = new.id_quartos;
    elseif new.estado_reserva = 'cancelada' then
        update quartos
            set estado_quarto = 'livre'
            where id_quartos = new.id_quartos;
    end if;
    return new;
end;
$status_trigger$ language plpgsql;

create trigger status_trigger
after insert or update on reservas
for each row
execute procedure status_trigger_func();
```

id_utilizadores	id_quartos	id_reservas	data_checkin	data_checkout	estado_reserva
1	1	1	2021-06-01	2021-06-05	cancelada
2	2	2	2021-06-01	2021-06-05	confirmada

Figura 8 - pos criação do trigger tabela reservas

id_quartos	numero	tipo_quarto	descricao	preco	imagem	estado_quarto
1	101	single	Quarto com vi...	50.00	quarto1.jpg	livre
2	102	single	Quarto com vi...	50.00	quarto2.jpg	ocupado

Figura 7 - pós criação do trigger tabela quartos

Os exercício 1 e 2 funcionam na sua totalidade, o exercício 3 altera corretamente o status do quarto diretamente associado a reserva no entanto não alterar qualquer outro quarto.