

Aluno A do Grupo – Lucas Sousa (1171589)

Exercício 1)

Para resolver este exercício temos que dado um ID de uma reserva, retornar o quarto a alocar, segundo restrições impostas no enunciado e em caso de erros levantar exceções sugestivas. *Para a resolução desta alínea foram criadas 3 funções que complementam a principal **fncGetQuartoReserva**, todas usadas nesta função principal.*

```
Function FNCRESERVAEXISTENTE compiled  
  
Function FNCESTADOADEQUADORESERVA compiled  
  
Function FNCRESERVACOMQUARTOASSOCIADO compiled  
  
Function FNCGETQUARTORESERVA compiled
```

Figura 1- Screenshot da compilação das funções

```
Quarto para alocar -> ID DO QUARTO: 6; ID DO ANDRA: 1; NÚMERO DO QUARTO: 6; ID DO TIPO DE QUARTO: 3; LOTÇÃO MÁXIMA: 2  
  
PL/SQL procedure successfully completed.
```

Figura 2- Screenshot dos resultados caso se insira um id válido

```
Não é possível atribuir um quarto uma vez que já existe um quarto para alocar a reserva: 132  
  
PL/SQL procedure successfully completed.
```

Figura 3- Screenshot dos resultados caso o quarto já tenha um quarto atribuído

```
A reserva inserida é NULL!!!  
  
PL/SQL procedure successfully completed.
```

Figura 4- Screenshot dos resultados caso o id seja null

Exercício 2)

Neste exercício devemos efetuar o check-out de um quarto e gerar uma fatura.

```
1 row inserted.  
  
1 row inserted.  
  
1 row inserted.  
  
1 row inserted.  
  
1 row inserted.
```

Figura 5- Screenshot das linhas adicionadas a base de dados

Function GET_LINHA_RESERVA compiled

Procedure PRCCHECKOUT compiled

Figura 6- Screenshot da compilação do procedimento e da função auxiliar

```
Checkout gerado -> Id da Reserva = 19977; Data = 20.12.21; Observações = ; Valor Extra = €;  
Gerada Fatura -> Id = 19977; Número = 20219977; Id Cliente = 1; Id Reserva = 19977; Valor Reserva = 180€; Valor Consumo = 177€;  
  
PL/SQL procedure successfully completed.
```

Figura 7- Screenshot dos resultados caso se insira uma reserva válida

```
Error starting at line : 147 in command -  
DECLARE  
    linha_reserva RESERVA%rowtype;  
BEGIN  
    linha_reserva := get_linha_Reserva(3620);  
    prcCheckOut(linha_reserva);  
END;  
Error report -  
ORA-20001: Parametros inválidos, reserva sem checkin, ou linha NULL !!!  
ORA-06512: na "BDDAD_DG2_B.PRCCHECKOUT", linha 94  
ORA-06512: na linha 5
```

Figura 8- Screenshot dos resultados caso se insira uma reserva sem checkin

```
Error starting at line : 140 in command -  
DECLARE  
    linha_reserva RESERVA%rowtype;  
BEGIN  
    linha_reserva := get_linha_Reserva(3651);  
    prcCheckOut(linha_reserva);  
END;  
Error report -  
ORA-20001: Parametros inválidos, reserva sem checkin, ou linha NULL !!!  
ORA-06512: na "BDDAD_DG2_B.PRCCHECKOUT", linha 94  
ORA-06512: na linha 5
```

Figura 9- Screenshot dos resultados caso se insira uma reserva inexistente

```
Error starting at line : 154 in command -  
DECLARE  
    linha_reserva RESERVA%rowtype;  
BEGIN  
    linha_reserva := get_linha_Reserva(3330);  
    prcCheckOut(linha_reserva);  
END;  
Error report -  
ORA-20001: Já existe CHECKOUT!!!  
ORA-06512: na "BDDAD_DG2_B.PRCCHECKOUT", linha 92  
ORA-06512: na linha 5
```

Figura 10- Screenshot dos resultados caso se insira uma reserva já com checkout

Exercício 3)

Este exercício tinha que garantir que a inserção/alteração de uma época não conduz a sobreposição entre de datas entre épocas.

Trigger TRGEPOCASNAOSOBREPOSTAS compiled

Figura 11- Screenshot da compilação do trigger

1 row inserted.

Figura 12- Screenshot da inserção da linha relativa ao exercício

```
Error starting at line : 38 in command -
insert into epoca (id,nome,data_ini,data_fim) values (112,'Epoca ERRADA',TO_DATE('03-03-2020','dd-mm-yyyy'),TO_DATE('01-10-2021','dd-mm-yyyy'))
Error report -
ORA-20006: Existe sobreposição entre as datas das época no sistema!!!
ORA-06512: na "BDDAD_DG2_B.TRGEPOCASNAOSOBREPOSTAS", linha 27
ORA-04088: erro durante a execução do trigger 'BDDAD_DG2_B.TRGEPOCASNAOSOBREPOSTAS'
```

Figura 13- Screenshot da inserção de uma época sobreposta

```
Error starting at line : 36 in command -
UPDATE epoca SET data_ini=TO_DATE('02-07-2020','dd-mm-yyyy') WHERE id=2
Error report -
ORA-20006: Existe sobreposição entre as datas das época no sistema!!!
ORA-06512: na "BDDAD_DG2_B.TRGEPOCASNAOSOBREPOSTAS", linha 27
ORA-04088: erro durante a execução do trigger 'BDDAD_DG2_B.TRGEPOCASNAOSOBREPOSTAS'

Error starting at line : 37 in command -
UPDATE epoca SET data_fim=TO_DATE('20-11-2020','dd-mm-yyyy') WHERE id=2
Error report -
ORA-20006: Existe sobreposição entre as datas das época no sistema!!!
ORA-06512: na "BDDAD_DG2_B.TRGEPOCASNAOSOBREPOSTAS", linha 27
ORA-04088: erro durante a execução do trigger 'BDDAD_DG2_B.TRGEPOCASNAOSOBREPOSTAS'
```

Figura 14- Screenshot do update de uma época sobreposta

Aluno B do Grupo – Guilherme Daniel (1181743)

Exercício 4)

Este exercício tinha como objetivo devolver um cursor com informação relativa aos consumos registados por cada uma das camareiras. Para isso recorreu-se a blocos anónimos e foram lançadas algumas exceções para dados inválidos ou resultados vazios.

```
idCamareira: 13 | Nome: Camareira 3 | Valor total: 121 | Data primeiro registo: 20.06.01 | Data ultimo registo: 20.06.30 | Qtd dias sem registos: 10
idCamareira: 11 | Nome: Camareira 1 | Valor total: 180 | Data primeiro registo: 20.06.03 | Data ultimo registo: 20.06.30 | Qtd dias sem registos: 11
idCamareira: 14 | Nome: Camareira 4 | Valor total: 117 | Data primeiro registo: 20.06.01 | Data ultimo registo: 20.06.29 | Qtd dias sem registos: 10
idCamareira: 15 | Nome: Camareira 5 | Valor total: 79 | Data primeiro registo: 20.06.01 | Data ultimo registo: 20.06.30 | Qtd dias sem registos: 9
idCamareira: 12 | Nome: Camareira 2 | Valor total: 151 | Data primeiro registo: 20.06.01 | Data ultimo registo: 20.06.29 | Qtd dias sem registos: 13
idCamareira: 16 | Nome: Camareira 6 | Valor total: 183 | Data primeiro registo: 20.06.01 | Data ultimo registo: 20.06.30 | Qtd dias sem registos: 7
idCamareira: 17 | Nome: Camareira 7 | Valor total: 162 | Data primeiro registo: 20.06.02 | Data ultimo registo: 20.06.29 | Qtd dias sem registos: 9
idCamareira: 18 | Nome: Camareira 8 | Valor total: 204 | Data primeiro registo: 20.06.01 | Data ultimo registo: 20.06.30 | Qtd dias sem registos: 11
idCamareira: 19 | Nome: Camareira 9 | Valor total: 170 | Data primeiro registo: 20.06.01 | Data ultimo registo: 20.06.29 | Qtd dias sem registos: 10
idCamareira: 20 | Nome: Camareira 10 | Valor total: 116 | Data primeiro registo: 20.06.02 | Data ultimo registo: 20.06.30 | Qtd dias sem registos: 8

PL/SQL procedure successfully completed.
```

Figura 15- Screenshot dos resultados do exercício 4 para o mês de junho de 2020

Valores inválidos!

PL/SQL procedure successfully completed.

Figura 16- Screenshot dos resultados caso se insira um parâmetro inválido

```
Não há dados para a data inserida!  
  
PL/SQL procedure successfully completed.
```

Figura 17- Screenshot dos resultados caso não haja dados no mês/mês e data inseridos por parâmetro

Exercício 5)

Para a resolução deste exercício e do próximo, foi criada uma entidade Bónus, especificada no modelo relacional.

```
Bonus da camareira com id 14, atualizado para 0 para 1/2020  
Bonus da camareira com id 15, atualizado para 0 para 1/2020  
Bonus da camareira com id 11, atualizado para 6,5 para 1/2020  
Bonus da camareira com id 12, atualizado para 5,6 para 1/2020  
Bonus da camareira com id 17, atualizado para 0 para 1/2020  
Bonus da camareira com id 18, atualizado para 5,15 para 1/2020  
Bonus da camareira com id 20, atualizado para 0 para 1/2020  
Bonus da camareira com id 13, atualizado para 0 para 1/2020  
Bonus da camareira com id 19, atualizado para 6,7 para 1/2020  
Bonus da camareira com id 16, atualizado para 0 para 1/2020  
Bonus da camareira com id 14, atualizado para 0 para 2/2020  
Bonus da camareira com id 15, atualizado para 0 para 2/2020  
Bonus da camareira com id 17, atualizado para 11,1 para 2/2020  
Bonus da camareira com id 11, atualizado para 9,25 para 2/2020
```

Figura 18- Output dos resultados caso a atualização das camareiras seja executada com sucesso

```
O mês 13 introduzido por parametro é inválido!  
O mês -1 introduzido por parametro é inválido!
```

Figura 19- Output caso os parâmetros inseridos sejam inválidos

Valores na tabela Bonus:

ID Camareira:	14		Mes:	1		Ano:	2020		Valor de bonus:	0
ID Camareira:	15		Mes:	1		Ano:	2020		Valor de bonus:	0
ID Camareira:	11		Mes:	1		Ano:	2020		Valor de bonus:	6,5
ID Camareira:	12		Mes:	1		Ano:	2020		Valor de bonus:	5,6
ID Camareira:	17		Mes:	1		Ano:	2020		Valor de bonus:	0
ID Camareira:	18		Mes:	1		Ano:	2020		Valor de bonus:	5,15
ID Camareira:	20		Mes:	1		Ano:	2020		Valor de bonus:	0
ID Camareira:	13		Mes:	1		Ano:	2020		Valor de bonus:	0
ID Camareira:	19		Mes:	1		Ano:	2020		Valor de bonus:	6,7

Figura 20- Excerto dos valores que a tabela bônus apresenta depois da chamada do procedimento

Exercício 6)

Inicialmente, serão inseridos dois bônus para duas camareiras distintas (11 e 14). Os testes para cada uma serão efetuados a partir apenas destas duas inserções.

ID_CAMAREIRA	MES_BONUS	ANO_BONUS	VALOR_BONUS
1	14	6	2020
2	11	6	2020

Figura 21- Bônus de camareiras inseridos

Caso o valor de bônus a tentar ser atualizado for inferior ao valor de bônus previamente registado, ou o aumento seja superior a 50%, serão lançadas duas exceções distintas:

```
Error starting at line : 36 in command -
UPDATE bonus
SET valor_bonus = 99
WHERE id_camareira = 11 AND mes_bonus = 6 AND ano_bonus = 2020
Error report -
ORA-20000: O bônus a atualizar é menor que o bônus previamente inserido
ORA-06512: na "BDDAD_DG2_A.TRGCORRIGIRALTERACAOBONUS", linha 20
ORA-04088: erro durante a execução do trigger 'BDDAD_DG2_A.TRGCORRIGIRALTERACAOBONUS'
```

Figura 22- Resultado obtido caso o aumento seja inferior ao anterior

```
Error starting at line : 40 in command -
UPDATE bonus
SET valor_bonus = 200
WHERE id_camareira = 11 AND mes_bonus = 6 AND ano_bonus = 2020
Error report -
ORA-20001: O valor de aumento do bônus tem de ser no máximo 50%
ORA-06512: na "BDDAD_DG2_A.TRGCORRIGIRALTERACAOBONUS", linha 22
ORA-04088: erro durante a execução do trigger 'BDDAD_DG2_A.TRGCORRIGIRALTERACAOBONUS'
```

Figura 23- Resultado obtido caso o aumento seja superior a 50%

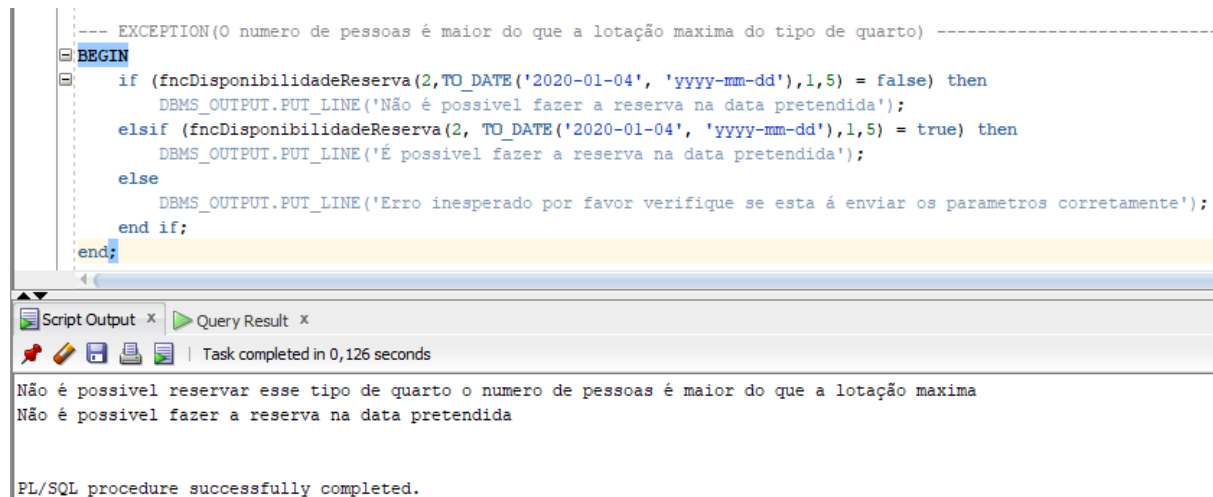
Caso contrário, será obtido o resultado habitual de uma atualização bem sucedida: **"1 row updated."**

Aluno C do Grupo – Kevin Sousa (1180853)

Exercício 7)

O objetivo de este exercício era criar uma função designada fncDisponibilidadeReserva que permitirá verificar a disponibilidade para uma possível reserva onde recebíamos como parâmetros o tipo de quarto , a data pretendida , a duração em dias e o numero de pessoas.

Caso o numero de pessoas seja maior do que a lotação máxima do tipo de quarto , será lançada uma exceção (lotacao_maxima_excedida) :



```
----- EXCEPTION(0 numero de pessoas é maior do que a lotação máxima do tipo de quarto) -----  
BEGIN  
  if (fncDisponibilidadeReserva(2,TO_DATE('2020-01-04', 'yyyy-mm-dd'),1,5) = false) then  
    DBMS_OUTPUT.PUT_LINE('Não é possível fazer a reserva na data pretendida');  
  elsif (fncDisponibilidadeReserva(2, TO_DATE('2020-01-04', 'yyyy-mm-dd'),1,5) = true) then  
    DBMS_OUTPUT.PUT_LINE('É possível fazer a reserva na data pretendida');  
  else  
    DBMS_OUTPUT.PUT_LINE('Erro inesperado por favor verifique se esta á enviar os parametros corretamente');  
  end if;  
end;
```

Script Output x Query Result x

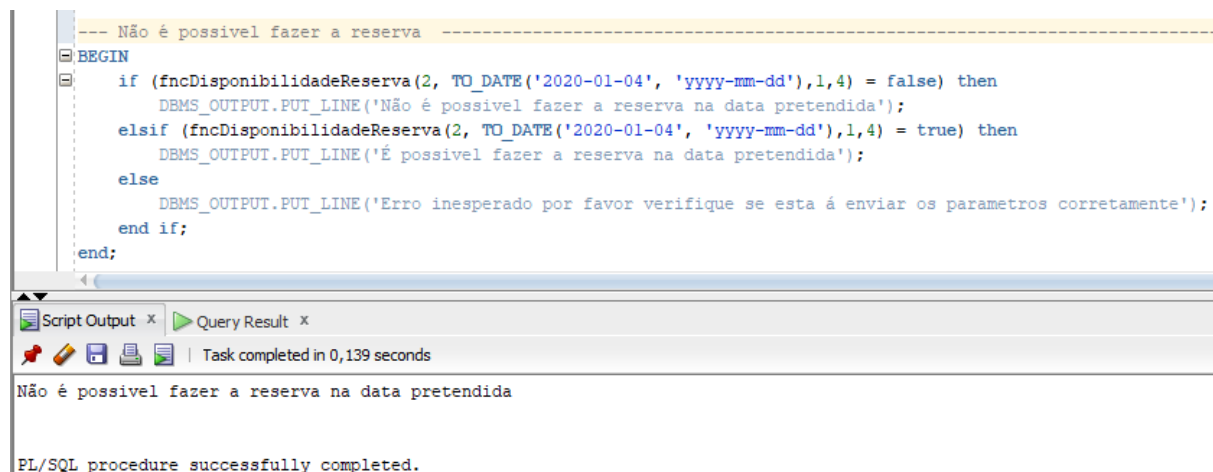
Task completed in 0,126 seconds

Não é possível reservar esse tipo de quarto o numero de pessoas é maior do que a lotação máxima
Não é possível fazer a reserva na data pretendida

PL/SQL procedure successfully completed.

Figura 24 - Exceção numero de pessoas maior que a lotação máxima do tipo de quarto

Caso não aconteça nenhuma exceção retorna false se não é possível fazer a reserva na data pretendida ou true caso seja possível fazer a reserva.



```
----- Não é possível fazer a reserva -----  
BEGIN  
  if (fncDisponibilidadeReserva(2, TO_DATE('2020-01-04', 'yyyy-mm-dd'),1,4) = false) then  
    DBMS_OUTPUT.PUT_LINE('Não é possível fazer a reserva na data pretendida');  
  elsif (fncDisponibilidadeReserva(2, TO_DATE('2020-01-04', 'yyyy-mm-dd'),1,4) = true) then  
    DBMS_OUTPUT.PUT_LINE('É possível fazer a reserva na data pretendida');  
  else  
    DBMS_OUTPUT.PUT_LINE('Erro inesperado por favor verifique se esta á enviar os parametros corretamente');  
  end if;  
end;
```

Script Output x Query Result x

Task completed in 0,139 seconds

Não é possível fazer a reserva na data pretendida

PL/SQL procedure successfully completed.

Figura 25- Não é possível fazer a reserva para a data pretendida

```

-- success -----
BEGIN
if (fncDisponibilidadeReserva(2,TO_DATE('2021-02-04', 'yyyy-mm-dd'),6,4) = false) then
    DBMS_OUTPUT.PUT_LINE('Não é possível fazer a reserva na data pretendida');
elsif (fncDisponibilidadeReserva(2, TO_DATE('2021-02-04', 'yyyy-mm-dd'),6,4) = true) then
    DBMS_OUTPUT.PUT_LINE('É possível fazer a reserva na data pretendida');
else
    DBMS_OUTPUT.PUT_LINE('Erro inesperado');
end if;
end;
```

Script Output x Query Result x

Task completed in 0,599 seconds

É possível fazer a reserva na data pretendida

PL/SQL procedure successfully completed.

Figura 26- É possível fazer a reserva para a data pretendida

Exercício 8)

O objetivo de este exercício era criar um procedimento designado prcRegistrarReserva que permitirá registar uma reserva. O procedimento deve recebe dois tipos de parâmetros de entrada: obrigatórios e opcionais. Os parâmetros obrigatórios são: o tipo de quarto, a data de entrada, a data de saída e o número de pessoas. Os parâmetros opcionais são: o id do cliente, o nome do cliente, o NIF, o telefone e o email.

Caso o id já fosse especificado não pode especificar outros parâmetros se chega a especificar outros parâmetros do cliente alem do id será lançada a exceção nao_especificar_outros , ou se não especificam o nome ou o id do cliente será lançada a exceção especificar_nome_id_cliente , ou no caso de ter especificado o nome é obrigatorio especificar o nif caso o nif não seja especificado lança a exceção especificar_nif :

```

-- Exception vou especificar nome e id (se especificamos o id não pode especificar outro parametro)-----
begin
prcRegistrarReserva(2,TO_DATE('2020-01-04', 'yyyy-mm-dd'),TO_DATE('2020-01-10', 'yyyy-mm-dd'),4,2,'Guilherme');
end;
```

Script Output x Query Result x

Task completed in 0,356 seconds

Se já esta especifico o id do cliente não pode especificar o nome, o NIF, o telefone nem o email .

PL/SQL procedure successfully completed.

Figura 27- Exceção nao_especificar_outros

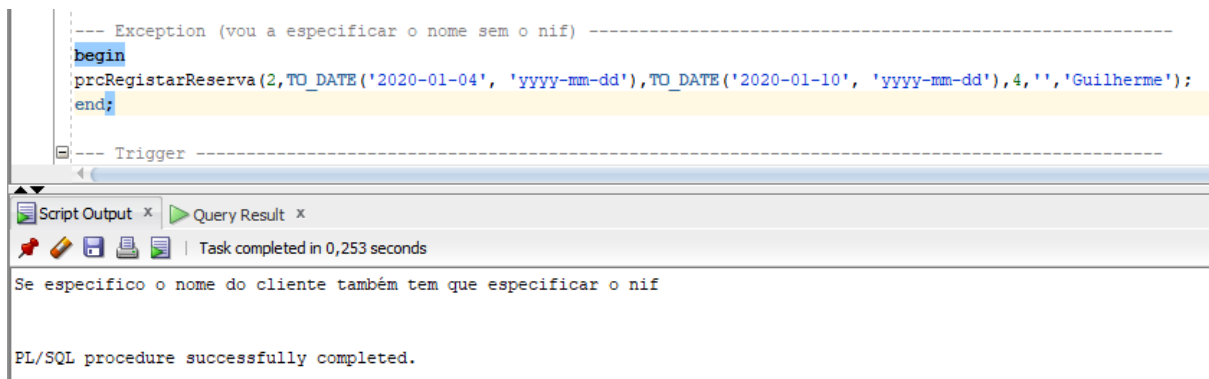


Figura 28- Exceção especificar_nif

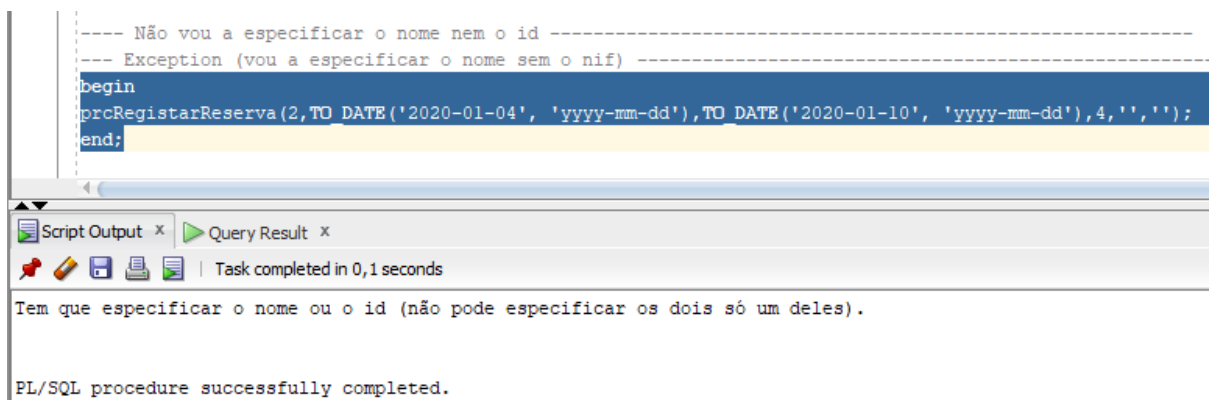


Figura 29- Exceção especificar_nome_id_cliente

Caso não aconteça nenhuma exceção avisa se foi possível ou não registar a reserva na data pretendida.

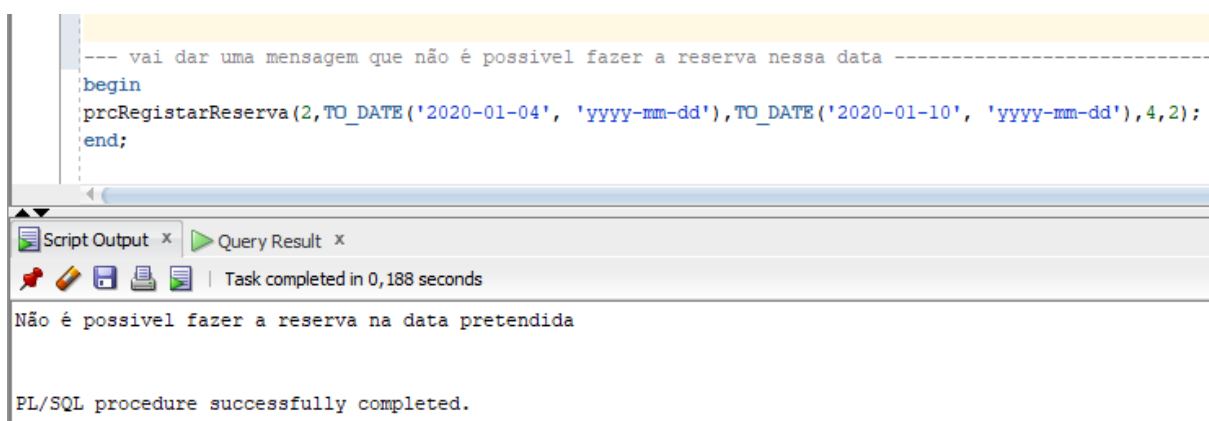


Figura 30- Não é possível registar a reserva na data pretendida

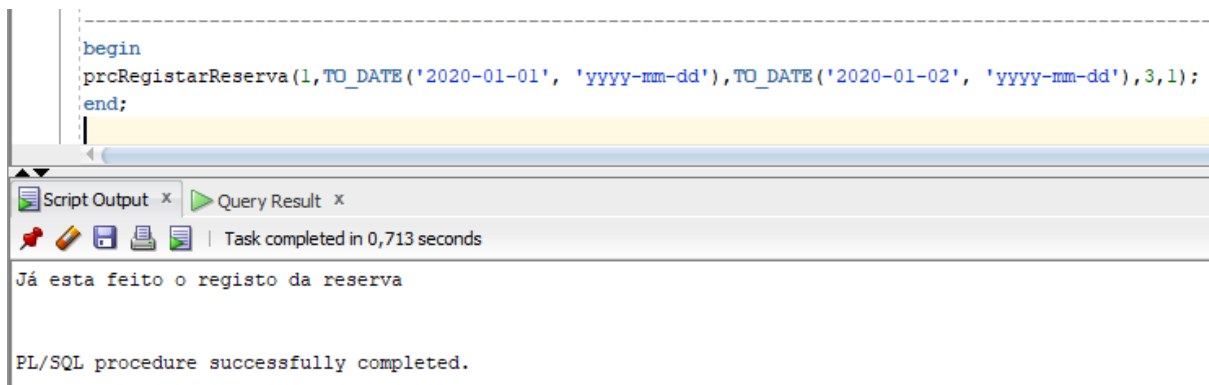


Figura 31- Sucesso no registo da reserva

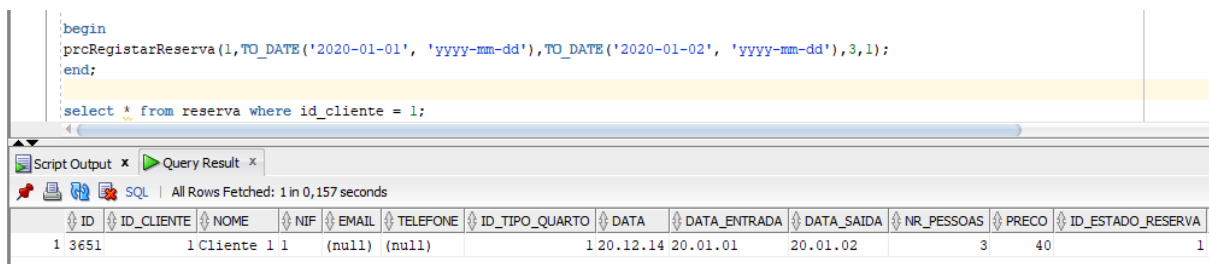


Figura 32- consulta para ver se tinha feito o registo da reserva

Neste caso os dados como o nome , nif , email, telefone já aparecem por causa do trigger implementado no exercício seguinte.

Exercício 9)

O objetivo de este exercicio era criar um trigger designado trgAtualizaCliente, que permitira atualizar a informação do cliente na reserva assim que defina o seu id.

Neste caso temos duas exceções onde uma verifica se esse cliente existe na tabela cliente pelo id que estamos a definir na tabela reserva se não existe acontece a exceção (ClienteNaoExiste) , outra que verifica que o novo id do cliente que estamos a definir não seja null (dado_novo_invalido).

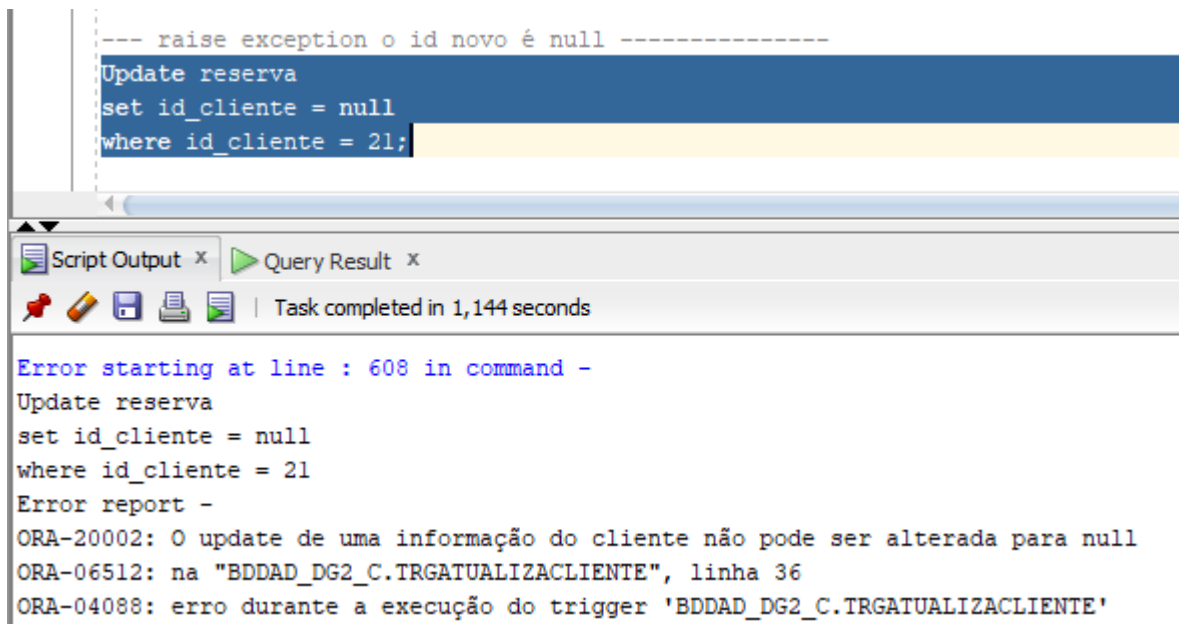


Figura 33- Exeção dado_novo_invalido

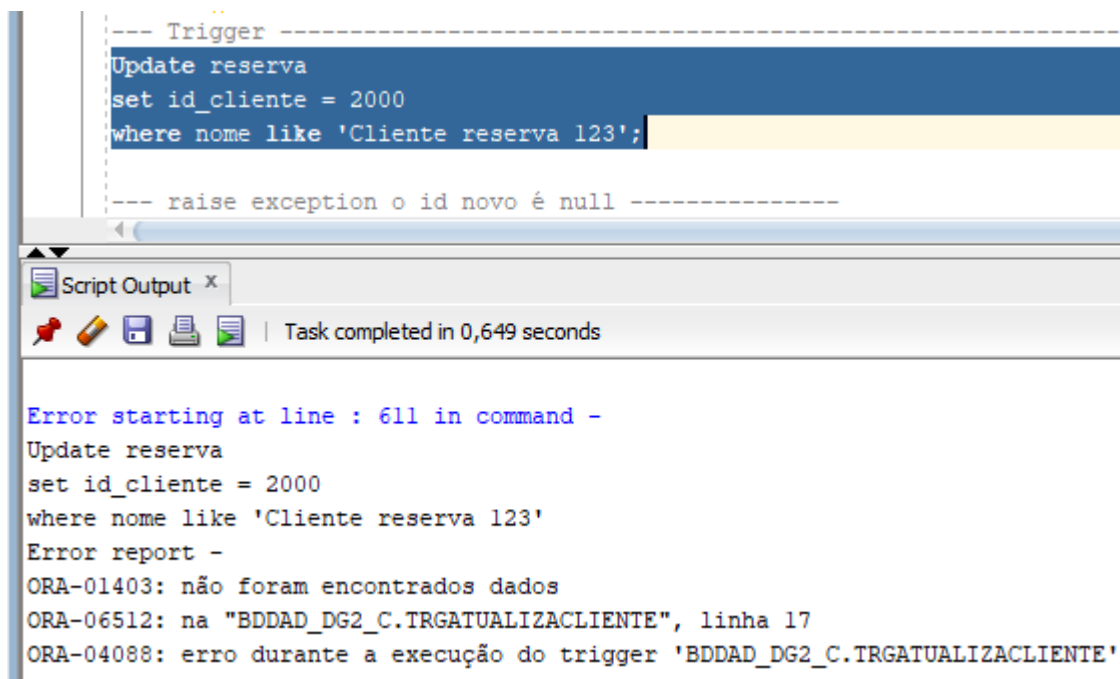


Figura 34- Exceção cliente_nao_existe

Caso contrario atualiza as informações do cliente na tabela reserva segundo a informação que esta na tabela clientes.

