Exercício 1

```
CREATE OR REPLACE FUNCTION Fnc_Get_Quarto_Reserva(V_Id_Reserva Reserva.Id%TYPE) RETURN Quarto.Id%TYPE
    IS
    Dummy               INTEGER;
    V_Id_Estado_Reserva Reserva.Id_Estado_Reserva%TYPE;
    V_Id_Quarto         Quarto.Id%TYPE;
    V_Data_Entrada      Reserva.Data_Entrada%TYPE;
    V_Data_Saida        Reserva.Data_Saida%TYPE;
    Ex_Parametro_Invalido EXCEPTION;
BEGIN
    -- validar parâmetro
    -- a) parâmetro é null
    IF V_Id_Reserva IS NULL THEN
        RAISE Ex_Parametro_Invalido;
    END IF;
    -- b) reserva não existe
    SELECT COUNT(*)
    INTO Dummy
    FROM Reserva
    WHERE Id = V_Id_Reserva;
    IF Dummy = 0 THEN
        RAISE Ex_Parametro_Invalido;
    END IF;
    -- c) reserva já tem quarto associado
    SELECT COUNT(*)
    INTO Dummy
    FROM Checkin
    WHERE Id_Reserva = V_Id_Reserva;
    IF Dummy = 1 THEN
        RAISE Ex_Parametro_Invalido;
    END IF;
    -- d) estado da reserva inválido
    SELECT Id_Estado_Reserva
    INTO V_Id_Estado_Reserva
    FROM Reserva
    WHERE Id = V_Id_Reserva;
    IF V_Id_Estado_Reserva NOT IN (1, 2) THEN
        RAISE Ex_Parametro_Invalido;
    END IF;

    -- ir buscar datas de entrada e saída da reserva pedida
    SELECT Data_Entrada, Data_Saida
    INTO V_Data_Entrada, V_Data_Saida
    FROM Reserva
    WHERE Id = V_Id_Reserva;

    SELECT Q1.Id
    INTO V_Id_Quarto
    FROM Quarto Q1
```

```
    FROM Quarto Q1
    WHERE Q1.Id NOT IN
          ( -- quarto não ocupado na data de reserva
              SELECT Q1.Id
              FROM Quarto Q1
              JOIN Checkin C ON C.Id_Quarto = Q1.Id
              JOIN Reserva R ON R.Id = C.Id_Reserva
              WHERE (V_Data_Saida > R.Data_Entrada -- data de entrada não
                  AND V_Data_Saida < R.Data_Saida) -- pertence ao intervalo
                 OR (V_Data_Entrada > R.Data_Entrada -- e data de saída também
                  AND V_Data_Entrada < R.Data_Saida)
          )
      AND Q1.Id_Tipo_Quarto =
          ( -- tipo de quarto especificado
              SELECT Id_Tipo_Quarto
              FROM Reserva
              WHERE Id = V_Id_Reserva
          )
    ORDER BY Q1.Id_Andar, -- menor andar
             ( -- mínimo de dias de ocupação no último ano
                 SELECT SUM(R2.Data_Saida - R2.Data_Entrada + 1)
                 FROM Quarto Q2,
                      Checkin C2,
                      Reserva R2
                 WHERE R2.Id = C2.Id_Reserva
                   AND Q2.Id = C2.Id_Quarto
                   AND SYSDATE - R2.Data_Saida <= 365
                   AND Q2.Id = Q1.Id
                 GROUP BY Q2.Id
             )
        FETCH FIRST ROW ONLY;

    RETURN V_Id_Quarto;

EXCEPTION
    WHEN Ex_Parametro_Invalido THEN
        RETURN NULL;
END;
```

Pretende provar-se que, quando invocamos a função dando como parâmetro uma reserva sem checkin efetuado, esta vai retornar o quarto do andar mais baixo, com o mínimo de dias de ocupação no último ano, e que não esteja ocupado no período de estada da reserva enviada por parâmetro.

Esta reserva ainda não tem o checkin efetuado:

ID	ID_TIPO_QUARTO	data entrada	data saida
6743	3	28/11/2020	01/12/2020

Todos os quartos do tipo 3, ordenados pelo andar e pelo número de dias de ocupação no último ano:

ID	ID_TIPO_QUARTO	ID_ANDAR	N_DIAS
3	3	1	486
6	3	1	492
9	3	2	480
12	3	2	480
21	3	3	240
18	3	3	441
15	3	3	459

Todos os quartos do tipo 3 ocupados durante o período de estada da reserva 6743:

"id quarto"	"id reserva"	"data entrada"	"data saida"
3	6824	27/11/2020	30/11/2020
6	6851	27/11/2020	30/11/2020
6	6878	26/11/2020	29/11/2020
9	6905	26/11/2020	29/11/2020
12	6770	27/11/2020	30/11/2020
15	6797	27/11/2020	30/11/2020
18	6932	26/11/2020	29/11/2020

Assim, espera-se que a função retorne o valor 21, já que os quartos 3, 6, 9 e 12 (que teriam prioridade de escolha por serem de um andar inferior ou terem tido menos dias de ocupação no último ano) estão ocupados durante o período de estada da reserva 6853. Aqui está um bloco anónimo que demonstra isso mesmo, e também testa as condições em que a função deve retornar NULL:

```
BDDAD_DH2> BEGIN
               Dbms_Output.Put_Line('Quarto para reserva 6743: '
                   || NVL(TO_CHAR(Fnc_Get_Quarto_Reserva(6743)), 'NULL') || '.');
               Dbms_Output.Put_Line('Quarto para reserva NULL: '
                   || NVL(TO_CHAR(Fnc_Get_Quarto_Reserva(NULL)), 'NULL') || '.');
               Dbms_Output.Put_Line('Quarto para reserva 7500 (não existe): '
                   || NVL(TO_CHAR(Fnc_Get_Quarto_Reserva(7500)), 'NULL') || '.');
               Dbms_Output.Put_Line('Quarto para reserva 100 (já tem um quarto associado): '
                   || NVL(TO_CHAR(Fnc_Get_Quarto_Reserva(100)), 'NULL') || '.');
               UPDATE Reserva SET Id_Estado_Reserva = 5 WHERE Id = 71;
               Dbms_Output.Put_Line('Quarto para reserva 71 (estado inválido): '
                   || NVL(TO_CHAR(Fnc_Get_Quarto_Reserva(71)), 'NULL') || '.');
           END;
[2020-12-18 10:15:45] completed in 65 ms
[2020-12-18 10:15:45] Quarto para reserva 6743: 21.
[2020-12-18 10:15:45] Quarto para reserva NULL: NULL.
[2020-12-18 10:15:45] Quarto para reserva 7500 (não existe): NULL.
[2020-12-18 10:15:45] Quarto para reserva 100 (já tem um quarto associado): NULL.
[2020-12-18 10:15:46] Quarto para reserva 71 (estado inválido): NULL.
```