RELATÓRIO TRABALHO PRÁTICO

BASES DE DADOS

Grupo 3 – 2DL

Gonçalo Jordão - 1190633
Guilherme Mendes - 1190641
João Gonçalves - 1190731

Objetivo do trabalho

Este trabalho consistia na elaboração de uma aplicação de base de dados que permitisse o hotel SweetDreams gerir reservas e alojamentos nos seus quartos segundo uma série de requisitos.

Modelo relacional

Assim, elaboramos um modelo relacional, identificando todas as entidades e atributos relativos ao problema. Este encontra-se na terceira forma normal , e tem todas as cardinalidades, bem como chaves e tributos definidos.

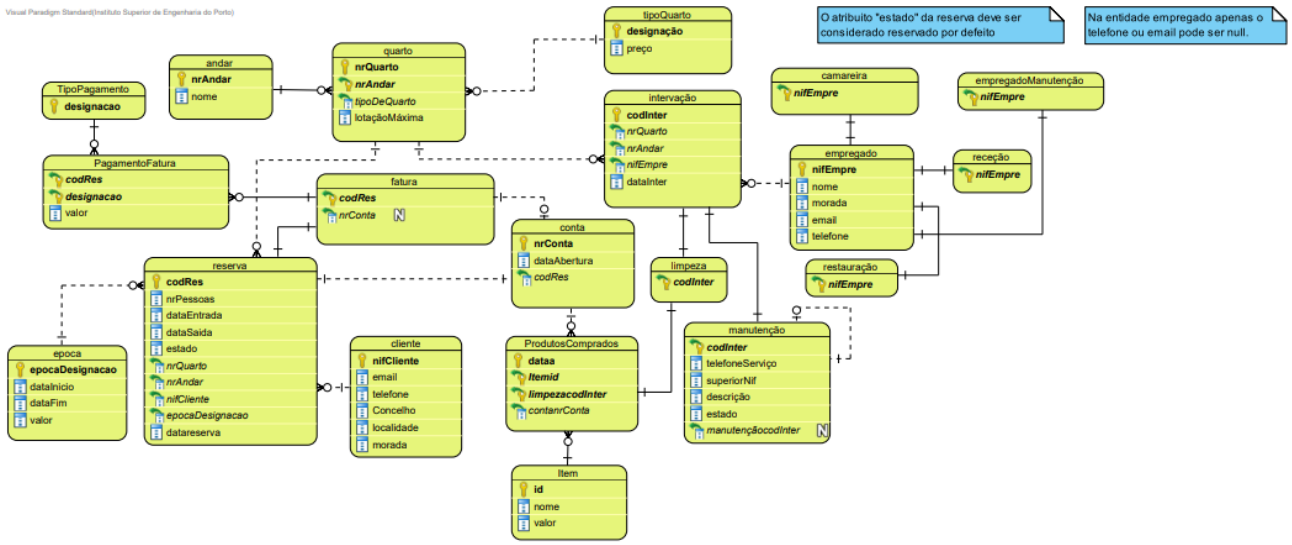

Figura 1 - Modelo Relacional

Para realizar o modelo relacional decidimos criar 20 entidades.

A classe reserva foi criada com o objetivo de armazenar dados relativos a mesma, bem como servir de associação entre outras entidades (por exemplo: quarto, cliente, época, etc.). Esta tem como chave primaria o “codres” que a identifica. Além disso, esta tem uma relação de um para um com a fatura, porque apenas existirá uma fatura por reserva e vice-versa.

O quarto, que pode estar em várias reservas (um para muitos) é identificado pelo seu número e pela chave estrangeira “nrAndar” que juntas formam uma chave composta. Está também associada a um tipo de quarto que contém a sua designação e o preço relativo.

O cliente, associado á reserva, guarda todas as suas informações.

Passando agora parta a gestão de pagamentos, resolvemos criar a classe “TipoPagamento” que apenas guarda a sua designação, estando associada a uma fatura, previamente descrita, através da classe “PagamentoFatura”, criada para manter a terceira forma normal. A fatura relaciona-se com a conta numa relação um para um, contendo a data de abertura, o seu número e a chave estrangeira código da reserva para ser possível efetuar o pagamento de acordo com a estadia. Por conseguinte, a classe “ProdutosComprados” permite-nos manter a terceira forma normal associando a conta e a entidade “Item”.

Na gestão de funcionários, criamos entidades para “empregadoManutenção”, “receção” e “restauração” de modo a identificar os vários tipos de empregados. A última, guarda as informações relativas ao mesmo, tendo como chave primária o seu NIF.

Finalmente, a classe intervenção é responsável por conter os registos das intervenções, guardando o local onde é efetuada e quem o realizou, bem com a data. A entidade limpeza e manutenção associam-se á intervenção , tendo a chave estrangeira do código da mesma, armazenando também outras informações importantes.

Consulta – Parte I – João Gonçalves

Na primeira alínea da parte 1 era pedido que, utilizando comandos SQL, apresentássemos todos os pedidos de intervenção em aberto alocados a funcionários que não fizeram nenhuma intervenção nas últimas 24 horas. Para resolver este problema, comecei por identificar a informação a apresentar:

```
SELECT
    m.codinter,
    m.estado,
    i.nrandar,
    i.nrquarto,
    i.nifempre,
    i.datainter
FROM
      manutencao m
```

Em seguida, necessitava de intersetar estes dados com a tabela intervenção, para conseguir ter acesso á data de intervenção, bem como ao seu estado. Para tal utilizei um inner join na condição de ter o mesmo código de intervenção.

```
    INNER JOIN intervencao i ON m.codinter = i.codinter
```

No próximo passo, na clausula *where* faço a validação para que não sejam mostradas intervenções futuras. Além disto, é necessário verificar os funcionários que não fizeram intervenções nas ultimas 48 horas, e que o estado seja aberto. Para tal, utilizei uma *subquerry* em que seleciono todos os empregados das intervenções e verifico se a data de intervenção esta no intervalo especificado, e o seu estado é “aberto”

```
WHERE
    ( i.datainter <= sysdate )
    AND i.nifempre NOT IN (
        SELECT DISTINCT
            nifempre
        FROM
            intervencao ii
        WHERE
                ii.datainter >= ( sysdate - ( 72 / 24 ) )
            AND estado LIKE ( 'aberto' )
            AND ( ii.datainter <= sysdate )
    )
```

Por fim, resta-me ordenar a informação selecionada por ordem crescente de data e, nos casos iguais, ordenar pelo sei código.

```
ORDER BY
    i.datainter,
    i.codinter DESC;
```

Resultado final:

```
SELECT
    m.codinter,
    m.estado,
    i.nrandar,
    i.nrquarto,
    i.nifempre,
    i.datainter
FROM
         manutencao m
    INNER JOIN intervencao i ON m.codinter = i.codinter
WHERE
    ( i.datainter <= sysdate )
    AND i.nifempre NOT IN (
        SELECT DISTINCT
            nifempre
        FROM
            intervencao ii
        WHERE
                ii.datainter >= ( sysdate - ( 72 / 24 ) )
            AND estado LIKE ( 'aberto' )
            AND ( ii.datainter <= sysdate )
    )
ORDER BY
    i.datainter,
    i.codinter;
```

Resultado (Realizado as 11:00h do dia 20 de Novembro 2020):

CODINTER	ESTADO	NRANDAR	NRQUARTO	NIFEMPRE	DATAINTER
1	aberto	1	1	111111111	2020-01-14 00:00:00
2	aberto	1	1	111111111	2020-01-15 00:00:00
3	aberto	1	1	111111111	2020-11-17 00:00:00

Por conseguinte, na segunda alínea, era pedido que fosse apresentada a data, a hora e o nome dos clientes que reservaram quartos nos meses de abril e junho deste ano. Além disso no caso de ter sido reservado um quarto tipo suite, deverá ser apresentada a localidade desse cliente numa coluna “zona do país”, por ordem alfabética e decrescente da data e hora da reserva.

Para resolver este exercício comecei por selecionar a informação pretendida, utilizando a função “to_char” para obter a hora da reserva. Além disso, foi necessário um Case onde é feita a verificação do tipo de quarto, para só assim ser impressa a zona do país.

```
SELECT
    cliente.nome,
    to_char(reserva.datareserva, 'HH24:MI:SS')        AS hora,
    reserva.dataentrada,
    CASE
        WHEN ( quarto.tipoquarto LIKE ( 'suite' ) ) THEN
            cliente.morada
        ELSE
            ' '
    END
```

De seguida, especifiquei a origem dos dados com a clausula From, fazendo dois inner joins consecutivos, pois necessitava da intersecção da informação relativa aos quartos e relativa a reserva. Foi utilizada a condição em que o nif do cliente era igual, bem como referiam-se ao mesmo andar e quarto.

```
FROM
         cliente
    INNER JOIN reserva ON reserva.nifcliente = cliente.nifcliente
    INNER JOIN quarto ON ( reserva.nrquarto = quarto.nrquarto
                           AND reserva.nrandar = quarto.nrandar )
```

Por fim, apenas restava fazer a validação da data da reserva, utilizando a função Extract da data, e ordenar por ordem crescente em relação ao nome, e no caso de ser igual, por ordem decrescente da data de reserva.

```
WHERE
    (EXTRACT(MONTH FROM reserva.datareserva) = 4 AND EXTRACT(YEAR FROM
reserva.datareserva)= EXTRACT(YEAR FROM sysdate))or
    (EXTRACT(MONTH FROM reserva.datareserva) =6 AND EXTRACT(YEAR FROM
reserva.datareserva)= EXTRACT(YEAR FROM sysdate))
```

Resultado final:

```
SELECT
    cliente.nome,
    to_char(reserva.datareserva, 'HH24:MI:SS')        AS hora,
    reserva.dataentrada,
    CASE
        WHEN ( quarto.tipoquarto LIKE ( 'suite' ) ) THEN
            cliente.morada
        ELSE
            ' '
    END
AS zona_pais
FROM
         cliente
    INNER JOIN reserva ON reserva.nifcliente = cliente.nifcliente
    INNER JOIN quarto ON ( reserva.nrquarto = quarto.nrquarto
                           AND reserva.nrandar = quarto.nrandar )
WHERE
    (EXTRACT(MONTH FROM reserva.datareserva) = 4 AND EXTRACT(YEAR FROM
reserva.datareserva)= EXTRACT(YEAR FROM sysdate))or
    (EXTRACT(MONTH FROM reserva.datareserva) =6 AND EXTRACT(YEAR FROM
reserva.datareserva)= EXTRACT(YEAR FROM sysdate))
ORDER BY
    cliente.nome ASC,
    reserva.datareserva DESC;
```

Resultado (Realizado as 11:00h do dia 20 de Novembro 2020):

NOME	HORA	DATAENTRADA	ZONA_PAIS
Duarte Veiga	01:00:00	2020-04-03 00:00:00	Coimbra
Emanuela Santos	01:00:00	2020-06-15 00:00:00	Lisboa
Madalena Pinto	01:00:00	2020-04-24 00:00:00	Aveiro

Consulta – Parte II – Guilherme Mendes

Na primeira alínea da parte 2 era necessário apresentar o nome, a localidade e o nome do concelho dos clientes que já estiveram alojados nos quartos já reservados pelo cliente cujo nome é José Silva, que é do concelho de Vila nas suas reservas finalizadas.
Para alcançar este objetivo começamos por fazer dum select da informação necessária.

```
SELECT c1.nome
        ,c1.localidade
        ,c1.concelho
FROM cliente c1
```

A seguir fazemos um inner join da reserva aonde igualamos o nif do cliente da tabela da reserva com o nif do cliente da tabela cliente. De seguida limitamos o cliente aonde o nome é diferente de José Silva.

```
INNER JOIN reserva r1 ON r1.nifcliente = c1.nifcliente
WHERE c1.nome NOT LIKE 'José Silva'
```

E depois vemos se o número de quarto e o número de andar encontram-se na subquerry que indica o número de quarto e o número de andar aonde existe uma reserva para esse quarto para o cliente cujo nome é José Silva e o estado é finalizada

```
AND (
                r1.nrquarto
                ,r1.nrandar
                ) IN (
                (
                        SELECT r2.nrquarto
                                ,r2.nrandar
                        FROM reserva r2
                        INNER JOIN cliente c2 ON r2.nifcliente = c2.nifcliente
                        WHERE c2.nome LIKE 'José Silva'
                                AND r2.estado = 'finalizada'
                        )
                );
```

Resultado final:

```
SELECT c1.nome
        ,c1.localidade
        ,c1.concelho
FROM cliente c1
INNER JOIN reserva r1 ON r1.nifcliente = c1.nifcliente
WHERE c1.nome NOT LIKE 'José Silva'
        AND (
                r1.nrquarto
                ,r1.nrandar
                ) IN (
                (
                        SELECT r2.nrquarto
                                ,r2.nrandar
                        FROM reserva r2
                        INNER JOIN cliente c2 ON r2.nifcliente = c2.nifcliente
                        WHERE c2.nome LIKE 'José Silva'
```

```
                        AND r2.estado = 'finalizada'
                )
        );
```

Resultado:

1	Jorge Miguel	Portalegre	Portalegre
2	Maria Maia	Setubal	Setubal
3	Sara Oliveira	Faro	Faro
4	Jorge Miguel	Portalegre	Portalegre
5	Maria Maia	Setubal	Setubal

Para o exercicio 2 foi requerido apresentar por cada mês para os ultimos 6 meses anteriores à data que o select é corrido, a camareira que fez mais intervenções em quartos cuja estadia foi superior à media por tipo de quarto.

Para tal, foi necessário escolher a informação a demonstrar e para isso escolhemos apresentar o mês em questão, o nif da camareira e o respetivo nome.

```
SELECT (EXTRACT(MONTH FROM r.dataentrada)) AS mes
        ,E.nifempre AS nif
        ,e.nome
FROM reserva r
INNER JOIN intervencao i ON (
                r.nrquarto = i.nrquarto
                AND r.nrandar = i.nrandar
                AND i.datainter BETWEEN r.dataentrada
                        AND r.datasaida
                )
INNER JOIN limpeza ON i.codinter = limpeza.codinter
INNER JOIN EMPREGADO E ON E.NIFEMPRE = I.NIFEMPRE
INNER JOIN camareira c ON i.nifempre = c.nifempre
```

Os inner join associam a intervenção à reserva e certificam-se que a intervenção é de limpeza e o empregado é uma camareira.

De seguida, o sql verifica se o mês da reserva encontra-se nos últimos 6 meses da data de execução através dum where. Depois ele faz a count do número de intervenções associados a reservas que a diferença da data de saída e da data de entadas são superiores à média por tipo de quarto. Para ele conseguir a média por tipo de quarto é utilizada uma subquerry que devolve uma tabela com a média por tipo de quarto e por mês para todos os tipos de quarto para todos os meses através dum group by. Depois utiliza um where para ter a média para ter o tipo de quarto e a o mês para ser igual à reserva r1 fora da querry. De seguida ele compara a ver se o valor deste count é igual ao fazer o all da querry que devolve o número das intervenções por camareira.

```
WHERE (EXTRACT(MONTH FROM r.dataentrada) >= (EXTRACT(MONTH FROM sysdate) - 6))
        AND (
                SELECT COUNT(*)
```

```
                FROM intervencao i2
                INNER JOIN reserva r2 ON (
                                r2.nrquarto = i2.nrquarto
                                AND r2.nrandar = i2.nrandar
                                AND i2.datainter BETWEEN r2.dataentrada
                                        AND r2.datasaida
                                )
                WHERE i.nifempre = i2.nifempre
                        AND (r2.codres) IN (
                                SELECT r1.codres
                                FROM reserva r1
                                INNER JOIN quarto q1 ON (
                                                q1.nrandar = r1.nrandar
                                                AND q1.nrquarto = r1.nrquarto
                                                )
                                INNER JOIN (
                                        SELECT q2.tipoquarto tipo
                                                ,AVG(r2.datasaida - r2.dataentrada)
media
                                                ,(EXTRACT(MONTH FROM
r2.dataentrada)) mes
                                        FROM quarto q2
                                        INNER JOIN reserva r2 ON (
                                                        r2.nrandar = q2.nrandar
                                                        AND r2.nrquarto = q2.nrquarto
                                                        )
                                        GROUP BY q2.tipoquarto
                                                ,(EXTRACT(MONTH FROM
r2.dataentrada))
                                        ) quer ON (
                                                quer.tipo = q1.tipoquarto
                                                AND quer.mes = EXTRACT(MONTH FROM
r1.dataentrada)
                                                )
                                WHERE (EXTRACT(MONTH FROM r.dataentrada)) =
(EXTRACT(MONTH FROM r1.dataentrada))
                                        AND ((r1.datasaida - r1.dataentrada) >
quer.media)
                                )
                ) >= ALL (
                SELECT COUNT(i3.codinter)
                FROM intervencao i3
                INNER JOIN reserva r3 ON i3.nrquarto = r3.nrquarto
                        AND i3.nrandar = r3.nrandar
                INNER JOIN limpeza l1 ON i3.codinter = l1.codinter
                INNER JOIN quarto q3 ON (
                                q3.nrquarto = r3.nrquarto
                                AND q3.nrandar = r3.nrandar
                                )
                INNER JOIN (
                        SELECT q2.tipoquarto tipo
                                ,AVG(r2.datasaida - r2.dataentrada) media
                                ,(EXTRACT(MONTH FROM r2.dataentrada)) mes
                        FROM quarto q2
                        INNER JOIN reserva r2 ON (
                                        r2.nrandar = q2.nrandar
                                        AND r2.nrquarto = q2.nrquarto
                                        )
                        GROUP BY q2.tipoquarto
                                ,(EXTRACT(MONTH FROM r2.dataentrada))
```

```
                        ) quer ON (
                                quer.tipo = q3.tipoquarto
                                AND quer.mes = EXTRACT(MONTH FROM r3.dataentrada)
                                )
                WHERE ((r3.datasaida - r3.dataentrada) > quer.media)
                        AND i3.datainter BETWEEN r3.dataentrada
                                AND r3.datasaida
                        AND (EXTRACT(MONTH FROM i3.datainter)) = (EXTRACT(MONTH
FROM i.datainter))
                GROUP BY i3.nifempre
                        ,i3.nrquarto
                        ,i3.nrandar
                )
GROUP BY (EXTRACT(MONTH FROM r.dataentrada))
        ,E.nifempre
        ,E.NOME
ORDER BY mes;
```

No final faz-se um group by com o mês, o nif do empregado e o nome e ordena por mês

Resultado final :

```
SELECT (EXTRACT(MONTH FROM r.dataentrada)) AS mes
        ,E.nifempre AS nif
        ,e.nome
FROM reserva r
INNER JOIN intervencao i ON (
                r.nrquarto = i.nrquarto
                AND r.nrandar = i.nrandar
                AND i.datainter BETWEEN r.dataentrada
                        AND r.datasaida
                )
INNER JOIN limpeza ON i.codinter = limpeza.codinter
INNER JOIN EMPREGADO E ON E.NIFEMPRE = I.NIFEMPRE
INNER JOIN camareira c ON i.nifempre = c.nifempre
WHERE (EXTRACT(MONTH FROM r.dataentrada) >= (EXTRACT(MONTH FROM sysdate) - 6))
        AND (
                SELECT COUNT(*)
                FROM intervencao i2
                INNER JOIN reserva r2 ON (
                                r2.nrquarto = i2.nrquarto
                                AND r2.nrandar = i2.nrandar
                                AND i2.datainter BETWEEN r2.dataentrada
                                        AND r2.datasaida
                                )
                WHERE i.nifempre = i2.nifempre
                        AND (r2.codres) IN (
                                SELECT r1.codres
                                FROM reserva r1
                                INNER JOIN quarto q1 ON (
                                                q1.nrandar = r1.nrandar
                                                AND q1.nrquarto = r1.nrquarto
                                                )
                                INNER JOIN (
                                        SELECT q2.tipoquarto tipo
                                                ,AVG(r2.datasaida - r2.dataentrada)
media
                                                ,(EXTRACT(MONTH FROM
r2.dataentrada)) mes
```

```
                                        FROM quarto q2
                                        INNER JOIN reserva r2 ON (
                                                        r2.nrandar = q2.nrandar
                                                        AND r2.nrquarto = q2.nrquarto
                                                        )
                                        GROUP BY q2.tipoquarto
                                                ,(EXTRACT(MONTH FROM
r2.dataentrada))
                                        ) quer ON (
                                                quer.tipo = q1.tipoquarto
                                                AND quer.mes = EXTRACT(MONTH FROM
r1.dataentrada)
                                                )
                                WHERE (EXTRACT(MONTH FROM r.dataentrada)) =
(EXTRACT(MONTH FROM r1.dataentrada))
                                        AND ((r1.datasaida - r1.dataentrada) >
quer.media)
                                )
                ) >= ALL (
                SELECT COUNT(i3.codinter)
                FROM intervencao i3
                INNER JOIN reserva r3 ON i3.nrquarto = r3.nrquarto
                        AND i3.nrandar = r3.nrandar
                INNER JOIN limpeza l1 ON i3.codinter = l1.codinter
                INNER JOIN quarto q3 ON (
                                q3.nrquarto = r3.nrquarto
                                AND q3.nrandar = r3.nrandar
                                )
                INNER JOIN (
                        SELECT q2.tipoquarto tipo
                                ,AVG(r2.datasaida - r2.dataentrada) media
                                ,(EXTRACT(MONTH FROM r2.dataentrada)) mes
                        FROM quarto q2
                        INNER JOIN reserva r2 ON (
                                        r2.nrandar = q2.nrandar
                                        AND r2.nrquarto = q2.nrquarto
                                        )
                        GROUP BY q2.tipoquarto
                                ,(EXTRACT(MONTH FROM r2.dataentrada))
                        ) quer ON (
                                quer.tipo = q3.tipoquarto
                                AND quer.mes = EXTRACT(MONTH FROM r3.dataentrada)
                                )
                WHERE ((r3.datasaida - r3.dataentrada) > quer.media)
                        AND i3.datainter BETWEEN r3.dataentrada
                                AND r3.datasaida
                        AND (EXTRACT(MONTH FROM i3.datainter)) = (EXTRACT(MONTH
FROM i.datainter))
                GROUP BY i3.nifempre
                        ,i3.nrquarto
                        ,i3.nrandar
                )
GROUP BY (EXTRACT(MONTH FROM r.dataentrada))
        ,E.nifempre
        ,E.NOME
ORDER BY mes;
```

Resultado:

	MES	NIF	NOME
1	5	555555555	Maria Julia de Jesus Santos
2	6	222222222	Maria Adelaide dos Santos Bandeira
3	6	777777777	Vanessa dos Alves Maia
4	6	888888888	Manuel Ribeiro Santos
5	7	777777777	Vanessa dos Alves Maia
6	8	888888888	Manuel Ribeiro Santos

Consulta – Parte III –Gonçalo Jordão

A) Apresentar por andar, o quarto e o tipo de quarto, que teve o maior número de reservas. Deverão ser excluídos todos os quartos em que o número de reservas é inferior a 2 e são do tipo “single”. Não incluir reservas canceladas.

```
SELECT reserva.nrandar
      ,reserva.nrquarto
      ,quarto.tipoquarto
FROM reserva
```

- A parte do *script* que se encontra em cima serve para apresentar a informação necessária (“por andar, o quarto e o tipo de quarto”).

```
INNER JOIN quarto ON quarto.nrquarto = reserva.nrquarto
                  AND quarto.nrandar = reserva.nrandar
```

- O *inner join* vai ser necessário para realizar a ligação entre a tabela *reserva* e tabela *quarto*, através do *número de andar* e o *número do quarto*, de maneira a ser possível apresentar o *tipo de quarto*, existente na tabela *quarto*.

```
WHERE upper(reserva.estado) NOT LIKE 'CANCELADA'
```

- Esta primeira validação é relativa ao *estado* da *reserva*, pois não podem ser consideradas reservas *canceladas* (“Não incluir reservas canceladas”).

```
AND upper(quarto.tipoquarto) NOT LIKE 'SINGLE'
```

- Já esta segunda validação é relativa ao *tipo de quarto*, porque não podem ser considerados os quartos *singles* (“… excluídos todos os quartos … são do tipo “single””).

```
GROUP BY reserva.nrandar
        ,reserva.nrquarto
        ,quarto.tipoquarto
```

- Foi utilizado o *group by* de maneira a que no passo seguinte fosse possível comparar o contador da maneira necessária, para o problema em questão, e fazer também as respetivas verificações.

```
HAVING COUNT(*) = (
        SELECT MAX(COUNT(*))
        FROM reserva r
        WHERE reserva.nrandar = r.nrandar
                AND upper(r.estado) NOT LIKE 'CANCELADA'
        GROUP BY nrandar
                ,nrquarto
        )
```

- Este contador vai verificar se a quantidade de reservas de cada quarto é igual ao máximo de reservas (*MAX(COUNT(*))*) que houve para um quarto no andar (*r.nrandar*) em que este mesmo se encontra.

```
AND COUNT(*) >= 2
```

- O contador definido anteriormente além de ter de ser igual ao máximo de reservas que houve para um quarto no andar em que este mesmo se encontra, também tem de ser superior a dois ("Deverão ser excluídos todos os quartos em que o número de reservas é inferior a 2 …").

```
ORDER BY reserva.nrandar
        ,reserva.nrquarto;
```

- A ordenação é feita através do *andar* ("Apresentar por andar") e através do *número do quarto* caso existam dois quartos ou mais por andar com o número de reservas igual ao máximo de reservas num quarto desse andar.

	NRANDAR	NRQUARTO	TIPOQUARTO
1	1	3	suite
2	2	3	suite
3	3	2	suite

B) Apresentar os clientes que ocuparam quartos do tipo suite na última época alta e consumiram os dois produtos com maior consumo nos últimos dois anos. O resultado deve ser ordenado por ordem decrescente do valor total do consumo.

```
SELECT cliente.nome
        ,(
                SELECT SUM(valor)
                FROM item
                INNER JOIN produtoscomprados ON item.itemid =
produtoscomprados.itemid
                INNER JOIN conta ON produtoscomprados.contanrconta = conta.nrconta
                INNER JOIN reserva ON conta.codres = reserva.codres
                INNER JOIN cliente c1 ON reserva.nifcliente = c1.nifcliente
                WHERE c1.nome = cliente.nome
                ) AS valor_total_consumo
```

- Esta *subquery* é necessária para ir buscar o total de consumo de cada cliente. Através de *inner join* encadeados, fazendo as ligações entre as tabelas necessárias (*item*, *produtoscomprados*, *conta*, *reserva* e *cliente*) , podemos aceder aos produtos consumidos pelo cliente (*c1.nome = cliente.nome*) e ao seu valor correspondente, calculando assim o consumo total.

```
FROM Cliente
```

- A parte do *script* que se encontra em cima serve para apresentar a informação necessária ("Apresentar os clientes" e "ordenado por ordem decrescente do valor total do consumo").

```
INNER JOIN reserva ON cliente.nifcliente = reserva.nifcliente
INNER JOIN quarto ON quarto.nrandar = reserva.nrandar
                AND quarto.nrquarto = reserva.nrquarto
```

- Estes *inner join* vão ser necessários para aceder ao *tipo de quarto* e à *data da reserva.*

```
WHERE upper(reserva.epocadesignacao) LIKE 'ALTA'
```

- Esta primeira validação é relativa à *época* da *reserva*, pois só podem ser consideradas reservas *na época alta* ("época alta").

```
AND EXTRACT(YEAR FROM reserva.dataentrada) = (
                SELECT MAX(EXTRACT(YEAR FROM dataentrada))
                FROM reserva
                WHERE upper(epocadesignacao) LIKE 'ALTA'
                )
```

- Para complementar a verificação anterior temos de verificar também se esta se encontra na última época alta ("na última época alta"), vendo assim o ano da última época alta existente na tabela *reserva* ("MAX(EXTRACT(YEAR FROM dataentrada))") para comparar com o ano da reserva em questão.

```
AND upper(reserva.estado) LIKE 'FINALIZADA'
```

- Já esta segunda validação é relativa ao *estado da reserva*, porque só podem ser considerados reservas finalizadas ("clientes que ocuparam quartos").

```
    AND upper(quarto.tipoquarto) LIKE 'SUITE'
```

- No que diz respeito à terceira validação, esta serve para verificar se o tipo de quarto é suite (“quartos do tipo suite”).

```
AND cliente.nifcliente IN (
                SELECT reserva.nifcliente
                FROM reserva
                INNER JOIN conta ON reserva.codres = conta.codres
                INNER JOIN produtoscomprados p1 ON conta.nrconta = p1.contanrconta
                WHERE (
                                SELECT COUNT(DISTINCT p2.itemid)
                                FROM produtoscomprados p2
                                WHERE p1.contanrconta = p2.contanrconta
                                        AND p2.itemid IN (
                                                SELECT p3.itemid
                                                FROM produtoscomprados p3
                                                WHERE EXTRACT(YEAR FROM dataa) =
EXTRACT(YEAR FROM sysdate)
                                                        OR EXTRACT(YEAR FROM dataa) =
((EXTRACT(YEAR FROM sysdate)) - 1)
                                                GROUP BY p3.itemid
                                                ORDER BY COUNT(*) DESC FETCH FIRST 2
ROWS ONLY
                                                )
```

- Neste *select* vão ser contadas as quantidades de consumos de cada item existente nos últimos 2 anos, sendo depois ordenadas por esse contador para ficarem ordenados por ordem decrescente de consumo. Assim, aplicando o *fetch* ficamos com o top 2 de produtos mais consumidos como queríamos.

```
                ) = 2
```

- Nesta subquery vão ser guardados os NIF de cada cliente que consumiram os dois produtos que estão na lista dos 2 produtos mais comprados nos últimos dois anos como era pedido (“consumiram os dois produtos com maior consumo nos últimos dois anos”).

```
        )
```

- Por último decidi optar por utilizar o comando *in* para verificar se o NIF do cliente se encontrava numa lista de NIF de clientes que compraram os 2 produtos mais comprados nos últimos 2 anos.

```
ORDER BY valor_total_consumo DESC;
```

- A ordenação é feita através do *valor de consumo* (“resultado deve ser ordenado por ordem decrescente do valor total do consumo”).

	NOME	VALOR_TOTAL_CONSUMO
1	Emaneul Santos	11,1
2	Felipe Jesus	8,1

Conclusão

Com este trabalho conseguimos juntar o nosso conhecimento de realização de bases de dados relacionais com o de programação em Oracle SQL de uma maneira mais semelhante a um ambiente de trabalho. Uma das maiores dificuldades que tivemos durante a realização deste trabalho associou-se a à instabilidade da conexão, o que nos dificultou a testagem.