

Project Report

BDDAD

**Class 2DE**

1170500 \_ Hugo Frias

1180782 \_ Diogo Ribeiro

1181628 \_ André Novo

**Teachers/Advisors**

**Course Unit**

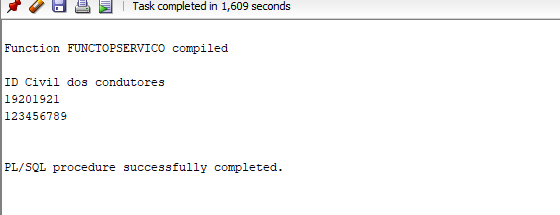
BDDAD

**Exercício 1**

**Enunciado: “**Implemente uma função, denominada funcTopServico, que retorne um cursor que contenha, para um dado tipo de serviço e período, o id dos n condutores com os maiores valores de custos totais das viagens. A função recebe como parâmetros o tipo de serviço, o período e o n. Caso um dos parâmetros seja inválido, a função deve retornar o valor NULL.”.

**Interpretação:** Neste exercício fomos buscar os ids dos N condutores com maiores valores de custo de viagens, que fizessem um determinado tipo de serviço (passado por parâmetro) e cujas data\_hora\_recolha\_passageiro estivessem entre 2 datas passadas novamente por parâmetro.

**Output:**



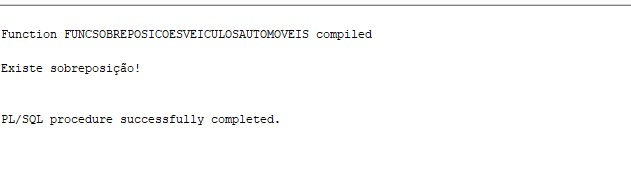
**Problemas:** Nenhum encontrado.

**Exercício 2**

**Enunciado: “**Implemente uma função, denominada funcSobreposicoesVeiculosCondutores, que retorne um valor booleano no caso de existirem sobreposições temporais nas associações de veículos a condutores.”.

**Interpretacao:** Para realizarmos o exercício, criámos uma função que retorna um valor boolean. Esse valor será True se existirem sobreposições temporais nos veículos condutores, e False se não existirem. Para a construção da função, recorremos ao uso de dois ciclos for encadeados, sendo que cada um deles irá percorrer uma query com todos os dados relativos aos veículos condutores presentes na base de dados. Para cada um dos dados, ele irá percorrer os restantes e verificar se as datas de início e fim se encontram no mesmo período. Isto é, se existe sobreposição entre elas. Esta verificação só será feita caso as matriculas ou o número de identificação civil entre os veículos condutores a serem comparados sejam os mesmos. Porque, de acordo com a nossa interpretação, não podem existir dois veículos iguais ou duas pessoas iguais em dois sítios diferentes ao mesmo tempo.

**Output:**



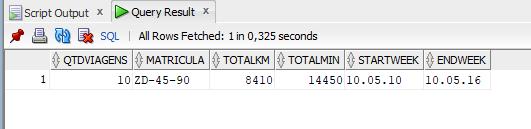
**Problemas:** Nenhum encontrado.

**Exercício 3**

**Enunciado: “**Implemente uma função, denominada funcObterInfoSemanalVeiculos, que retorna um cursor com informação relativa a todos os veículos. A função recebe como parâmetro um valor de uma data e retorna um cursor com a seguinte informação: matrícula, data de início, data de fim, número de viagens, número de quilómetros e tempo gasto em viagens. Os valores de número de viagens, número de quilómetros e tempo gasto em viagens são relativos à semana, de segunda a domingo, à qual a data recebida por parâmetro pertence, ou seja, está compreendida entre os valores de data de início e data de fim.”.

**Interpretação:** Para realizarmos este exercício, nos suposemos que a função deveria retornar toda a informação requesitada pelo enunciado em que a seguinte condição se verificasse: o atributo data\_inicio desse veiculo\_condutor deveria ser maior ou igual ao inicio da semana (segunda) da data que entrasse por parâmetro, já o atributo data\_fim desse mesmo veiculo\_condutor deveria ser menor ou igual ao final da semana (domingo) da data que entrasse por parâmetro.

**Output:**



**Problemas:** A função considera qualquer data passada por parâmetro como sendo inválida.

**Exercício 4**

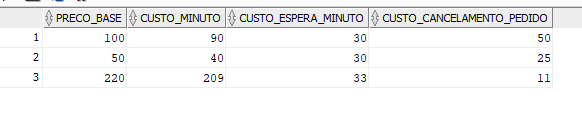
**Enunciado:** “Implemente um procedimento denominado procAtualizarCustosServico, que permita atualizar em X% o valor do preço\_base, do custo\_minuto, do custo\_espera e do custo\_cancelamento do serviço que tenha tido o volume de negócio mais elevado. O procedimento recebe como parâmetros de entrada uma data para atualização e o valor da percentagem a atualizar (X%). Caso não seja passada nenhuma data para atualização, o procedimento deve considerar a data do sistema. Se se entender necessário podem ser realizadas alterações ao modelo relacional.”.

**Interpretação:** Neste exercício procuramos pelo serviço com maior quantidade de pedidos de viagens nos últimos 12 meses e, caso não tenha sido updated nos 6 meses anteriores á data passada por parametro, demos update aos valores do valor do preço\_base, do custo\_minuto, do custo\_espera e do custo\_cancelamento.

**Output:**

Antes: 

Depois:



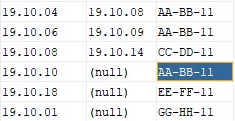
**Problemas:** Nenhum encontrado.

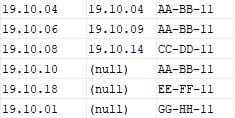
**Exercício 5**

**Enunciado: “**Implemente um procedimento denominado procDetetarAssociacoes, que permita detetar e corrigir, as associações de condutores a veículos que se sobrepõem temporalmente. Considere que os atributos data\_inicio e data\_fim são datas e, que os intervalos são fechados na data de início e abertos na data de fim ( [data\_inicio, data\_fim[ ).”.

**Interpretação:** O nosso grupo interpretou que este exercício deveria solucionar os casos onde um mesmo veículo estava associado a dois diferentes condutores no mesmo período de tempo.

**Output:**

****



**Problemas:** Não conseguimos alterar a data\_fim para a data\_inicio do período seguinte quando encontrada a sobreposição, apenas conseguimos alterar para a própria data\_inicio.

**Exercício 6**

**Enunciado: “**Implemente um procedimento em PL/SQL, designado procGuardarInformacaoSemanal, que insira na tabela ResumosVeiculos a informação retornada pela função funcObterInfoSemanalVeiculos e também escreva no ecrã um resumo da informação. O procedimento recebe como parâmetro o valor de uma data e insere na tabela ResumosVeiculos a informação retornada pela função funcObterInfoSemanalVeiculos, acrescida do instante temporal em quem a mesma foi inserida (Figura 1). Tenha em consideração que, para efeitos de teste, os registos da tabela ResumosVeiculos devem ser manualmente eliminados recorrendo, e.g. à instrução de SQL delete. No ecrã devem ser escritos os valores das percentagens de veículos que fizeram ou não viagens nessa semana, complementados com a lista das matrículas dos veículos associados a cada uma delas.”.

**Interpretação:** Para realizarmos este exercício, criámos um procedimento que recebia uma determinada data por parâmetro. Para a elaboração do procedimento, recorremos á implementação de 3 loops. O primeiro foi usado para extrair a informação contida num cursor que é retornado pela função implementanda no exercício 3, e adicioná-la aos dados relativos á tabela resumos veiculos. O segundo loop foi usado para para imprimir as matrículas dos veículos que tinham sido usados para a realização de viagens na semana á qual a data que é passada por parâmetro pretence, assim como a percentage de veículos que fizeram viagens nessa mesma semana. Já o terceiro e ultimo loop foi usado imprimir as matrículas dos veículos que não tinham sido usados para a realização de viagens nessa mesma semana, assim como a respetiva percentagem.

**Output:**

**(Não temos!)**

**Problemas:** O Código aparenta estar correto, no entanto, tivemos problemas em testá-lo, uma vez que nos surgirão erros de compilação, erros esses que não conseguimos resolver.