The assignment involves an FPSO (Floating Production, Storage and Offloading) vessel data. The vessel contains some equipment and each equipment have multiple sensors. Every time a failure happens, we get all the sensors data from the failed equipment and we store this information in a log file, the time is in GMT time zone.

You are provided 3 files: the log file named **equipment\_failure\_sensors.log**; the file named **equipment\_sensors.csv**with the relationships between sensors and equipment and the file named **equipment.json**with the equipment data.

To solve this problem, we expect you to answer a few questions related to January 2020 considering GMT time zone:

1 – Total equipment failures that happened?

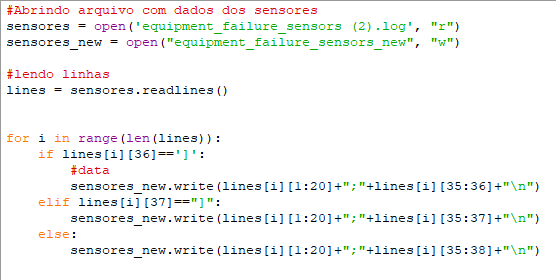
2 – Which equipment code had most failures?

3 – Average amount of failures across equipment group, ordering by the amount of failures in ascending order?

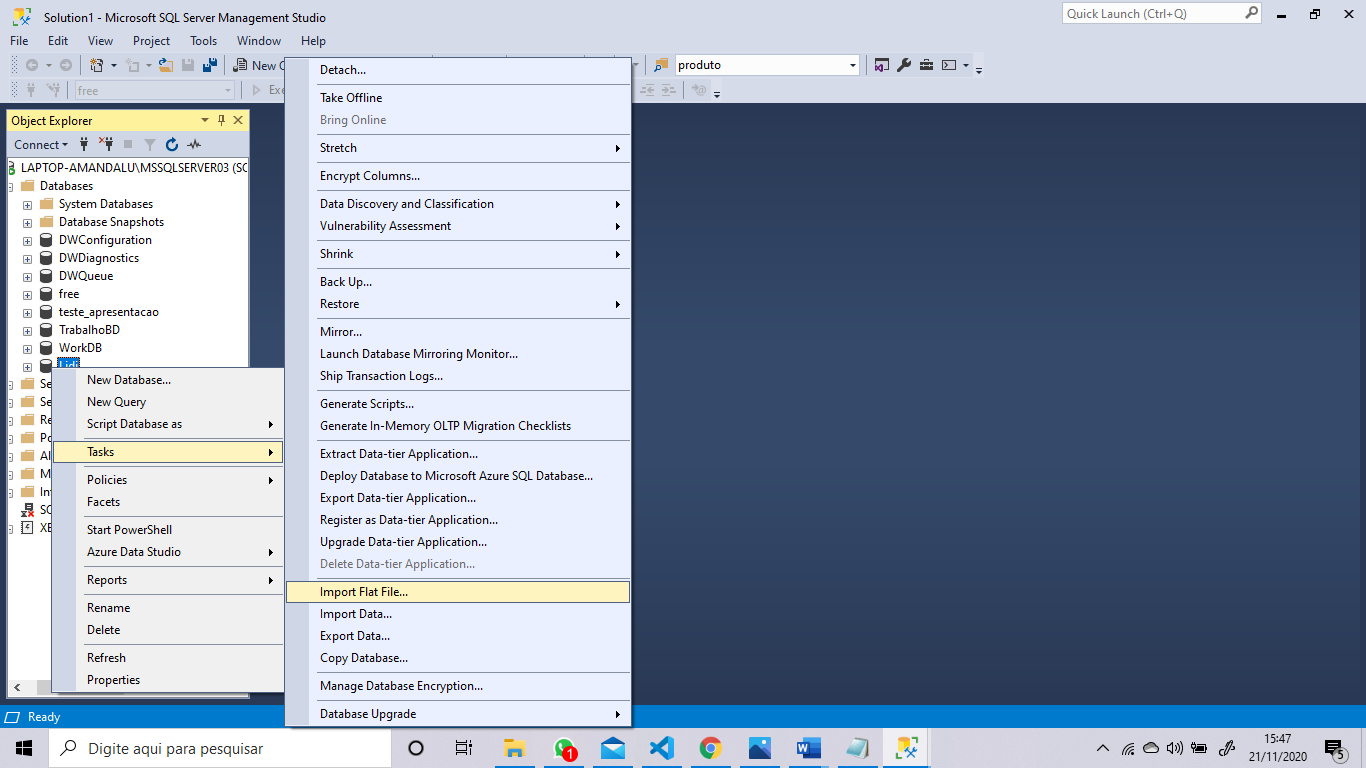
Feel free to use the programming language and tools (cloud, SQL databases, ...) you would like. Once you’re done, please send us a zip file containing all your code and a document explaining the steps that were taken with the responses for the questions, but also other conclusions and insights that you think are relevant to the project. We value creativity!

If the zip file is bigger than the attachment limit, please upload the zip file in a file sharing service (Google Drive, Dropbox, WeTransfer, ...) and send us the link.

Inicialmente foi realizada a conversão de todos os arquivos para csv. O procedimento foi realizado através do convertor online de json para csv para o arquivo equipment.json. Já para o arquivo equipment\_failure\_sensors.log, devido a falta de padronização na formatação, e a existência de dados que não seria utilizados na análise foi feita uma reparação em python. Essa foi repassada para excel e salva em csv. No código a seguir temos a leitura do arquivo (.log) e a criação de um novo (.txt), inserindo no zip deste documento. A identificação do sensor\_id foi feita através da quantidade de caracteres da string de cada linha. No entanto poderia ser feita através da indentificação de termos chave da mesma (como “sensor”)



O SGBD utilizado foi o SQL Server SMS, versão 18.6. O banco de dados permite a inserção de arquivos csv. Através dos passos apresentados na figura abaixo. Posteriormente foram realizadas as determinações de cada campo da tabela. Os campos id não foram inseridos como chave primária ou estrangeira (associando a tabela intermediária as outras duas), por não ser uma boa prática de uso. Utilizando como chave a criada pelo sistema.



Após a transferência das tabelas, iniciou-se o processo de produção de query, permitindo a análise dos dados. Inicialmente para contagem de equipment failures realizou-se um count, acessando assim a contagem de dados. Como o enunciado solicitou somente os dados de janeiro de 2020 utilizou-se um where, para filtrar o período ao qual estaríamos visualizando e contabilizando.

SELECT \* FROM equipment\_failure\_sensors;

SELECT \* FROM equipment\_sensors;

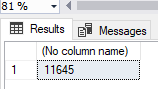
SELECT \* FROM equipment;

/\* LETRA A \*/

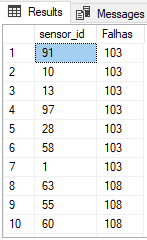
/\* o where realiza o filtro pela data \*/

SELECT Count(sensor\_id) FROM equipment\_failure\_sensors WHERE date BETWEEN '2020-01-01'

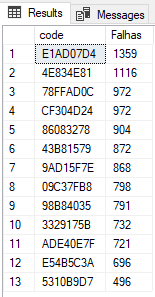
AND '2020-02-01' ;



Após a análise 1 ser realizado, foi criada uma tabela *equipment\_failure\_sensors\_total*,. Na mesma fora inseridos dados relacionados a visualização de dados do arquivo equipment\_failure\_sensors.log a partir do agrupamento por sensor\_id. Isso possibilitou que os diversos erros associados a um sensor de número x fossem somados e representados de forma conjunta.



Após a criação da tabela, bastou uni-la a intermediária (equipment\_sensors – referente ao arquivo equipment\_sensors.csv) e referente aos equipamentos (equipment). Observer que o 14 não aparece no resultado, pois o mesmo não está associado a nenhuma falha. A tabela, está ordenada em ordem decrescente. Desta forma, para filtra o código que apresentou mais falhas basta realizar um SELECT TOP 1. O valor está em destaque na figura a seguir.



/\* LETRA B \*/

/\* para saber qual equipamento obteve mais erros, precisamos relacionar a tabela equipamento a tabela failure.

Observar quantas falhas cada sensor teve e qual equipamento está conectado a mais sensores \*/

/\*Criando nova tabela com sensor\_id e numero total de falhas \*/

SELECT sensor\_id, sum(1) as Falhas INTO equipment\_failure\_sensors\_total

FROM equipment\_failure\_sensors WHERE date BETWEEN '2020-01-01' AND '2020-02-01'

GROUP BY sensor\_id order by sum(1) asc;

/\*unindo tabelas de forma que as chaves relacionadas da tabela intermediárias estejam conectadas para observar quantidade de falha por equipamento\*/

SELECT TOP 1 equipment.code, sum(equipment\_failure\_sensors\_total.Falhas) as Falhas FROM equipment

INNER JOIN equipment\_sensors

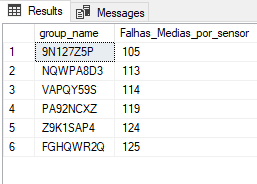
on equipment.equipment\_id=equipment\_sensors.equipment\_id

FULL OUTER JOIN equipment\_failure\_sensors\_total

on equipment\_failure\_sensors\_total.sensor\_id=equipment\_sensors.sensor\_id GROUP BY equipment.code

order by Falhas desc;

Para o item 3, assim como item 2, é necessário realizar um filtragem e união de dados. Desta vez, considerando o agrupamento pelo nome do grupo. Agrupamos em FULL OUTER JOIN para equipment\_sensors e em INNER JOIN para equipment\_failure\_sensors\_total. Isso faz com que a tabela intermediária seja sobrepostamente incluída, considerando a modelagem N para N das tabelas as quais a mesma interliga.



/\* LETRA C \*/

SELECT equipment.group\_name, sum(equipment\_failure\_sensors\_total.Falhas)/count(equipment.equipment\_id) as Falhas\_Medias\_por\_sensor FROM equipment

FULL OUTER JOIN equipment\_sensors

on equipment.equipment\_id=equipment\_sensors.equipment\_id

INNER JOIN equipment\_failure\_sensors\_total

on equipment\_failure\_sensors\_total.sensor\_id=equipment\_sensors.sensor\_id GROUP BY equipment.group\_name

order by Falhas\_Medias\_por\_sensor asc;