Otimização do *dataflow* e armazenamento no IGC

João Funenga
j.funenga@campus.fct.unl.pt

Faculdade de Ciências e Tecnologias da Universidade Nova de Lisboa

Abstract. Atualmente, o *dataflow* no Instituto Gulbenkian de Ciência processa-se de forma ainda algo manual e sem tomar partido de abordagens mais apropriadas como é o caso de bases de dados relacionais ou o de processamento automático dos dados. O objetivo deste relatório será refletir sobre várias abordagens que possam minimizar a dificuldade existente, aumentando assim a eficiência do trabalho laboratorial no que toca à gestão e armazenamento dos dados produzidos pelos pratos de análise de amostras, os ensaios ELISA. Isto refletir-se-á num aumento tanto da produtividade por ser mais fácil aceder aos dados que se querem utilizar, como no âmbito da segurança pelos dados estarem numa base de dados protegida com credenciais específicas para diferentes níveis de permissão ou até pela questão de controlo de versões, para que nunca se trabalhe sobre versões desatualizadas. Este último problema foi um dos mais referidos pelos investigadores devido à troca de ficheiros entre funcionários o que dá azo a incorreções.

Keywords: Dataflow · Automatização · ELISA.

1 Introdução

O IGC, Instituto Gulbenkian de Ciência, foca-se maioritariamente em trabalho laboratorial e, como esperado, produz inúmeros dados provenientes de diferentes amostras e, por sua vez, inúmeras observações o que devido às possíveis diferenças entre as amostras pode requerer um processamento diferente mas entraremos nesse aspeto mais à frente. Este relatório focar-se-á maioritariamente sobre o *dataflow* associado às observações feitas pelo ensaio ELISA, que usa um prato que contém 384 poços para fazer diferentes amostragens. O ensaio ELISA é um teste de imunoabsorção enzimática que tem como propósito detetar anticorpos ou antigénios específicos numa amostra. Laboratoriamente [1][2], num prato de ensaio ELISA, o antigénio (a macromolécula *target*) é colocado na superfície do prato e depois junta-se com um anticorpo que está ligado a uma enzima marcadora. A deteção é feita medindo a atividade dessa enzima via encubação com um substrato apropriado para produzir um resultado mensurável. Dito isto, a parte mais importante deste ensaio e o que o torna relevante é o da interação bastante específica entre o anticorpo e o antigénio. A determinação da concentração do antigénio numa amostra requer a construção de uma curva usando antigénios com concentração conhecida. A concentração dos antigénios

na amostra pode assim ser calculada relacionando a concentração da amostra conhecida com os resultados da amostra que desejamos de um teste de densidade óptica.

Este desafio prende-se com a otimização bem como a automatização no *pipeline* de tudo o que é *data-related*. Isto porque, num ambiente laboratorial, desde o momento em que são feitas as análises às amostras, passando pelo pré-processamento dos dados registados, análises posteriores aos dados observados e o armazenamento dos dados finais tratados numa base de dados já estabelecida com dados históricos, estão envolvidas muitas trocas de dados. Adicionando ao facto de haver todo este fluxo de dados envolvidos, temos ainda que ter em conta que por cada prato temos 384 amostras, logo o volume de dados é significativo.

Devido a estes fatores, este tipo de problema insere-se na área de *Big Data* que, como aprendido até agora no mestrado, ensina e dá ferramentas para lidarmos com um grande fluxo de dados e como devem ser processados. Possivelmente, não será necessário usar ferramentas demasiado especializadas para grandes dados, visto que o output de cada uma das amostras não será muito extenso mas tal será analisado posteriormente.

2 Problema

2.1 Situação Atual

De modo a poder expor possíveis soluções para este problema, primeiro descreverei a situação corrente de como o *dataflow* se processa de modo a perceber em que partes do processo é possível atuar e de que forma, de modo a facilitar o trabalho.

Começaremos pelo início, onde os dados em si são recolhidos. Como explicado acima, o ensaio ELISA é feito num prato que tem 384 poços e em cada um existe uma amostra a ser analisada. Destes 384 poços, existem alguns que têm amostras de controlo (64 deles) e outros que estão de facto a ser analisadas. Por uma questão de consistência, das amostras que não são de controlo, ao invés de termos apenas uma amostra de cada tipo de anticorpo, todas as amostras são duplicadas. Depois disto as amostras são passadas por um sensor de densidade óptica.

O *output* final do *software* usado para determinar as concentrações dos anticorpos nas amostras para cada poço do prato do teste ELISA é um ficheiro de texto que contém os dados retirados diretamente das amostras propriamente ditas. Por dia, são feitas observações em 4 pratos diferentes logo temos 4 ficheiros de texto para processar diariamente.

Durante a pandemia, foi desenvolvida uma aplicação para auxiliar na visualização nos resultados das amostras no contexto de uma tese de doutoramento. Esta aplicação é utilizada nesta fase do *pipeline* para verificar se existem diferenças significativas entre os resultados das amostras duplicadas. Visto que as amostras são constituídas do mesmo material, é de esperar que os resultados sejam muito semelhantes e apenas difiram com uma pequena margem de erro

uma da outra. Caso se verifique que há uma diferença anormal entre as amostras duplicadas então sabemos que o procedimento terá de ser repetido. Caso tudo esteja conforme esperado, então é agora feita a normalização dos dados para cada quadrante relativamente à amostra de controlo respetiva e é feita uma média de entre as amostras duplicadas para termos um valor final relativo à amostra. Este passo funciona como uma fase de controlo de qualidade para percebermos se os dados retirados das amostras estão todos conforme devem estar e é fundamental de se fazer em qualquer tipo de análise laboratorial. Para além deste passo de controlo de qualidade existe um outro igualmente importante relativo às amostras de controlo. Com o passar do tempo, por serem amostras com material biológico, é normal a qualidade do que foi definido como controlo se deteriorar levando assim a ser necessário serem substituídas por amostras novas.

Neste momento, esta aplicação é maioritariamente usada para fazer *download* dos dados e organizá-los de modo a termos forma de aceder ao conteúdo de cada um dos poços. Isto é importante porque o ensaio ELISA pode ser feito com diferentes concentrações do antigénio que captura os anticorpos e também diferentes concentrações do segundo anticorpo, diferentes anticorpos em si ou até vários níveis de diluição da própria amostra e devido a todas estas diferenças que podem existir em cada um dos poços é importante conseguirmos obter a descrição do que está a ser analisado.

O output final com as concentrações tem uma estrutura tabular o que, por ser estruturado, facilita o tratamento posterior por qualquer tipo de ferramenta que queiramos, desde aplicações específicas para análise de dados como algo mais genérico como um *script* de *python*.

Um aparte importante a fazer é que, nem todos os dias se usam os 4 pratos, nem todos os poços de cada prato. O importante de denotar aqui é que, para cada prato, existe um ficheiro de metadados associado que tem representado uma descrição de cada uma das amostras.

3 Estruturas

3.1 Estrutura do prato do ensaio ELISA

O prato no qual é feito o ensaio ELISA é constituido pelos 384 poços falados anteriormente mas está agrupado em secções de 96 poços, com 4 secções destas perfazendo assim os 384. Isto é importante de referir porque os metadados representantes da descrição de cada uma das amostras chega num formato de 96 descrições agrupadas. Cada um dos pratos está assim dividido em 4 quadrantes, A1, A2, B1, B2.

Esta divisão é importante porque as amostras existentes em cada um destes quadrantes é processada de maneira diferente. Isto porque, para cada quadrante temos amostras de controlo diferentes e assim, os dados resultantes dos poços de cada um dos quadrantes têm de ser normalizados de uma determinada maneira (de acordo com a amostra de controlo do seu quadrante).

3.2 Estrutura dos dados

Como referido anteriormente, o *software* para a análise de concentração dos anticorpos dos pratos do teste ELISA terá como output um ficheiro com uma determinada estrutura. Para esta tabela haverá várias colunas, sendo a chave principal destas o `sample_id` (*id de cada uma das amostras*) e os restantes atributos correspondentes aos valores do que foi medido nas amostras. Relativamente à natureza do processo, analisar os anticorpos de indivíduos, existem várias amostras que podem corresponder à mesma pessoa em várias alturas no tempo. No entanto, nós podemos não ter a pessoa a que corresponde uma determinada amostra e apenas termos o `id` de participante (que é anónimo). Deste modo é necessário agregar posteriormente a informação que relaciona os `ids` dos participantes com os participantes em si já depois de termos os atributos medidos para cada um dos *participant ids*. As próprias pessoas também têm atributos como é o caso da idade e do género. Existe um pormenor importante de referir relativamente às pessoas que estão a ser seguidas. Para as análises que estão a ser feitas, existem vários *cohorts* de pessoas a serem observadas sendo estes por exemplo, funcionários de um hospital. Embora hajam alguns atributos que são comúns às pessoas de todos os *cohorts* como é o caso da idade e do género, em particular para cada um destes grupos existem diferentes atributos a ter em conta. Um exemplo disto é, nas amostras do pessoal médico um dos atributos pode ser o seu cargo dentro do hospital mas para um *cohort* de residentes de um lar podemos ter alguns atributos relativamente ao seu historial médico por exemplo se têm alguma doença crónica, diabetes ou mesmo um índice de saúde que é usado para classificar o estado de saúde de uma pessoa.

O que parece ser intuitivo para se fazer com uma base de dados relacional relativamente às tabelas finais para armazenarem os dados históricos é ter uma para as medições dos ensaios ELISA com os respetivos atributos medidos e descrição de cada uma das amostras, uma tabela relativa aos participantes dos quais foram extraídas as amostras e uma tabela para cada um dos *cohorts* contendo os atributos específicos de cada um. Depois de construídas as tabelas, a leitura será feita diretamente da *database* de modo a garantirmos estar sempre a ler os dados mais recentes e estarmos protegidos pelas outras garantias que lhe são inerentes.

4 Porquê uma base de dados relacional?

Uma base de dados relacional é um dos tipos de bases de dados existentes que faz o armazenamento dos dados usando relações entre estes. Este tipo de bases de dados materializam um modelo relacional que é bastante intuitivo ao se representarem os dados em diferentes tabelas para cada uma das entidades, ligadas entre si com `ids` únicos. Cada linha de uma tabela representará um registo e cada um dos registos terá valores para os N atributos dessa tabela (representados pelas colunas). O modelo relacional representa também o facto de que a estrutura lógica dos dados (por exemplo as tabelas) está separada da estrutura física (relativamente ao armazenamento), o que resulta por exemplo nas operações

feitas à base de dados para manipular os dados existentes nestas permitirem que uma aplicação especifique o conteúdo necessário e a parte que determina como é que esses dados devem ser *retrieved* fica do lado das operações físicas. Nos primórdios das bases de dados, todas as aplicações armazenavam os dados numa única estrutura e quando era preciso criar aplicações que usassem estes dados era preciso conhecer ao pormenor a estrutura dos dados específica para encontrarem o que precisavam. Esta metodologia era claramente ineficiente e bastante difícil de integrar com qualquer tipo de sistema e foi por isto que surgiram as *databases* relacionais. Estas visam padronizar o armazenamento dos dados para ser mais fácil a gestão e uso dos mesmos. Um dos principais motivos para se utilizar um sistema próprio de armazenamento de dados ao invés de ficheiros *excel* é relativamente à questão de, por haver mais do que um ensaio ELISA diariamente e se querer sempre que uma leitura seja sobre a última escrita, este tipo de garantias é dada por um sistema destes. Como referido pelos investigadores, andar-se a trocar ficheiros entre eles não é nada prático e muitas das vezes acaba-se por se trabalhar e analisar dados que não são os mais recentes. Existem inúmeras vantagens em usar este tipo de bases de dados e estas têm de seguir obrigatoriamente algumas propriedades.

4.1 Propriedades das Bases de Dados

As propriedades que as Bases de Dados têm de seguir são as representantes do acrónimo ACID:

- Atomicidade
- Consistência
- Isolamento
- Durabilidade

Relativamente à **atomicidade**, esta representa que cada operação deverá ser indivisível. Isto é, as operações que uma transação contém ou são todas executadas (no caso de não haver nenhum erro) ou então nenhuma das operações executa (em caso de erro) logo, após findada uma transação a base de dados contém o resultado de todas as operações existentes nessa mesma transação e não apenas partes.

Para a **consistência**, temos que esta representa que a execução de uma transação sobre a base de dados deve resultar sempre num estado válido, sem nada ficar corrompido.

Relativamente ao **isolamento**, por estarmos num contexto em que várias transações podem estar a ocorrer de forma concorrente (em paralelo), com vários utilizadores a fazerem operações simultaneamente na base de dados, esta propriedade garante que o resultado de todas estas operações seria igual caso fossem realizadas sequencialmente.

Por último, para a **durabilidade**, esta apenas diz que depois de uma transação ser executada, esta será mantida mesmo se houver uma falha no servidor. Isto é possível de se concretizar devido aos dados serem armazenados em discos rígidos que guardam os dados mesmo que o sistema se desligue.

5 Melhoramentos na aplicação existente

Neste momento na aplicação criada para auxiliar a visualização do conteúdo das amostras, no output apenas obtemos o *participant id* (anónimo e único) e seria muito mais interessante poder ter toda a informação relacionada com cada uma das amostras para podermos fazer *backtrack* e verificarmos tudo o que quisermos acerca de uma amostra em especifico. Com o novo *dataflow* explicado abaixo implementado e o sistema de base de dados a suportar o armazenamento dos dados, será possível fazer *queries* de modo a termos os dados que queremos.

6 Novo Dataflow

O que precisaremos para melhorar o atual funcionamento do laboratório é termos uma *pipeline* de ETL [3][4](*Extraction, Transformation, Load*) bem definida. Estas três fases do processo de ETL são os 3 pontos essenciais, basais a qualquer *pipeline*.

A *extraction* representa a fase da recolha dos dados a partir da *source* que neste caso serão os ficheiros produzidos pelos ensaios ELISA com a respetiva análise de densidade óptica, a fase de *transformation* representa a parte de estruturar, limpar e aplicar algum tipo de controlo de qualidade necessário antes de passar à ultima fase, a de *load* na qual os dados tratados serão escritos e armazenados numa base de dados. Devido a termos 4 ELISAs a fazerem a mesma tarefa diariamente, esta *pipeline* será aplicada em paralelo a cada uma delas.

De modo a que os dados produzidos pelo ensaio ELISA sejam *retrieved* por outra/mesma máquina que fará o processamento dos mesmos, é necessário a existência de um SQL server. De uma forma simplificada, esta é uma aplicação que permite que outras aplicações, como por exemplo um *python script*, leia os dados armazenados nesse servidor e possa fazer as tarefas necessárias relativas à fase do *transform*. Depois destas transformações e controlos de qualidade, pode ser carregado para a base de dados final onde se encontram os dados históricos.

6.1 Primeira Parte - *Extraction*

O primeiro passo nesta pipeline será então a recolha do output dado pelo ensaio ELISA. Este processo gera um ficheiro de texto que é guardado numa máquina. Devido a este ficheiro ter sempre a mesma estrutura (entre as várias "runs", mudando apenas o número de amostras a serem analisadas), este pode ser diretamente carregado com um simples comando SQL[5] para uma tabela temporária representativa da ELISA. No entanto, para concretizar isto é preciso saber quando é que o ficheiro é escrito e ter alguma maneira de executar a operação. Para isto, pode-se criar um pequeno *script* cuja função seja apenas verificar se existe um novo ficheiro gerado, e executar o comando explicado acima para a base de dados temporária. Em alternativa, caso não se queira ter sempre um script a correr, pode-se definir uma *scheduled task* no Windows[6] para correr este *script* no final do dia para inserir os dados.

Outro aspeto extremamente importante é guardarmos a *timestamp* de quando a análise numa ELISA foi feita. Para isto, ao ser gerado o ficheiro de texto por cada análise das concentrações, pode-se criar uma nova coluna na tabela temporária relativa às amostras e colocar a *timestamp* para todas estas de modo a sabermos quando foi feita a análise. Isto será importante mais à frente, no passo de *loading* para percebermos se alguma coisa falhou (explicado na secção de recuperação e *backup*). Relativamente a isto, é recomendado ter bem definido o formato da data para que a ordenação dos dados seja correta, isto é, pode-se definir como Ano/Mês/Dia ao invés do contrário, mais habitual em Portugal.

Todo este processo de extração ocorre em 4 pratos de ensaios ELISA e assim, por cada output das análises de concentrações ser para um computador, isto poderá ser paralelizado. Não existe problema relativamente a escritas ou leituras dessincronizadas devido às propriedades que as transações em bases de dados conferem.

6.2 Segunda Parte - *Transformation*

Depois de termos os dados de cada ensaio ELISA em tabelas temporárias (1 amostra por *row*) e os da densidade óptica noutra (associadas pelo `id` da amostra), chega a parte de fazer o controlo de qualidade e todas as tarefas subjacentes. Antes de se passar à transformação dos mesmos, pode-se neste momento inserir os meta-dados representantes do conteúdo de cada um dos poços nesta base de dados. Deste modo, todas as amostras estarão associadas a uma descrição (indicadora do conteúdo da mesma). Isto é particularmente importante devido à segregação das amostras num prato, como explicado na parte inicial, por existirem diferentes quadrantes. Isto é significativo pois agora, a standardização dos dados recolhidos terá de ser feita de acordo com cada um dos quadrantes, tendo amostras de controlo específicas também para cada um. Por termos então as coordenadas de cada um dos poços tabelada, sabendo assim a que quadrante cada um deles pertence, poderemos então passar para o tratamento dos dados fazendo verificações de qualidade e standardizações.

Primeiramente dever-se-á verificar a qualidade dos dados. Isto pode passar pela verificação da presença de atributos vazios, caratéres especiais ou outras anomalias gerais, o que é pouco provável. Isto pode ser feito com recurso a comandos simples de SQL, transformando diretamente os dados que temos tabelados.

De seguida, tendo apenas dados passíveis de serem analisados, verificar-se-á qualidade dos mesmos no que toca aos atributos. Isto será feito comparando os valores registados com as amostras de controlo para cada quadrante e descartando as que cujos valores estejam fora do "normal" laboratorial. Isto é facilmente feito com um outro *script* que poderá também alertar caso todos os valores se desviem dos de controlo. Quando isto acontece, é quando chega à altura de se ter de trocar as amostras de controlo, por serem material biológico e estarem sujeitas a se deteriorarem com o passar do tempo.

Depois de se fazer as comparações com as amostras de controlo, como explicado na subsecção da estrutura dos pratos do ensaio ELISA, devido às amostras

estarem todas duplicadas, o resultado final de cada uma delas é uma média das duplicadas de modo a ter um valor mais correto intermédio por haver sempre pequenos desvios nos valores de cada uma.

Finalmente, depois de todas estas verificações, dever-se-á standardizar os dados registados de acordo com as amostras de controlo para cada um dos quadrantes.

6.3 Terceira Parte - *loading*

Depois de tudo isto feito, deveremos adicionar estes dados tratados aos dados já existentes numa base de dados com dados históricos, já transformados anteriormente e colocados nesta. Deste modo, conseguimos ter todos os dados históricos disponíveis num sítio em que, dependendo do nível de segurança que se queira no que toca a permissões, possa ser definida e personalizada para diferentes pessoas do laboratório para terem acesso à base de dados. Estas poderão depois consultar ou extrair os dados para possíveis análises mais aprofundadas noutras ferramentas como por exemplo em R, python ou em plataformas de visualização de dados.

Relativamente a esta fase, dever-se-á também criar uma outra coluna relativa à *timestamp* de quando os dados foram mesmo inseridos com sucesso na base de dados. Isto permite salvaguardar caso haja algum erro e sabermos precisamente quando é que os dados foram armazenados.

É nesta fase que provavelmente está a parte mais laboriosa da implementação deste novo *dataflow* mas que felizmente apenas precisa de ser feita uma vez, antes de se começar a por este novo sistema em uso. Isto será trabalhoso porque atualmente todos os dados históricos existentes ainda se encontram em excel, não estruturados devidamente para que seja imediata a inserção dos mesmos na base de dados.

6.4 Recuperação e Backup

Num ambiente laboratorial é extremamente importante manter a integridade dos dados e ter garantias que caso exista algum tipo de falhas, se possa saber exatamente o que se perdeu e quando é que tal aconteceu. Para concretizar isto, existem algumas verificações que podem ser feitas. Como falado inicialmente, para cada dia são criadas tabelas temporárias cujos dados depois serão limpos e transformados e finalmente colocados na base de dados com a informação histórica. No entanto, este último passo pode falhar, podem existir problemas nesta transação ao agregar esta nova informação. Deste modo, o que se pode fazer é para cada um dos dias, verificar se as últimas *timestamps* dos dados históricos correspondem às *timestamp* dos dados temporários e caso isto se verifique apagamos os dados temporários (representantes daquele dia).

Outro problema que pode ocorrer é na fase de transformação dos dados, por não termos nenhuma cópia dos dados temporários, caso haja uma falha perdemos os dados com que estávamos a trabalhar. Uma solução para mitigar este problema é criar uma cópia das tabelas temporárias (do dia) para que caso

ocorra alguma falha nas máquinas que guardam e trabalham sobre estes dados, termos uma réplica num outro disco para que não se perca nenhuma informação.

Finalmente, a informação histórica tem muito valor subjacente e é algo que se deve ter a certeza que não se perde pois engloba informação histórica e não se consegue simplesmente repetir isto. Por isso, tem de ser implementado um mecanismo de backup que assegure os dados históricos. O que se pode fazer é, no final de cada dia, fazer um backup da tabela com os dados históricos para um suporte de armazenamento diferente para que, no caso de existir alguma falha, apenas perdemos um dia no pior dos casos devido a termos sempre o backup da informação histórica aquando o dia anterior armazenada noutro sítio.

6.5 Performance

Com o aumento da quantidade dos dados, a performance do sistema pode começar a ser um problema e a impactar a viabilidade do uso de armazenamento deste género. A informação ao ser armazenada fica em blocos de dados. Ao fazer uma *query*, todos os blocos são acedidos como se fossem um só o que leva a que os registos contínuos dos ensaios ELISA ao longo dos anos seja todo acedido, impactando assim a velocidade de uma *query* sem necessidade. Fazer *indexing*[7] permite ter uma tabela de "chaves" que funcionam como apontadores para as linhas correspondente dessas chaves da tabela histórica o que permite que uma linha seja *retrieved* eficientemente e não seja necessário percorrer toda a tabela (análogo a um *hash-map*). Essa tabela extra por ser mais pequena torna as *queries* muito mais rápidas. O que faz sentido fazer é usar indexing para o campo da data de produção dos dados (pelo teste ELISA) e ao querermos extrair dados de um determinado período ou de uma data em específico, esta operação torna-se muito mais rápida.

Outra abordagem para aumentar a *performance* deste sistema é fazer *partitioning*[8]. Isto representa nada mais nada menos que separar uma tabela grande em várias pequenas. Como é natural, ao trabalharmos com tabelas mais pequenas, os tempos de acesso aos dados tornam-se menores. Existem dois tipos de *partitioning* que pode ser feito, vertical e horizontal. No vertical temos que as tabelas são separadas de modo a manterem o mesmo número de linhas mas cada uma terá menos colunas. Neste caso, a análise dos dados do laboratório usa as mesmas colunas para as análises e não se justifica fazer este tipo de partição das tabelas. No entanto, a partição horizontal pode ser útil caso se queira fazer análise dos dados por períodos de tempo. Assim, pode-se separar a tabela em várias por mês por exemplo de modo a trabalhar-se com menos dados.

De acordo com as necessidades do laboratório, a abordagem de utilizar *indexing* parece ser menos indicada por ser mais utilizada para fazer *queries* cujos *result sets* sejam pequenos. Neste caso, de acordo com as necessidades dos investigadores, a maioria das *queries* será de um grande volume de dados para se conseguir fazer análises mais aprofundadas. Assim faz mais sentido utilizar-se *partitioning* pela data de criação dos dados e quando se quiser fazer o *retrieval* dos dados de por exemplo 3 meses, se aceda apenas às tabelas correspondentes a estes.

7 Conclusão

O problema no qual este relatório se focou é um dos pontos fulcrais a ter-se bem solidificado num ambiente em que estão constantemente a gerar-se dados. Todos os dias são analisadas novas amostras e ter de se lidar constantemente com ficheiros individuais não é nada prático. Tendo isto em conta, a abordagem explicada neste relatório de construir uma *pipeline* adequada para os dados automatizaria o que atualmente é feito manualmente e facilitaria a vida de todos os envolvidos.

References

1. https://www.immunology.org/public-information/bitesized-immunology/experimental-techniques/enzyme-linked-immunosorbent-assay, Acedido a 15/7/2022
2. https://www.thermofisher.com/pt/en/home/life-science/protein-biology/protein-biology-learning-center/protein-biology-resource-library/pierce-protein-methods/overview-elisa.html, Acedido a 15/7/2022
3. A. Simitsis, P. Vassiliadis and T. Sellis, "Optimizing ETL processes in data warehouses," 21st International Conference on Data Engineering (ICDE'05), 2005, pp. 564-575, doi: 10.1109/ICDE.2005.103.
4. https://www.talend.com/resources/what-is-etl/, Acedido a 6/7/2022
5. https://stackoverflow.com/questions/13579810/how-to-import-data-from-text-file-to-mysql-database, Acedido a 6/7/2022
6. https://www.windowscentral.com/how-create-automated-task-using-task-scheduler-windows-10, Acedido a 7/6/2022
7. https://stackoverflow.com/questions/1108/how-does-database-indexing-work, Acedido a 7/6/2022
8. https://stackoverflow.com/questions/15425230/is-it-a-good-idea-to-index-datetime-field-in-mysql, Acedido a 7/6/2022