**5.3 Tradução das interrogações do utilizador para SQL (alguns exemplos)**

Todos os requisitos de exploração definidos na secção ‘x’ foram criados através de procedimentos com exceção do requisito RE9 que foi criado como uma função. Essa decisão baseou-se no facto de se pretender procurar informações na base de dados através de parâmetros, pelo que apenas é necessário projetar informação das tabelas. No requisito RE9 optamos por criar uma função pois o objetivo é devolver um número (número total de consultas das revistas de uma determinada empresa).

Para além dos conceitos abordados acima, foram também usadas as funções elementares do *SQL* como *SELECT, WHERE, ORDER BY, GROUP BY, INNER JOIN.* e

De seguida encontram-se alguns exemplos do código SQL criado de forma a responder aos requisitos de exploração.

**5.3.1 Listar todos os escritores que já contribuíram para a escrita de revistas de uma dada empresa**

Este procedimento faz a pesquisa em cinco tabelas, daí serem usados quatro *join’s* para obter a tabela resultante. É ainda feita uma restrição ao nome da empresa.

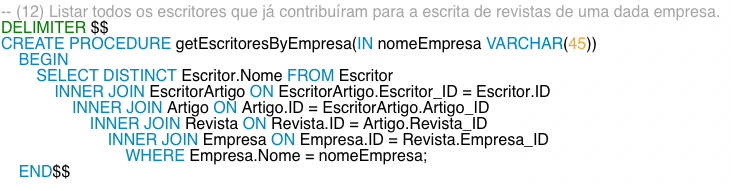


Figura x1 – Tradução do RE10 para SQL

**5.3.2 Calcular o número total de consultas das revistas de uma empresa**

Esta função foi a única que necessitamos de criar pois é o único requisito que devolve apenas um valor. Primeiro, tabelámos a coluna que contém o número de consultas das revistas, selecionando as revistas que referenciam a empresa pretendida. De seguida, aplicámos a função *SUM* do *SQL* para somar o número de consultas obtendo o resultado pretendido.

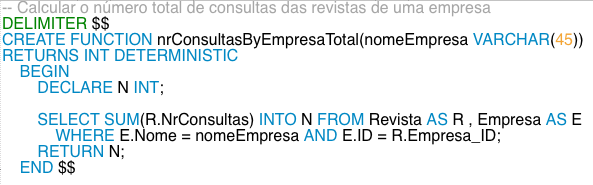


Figura x2 – Tradução do RE9 para SQL

**5.3.3 Listar as categorias por frequências relativas de consultas**

Este procedimento traduz o requisito que pede para calcular as frequências relativas de consultas por categoria. Numa primeira fase, calculámos a tabela que contém as categorias e os respetivos números de consultas totais usando a função *GROUP BY*, que permite agrupar as categorias pelo respetivo somatório do número de consultas, eliminando as repetições. De seguida, a cada elemento da coluna que contém a soma total do número de consultas de uma certa categoria, vamos dividi-lo pelo número total de visualizações de todas as revistas e multiplicámos por cem, obtendo assim as frequências relativas.

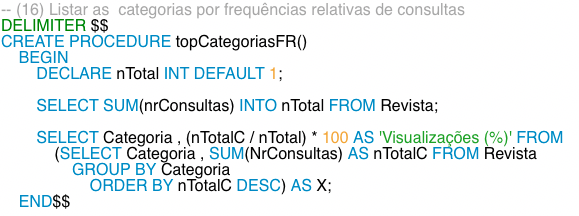


Figura x3 – Tradução do RE15 para SQL

**5.5 Escolha, definição e caracterização dos índices em SQL**

O *MySQl* faz a atribuição automática de índices aos atributos que sejam chaves primárias, pelo que todas as chaves primárias envolvidas nas tabelas da base de dados são índices. Para além disso, todos os atributos que sejam restringidos como *UNIQUE,* também são indexados.

Face às traduções das interrogações dos utilizadores, podemos verificar que a maior parte das consultas às tabelas são feitas através do atributo ‘Nome’ das diferentes entidades. Com isto, considerámos restringir esses mesmos atributos como *UNIQUE* para que, consequentemente, a pesquisa sobre as tabelas fosse mais eficiente. No entanto, tal decisão não fazia sentido dado o contexto do problema pois é permitido haver, por exemplo, duas empresas com o mesmo nome ou duas revistas com o mesmo nome.

**5.7 Definição e caracterização de algumas vistas de utilização do SQL**

As interrogações traduzidas para *SQL* baseiam-se todas em pesquisas personalizadas pelo que necessitam quase sempre de parâmetros de entrada para poderem ser executadas. Consequentemente, decididos criar vistas sobre todas as tabelas com informação relevante tanto para o visitante do site como para o utilizador da aplicação. Foram então criadas vistas sobre todas as entidades da base de dados com exceção das tabelas que surgiram do relacionamento entre entidades. Para além das vistas básicas sobre as entidades, foram também criadas mais duas vistas, uma para a consulta das empresas e o número total de consultas das suas revistas e outra para listar o *Top 5* de escritores, ou seja, os escritores cujos artigos têm mais visualizações. De seguida, encontram-se alguns exemplos do código *SQL* de algumas vistas.

**5.7.1 Vista da tabela Escritor**

Vista muito simples que permite aceder ao nome de todos os escritores que estão na base de dados do arquivo de revistas.



Figura x4 – Vista sobre a tabela Escritor

**5.7.2 Vista das empresas e respetivas visualizações das suas revistas**

Vista que permite visualizar o nome de todas as empresas e respetivas visualizações das suas revistas. Para tal, juntámos as duas tabelas Revista e Empresa, associando uma empresa às respetivas revistas. De seguida, aplicámos a função *GROUP BY* de modo a eliminar as repetições e somar o número de consultas. O resultado é apresentado ordenado por ordem decrescente, para tal, usámos a função *ORDER BY.*

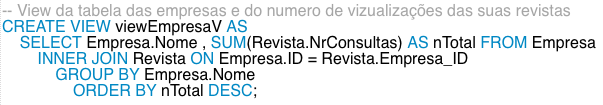


Figura x5 – Vista das empresas e respetivas visualizações totais das suas revistas

**5.7.3 Vista do *Top 5* de escritores**

Vista que permite visualizar o *Top 5*  de escritores. Para a vista ser criada, simplesmente juntámos as tabelas Escritor e EscritorArtigo e agrupámos a tabela pelo nome de escritor contando o número de ocorrências que um dado escritor tem na última tabela mencionada. Para o resultado final pretendido, usámos a função de ordenação *ORDER BY* para ordenar as entradas pelo número total de artigos e limitando a tabela em cinco entradas através da função LIMIT com o parâmetro cinco.



Figura x6 – Vista do *Top 5* de escritores

**5.8 Definição dos mecanismos de segurança em SQL**

O site desenvolvido para a Dona Ritinha é utilizado por visitantes (ações de consulta) e pela própria Dona Ritinha (ações de inserção, atualização, remoção e consulta). Dado isto, decidimos criar apenas dois tipos de utilizadores, o administrador (*admin*) da base de dados e o da aplicação (*app*), visto que é esta que vai gerir os movimentos da base de dados.

O administrador da base de dados tem todas as permissões possíveis sobre a base de dados visto que é este que vai gerir a mesma.

O utilizador da aplicação tem permissões de seleção, inserção, execução, atualização e de remoção sobre todas as tabelas com exceção das tabelas que tiveram origem das ligações (N:N).

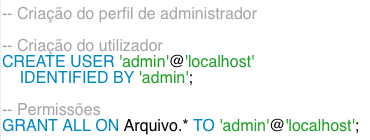


Figura x5 – Código SQL para a criação da vista de utilização e respetivas permissões do utilizador *admin*

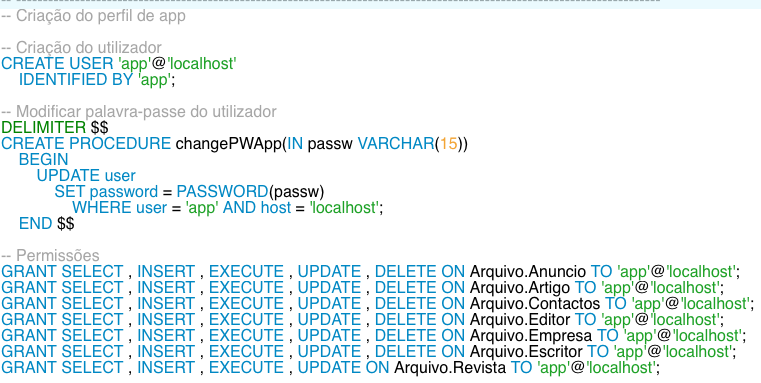
**

Figura x6 – Código SQL para a criação da vista de utilização e respetivas permissões do utilizador *app*

**5.9 Revisão do Sistema Implementado com o utilizador**

Terminada a implementação do modelo físico, a equipa reuniu-se com a Dona Ritinha uma última vez para garantir que todos os requisitos foram cumpridos e que o projeto desenvolvido se encontra apto para ser lançado.

Tendo em conta que a Dona Ritinha não possui conhecimentos de informática, a equipa decidiu realizar uns requisitos extra que virão a ser necessários para a aplicação que irá gerir a base de dados, como por exemplo, as definições das vistas sobre as tabelas mais importantes da base de dados, assim como um *trigger* que atribui o número total de revistas de uma empresa ao atributo que guarda tal informação na entidade Empresa.

No geral, a Dona Ritinha está satisfeita com o trabalho desenvolvido para o seu arquivo de jornais e considera que todas as suas exigências foram cumpridas.

**5.4 Tradução das Transações estabelecidas para SQL**

Nesta seção apresentamos as transações criadas para a base de dados. Foram desenvolvidas transações para inserção de dados em todas as tabelas do arquivo de revistas. De seguida, apresentamos alguns exemplos dessas transações.

**5.4.1 Inserir uma empresa**

O procedimento recebe como parâmetros o nome, a cidade e a rua da empresa. Primeiro é calculado o índice que esta nova entrada na tabela vai ter como chave primária. De seguida, testamos se existem empresas com o mesmo nome. Caso não haja, a nova empresa é adicionada à tabela usando o comando *COMMIT* e é lançada uma mensagem de sucesso. Caso haja uma empresa com o mesmo nome, através do comando *ROLLBACK,* desfazemos as alterações feitas e é lançada uma mensagem de erro.

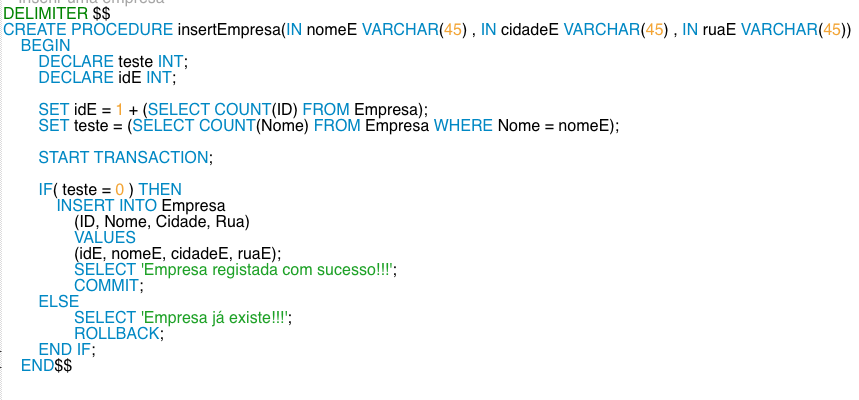


Figura x – Tradução da transação de inserção de uma empresa em SQL

**5.4.2 Inserir um artigo**

O procedimento recebe como parâmetros o nome da revista à qual pertence o artigo a inserir, a data em que foi escrito, o seu título, o corpo (conteúdo), a categoria em que está inserido e, por fim, o seu escritor. Numa primeira fase é calculado os *ID’s* da revista e do escritor do artigo bem como o *ID* que a nova entrada irá ter. De seguida, caso os *ID’s* da revista e do escritor existam , é adicionado o artigo à tabela com referência à respetiva revista, e é atualizada a tabela EscritorArtigo, adicionando o novo relacionamento entre o artigo adicionado e o seu escritor através do comando  *COMMIT*. Caso os *ID’s* da revista ou do escritor do artigo não existam, a inserção fica sem efeito aplicando o comando *ROLLBACK.*

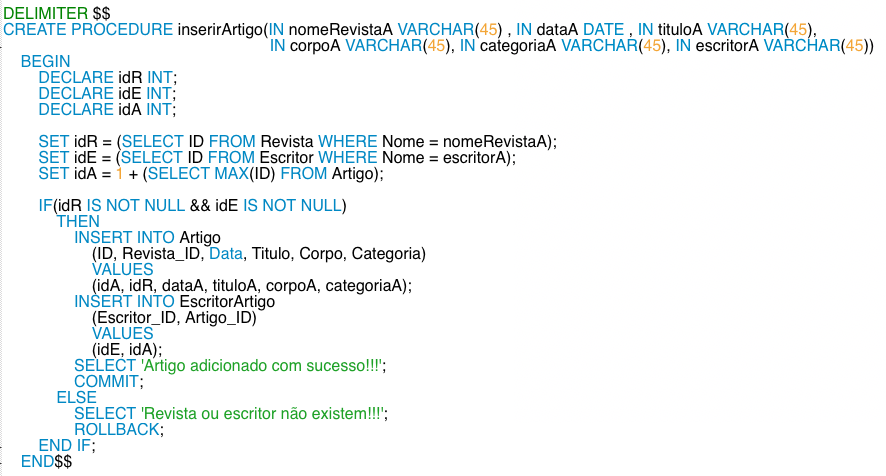
****

Figura x – Tradução da transação da inserção de um artigo em SQL