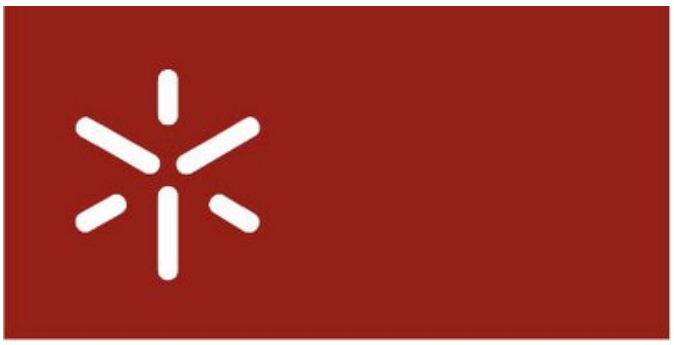


Universidade do Minho

Aplicações Informáticas na Biomedicina

Trabalho Prático - Grupo 3

# Realização de urgências gerais num determinado hospital nacional

**Working Group:** Filipe Fortunato - A75008
João Leal - A75569
José Sousa - A74678
Ricardo Canela - A74568

17 de Dezembro de 2019

# Conteúdo

# 1 Introdução

Este trabalho foi realizado no âmbito da disciplina de *Aplicações Informáticas na Biomedicina* e foi-nos pedido que para fazermos, através de um ficheiro *csv*, uma análise sobre "**A realização de urgências gerais num determinado hospital**". Nesse ficheiro estavam presentes os seguintes campos:

- Número de episódio da urgência (identificador único)
- Data e hora de admissão do paciente nas urgências
- Data e hora da alta
- Descrição da especialidade da alta
- Descrição do local
- Descrição da proveniência
- Descrição da causa da entrada nas urgências
- Género do paciente
- Data de Nascimento do paciente

Foi-nos pedido para importarmos esse ficheiro *csv* para uma base de dados em MySQL denominada **BD_URG**, sendo este o ponto de partida para a realização da análise pretendida.

Nas seguintes secções vai ser discutido a criação do modelo dimensional e consequentemente do Data Warehouse (**DW_URG**), as duas alternativas usadas para o povoamento do **DW_URG**, dos indicadores clínicos usados para analisar a informação do Data Warehouse criados usando a ferramenta *Power BI* e também será sugerido uma aplicação para a inclusão de indicadores clínicos.

# 2 Criação do Data Warehouse

Após fazermos a análise da tabela criada na base de dados **BD_URG** decidimos criar as seguintes dimensões para o modelo dimensional:

- Dimensão Especialidade
  - Dimensão criada para guardar a descrição da especialidade.
- Dimensão Local
  - Dimensão criada para guardar a descrição do local.
- Dimensão Proveniência
  - Dimensão criada para guardar a descrição da proveniência.
- Dimensão Género

  - Dimensão criada para guardar o género do paciente.

- Dimensão Causa
  - Dimensão criada para guardar a descrição da causa de entrada nas urgências

- Dimensão Data
  - Dimensão criada para guardar a data e hora de admissão do paciente nas urgências

Depois de identificarmos as diferentes dimensões a usar, passamos para do Data Warehouse do *MySQL Workbench*, que é apresentado de seguida:

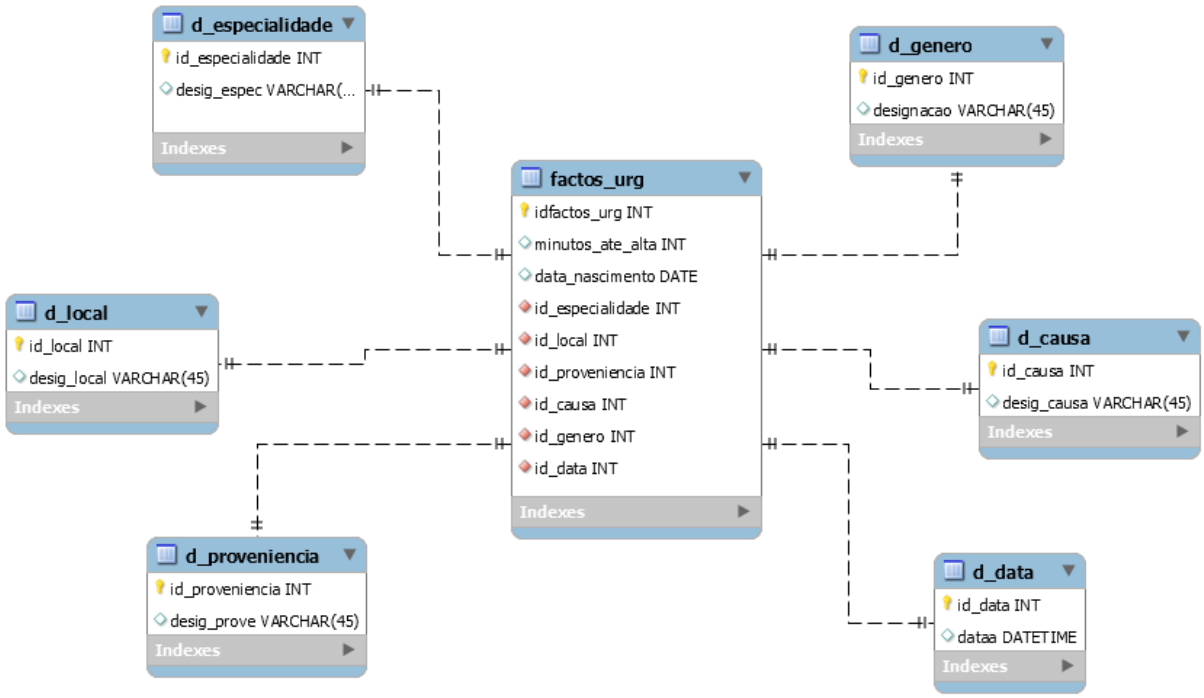



Figura 1: Modelo Dimensional para o Data Warehouse das Urgências de um Hospital

Podemos verificar a informação sobre a data de nascimento e os minutos até o paciente receber a alta estão presentes na tabela de factos e não na tabela das datas, pois nessa tabela deve apenas constar a data de entrada do registo referente ao facto relacionado, sendo neste caso a data e hora de admissão do paciente nas urgências. As restantes dimensões correspondem aos restantes campos presentes. É de salientar que o *id* de todas as tabelas têm a propriedade de *auto-increment*.

# 3 Povoamento do Data Warehouse

Como referido anteriormente, foi-nos pedido para povoarmos o Data Warehouse recorrendo ou a cursores SQL ou a Jobs criados no Talend. De seguida explicamos como foi feito o povoamento para as duas possibilidades.

## 3.1 Povoamento através de Cursores SQL

Para povoarmos o Data Warehouse utilizando cursores SQL criamos um cursor para cada tabela de dimensão e também para a tabela de factos. Os dados usados para o povoamento foram retirados da Base de dados criada inicialmente **BD_URG**, que foi criada importando o ficheiro *csv*.

```
01 |
02 |  DELIMITER $$
03 |
04 |  CREATE DEFINER=`root`@`localhost` PROCEDURE `
       cursor_especialidade`()
05 |
06 |  BEGIN
07 |          DECLARE done boolean DEFAULT FALSE;
08 |
09 |      declare desig_espec varchar(45);
10 |      declare cur1 cursor for select distinct
       ALTA_DES_ESPECIALIDADE from
                              bd_urg.urg_inform_geral;
11 |
12 |      declare continue handler for not found set done = true;
13 |
14 |      open cur1;
15 |
16 |      read_loop: loop
17 |                  fetch cur1 into desig_espec;
18 |
19 |          if done then
20 |                              leave read_loop;
21 |                  end if;
22 |
23 |          insert into dw_urg.d_especialidade(desig_espec)
24 |                      values (desig_espec);
25 |
26 |          end loop;
27 |
28 |          close cur1;
29 |  END$$
```

Listing 1: Cursor para a Tabela de Dimensão d_especialidade

Podemos verificar que primeiramente declaramos os atributos presentes na tabela dimensão de especialidade e de seguida selecionamos através de uma querie SQL a informação que pretendemos retirar da base de dados **BD_URG**. Finalmente criamos um loop, que enquanto existir informação a recolher este insere a mesma na tabela de dimensão no Data Warehouse **DW_URG**.

É de salientar que todos os outros cursores para povoar as tabelas de dimensão são semelhantes ao apresentado em cima, tendo apenas como diferença a informação recolhida da **BD_URG** e a tabela onde a mesma é inserida no Data Warehouse **DW_URG**. Em relação à tabela de factos, esta possui algumas diferenças pois é necessário inserir as *foreign keys* correspondentes às *primary keys* das tabelas de dimensão e precisamos de processar informação para obtermos o

tempo até há alta do paciente, em minutos.

```
01 |
02 |  DELIMITER $$
03 |
04 |  CREATE DEFINER=`root`@`localhost`PROCEDURE `cursor_factos`()
05 |
06 |  BEGIN
07 |  declare done boolean default false;
08 |  declare id int;
09 |  declare minutos_ate_alta int;
10 |  declare data_nascimento date;
11 |  declare ID_ESPECIALIDADE int;
12 |  declare ID_LOCAL int;
13 |  declare ID_PROVENIENCIA int;
14 |  declare ID_CAUSA int;
15 |  declare ID_GENERO int;
16 |  declare ID_DATA int;
17 |
18 |
19 |  declare cur1 cursor for SELECT DISTINCT GERAL.URG_EPISODIO,
20 |  TIMESTAMPDIFF(MINUTE, GERAL.DATAHORA_ADM, GERAL.
       DATAHORA_ALTA),
21 |  GERAL.DTA_NASCIMENTO, DE.id_especialidade, DL.id_local, DP.
       id_proveniencia, DC.id_causa, DG.id_genero, DD.id_data
22 |  FROM dw_urg.d_especialidade DE, dw_urg.d_local DL, dw_urg.
       d_proveniencia DP,dw_urg.d_causa DC, dw_urg.d_genero DG,
        dw_urg.d_data DD, bd_urg.urg_inform_geral GERAL
23 |
24 |  WHERE DE.desig_espec= GERAL.ALTA_DES_ESPECIALIDADE AND DL.
       desig_local=GERAL.DES_LOCAL AND DP.desig_prove = GERAL.
       DES_PROVENIENCIA AND DC.desig_causa=GERAL.DES_CAUSA AND
       DG.designacao=GERAL.SEXO AND DD.dataa=GERAL.DATAHORA_ADM
       ;
25 |
26 |  declare continue handler for not found set done = true;
27 |
28 |  open cur1;
29 |
30 |  read_loop: loop
31 |
32 |    fetch cur1 into id, minutos_ate_alta, data_nascimento,
        ID_ESPECIALIDADE, ID_LOCAL, ID_PROVENIENCIA, ID_CAUSA,
        ID_GENERO, ID_DATA;
33 |
34 |    if done then
35 |          leave read_loop;
36 |    end if;
37 |
38 |  insert into dw_urg.factos_urg(idfactos_urg, minutos_ate_alta
       , data_nascimento, id_especialidade,id_local,
       id_proveniencia,id_causa,id_genero,id_data) values (id,
       minutos_ate_alta, data_nascimento,ID_ESPECIALIDADE,
       ID_LOCAL, ID_PROVENIENCIA, ID_CAUSA,ID_GENERO,ID_DATA);
39 |
40 |  end loop;
```

```
41 |
42 |  close cur1;
43 |  END$$
```

Listing 2: Cursor para a Tabela de Factos factos_urg

Podemos verificar que primeiramente declaramos os atributos presentes na tabela de factos e de seguida selecionamos através de uma querie SQL a informação que pretendemos retirar da base de daldos **BD_URG**, sendo que neste caso precisamos também seleionar as diferentes *primary keys* de todas as tabelas de dimensão e também fazer o cálculo dos minutos até o paciente receber a alta . Finalmente criamos um loop, que enquanto existir informação a recolher este insere a mesma na tabela de dimensão no Data Warehouse **DW_URG**.

## 3.2 Povoamento através do Talend

Para fazermos o povoamento do Data Warehouse criamos 2 jobs distintos, um para povoar as dimensões e outro para povoar a tabela de factos.


Figura 2: Job no Talend para povoar as diferentes tabelas de dimensão

Para povoar as tabelas de dimensão recorremos ao procedimento **tDBInput** que nos permite, definindo nas suas definições uma base de dados a utilizar, criar queries em SQL para recolhermos o campo pretendido da mesma, sendo que no nosso caso utilizamos a tabela **urg_inform_geral** proveniente da base de dados criada com o ficheiro inicial. De seguida segue o exemplo da querie utilizada para povoar a dimensão especialidade :

```
01 |  select distinct urg_inform_geral.ALTA_DES_ESPECIALIDADE
02 |                  from urg_inform_geral
```

Listing 3: Cursor para a Tabela de Factos factos_urg

Esta querie permite-nos então recolher o campo 'ALT_DES_ESPECIALIDADE', obtendo este campo sem que haja qualquer repetido. Para outras dimensões basta alterar o campo a capturar.

Depois de recolhidos os dados pretendidos, criamos um **tMap** para fazer a associação dos resultados das queries com a tabela final a ser criada. Depois desse mapeamento basta-nos criar um **tDBOutput** para podermos guardar os dados recolhidos no Data Warehouse **DW_URG**.

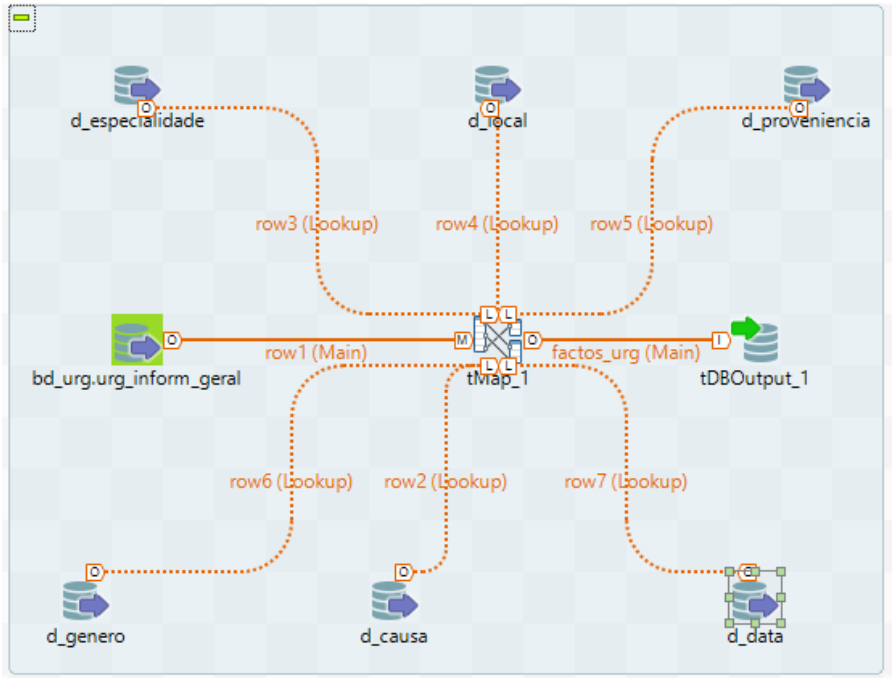



Figura 3: Job no Talend para povoar as diferentes tabela de factos

Para fazermos o povoamento da tabela de factos começamos por usar o procedimento **tDBInput** para podermos recolher a informação das diferentes tabelas de dimensão já criadas, utilizando queries SQL como o exemplo seguinte mostra:

```
01 |  SELECT
02 |    `d_especialidade`.`id_especialidade`,
03 |    `d_especialidade`.`desig_espec`
04 |  FROM `d_especialidade`
```

Listing 4: Cursor para a Tabela de Factos factos_urg

Esta querie permite-nos então recolher da tabela de dimensão *d_especialidade* todos os campos presentes na mesma. Para as outras tabelas basta alterar a tabela em questão e os atributos da mesma. De seguida utilizamos um **tMap** que nos permite realizar uma procura nas tabelas de dimensão correspondentes ao campo obtido na tabela geral, restando apenas fazer a ligação ao Data Warehouse e respetiva tabela de factos através do **tDBOutput**. No tMap fazemos também o cálculo do tempo em minutos até o paciente receber a alta.

```
TalendDate.diffDate(row1.DATAHORA_ALTA,row1.DATAHORA_ADM,"mm")
```

Figura 4: Calculos dos minutos até o paciente receber a alta

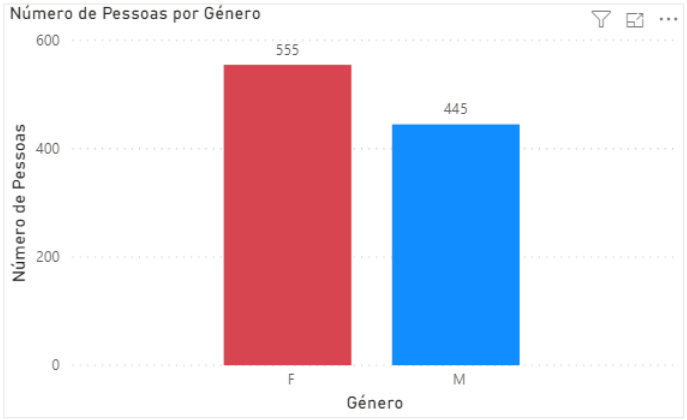
### 3.3 Comparação entre Cursores e Talend

Após a criação dos cursores em SQL para as tabelas de dimensão achamos que foi um processo razoavelmente simples, pois para a criação dos mesmos bastou apenas aceder ao campo a recolher na tabela que continha toda a informação. Já em relação ao cursor da tabela de factos, esse processo já foi mais trabalhoso e complicado pois a querie acabou por ficar com um tamanho bastante grande e os dados a tratar eram em maior número, elevando assim ainda mais a dificuldade do cursor. Posto isto achamos que, da parte do programador, este processo de criação de cursores em SQL para povoar um Data Warehouse acaba por ser um processo ineficiente em termos de tempo gasto.

Depois da utilização da ferramenta Talend achamos que este processo é simplificado, pois não temos tanta preocupação com questões relativas a SQL pois o uso do mesmo é reduzido. A ferramenta possui também uma interface gráfica bastante acessível e de fácil uso e perceção. É nos possibilitado fazer o povoamento de todas as tabelas de dimensão de uma só vez, facto que ajuda na maior simplicidade e rapidez do processo de povoamento. Achamos também que o processamento de datas é mais simplificado nesta ferramenta pois conseguimos mudar o tipo de formato da data muito facilmente.

## 4 Indicadores do PowerBI

Como referido anteriormente, a ferramenta Power BI é utilizada para fazermos uma análise, usando gráficos, dos dados presentes no Data Warehouse. Sendo assim, criamos alguns indicadores de interesse clínico os quais achamos que eram os mais adequados há informação presente.



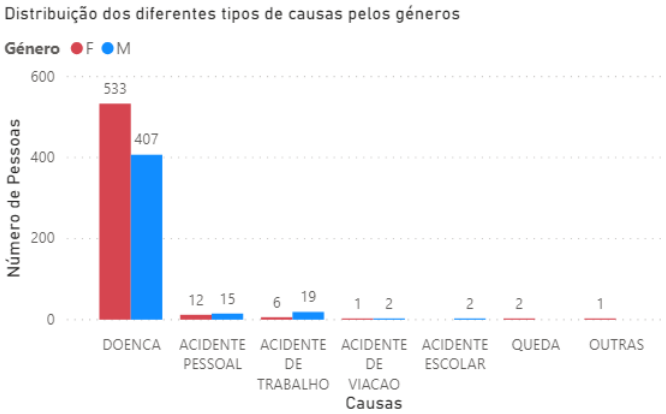
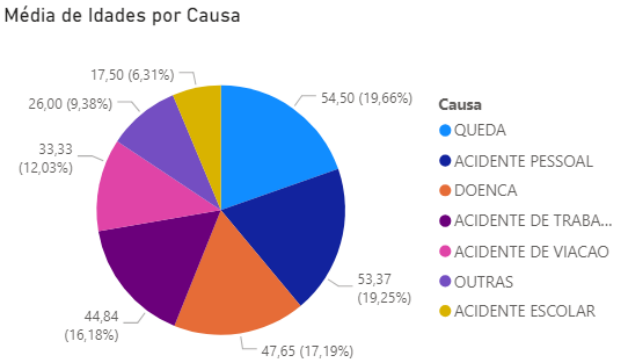




Figura 5: Indicadores sobre a distribuição dos géneros e também a distribuição das causas pelos diferentes géneros

Começamos por criar um indicador que nos ajudasse a perceber qual é a distribuição em termos de género dos pacientes em questão. Fizemos também um indicador para percebermos a distribuição das causas de entrada nas urgências pelos dois géneros o que nos ajuda a entender qual o género que sofre mais das diferentes causas.



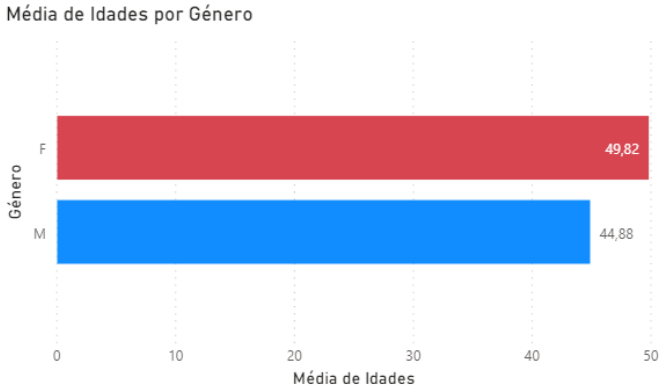




Figura 6: Indicadores sobre a média de idades por causa e por género

De seguida criamos mais dois indicadores, o primeiro sobre a média de idades dos diferentes géneros e o segundo sobre a média de idades dos pacientes pelas diferentes causas de entrada nas urgências, indicador que nos ajuda a perceber qual a faixa etária que é mais afetada pelas diferentes causas, sendo assim possível usar esta informação para prevenção entre os grupos etários.

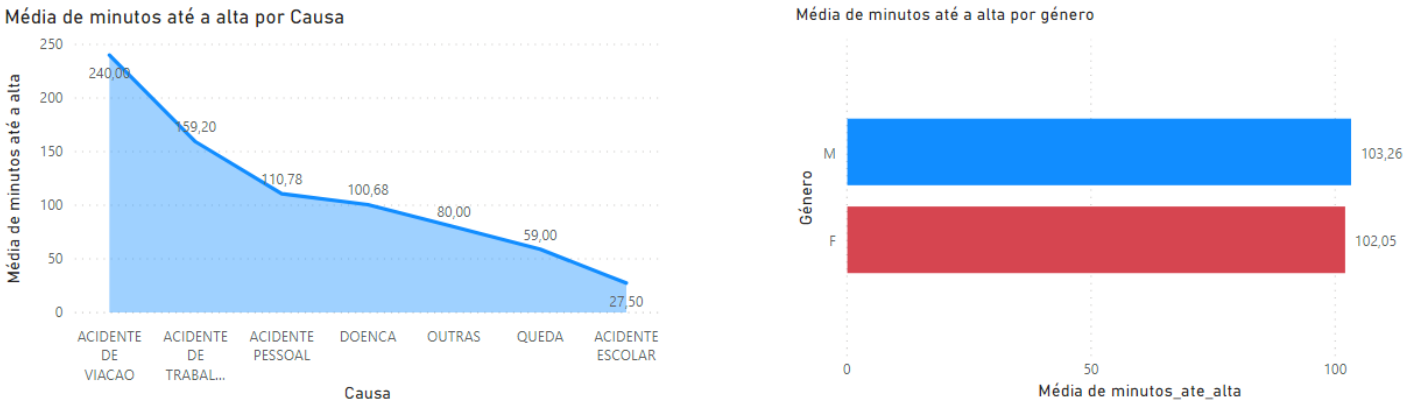



Figura 7: Indicadores sobre a média de minutos até a alta por causa e género

Em ultimo lugar, os dois indicadores seguintes servem para, em primeiro lugar para nos ajudar a perceber qual a causa de entrada nas urgências tem um maior tempo de espera ate a alta, havendo assim a possibilidade de dar prioridade ás causas de entrada nas urgências com um tempo de espera mais alto, e em segundo lugar wajuda-nos a perceber se existe alguma diferença significativa entre o tempo até receber a alta ao compararmos os dois géneros .

# 5 Aplicação com indicadores clínicos

Uma aplicação informática poderia, facilmente, disponibilizar a informação gerada pelo PowerBI, de uma forma bastante prática poderiam ser adicionados e embebidos dashboards numa plataforma web. Seriam então, disponibilizadas informações preciosas de maneira bastante percetível e intuitiva, para facilitar o trabalho da pessoa que está a interpretar essa informação para tomar decisões.

Através da análise dos indicadores clínicos, a administração das urgências possa retirar informação valiosas e aplica-la de forma imediata. Introduzindo assim um sistema que contribui para um aumento do desempenho das urgências gerais num determinado hospital nacional.

Uma hipótese tecnológica que viabilizava o desenvolvimento desta aplicação, seria a utilização de uma plataforma de desenvolvimento web como o NodeJS.

Para tal, a aplicação informática poderia demonstrar e caracterizar os dados por tópicos, cada página iria ter a informação representada em forma de gráfico ou tabela.


Figura 8: Interface da aplicação

# 6 Conclusão

Ao concluir a resolução dos exemplos propostos conseguimos adquirir uma melhor perceção dos métodos de processamento de um conjunto de dados e, de forma complementar, perceção sobre os diferentes resultados possíveis de obter.

A ferramenta, *Talend*, foi importante na resolução dos exemplos permitindo de uma forma, relativamente, simples a criação e ligação dos diferentes componentes úteis ao tratamento de dados (ex: *tMap*, *tFilterRow*, etc.)

Foram portanto adquiridas boas práticas de processamento de um *dataset* assim como experiência com uma ferramenta fundamental para tal, como o *Talend*.