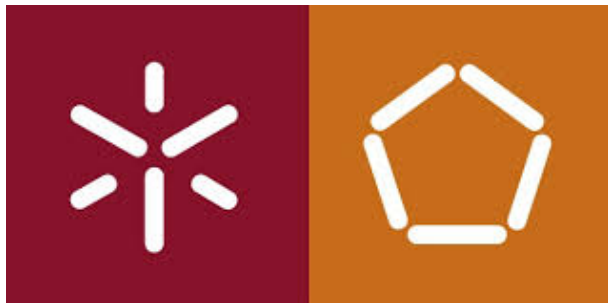


UNIVERSIDADE DO MINHO



Trabalho Prático

Realização de urgências gerais num hospital nacional

MESTRADO INTEGRADO EM ENGENHARIA INFORMÁTICA

APLICAÇÕES INFORMÁTICAS NA BIOMEDICINA
(1º SEMESTRE - 2018/2019)

a70565 Bruno Arieira

a74264 Rafael Silva

5 de Janeiro de 2019

Conteúdo

1	Introdução	3
2	MySQL	4
2.1	Modelo Dimensional	4
2.1.1	Tabelas de dimensão	6
2.2	Tabela de Factos	6
3	Talend	9
3.1	Povoamento	9
3.1.1	Tabelas dimensão	9
3.1.2	Tabela de Factos	11
4	Vantagens e Desvantagens em MySQL e Talend	12
5	Power BI	13
5.1	Idade do paciente por causa	13
5.2	Ocorrência de causas	14
5.3	Tempo médio entre hora de admissão e alta por causa	14
5.4	Contagem por género	15
5.5	Tempo médio por género	15
5.6	Género para cada causa	16
5.7	Interfaces credíveis	16
6	Conclusão	17

Lista de Figuras

1	Representação dos dados no Schema bd_urg	4
2	Modelo dimensional em formato estrela	5
3	Extrato da geração do respetivo Modelo Físico	5
4	Procedimento "PREENCHE_DIM_PROVENIENCIA()"	6
5	Variáveis finais para povoamento da tabela de factos	7
6	Variáveis auxiliares para os cursores	7
7	Declaração dos cursores	7
8	Inicialização do Loop exterior	8
9	Inicialização de um Loop interior	8
10	Atribuição das variáveis finais	8
11	Job no Talend para ETL.	9
12	dim_genero e query em SQL.	9
13	dim_especialidade e query em SQL.	10
14	dim_proveniencia e query em SQL.	10
15	dim_local e query em SQL.	10
16	dim_causa e query em SQL.	10
17	dim_data e query em SQL.	10
18	facts_urg e query em SQL.	11
19	Bases de dados na plataforma Power BI	13
20	Indicador de idade por causa	13
21	Indicador com contagem de cada causa	14
22	Tempo médio de cada paciente por causa	14
23	Contagem por sexo	15
24	Tempo médio por sexo	15
25	Sexo por causa	16

1 Introdução

O presente relatório , juntamente com o trabalho desenvolvido, realizado no âmbito da unidade curricular Aplicações Informáticas na Biomedicina, destina-se a apresentar uma solução coerente relativamente ao pedido no enunciado com principal objetivo de cimentar todos os conceitos abordados nas aulas laboratoriais e teóricas.

Os softwares utilizados para a resolução deste trabalho prático, permitem guardar, manipular, gerar, apresentar e partilhar dados. Para este trabalho prático, foi-nos proposto que apartir de um *dataset* inicial com informações relativas á realização de urgências gerais num determinado hospital nacional, tirássemos o máximo partido de todas as ferramentas e funcionalidades nelas implícitas de modo a gerar resultados que eventualmente possam ser úteis e críticos.

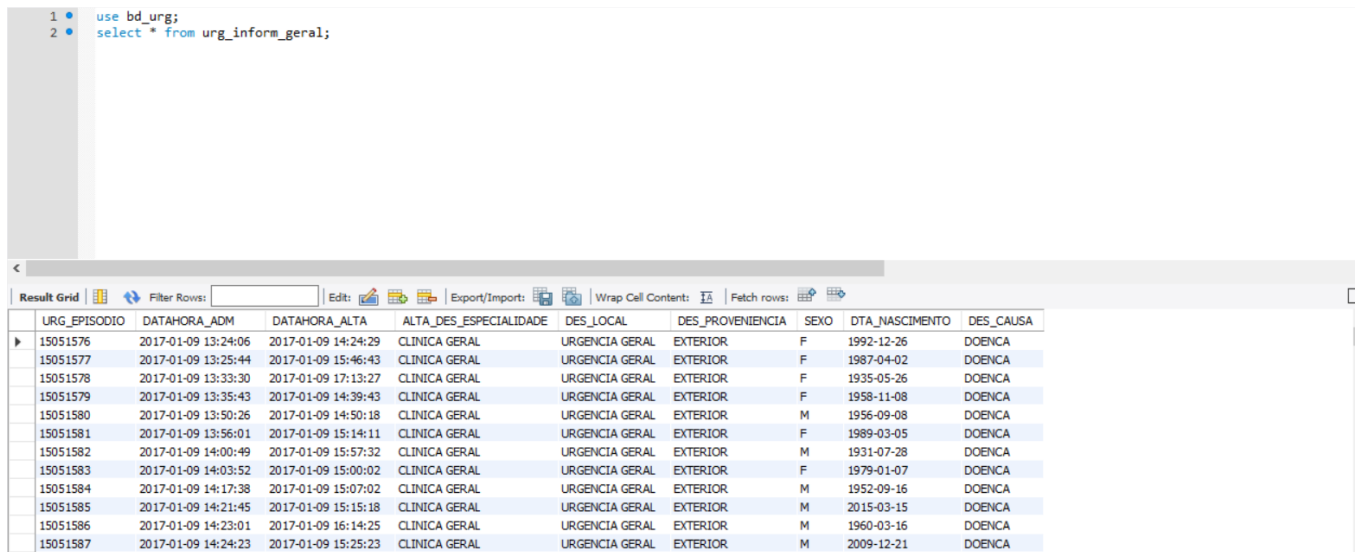
Em suma, este relatório está dividido da mesma maneira que foi feito o processo de trabalho, com os passos que nos foram propostos no enunciado do trabalho prático, todos eles devidamente contextualizados, separando cada secção conforme o software que fomos utilizando. Este relatório finaliza com uma pequena conclusão do grupo face ás dificuldades encontradas assim como aprendizagens sobre o mesmo.

2 MySQL

Inicialmente, foi nos fornecido um *dataset* de formato ".csv", com dados reais da lista de realização de urgências gerais num determinado hospital nacional.

Estes dados contém, relativamente a cada episódio, o número do respetivo episódio da urgência (identificador único), a data e a hora da admissão do paciente nas urgências, a data e a hora da alta, a descrição da especialidade da alta, a descrição do local, a descrição da proveniência, a descrição da causa da entrada nas urgências, o género do paciente, bem como a sua data de nascimento.

Para inicializar o tratamento de dados, tal como pedido no enunciado, criámos um novo *schema* denominando a base de dados de "bd_urg" e importamos todos os dados do *dataset* (urg_inform_geral.csv), através da opção "Schema Transfer Wizard", para a única tabela existente (urg_inform_geral), no novo *schema* criado.



```
1 • use bd_urg;
2 • select * from urg_inform_geral;
```

URG_EPISODIO	DATAHORA_ADM	DATAHORA_ALTA	ALTA_DES_ESPECIALIDADE	DES_LOCAL	DES_PROVENIENCIA	SEXO	DTA_NASCIMENTO	DES_CAUSA
15051576	2017-01-09 13:24:06	2017-01-09 14:24:29	CLINICA GERAL	URGENCIA GERAL	EXTERIOR	F	1992-12-26	DOENCA
15051577	2017-01-09 13:25:44	2017-01-09 15:46:43	CLINICA GERAL	URGENCIA GERAL	EXTERIOR	F	1987-04-02	DOENCA
15051578	2017-01-09 13:33:30	2017-01-09 17:13:27	CLINICA GERAL	URGENCIA GERAL	EXTERIOR	F	1935-05-26	DOENCA
15051579	2017-01-09 13:35:43	2017-01-09 14:39:43	CLINICA GERAL	URGENCIA GERAL	EXTERIOR	F	1958-11-08	DOENCA
15051580	2017-01-09 13:50:26	2017-01-09 14:50:18	CLINICA GERAL	URGENCIA GERAL	EXTERIOR	M	1956-09-08	DOENCA
15051581	2017-01-09 13:56:01	2017-01-09 15:14:11	CLINICA GERAL	URGENCIA GERAL	EXTERIOR	F	1989-03-05	DOENCA
15051582	2017-01-09 14:00:49	2017-01-09 15:57:32	CLINICA GERAL	URGENCIA GERAL	EXTERIOR	M	1931-07-28	DOENCA
15051583	2017-01-09 14:03:52	2017-01-09 15:00:02	CLINICA GERAL	URGENCIA GERAL	EXTERIOR	F	1979-01-07	DOENCA
15051584	2017-01-09 14:17:38	2017-01-09 15:07:02	CLINICA GERAL	URGENCIA GERAL	EXTERIOR	M	1952-09-16	DOENCA
15051585	2017-01-09 14:21:45	2017-01-09 15:15:18	CLINICA GERAL	URGENCIA GERAL	EXTERIOR	M	2015-03-15	DOENCA
15051586	2017-01-09 14:23:01	2017-01-09 16:14:25	CLINICA GERAL	URGENCIA GERAL	EXTERIOR	M	1960-03-16	DOENCA
15051587	2017-01-09 14:24:23	2017-01-09 15:25:23	CLINICA GERAL	URGENCIA GERAL	EXTERIOR	M	2009-12-21	DOENCA

Figura 1: Representação dos dados no Schema bd_urg

2.1 Modelo Dimensional

Depois de criado o *schema* através do ficheiro .csv, procedemos então para o desenho e criação de um modelo dimensional em formato estrela, numa base de dados designada "DW_URG". Para a criação deste modelo, definimos seis tabelas de dimensão e uma tabela de factos (tal como foi sugerido):

- DIM_DATA
- DIM_GENERO
- DIM_ESPECIALIDADE
- DIM_LOCAL
- DIM_PROVENIENCIA
- DIM_CAUSA
- FACTS_URG

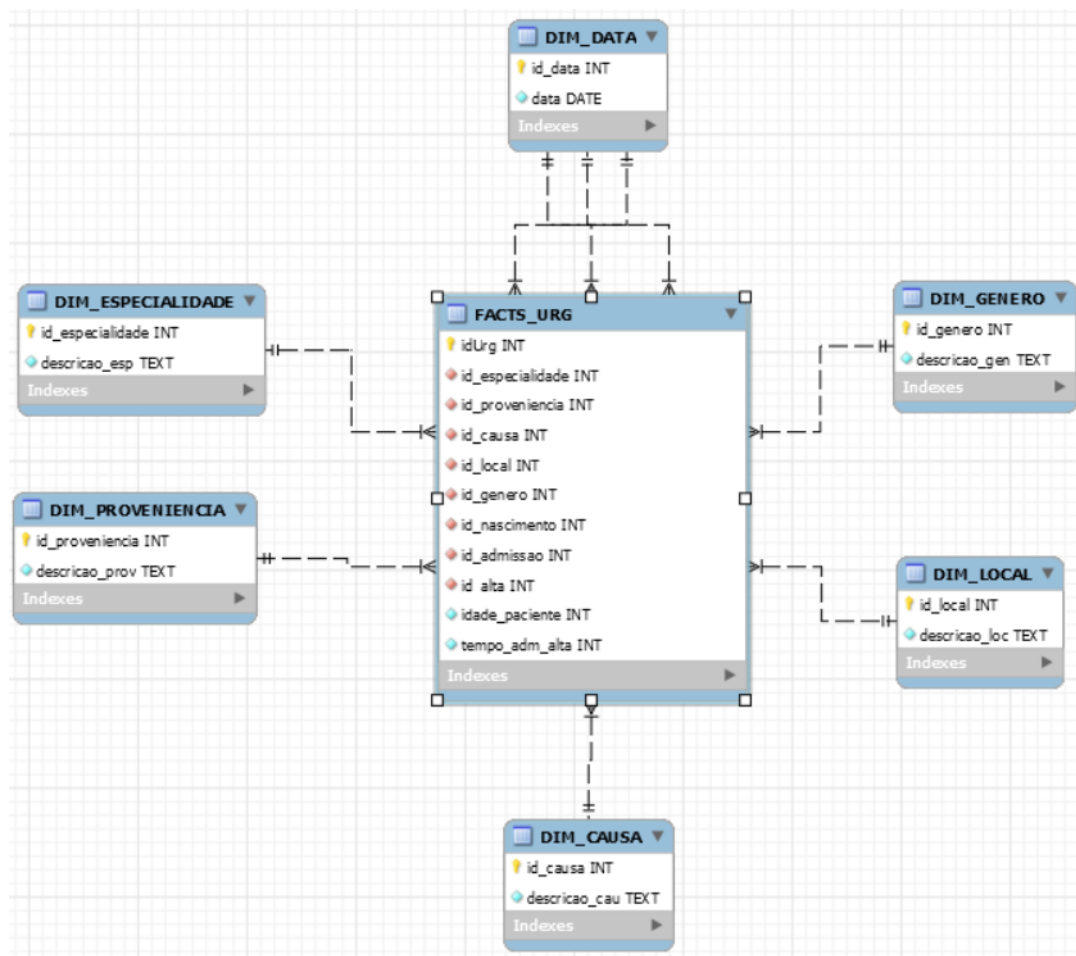


Figura 2: Modelo dimensional em formato estrela

Tal como foi proposto no enunciado depois de todas as tabelas criadas, com os respectivos atributos, procedemos à geração do modelo físico do modelo dimensional, através da opção "*Forward Engineer*".

```
-- Schema DW_URG
-----
CREATE SCHEMA IF NOT EXISTS `DW_URG` DEFAULT CHARACTER SET utf8 ;
USE `DW_URG` ;

-----
-- Table `DW_URG`.`DIM_DATA`
-----
DROP TABLE IF EXISTS `DW_URG`.`DIM_DATA` ;
CREATE TABLE IF NOT EXISTS `DW_URG`.`DIM_DATA` (
  ENGINE = InnoDB;

-----
-- Table `DW_URG`.`DIM_ESPECIALIDADE`
-----
DROP TABLE IF EXISTS `DW_URG`.`DIM_ESPECIALIDADE` ;
CREATE TABLE IF NOT EXISTS `DW_URG`.`DIM_ESPECIALIDADE` (
  ENGINE = InnoDB;

-----
-- Table `DW_URG`.`DIM_LOCAL`
-----
DROP TABLE IF EXISTS `DW_URG`.`DIM_LOCAL` ;
CREATE TABLE IF NOT EXISTS `DW_URG`.`DIM_LOCAL` (
  ENGINE = InnoDB;
```

Figura 3: Extrato da geração do respetivo Modelo Físico

Depois de gerado o modelo físico, passamos então ao povoamento do *data warehouse* criado, utilizando os dados da tabela "urg_inform_geral" da base de dados "BD_URG". Para o respetivo povoamento, começamos por fazê-lo nas tabelas de dimensão e posteriormente, na tabela de factos, através de procedimentos e cursores.

2.1.1 Tabelas de dimensão

Nesta secção iremos abordar apenas um exemplo, devido aos mecanismos a serem utilizados para a construção dos procedimentos e cursores serem semelhantes em todas as tabelas de factos.

Inicialmente, para a criação do procedimento demonstrado na figura, iniciámos por declarar as variáveis que provêm da bases de dados "BD_URG", começando por criar um cursor que irá percorrer, através do ciclo "get_proveniencia: LOOP" todas as linhas com dados da coluna "des_proveniencia" e guardar esses dados na tabela de dimensão "DIM_PROVENIENCIA", acabando quando não houver mais linhas a percorrer ("END LOOP get_proveniencia").

```
DELIMITER //
CREATE PROCEDURE `PREENCHE_DIM_PROVENIENCIA`()
BEGIN
    -- Variáveis
    DECLARE v_finished INTEGER DEFAULT 0;
    DECLARE v_des_proveniencia TEXT;

    -- Cursor
    DECLARE cursor_proveniencia CURSOR FOR
    SELECT DISTINCT des_proveniencia FROM bd_urg.urg_inform_geral
    ORDER BY des_proveniencia ASC;

    DECLARE CONTINUE HANDLER FOR NOT FOUND SET v_finished = 1;

    OPEN cursor_proveniencia;

get_proveniencia: LOOP

    FETCH cursor_proveniencia INTO v_des_proveniencia;

    IF v_finished = 1 THEN
        LEAVE get_proveniencia;
    END IF;

    -- Preencher a tabela DIM_PROVENIENCIA
    INSERT INTO DIM_PROVENIENCIA(descricao_prov) VALUES (v_des_proveniencia);
    COMMIT;

    END LOOP get_proveniencia;

    CLOSE cursor_proveniencia;

END //
DELIMITER ;
```

Figura 4: Procedimento "PREENCHE_DIM_PROVENIENCIA()"

2.2 Tabela de Factos

Tal como anteriormente, em alguns passos onde são tratadas as tabelas de dimensão iremos abordar apenas um exemplo, já que para as restantes o processo a ser usado é semelhante.

Como se demonstra na figura a seguir apresentada, tal como nos passos atrás referidos, iniciamos o desenvolvimento do procedimento para o povoamento desta tabela, começando por inicializar as variáveis que dizem respeito às chaves estrangeiras das respetivas tabelas de dimensões, que serão usadas para guardar os dados finais.

```

/*----- Variáveis -----*/

DECLARE v_finished INTEGER DEFAULT 0;
DECLARE v_idUrgencia INT(11);
DECLARE v_tempo_adm_alta INT(11);
DECLARE v_idade_paciente INT(11);
DECLARE v_cod_local INT(11);
DECLARE v_cod_proveniencia INT(11);
DECLARE v_cod_data_adm INT(11);
DECLARE v_cod_data_alta INT(11);
DECLARE v_cod_data_nascimento INT(11);
DECLARE v_cod_especialidade INT(11);
DECLARE v_cod_causa INT(11);
DECLARE v_cod_genero INT(11);

```

Figura 5: Variáveis finais para povoamento da tabela de factos

Depois das variáveis finais criadas, passamos á declaração de uma espécie de variáveis auxiliares para uma utilização eficiente dos cursores, no momento da sua utilização, que foram feitas para todas as tabelas de dimensão.

```

/*-- Variáveis para os cursores --*/

-- GENERO
DECLARE v_finished_genero INTEGER DEFAULT 0;
DECLARE cod_genero1 INT(11);
DECLARE genero1 TEXT;
DECLARE genero2 VARCHAR(250);

```

Figura 6: Variáveis auxiliares para os cursores

Em continuação, declaramos os cursores propriamente ditos, para todas as tabelas, seleccionando todas as chaves primárias de cada tabela de dimensões, e ordenando pelas mesmas. Relativamente, ao primeiro cursor, selecciona-se as colunas referentes á data de nascimento, hora de admissão e de chegada, para fazer o cálculo da idade de cada paciente relativamente á chegada nas urgências e o cálculo do tempo que o paciente passou nas urgências.

```

/*-- Declaração dos cursores --*/

-- URGENCIA
DECLARE cursor_idUrgencia CURSOR FOR
SELECT DISTINCT URG_EPISODIO,
TIMESTAMPDIFF(MINUTE, DATAHORA_ADM, DATAHORA_ALTA),
TIMESTAMPDIFF(YEAR, DTA_NASCIMENTO, DATAHORA_ADM)
FROM bd_urg.urg_inform_geral
ORDER BY URG_EPISODIO ASC;

-- GENERO
DECLARE cursor_genero CURSOR FOR
SELECT DISTINCT id_genero
FROM DIM_GENERO
ORDER BY id_genero ASC;

```

Figura 7: Declaração dos cursores

Para percorrer todas as linhas, tal como anteriormente, foi necessário fazer um LOOP de todos os cursores, por forma a começar a preencher a tabela de factos, onde se guarda os valores nas variáveis finais. Este tipo de LOOP funciona como um ciclo externo que irá percorrer a tabela da base de dados "BD_URG", como se demonstra na figura a seguir.


```

DECLARE CONTINUE HANDLER FOR NOT FOUND SET v_finished = 1;

OPEN cursor_idUrgencia;

get_idUrgencia: LOOP

SET v_finished = 0;

FETCH cursor_idUrgencia INTO v_idUrgencia, v_tempo_adm_alta, v_idade_paciente;

IF v_finished = 1 THEN
    LEAVE get_idUrgencia;
END IF;

```

Figura 8: Inicialização do Loop exterior

Relativamente a cada iteração do loop exterior, irá ser criado um loop interior para cada cursor de cada tabela de dimensão, onde será verificado o identificador em questão, que está a ser tratado, na respetiva iteração do ciclo exterior. Este processo é feito para todas as tabelas de dimensão a cada iteração.

```

-- GENERO
SELECT SEXO INTO genero1 FROM bd_urg.urg_inform_geral WHERE URG_EPISODIO=v_idUrgencia;
SET v_finished_genero = 0;
OPEN cursor_genero;
get_genero: LOOP
FETCH cursor_genero INTO cod_genero1;
IF v_finished_genero = 1 THEN
    LEAVE get_genero;
END IF;
-- Atribuição de v_cod_genero
SELECT descricao_gen INTO genero2 FROM DIM_GENERO WHERE id_genero = cod_genero1;
IF (genero1 LIKE genero2) THEN
    SET v_cod_genero = cod_genero1;
    SET v_finished_genero = 1;
END IF;
END LOOP get_genero;
CLOSE cursor_genero;

```

Figura 9: Inicialização de um Loop interior

Como passo final desta secção, adicionamos as variáveis finais na tabela de factos e finaliza-se o loop criado, que significa que não existem mais linhas ou colunas a analisar, sendo a tabela de factos preenchida com todos os dados fornecidos da base de dados "BD_URG".

```

-- Preenche a tabela de factos FACTS_URG
INSERT INTO FACTS_URG(idUrg, id_proveniencia, id_admissao, id_alta,
id_nascimento, id_especialidade, id_causa, id_genero, id_local,
tempo_adm_alta, idade_paciente)
VALUES (v_idUrgencia, v_cod_proveniencia, v_cod_data_adm, v_cod_data_alta,
v_cod_data_nascimento, v_cod_especialidade, v_cod_causa, v_cod_genero,
v_cod_local, v_tempo_adm_alta, v_idade_paciente);
COMMIT;

END LOOP get_idUrgencia;

CLOSE cursor_idUrgencia;

```

Figura 10: Atribuição das variáveis finais

3 Talend

Com o uso do Talend para ETL, o processo fica simplificado, visto que, a interface com o Utilizador é mais intuitiva do que com os cursores. Porém, a utilização do Talend não é tao dinâmica como nos cursores. Como o primeiro método a ser utilizado para o povoamento, já se sabia que tipo de queries iria-mos usar para o Talend.

Foram criadas várias expressões diferente em SQL para fazer o povoamento das mesmas tabelas, que se verá nas imagens em baixo.

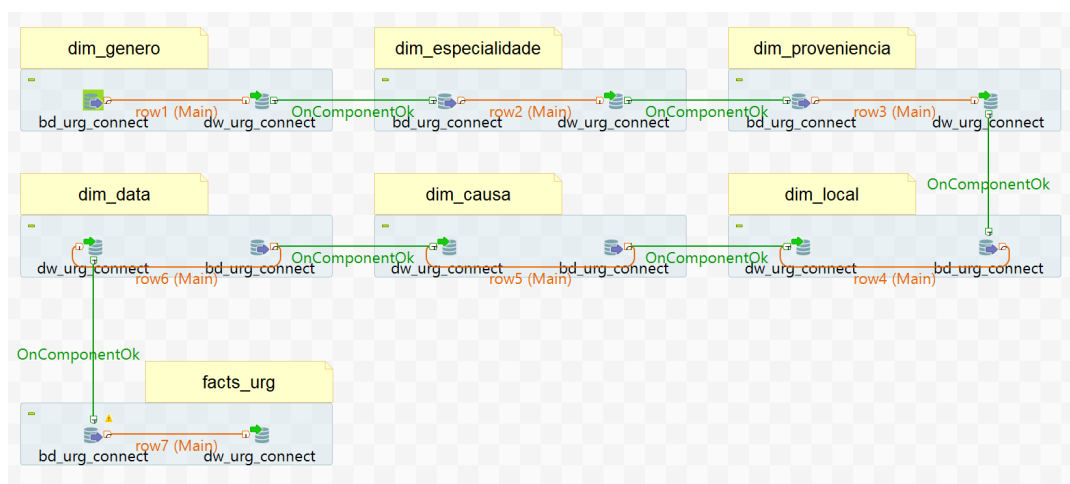


Figura 11: Job no Talend para ETL.

3.1 Povoamento

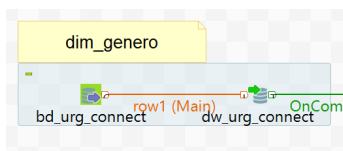
Para povoarmos este Data Warehouse tivemos que usar uma única Job no talend que executa sequencialmente, ou seja, que olhando para a imagem a cima, podemos verificar que a Job se inicia na tabela *dim_genero* e por fim acaba na tabela *facts_urg*.

Cada tabela terá uma query em SQL diferente uma da outra para ir buscar a informação pretendida para cada uma das tabelas.

Isso é feito da seguinte forma: Primeiro acede a base de dados *bd_urg* e depois executa uma query na mesma, essa query depende da tabela dimensão que estaremos a consultar, e de seguida é criada uma conexão a base de dados *dw_urg* em que se passará para esta a informação devolvida pela query em SQL.

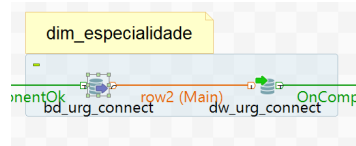
Por fim o Talend cria uma chave primária para essa informação.

3.1.1 Tabelas dimensão



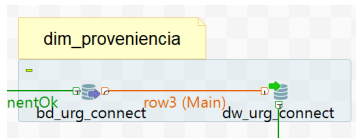
```
"SELECT distinct SEXO from urg_inform_geral"
```

Figura 12: *dim_genero* e query em SQL.



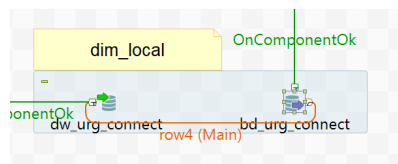
"SELECT distinct ALTA_DES_ESPECIALIDADE from urg_inform_geral"

Figura 13: dim_especialidade e query em SQL.



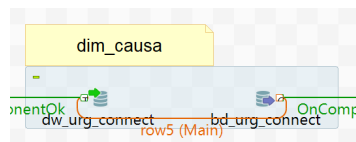
"SELECT distinct DES_PROVENIENCIA from urg_inform_geral"

Figura 14: dim_proveniencia e query em SQL.



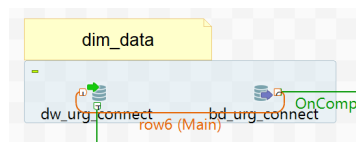
"SELECT distinct urg_inform_geral.DES_LOCAL from urg_inform_geral"

Figura 15: dim_local e query em SQL.



"SELECT distinct DES_CAUSA from urg_inform_geral"

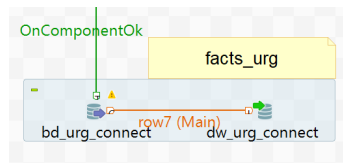
Figura 16: dim_causa e query em SQL.



"SELECT DTA_NASCIMENTO FROM urg_inform_geral
UNION SELECT DATE_FORMAT(DATAHORA_ADM, '%Y-%m-%d') FROM urg_inform_geral
UNION SELECT DATE_FORMAT(DATAHORA_ALTA, '%Y-%m-%d') FROM urg_inform_geral"

Figura 17: dim_data e query em SQL.

3.1.2 Tabela de Factos



```
"SELECT URG_EPISODIO, id_especialidade, id_proveniencia, id_causa, id_local, id_genero,
  (SELECT id_data FROM dw_urg.dim_data
   WHERE data LIKE DATE_FORMAT(bd_urg.urg_inform_geral.DTA_NASCIMENTO, '%Y-%m-%d')) AS id_nascimento,
  (SELECT id_data FROM dw_urg.dim_data
   WHERE data LIKE DATE_FORMAT(bd_urg.urg_inform_geral.DATAHORA_ADM, '%Y-%m-%d')) AS id_admissao,
  (SELECT id_data FROM dw_urg.dim_data
   WHERE data LIKE DATE_FORMAT(bd_urg.urg_inform_geral.DATAHORA_ALTA, '%Y-%m-%d')) AS id_alta,
  TIMESTAMPDIFF(YEAR,DTA_NASCIMENTO,DATAHORA_ADM) AS idade_paciente,
  TIMESTAMPDIFF(MINUTE,DATAHORA_ADM,DATAHORA_ALTA) AS tempo_adm_alta
FROM bd_urg.urg_inform_geral, dw_urg.dim_genero, dw_urg.dim_causa, dw_urg.dim_proveniencia, dw_urg.dim_local, dw_urg.dim_especialidade
WHERE
  bd_urg.urg_inform_geral.ALTA_DES_ESPECIALIDADE = dw_urg.dim_especialidade.descricao_esp
AND
  bd_urg.urg_inform_geral.DES_PROVENIENCIA LIKE dw_urg.dim_proveniencia.descricao_prov
AND
  bd_urg.urg_inform_geral.DES_CAUSA LIKE dw_urg.dim_causa.descricao_cau
AND
  bd_urg.urg_inform_geral.DES_LOCAL LIKE dw_urg.dim_local.descricao_loc
AND
  bd_urg.urg_inform_geral.SEXO = dw_urg.dim_genero.descricao_gen
ORDER BY URG_EPISODIO"
```

Figura 18: facts_urg e query em SQL.

Esta query vai fazer uma pesquisa na tabela `urg_inform_geral` da base de dados `bd_urg`, e nas diferentes tabelas da base de dados `dw_urg` e vai comparar as informações das duas bases de dados para juntar tudo na tabela `facts_urg`, ao mesmo tempo irá calcular a diferença entre a data de alta e de admissão, bem como a idade do paciente quando foram admitidos.

4 Vantagens e Desvantagens em MySQL e Talend

O uso de cursores em MySQL é mais complexo do que o uso do Talend, pois o uso de cursores implica um conhecimento da linguagem SQL profundo.

Ao contrário o Talend para fazer trabalhos de ETL é mais intuitivo, pois oferece uma interface dinâmica ao Utilizador.

Porém o Talend só oferece a maneira de processamento de ETL, enquanto que os cursores fornecem mais manobrabilidade no tratamento dos dados.

Na opinião do grupo, é mais vantajoso para o tema do trabalho em si, e a dimensão do mesmo, isto é, em termos de número de tabelas, o uso do Talend, visto que é mais simples e intuitivo do que o uso de cursores, tomando em conta que o uso de cursores torna o trabalho mais flexível.

5 Power BI

Para conseguirmos criar os indicadores clínicos nesta plataforma, inicialmente foi necessário obter os dados, fazendo a conexão com a bases de dados MySQL onde se encontrava a base de dados "DW_URG". Para isso foi necessária a ligação ao respetivo servidor e introduzir o nome da base de dados em questão, para fazer a consulta das tabelas. Depois disto, podemos entao carregar todas as tabelas implícitas.

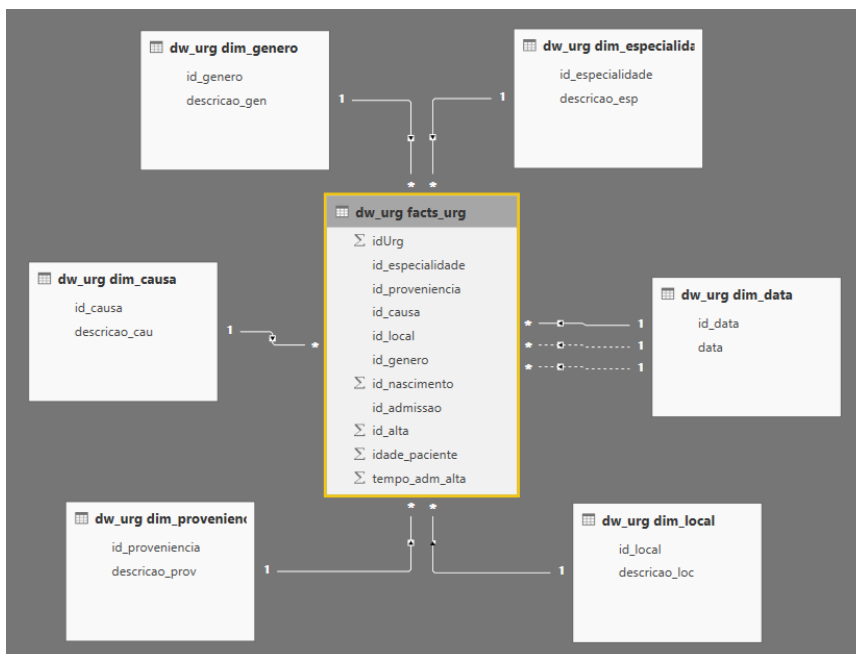


Figura 19: Bases de dados na plataforma Power BI

Depois de feita a conexão, desenvolvemos os indicadores clínicos, como iremos demonstrar, tal como pedido no enunciado.

5.1 Idade do paciente por causa

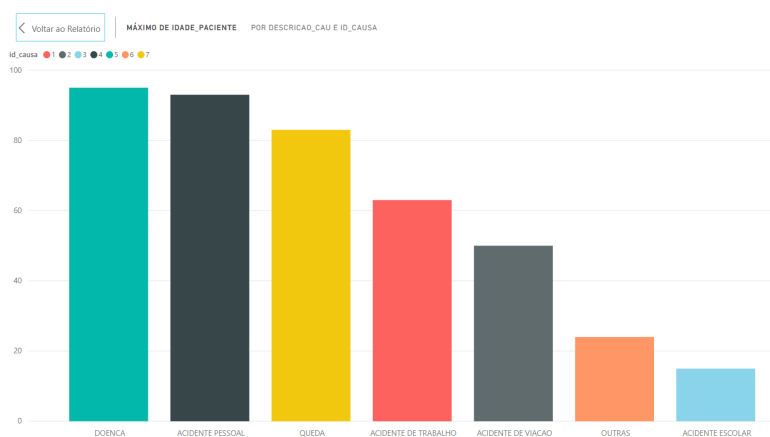


Figura 20: Indicador de idade por causa

Este indicador tem como finalidade de apresentar uma estatística da idade máxima dos pacientes por diferentes causas, que dão entrada nas urgências. Como podemos denotar, para causas relacionadas com doenças, acidentes pessoais e quedas inserem-se frequentemente em pessoas na faixa etária entre os 60 a 95 anos, para acidentes de trabalho e acidentes de viação acontece muito frequentemente a pessoas entre os 25 e 60 anos e para acidentes escolares e outras causas acontece a pessoas entre os 0 a 25 anos. É de notar que a altura das barras relacionam-se com a idade e não com a frequência que acontecem.

5.2 Ocorrência de causas

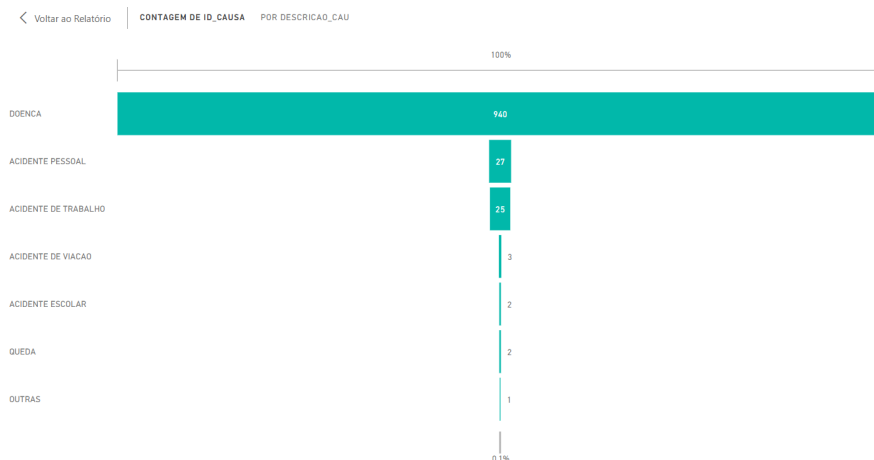


Figura 21: Indicador com contagem de cada causa

Para este caso, encontramos a ocorrência por causa, que dão entrada nas urgências. Como se pode verificar a causa com mais frequência nas urgências é a relacionada com doenças com 940 casos, depois a de acidentes pessoais e acidentes de viação, com 27 e 25 casos, respetivamente, sendo que tem apenas um caso relacionado com outras causas. Este indicador pode ser bastante importante, pois dá uma indicação do número de especialistas médicos que se devem encontrar disponíveis para as situações mais frequentes.

5.3 Tempo médio entre hora de admissão e alta por causa

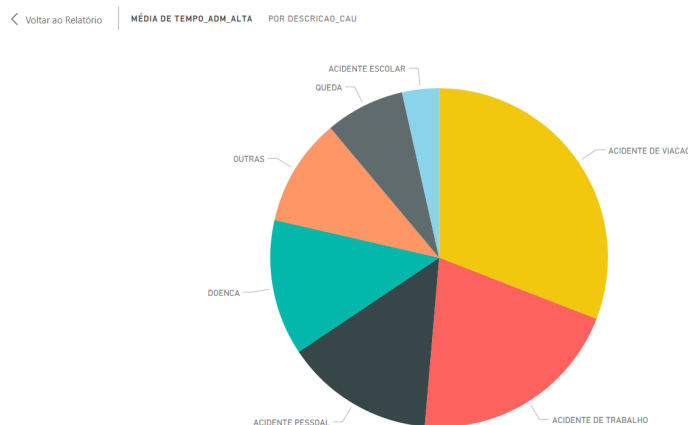
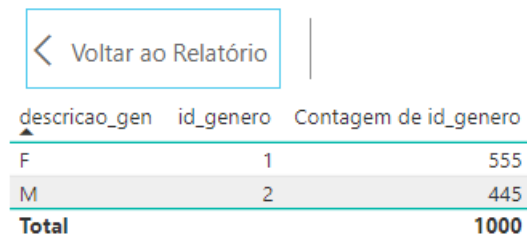


Figura 22: Tempo médio de cada paciente por causa

Com este indicador podemos reparar no tempo médio em que um paciente esteve desde a hora de admissão até á hora em que teve alta para cada causa. Como se pode visualizar, relativamente ás outras causas, o tempo de um paciente que teve um acidente de viação ou um acidente de trabalho, é bastante significativo. O tempo médio para acidentes pessoais, causas relacionadas com doenças e outras são praticamente os mesmos e posteriormente com tempos menores situam-se causas relacionadas com acidentes escolares e quedas.

5.4 Contagem por gênero



A table with three columns: 'descricao_gen', 'id_genero', and 'Contagem de id_genero'. The first row shows 'F' for female with a count of 555. The second row shows 'M' for male with a count of 445. The third row shows 'Total' with a count of 1000. Above the table is a button labeled 'Voltar ao Relatório'.

descricao_gen	id_genero	Contagem de id_genero
F	1	555
M	2	445
Total		1000

Figura 23: Contagem por sexo

Este indicador é relativamente simples e é um indicador estatístico que nos fornece informação, dentro do conjunto de dados que nos foi fornecido, relativamente à contagem de pessoas de sexo masculino e feminino que deram entrada nas urgências de determinado hospital. Como podemos verificar pela figura acima, são mais os pacientes do sexo feminino (555) que do sexo masculino (445), que perfazem um total de 1000 pacientes, ou seja, o *dataset* contém 1000 casos no total.

5.5 Tempo médio por gênero

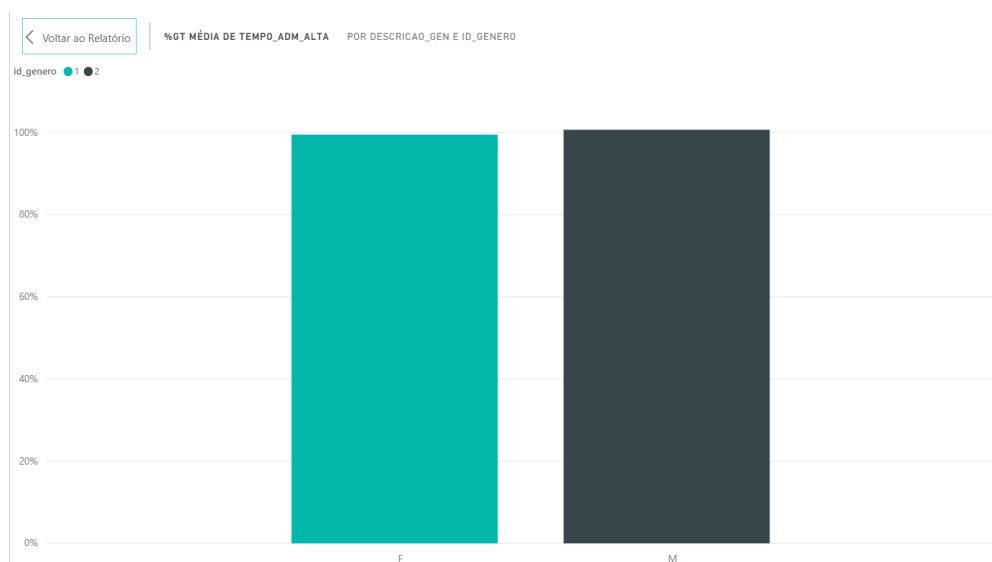


Figura 24: Tempo médio por sexo

Para este caso, é nos indicado o tempo médio, decorrente entre a hora de admissão e de alta de uma paciente, para o sexo masculino e para o sexo feminino. Apesar de já termos informação do sexo feminino ter mais entradas nas urgências, com este indicador podemos analisar que o sexo masculino, além de ser menor, tem um tempo médio substancialmente maior que o feminino.

5.6 Género para cada causa

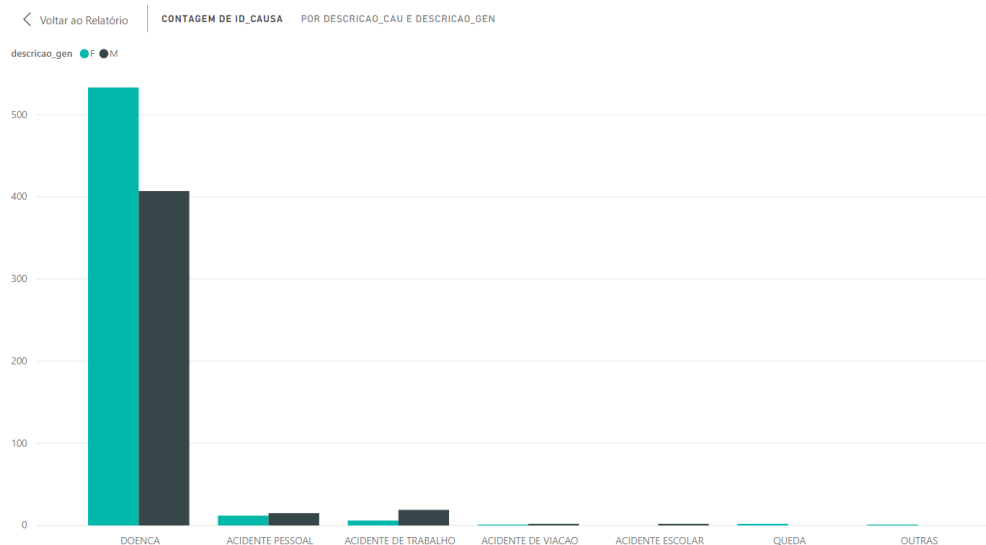


Figura 25: Sexo por causa

Por fim, decidimos elaborar um gráfico de colunas agrupadas que nos dá informação da contagem por género para as diferentes causas dos episódios que ocorrem nas urgências. Tal como já foi descrito anteriormente, a causa mais frequente, neste conjunto de dados é a de doença, onde 533 pacientes são do sexo feminino e 407 do sexo masculino, sendo que com valores menores situam-se as causas relativas a quedas e outras, com 2 e 1 casos, respetivamente, do género feminino.

5.7 Interfaces credíveis

Com estes indicadores criados poderia ser desenvolvida uma aplicação, relacionada com a gestão dos trabalhadores de um determinado hospital. Com o registos dos vários episódios nas urgências, poderia ser criado um indicador que permitisse a análise das maiores ocorrências de causas, e com base nesse factor, gerir a nível de médicos e enfermeiros, quantos destes funcionários especializados eram necessários, de modo a se precaverem. Ainda referente a este indicador, para um leque de causas mais específicas, poderia ser útil por exemplo, na existência de um vírus, em que se analisava de forma coerente o número de pacientes que iam dando entrada.

Uma outra possibilidade de uma interface, seria apartir dos registos dos últimos dados relativos a pacientes que deram entrada e já tiveram alta, informar os utentes que iram ser atendidos, ou possíveis familiares, de uma estimativa de tempo conforme a situação. Essa informação poderia estar a ser anunciada numa televisão numa determinada sala de espera, com uma percentagem relativa ao grau da situação, o tempo médio do paciente, e a causa, ocultando os dados pessoais do paciente em questão. Esta interface seria mais adequada para clínicas, ou para consultas gerais, onde o utente teria uma média do tempo envolvido.

6 Conclusão

Com a realização deste trabalho prático, inserido no âmbito da unidade curricular de Aplicações Informáticas na Biomedicina, podemos concluir positivamente relativamente á execução deste projeto, tanto em relação ás ferramentas utilizadas bem como para o tratamento dos dados, mesmo nos deparando com algumas adversidades.

Relativamente aos problemas durante a execução deste trabalho, focaram-se principalmente na conexão do MySQL com a plataforma Power BI, que ao fazermos as consultas das tabelas dava erro, sendo que posteriormente, com algum tempo, conseguimos superar o problema. Por último, outro problema encontrado e não tão relevante, foi referente ao *dataset*, que como não havia grande diversidade de dados para cada atributo, foi-nos um pouco difícil criar vários indicadores diferentes cada um com dados relevantes e distintos.

No entanto, achámos um trabalho interessante, pois mais uma vez, ofereceu-nos bastante experiência com os softwares desenvolvidos, o que nos proporciona uma bagagem aumentada em termos profissionais.