Solução de Distribuição de Custos

Relatório Final do Projeto Integrador - ARMIS — FEUP

José Pedro Evans de Carvalho

U.PORTO

Licenciatura em Engenharia Informática e Computação

Tutor na U.Porto: Prof. Gil Gonçalves
Orientador na empresa/Proponente: Eng. Sávio Castro

Junho de 2024

Conteúdo

1 Introdução

1.1 Enquadramento

Na sequência do Projeto Integrador da Licenciatura em Engenharia Informática e Computação na FEUP, desenvolvi um estágio curricular com a ARMIS Porto, com o tema "Solução de Distribuição de Custos". Apesar de estar colocado na equipa de Data Management da empresa, juntamente com mais dois alunos da FEUP, o projeto foi realizado individualmente.

Este estágio surge a partir da necessidade por parte da ARMIS de melhorar o processo de distribuição de custos pelos diferentes centros de custo da empresa, aplicando estratégias de integração, transformação e modelagem de dados. Para isso, foi necessário integrar dois sistemas independentes, o sistema de pagamentos e o sistema de imputações, normalizando e transformando os dados para os tornar compatíveis, assim como colmatando falhas existentes nas diferentes tabelas dos sistemas.

Desta forma, através de ferramentas como o Microsoft SSIS, SQL Server Management Studio e Microsoft Power BI, construí uma solução para o problema apresentado, constituindo este relatório a documentação deste processo.

1.2 Objetivos e resultados esperados

- Desenvolvimento de um Data Warehouse com dados de imputações, projetos e despesas.
- Desenvolvimento de um processo ETL para carregamento da Data Warehouse mencionada.
- Desenvolvimento de um relatório Power BI para apresentação dos resultados da distribuição por centro de custos.

1.3 Estrutura do relatório

O relatório está organizado em três partes.

- O capítulo 2 faz o enquadramento do projeto, especificando e documentando cada atividade desenvolvida, no contexto dos sprints planeados.
- O capítulo 3 explora extensivamente a solução desenvolvida, começando por apresentar os requisitos funcionais e não funcionais da mesma. Seguidamente, explora a arquitetura geral do modelo, e de cada uma das tabelas do DW, assim como as tecnologias utilizadas em todas as fases do processo. Finalmente, através de capturas de ecrã regista as soluções produzidas para o utilizador.
- O capítulo 4 apresenta as conclusões do projeto e trabalho futuro.

2 Metodologia utilizada e principais atividades desenvolvidas

2.1 Metodologia utilizada

O estágio teve sempre lugar à sexta-feira, nos escritórios da ARMIS, onde ia sendo acompanhado pelos colaboradores mais experientes. O projeto foi organizado em 6 sprints, com duração variável. Além disso, no início de cada dia de trabalho, reunia com o supervisor para delinear trabalho para o dia, rever o trabalho já feito e esclarecer dúvidas.

2.2 Intervenientes, papéis e responsabilidades

O principal interveniente no acompanhamento do trabalho foi o supervisor por parte da empresa, especialista em Data Analytics, e integrante da equipa de Data Management da ARMIS. Por outro lado, muitos dos requisitos foram estabelecidos pelo diretor do departamento previamente, servindo o mesmo de "stakeholder".

2.3 Atividades desenvolvidas

As atividades desenvolvidas no decorrer do projeto foram ao encontro do plano estabelecido inicialmente:

1. **Exploração das tecnologias a utilizar** (09/02/2024 - 16/02/2024) - Na primeira etapa do projeto, investiguei sobre os paradigmas centrais de bases de dados analíticas, nomeadamente:

   - Distinção entre arquitetura OLTP[1] e OLAP [2], respetivos contextos de utilização e aplicação a bases de dados transacionais e armazéns de dados.
   - O que é e como funciona um processo ETL (extract, transform and load). [3]
   - Definição de Data Lake e Data Warehouse [3]
   - Definição de Business intelligence, qual a sua utilização, relevância na atualidade e principais ferramentas de trabalho. [4]
   - Definição dos conceitos de modelo estrela [5], modelo snowflake [6], e respetivos casos de utilização, assim como os conceitos de facto e dimensão [7].

   Esta fase de investigação foi fulcral para o resto do trabalho, já que ao longo do percurso da LEIC não tivemos contacto com estes conceitos, mas estes revelaram-se essenciais para o desenvolvimento do projeto.

2. **Definição de um modelo de dados que combine a informação pretendida** (16/02/24 - 01/03/2024) - Esta etapa teve como foco a análise dos requisitos pretendidos, e construção do modelo de dados adequado à análise dos mesmos. Estes requisitos vão ser explorados em detalhe na próxima secção.

3. **Desenvolvimento da área de staging e DW (Microsoft SQL Server On-Premises e Integration Services - SISS)** (01/03/2024 - 12/04/2024):

   - 15/03 - 29/03: Nesta fase desenvolvi o processo de ingestão de dados, recorrendo a soluções da Stack Microsoft OnPremises (Integration Services - SSIS), que teve como resultado a área de staging. Foi ainda uma fase de instalação e aprendizagem de software, recorrendo aos tutoriais da Microsoft, e de assimilar os conceitos de control flow e data flow, centrais no desenvolvimento com SSIS [8][9][10][11]. Instalei também SQL Server Management Studio [12], assim como o SQL Server Express , que foram essenciais na manipulação e gestão das bases de dados.
   - 29/03 - 12/04: Iniciei a etapa de transformação de dados recorrendo ao SSIS e aos conceitos explorados anteriormente. Esta foi a fase mais demorada do processo, porque exigiu transformações em dados incompatíveis entre si, originando grande parte dos desafios do projeto, entre os quais gerar dados mock adequados para popular uma base de dados e uniformizar dados de origens diferentes.

4. **Desenvolvimento de um relatório em Power BI[13] para apresentação de resultados** (12/04 - 24/05) :

- 12/04 - 26/05: Investiguei sobre a ferramenta Power BI, nomeadamente DAX [14] e power queries [15], e fiz as configurações necessárias para aceder aos dados locais remotamente [16]. Inicializei também o processo de ingestão do modelo gerado no DW, para a ferramenta de reporting.
- 10/05 - 24/05: Desenvolvimento do relatório em Power BI, recorrendo a métricas e visuais para produzir informação relevante sobre os dados, de acordo com diretrizes dadas pelo supervisor.

5. **Testes e correções necessárias** (24/05 - 14/06): Testes para garantir a validade do modelo construído e a integridade dos dados, assim como feedback por parte do supervisor sobre a usabilidade do relatório e respetivas melhorias.

6. **Redação do relatório** (14/06 - 28/06)

Apesar de todas estas etapas terem sido concluídas, o período de execução não foi sempre estático, sofrendo alterações conforme a disponibilidade, melhor ou pior adaptabilidade aos diferentes tópicos, e dificuldades que foram surgindo.

3 Desenvolvimento da solução

3.1 Requisitos Funcionais

Os requisitos para o projeto foram definidos no início, mas foram sendo adaptados tendo em conta as orientações do supervisor, que desempenhou simultaneamente o papel de cliente.

3.1.1 Desenvolvimento do Data Warehouse

1. Modelagem do Data Warehouse:
   (a) Estruturação do DW recorrendo ao modelo estrela ou floco de neve, utilizando tabelas de fatos e dimensões.
   (b) Definição das tabela facto para despesas e imputações e dimensões para data, centro de custo, projeto, empresa, colaborador e perfil.

2. Integração de Dados:
   (a) Integração com as fontes de dados existentes na empresa: sistema de imputações e sistema de pagamentos (Primavera).
   (b) Desenvolvimento de um processo de ingestão de dados para área de staging.

3.1.2 Desenvolvimendo do Processo ETL

1. Extração de Dados:
   (a) Extração de dados das fontes originais em intervalos regulares.
   (b) Manipulação de dados com origem em bases de dados SQL.

2. Transformação de Dados:
   (a) Limpeza para remover inconsistências e duplicações.
   (b) Transformação de dados para os compatibilizar com o modelo definido para o Data Warehouse.

(c) Implementação das regras de negócio definidas para a distribuição das despesas pelos centros de custo.

3. Carregamento dos dados transformados no Data Warehouse.

3.1.3 Desenvolvimento do Relatório em Power BI

1. Design do Relatório:
   (a) Criação de dashboards interativos para visualização da distribuição das despesas e horas imputadas por centros de custo.
   (b) Implementação de gráficos, tabelas e componentes visuais para análise de dados.
   (c) Desenvolvimento de relatórios detalhados e resumidos em diferentes níveis hierárquicos.
2. Interatividade e navegação:
   (a) Utilização de filtros para permitir a segmentação de dados por período, projeto, empresa, colaborador ou centro de custo.
   (b) Implementação de drill-through e drill-down para explorar detalhes dos dados.
3. Acesso:
   (a) Publicação do relatório no serviço Power BI para acesso remoto.

3.1.4 Validação

1. Verificação da integridade dos dados carregados no DW, utilizando tabelas de controlo.
2. Comparação dos resultados obtidos para as distribuições nas tabelas do DW, com os valores esperados, corrigindo possíveis divergências.

3.2 Requisitos Não Funcionais

3.2.1 Desempenho do ETL e DW

1. Deve ser capaz de processar e carregar os dados dentro de um tempo aceitável.
2. Deve ser escalável para acomodar um aumento do volume de dados.

3.2.2 Usabilidade

1. Interface Power BI intuitiva e de fácil utilização, permitindo fácil navegação entre os dados.
2. Documentação geral do sistema para futura referência.

3.2.3 Manutenibilidade

1. O sistema deve ser de fácil manutenção, nomedaente o código do processo ETL e do DW devem ser estruturados para facilitar a manutenção e realização de alterações futuras.

3.2.4 Tempo de Desenvolvimento

O projeto deve ser concluído durante o segundo semestre, entre 9 de fevereiro e 28 de junho de 2024.

3.3 Arquitetura e tecnologias

3.3.1 Arquitetura do Modelo

Dados os requisitos para este sistema, decidi modelar a base de dados usando o modelo estrela. Este é construído a partir de uma tabela facto central, que contém chaves estrangeiras para as diferentes tabelas dimensão; valores numéricos (horas, pagamentos, descontos e segurança social) e métricas. Por sua vez, as dimensões contêm um ID como chave primária, e informação adicional como restantes atributos. Isto é útil para poupar espaço na tabela facto (mais numerosa a nível de linhas), já que fica apenas guardado um id. As relações são sempre estabelecidas many-to-one, sendo o lado das dimensões unário.

Esta arquitetura é essencial para um modelo de análise de dados, já que permitirá filtrar a tabela facto por cada uma das dimensões.

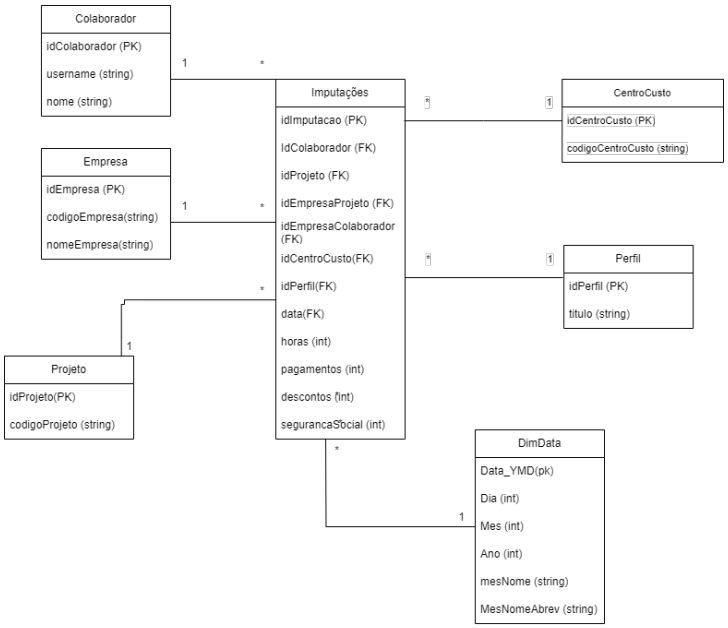

Figura 1: Diagrama UML do modelo desenvolvido

3.3.2 Construção da tabela de movimentos

Como a ARMIS não pode divulgar os valores ganhos por cada colaborador, foi necessário criar dados falsos para preencher a tabela de movimentos do sistema primavera. Para isso criei um script em python com as regras de negócio que me foram indicadas pelo supervisor, em que os códigos iniciados por R correspondiam e vencimentos e os códigos começados por D a descontos. Os valores foram arbitrários, mas incrementando 5%, por trimestre.

3.3.3 Construção das dimensões

Para construir o modelo pretendido foi necessário perceber a origem de dos dados para as diferentes tabelas, assim como que incoerências existiam entre os sistemas e como as colmatar:

1. **Colaborador** - Contém toda a informação útil sobre cada colaborador. Para gerar esta dimensão, foi necessário juntar os utilizadores presentes no sistema primavera (tabela Funcionários), com os utilizadores da tabela de colaboradores da base de dados da ARMIS, e ainda com os utilizadores presentes exclusivamente na tabela de imputações. Uma solução seria simplesmente unir todas as tabelas, mas isso iria ignorar estes problemas, que poderiam ser do interesse do cliente, para compreender que utilizadores estavam registados onde. Assim, decidi uniformizar os id's de cada colaborador de acordo com as seguintes regras fornecidas pelo supervior, por ordem de prioridade:

   (a) p.(numero) - utilizador presente na tabela funcionários do sistema primavera.

   (b) i.(numero) - utilizador que não está na tabela de funcionários do primavera, mas que está na tabela de colaboradores da ARMIS.

   (c) ip.(numero) - utilizador que imputou horas, mas não está presente na tabela de funcionários do primavera, nem na de colaboradores da ARMIS.

   (d) O nome vem das imputações, se vier do primavera (por não estar presente nas imputações), fica com (p) no início (Ex.: (p) Utilizador #315).

   (e) O username vem da tabela Funcionário do Primavera, se existir, caso contrário vem das imputações.

   Atributos da tabela

   (a) idColaborador (PK) - código gerado pela transformação.

   (b) username (String) - username na tabela de colaboradores, ou na coluna funcionário do sistema Primavera.

   (c) nome (String) - nome na tabela de colaboradores. Se estiver presente apenas no sistema primavera, é gerado como (p) + Username. Se estiver presente apenas na tabela de imputações é gerado como (ip) + Username.

2. **Empresa** - Contém as informações sobre as diferentes empresas existentes. No modelo final a tabela facto tem dois atributos distintos relacionados com esta dimensão, isto porque a empresa associada a cada colaborador na tabela de colaboradores era muitas vezes diferente da empresa no qual o utilizador imputou horas. Na prática significa que o colaborador está a fazer um serviço para outra sub-empresa da ARMIS que não aquela em que ele trabalha. Atributos:

   (a) idEmpresa (PK) - chave serializada, única para cada empresa.

   (b) codigoEmpresa (String) - codigo registado na tabela Empresa.

   (c) nomeEmpresa (String) - nome registado na tabela Empresa.

3. **Projeto** - Contém as informações sobre cada projeto. Para tornar possível manter os registos relativos ao passado/futuro de um projeto, este não fica associado a uma empresa/centro de custo na tabela dimensão. Desta forma, se o projeto mudar estes atributos com o tempo, será possível ter um registo temporal sobre o histórico do projeto. Este é um princípio importante a ter em conta na conceção de modelos de BI.

(a) idProjeto (PK) - chave serializada, única para cada projeto.

(b) codigoProjeto (String) - código identificativo do projeto presente na tabela Projeto.

4. **Calendário** - Tabela gerada a partir de um script SQL, que contém informação útil sobre cada dia, desde 2010 até 2029. Determinou-se que a granularidade temporal dos dados iria ser considerada apenas até ao mês. Ou seja, apesar da tabela calendário conter todos os dias dos anos, na tabela facto estarão apenas primeiros dias de cada mês.

   (a) DataYMD (PK) - Data completa no formato date.

   (b) Dia (int)

   (c) Mes (int)

   (d) Ano (int)

   (e) mesNome (String)

   (f) mesNomeAbrev (String)

5. Perfil - Contém a informação sobre cada perfil existente entre os colaboradores. Obtida a partir da tabela Perfil.

   (a) idPerfil (PK) -chave serializada, única para cada perfil.

   (b) titulo (String) - descrição do perfil.

6. Centro de Custo - Contém a informação sobre cada centro de custo. Determinados a partir da coluna Centro de Custo, da tabela Projeto, através da remoção de duplicados desta coluna e carregamento na nova tabela. Representam o centro de custo em que a imputação foi realizada, ou seja, o centro de custo do projeto em questão.

   (a) idCentroCusto (PK)

   (b) codigoCentroCusto (String)

3.3.4 Construção da tabela facto

A tabela facto do modelo serão as imputações feitas pelos utilizadores no sistema. Cada linha representa uma imputação de n horas, feita por um utilizador. Na construção da tabela foi necessário realizar a distribuição dos valores mensais registados na tabela MovimentosPrimavera pelas diferentes imputações do mês, seguindo a seguinte regra de negócio: o valor associado a cada imputação será calculado tendo em conta a percentagem de horas que essa imputação representa no total de horas imputadas nesse mês.

Estão ainda registadas na tabela chaves para todas as dimensões pelas quais a vamos filtrar, permitindo o estudo das medidas de forma multi-dimensional. Por vezes, faltavam dados às imputações, ou seja, para que não fossem violadas as restrições de chave estrangeira, foi necessário criar entradas nas tabelas das diferentes dimensões, correspondentes à inexistência dessa dimensão.

Atributos:

- idImputação (PK) - Chave serializiada, única para cada imputação.
- idColaborador (FK) - Criado a partir da junção da tabela imputações com a tabela colaborador, no atributo username.

- idProjeto (FK) - Criado a partir da junção da tabela de imputação com a tabela de projetos no atributo codigoProjeto.
- idEmpresaProjeto - Criado a partir da junção entre a coluna empresa na tabela do projeto, com a tabela de empresas já serializada e sem duplicados.
- idEmpresaColaborador - Criado a partir da junção entre a coluna empresa na tabela do colaborador, com a tabela de empresas já serializada e sem duplicados.
- idCentroCusto - Criado a partir da junção entre a coluna centroCusto na tabela do projeto, com a tabela de CentroCusto já serializada e sem duplicados.
- data - Primeiro dia do mês da coluna mes da tabela de imputações.
- totalHoras - valor registado na tabela de imputações, correpondente às horas imputadas num determinado contexto.
- SegurancaSocial - Valor calculado seguindo a regra de distribuição, a partir da tabela de movimentos do primavera.
- Descontos - Valor calculado seguindo a regra de distribuição, a partir da tabela de movimentos do primavera, assume sempre valor negativo.
- Vencimentos - Valor calculado seguindo a regra de distribuição, a partir da tabela de movimentos do primavera.

Ano	Mes	Funcionario	idProjeto	idEmpresaProjeto	idUtilizador	idCentroCusto				
2010	11	utilizador.710	774	1	p.710	12	63	0,75	146,25	
2010	11	utilizador.710	3802	1	p.710	4	21	0,25	48,75	

PRIMAVERA			
2010	11	utilizador.710	195

Figura 2: Exemplo de distribuição

Na figura 2 podemos analisar um exemplo da distribuição pretendida. O sistema primavera (parte a cinzento) registava para novembro de 2010 um valor total de 195€ para o utilizador p.710. Na tabela de imputações registavam-se duas imputaçõees, em 2 projetos distintos (774 e 3802), em 2 centros de custo distintos (12 e 4). A distribuição será feita por projeto, tendo em conta a percentagem de horas que cada projeto representa do total mensal (63+21=84). Desta forma, sendo as 63 horas do projeto 774 representativas de 75% das horas totais de novembro, e as 21 horas do projeto 3802, 25% das horas de novembro, os 195€ ficam distribuídos tendo em conta estas percentagens, ficando 146,25 para o projeto 774 e 48,75 para o projeto 3802.

Em termos de granularidade temporal, o estudo foi feito até ao mês, apesar das imputações terem informações a quase diário. No entanto, granularidades mais baixas exigiriam, por um lado, mais armazenamento, por outro regras de negócio mais complexas que determinassem como fazer a distribuição a este nível. Desta forma, definiu-se como requisito por parte do cliente o estudo temporal só até ao nível do mês.

3.3.5 Gestão da Base de Dados

Como software de gestão e alocação da base de dados foi utilizado o SQL Server Express, aliado ao SQL Server Management Studio. A base de dados resultante foi alocada localmente na minha máquina pessoal, utilizando este software. O acesso às bases de dados da ARMIS foi possível utilizando as devidas credenciais, e estabelecendo a ligação no mesmo software.

Como método de organização da base de dados, criei diferentes esquemas em função da fase do processo ETL em questão. Inicialmente são carregados os dados para o esquema [stg], depois realiza-se uma primeira transformação para o esquema [stg2] e por fim os dados são carregados para o DW, no esquema [DW].

Para aceder remotamente às bases de dados guardadas localmente, tendo o modelo semântico e relatório publicados no Power BI, foi utilizado On-Premises Data Gateway [17].

3.3.6 Integração de Dados e Processo ETL

Como software de integração de dados utilizei o **SQL Server Integration Services** (SSIS) em conjunto com o Visual Studio, recorrendo tanto às tecnologias específicas do software, tais como utilização de scripts em C# para transformação de dados, como a queries SQL. O objetivo foi fazer um pacote com a solução pretendida, e executá-lo periodicamente no SQL Server.

O pacote está organizado por módulos responsáveis pelas diferentes partes dos processos de ingestão, transformação e carregamento.

3.3.7 Redação do Relatório

Para redigir o relatório com visuais e métricas apropriadas foi utilizado o Microsoft Power BI,

3.4 Solução desenvolvida

A solução desenvolvida divide-se em duas grandes partes: o pacote de ingestão e transformação dos dados, e o relatório em Power BI.

Relativamente ao primeiro, foi importante analisar as bases de dados originais para perceber na generalidade que informação continham, e de que forma poderiam ser reorganizadas as tabelas. Além disto, foi importante detetar que tipos de limpeza de dados se teriam de realizar, que inconsistências existiam entre as tabelas e definir formas de as contornar tendo em conta as preferências do cliente. Finalmente, o maior desafio desta fase foi construir a tabela fato, juntando as duas tabelas vindas de sistemas distintos. 3.3.4 e 3.3.3

3.4.1 Integração e Transformação - SSIS

O pacote desenvolvido foi organizado de forma a dividir cada etapa do processo em subpacotes, e os prefixos dos componentes foram nomeados tendo em conta as diretrizes fornecidas por um colaborador da ARMIS que trabalha nesta área [18]. Cada etapa tem uma tabela de controlo dedicada, onde é registado o número de linhas processadas em cada tabela carregada/transformada.

Master - Pacote principal e onde a execução deve ser inicializada. Contém containers sequenciais cuja função é a limpeza das tabelas, para evitar conflitos com múltiplas execuções, e posterior execução de cada package especializado.

Figura 3: Organização do Master Package

Data Ingestion - É o primeiro pacote a ser executado na sequência, e é responsável pela ingestão dos dados da base da ARMIS para as tabelas na área de staging [stg]. Esta passagem é direta sem quaisquer transformações. Num processo real isto é útil porque isola o processo de carregamento dos restantes processos, fazendo com que o sistema fique indisponível menos tempo.

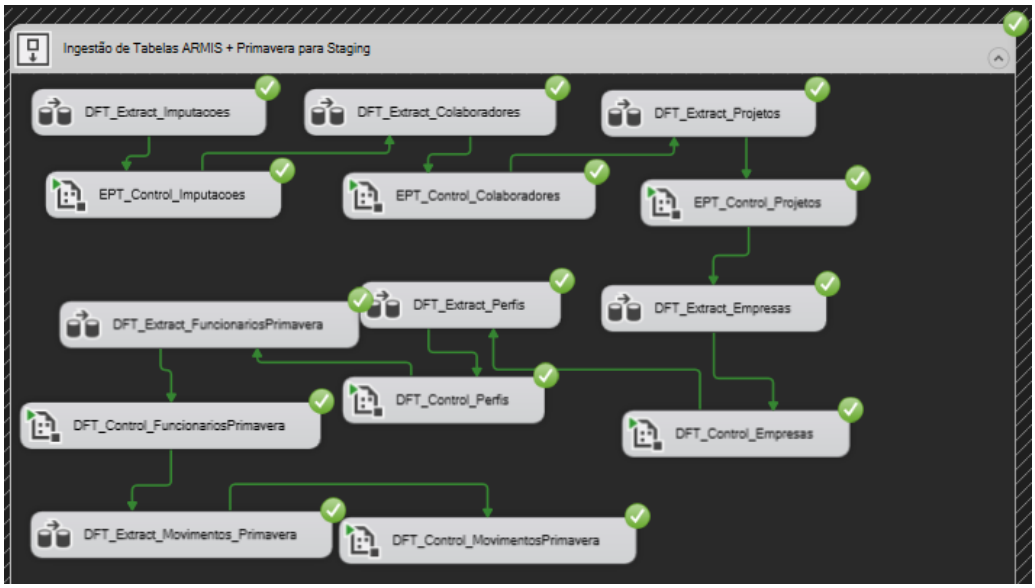

Figura 4: Control Flow do package de ingestão de dados

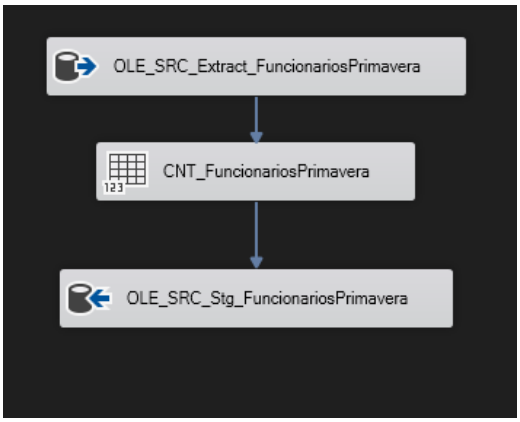

Figura 5: Exemplo de data flow task no processo de extração de uma tabela com origem no sistema primavera, para uma tabela na área de staging. O elemento count rows é importante para manter a tabela de controlo atualizada.

Stage2 Transformation - Este pacote é responsável por grande parte das transformações feitas nas tabelas, desde limpeza de dados, à criação de índices únicos para as entradas das tabelas dimensão. É então a passagem das tabelas do esquema [stg] para o esquema [stg2].

A este nível a transformação mais desafiante foi a da tabela de colaboradores, para seguir as regras já especificadas em 3.3.3. Para isto, dividi o processo em duas tarefas distintas, a primeira que une os colaboradores com os funcionários do sistema Primavera, e a segunda que o une a tabela resultante ([stg2].[Colaboradores]) com os colaboradores vindos da tabela de imputações, dando origem a [stg2].[ColaboradoresMerged]. A estrutura de ambas é semelhante, começando por usar um componente de merge join entre as duas tabelas, para seguidamente usar um componente de script para gerar as novas colunas com os respetivos códigos.

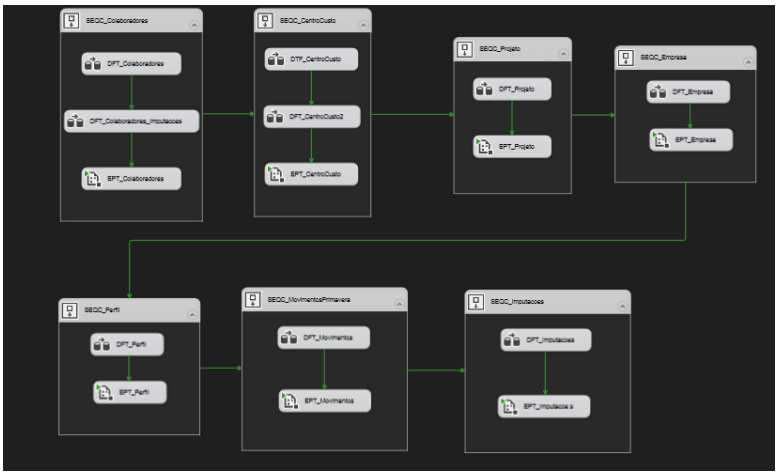

Figura 6: Control Flow da transformação para a stage 2. Todas as linhas processadas são também contadas e registadas na tabela de controlo.

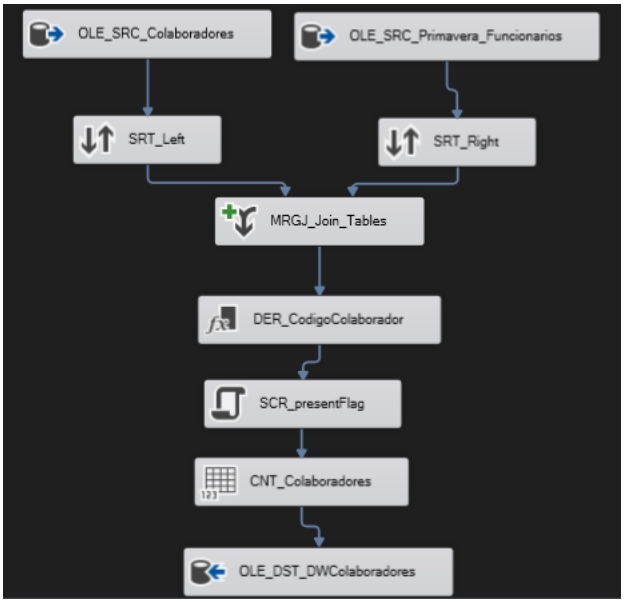

Figura 7: Data Flow Task que une a tabela de colaboradores com a tabela de funcionários do sistema Primavera. Depois de juntar as duas tabelas, executa um script em C# responsável por modificar as colunas necessárias, originando os códigos pretendidos

```
public override void Input0_ProcessInputRow(Input0Buffer Row)
{

    if (Row.numeroimputacao_IsNull == false && Row.numeroprimavera_IsNull == false)
    {
        Row.codigoColaborador = "p." + Row.numeroprimavera;
        Row.Username = Row.usernameprimavera;
        Row.Nome = Row.nomeimputacao;
    }

    else if (Row.numeroimputacao_IsNull == false && Row.numeroprimavera_IsNull == true)
    {
        Row.codigoColaborador = "i." + Row.numeroimputacao;
        Row.Username = Row.usernameimputacao;
        Row.Nome = Row.nomeimputacao;
    }
    else
    {
        Row.codigoColaborador = "p." + Row.numeroprimavera;
        Row.Username = Row.usernameprimavera;
        Row.Nome = "(p)" + Row.usernameprimavera;

    }
}
```

Figura 8: Transformação das colunas na junção da tabela de colaboradores, tendo em conta as regras especificadas em 3.3.3 para esta tabela. Este código é executado para cada linha que passa por este componente de script

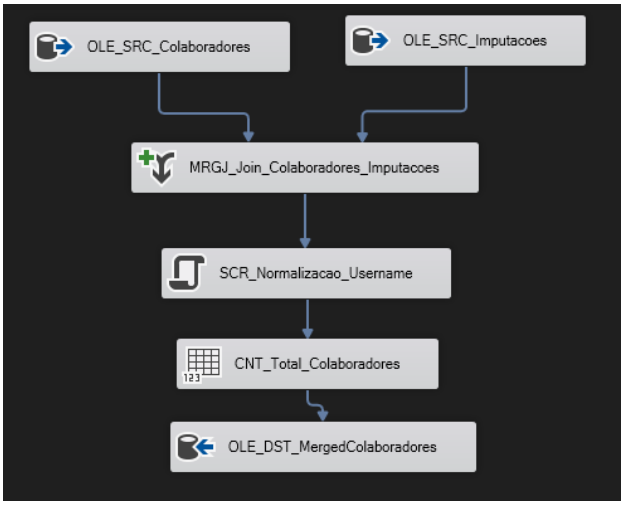

Figura 9: Data Flow Task relativa à união da tabela resultante da transformação anterior com os colaboradores que registaram horas na tabela de imputações.

Todas as tabelas da área de staging foram transformadas para a área de stage 2, tendo-lhes sido aplicadas as transformações necessárias para dar origem às tabelas dimensão com a forma pretendida.

Dimensions Transformation - Este pacote é responsável pelo carregamento dos dados transformados da stage 2 para as tabelas finais no DW. É também aqui que se trata de alterar as tabelas das dimensões para contemplarem entradas no caso da empresa/perfil/centro de custo não existir ou estiver em branco. Isto é feito a partir de scripts SQL antes de cada carregamento. É ainda aqui que a tabela calendário é gerada

e carregada para o DW, o código pode ser consultado no A.1.

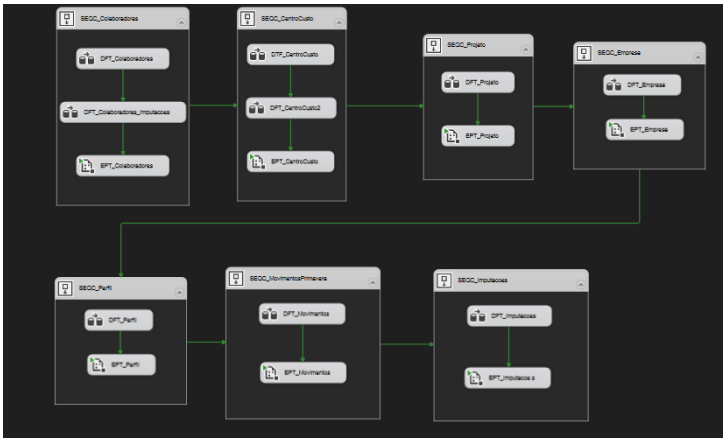

Figura 10: Control Flow do package Dimensions Transformation

Fact Transformation - Este package é responsável pela criação da tabela facto, a partir das tabelas do stage 2. Devido à complexidade da transformação, optei por recorrer a uma query SQL, juntando todas as tabelas presentes na tabela fato e fazendo a distribuição dos valores do sistema de pagamentos.

O código para gerar a tabela facto e a sua explicação resumida podem ser consultados no A.2.

Pacotes de controlo - Utilizados apenas para registar em tabelas o número de linhas que são processadas. Há 3 pacotes distintos (StageControl, TransformationControl,DWControl), mas construídos da mesma forma: um único componente script de SQL que preenche as tabelas de acordo com os parâmetros que lhe são passados.

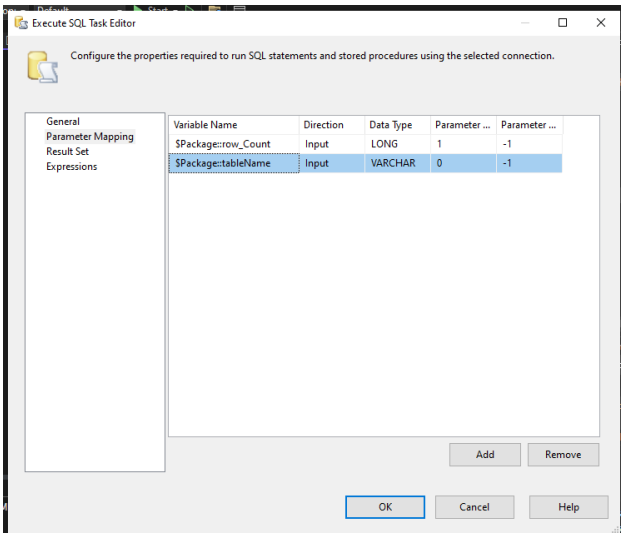

Figura 11: Parâmetros passados para o package de controlo do DW: o contador de linhas e o nome da tabela em questão

```
INSERT INTO [dw].[controlTable] (tableName, row_count,
LoadDate)
VALUES (?, ?, GETDATE())
```

Figura 12: Query executada para inserir os dados na tabela de controlo do DW.

3.4.2 Programação de execução da solução

No **SQL Server Management Studio**, recorrendo ao **SQL Server Agent**, é possível programar a execução periódica do pacote, criando "jobs"

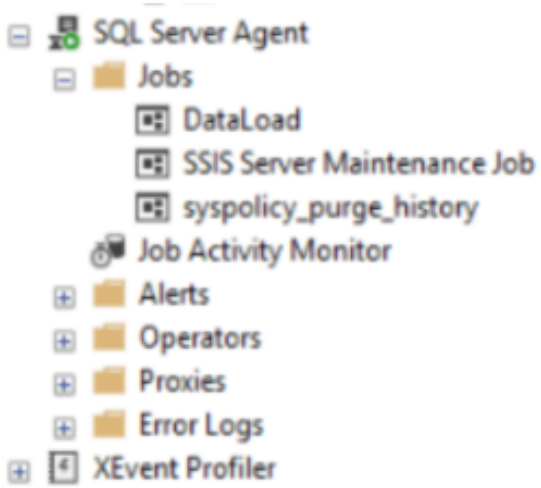

Figura 13: Server Agent e possibilidade de criação de jobs para correr o package periodicamente

Neste caso criámos um job chamado DataLoad, que irá executar o MasterPackage todos os dias, pelas 2h00.

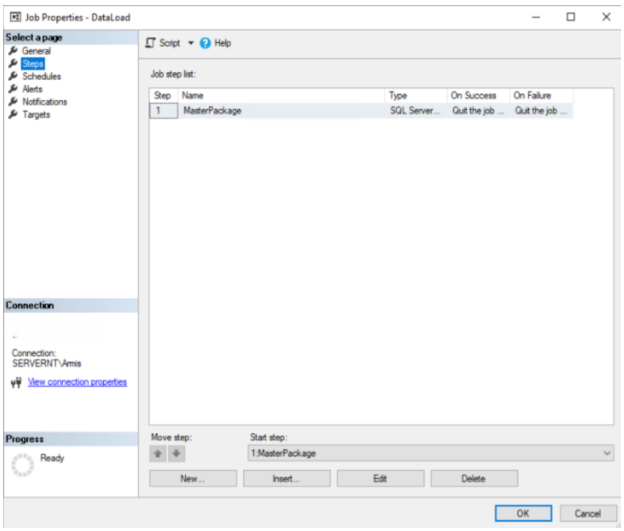

Figura 14: Steps do Job DataLoad, onde é indicada a execução do Master Package.

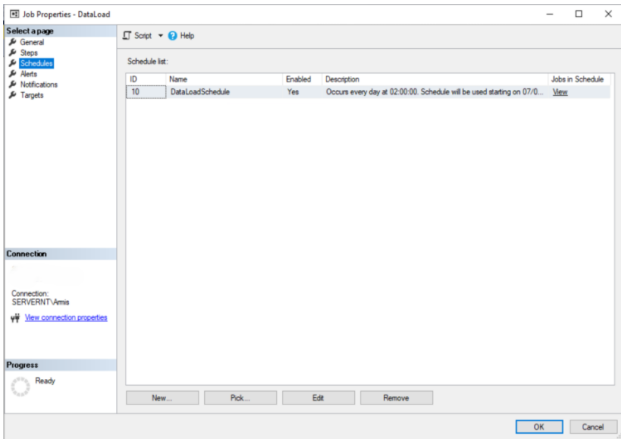

Figura 15: Schedules programados para a execução dos steps, neste caso todos os dias, pelas 2h00

3.4.3 Relatório em Power BI

Inicialmente carreguei as tabelas do DW para o Power BI, utilizando **Power Queries**. Esta ferramenta dá-nos a capacidade de realizar transformações nos dados conforme necessário, mas como o processo de transformação já tinha sido realizado, apenas extraí diretamente as tabelas recorrendo a queries, e garanti que cada atributo assumiu o tipo certo. Além disto, foi necessário estabelecer as relações entre a tabela facto e as dimensões. Por fim, criei na tabela fato as métricas , figura 16, originando assim o modelo semântico de BI, figura 17

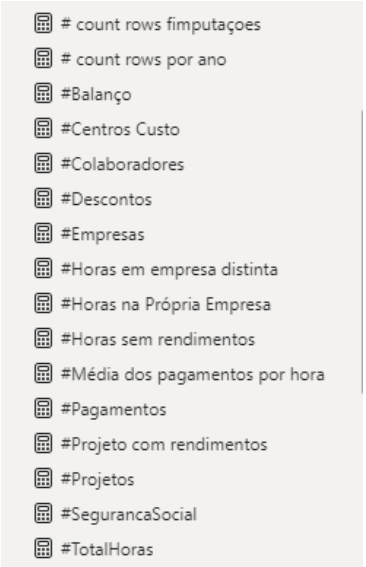

Figura 16: Métricas utilizadas para o estudo dos dados.

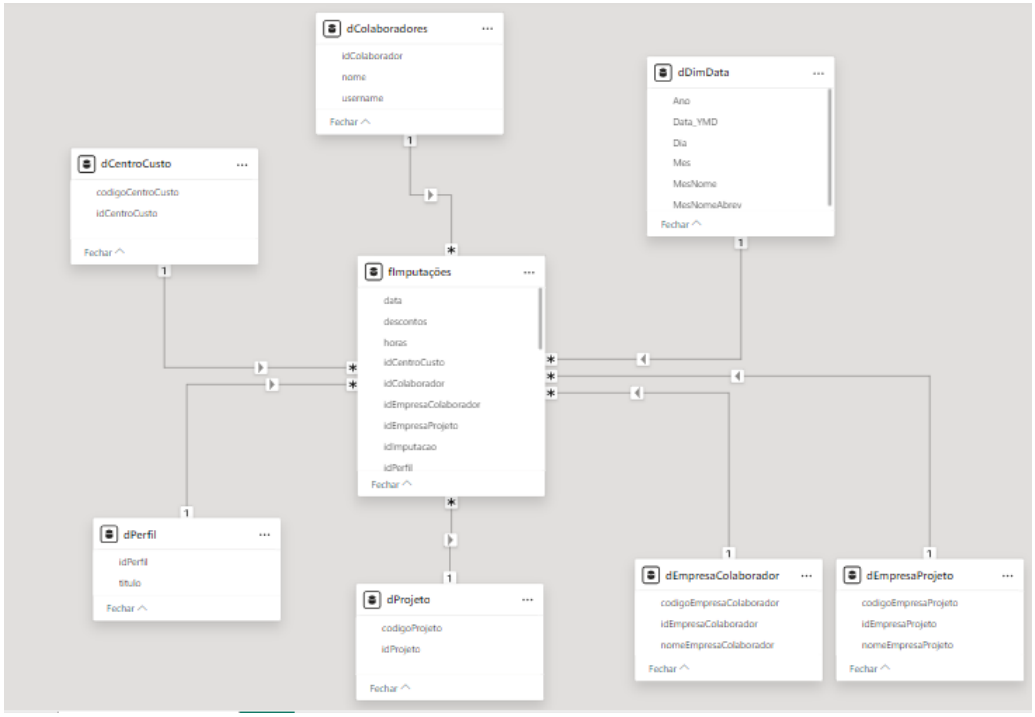

Figura 17: Vista do modelo no Power BI.

Quanto às páginas de análise, todas oferecem a capacidade de filtrar os dados pelas diferentes dimensões, recorrendo a filtros na parte superior da página; visualizar uma contagem das principais métricas; explorar em detalhe qualquer dado na página, sendo reencaminhado para uma página dedicada para este efeito. Existe uma página de Overview, figura 18,com informação sobre os valores gerais da empresa (balanço por empresa, centro

de custo e projeto), além de um visual de distribuição para analisar como se distribui o balanço pelas diferentes dimensões, e ainda páginas dedicadas a cada dimensão. Nestas, estuda-se o caso de um colaborador/projeto/empresa/centro de custo, em particular, percebendo com o tempo como varia o valor de algumas métricas.

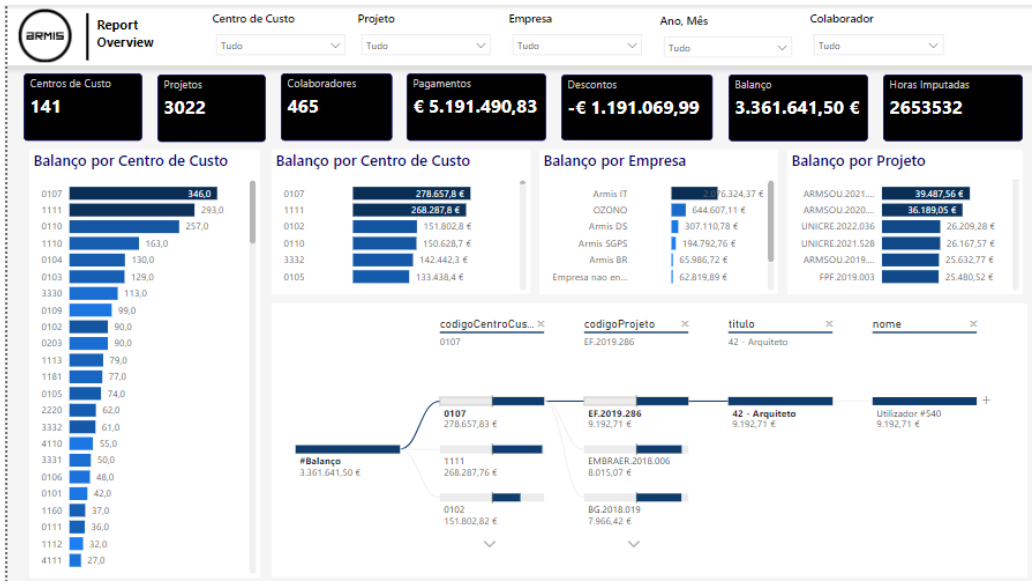

Figura 18: Página de overview. Análise de balanço por diferentes dimensões.

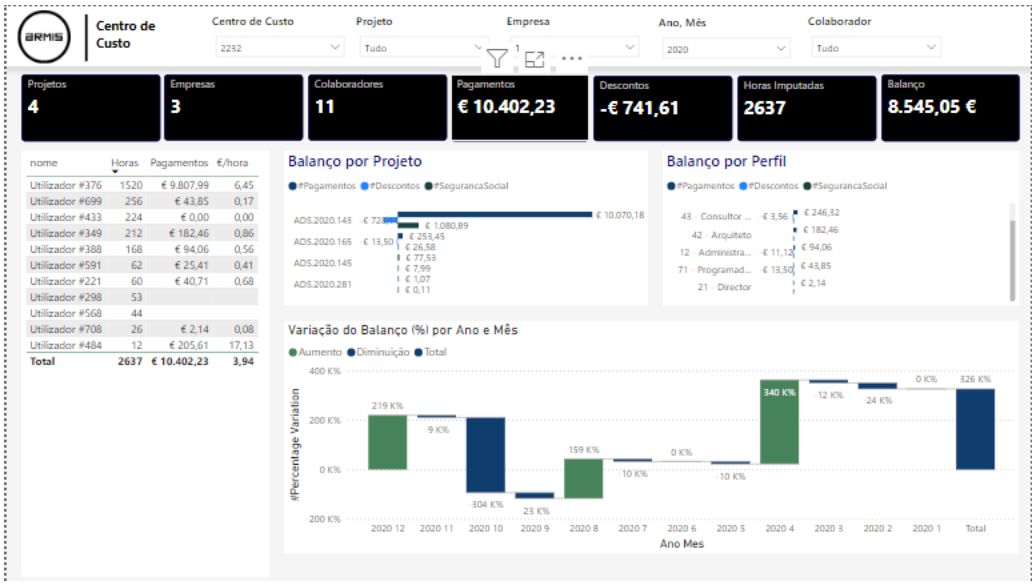

Figura 19: Análise do centro de custo 0103 em 2019 (aplicando filtros), permite perceber o balanço de cada projeto associado a este centro de custo, todos os trabalhadores associados ao mesmo, as horas que imputaram e quanto receberam, os perfis que mais recebem neste c.c. e a variação do balanço, em percentagem, ao longo deste ano

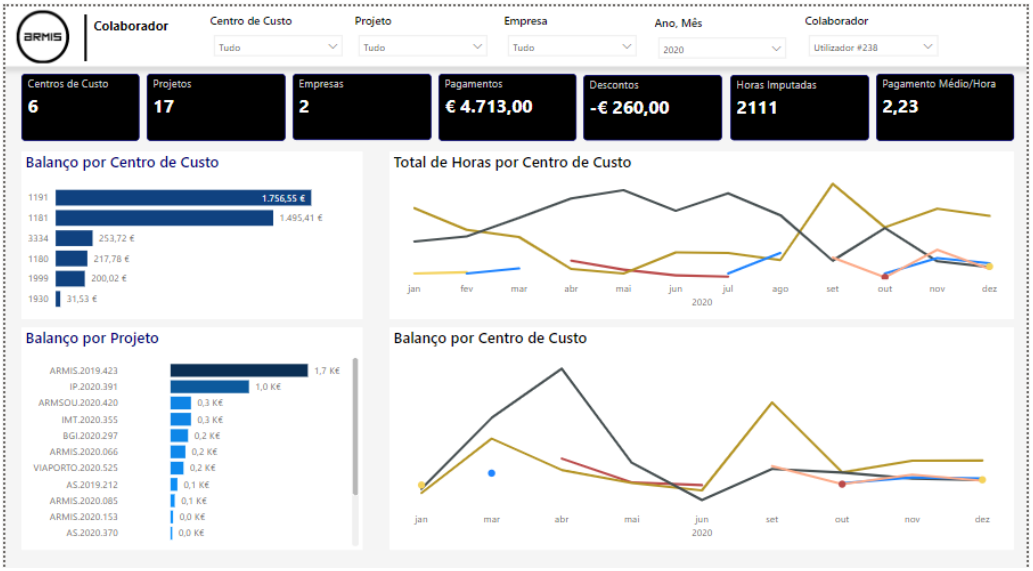

Figura 20: Análise do colaborador 238 no ano de 2020. É possível ver em que centros de custo e projetos o balanço do colaborador foi maior, assim como perceber ao longo do tempo, em que centros de custo o utilizador imputou e recebeu horas, através dos dois gráficos de linhas.

Ano codigoEmpresaColaborador	2020 #TotalHoras	#Pagamentos	#Descontos	#SegurancaSocial	#Balanço	#Média dos pagamentos por hora	Total #TotalHoras	#Pagamentos	#Descontos	#SegurancaSocial
Armis IT	**972**	**€ 2.129,92**	**-€ 145,43**	**€ 227,94**	**1.756,55 €**	**2,19**	**972**	**€ 2.129,92**	**-€ 145,43**	**€ 227,94**
1191	**972**	**€ 2.129,92**	**-€ 145,43**	**€ 227,94**	**1.756,55 €**	**2,19**	**972**	**€ 2.129,92**	**-€ 145,43**	**€ 227,94**
ARMIS.2019.423	965	€ 2.072,56	-€ 145,43	€ 222,18	1.704,95 €	2,15	965	€ 2.072,56	-€ 145,43	€ 222,18
ARMIS.2020.085	7	€ 57,36	€ 0,00	€ 5,76	51,60 €	8,19	7	€ 57,36	€ 0,00	€ 5,76
Total	**972**	**€ 2.129,92**	**-€ 145,43**	**€ 227,94**	**1.756,55 €**	**2,19**	**972**	**€ 2.129,92**	**-€ 145,43**	**€ 227,94**

Figura 21: Esta figura apresenta o detalhe no ano de 2020 do centro de custo 1191 para o utilizador 238, através da interação com o gráfico de barras do balanço por centro de custo da figura 20.

3.5 Validação

Como validação concebi uma queries que confirmam se os valores iniciais de pagamentos e de horas imputadas se mantiveram iguais no final do processo. Para confirmar os valores de pagamentos, agreguei os valores na tabela primavera original, e comparei-os com a soma dos valores das distribuições (A.3). Para validar as horas imputadas, foi apenas necessário comparar a soma das horas na tabela de imputações inicial com a tabela facto final.

	idColaborador	data	rendimento facto	desconto facto	(No column name)	redimento prim	desconto prim
40	p.228	2013-10-01	21.00	-96.00	-96.00	21	96
41	p.550	2014-06-01	155.99	-38.00	-38.00	156	38
42	p.371	2021-04-01	39.00	0.00	0.00	39	NULL
43	p.484	2023-02-01	586.00	0.00	0.00	586	NULL
44	p.463	2020-06-01	491.00	0.00	0.00	491	NULL
45	p.555	2020-11-01	NULL	NULL	NULL	NULL	NULL
46	p.530	2022-02-01	441.00	0.00	0.00	441	NULL
47	p.379	2014-02-01	34.00	0.00	0.00	34	NULL
48	p.407	2018-01-01	130.01	0.00	0.00	130	NULL
49	p.588	2022-09-01	185.00	0.00	0.00	185	NULL
50	p.390	2013-11-01	1592.00	0.00	0.00	1592	NULL
51	p.445	2022-08-01	NULL	NULL	NULL	NULL	NULL
52	p.466	2014-03-01	NULL	NULL	NULL	NULL	NULL
53	p.488	2018-11-01	0.00	-176.00	-176.00	NULL	176
54	p.388	2020-12-01	1697.00	0.00	0.00	1697	NULL
55	p.411	2022-09-01	0.00	-164.00	-164.00	NULL	164

Figura 22: Tabela resultante da query de validação. Para cada utilizador, os valores da fato e do primavera são agrupados e comparados. É necessário fazer uma pequena transformação devido á alteração do tipo de dados. Quando a query é executada apenas para mostrar valores diferentes, retorna 0 resultados, demonstrando a validade da distribuição.

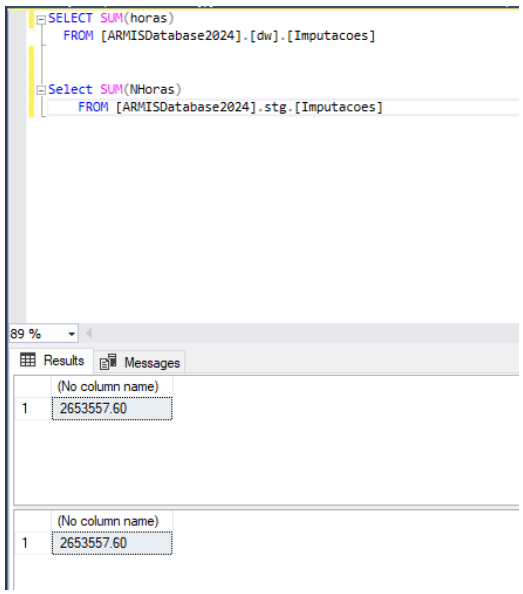

Figura 23: Queries de validação do número de horas inicial e final.

O processo ETL foi acompanhado por tabelas de controlo, que permitem perceber se há linhas que estão a ser duplicadas, ou não estão a ser carregadas. Estas tabelas também se mostraram coerentes.

	id	tableName	row_count	LoadDate
1	1	stg.imputacoes	177648	2024-06-21 13:23:23.467
2	2	stg.colaboradores	431	2024-06-21 13:23:24.700
3	3	stg.projetos	4462	2024-06-21 13:23:26.123
4	4	stg.empresas	9	2024-06-21 13:23:27.293
5	5	stg.perfil	991	2024-06-21 13:23:28.480
6	6	stg.funcionarios_primavera	432	2024-06-21 13:23:28.800
7	7	stg.movimentos_primavera	100000	2024-06-21 13:23:29.517

Figura 24: Tabela de controlo da ingestão de dados

	id	tablename	row_count	LoadDate
1	1	stg2.ColaboradoresMerged	471	2024-06-21 13:23:33.377
2	2	stg2.CentroCusto	207	2024-06-21 13:23:33.960
3	3	stg2.projetos	4462	2024-06-21 13:23:34.223
4	4	stg2.empresas	9	2024-06-21 13:23:34.487
5	5	stg2.movimentos_primavera	58714	2024-06-21 13:23:35.980
6	6	stg2.imputacoes	177648	2024-06-21 13:23:37.367

Figura 25: Tabela de controlo da transformação dos dados para a Stage 2

Finalmente, a usabilidade do relatório desenvolvido foi aprovada por parte do supervisor da ARMIS.

4 Conclusões

4.1 Resultados alcançados

Tendo em conta os objetivos delineados inicialmente, e os requisitos apontados para o sistema, o projeto atingiu bons resultados. Todos os objetivos foram cumpridos, e todas as etapas foram cumpridas dentro do tempo estipulado.

4.2 Lições aprendidas

Em primeiro lugar, o projeto proporcionou-me uma excelente oportunidade para estabelecer contacto direto com o mundo empresarial, oferecendo-me uma visão abrangente da estrutura e do quotidiano de uma empresa tecnológica.

Além disso, o acompanhamento por um especialista que atua diariamente na área foi extremamente valioso. Aprendi muito sobre conceitos de engenharia e ciência de dados, adquirindo uma compreensão global desses processos e do seu funcionamento. Embora o projeto tenha sido realizado em pequena escala, em comparação com o volume real de dados, essa limitação foi constantemente considerada ao longo de todo o processo, algo que me sensibilizou para questões de otimização, decisões relacionadas à arquitetura de

bases de dados e a importância crucial dessas escolhas para o funcionamento adequado dos sistemas.

Finalmente, o desenvolvimento deste projeto mostrou-me a importância do diálogo com o cliente para o desenvolvimento de uma solução adequada. Muitas questões que superficialmente parecem irrelevantes, podem ser importantes para o negócio, e portanto esclarecidas com o cliente.

Em suma, esta experiência foi extremamente enriquecedora, tanto do ponto de vista técnico quanto pessoal. Além de adquirir e aplicar conceitos e boas práticas de engenharia e ciência de dados, ganhei uma nova perspetiva sobre o funcionamento de uma empresa de IT, compreendendo a importância vital da comunicação eficaz com colegas e clientes no desenvolvimento de soluções abrangentes e com qualidade.

4.3 Trabalho futuro

Este trabalho poderá ser aplicado e adaptado a dados reais. O aumento significativo do volume de dados poderia evidenciar algumas ineficiências no processo de transformação que teriam de ser revistas. Ademais, aplicando a solução num contexto real, os cuidados com controlo de erros durante o processo teriam de ser redobrados, acrescentando mais mecanismos que garantissem o correto report de quaisquer anomalias.

Muitas ideias foram simplificadas ao construir este modelo. Por exemplo, quando falamos em análise temporal, pode-se vir a incluir análises a nível do trimestre/semestre, e conceitos de time intelligence. Além disto, numa distribuição de horas real, a nível mensal, questões como os feriados do mês, férias e motivo da falta poderiam ter de ser tidas em conta para adaptar o modelo à realidade da empresa, assim como todas as questões de proteção e segurança dos dados.

Referências

[1] Z. Tejada, "Online transaction processing (oltp) - azure architecture center," Feb. 2023. [Online]. Available: https://learn.microsoft.com/en-us/azure/architecture/data-guide/relational-data/online-transaction-processing#oltp-in-azure

[2] ——, "Online analytical processing (olap) - azure architecture center." [Online]. Available: https://learn.microsoft.com/en-us/azure/architecture/data-guide/relational-data/online-analytical-processing#olap-in-azure

[3] "Data lake vs data warehouse: 6 key differences — qlik." [Online]. Available: https://www.qlik.com/us/data-lake/data-lake-vs-data-warehouse

[4] P. B. Team, "O que é o business intelligence?" [Online]. Available: https://powerbi.microsoft.com/pt-pt/what-is-business-intelligence/

[5] Peter-Myers, "Compreender o esquema em estrela e a importância para o power bi - power bi," Sep. 2023. [Online]. Available: https://learn.microsoft.com/pt-pt/power-bi/guidance/star-schema

[6] "What is snowflake schema - javatpoint." [Online]. Available: https://www.javatpoint.com/data-warehouse-what-is-snowflake-schema

[7] Shsagir, "Fact and dimension tables - azure data explorer," Mar. 2022. [Online]. Available: https://learn.microsoft.com/en-us/azure/data-explorer/kusto/concepts/fact-and-dimension-tables

[8] Dzsquared, "Download sql server data tools (ssdt) - sql server data tools (ssdt)," May 2024. [Online]. Available: https://learn.microsoft.com/en-us/sql/ssdt/download-sql-server-data-tools-ssdt?view=sql-server-ver16

[9] Chugugrace, "Integration services tutorials - sql server integration services (ssis)," Feb. 2023. [Online]. Available: https://learn.microsoft.com/en-us/sql/integration-services/integration-services-tutorials?view=sql-server-ver16

[10] ——, "Control flow - sql server integration services (ssis)," Feb. 2023. [Online]. Available: https://learn.microsoft.com/en-us/sql/integration-services/control-flow/control-flow?view=sql-server-ver16

[11] ——, "Data flow - sql server integration services (ssis)," Feb. 2023. [Online]. Available: https://learn.microsoft.com/en-us/sql/integration-services/data-flow/data-flow?view=sql-server-ver16

[12] Erinstellato-Ms, "Download sql server management studio (ssms) - sql server management studio (ssms)," Apr. 2024. [Online]. Available: https://learn.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16

[13] "Power bi - data visualization — microsoft power platform." [Online]. Available: https://www.microsoft.com/en-us/power-platform/products/power-bi

[14] Kfollis, "Visão geral do dax - dax," Oct. 2023. [Online]. Available: https://learn.microsoft.com/pt-pt/dax/dax-overview

[15] Ptyx507x, "What is power query? - power query," Jan. 2024. [Online]. Available: https://learn.microsoft.com/en-us/power-query/power-query-what-is-power-query

[16] Miquelladeboer, "Install an on-premises data gateway," May 2024. [Online]. Available: https://learn.microsoft.com/en-us/data-integration/gateway/service-gateway-install

[17] ——, "What is an on-premises data gateway?" May 2024. [Online]. Available: https://learn.microsoft.com/en-us/data-integration/gateway/service-gateway-onprem

[18] K. Verbeeck and K. Verbeeck, "Ssis naming conventions 2.0 — under the kover of business intelligence," Dec. 2016. [Online]. Available: https://sqlkover.com/ssis-naming-conventions-2-0/

A Anexos

A.1 Anexo I

```sql
SET DATEFIRST  1,
    DATEFORMAT ymd,
    LANGUAGE   Portuguese;

DECLARE @StartDate  date = '2010-01-01';

DECLARE @CutoffDate date = '2029-12-31';

WITH seq(n) AS
(
  SELECT 0 UNION ALL SELECT n + 1 FROM seq
  WHERE n < DATEDIFF(DAY, @StartDate, @CutoffDate)
),
d(d) AS
(
  SELECT DATEADD(DAY, n, @StartDate) FROM seq
),
src AS
(
  SELECT
    Data_YMD        = CONVERT(date, d),
    Dia             = DATEPART(DAY, d),
    Mes             = DATEPART(MONTH, d),
    Ano             = DATEPART(YEAR, d),
    MesNome         = DATENAME(MONTH, d),
    MesNomeAbrev    = LEFT(DATENAME(MONTH, d), 3)
  FROM d
)

INSERT INTO dw.DimData (Data_YMD, Dia, Mes, Ano, MesNome, MesNomeAbrev)
SELECT *
FROM src
ORDER BY Data_YMD
OPTION (MAXRECURSION 0);
```

Figura 26: Script para gerar a tabela calendário

A.2 Anexo II

```
WITH CTE AS (
    SELECT
        DATEFROMPARTS([Ano], [Mes], 1) AS [FirstDayOfMonth],
        [CodigoProjeto],
        [Username],
        SUM([NHoras]) AS TotalHoras,
        SUM(SUM([NHoras])) OVER (PARTITION BY DATEFROMPARTS([Ano], [Mes], 1), [Username]) AS TotalHorasAnoMesColaborador
    FROM [ARMISDatabase2024].[stg2].[imputacoes]
    GROUP BY DATEFROMPARTS([Ano], [Mes], 1), [CodigoProjeto], [Username]
),
Percentages AS (
    SELECT
        [FirstDayOfMonth],
        [CodigoProjeto],
        [Username],
        TotalHoras,
        CASE
            WHEN TotalHorasAnoMesColaborador = 0 THEN 1
            ELSE TotalHoras * 1.0 / TotalHorasAnoMesColaborador
        END AS Percentage
    FROM CTE
)
SELECT
    c.[FirstDayOfMonth],
    p.idProjeto,
    ep.[id] AS [idEmpresaProjeto],
    col.idEmpresa AS [idEmpresaColaborador],
    col.codigo AS [idUtilizador],
    perf.idPerfil AS [Perfil],
    SUM(mp.rendimentos) AS [TotalRendimentos],
    SUM(c.TotalHorasAnoMesColaborador) as [TotalHorasAnoMes],
    cc.idCentroCusto,
    CAST(SUM(mp.Rendimentos * perc.Percentage) AS DECIMAL(10, 2)) AS [Rendimentos],
    CAST(SUM(mp.SegurancaSocial * perc.Percentage) AS DECIMAL(10, 2)) AS [SegurancaSocial],
    CAST(SUM(mp.Descontos * perc.Percentage) AS DECIMAL(10, 2)) AS [Descontos],
    c.TotalHoras AS [NHoras]
FROM CTE c
LEFT JOIN Percentages perc ON c.[FirstDayOfMonth] = perc.[FirstDayOfMonth]
                          AND c.[CodigoProjeto] = perc.[CodigoProjeto]
                          AND c.[Username] = perc.[Username]
LEFT JOIN [stg2].[projetos] p ON c.[CodigoProjeto] = p.[codigoProjeto]
LEFT JOIN [stg2].[empresas] ep ON p.[idEmpresa] = ep.[id]
LEFT JOIN [stg2].[ColaboradoresMerged] col ON c.[Username] = col.[Username]
LEFT JOIN [stg2].[CentroCusto] cc ON p.centroCusto = cc.codigoCentroCusto
LEFT JOIN [stg2].[perfil] perf ON col.perfil = perf.Perfil
LEFT JOIN [stg2].movimentos_primavera mp ON c.[FirstDayOfMonth] = DATEFROMPARTS(mp.[Ano], mp.[Mes], 1) AND c.Username = mp.Funcionario

GROUP BY
    c.[FirstDayOfMonth],
    p.idProjeto,
    ep.[id],
    col.idEmpresa,
    col.codigo,
    perf.idPerfil,
    cc.idCentroCusto,
    c.TotalHoras;
```

Figura 27: Script para criação da tabela facto. Inicialmente utiliza um CTE para calcular o total de horas trabalhadas num mês por um trabalhador, a partir da tabela de imputações. Depois calcula as percentagens tendo em conta este valor e as horas trabalhadas em cada projeto. Finalmente, junta-se a tabela de imputações (CTE) com as restantes tabelas de dimensões, e realiza-se a distribuição tendo em conta as percentagens calculadas.

A.3 Anexo III

```
with cte as (
select
p.Funcionario
,p.ano, p.MesFiscal
,LEFT(p.CodMov, 1) as mov
,sum(p.Valor) as valor
from stg.movimentos_primavera p
group by p.Funcionario,p.ano, p.MesFiscal,LEFT(p.CodMov, 1)
)
,
cte1 as (
select
f.idColaborador,
f.data,
sum(f.rendimentos) as rend
,sum(f.descontos) as descon
from dw.Imputacoes f
group by f.idColaborador, f.data
)

select
c1.idColaborador,
c1.data,
c1.rend as [rendimento facto],
c1.descon as [desconto facto],
round(c1.descon,0,0),
c.valor as [redimento prim],
c2.valor as [desconto prim]
from cte1 c1
left join cte c on c.Funcionario = REPLACE(c1.idColaborador,'p.','utilizador.')
and c.Ano = year(c1.data) and c.MesFiscal = MONTH(c1.data) and c.mov = 'R'
left join cte c2 on c2.Funcionario = REPLACE(c1.idColaborador,'p.','utilizador.')
and c2.Ano = year(c1.data) and c2.MesFiscal = MONTH(c1.data)  and c2.mov = 'd'
```

Figura 28: Script utilizado para verificar se a distribuição de valores foi feita corretamente