

MESTRADO

MÉTODOS QUANTITATIVOS PARA A DECISÃO ECONÓMICA E EMPRESARIAL

TRABALHO FINAL DE MESTRADO

RELATÓRIO DE ESTÁGIO

**INTEGRAÇÃO DAS FUNCIONALIDADES DO PROGRAMA
SQL SERVER R SERVICES 2016 NA EMPRESA QUIDGEST**

CLÁUDIA MARISA ARAÚJO FREITAS

OUTUBRO 2017

MESTRADO EM
MÉTODOS QUANTITATIVOS PARA A DECISÃO
ECONÓMICA E EMPRESARIAL

TRABALHO FINAL DE MESTRADO
RELATÓRIO DE ESTÁGIO

**INTEGRAÇÃO DAS FUNCIONALIDADES DO PROGRAMA SQL SERVER
R SERVICES 2016 NA EMPRESA QUIDGEST**

CLÁUDIA MARISA ARAÚJO FREITAS

ORIENTAÇÃO:

PROFESSOR RUI MIGUEL BATISTA PAULO
PROFESSOR JOSÉ PEDRO ROMANA GAIVÃO
DOUTOR JOÃO PAULO CARVALHO

OUTUBRO 2017

Agradecimentos

Primeiramente, agradeço ao ISEG pela minha formação académica.

Às Coordenadoras do Mestrado, agradeço todo o apoio, incentivo e disponibilidade, em especial a Professora Margarida Vaz Pato pela paciência e palavras sábias.

Ao Prof.º Rui Paulo e ao Prof.º José Pedro Gaivão agradeço pela proposta de estágio, pelos conselhos e sugestões e, por toda a disponibilidade dispensada ao longo do processo de concretização do relatório.

À Quidgest, os meus mais sinceros agradecimentos por esta oportunidade, em especial o Dr. João Paulo Carvalho que permitiu explorar competências úteis para o meu futuro profissional. Agradeço a todos os colaboradores com quem tive contacto, mostraram-se sempre disponíveis a esclarecer dúvidas durante o estágio.

A todos os meus amigos, agradeço a paciência e compreensão.

À minha irmã, agradeço os conselhos, a força e motivação ao longo deste ano.

Finalmente agradeço aos meus pais que sempre me ensinaram a saber ter calma e ponderar decisões, acreditando sempre nos meus objetivos. Obrigada por realizar mais esta etapa. A vocês, dedico este trabalho.

Resumo

O programa *R Services* é utilizado por empresas de modo a desenvolver capacidades estatísticas, a partir das suas bases de dados. Por outro lado, o programa *Microsoft SQL Server* é um *software* de gestão de bases de dados que permite a criação de código em linguagem de programação, fazendo a atualização automática dos dados. Neste contexto surgiu o programa *Microsoft SQL Server R Services 2016*, que combina integração dos sistemas de gestão de bases de dados das empresas com as análises estatísticas.

O objetivo desta dissertação é explorar o potencial da integração do *Microsoft SQL Server R Services 2016* utilizando as linguagens *Transact-SQL* e *R*, nas quais a partir das competências computacionais e estatísticas permite a criação de procedimentos com código *R* a ser executado dentro do programa *SQL Server*.

De modo a explorar estas potencialidades foi efetuado um estágio curricular na empresa *Quidgest*, no qual se teve acesso à informação necessária de uma base de dados. Neste contexto, foi desenvolvido um caso de estudo, em que se analisou o sucesso desta nova integração dos dois *software* mencionados anteriormente.

Palavras-chave: *Microsoft SQL Server R Services 2016, R Services, Microsoft SQL Server, Base de dados, Programação, Estatística, Transact-SQL.*

Abstract

The *R Services* program is used by companies to develop cognitive and predictive capabilities from their databases. On the other hand, the *Microsoft SQL Server* program is a database management software that allows to create programming code, and automatically updating the data. In this context, the *Microsoft SQL Server R Services 2016* was created, combining the integration of the companies' database management systems with statistical analyses.

The purpose of this thesis is to explore the potential of the integration of *Microsoft SQL Server R Services 2016* using *Transact-SQL* and *R* languages, in which computational and statistical skills allow the creation of procedures with *R* code to be executed within the *SQL program Server*.

In order to explore these potentialities, an internship was carried out at *Quidgest S.A*, from which the necessary information was obtained from a database. From here, a case study was developed, where the success of this new integration of the two software mentioned above, was analysed.

Keywords: *Microsoft SQL Server R Services 2016, R Services, Microsoft SQL Server, Database, Programming, Statistics, Transact-SQL.*

Índice

Agradecimentos	1
Resumo	2
Abstract	3
Índice	4
Índice de tabelas	5
Índice de figuras	5
Glossário	6
Capítulo I: Introdução.....	8
<i>1.1. Quidgest S.A: apresentação, visão, valores e missão.....</i>	<i>8</i>
<i>1.2. Enquadramento do estágio curricular e metodologia.....</i>	<i>10</i>
<i>1.3. Problema proposto e desafios</i>	<i>11</i>
<i>1.4. Resultados obtidos a partir do caso do estudo</i>	<i>12</i>
<i>1.5. Estrutura do relatório de estágio.....</i>	<i>12</i>
Capítulo II: Integração do programa SQL Server R Services 2016.....	13
<i>II.1. Manual de instalação</i>	<i>14</i>
<i>II.2. Operacionalização do código R com linguagem T-SQL</i>	<i>16</i>
Capítulo III: Análise da Base de Dados.....	18
<i>III.1. Clientes e Programas</i>	<i>20</i>
<i>III.2. Recolha de dados.....</i>	<i>20</i>
Capítulo IV: Análises estatísticas em linguagem de programação a partir da base de dados QUI2016.....	24
<i>IV.1. Exposição do Case Study</i>	<i>25</i>
<i>IV.2. Análise exploratória dos dados</i>	<i>25</i>
<i>IV.3. Modelo estatístico adotado</i>	<i>26</i>
<i>IV.4. Resultados</i>	<i>27</i>
<i>IV.5. Desenvolvimento e explicação do código em MSSMS</i>	<i>30</i>
Capítulo V: Conclusões	33

Referências	36
Anexos	i
<i>Anexo 1. Plano de estágio</i>	<i>i</i>
<i>Anexo 2. Manual de instalação do Programa SQL Server R Services 2016</i>	<i>i</i>
<i>Anexo 3. Modelos de regressão linear múltipla</i>	<i>vi</i>
<i>Anexo 4. Código desenvolvido no MSSMS</i>	<i>vii</i>
<i>Anexo 5. Relatório no Report Builder</i>	<i>ix</i>

Índice de tabelas

Tabela 1 - Conjunto das variáveis selecionadas da base de dados QUI2016..	23
Tabela 2 – Representação temporal do plano de estágio	i
Tabela 3 – Modelo 1	vi
Tabela 4 – Modelo 2	vi

Índice de figuras

Figura 1- Configuração para ativar a funcionalidade estatística	14
Figura 2 – Procedimento com linguagem R e T-SQL	16
Figura 3 – Procedimento com código genérico criado com R e T-SQL	17
Figura 4 – Código de criação da tabela global	21
Figura 5 – Exemplo para criar a tabela individual para a variável S1	22
Figura 6 - Exemplo da junção das tabelas individuais em tabela global	23
Figura 7 - Output para a criação de um modelo de regressão linear	32
Figura 8 – Componentes a instalar	i
Figura 9- Configuração do servidor e Collation	ii
Figura 10 - SQL Server Configuration Manager	iv

Figura 11 – Exemplo de lista de serviços do SQL.....	iv
Figura 12 - Reporting Services Configuration Manager: Web Portal URL.....	v
Figura 13 - Conectar o Report Builder ao Reporting Services.....	v
Figura 14 - Alteração do nome das variáveis	vii
Figura 15 - Criação de uma tabela com a identificação de cada variável	vii
Figura 16 – Logaritmização das variáveis	vii
Figura 17 – Criação de nova tabela global com as variáveis transformadas ...	viii
Figura 18 - Armazenamento dos resultados do procedimento dataTransf_16.	viii
Figura 19 - Procedimento para criar o modelo de regressão linear.....	viii
Figura 20 - Relatório no Report Builder.....	ix

Glossário

Ano	<i>Ano</i>
Base de dados QUI2016	<i>QUI2016</i>
Cliente efetivo	<i>Cliefect</i>
Código do Client	<i>Cod_cliente</i>
Investigação e desenvolvimento	<i>I&D</i>
Logaritmo das horas totais	<i>Loghorastot</i>
Logaritmo das vendas	<i>Logvendas</i>
Logaritmo do total dos incidentes	<i>Logtincid</i>
<i>Microsoft SQL Server Management Studio</i>	<i>MSSMS</i>
Número de áreas	<i>Nareas</i>
Número de programas	<i>Nprog</i>

Programa <i>Microsoft SQL Server 2016</i>	<i>Report Builder</i>
<i>Report Builder</i>	
Programa <i>Microsoft SQL Server R Services</i>	<i>SQL Server R Services 2016</i>
<i>Query/Queries</i>	<i>Query</i>
<i>Quidgest S.A.</i>	<i>Quidgest</i>
<i>Reporting Services Configuration Manager</i>	<i>Reporting Services</i>
<i>Revolution E Enterprise</i>	<i>RRE</i>
<i>Revolution R Open</i>	<i>RRO</i>
<i>RStudio/ R Services</i>	<i>R</i>
Sistema de incentivos Fiscais à I&D Empresarial	<i>SIFIDE</i>
Sistema integrado de gestão	<i>SGI</i>
<i>Structured Query Language</i>	<i>SQL Server</i>
<i>Transact-SQL</i>	<i>T-SQL</i>

Capítulo I: Introdução

I.1. Quidgest S.A: apresentação, visão, valores e missão

A *Quidgest S.A.* surgiu em 1988 e caracteriza-se por ser uma empresa multinacional de origem portuguesa, que se iniciou na informatização da Administração Pública, e que está presente em países como Portugal, Alemanha, Marrocos, Timor-Leste ou Moçambique.

A sua atividade centra-se essencialmente, na geração automática de *software* e na inovação e desenvolvimento de sistemas de gestão da informação. Atualmente, é uma empresa de referência nos mercados em Sistemas de Informação que tem como principal objetivo prosperar na “revolução tecnológica, abrir horizontes, fazer e fazer melhor”.

A *Quidgest* apresenta como visão a “inovação, qualidade, participação e desenvolvimento de cada colaborador e adota como política de qualidade “superar as expectativas dos seus clientes mantendo uma reputação de liderança quer nos sistemas de informação que produz como também nos serviços complementares que fornece”¹.

A empresa assegura um *software* designado *GENIO*, inteiramente gerado por código desenvolvido na empresa, onde é possível responder de forma rápida às exigências de desenvolvimento de sistemas de informação para várias plataformas tais como *Web*, *Cloud* e dispositivos móveis.

¹ Informação detalhada no site oficial da empresa: www.quidgest.pt

Os sistemas de *software* desenvolvidos na empresa são adaptados à necessidade de cada cliente a partir da utilização de sistemas de informação únicos e adequados.

As vendas da empresa concentram-se, substancialmente, no desenvolvimento de *software* e na atividade de gestão do envolvimento do mesmo, nomeadamente, fornecimento de serviços de assistência e formação. Deste modo, cada cliente ao comprar um *software* recebe toda a informação relativa à sua utilização, assim como os respetivos serviços de manutenção e assistência. Neste sentido, caso existam novas versões e/ou ocorrência de falhas ou incidentes, é facultado o apoio necessário de modo a suprir as necessidades e exigências de cada cliente, dotando-o assim das melhores capacidades para poder usufruir do produto/serviço adquirido.

A *Quidgest*, no ano de 2016 contava com 101 técnicos que se dividem por áreas e departamentos cujas funções envolvem o desenvolvimento de projetos para a construção de programas, resolução de pedidos de assistência por parte dos clientes, fornecimento de manutenção do *software* e procura e exploração de novas oportunidades de negócio.

Toda a atividade da empresa permanece registada numa plataforma desenvolvida pela própria empresa, o *Quienio*, que utiliza o *software Microsoft SQL Server* para fazer atualizações diárias em *backup*.

O *Quienio* é um sistema integrado de gestão (*SGI*) desenvolvido para a utilização interna dos colaboradores da empresa, com o intuito de se obter informação das tarefas desenvolvidas no âmbito da transação de sistemas de informação. Este é constituído pelo preenchimento de um formulário de trabalho

que solicita indicações, tais como: o nome do projeto ou programa; quantas horas pagas e/ou por pagar; entre outras informações.

Neste contexto, a atualização diária dos dados requeridos a cada colaborador permite consciencializar a empresa relativamente à conjuntura económica proporcionando, também, uma melhor perceção sobre o seu desempenho na atividade, tornando possível uma atuação cada vez mais rápida e eficaz dentro do mercado.

A plataforma *Quiogenio* dispõe do armazenamento de todos os dados que vão sendo registados ao longo da atividade diária da empresa suportado pela utilização do programa de gestão de base de dados: *Microsoft SQL Server*.

I.2. Enquadramento do estágio curricular e metodologia

O estágio iniciou-se em outubro de 2016, teve a duração de 3 meses e enquadrou-se na equipa de Investigação e Desenvolvimento (*I&D*). A oportunidade de estágio curricular surgiu por contacto com os orientadores do mestrado do ISEG, no qual foi proposto a exploração das novas capacidades do programa *Microsoft SQL Server R Services 2016*.

O principal objetivo do estágio foi então o desenvolvimento de competências nas áreas de computação e estatística utilizando o programa desenvolvido pela *Microsoft: SQL Server 2016 R Services* e proceder à investigação das novas funcionalidades, que a mais recente versão do *Microsoft SQL Server* contém em integrar a linguagem de programação *R* amplamente utilizada em estatística.

Estas novas potencialidades foram então exploradas no estágio a partir do desenvolvimento de *scripts* em *R* a serem executados no ambiente *SQL Server*,

de modo a poderem-se fazer análises estatísticas e reportar os resultados, com o intuito de acompanhar/monitorizar o desempenho da empresa *Quidgest S.A.*

Inicialmente, durante o período de adaptação dos colaboradores da empresa, ocorreram situações menos positivas tais como, incumprimento do protocolo ou esquecimento do registo dos dados na plataforma *Quiigenio* por parte de alguns colaboradores. Esta situação explica alguns erros ou falta de dados encontrados neste período experimental, daí que os dados passaram somente a ser rigorosamente utilizados para fins estatísticos a partir do ano de 2014.

A base de dados cedida pela empresa permite o acesso a toda a informação das relações entre clientes e empresa, empresa e colaboradores, e empresa e atividades. Assim, e em consenso com o *CEO* da empresa, Dr. João Paulo Carvalho, foi possível analisar as vendas da empresa e inferir acerca do seu crescimento ao longo dos anos.

I.3. Problema proposto e desafios

O Projeto SIFIDE enquadra-se no Portugal 2020 e apoia as áreas de Investigação e Desenvolvimento (I&D) nas empresas através da atribuição de subsídios. Apresenta especial interesse no desenvolvimento de novas competências das diversas entidades empresariais, as quais se podem candidatar com a finalidade de terem um suporte financeiro para ensaios preliminares à inovação.

No contexto do Projeto SIFIDE, o desafio é estudar sistemas de informação com capacidades cognitivas e preditivas, para que seja possível antecipar resultados de acontecimentos menos favoráveis, quer em grandes empresas, quer em

organismos governamentais e, entendê-los a fim de tomar decisões racionais futuras.

O principal incentivo à realização deste relatório surge a partir da ideia de que os sistemas de informação juntamente com os sistemas de gestão poderão antever necessidades de financiamento, a qualidade de investimento e o impacto de decisões a médio e longo prazo. Assim, surgiu a necessidade de e incorporar conceitos e metodologias (com ajuda das capacidades de computação e estatística) que aproximassem a previsão desse futuro, mesmo tendo em conta que nunca será atingido na sua plenitude.

O aparecimento de sistemas de gestão empresarial possibilitou colmatar algumas limitações de outros sistemas na medida em que permitem a recolha de dados, o seu tratamento e agrupamento com o intuito de estimar de uma forma fundamentada a futura atividade empresarial, antecipando decisões vindouras.

I.4. Resultados obtidos a partir do caso do estudo

O estudo de caso desenvolvido no âmbito deste estágio consiste em analisar um conjunto de variáveis, inseridas na base de dados *Quigenio*, em três processos diferentes: preparação, modelação e operacionalização para um certo período temporal e, estudar os resultados através das funcionalidades do programa *Microsoft SQL Server R Services 2016*.

I.5. Estrutura do relatório de estágio

O relatório de estágio está organizado em cinco capítulos mais anexos.

No capítulo II explica-se a integração do programa *R Services* em *SQL Server*, desde a instalação à operacionalização das linguagens dos dois *software*.

O capítulo III descreve o processo de criação de código para a seleção dos dados das variáveis a analisar.

No capítulo IV realizou-se um estudo de caso que analisa estatisticamente as vendas dos clientes da empresa *Quidgest*, de forma a integrar a linguagem *R* no ambiente de linguagem de programação. Neste capítulo inclui-se, também, descrita a criação de código a utilizar no relatório suportado pela aplicação *Report Builder*.

O capítulo V apresenta os resultados e conclusões retiradas a partir dos capítulos anteriores e revela os desafios possíveis para trabalhos futuros.

Capítulo II: Integração do programa *SQL Server R Services 2016*

O *Microsoft SQL Server R Services 2016* permite armazenar em bases de dados conjuntos de informações e, através de manuseamento, explorar estatisticamente essa mesma informação sem a necessidade de transferir os dados para outros ambientes de desenvolvimento estatísticos.

A junção dos programas *SQL Server* e *R* possibilita a interação de sistemas de gestão com sistemas de índole estatística. O *R* contém uma considerável capacidade estatística e esta fusão permite o desenvolvimento do código estatístico no interior do programa *SQL Server*.

A integração de *scripts* estatísticos em *scripts* de programação *SQL* possibilita a repetição dos mesmos para diferentes bases de dados, uma vez que este código

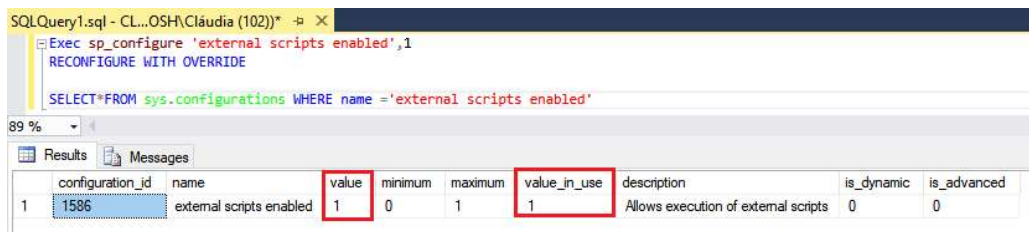
se adapta a diversos conjuntos de dados para estudar tendências, descobrir padrões estatísticos e, eventualmente, para construir modelos de previsão.

II.1. Manual de instalação

A instalação do programa *Microsoft SQL Server R Services 2016* requer um conjunto de procedimentos genéricos para interpretar código *R*.

Primeiramente, foi essencial realizar um estudo empírico com recurso a tutoriais da *Microsoft* e disponibilizados *online*, para iniciar a instalação². Posteriormente, foi fundamental a instalação de componentes que contêm as funcionalidades *R* e *Reporting Services* e, também, as configurações *SQL Server 2016 CTP3*, *Revolution R Open (RRO)* e *Revolution E Enterprise (RRE)* (a instalação destas componentes encontra-se em detalhe nos anexos deste relatório).

O processo terminou com a importação da base de dados para o *Microsoft SQL Server Management Studio (MSSMS)*, através da criação de um *login* e, configuração em *script* para testar e ativar o código *R*.



```
SQLQuery1.sql - CL...OSH\Cláudia (102))* -> X
Exec sp_configure 'external scripts enabled',1
RECONFIGURE WITH OVERRIDE
SELECT*FROM sys.configurations WHERE name ='external scripts enabled'
```

configuration_id	name	value	minimum	maximum	value_in_use	description	is_dynamic	is_advanced
1	external scripts enabled	1	0	1	1	Allows execution of external scripts	0	0

Figura 1- Configuração para ativar a funcionalidade estatística

² Anexo 2 – Manual de instalação do programa *Microsoft SQL Server R Services 2016*.

O *software* que reúne as funcionalidades do *Microsoft SQL Server R Services 2016* é o *Microsoft SQL Server Management Studio (MSSMS)*, no qual todo o processo de desenvolvimento de código é executado.

II.1.1. Pré-requisitos

Relativamente aos pré-requisitos, foram necessários três programas que viabilizam a fundamentação e execução da análise e gestão dos dados: *RStudio*, *Microsoft SQL Server Management Studio* e *Report Builder*.

No que respeita à funcionalidade dos programas, o *RStudio* permite a instalação dos pacotes estatísticos, enquanto que o *Microsoft SQL Server Management Studio* executa os códigos de programação com os *scripts* externos. Através das competências integradas em *R*, incorpora-os em procedimentos armazenados com linguagem T-SQL³, e com o recurso ao Programa *Microsoft SQL Server 2016 Report Builder* geram-se os resultados obtidos desse código em relatórios (*reports*). Recorde-se que estes *scripts* podem ser armazenados em procedimentos e poderão ser invocados, novamente, com o fim de os executar numa nova utilização.

A linguagem T-SQL contribui para a otimização do tempo de execução dos resultados que, conseqüentemente serão lidos pela aplicação *SQL Server 2016 Report Builder* e, por fim devolvidos em *scores* e gráficos. O script externo é a única forma de ser executada a linguagem *R*, e interligada com a linguagem T-SQL que permite que se obtenham análises estatísticas integradas no código de programação. Eventualmente, se existir algum pacote estatístico em falta e, se

³ Linguagem em código utilizada na *query* do *Microsoft SQL Server Management Studio (MSSMS)*

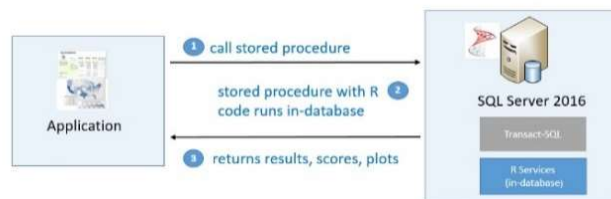
se gerar algum erro no output do MSSMS, é necessário proceder à instalação de pacotes estatísticos no SQL Server R Services 2016 com a execução do RStudio como “administrador” e, percorrer o código “*install_packages.r*”.

II.2. Operacionalização do código R com linguagem T-SQL

A criação do procedimento com código R numa *query* e, na instância SQL Server, permite a execução do código estatístico diretamente na base de dados defendendo a confidencialidade dos dados e, sem necessidade de movê-la para outra diretoria. Desta forma, as funcionalidades do R mantêm-se inalteradas na execução dentro da base de dados.

Um *script* genérico de código R é percorrido através de um conjunto de parâmetros fixos. O parâmetro *@language* indica que o *script* externo apenas suporta a linguagem R, por ser a única linguagem estatística suportada. Posteriormente, o parâmetro *@script* executa o *script* externo com código R, igual à linguagem utilizada em ambiente RStudio. Por fim, o *@input_data_1* especifica qual o *input* da base de dados, de modo que o *script* R seja invocado e, é neste parâmetro que existe a interligação da linguagem T-SQL com a linguagem R para replicar um conjunto de resultados.

Com base na figura seguinte, visualiza-se o processo de execução com o apoio do conjunto dos programas SQL Server (1), R (2) e Report Builder (3).



Fonte: Tutorial Microsoft

Figura 2 – Procedimento com linguagem R e T-SQL

Assim, é importante organizar o código criado em linguagem T-SQL e R em procedimentos armazenados (*stored procedure*), de modo a tornar eficientes as chamadas de código necessárias ao longo do processo de apuramento de resultados.

Na figura seguinte encontra-se um exemplo do código genérico criado com a integração do programa SQL Server R Services 2016.

```

--2 alterar o nome das variáveis
drop procedure if exists data_base_16
go
create procedure data_base_16
as
begin
EXECUTE sp_execute_external_script
@language = N'R'
,@script = N'
RESULTADO_16 <- InputDataSet
names(RESULTADO_16) <- c("COD","ANO","Vendas","Nareas","Nprog","Tncid","HorasTot","Anosemp")
#Dummies
RESULTADO_16$ANO <- factor(RESULTADO_16$ANO)
OutputDataSet <- as.data.frame(RESULTADO_16);'
,@input_data_1 = N'select*from RESULTADO_16'
WITH RESULT SETS ((COD NVARCHAR(255),Ano INT, Vendas FLOAT, Nareas INT, Nprog INT, Tncid INT, HorasTot FLOAT, Anosemp INT));
end;
go
exec data_base_16

```

	COD	Ano	Vendas	Nareas	Nprog	Tncid	HorasTot	Anosemp
1	00000000-0000-0000-0000-000000000008	2014	923850.549999999	8	59	573	22125	16
2	00000000-0000-0000-0000-000000000009	2014	517767.44	4	14	2019	9923	14
3	00000000-0000-0000-0000-000000000010	2014	90779.16	4	12	11	1889	14
4	00000000-0000-0000-0000-000000000011	2014	278887.75	8	44	473	7533	17
5	00000000-0000-0000-0000-000000000012	2014	427037.06	9	58	658	9056	17
6	00000000-0000-0000-0000-000000000013	2014	201491.8	5	15	326	4343	16
7	00000000-0000-0000-0000-000000000014	2014	187765.2	3	6	60	2379	16
8	00000000-0000-0000-0000-000000000015	2014	147460.32	7	19	135	1736	15
9	00000000-0000-0000-0000-000000000016	2014	378272.08	7	22	115	4732	15
10	00000000-0000-0000-0000-000000000017	2014	378546.3	7	45	201	3599	13

Figura 3 – Procedimento com código genérico criado com R e T-SQL

Neste script de código compreende-se que no desenvolvimento de código, primeiro, existe a criação de um procedimento (*data_base_summary*), depois recorre-se a um procedimento externo (*sp_execute_external_script*). Posteriormente, introduz-se nos diversos parâmetros genéricos:

- *@language* indica que se segue linguagem R;
- *@script* insere-se o habitual código R sob certos ajustes para gerar o output através do conjunto organizado dos dados;

- `@input_data_1` integra em linguagem T-SQL a localização do conjunto de dados;
- `@input_data_1_name` e `@output_data_1_name` são parâmetros que reforçam o que se encontra dentro de `@script`.

Para finalizar o procedimento indica-se o resultado que se pretende obter com respetiva dimensão, neste caso: `summary varbinary(max)`.

A partir do exemplo anterior, é possível identificar que o processo de criação do procedimento é, essencialmente, o mesmo para posteriores análises, apenas é necessário adequar o procedimento ao que se pretende calcular, efetuando alterações no que se encontra dentro de cada parâmetro genérico.

Assim, com a criação de um conjunto destes procedimentos para diferentes cálculos é possível reaproveitá-los para um novo conjunto de dados, uma vez que estes procedimentos ficam armazenados na base de dados.

Capítulo III: Análise da Base de Dados

A análise estatística visa investigar a informação contida num conjunto de dados, suportando dois tipos de estatística: a descritiva e a inferencial. A primeira facilita a organização dos dados obtidos sumariando-os, e a segunda, aplica métodos específicos (estimações ou testes de hipóteses) para obter conclusões consolidadas baseadas na informação contida nesse conjunto de dados.

Como a atividade da *Quidgest* é desenvolver *software* adequado às necessidades de cada cliente, decidiu-se estudar as vendas dos programas criados pelo software *GENIO*, relativamente a um determinado ano e cliente.

Os dados estão inseridos por tabelas, que estão organizadas e interligadas. A seleção dos dados tem de respeitar a *primary-key*, isto é, permitir identificar de forma unívoca cada linha das tabelas na base de dados, em que cada tabela apenas pode conter uma *primary-key*. No entanto, devido ao processo de adaptação dos colaboradores, apenas serão analisados dados para os anos de 2014 a 2016.

A partir da seleção do conjunto de variáveis, o cruzamento dos dados originou a existência de poucas observações. Com o objetivo de contornar este problema decidiu-se analisar, primeiramente, os dados de 2014 e 2015 e, seguidamente, adicionar os dados do ano de 2016, sobretudo para verificar se surgem alterações significativas com a introdução de nova informação.

A qualidade da informação contida em cada uma das variáveis depende da forma como esta foi obtida, isto é, se através de registos automáticos ou inseridas pelo próprio colaborador.

Transversalmente à boa integração dos dados registados foi possível criar com o código em linguagem *T-SQL* um conjunto de dados relevantes inseridos numa única tabela para a análise, descritiva e estatística, das vendas.

Iniciou-se o processo de concretização do estudo com a seleção do conjunto de informação a analisar, com a criação de código de programação armazenado por procedimentos individuais. Os *outputs* gerados, após o programa percorrer o código, foram agrupados por tabelas individuais (variável a variável). A informação contida nas tabelas criadas, para cada variável, originou o cruzamento de informação para se obter uma tabela geral e única. De salientar que, foi necessário a criação de restrições quanto à seleção do ano e do cliente.

A tabela global e única contém o conjunto total agrupado dos dados para cada cliente para as restrições estabelecidas, ou seja, o mesmo período temporal e código do cliente a selecionar.

III.1. Clientes e Programas

A empresa tem uma carteira de clientes de diversos sectores e, a estes são vendidos programas e/ ou projetos para desenvolver. Aqui destacam-se as vendas do *software GENIO*, vendido por parcerias, adaptado às necessidades de cada cliente. Importa referir que, as observações, inicialmente, consideradas como informação de clientes efetivos (*Cliefect*) representam apenas os clientes fictícios, ou seja, aqueles em que a empresa ofereceu propostas e incorreu em gastos de trabalho, mas em que não foram, efetivamente, realizadas vendas.

O *software GENIO*, vendido aos clientes através de programas informáticos, define que o uso do mesmo não é definitivo, o que significa que a empresa auferem um conjunto de benefícios tanto com a manutenção e formação de utilização, mas também com a venda de novas versões e resolução de incidentes.

III.2. Recolha de dados

Anteriormente, a recolha dos dados foi concretizada a partir do programa que gere o *Microsoft SQL Server R Services 2016*, nomeadamente, o *Microsoft SQL Server Management Studio (MSSMS)*. Neste contexto, em uma nova *query*, criou-se um procedimento geral com a intenção de substituir o conceito de variável global para a indicação do período temporal a analisar.

A variável global deve ser declarada no início de um procedimento e, pode ser invocada ao longo do código por subprocedimentos. Contudo, em linguagem T-SQL não é permitido a atribuição do conceito de variável global ao resultado que

se obtém a partir de um procedimento, visto que um procedimento retorna um conjunto de dados e a variável global apenas suporta um único valor específico, a alternativa foi a criação de uma tabela global.

```
--clientes do ano i como diferentes dos do ano j, para todo o i diferente de j
USE QUI2016
GO
--Não é permitido criar em T-SQL variáveis globais
--É preciso criar uma tabela global temporária
DROP PROCEDURE IF EXISTS PROC_ANO
GO
Create procedure PROC_ANO
AS
BEGIN
    Select quiclien.codclien AS CLIEN_ID, year(QUIfactu.DATA) as ANO from QUIclien
    JOIN QUIfactu on QUIfactu.codclien =quiclien.codclien
    JOIN QUIprcli on QUIprcli.codclien =quiclien.codclien
    JOIN QUIincid on QUIincid.codclien =quiclien.codclien
    Join Quiaccao on Quiaccao.codclien=quiclien.codclien

    Where
    year(QUIfactu.DATA)>=2014 and
    year(QUIprcli.DATACRIA)>=2014 and
    year(QUIprcli.DATAMUDA)>=2014 and
    year(QUIincid.DTINICIO)>=2014 and
    year(QUIaccao.DATA)>=2014

    Group by quiclien.codclien, (QUIfactu.DATA)
    Order by QUIfactu.DATA asc

END;
GO

EXEC PROC_ANO

DROP TABLE IF EXISTS #DATA
GO
Create TABLE #DATA (CLIEN_ID NVARCHAR(255), ANO NVARCHAR(4));
INSERT INTO #DATA EXEC PROC_ANO;
SELECT*FROM #DATA
```

Figura 4 – Código de criação da tabela global

A tabela global incorpora um conjunto de informações num único procedimento, que serve de restrição para a criação do conjunto de subprocedimentos individuais. Desta forma, restringe os dados para apenas selecionar da base de dados as informações para os anos em análise das vendas dos clientes.

A criação de uma tabela global é fundamental para que, em *output*, devolva novos resultados, sem haver necessidade de se introduzir mais alterações ao código criado. O que, além de permitir analisar mais do que um ano, facilita a modificação desse mesmo período temporal de acordo com o que se pretenda estudar, diminuído, assim, o tempo de adaptação do código a novas análises.

Ao executar o script armazenado, obtém-se um conjunto de dados sobre os clientes, relativamente aos anos de 2014 a 2016. Este resultado é gerado pelo procedimento inicial (*PROC_ANO*) ficando associado a uma tabela global,

armazenada na base de dados (*#DATA*) e, que serve de restrição para a seleção dos restantes dados através da criação de procedimentos individuais.

Procedeu-se à execução de um conjunto de procedimentos atribuídos de S1 a S8, cada um associado a uma das variáveis selecionadas e, que devolvem, em *output*, a informação do código do cliente com a da variável inserida por tabelas individuais.

A estratégia de organizar, em código T-SQL, as variáveis individuais com a criação e, posterior, armazenamento da informação das mesmas por tabelas individuais, permite a rápida deteção e resolução de possíveis erros na criação do restante código.

```
--criar as variáveis
--S1: valor das vendas faturadas
DROP PROCEDURE IF EXISTS S1
GO
CREATE PROCEDURE S1
AS
BEGIN
    Select quiclien.codclien AS CLIEN_ID, sum(quifactu.valortot) as V1 from quiclien
    join quifactu on quiclien.codclien=quifactu.codclien
    Group by quiclien.codclien

END;
GO
EXEC S1
--Tabela #S1
DROP TABLE IF EXISTS #S1
GO
CREATE TABLE #S1 (CLIEN_ID NVARCHAR(255), V1 FLOAT);
INSERT INTO #S1 EXEC S1;

select* from #S1
```

Figura 5 – Exemplo para criar a tabela individual para a variável S1

A interligação de cada tabela individual, é realizada através da informação que a *primary-key* devolve, isto é, transmite ao utilizador de que os dados de tabelas com a mesma *primary-key* coincidem. Esta vantagem permite associar informações de uma dada variável ou variáveis, através da indicação que a coluna com a mesma marca de *primary-key* em outra tabela contém para a mesma variável, novas e complementares, informações.

Posteriormente, foram criados subprocedimentos para as variáveis, assinalando qual a coluna e, de que tabela se pretende seleccionar. Introduziu-se a informação

dos subprocedimentos em tabelas e, realizou-se o cruzamento das informações de todos esses *outputs* em uma e única tabela geral.

```
--COM DADOS DE 2016
Drop table if exists RESULTADO_16
go
Create table RESULTADO_16 (CLIEN_ID NVARCHAR(255), ANO nvarchar(4), V1 FLOAT, V2 INT, V3 INT, V4 INT, V5 FLOAT,
V6 INT)
INSERT INTO RESULTADO_16
SELECT #S1.CLIEN_ID, #DATA.ANO, #S1.V1, #S2.V2,#S3.V3,#S4.V4, #S5.V5, #S6.V6 FROM #S1
JOIN #S2 ON #S1.CLIEN_ID=#S2.CLIEN_ID
JOIN #S3 ON #S1.CLIEN_ID=#S3.CLIEN_ID
JOIN #S4 ON #S1.CLIEN_ID=#S4.CLIEN_ID
JOIN #S5 ON #S1.CLIEN_ID=#S5.CLIEN_ID
JOIN #S6 ON #S1.CLIEN_ID=#S6.CLIEN_ID

JOIN #DATA ON #S1.CLIEN_ID=#DATA.CLIEN_ID

where #DATA.ANO <=2016
group by #S1.CLIEN_ID,#DATA.ANO, #S1.V1, #S2.V2,#S3.V3,#S4.V4,#S5.V5,#S6.V6
Order by #DATA.ANO asc

select*from RESULTADO_16
```

Figura 6 - Exemplo da junção das tabelas individuais em tabela global

A tabela geral, acima descrita, devolve ao utilizador os dados compatíveis com as indicações dadas, relativamente a cada ano e a cada uma das variáveis.

A tabela *infra* contem o registo do tipo de dimensão de cada variável inserida na tabela geral, conseguida a partir das informações contidas da base de dados QUI2016.

Tabela 1 - Conjunto das variáveis selecionadas da base de dados QUI2016

Variável	Tipo	Descrição
CLIENT_ID	NVARCHAR(255)	Código do cliente
V1	FLOAT	Valor faturado das vendas por cliente
V2	INT	Número de áreas por cliente
V3	INT	Número de programas vendidos por cliente
V4	INT	Total de incidentes por cliente
V5	FLOAT	Total de horas despendidas ao cliente
V6	INT	Número de anos da empresa como cliente

Ao longo da criação do código e, respetiva estruturação dos procedimentos e tabelas, obteve-se as informações sobre os clientes tais como: o valor total das vendas, a partir das faturas registadas, os programas alocados, total de incidentes e as horas totais despendidas.

Assim, concretizou-se o processo de criação da tabela global. Nas secções seguintes desenvolve-se o processo de interação das novas funcionalidades de programação e estatística.

Capítulo IV: Análises estatísticas em linguagem de programação a partir da base de dados QUI2016

O estágio permitiu explorar as funcionalidades do *Microsoft SQL Server R Services 2016* com linguagem de programação para a produção de gráficos e cálculos estatísticos. A integração da linguagem de programação, que contém ampla capacidade estatística (*R*) com um sistema de gestão de base de dados (*SQL: Structured Query Language*), necessita para a sua utilização a junção de aplicações para reportar os resultados, tais como o *Reporting Services* e *Reporting Builder*. Estas aplicações proporcionam a geração de resultados através do código desenvolvido e, respeitam a atualização do *backup* diário da base de dados *Quigenio*.

O *Microsoft SQL Server R Services 2016* permite, através *script* específico, tanto a manipulação de dados como a, posterior, análise estatística. Assim, mantém as linguagens *R* e *T-SQL* a funcionar dentro do ambiente *MSSMS* e, cujos *outputs* devolvem, exatamente, os mesmos resultados executados em ambiente *R IDE*.

Na aplicação de *Report Builder* processasse o *report* dos procedimentos armazenados, ao longo da criação de código, com o recurso a gráficos e tabelas.

Adicionalmente, na aplicação de *Reporting Services* é possível aceder a esses resultados armazenados, mas com ao servidor (*Cloud*).

IV.1. Exposição do Case Study

O caso de estudo que se segue, pretende analisar o comportamento das vendas anuais de um dado cliente, em função das variáveis que caracterizam o envolvimento da atividade desse mesmo cliente com a empresa.

Assim, a partir do conjunto de informações contidas na tabela global, desenvolveu-se, em linguagem de programação e estatística, o código em *script* externo para introduzir, no ambiente *MSSMS*, as capacidades estatísticas.

IV.2. Análise exploratória dos dados

As competências da análise exploratória dos dados têm a finalidade de compreender a relação entre as variáveis, extraindo informações da distribuição, das diferenças e comportamentos entre determinados conjuntos de dados.

Na análise, em ambiente *RStudio*, definiu-se a variável *Ano* como uma variável categórica codificada, a qual foi identificada como um *fator* no contexto de modelos de regressão.

A partir da informação obtida, detetou-se a existência de multicolinearidade entre variáveis e a presença de *outliers*. Devido à estreita relação das vendas e das encomendas, assim como horas a cobrar e horas totais no cliente, eliminou-se as variáveis: encomendas e horas totais, para melhorar os resultados da regressão, pois a correlação entre estas variáveis é muito próxima de um, o que permite concluir que são muito semelhantes e, que não contribuem com nova

informação para o modelo, mas simplesmente para a existência de multicolinearidade (teste de hipóteses para o coeficiente de correlação).

Relativamente às observações consideradas por *outlier*, entende-se que contêm um comportamento extremamente diferente das restantes interferindo, à posteriori, na criação da regressão linear múltipla. Este tipo de desigualdade explica-se por haver erros na introdução de dados, por eventuais registos em anos diferentes, ou simplesmente, pela existência de situações irregulares que o cliente ou empresa passaram, nomeadamente, endividamento ou investimento.

IV.3. Modelo estatístico adotado

O modelo poderia assumir uma regressão linear no contexto de dados de painel, uma vez que, se está a observar as vendas a clientes ao longo do período de 2014 a 2016. Porém, devido à atualização automática e diária dos dados podem surgir novos clientes nas observações da base de dados, que em alguns anos possam existir informações, mas que noutros possam estar omitidos. É neste contexto que, se em dados de painel existir omissão de dados para os anos em análise, $T = 2$ e, posteriormente, $T = 3$, o modelo não é razoável.

Para contornar este problema, procedeu-se à construção de um modelo mais adequado que se insere em regressão linear múltipla. Tal justifica-se, pelo facto de, na regressão múltipla serem tratadas duas ou mais variáveis explicativas. Assim, assume-se como variável dependente as *Vendas* e as restantes variáveis independentes ou explicativas como *Ano*, *Nareas*, *Nprog*, *Tincid*, *Horastot*, *Anosemp*.

As variáveis foram ajustadas através da transformação logarítmica para adequar os dados aos processos estatísticos e, identificar factos que não eram observáveis na forma inicial, nomeadamente, os *outliers*. A logaritmização das variáveis permitiu corrigir os dados, tornando-os mais estáveis e coesos.

Definindo o modelo de regressão linear múltipla dado por:

$$(1) \text{Logvendas} = \beta_0 + \beta_1 \text{Ano}_1 + \beta_2 \text{Nareas}_i + \beta_3 \text{Nprog}_i + \beta_4 \text{Logtincid}_i + \beta_5 \text{Loghorastot}_i \\ + \beta_6 \text{Anosemp}_i + \varepsilon_i$$

Em que *Logvendas* representa o ganho das vendas com o cliente *i*; *Ano* como variável binária; e, *Nareas*, *Nprog*, *Logtincid*, *Loghorastot* e *Anosemp* variáveis explicativas para cada cliente *i*, ε representa o erro experimental.

IV.3.1. Estimação dos parâmetros do modelo

Neste contexto, procedeu-se à análise estatística para os anos de 2014 a 2016, de modo a explorar os dados para os períodos de tempo em estudo e, compreender comportamentos na estimação do modelo.

A estimação do modelo estatístico: (1.1) $\widehat{\text{Logvendas}}_i = \widehat{\beta}_0 + \widehat{\delta}_0 \text{Ano2015}_i + \widehat{\delta}_1 \text{Ano2016}_i + \widehat{\beta}_1 \text{Nareas}_i + \widehat{\beta}_2 \text{Nprog}_i + \widehat{\beta}_3 \text{logtincid}_i + \widehat{\beta}_4 \text{loghorastot}_i + \widehat{\beta}_5 \text{anosemp}_i$.

Em que *ano2015_i* e *ano2016_i* são *variáveis* binárias, *i* representa a identificação do ($i = 1, \dots, 97$). Note-se que, é necessário fazer esta distinção, principalmente, quando existe um pequeno conjunto de dados.

IV.4. Resultados

A partir das observações para os anos de 2014 e 2016, no modelo de regressão linear múltipla obteve-se o modelo 1⁴, com 97 observações, no qual o teste de significância estatística individual dos parâmetros permite considerar que, para um nível significância de 5% os parâmetros β_2 , β_3 são estatisticamente significativos, ou seja, não se rejeita a hipótese nula ($H_0: \beta_{j_i}=0$).

Para $\widehat{\beta}_2 = -0,014381$, estima-se, que em média, por cada programa que o cliente adquire, o valor das vendas da empresa diminui cerca de 1,43% (o valor exato é igual $(e^{-0,014381} - 1) \times 100\% = -1,4278\%$) mantendo tudo o resto constante. Este resultado pressupõe que quanto mais programas forem vendidos, mais despesas o cliente gera à empresa, nomeadamente, despesas de investigação e desenvolvimento. Porém, na verdade, quanto mais programas o cliente adquirir mais valor gera à empresa, o que indica que o parâmetro não está bem estimado.

Para $\widehat{\beta}_4 = 0.374809$ estima-se que, em média, por cada pedido adicional de assistência, o valor das vendas da empresa aumenta cerca de 37,48%, mantendo tudo o resto constante. Este resultado pressupõe que mais incidentes conduzem a mais custos de assistência pagos à empresa.

Relativamente ao teste *White*⁵, este gera um valor-p igual a zero, como não se rejeita a hipótese nula de existência de homocedasticidade a 5%, não há evidência de heterocedasticidade no modelo 1. Contudo, como os resíduos do modelo 1 variam de 0 a 1 e, o ideal seria ter de variar em torno de zero (comportamento estacionário), procedeu-se à eliminação de observações,

⁴ Anexo 4 - Tabela 3 - Modelo 1

⁵ Anexo 4 – Tabela 3 – Modelo 1

selecionadas a partir da análise em gráficos de *clusters*, mas também da representação gráfica dos resíduos do modelo, que por apresentarem um comportamento diferente das restantes, procedeu-se à estimação de um novo modelo, 2⁶, com 87 observações.

Apesar da exclusão de observações, não se verificaram melhorias significativas ao nível da significância individual das variáveis. Este facto, deve-se essencialmente à discrepância entre os valores contidos em cada variável.

Considerando que se verificam as hipóteses clássicas, o teste de significância individual dos parâmetros permite que para um nível significância de 5% β_3 é estatisticamente significativo, não se rejeita a hipótese nula ($H_0: \beta_j=0$).

Com $\widehat{\beta}_3 = 0,288831$ estima-se que, em média, por cada incidente ocorrido ao cliente *i* as vendas aumentam cerca de 28,88%, *ceteris paribus*.

O modelo 2 explica em cerca de 67,39% da realidade do estudo, com um valor-p reduzido e inferior aos 5% e, um erro padrão de 0,5303. Quanto aos coeficientes houve alterações apenas na variável *Nprog* que deixou de ser significativa e a variável *Logtincid* apresentou um comportamento de significância estatística.

Adicionalmente, com o teste *White*⁷ regista-se um valor-p igual a zero, confirmando que se não rejeita a hipótese nula da existência de homocedasticidade logo as variâncias dos resíduos são homogéneas.

Quanto aos resíduos da regressão estes apresentam um comportamento em torno de zero assemelhando-se a um ruído branco.

⁶ Anexo 4 - Tabela 4 – Modelo 2

⁷ Anexo 4 - Tabela 4 – Modelo 2

Para confirmar se o modelo linear 2 é razoável ou se apresenta erros de especificação, realizou-se um teste *Reset* que gera um valor-p de 0.5374. O que significa que não se rejeita a hipótese nula, de que os coeficientes são iguais a zero, ou seja, o modelo está bem especificado.

IV.5. Desenvolvimento e explicação do código em MSSMS

A criação da tabela global, que possui os dados no intervalo de tempo entre 2014 a 2016 em ambiente MSSMS, permite que um conjunto de procedimentos de programação, com funcionalidades estatísticas, e o objetivo de que se obtém os mesmos outputs que o RStudio gera, mas com a diferença de que, este último, corre de forma externa ao local onde a base de dados está armazenada e o MSSMS fá-lo de forma interna devolvendo, exatamente, com os mesmos resultados.

O programa *Report Builder*, surge para estabelecer a ligação interativa com o MSSM e, transformar os resultados, anteriormente, obtidos dos procedimentos em formato de gráficos e tabelas, cujo objetivo é o de assim se construir relatórios dinâmicos. Cada utilizador da empresa, pode ter acesso a esse relatório através do servidor suportado pela aplicação *Reporting Services*.

O programa *Reporting Services* reporta o relatório construído no *Report Builder* através de uma hiperligação. Neste contexto, procedeu-se ao desenvolvimento do restante código⁸ no *Microsoft Server SQL Management Studio*, com a inserção das competências estatísticas que, suportadas pelas aplicações *Report Builder e Reporting Services*, reproduzem os resultados.

⁸ Anexo 5 – Código desenvolvido em ambiente MSSMS

O processo de desenvolvimento dos relatórios com os resultados dos procedimentos gerados, inicia-se com um script estatístico, no qual através de um procedimento (*data_base_16*), se pretende modificar o nome das variáveis (V1 a V7). O respetivo output que este procedimento devolve, é armazenado como uma nova tabela global (*dataTransf_16*) para de proteger os resultados obtidos e facilitar a posterior chamada ao código.

A tabela *dataTransf_16* contém a mesma informação da tabela global inicial, apenas com a diferença de que agora se encontram enunciadas o nome de cada coluna da tabela, com o respetivo nome das variáveis selecionadas.

Com esta alteração e, uma vez simplificada a análise exploratória realizada a cada variável, com o apoio do RStudio, transformou-se as seguintes variáveis: as vendas, o total de incidentes e total de horas totais através da logaritmação (*Dbase_transformed_model_16*).

O resultado obtido apresenta as novas variáveis, posteriormente guardadas em uma nova tabela *Dbase_transformada_16*. Salieta-se que, devido à complexa adaptação às funcionalidades do programa *Microsoft SQL Server R Services*, o *RStudio* permitiu criar código para a análise e testes iniciais, sendo depois adaptado na respetiva linguagem para o *MSSMS*.

A criação de um modelo de regressão linear múltipla num programa de interação de programação e estatística, requer um conhecimento consolidado das competências dos programas em conjunto, uma vez que, apesar de o código estar criado em linguagem *R*, ao correrem o procedimento no *MSSMS*, este gera um output em linha com uma quantidade de caracteres desconhecidos e, não restitui o modelo (*rxLinMod_16*)² no formato pretendido.

```

--7 criar um modelo de regressao linear
drop procedure if exists Dbase_transformed_model_16
go
create procedure Dbase_transformed_model_16
as
begin
EXECUTE sp_execute_external_script
    @language = N'R'
    , @script = N'
        require("RevoScaleR");
        DBMod <- rxLinMod(logvendas~Nareas+Nprog+logtincid+loghorastot+Anosemp, data=Dbase_transformada_16)
        trained_model <- data.frame(payload=as.raw(serialize(DBMod, connection=NULL)));
        ,@input_data_1=N'SELECT logvendas,Nareas,Nprog,logtincid,loghorastot,Anosemp FROM Dbase_transformada_16'
        ,@input_data_1_name = N'Dbase_transformada_16'
        ,@output_data_1_name = N'trained_model'
        WITH RESULT SETS ((model varbinary(max)));
    end;
go
exec Dbase_transformed_model_16

```

110 %

Results Messages

model
0x580A000000020003020200020300000003130000001A0000020E00000006401867A918F1E0CD53F967B728EB427338F98568FF4FC86193FD1D84FB10F49FD3FE1B7B8CC0951AD3FA6C25561D

Figura 7 - Output para a criação de um modelo de regressão linear

Numa tentativa de ultrapassar este problema, utilizou-se o programa *PowerShell*, no entanto, as funcionalidades deste programa não são intuitivas para um utilizador comum, uma vez que requer um conjunto de código específico para retornar o output num dado formato também específico.

Por conseguinte, aplicou-se os procedimentos desenvolvidos na criação do relatório⁹ da análise, com o recurso ao programa *Report Builder* para gerar resultados dinâmicos e atualizáveis. Este programa de construção de *reports*, inicia-se criando *Datasets*, que correspondem aos procedimentos criados e armazenados na base de dados. Após estabelecida a ligação entre o *Report Builder* e o MSSMS e, posteriormente com o *Reporting Services*, é possível apresentar os resultados de forma interativa através da elaboração de tabelas e gráficos. Estes resultados podem ser atualizados, ou a partir da atualização da base de dados ou através de possíveis adaptações dos procedimentos, sempre que seja necessário.

⁹ Anexo 6 – Relatório no *Report Builder*

Capítulo V: Conclusões

O relatório de estágio permitiu verificar que as funcionalidades do *SQL Server R Services 2016* podem apoiar o estudo de um conjunto diversificado de bases de dados de empresas, que pretendam explorar o comportamento da atividade desenvolvida pelas mesmas. No entanto, este programa pode ter um custo acrescido, uma vez que, além de ser necessário a compra de uma licença de utilização disponibilizada pela *Microsoft*, é essencial e indispensável a formação de utilização aos seus colaboradores. Estes tipos de programas requerem conhecimentos consolidados, especialmente, nas áreas de programação e estatística para que possam ser exploradas todas as competências que o programa *Microsoft SQL Server R Server 2016* contém e oferece.

Nesta investigação, a empresa *Quidgest* disponibilizou o programa *Microsoft SQL Server*, com a novidade de que este suporta capacidades estatísticas: *R Services*, e que por ser um sistema de gestão de bases de dados a empresa já utiliza na gestão da plataforma *GENIO*.

Assim, em contexto com o Programas SIFIDE, desenvolveu-se um conjunto de técnicas e competências na utilização do *Microsoft SQL Server* e linguagem *Transact-SQL* com as quais as empresas pensam ganhar capacidades cognitivas e preditivas a partir das bases de dados.

As funcionalidades do *Microsoft SQL Server R Services 2016* permitem proceder a análises estatísticas da informação contida em bases de dados usando a linguagem R, sem ter que copiar essas bases de dados para outro suporte informático.

Relativamente aos resultados obtidos, concluiu-se que a empresa apresentou ganhos com o número adicional de incidentes, que por sua vez aumentou o número total de horas despendidas pelo cliente. Além disso, quanto mais antigo o cliente for, mais estáveis são as vendas. Ainda assim, o número de programas vendidos também se destacou, inicialmente com uma interpretação indutiva a erro, provou-se que quanto mais programas forem vendidos mais ganhos gera à empresa, o que indiretamente, reflete que as variáveis se influenciam umas às outras, sobretudo porque repercutem resultados a partir umas das outras.

O presente relatório de estágio desafiou a vários níveis, principalmente porque houve o primeiro contacto com um programa de gestão de bases de dados, o qual exigiu um estudo empírico com a criação de vários testes experimentais ao *software*.

Na elaboração deste relatório existiram muitas dificuldades tanto a nível da seleção dos dados a estudar como também na criação e integração das duas linguagens abordadas (linguagem *T-SQL* e linguagem *R*). Destaca-se o número vasto de observações que existe na base de dados, para os anos a analisar e, que na prática apenas se pôde utilizar um conjunto mais restrito.

A criação do modelo de regressão linear no programa em estudo, revelou-se a etapa mais difícil da análise, devido aos resultados com este serem obtidos em linha de código com caracteres e, não apresentarem o formato pretendido.

Outro problema foi compreender a atividade da empresa e conhecer como é que o programa *Microsoft SQL Server R Services 2016* poderia contribuir para o sucesso futuro das vendas em conjunto com as aplicações de construção de *reports*.

Pode-se concluir que o programa *Microsoft SQL Server R Services 2016* apresenta um vasto conjunto de competências muito amplas no que se refere à gestão de grandes conjuntos de dados que quando integrado na análise estatística facilita a função dos colaboradores familiarizados com a utilização dos programas em separado.

Em suma, futuramente será desafiante analisar as vendas da perspectiva dos incidente e assim, compreender como obter, em output, a linha de código que se gera quando se pretende estimar um modelo estatístico no ambiente *MSSMS*, bem como proceder ao estudo das ferramentas contidas no programa *Microsoft SQL Server* para a exploração da base de dados com modelos adequados a grandes conjuntos temporais de observações, para proceder à construção de *reports* indicativos para as diversas empresas e/ou utilizadores.

Referências

- [1] Microsoft (2016). Getting Started with SQL Server Machine Learning [Online]. Available: <https://docs.microsoft.com/en-us/sql/advanced-analytics/r/getting-started-with-sql-server-r-services>. [Acedido em 2016/10/20]
- [2] Microsoft (2017). O que é o SQL Server Reporting Services (SSRS) [Online]. Available: <https://docs.microsoft.com/pt-br/sql/reporting-services/create-deploy-and-manage-mobile-and-paginated-reports>. [Acedido em 2017/07/29]
- [3] Microsoft (2017). Tutorials for SQL Server [online]. Available: <https://docs.microsoft.com/en-us/sql/sql-server/tutorials-for-sql-server-2016>. [Acedido em 2017/07/08]
- [4] João Maroco (2007). *Análise Estatística Com utilização do SPSS*, 3ªEd. Lisboa: Silabo
- [5] Christian Kleiber, Achim Zeileis (2008). *Applied Econometrics with R*. USA: Springer
- [6] Jeffrey M. Wooldridge (2013). *Introductory Econometrics: A Modern Approach*, 5ªEd. South-Western: Cengage Learning
- [7] Buck Woody, Danielle Dean, Debraj GuhaThakurta Gagan Bansal, Matt Conners, Wee-Hyong Tok (2016). *Data Science with Microsoft SQL Server 2016*. Washington: Microsoft Corporation
- [8] Frank A. Banin (2017). Advanced Analytics with R and SQL Part II – Data Science Scenarios [Online]. Available: http://www.sqlservercentral.com/articles/Data+Science/158498/?utm_source=SC&utm_medium=pubemail. [Acedido em 2017/08/03]

- [9] Quidgest, “Sobre a QUIDGEST” [Online]. Available: <http://www.quidgest.pt/boasvindasPT.asp?LT=PTG>. [Acedido em 2016/10/17]
- [10] Princeton University (2010). Getting Started in Fixed/Random Effects Models using R [Online]. Available: <http://www.princeton.edu/~otorres/Panel101R.pdf>. [Acedido em 2017/02/9]
- [11] Pedro Campos, Rita Sousa (2009). XIV – Estatística com R: Uma iniciação para o Ensino Básico e Secundário [Online]. Available: <http://alea-estp.ine.pt/Html/statofic/html/dossier/doc/dossie14.pdf>. [Acedido em 2007/03/21]
- [12] Bill Palace (2017). View R Plots from within SQL Server Management Studio [Online]. Available: <http://www.sqlservercentral.com/articles/R+Services/156107/>. [Acedido em 2017/05/15]
- [13] Umberto Guarnier Mignozzetti (2009). Introdução ao R Commander [Online]. Available: <http://www.nadd.prp.usp.br/cis/args/aprcmdr.pdf>. [Acedido em 2017/06/25]
- [14] Microsoft (2017). Report Builder in SQL Server2016 [Online]. Available: <https://docs.microsoft.com/en-us/sql/reporting-services/report-builder/report-builder-in-sql-server-2016>. [Acedido em 2017/04/17]

Anexos

Anexo 1. Plano de estágio

Tabela 2 – Representação temporal do plano de estágio



Anexo 2. Manual de instalação do Programa SQL Server R

Services 2016

A 2.1. Instalação

A instalação do software está disponível através da *Microsoft* sob compra de licença de utilizador, neste caso, pela empresa *Quidgest Portugal*.

De seguida, indicam-se algumas recomendações a ter em conta na instalação do *SQL Server*.

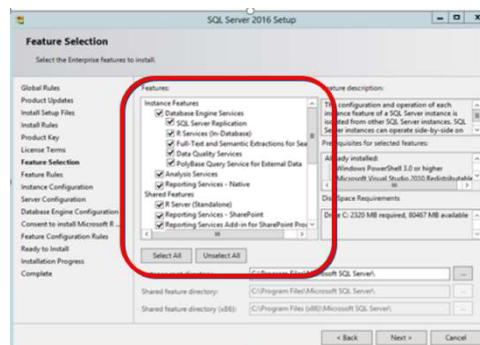


Figura 8 – Componentes a instalar

No passo “Server Configuration”, no separador *Collation* é necessário verificar se a *Collation* é “Latin1_General_CI_AI”. Esta configuração facilita a pesquisa de dados porque a introdução de dados está sujeita a muitos erros de acentuação.

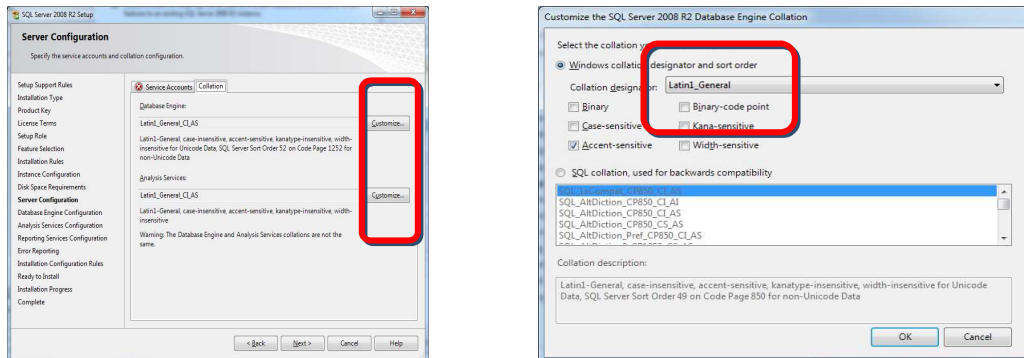


Figura 9- Configuração do servidor e Collation

Posteriormente, requer a ativação de filtros da *FullTextSearch* associados aos formatos *Office* e *PDF* para possibilitar a pesquisa de texto dos documentos inseridos no sistema.

Após a instalação, para verificar que filtros se encontram ativos deve-se correr o seguinte *script*:

(1) `Select document_type, path from sys.fulltext_document_types`

De seguida, devem ser corridos os executáveis correspondentes aos filtros não instalados e as *queries* para os ativar:

(2) `exec sp_fulltext_service 'load_os_resources', 1;`
`exec sp_fulltext_service 'verify_signature', 0;`
`go`

Depois de executar estas *queries* deve reiniciar-se a instância do SQL e de seguida executar novamente esta *query*:

(3) `Select document_type, path from`
`sys.fulltext_document_types`

Deste modo gera-se o output que dá a indicação que já se encontram ativos os filtros para os ficheiros do *Microsoft Office* e *PDF*.

A 2.1.1-Criação de utilizador para Base de Dados

De forma a ter acesso ao sistema do SQL Server é preciso a criação de um utilizador *Quidgest* com privilégios de administração. Deste modo, deve-se executar a seguinte *query*:

```
(4) EXEC sp_addlogin  
      'Quidgest',0x01006A648B9B2F3D17D320AE5C5782CFEEFF4F415DAF0647C  
      C7D,  
      @encryptopt = 'skip_encryption'  
      exec sp_addsrvrolemember 'Quidgest', 'sysadmin'
```

A 2.1.2-Ativação de *Named Pipes*

Aconselha-se a ativação das ligações e protocolos para que seja garantida uma ligação mais flexível ao sistema e à base de dados.

No *SQL Configuration Manager* (exemplo de caminho: *Start menu -> All Programs -> Microsoft SQL Server 2016-> Configuration Tools -> SQL Server Configuration Manager*) deve selecionar-se a instância de *SQL* e escolher a opção *Enabled*.

Seguidamente, é recomendado reiniciar a instância *SQL*.

A 2.1.3 – Verificação de *Filestream*

Caso o *script* da *query* dê erros, convém verificar se estão relacionados com o *Filestream*. Se sim, é necessário no *SQL Configuration Manager* (exemplo de caminho: *Start menu -> All Programs -> Microsoft SQL Server 2016-> Configuration Tools -> SQL Server Configuration Manager*) deve selecionar-se a instância de *SQL* e escolher a opção *Propriedades*.

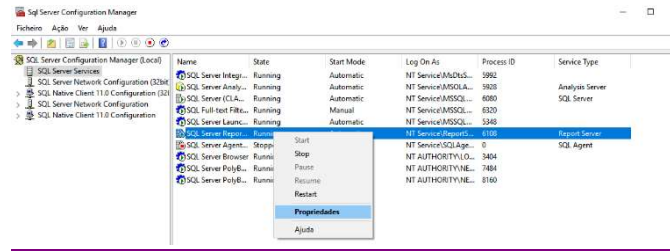


Figura 10 - SQL Server Configuration Manager

De seguida deve iniciar os estados “*Stopped*” para “*Running*” e reiniciar novamente a instância *SQL*.

A 2.2 – Estabelecer ligação com o *R Services*

Após a instalação, a mesma só termina depois de configurar na instância *SQL* a leitura de código *R*. Caso dê problemas, verificar se está ativo o *SQL Launchpad*.

A 2.3 – Configuração de *Reporting Services*

O *Reporting Services* é um serviço auxiliar do *SQL Server* para permitir visualizar os relatórios do sistema. É necessário assim efetuar algumas configurações para os mesmos funcionarem na aplicação.

Primeiramente, deve-se confirmar se o mesmo já se encontra instalado no *SQL Configuration Manager* (*Start menu -> All Programs -> Microsoft SQL Server 2016-> Configuration Tools -> SQL Server Configuration Manager*). Caso esteja instalado tem de estar na lista como *SQL Server Reporting Services (CLAUDIA)*.

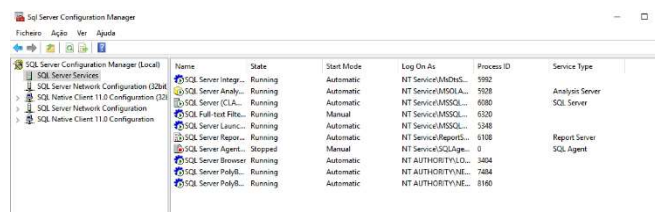


Figura 11 – Exemplo de lista de serviços do *SQL*

Seguidamente, verificar se o processo está a correr pois é necessário para o passo seguinte. Abrir o *Reporting Services Configuration Manager (Start menu -> All Programs -> Microsoft SQL Server 2016 -> Configuration Tools -> Reporting Services Configuration Manager)*. Depois selecciona-se a opção “*Web Portal URL*” e clica-se no respetivo *URL*.



Figura 12 - Reporting Services Configuration Manager: Web Portal URL

Após abrir o *URL* no browser Conectar *Microsoft SQL Services Management Studio, Reporting Services e Report Builder*.

O *Report Builder* é uma ferramenta auxiliar do Microsoft SQL Server e que se conecta com o *Reporting Services* para a criação de relatórios.

Numa primeira tarefa, conecta-se o *Report Builder* ao *Reporting Services*.

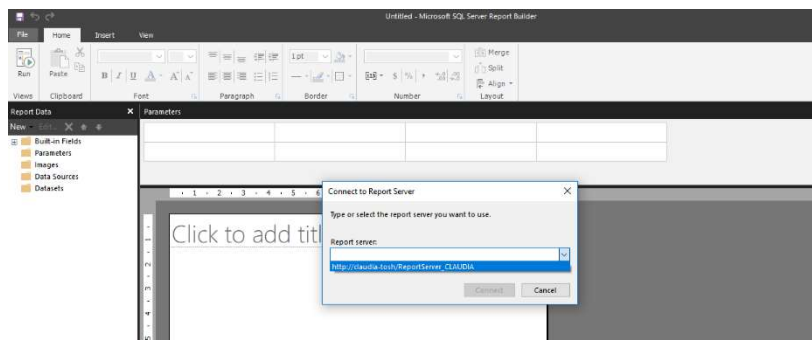


Figura 13 - Conectar o Report Builder ao Reporting Services

Se a ligação der erro, a causa poderá ser o facto do *SQL Server* não estar a correr o *Reporting Services*, o *URL* estar incorreto ou não ter permissões para usar o *SQL server*. Para isso, verificamos se o *Reporting Services* se encontra na instância *SQL*. Seguidamente colocamos o *Web Portal URL*.

Anexo 3. Modelos de regressão linear múltipla

Tabela 3 – Modelo 1

Modelo 1

Dependent Variable: LOG(VENDAS)
Method: Least Squares
Included observations:97

Variable	Coefficient	Std. Error	t-statistics	Prob.
C	10.0851108	0.462136	21.823	<2e+16
Ano2015	-0.136893	0.222842	-0.614	0.541
Ano 2016	-0.108197	0.231021	-0.468	0.641
Nareas	0.093732	0.075465	1.242	0.217
Nprog	-0.014381	0.005209	-2.761	0.007
Logtincid	0.374809	0.070786	5.295	8.51e-07
Loghorastot	-0.285210	0.401270	--0.711	0.479
Anosemp	0.064275	0.060835	1.057	0.294
R-squared	0.3612	Residuals:	Min	-2.8312
Adjusted R-squared	0.311		1Q	-0.4888
S.E of regression	0.8745		Median	0.1547
F-statistic	7.19		3Q	0.5308
Prob (F-statistics)	8.238e-07		Max	1.9219
Teste White	Test-statistic:	140.2166	P-value	0.0000

Tabela 4 – Modelo 2

Modelo 2

Dependent Variable: LOG(VENDAS)
Method: Least Squares
Included observations:87

Variable	Coefficient	Std. Error	t-statistics	Prob.
C	9.971291	0.324435	30.734	<2e-16
Ano2015	-0.088924	0.141077	-0.630	0.530
Ano2016	0.077788	0.151556	0.513	0.609
Nareas	0.027952	0.052367	0.534	0.595
Nprog	0.010181	0.006889	1.478	0.144
Logtincid	0.288831	0.048326	5.977	7.14e-08
Loghorastot	0.187769	0.259198	0.724	0.471
Anosemp	0.004715	0.038261	0.123	0.902
R-squared	0.6739	Residuals:	Min	-1.11330
Adjusted R-squared	0.6435		1Q	-0.3947
S.E of regression	0.5303		Median	0.06653
F-statistic	22.14		3Q	0.37193
Prob (F-statistics)	6.274e-16		Max	0.88955
Teste White	Test-statistic:	136.0672	P-value	0.000
Teste Reset	Reset:	0.62641	P-value	0.5374

Anexo 4. Código desenvolvido no MSSMS

```
--2 alterar o nome das variáveis

drop procedure if exists data_base_16
go
create procedure data_base_16
as
begin
EXECUTE sp_execute_external_script
    @language = N'R'
    , @script = N'

        RESULTADO_16 <- InputDataSet
        names(RESULTADO_16) <- c("COD", "ANO", "Vendas", "Nareas", "Nprog", "Tincid", "HorasTot", "Anosemp")

        RESULTADO_16$Nareas <- as.factor(RESULTADO_16$Nareas)

        OutputDataSet <- as.data.frame(RESULTADO_16);'
    , @input_data_1 = N'select*from RESULTADO_16'

    WITH RESULT SETS ((COD NVARCHAR(255), Ano INT, Vendas FLOAT, Nareas INT, Nprog INT, Tincid INT, HorasTot FLOAT, Anosemp INT));

end;
go
exec data_base_16
```

Figura 14 - Alteração do nome das variáveis

```
--3. Criar uma tabela com o procedimento anterior

Drop table if exists Dbase_16
go
Create table Dbase_16 (COD_CIEN NVARCHAR(255), Ano INT, Vendas FLOAT, Nareas INT, Nprog INT, Tincid INT, HorasTot FLOAT, Anosemp INT)
INSERT INTO Dbase_16 exec data_base_16

Select*from Dbase_16
```

Figura 15 - Criação de uma tabela com a identificação de cada variável

```
--4 Criação de novas variáveis (transformação)

drop procedure if exists data_base_Transf_16
go
create procedure data_base_Transf_16
as
begin
EXECUTE sp_execute_external_script
    @language = N'R'
    , @script = N'

        names(RESULTADO_16) <- c("COD", "Ano", "Vendas", "Nareas", "Nprog", "Tincid", "HorasTot", "Anosemp")
        RESULTADO_16$Nareas <- as.numeric(RESULTADO_16$Nareas)
        RESULTADO_16$Ano <- as.factor(RESULTADO_16$Ano)
        RESULTADO_16[,9] <- log(RESULTADO_16[,3])
        names(RESULTADO_16)[9] <- "logvendas"
        RESULTADO_16[,10] <- log(RESULTADO_16[,6])
        names(RESULTADO_16)[10] <- "logtincid"
        RESULTADO_16[,11] <- log(RESULTADO_16[,7])
        names(RESULTADO_16)[11] <- "loghorastot"

attach(RESULTADO_16)
DB <- summary(RESULTADO_16)
print(DB)
OutputDataSet <- data.frame(serialize(DB, NULL))

    , @input_data_1 = N'select*from RESULTADO_16'
    , @input_data_1_name = N'RESULTADO_16'
    , @output_data_1_name = N'OutputDataSet'

    WITH RESULT SETS ((DB varbinary(255)));

end;
go
Exec data_base_Transf_16
```

Figura 16 – Logaritmização das variáveis

```
--5 Criar tabela com as variáveis novas
drop procedure if exists dataTransf_16
go
create procedure dataTransf_16
as
begin
EXECUTE sp_execute_external_script
    @language = N'R'
    , @script = N'
        names(RESULTADO_16) <- c("COD", "Ano", "Vendas", "Nareas", "Nprog", "Tincid", "HorasTot", "Anosemp")
        RESULTADO_16$Nareas <- as.numeric(RESULTADO_16$Nareas)
        RESULTADO_16$Ano <- as.factor(RESULTADO_16$Ano)
        RESULTADO_16[,9] <- log(RESULTADO_16[,3])
        names(RESULTADO_16)[9] <- "logvendas"
        RESULTADO_16[,10] <- log(RESULTADO_16[,6])
        names(RESULTADO_16)[10] <- "logtincid"
        RESULTADO_16[,11] <- log(RESULTADO_16[,7])
        names(RESULTADO_16)[11] <- "loghorastot"
        OutputDataset <- data.frame(RESULTADO_16)

        , @input_data_1 = N'select*from RESULTADO_16'
        , @input_data_1_name = N'RESULTADO_16'
        , @output_data_1_name = N'OutputDataset'
    WITH RESULT SETS ((COD_CLIENTE NVARCHAR(255), Ano FLOAT, Vendas FLOAT, Nareas INT, Nprog INT, Tincid INT, HorasTot FLOAT, Anosemp INT,
        logvendas FLOAT, logtincid FLOAT, loghorastot FLOAT));
end;
go
Exec dataTransf_16
```

Figura 17 – Criação de nova tabela global com as variáveis transformadas

```
--Colocar o procedimento numa tabela
--6 Juntar à tabela todas as variáveis
Drop table if exists Dbase_transformada_16
go
Create table Dbase_transformada_16 (COD_CLIENTE NVARCHAR(255), Ano FLOAT, Vendas FLOAT, Nareas INT, Nprog INT, Tincid INT, HorasTot FLOAT, Anosemp INT,
logvendas FLOAT, logtincid FLOAT,
loghorastot FLOAT)
INSERT INTO Dbase_transformada_16 exec dataTransf_16
--utilizar esta tabela que contém a informação que se pretende estudar
Select*from Dbase_transformada_16
```

Figura 18 - Armazenamento dos resultados do procedimento dataTransf_16

```
--7 criar um modelo de regressao linear
drop procedure if exists Dbase_transformed_model_16
go
Create procedure Dbase_transformed_model_16
as
begin
EXECUTE sp_execute_external_script
    @language = N'R'
    , @script = N'
        require("RevoScaleR");
        DBMod <- rxLinMod(logvendas~Nareas+Nprog+logtincid+loghorastot+Anosemp, data=Dbase_transformada_16)
        trained_model <- data.frame(payload=as.raw(serialize(DBMod, connection=NULL)));
        , @input_data_1=N'SELECT logvendas,Nareas,Nprog,logtincid,loghorastot,Anosemp FROM Dbase_transformada_16'
        , @input_data_1_name = N' Dbase_transformada_16'
        , @output_data_1_name = N'trained_model'
    WITH RESULT SETS ((model varbinary(max)));
end;
go
exec Dbase_transformed_model_16
```

Figura 19 - Procedimento para criar o modelo de regressão linear

Anexo 5. Relatório no Report Builder

Funcionalidades SQL Server R Services 2016

Análise dos dados (2014 a 2016)



Figura 20 - Relatório no Report Builder