Universidade do Minho
Licenciatura em Ciências da Computação

Unidade Curricular de Bases de Dados

Ano Letivo de 2023/2024

Mademoiselle Borges: Um Sistema de Bases de Dados para a Gestão de Eventos em Eventopolis

Bruno Gião (A96544) João Pereira (A95375) Helena Salazar (A75635)
Tiago Teixeira (A97666)

Novembro, 2023

BD

Data de Recepção	
Responsável	
Avaliação	
Observações	

Mademoiselle Borges: Um Sistema de Bases de Dados para a Gestão de Eventos em Eventopolis

Bruno Gião (A96544) João Pereira (A95375) Helena Salazar (A75635) Tiago Teixeira (A97666)

Novembro, 2023

Resumo

Neste trabalho, foi inicializado o processo do desenho de um Sistema de Bases de Dados na forma da contextualização do problema, visando criar uma *blueprint* sólida e demonstrar que, efetivamente, é justificada a criação do presente Sistema de Bases de Dados, o levantamento de requisitos de manipulação, descrição, e controlo e a conceptualização do problema com recurso a um diagrama conceptual (*Entity-Relationship Diagram*).

Área de Aplicação: Desenho e arquitetura de Sistemas de Bases de Dados.

Palavras-Chave: Bases de Dados Relacionais, Definição de Sistema, SQL, Diagrama Conceptual, Recolha de Requisitos.

Índice

Lista de Figuras

Lista de Tabelas

1 Introdução e Definição do Sistema

1.1 Contextualização

Em Eventopolis, uma localidade remota no centro de uma grande floresta, a gestão dos eventos foi sempre baseada em *outsourcing* ou métodos manuais, devido à escassez de recursos humanos e à existência de um monopólio na área de Bases de Dados (BD). Este monopólio era controlado por uma seita de ocultistas tecnológicos, os quais praticavam preços exorbitantes e limitavam o acesso a uma parte significativa das informações nas suas BD. Após uma revolta interna motivada pela insatisfação com a direção da empresa, alguns ex-membros, descontentes com a situação, optaram por adotar uma abordagem mais humanista e criar uma *start-up* de Engenharia de Software em Eventopolis.

Ao tomar conhecimento desta informação, o Professor Doutor Henrique Borges, responsável atual pela Gestão de Eventos na Câmara Municipal da cidade, prontamente identificou a oportunidade de mitigar os prejuízos significativos dos últimos anos ao estabelecer um contrato com a referida *start-up* para a implementação de um Sistema de Bases de Dados (SBD) *open-source*.

O SBD seria batizado de "Mademoiselle Borges" em homenagem a Antoinette Borges, a antiga gestora de Eventos da Câmara Municipal de Eventopolis e esposa de Henrique Borges, que faleceu há alguns anos. Antoinette enfrentou uma pressão considerável ao ser forçada a gerir manualmente os eventos culturais de Eventopolis, com uma equipa de funcionários bastante limitada, desafios que foram fatores cruciais para o seu falecimento precoce.

Para Henrique Borges, este projeto tem então um significado profundamente pessoal. Além de simplificar o funcionamento dos eventos, diminuindo a mortalidade deste posto de trabalho, a criação deste Sistema também reflete a sua vontade de fomentar a promoção da arte e da cultura na sua pequena cidade, algo que era o maior sonho da sua falecida esposa. Antoinette queria ver a transformação da modesta e isolada cidade numa capital cultural, uma aspiração que, infelizmente, apenas se concretizaria após o seu falecimento.

Após a introdução do SBD, todos os eventos aprovados pela Câmara transformarão a cidade num cenário requintado que realça a estética do estilo *Art Nouveau*, o estilo artístico predileto da *Mademoiselle*, este estilo tira inspiração da vegetação exuberante, densa e colorida, característica das imensas florestas que rodeiam Eventopolis. O principal local de eventos será uma gigantesca estufa situada no parque central, construída no início do século anterior. Esta estrutura exibe uma cúpula central, vitrais coloridos e um esqueleto de ferro com linhas

detalhadas e artísticas, que ao longo do tempo oxidaram, apresentando agora uma tonalidade verde clássica.

1.2 Fundamentação

Considerando o modo prévio de gerir eventos em Eventopolis, onde o uso de serviços externos era considerado excessivamente dispendioso, e diante da escassez de recursos humanos para uma gestão manual, a única alternativa viável, na perspetiva de Henrique Borges, seria desenvolver um SBD interno.

1.3 Apresentação do Caso de Estudo

Este trabalho consistirá então na elaboração de um SBD que consiga, aptamente, ajudar Henrique Borges e a câmara municipal de Eventopolis a gerir e publicitar os seus eventos.

1.4 Motivação e Objectivos

O Professor Doutor Henrique Borges acredita que a introdução de uma base de dados trará sucesso aos eventos.

Os objetivos mencionados abaixo são fundamentais para refletir este sucesso:

- Aumentar a capacidade de armazenamento de informações;
- Saber em tempo real qual a previsão de afluência de cada evento, sendo assim possível planear os eventos com maior precisão;
- Perceber quais são os colaboradores com melhor desempenho nas vendas, permitindo o uso de incentivos para estimulá-los a alcançar novos patamares de vendas;
- Possibilitar uma gestão financeira mais abrangente e precisa;
- Garantir que é minimizada a possibilidade da capacidade do evento ser excedida;
- Obter, em tempo real, um registo preciso das compras de cada participante, bem como identificar os itens mais vendidos tanto em eventos específicos quanto globalmente;
- Melhorar a organização de horários para cada evento;
- Promover a cidade em âmbito nacional e internacional;

- Estimular a economia local por meio de injeção de capital na região.

1.5 Viabilidade

O Professor Doutor Henrique Borges defende que ao implementar um sistema de controlo de eventos será possível:

- Recuperar, no final no primeiro semestre, 40% das perdas anteriores e cerca de 20% do investimento inicial;
- Aumentar a participação nos eventos em 20% no primeiro ano.

1.6 Recursos

Recursos Humanos

- Pessoal de limpeza;
- Equipa de segurança;
- Vendedores;
- Equipa de multimédia;
- Funcionários da empresa de desenvolvimento;
- Potenciais Voluntários.

Recursos Materiais

- Hardware:
  - 1 servidor fornecido pela *start-up* com 128GiB;
  - 15 terminais "burros";
  - 10 computadores pessoais.
- Software:
  - SGBD;

  - Aplicação de vendas e aprovisionamento;
  - Redes sociais para divulgar o calendários de eventos.

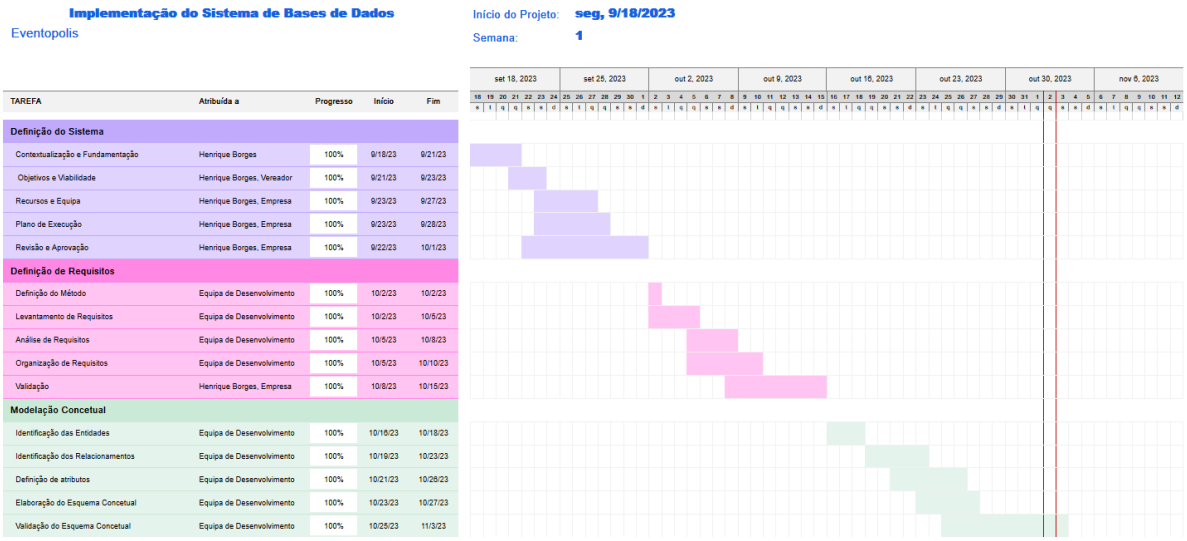
Equipa de Trabalho

- **Pessoal Interno** Na equipa de gestão de eventos da Câmara Municipal de Eventopolis temos:
  - Professor Doutor Henrique Borges: O coordenador principal da equipa;
  - Maria Ivanovna Ivanova: Colaboradora com experiência em *marketing* e co-coordenadora da equipa;
  - Herr Otto Mustermann: Trabalhador *part-time*.
- **Pessoal Externo** Já o pessoal externo, consiste na equipa de desenvolvimento da "start-up", que seria constituída por 4 engenheiros, nomeadamente:
  - Luke Bytespell;
  - Aurelius Cibernético;
  - Bella Firewall;
  - Aurora Matrix.

1.7 Plano de Execução

Com o intuito de desenvolver atempadamente o SBD "Mademoiselle Borges", Henrique Borges e a equipa de desenvolvimento reuniram-se e elaboraram o seguinte esquema GANTT[1]:

[1](ver 4.1 e 4.2 para esquema completo)



TAREFA	Atribuída a	Progresso	Início	Fim
Definição do Sistema				
Contextualização e Fundamentação	Henrique Borges	100%	9/18/23	9/21/23
Objetivos e Viabilidade	Henrique Borges, Vereador	100%	9/21/23	9/23/23
Recursos e Equipa	Henrique Borges, Empresa	100%	9/23/23	9/27/23
Plano de Execução	Henrique Borges, Empresa	100%	9/23/23	9/28/23
Revisão e Aprovação	Henrique Borges, Empresa	100%	9/22/23	10/1/23
Definição de Requisitos				
Definição do Método	Equipa de Desenvolvimento	100%	10/2/23	10/2/23
Levantamento de Requisitos	Equipa de Desenvolvimento	100%	10/2/23	10/5/23
Análise de Requisitos	Equipa de Desenvolvimento	100%	10/5/23	10/8/23
Organização de Requisitos	Equipa de Desenvolvimento	100%	10/5/23	10/10/23
Validação	Henrique Borges, Empresa	100%	10/8/23	10/15/23
Modelação Concetual				
Identificação das Entidades	Equipa de Desenvolvimento	100%	10/16/23	10/18/23
Identificação dos Relacionamentos	Equipa de Desenvolvimento	100%	10/19/23	10/23/23
Definição de atributos	Equipa de Desenvolvimento	100%	10/21/23	10/26/23
Elaboração do Esquema Concetual	Equipa de Desenvolvimento	100%	10/23/23	10/27/23
Validação do Esquema Concetual	Equipa de Desenvolvimento	100%	10/25/23	11/3/23

Figura 1.1: Diagrama de GANTT com conteúdos da primeira fase do Trabalho

1.8 Estrutura do Relatório

Após a introdução, seguir-se-ão mais dois capítulos, designadamente, metodologia e conclusão, acompanhados por um capítulo complementar para potenciais anexos. Na metodologia, abordaremos uma secção para cada fase do "ciclo de vida" do desenvolvimento de bases de dados, nomeadamente a definição de requisitos e modelação conceptual[2].

A conclusão seguirá os padrões convencionais de tal secção, fornecendo um breve resumo dos resultados finais e indicando as próximas etapas necessárias para o sucesso do trabalho. Prosseguimos com o esclarecimento de siglas e abreviaturas e, finalmente, o capítulo de anexos, que pode conter *scripts*, imagens de diagramas finais, entre outros.

[2]De notar que este relatório não abrange nas fases de modelação lógica nem implementação física, logo, não serão encontradas nesta fase do relatório.

2 Metodologia

2.1 Definição de Requisitos

2.1.1 Método de Levantamento e de Análise de Requisitos Adotados

Com o objetivo de determinar os objetivos a serem alcançados pelo SBD, foram agendadas diversas reuniões com o Prof. Dr. Henrique Borges, onde foram discutidas várias questões pertinentes. No final destas reuniões, é previsto obter-se uma compreensão abrangente dos requisitos a serem implementados.

2.1.2 Organização dos Requisitos Levantados

Sendo assim, um total de 88 requisitos foram levantados e, de acordo com a norma da organização de requisitos, separados em Descrição, Manipulação e Controlo, dos quais 44 são de Descrição, 33 de Manipulação e 11 de Controlo.

Requisitos de Descrição

Nr	Hour	Description	Source	Analist	Type
1	12:29	Each event must have an unique identifier.	Henrique Borges	Aurora Matrix	DR
2	12:29	Each event must have an unique name	Henrique Borges	Aurora Matrix	DR
3	12:29	Each event must have a description	Henrique Borges	Aurora Matrix	DR
4	12:29	Each event must have a beginning and ending date	Henrique Borges	Aurora Matrix	DR
5	12:29	Each event must have a maximum capacity	Henrique Borges	Aurora Matrix	DR
6	12:30	There are no concurrent events	Henrique Borges	Aurora Matrix	DR
7	12:30	Each employee must have an unique identifier which is formatted as XXXXXYYYYY where XXXXX is a descriptor of rank or function and YYYYY is a number	Henrique Borges	Aurora Matrix	DR
8	12:30	Each employee must have a name	Henrique Borges	Aurora Matrix	DR
9	12:30	Each employee must have a VAT	Henrique Borges	Aurora Matrix	DR
10	12:30	Each employee must have a birth date	Henrique Borges	Aurora Matrix	DR
11	12:30	Each employee must have a list of emails	Henrique Borges	Aurora Matrix	DR
12	12:30	Each employee must have a list of phone numbers	Henrique Borges	Aurora Matrix	DR
13	12:30	Each employee must have an address	Henrique Borges	Aurora Matrix	DR
14	12:30	An address is a composite of Street name, Locale and Postal Code	Henrique Borges	Aurora Matrix	DR
15	12:31	Each sale must have an unique identifier	Henrique Borges	Aurora Matrix	DR
16	12:31	Each sale must have a total sale value which is the result of a certain calculation	Henrique Borges	Aurora Matrix	DR
17	12:31	Each sale must have the quantity of products	Henrique Borges	Aurora Matrix	DR
18	12:31	Each sale must have the date the sale closed	Henrique Borges	Aurora Matrix	DR
19	12:32	Each participant must have an unique identifier	"Henrique Borges"	Aurora Matrix	DR
20	12:32	Each participant must have a name	Henrique Borges	Aurora Matrix	DR
21	12:32	Each participant must have date of birth	Henrique Borges	Aurora Matrix	DR
22	12:32	Each participant must have a list of phone numbers	Henrique Borges	Aurora Matrix	DR
23	12:32	Each participant must have, optionally, a list of emails	Henrique Borges	Aurora Matrix	DR
24	12:32	Each participant must have, optionally, their VAT	Henrique Borges	Aurora Matrix	DR
25	12:32	Each participant must have optionally, address	Henrique Borges	Aurora Matrix	DR
26	12:33	Each product must have an unique identifier	Henrique Borges	Aurora Matrix	DR
27	12:33	Each product must have an unique name	Henrique Borges	Aurora Matrix	DR
28	12:33	Each product must have a current price	Henrique Borges	Aurora Matrix	DR
29	12:33	Each product must have a stock which is a number that represents the total quantity of a given product in storage	Henrique Borges	Aurora Matrix	DR
30	12:33	Each product must have a description	Henrique Borges	Aurora Matrix	DR
31	12:33	A ticket is a product who's name is the same as the event's name	Henrique Borges	Aurora Matrix	DR
32	12:34	Each supplier must have an unique identifier	Henrique Borges	Aurora Matrix	DR
33	12:34	Each supplier must have an unique name	Henrique Borges	Aurora Matrix	DR
34	12:34	Each supplier must have an IBAN	Henrique Borges	Aurora Matrix	DR
35	12:34	Each supplier must have an list of emails	Henrique Borges	Aurora Matrix	DR
36	12:34	Each supplier must have an list of cellphones	Henrique Borges	Aurora Matrix	DR
37	12:34	Each supplier must have an address	Henrique Borges	Aurora Matrix	DR
38	12:34	The only type of identifier that should be manually inserted is the employee identifier, every other should be an automatic number	Henrique Borges	Aurora Matrix	DR
65	13:01	The amount of tickets does not exceed the maximum capacity of an event	Henrique Borges	Aurora Matrix	DR
70	13:07	The value of a product in a sale is dependant on it's base price at the moment of purchase and the desired quantity	Henrique Borges	Aurora Matrix	DR
79	13:23	Administrator info must be stored in the database as employees.	Henrique Borges	Aurora Matrix	DR
81	13:24	The value of a sale is sum of the value times the quantity of all products in a sale.	Henrique Borges	Aurora Matrix	DR
85	13:28	It must be possible to differenciate completed deliveries from ongoing reservations	Henrique Borges	Aurora Matrix	DR
86	13:28	It must be possible to differenciate failed reservations from ongoing reservations or completed reservations	Henrique Borges	Aurora Matrix	DR

Tabela 2.1: Requisitos de Descrição

Requisitos de Manipulação

Nr	Hour	Description	Source	Analist	Type
43	12:37	It must be possible to consult who is the manager of an employee.	Henrique Borges	Aurora Matrix	ER
44	12:39	It must be possible to consult products in a sale	Henrique Borges	Aurora Matrix	ER
45	12:41	It must be possible to consult every closed sale	Henrique Borges	Aurora Matrix	ER
46	12:42	It must be possible to consult all participants in a given event	Henrique Borges	Aurora Matrix	ER
47	12:43	It must be possible to consult all participants in the system	Henrique Borges	Aurora Matrix	ER
48	12:44	It must be possible to consult the participant associated with a given sale	Henrique Borges	Aurora Matrix	ER
49	12:45	It must be possible to consult all products in stock	Henrique Borges	Aurora Matrix	ER
50	12:46	It must be possible to consult all suppliers of a given product	Henrique Borges	Aurora Matrix	ER
51	12:46	It must be possible to consult only past suppliers of a given product	Henrique Borges	Aurora Matrix	ER
52	12:46	It must be possible to consult only eventual suppliers of a given product	Henrique Borges	Aurora Matrix	ER
53	12:47	It must be possible to consult all of a participant's purchases	Henrique Borges	Aurora Matrix	ER
54	12:48	It must be possible to consult all suppliers	Henrique Borges	Aurora Matrix	ER
55	12:49	It must be possible to consult all employees	Henrique Borges	Aurora Matrix	ER
56	12:50	It must be possible to consult all events	Henrique Borges	Aurora Matrix	ER
57	12:51	It must be possible to consult the value of sales in a particular day	Henrique Borges	Aurora Matrix	ER
58	12:51	It must be possible to consult the volume of sales in a particular day	Henrique Borges	Aurora Matrix	ER
59	12:52	It must be possible to determine who is the participant with highest volume of sales	Henrique Borges	Aurora Matrix	ER
60	12:53	It must be possible to determine the event with the highest volume of sales.	Henrique Borges	Aurora Matrix	ER
61	12:54	It must be possible to determine the event with the highest rate of participation	Henrique Borges	Aurora Matrix	ER
62	12:55	When the system closes, it must dump a sales report as a text file	Henrique Borges	Aurora Matrix	ER
63	12:59	A participant is inserted into the database when they buy a ticket	Henrique Borges	Aurora Matrix	ER
64	13:00	If an event is open-entry the sale of a ticket is still registered with 0 value	Henrique Borges	Aurora Matrix	ER
68	13:05	It must be possible to consult what events occured in a given timespan	Henrique Borges	Aurora Matrix	ER
69	13:06	It must be possible to consult which employee sold the most tickets for a given event.	Henrique Borges	Aurora Matrix	ER
73	13:16	It must be possible to consult the events in which someone participated in	Henrique Borges	Aurora Matrix	ER
74	13:17	It must be possible to insert new events into the database	Henrique Borges	Aurora Matrix	ER
75	13:18	It must be possible to insert new employees in to the database	Henrique Borges	Aurora Matrix	ER
76	13:19	A product, if it exists, must be updated as soon as the supplying is concluded	Henrique Borges	Aurora Matrix	ER
77	13:20	A product, if new, must be added to the database as soon as it is ordered	Henrique Borges	Aurora Matrix	ER
82	13:24	It must be possible to consult the employee managed by another employee	Henrique Borges	Aurora Matrix	ER
83	13:25	It must be possible to determine the participant with highest value of sales	Henrique Borges	Aurora Matrix	ER
84	13:27	It must be possible to determine the event with highest value in sales	Henrique Borges	Aurora Matrix	ER
87	13:30	It must be possible to consult all sales made by an employee	Henrique Borges	Aurora Matrix	ER

Tabela 2.2: Requisitos de Manipulação

Requisitos de Controlo

Nr	Hour	Description	Source	Analist	Type
39	12:35	Henrique Borges is a System Administrator	Henrique Borges	Aurora Matrix	AR
40	12:36	Maria Ivanovna Ivanova is a System Administrator	Henrique Borges	Aurora Matrix	AR
41	12:36	Herr Mustermann is a System Administrator	Henrique Borges	Aurora Matrix	AR
42	12:36	There must be a special way of accessing the database called guest for anyone who wishes to view information on events	Henrique Borges	Aurora Matrix	AR
66	13:02	Access to the database is only available from 07:00 to 02:00	Henrique Borges	Aurora Matrix	AR
67	13:04	Database Administrators may revoke access to the database if provided with a suitable/legal reason.	Henrique Borges	Aurora Matrix	AR
71	13:08	An administrator has access to any and all information in the database	Henrique Borges	Aurora Matrix	AR
72	13:09	An administrator has access to all and any functionalities of the database	Henrique Borges	Aurora Matrix	AR
78	13:23	Only administrators may update information	Henrique Borges	Aurora Matrix	AR
80	13:23	Only administrators have access to the total value of sales	Henrique Borges	Aurora Matrix	AR
88	13:30	The usernames in the database correspond directly to the identifier	Henrique Borges	Aurora Matrix	AR

Tabela 2.3: Requisitos de Controlo

Análise e Validação Geral dos Requisitos

Depois do levantamento dos requisitos, marcou-se uma reunião no intuito do pessoal externo tomar conhecimento dos requisitos documentados.

Essa reunião, por sua vez, foi realizada com sucesso, e o pessoal interno mostrou-se satisfeito com o progresso e nível de detalhe a que os membros da equipa de desenvolvimento de BD chegaram, especialmente o Prof. Dr. Henrique Borges, que viu muito potencial neste projeto.

2.2 Modelação Conceptual

2.2.1 Identificação Conceptual

Após analisar os requisitos anotados, a equipa de desenvolvimento procedeu com a modelação conceptual do SBD, tendo iniciado pela identificação das entidades, relacionamentos e os atributos de cada uma, não incluíndo chaves estrangeiras.

Entidades

- **Evento:** Evento a ser gerido.
  - ID: Chave Primária da entidade que estará no domínio INTEGER e será auto incrementável;
  - Nome: O nome de um evento que estará no domínio VARCHAR(75).
  - Descrição: A descrição de um elemento da tabela "Evento" será um breve texto que introduz o tema e qualquer tipo de subevento que possa estar inserido no evento. Sendo assim, este atributo estará no domínio TEXT;
  - DataFim: A data do fim de um evento será uma data com as horas a qual o evento será dado por oficialmente terminado. Sendo assim, estará no domínio DATETIME;
  - DataInicio: A data do inicio de um evento será uma data com as horas a qual o evento será dado por oficialmente iniciado. Sendo assim, estará no domínio DATETIME;
  - DataFim: A data do final de um evento será uma data com as horas a qual o evento será dado por oficialmente terminado. Sendo assim, estará no domínio DATETIME;
  - Capacidade: É importante ter noção da quantidade máxima de participantes num dado evento. Então, este atributo será um elemento numérico e estará no domínio INTEGER.
- **Funcionário:** Colaborador da câmara munincipal.
  - ID: Chave Primária da entidade que estará no domínio VARCHAR(10);
  - Nome: O nome legal ou social de um funcionário também será armazenado e estará no domínio VARCHAR(75);
  - NIF: O número de identificação fiscal é um número de 9 algarismos e, portanto

estará no domínio VARCHAR(9);

  - DataNascimento: Data de nascimento que estará no domínio DATE;

  - Email: Email é um atributo multivalorado que estará no domínio VARCHAR(75);

  - NTelemovel: Números de telemóvel. Podem conter letras e números. É um atributo multivalorado no domínio VARCHAR(20);

  - Morada: Morada composta por rua, localidade, código-postal. Atributo Composto de VARCHAR(), onde rua será VARCHAR(50), localidade VARCHAR(30) e código-postal VARCHAR(15);

- **Participante:** Pessoa que participa num evento(s). É registado no sistema quando efetua a compra do seu primeiro bilhete para um evento;

  - ID: Chave Primária da entidade que estará no domínio INTEGER;

  - Nome: O nome legal ou social da pessoa também será armazenado e estará no domínio VARCHAR(75);

  - NIF: O número de identificação fiscal é um número de 9 algarismos e, portanto estará no domínio VARCHAR(9). Este campo é opcional;

  - DataNascimento: Data de nascimento que estará no domínio DATE;

  - Email: Email do participante que estará no domínio VARCHAR(75). Este campo é multivalorado e opcional;

  - NTelemovel: Números de telemóvel. Podem conter letras e números. É um atributo multivalorado no domínio VARCHAR(20);

  - Morada: Morada composta por rua, localidade, código-postal. Atributo Composto de VARCHAR(), onde rua será VARCHAR(50), localidade VARCHAR(30) e código-postal VARCHAR(15). Este campo é opcional;

- **Artigo:** Artigo que é possível estar numa venda;

  - ID: Chave Primária da entidade que estará no domínio INTEGER;

  - Nome: O nome do artigo que estará no domínio VARCHAR(75);

  - Descrição: A descrição de um elemento da tabela "Artigo" será um breve texto que descreve o produto em questão e qualquer medida extra necessária a ter com o mesmo. Sendo assim, este atributo estará no domínio TEXT;

  - Preço: Valor de um artigo, logo estará no domínio DECIMAL(5,2);

  - Stock: Quantidade de um artigo que está disponível, estará no domínio INTEGER;

- **Venda:** Venda de artigo(s) a ser efetuada a um participante por parte de um funcionário;
  - ID: Chave Primária da entidade que estará no domínio INTEGER;
  - Valor: Valor total da venda que estará no domínio DECIMAL(5,2);
  - Quantidade: Valor total do número de artigos na venda que estará no domínio INTEGER;
  - Data: Data na qual a venda aconteceu logo estará no domínio DATE;
- **Fornecedor:** Quem fornece os artigos;
  - ID: Chave Primária da entidade que estará no domínio INTEGER;
  - Nome: O nome da empresa fornecedora que estará no domínio VARCHAR(75);
  - IBAN: Código de identificação de conta bancária para a qual devem ser feitos os pagamentos ao fornecedor. Está no domínio VARCHAR(50);
  - Email: Email do participante que estará no domínio VARCHAR(75) e é multivalorado;
  - Contacto: Pessoa que representa a empresa e o seu número de telemóvel, estará no domínio VARCHAR(50);
  - NTelemovel: Números de telemóvel. Podem conter letras e números. É um atributo multivalorado no domínio VARCHAR(20);
  - Morada: Morada composta por rua, localidade, código-postal. Atributo Composto de VARCHAR(), onde rua será VARCHAR(50), localidade VARCHAR(30) e código-postal VARCHAR(15);

Relacionamentos

- Evento emprega Funcionário: Relacionamento entre Evento e Funcionário, representado quais funcionários estão alocados para quais eventos. Sendo a cardinalidade muitos para muitos;
- Funcionário gere Funcionário: Relacionamento entre Funcionário e Funcionário, representado qual funcionário gere qual/quais funcionários são geridos por um Funcionário. Sendo a cardinalidade 1 para muitos;
- Funcionário realiza Venda: Relacionamento entre Funcionário e Venda, representado qual funcionário efetuou uma Venda. Sendo a cardinalidade um para muitos.
- Venda contém Artigo: Relacionamento entre Venda e Artigo, representando quais arti-

gos estão numa venda. Sendo a cardinalidade muitos para muitos.

  - Valor: Valor do artigo no momento da venda, estará então no domínio DECIMAL(5,2);

  - Quantidade: Valor total da quantidade de um dado artigo numa venda, estará então no domínio INTEGER;

- Venda para Participante: Relacionamento entre Venda e Participante, representando a que participante uma venda pertence. Sendo a cardinalidade muitos para 1;

- Artigo fornecido por Fornecedor: Relacionamento entre Artigo e Fornecedor, representado o fornecedor pelo qual um artigo foi fornecido. Sendo a cardinalidade muitos para muitos;

  - Data: Data na qual o artigo foi entregue pelo fornecedor, estará então no domínio DATETIME;

  - Quantidade: Valor total da quantidade de um dado artigo numa entrega por parte de um fornecedor, estará então no domínio INTEGER;

- Artigo encomendado do Fornecedor: Relacionamento entre Artigo e Fornecedor, representado o fornecedor pelo qual um artigo foi encomendado. Sendo a cardinalidade muitos para muitos;

  - DataEncomenda: Data na qual o artigo foi encomendado ao fornecedor, estará então no domínio DATETIME;

  - DataEsperada: Data na qual o artigo é esperado que seja entregue, estará então no domínio DATETIME. DataEncomenda e DataEsperada compõe a chave primária do relacionamento;

  - Quantidade: Valor total da quantidade de um dado artigo numa encomenda por parte de um fornecedor, estará então no domínio INTEGER;

2.2.2 Modelo Conceptual

Consoante os resultados da subsecção anterior, temos o seguinte diagrama ER, concebido na ferramenta BrModelo[1]:

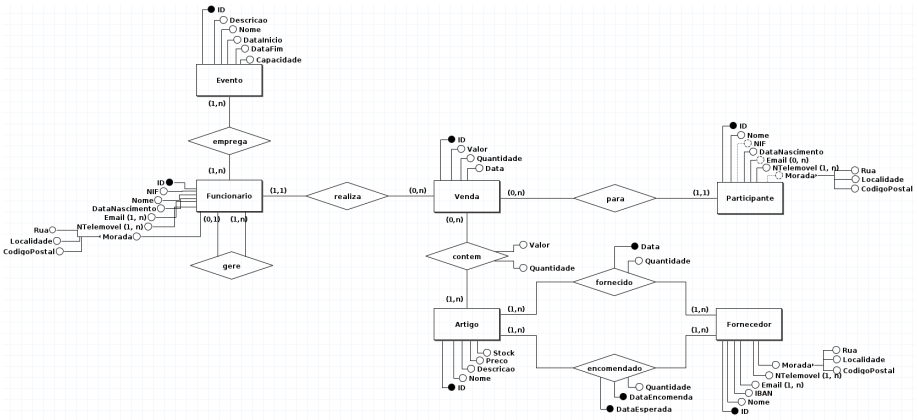


Figura 2.1: Diagrama Conceitual

2.3 Modelação Lógica

2.3.1 Construção e Validação do Modelo de Dados Lógico

Com a modelação conceptual completa e validada, a equipa de desenvolvimento procedeu com a modelação lógica do SBD, sendo que a mesma é uma tradução direta do modelo conceptual, introduzindo agora chaves estrangeiras.

2.3.2 Normalização de Dados

Devido à abordagem que adotámos no trabalho desde o modelo conceptual onde a leitura ocorre da esquerda para a direita e de cima para baixo, e considerando que atendemos a todos os requisitos das leis de normalização, os nossos dados encontram-se naturalmente normalizados na Terceira Forma Normal (3FN).

O facto de estarem normalizados em 3FN proporciona várias vantagens, incluindo a prevenção de anomalias como a redundância, e problemas de consistência decorrentes de processos de inserção, atualização ou remoção. Além disso, evita a atualização deficiente de um determinado registo com múltiplas ocorrências, onde nem todas são corretamente atualizadas.

[1]Note-se que os diagramas criados no brModelo não seguem a notação de Chen [Chen, 1976]

2.3.3 Apresentação e Explicação do Modelo Lógico Produzido

Tradução de Entidades

- **Evento → EventCal**
  - ID → EventID
  - Nome → EventName
  - Descricao → EventDescription
  - DataInicio → EventStart
  - DataFim → EventEnd
  - Capacidade → Capacity
- **Funcionário → Employee**
  - ID → EmployeeID
  - Nome → EmployeeName
  - NIF → EmployeeVAT
  - DataNascimento → EmployeeBirthDate

  Temos dois atributos multivalorados na entidade Funcionário que geram duas novas tabelas:

  - **Email → EmployeeEmail**
    - EmployeeID_eem: Chave estrangeira (chave primária e NOT NULL) que referencia a chave candidata EmployeeID na tabela Employee.
    - Email: Atributo que representa um email tal como no modelo conceptual.
  - **NTelemovel → EmployeePhone**
    - EmployeeID_ep: Chave estrangeira (chave primária e NOT NULL) que referencia a chave candidata EmployeeID na tabela Employee.
    - Phone: Atributo que representa um número de telemóvel como no modelo conceptual.
- **Venda → Sale**
  - ID → ReceiptNO

  - Valor → TotalValue
  - Quantidade → TotalQuantity
  - Data → DateOfSale
  - EmployeeID_s: Chave estrangeira (NOT NULL) que referencia a chave candidata EmployeeID na tabela Employee.
  - ParticipantID_s: Chave estrangeira (NOT NULL) que referencia a chave candidata ParticipantID na tabela Participant.

- **Participante → Participant**
  - ID → ParticipantID
  - Nome → ParticipantName
  - NIF → ParticipantVAT
  - DataNascimento → ParticipantBirthDate
  - Rua → Street
  - Localidade → Locale
  - CodigoPostal → Postal

  Temos dois atributos multivalorados na entidade Participante que geram duas novas tabelas:

  - **NTelemovel → ParticipantPhone**
    * ParticipantID_pp: Chave estrangeira (chave primária e NOT NULL) que referencia a chave candidata ParticipantID na tabela Participant.
    * Phone: Atributo que representa um número de telemóvel como no modelo conceptual.
  - **Email → ParticipantEmail**
    * ParticipantID_pem: Chave estrangeira (chave primária e NOT NULL) que referencia a chave candidata ParticipantID na tabela Participant.
    * Email: Atributo que representa um email como no modelo conceptual.

- **Artigo → Produto**
  - ID → ProductID

  - Nome → ProductName
  - Descricao → ProductDescription
  - Preco → BasePrice
  - Stock → QuantityInStock

- **Forncedor → Supplier**
  - ID → SupplierID
  - Nome → SupplierName
  - IBAN → IBAN
  - Rua → Street
  - Localidade → Locale
  - CodigoPostal → Postal

  Temos dois atributos multivalorados na entidade Artigo que geram duas novas tabelas:

  - **NTelemovel → SupplierPhone**
    * SupplierID_sp: Chave estrangeira (chave primária e NOT NULL) que referencia a chave candidata SupplierID na tabela Supplier.
    * Phone: Atributo que representa um número de telemóvel como no modelo conceptual.
  - **Email → SupplierEmail**
    * SupplierID_sem: Chave estrangeira (chave primária e NOT NULL) que referencia a chave candidata SupplierID na tabela Supplier.
    * Email: Atributo que representa um email como no modelo conceptual.

Tradução de Relacionamentos

Os relacionamentos com cardinalidade *n para n* geram novas tabelas. Veremos como são traduzidos os mesmos.

- **Evento emprega Funcionário → EventEmployee**
  - EventID_ee: Chave estrangeira (chave primária composta com EmployeeID_ee) que referencia a chave candidata EventID na tabela EventCal.

  - EmployeeID_ee: Chave estrangeira (chave primária composta com EventID_ee) que referencia a chave candidata EmployeeID na tabela Employee.

- **Venda contém Artigo → SaleProduct**

  - ReceiptNO_sp: Chave estrangeira (chave primária composta com ProductID_sp) que referencia a chave candidata ReceiptNO na tabela Sale.

  - ProductID_sp: Chave estrangeira que (chave primária composta com ReceiptNO_sp) referencia a chave candidata ProductID na tabela Product.

  - Valor → CurrentValue

- **Artigo fornecido por Fornecedor → ProductSupplierPast**

  - ProductID_psp: Chave estrangeira (chave primária composta com SupplierID_psp e DateOfDelivery) que referencia a chave candidata ProductID na tabela Product.

  - SupplierID_psp: Chave estrangeira (chave primária composta com ProductID_psp e DateOfDelivery) que referencia a chave candidata SupplierID na tabela Supplier.

  - Data → DateOfDelivery: Chave primária composta com ProductID_psp e SupplierID_psp.

  - Quantidade → Quantity

- **Artigo encomendado a Fornecedor → ProductSupplierFuture**

  - ProductID_psf: Chave estrangeira (chave primária composta com SupplierID_psf, DateOfReservation e DateOfSchedule) que referencia a chave candidata ProductID na tabela Product.

  - SupplierID_psf: Chave estrangeira (chave primária composta com ProductID_psf, DateOfReservation e DateOfSchedule) que referencia a chave candidata SupplierID na tabela Supplier.

  - DataEncomenda → DateOfReservation: Chave primária composta com ProductID_psf, SupplierID_psf e DateOfSchedule.

  - DataEsperada → DateOfSchedule: Chave primária composta com ProductID_psf, SupplierID_psf e DateOfReservation.

  - Quantidade → Quantity

2.3.4 Validação do Modelo com Interrogações do Utilizador

Por via a validar o modelo lógico explicado no ponto anterior, tomou-se a liberdade de escolher alguns requisitos de manipulação presentes na Tabela 2.2.

- Saber qual evento teve a maior taxa de participação de todos (RM61)

  $\tau_{rate} DESC \gamma_{ID,Name;rate} \pi_{\delta}((\rho_{EV} EventCal) \bowtie (\rho_{SP} SaleProduct) \bowtie_{\alpha \wedge \beta} (\rho_{P} Product))$

  Sabendo que:

  $\delta : EV.EventID \rightarrow ID, EV.EventName \rightarrow Name, SUM(SP.Quantity)/EV.Capacity \rightarrow rate.$

  $\alpha : P.ProductID = SP.ProductID\_sp$

  $\beta : P.ProductName = EV.EventName$

- Saber qual é o evento com maior valor de vendas (RM94)

  $$\tau_{TotVal} DESC \gamma_{ID,Name;TotVal} \pi_{\delta}((\rho_{EV} EventCal) \bowtie_{\alpha} (\rho_{S} Sale)$$

  Onde:

  $\alpha : EV.EventStart < S.DateOfSale < EV.EventEnd$

  $\delta : EV.EventID \rightarrow ID, EV.EventName \rightarrow Name, SUM(S.TotalValue) \rightarrow TotVal$

2.4 Modelação Física

2.4.1 Tradução do Esquema Lógico para o Sistema de Gestão de Bases de Dados Escolhido

Esta tradução é direta, sendo que até se pode utilizar uma funcionalidade do *MySQL Workbench* para a realizar. Listam-se exemplos de criação de tabelas. A restante criação de tabelas encontra-se nos anexos.

- Criação da Tabela SaleProduct

```
CREATE TABLE SaleProduct (
        ReceiptNO_sp INTEGER NOT NULL,
        ProductID_sp INTEGER NOT NULL,
        CurrentValue DECIMAL(5,2) NOT NULL,
        Quantity INTEGER NOT NULL,
        CONSTRAINT comp_key
            PRIMARY KEY (ReceiptNO_sp, ProductID_sp),
```

```
        FOREIGN KEY (ReceiptNO_sp)
            REFERENCES Sale (ReceiptNO),
        FOREIGN KEY (ProductID_sp)
            REFERENCES Product (ProductID)
);
```

Esta tabela representa o relacionamento Venda contém Artigo, onde indicamos todos os atributos como pedido no modelo lógico, bem como a identificação de chave primária composta com ReceiptNO_sp e ProductID_sp.

- Criação da Tabela EventCal

```
CREATE TABLE EventCal (
        EventID INTEGER AUTO_INCREMENT,
        EventName VARCHAR(75) NOT NULL UNIQUE,
        EventDescription TEXT NOT NULL,
        EventStart DATETIME NOT NULL,
        EventEnd   DATETIME NOT NULL,
        Capacity INTEGER NOT NULL,
        PRIMARY KEY (EventID)
);
```

No caso da tabela EventCal, a tradução é ainda mais simples, visto que não tem chaves estrangeiras. Sendo assim, só tivemos que indicar os atributos como pedido e indicar que a chave primária é EventID.

2.4.2 Tradução das Interrogações do Utilizador para SQL

Neste ponto, traduziremos alguns requisitos de manipulação em *queries* de MySQL, assim como as tabelas que resultam da execução das mesmas, tendo em conta o povoamento que usamos, que se encontra nos Anexos deste relatório.

- Saber qual evento teve a maior taxa de participação de todos (RM61)

```
SELECT EV.EventID,
       EV.EventName,
       SUM(SP.Quantity) / EV.Capacity * 100 AS rate
FROM EventCal as EV INNER JOIN SaleProduct as SP
    INNER JOIN Product AS P
        ON P.ProductID = SP.ProductID_sp AND
            P.ProductName = EV.EventName
    GROUP BY EV.EventID, EV.EventName
        ORDER BY rate DESC
    LIMIT 1;
```

#	EventID	EventName	rate
1	5	Puppet Show: 'Red Ridding Hood'	7.5000

Figura 2.2: Resultado RM61

- Saber qual é o evento com maior valor de vendas (RM94)

```
SELECT EV.EventID, EV.EventName, SUM(S.TotalValue) AS totVal
        FROM EventCal AS EV INNER JOIN Sale as S
                ON S.DateOfSale BETWEEN EV.EventStart AND EV.EventEnd
        GROUP BY EV.EventID, EV.EventName
                ORDER BY totVal DESC
LIMIT 1;
```

#	EventID	EventName	totVal
1	6	Arts and Crafts Fair	16.00

Figura 2.3: Resultado RM94

2.4.3 Definição e Caracterização das Vistas de Utilização em SQL

Definimos algumas vistas para conseguirmos abstrair o utilizador da interação direta com a base de dados. As vistas possibilitam-nos conceder permissão ou negar acesso a determinadas partes da BD. Desta forma, conseguimos garantir que, por exemplo, os funcionários não tenham acesso ao valor total das vendas. Veremos agora alguns exemplos de vistas que definimos. As restantes vistas podem ser consultadas nos anexos.

```
CREATE VIEW product_supplier AS
    SELECT P.ProductID AS ProductID,
           PSP.SupplierID_psp AS PastSupplierID,
           PSF.SupplierID_psf AS FutureSupplierID
      FROM Product AS P INNER JOIN ProductSupplierPast AS PSP
           ON P.ProductID = PSP.ProductID_psp
     INNER JOIN ProductSupplierFuture AS PSF
           ON P.ProductID = PSF.ProductID_psf
    GROUP BY P.ProductID, PSP.SupplierID_psp, PSF.SupplierID_psf;
```

A vista acima gera uma tabela virtual que contém todos os *suppliers* de produtos, ou seja, se um utilizador desejar obter os *suppliers* de um produto só terá que fazer uma simples *query*.

```
CREATE VIEW BestSellersEmployee AS
    SELECT EV.EventID AS EventID,
           S.EmployeeID_s AS EmployeeID,
           SUM(SP.Quantity) AS Quantity
        FROM EventCal AS EV INNER JOIN EventEmployee AS EE
            ON EV.EventID = EE.EventID_ee
        INNER JOIN Employee AS E
            ON EE.EmployeeID_ee = E.EmployeeID
        INNER JOIN Sale AS S
            ON E.EmployeeID = S.EmployeeID_s
        INNER JOIN SaleProduct AS SP
            ON S.ReceiptNO = SP.ReceiptNO_sp
        INNER JOIN Product AS P
            ON P.ProductName = EV.EventName
    GROUP BY EV.EventID, S.EmployeeID_s
        ORDER BY SUM(SP.Quantity) DESC;
```

Com esta vista conseguimos que se consiga obter o melhor ou melhores, como o utilizador desejar, vendedores em termos de quantidade de produtos vendida. A mesma resulta numa tabela virtual que expõe o ID do evento em causa, o ID do funcionário e a quantidade total de produtos vendida.

```
CREATE VIEW manager_employees AS
    SELECT E.EmployeeID AS ManagerEmployeeID,
           GROUP_CONCAT(Ei.EmployeeID) AS ManagedEmployeeIDs
    FROM Employee AS E LEFT OUTER JOIN Employee AS Ei
    ON Ei.EmployeeID_e = E.EmployeeID
    WHERE Ei.EmployeeID IS NOT NULL
    GROUP BY E.EmployeeID;
```

Esta vista tem como resultado uma tabela virtual composta pelos ID's dos funcionários que gerem funcionários, seguido do ID de todos os funcionários que o mesmo gere. Sendo assim, o utilizador consegue obter os funcionários que um funcionário gere.

2.4.4 Cálculo do Espaço da Base de Dados

De forma a Computar o espaço utilizado pela Base de Dados podemos utilizar a seguinte *query* de MariaDB/MySQL:

```
SELECT
    table_schema
        AS 'Database␣Name',
    SUM(data_length + index_length)
        AS 'Size␣in␣Bytes',
    ROUND(SUM(data_length + index_length) / 1024 / 1024, 2)
        AS 'Size␣in␣MiB'
```

```
FROM information_schema.tables
WHERE table_schema = 'mademoiselle_borges';
```

Ou, alternativamente, recorrendo a uma ferramenta, como "MySQL Workbench", no menu contexto esquerdo, sob *schemas*, clicando no símbolo 'i', conseguimos ver o espaço ocupado pelo nosso SBD.

Sendo assim, a nossa Base de Dados ocupa um total de 0.53 *Mebibytes* após o povoamento inicial. Como podemos ver na tabela resultante da *query* apresentada:

Database Name	Size in Bytes	Size in MebiBytes
mademoiselle_borges	557056	0.53

Após fazermos o cálculo do tamanho que a Base de Dados vai ocupar, tendo em conta o povoamento inicial, podemos concluir, assumindo que a BD tem um crescimento de 10% anual, que o seu tamanho previsto para o próximo será de 612762 bytes (0.58 MiB).

2.4.5 Indexação do Sistema de Dados

Apesar da indexação do sistema de dados, em geral, poder vir a trazer benefícios de *performance* optamos por não implementar este processo. Devido ao tamanho reduzido da base de dados e ao facto de usarmos maioritariamente os ID's de cada tuplo nas condições *WHERE* e *GROUP BY* o ganho de *performance* seria mínimo podendo até resultar em perda de *performance* em *queries* de atualização de tabelas. No futuro, com o crescimento da base de dados, se a *performance* baixar, a indexação de dados poderá voltar a ser considerada.

2.4.6 Procedimentos Implementados

Os procedimentos permitem ao utilizador encapsular uma série de *queries* numa só unidade. Para além disso, permite modularidade e reutilização. Veremos agora alguns procedimentos que implementamos. No caso, estes têm o intuito de povoar a base de dados.

```
DELIMITER &&
CREATE PROCEDURE register_supplier (IN s_name VARCHAR(75), iban VARCHAR(50),
                                    street VARCHAR(50), locale VARCHAR(30),
                                    postal VARCHAR(15), email VARCHAR(75),
                                    phone VARCHAR(20))
BEGIN
    DECLARE last_ins INTEGER;
```

```
    START TRANSACTION;

    INSERT INTO Supplier (SupplierName, IBAN, Street, Locale, Postal)
    VALUES (s_name, iban, street, locale, postal);

    IF ROW_COUNT() = 0 THEN
        ROLLBACK;
    END IF;

    SELECT SupplierID INTO last_ins
        FROM Supplier
    ORDER BY SupplierID DESC LIMIT 1;

    CALL register_supplier_phone(last_ins, phone);

    IF ROW_COUNT() = 0 THEN
        ROLLBACK;
    END IF;

    CALL register_supplier_email(last_ins, email);

    IF ROW_COUNT() = 0 THEN
        ROLLBACK;
    END IF;

    COMMIT;
END &&
```

O procedimento acima permite ao utilizador registar um novo fornecedor na base de dados.

```
DELIMITER &&
CREATE PROCEDURE add_prod_new_shop_new_part(IN e_id VARCHAR(10),
                                            pd_id INTEGER,
        part_name VARCHAR(75), part_vat VARCHAR(9),
    street VARCHAR(50), locale VARCHAR(30),
    postal VARCHAR(15), part_bd DATE,
        quant INTEGER, phone VARCHAR(20),
    email VARCHAR(75))
BEGIN
        DECLARE last_ins INTEGER;
    DECLARE last_sale_id INTEGER;
    DECLARE cur_val DECIMAL(5,2);

    START TRANSACTION;

    INSERT INTO Participant(ParticipantName,
                            ParticipantVAT,
                            ParticipantBirthDate,
```

```
                        Street, Locale, Postal)
VALUES (part_name, part_vat, part_bd, street, locale, postal);

IF ROW_COUNT() = 0 THEN
            ROLLBACK;
END IF;

SELECT ParticipantID INTO last_ins FROM Participant
ORDER BY ParticipantID DESC LIMIT 1;

INSERT INTO ParticipantPhone
VALUES (last_ins, phone);

IF ROW_COUNT() = 0 THEN
            ROLLBACK;
END IF;

IF email IS NOT NULL THEN
    INSERT INTO ParticipantEmail
    VALUES (last_ins, email);

    IF ROW_COUNT() = 0 THEN
                    ROLLBACK;
    END IF;
END IF;

INSERT INTO Sale(TotalValue, TotalQuantity, DateOfSale,
EmployeeID_s, ParticipantID_s)
VALUES ("0.00", "0", NULL, e_id, last_ins);

IF ROW_COUNT() = 0 THEN
            ROLLBACK;
END IF;

SELECT ReceiptNO INTO last_sale_id FROM Sale
ORDER BY ReceiptNO DESC LIMIT 1;

SELECT BasePrice INTO cur_val
FROM Product
WHERE ProductID = pd_id;

INSERT INTO SaleProduct(ReceiptNO_sp, ProductID_sp, CurrentValue,
Quantity)
VALUES (last_sale_id, pd_id, cur_val, quant);

IF ROW_COUNT() = 0 THEN
            ROLLBACK;
```

```
    END IF;

    UPDATE Product
    SET QuantityInStock = QuantityInStock - quant
    WHERE ProductID = pd_id;

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;
    COMMIT;
END &&
```

O procedimento permite criar um novo carrinho de compras de um participante novo. O participante será registado na base de dados e a compra será registada quando a compra for concluída.

```
DELIMITER &&
CREATE PROCEDURE register_new_employee (IN e_id VARCHAR(10),
    e_name VARCHAR(75),
    vat VARCHAR(9),
    bd DATE,
    street VARCHAR(50),
    locale VARCHAR(30),
        postal VARCHAR(15),
    manager VARCHAR(10),
    phone VARCHAR(20),
    email VARCHAR(75))
BEGIN
    START TRANSACTION;

    INSERT INTO Employee
    VALUES (e_id, e_name, vat, bd, street, locale, postal, manager);

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

    CALL register_employee_email(e_id, email);

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

    CALL register_employee_phone(e_id, phone);

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

    COMMIT;
END &&
```

O procedimento permite registar um novo funcionário na base de dados.

2.4.7 Plano de Segurança e de Recuperação de Dados

De forma a assegurar a possibilidade de Recuperação de Dados no casos de falhas, podemos utilizar o seguinte comando numa *shell* **POSIX** como a de sistemas operativos GNU-Linux/UNIX:

```
$ mysqldump -h localhost -u root -p mademoiselle_borges >\
$WORK_DIR/.backup/mademoiselle_backup.sql
```

Ou, em sistemas MS-DOS/Windows 3.x/Windows 9.x/Windows NT:

```
C:\MYSQL\BIN> MYSQLDUMP.EXE -u ROOT -p mademoiselle_borges^
> %BACKUPDIR%\MADEMOISELLE_BACKUP.SQL
```

3 Conclusões e Trabalho Futuro

O desenvolvimento e implementação de uma base de dados relacional (em MySQL) envolveu uma abordagem sistemática, desde a definição do sistema até à sua implementação física. O trabalho iniciou-se com a definição do sistema, durante a qual procedemos à contextualização, fundamentação, estabelecimento de objetivos e avaliação de viabilidade, delineando de seguida um plano de execução e identificando a equipa de trabalho responsável. Estes passos forneceram uma base sólida para o desenvolvimento subsequente.

Na fase de definição de requisitos, optou-se por um método presencial para levantar e analisar as necessidades do cliente. Realizamos reuniões diretas com intuito de descobrir de forma eficaz as exigências e expectativas do mesmo, e procedemos a organização dos requisitos obtidos em categorias distintas - descrição, exploração e controlo.

A modelação conceptual proporcionou uma visão abstrata do sistema, destacando as entidades, os relacionamentos e os atributos. O diagrama ER resultante tornou-se uma ferramenta valiosa para visualizar a estrutura de uma forma clara e compreensível.

Ao avançarmos para a modelação lógica, a construção do modelo de dados, juntamente com a sua normalização, garantiu a eficiência e integridade do sistema. A implementação física e a subsequente criação de *Queries* e a definição de *Views* evidenciaram a aplicabilidade prática do modelo desenvolvido.

Para um trabalho futuro, será fundamental refletir sobre os desafios enfrentados neste projeto e identificar áreas específicas para aprimoramento. Considerando as áreas onde tivemos maiores dificuldades, como no controlo de acesso no contexto *Grant* e *Revoke*, e nas funcionalidades de gestão e estatística, é imperativo traçar estratégias para melhorar estes pontos específicos. Ao refletirmos sobre estas dificuldades, esperamos, com base na experiência adquirida, estar mais preparados para enfrentar desafios futuros.

Referências

Begg, C., & Connolly, T. (2002). *Database Systems: A Practical Approach to Design, Implementation, and Management*. Addison-Wesley.

Belo, O. (2021). *Bases de Dados Relacionais - Implementação com MySQL*. FCA.

Chen, P. (1976). The Entity-Relationship Model—toward a Unified View of Data. *ACM Trans. Database Syst.*

4 Anexos

4.1 Anexo 1

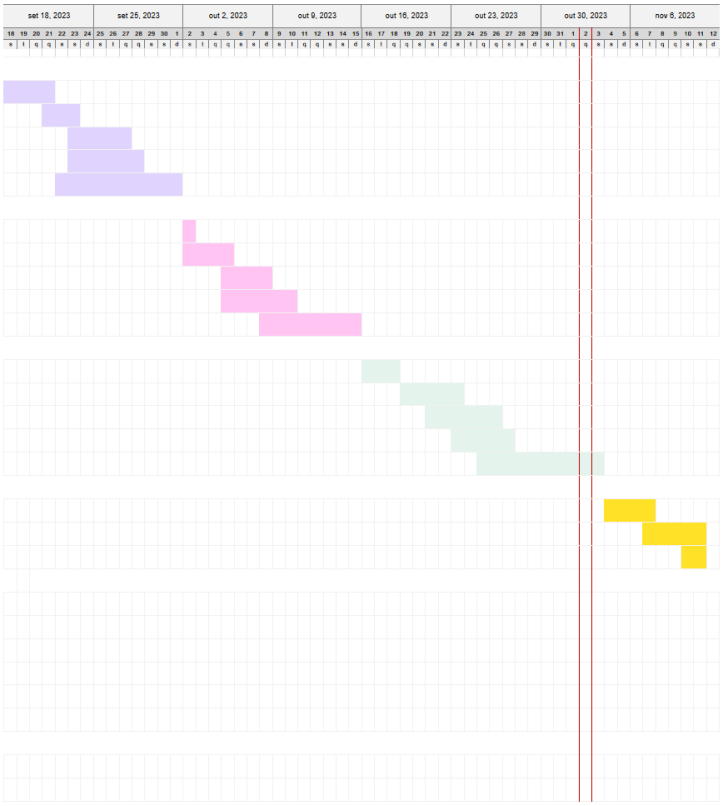
Implementação do Sistema de Bases de Dados

Eventopolis

Início do Projeto: **seg, 9/18/2023**

Semana: **1**

TAREFA	Atribuída a	Progresso	Início	Fim
Definição do Sistema				
Contextualização e Fundamentação	Henrique Borges	100%	9/18/23	9/21/23
Objetivos e Viabilidade	Henrique Borges, Vereador	100%	9/21/23	9/23/23
Recursos e Equipa	Henrique Borges, Empresa	100%	9/23/23	9/27/23
Plano de Execução	Henrique Borges, Empresa	100%	9/23/23	9/28/23
Revisão e Aprovação	Henrique Borges, Empresa	100%	9/22/23	10/1/23
Definição de Requisitos				
Definição do Método	Equipa de Desenvolvimento	100%	10/2/23	10/2/23
Levantamento de Requisitos	Equipa de Desenvolvimento	100%	10/2/23	10/5/23
Análise de Requisitos	Equipa de Desenvolvimento	100%	10/5/23	10/8/23
Organização de Requisitos	Equipa de Desenvolvimento	100%	10/5/23	10/10/23
Validação	Henrique Borges, Empresa	100%	10/8/23	10/15/23
Modelação Concetual				
Identificação das Entidades	Equipa de Desenvolvimento	100%	10/15/23	10/18/23
Identificação dos Relacionamentos	Equipa de Desenvolvimento	100%	10/19/23	10/23/23
Definição de atributos	Equipa de Desenvolvimento	100%	10/21/23	10/26/23
Elaboração do Esquema Concetual	Equipa de Desenvolvimento	100%	10/23/23	10/27/23
Validação do Esquema Concetual	Equipa de Desenvolvimento	100%	10/25/23	11/3/23
Modelação Lógica				
Derivação de Relações do Modelo de Dados Local	Equipa de Desenvolvimento	0%	11/4/23	11/7/23
Elaboração do Esquema Lógico	Equipa de Desenvolvimento	0%	11/7/23	11/11/23
Validação do Esquema Lógico	Empresa	0%	11/10/23	11/18/23
Implementação Física				
Criação do Sistema de Dados	Equipa de Desenvolvimento	0%	11/17/23	11/20/23
Criação do Esquema Físico da Base de Dados	Equipa de Desenvolvimento	0%	11/19/23	11/25/23
Povoamento da Base de Dados	Empresa	0%	11/25/23	12/1/23
Desenvolvimento de Queries	Equipa de Desenvolvimento	0%	11/28/23	12/4/23
Estabelecimento de um Plano de Indexação	Equipa de Desenvolvimento	0%	12/5/23	12/10/23
Desenvolvimento de Conjunto de Procedimentos	Equipa de Desenvolvimento	0%	12/10/23	12/13/23
Outros				
Desenvolvimento de um Sistema de Dashboarding	Equipa de Desenvolvimento	0%	12/11/23	12/14/23
Análise Crítica do Trabalho	Empresa	0%	12/14/23	12/17/23



4.2 Anexo 2

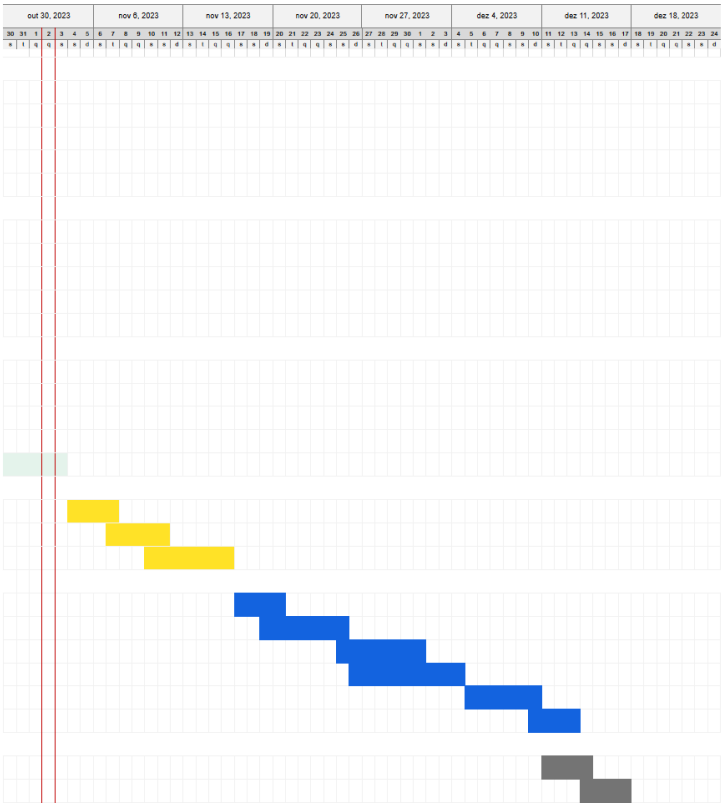
Implementação do Sistema de Bases de Dados

Eventopolis

Início do Projeto: **seg, 9/18/2023**

Semana: **7**

TAREFA	Atribuída a	Progresso	Início	Fim
Definição do Sistema				
Contextualização e Fundamentação	Henrique Borges	100%	9/18/23	9/21/23
Objetivos e Viabilidade	Henrique Borges, Vereador	100%	9/21/23	9/23/23
Recursos e Equipa	Henrique Borges, Empresa	100%	9/23/23	9/27/23
Plano de Execução	Henrique Borges, Empresa	100%	9/23/23	9/28/23
Revisão e Aprovação	Henrique Borges, Empresa	100%	9/22/23	10/1/23
Definição de Requisitos				
Definição do Método	Equipa de Desenvolvimento	100%	10/2/23	10/2/23
Levantamento de Requisitos	Equipa de Desenvolvimento	100%	10/2/23	10/5/23
Análise de Requisitos	Equipa de Desenvolvimento	100%	10/5/23	10/8/23
Organização de Requisitos	Equipa de Desenvolvimento	100%	10/5/23	10/10/23
Validação	Henrique Borges, Empresa	100%	10/8/23	10/15/23
Modelação Concetual				
Identificação das Entidades	Equipa de Desenvolvimento	100%	10/16/23	10/18/23
Identificação dos Relacionamentos	Equipa de Desenvolvimento	100%	10/19/23	10/23/23
Definição de atributos	Equipa de Desenvolvimento	100%	10/21/23	10/26/23
Elaboração do Esquema Concetual	Equipa de Desenvolvimento	100%	10/23/23	10/27/23
Validação do Esquema Concetual	Equipa de Desenvolvimento	100%	10/25/23	11/3/23
Modelação Lógica				
Derivação de Relações do Modelo de Dados Local	Equipa de Desenvolvimento	0%	11/4/23	11/7/23
Elaboração do Esquema Lógico	Equipa de Desenvolvimento	0%	11/7/23	11/11/23
Validação do Esquema Lógico	Empresa	0%	11/10/23	11/16/23
Implementação Física				
Criação do Sistema de Dados	Equipa de Desenvolvimento	0%	11/17/23	11/20/23
Criação do Esquema Físico da Base de Dados	Equipa de Desenvolvimento	0%	11/19/23	11/25/23
Povoamento da Base de Dados	Empresa	0%	11/25/23	12/1/23
Desenvolvimento de Queries	Equipa de Desenvolvimento	0%	11/26/23	12/4/23
Estabelecimento de um Plano de Indexação	Equipa de Desenvolvimento	0%	12/5/23	12/10/23
Desenvolvimento de Conjunto de Procedimentos	Equipa de Desenvolvimento	0%	12/10/23	12/13/23
Outros				
Desenvolvimento de um Sistema de Dashboarding	Equipa de Desenvolvimento	0%	12/11/23	12/14/23
Análise Crítica do Trabalho	Empresa	0%	12/14/23	12/17/23



4.3 Anexo 3

```
DROP SCHEMA IF EXISTS mademoiselle_borges;
CREATE SCHEMA IF NOT EXISTS mademoiselle_borges;

USE mademoiselle_borges;
SET GLOBAL event_scheduler = ON;
CREATE TABLE EventCal (
        EventID INTEGER AUTO_INCREMENT,
        EventName VARCHAR(75) NOT NULL UNIQUE,
        EventDescription TEXT NOT NULL,
        EventStart DATETIME NOT NULL,
        EventEnd   DATETIME NOT NULL,
        Capacity INTEGER NOT NULL,
        PRIMARY KEY (EventID)
);

CREATE TABLE Employee (
        EmployeeID VARCHAR(10),
        EmployeeName VARCHAR(75) NOT NULL,
        EmployeeVAT VARCHAR(9) NOT NULL UNIQUE,
        EmployeeBirthDate DATE NOT NULL,
        Street VARCHAR(50) NOT NULL,
        Locale VARCHAR(30) NOT NULL,
        PostalCode VARCHAR(15) NOT NULL,
        EmployeeID_e VARCHAR(10),
        PRIMARY KEY (EmployeeID),
        FOREIGN KEY (EmployeeID_e)
            REFERENCES Employee (EmployeeID)
);

CREATE TABLE EmployeePhone (
        EmployeeID_ep VARCHAR(10) NOT NULL,
        Phone VARCHAR(20) NOT NULL UNIQUE,
        PRIMARY KEY (EmployeeID_ep),
        FOREIGN KEY (EmployeeID_ep)
            REFERENCES Employee (EmployeeID)
);
CREATE TABLE EmployeeEmail (
        EmployeeID_eem VARCHAR(10) NOT NULL,
        Email VARCHAR(75) NOT NULL UNIQUE,
        PRIMARY KEY (EmployeeID_eem),
        FOREIGN KEY (EmployeeID_eem)
            REFERENCES Employee (EmployeeID)
);

CREATE TABLE EventEmployee(
```

```
        EventID_ee INTEGER,
        EmployeeID_ee VARCHAR(10),
        PRIMARY KEY (EventID_ee, EmployeeID_ee),
        FOREIGN KEY (EventID_ee)
            REFERENCES EventCal (EventID),
        FOREIGN KEY (EmployeeID_ee)
            REFERENCES Employee (EmployeeID)
);

CREATE TABLE Participant (
        ParticipantID INTEGER AUTO_INCREMENT,
        ParticipantName VARCHAR(75) NOT NULL,
        ParticipantVAT VARCHAR(9) NULL,
        ParticipantBirthDate DATE NOT NULL,
        Street VARCHAR(50) NULL,
        Locale VARCHAR(30) NULL,
        Postal VARCHAR(15) NULL,
        PRIMARY KEY (ParticipantID)
);

CREATE TABLE ParticipantEmail (
        ParticipantID_pem INTEGER NOT NULL,
        Email VARCHAR(75) NULL UNIQUE,
        PRIMARY KEY (ParticipantID_pem),
        FOREIGN KEY (ParticipantID_pem)
            REFERENCES Participant (ParticipantID)
);

CREATE TABLE ParticipantPhone (
        ParticipantID_pp INTEGER NOT NULL,
        Phone VARCHAR(20) NOT NULL UNIQUE,
        PRIMARY KEY (ParticipantID_pp),
        FOREIGN KEY (ParticipantID_pp)
            REFERENCES Participant (ParticipantID)
);

CREATE TABLE Sale (
        ReceiptNO INTEGER AUTO_INCREMENT,
        TotalValue DECIMAL(5,2) NOT NULL,
        TotalQuantity INTEGER NOT NULL,
        DateOfSale DATETIME NULL,
        EmployeeID_s VARCHAR(10) NOT NULL,
        ParticipantID_s INTEGER NOT NULL,
        PRIMARY KEY (ReceiptNO),
        FOREIGN KEY (EmployeeId_s)
            REFERENCES Employee (EmployeeID),
        FOREIGN KEY (ParticipantID_s)
```

```
            REFERENCES Participant (ParticipantID)
);

CREATE TABLE Product (
        ProductID INTEGER AUTO_INCREMENT,
        ProductName VARCHAR(75) NOT NULL UNIQUE,
        ProductDescription TEXT NOT NULL,
        BasePrice DECIMAL(5,2) NOT NULL,
        QuantityInStock INTEGER NOT NULL,
        CONSTRAINT Stock CHECK (QuantityInStock >= 0),
        PRIMARY KEY (ProductID)
);

CREATE TABLE SaleProduct (
        ReceiptNO_sp INTEGER NOT NULL,
        ProductID_sp INTEGER NOT NULL,
        CurrentValue DECIMAL(5,2) NOT NULL,
        Quantity INTEGER NOT NULL,
        PRIMARY KEY (ReceiptNO_sp, ProductID_sp),
        FOREIGN KEY (ReceiptNO_sp)
                    REFERENCES Sale (ReceiptNO),
        FOREIGN KEY (ProductID_sp)
                    REFERENCES Product (ProductID)
);

CREATE TABLE Supplier (
        SupplierID INTEGER AUTO_INCREMENT,
        SupplierName VARCHAR(75) NOT NULL UNIQUE,
        IBAN VARCHAR(50) NOT NULL UNIQUE,
        Street VARCHAR(50) NOT NULL,
        Locale VARCHAR(30) NOT NULL,
        Postal VARCHAR(15) NOT NULL,
        PRIMARY KEY (SupplierID)
);

CREATE TABLE SupplierEmail (
        SupplierID_sem INTEGER NOT NULL,
        Email VARCHAR(75) NOT NULL UNIQUE,
        PRIMARY KEY (SupplierID_sem),
        FOREIGN KEY (SupplierID_sem)
            REFERENCES Supplier (SupplierID)
);

CREATE TABLE SupplierPhone(
        SupplierID_sp INTEGER NOT NULL,
        Phone VARCHAR(20) NOT NULL UNIQUE,
        PRIMARY KEY (SupplierID_sp),
```

```
        FOREIGN KEY (SupplierID_sp)
            REFERENCES Supplier (SupplierID)
);

CREATE TABLE ProductSupplierPast(
        ProductID_psp INTEGER NOT NULL,
        SupplierID_psp INTEGER NOT NULL,
        DateOfDelivery DATETIME NOT NULL, -- date of delivery
                Quantity INTEGER NOT NULL,
                PRIMARY KEY (ProductID_psp,
                SupplierID_psp,
                DateOfDelivery),
        FOREIGN KEY (ProductID_psp)
            REFERENCES Product (ProductID),
        FOREIGN KEY (SupplierID_psp)
            REFERENCES Supplier (SupplierID)
);

CREATE TABLE ProductSupplierFuture (
        ProductID_psf INTEGER NOT NULL,
        SupplierID_psf INTEGER NOT NULL,
        DateOfReservation DATETIME NOT NULL, -- date of reservation
                DateOfSchedule DATETIME NOT NULL, -- expected date
        Quantity INTEGER NOT NULL,
        PRIMARY KEY (ProductID_psf,
                SupplierID_psf,
                DateOfReservation,
                DateOfSchedule),
        FOREIGN KEY (ProductID_psf)
            REFERENCES Product (ProductID),
        FOREIGN KEY (SupplierID_psf)
            REFERENCES Supplier (SupplierID)
);
```

4.4 Anexo 4

```
-- (65)
CREATE VIEW total_event_sale_value AS
        SELECT EV.EventID AS EventID, SUM(S.TotalValue) AS TotalValue
                FROM EventCal AS EV INNER JOIN Sale AS S
                        ON S.DateOfSale BETWEEN EV.EventStart AND EV.EventEnd
        GROUP BY EV.EventID;

CREATE VIEW employee_sales AS
        SELECT E.EmployeeID AS EmployeeID, E.EmployeeName AS EmployeeName, SUM(S.TotalValue) AS TotalValue
                FROM Employee AS E INNER JOIN Sale AS S
                        ON E.EmployeeID = S.EmployeeID_s
        GROUP BY E.EmployeeID;

CREATE VIEW product_supplier AS
        SELECT P.ProductID AS ProductID, PSP.SupplierID_psp AS PastSupplierID, PSF.SupplierID_psf AS FutureSupplierID
                FROM Product AS P INNER JOIN ProductSupplierPast AS PSP
                        ON P.ProductID = PSP.ProductID_psp
                INNER JOIN ProductSupplierFuture AS PSF
                        ON P.ProductID = PSF.ProductID_psf
        GROUP BY P.ProductID, PSP.SupplierID_psp, PSF.SupplierID_psf;

CREATE VIEW purchase_history AS
        SELECT P.ParticipantID AS ParticipantID, S.ReceiptNO AS ReceiptNO, S.DateOfSale AS PurchaseDate
                FROM Sale AS S INNER JOIN Participant AS P
                        ON S.ParticipantID_s = P.ParticipantID
        GROUP BY P.ParticipantID, S.ReceiptNO;

-- DROP VIEW IF EXISTS manager_employees;
CREATE VIEW manager_employees AS
    SELECT E.EmployeeID AS ManagerEmployeeID,
                GROUP_CONCAT(Ei.EmployeeID) AS ManagedEmployeeIDs
    FROM Employee AS E LEFT OUTER JOIN Employee AS Ei
    ON Ei.EmployeeID_e = E.EmployeeID
    WHERE Ei.EmployeeID IS NOT NULL
    GROUP BY E.EmployeeID;
-- SELECT * FROM manager_employees;

CREATE VIEW event_volume AS
        SELECT EC.EventID AS EventID, SUM(S.TotalQuantity) AS TotalQuantity
                FROM EventCal As EC INNER JOIN Sale AS S
                        ON S.DateOfSale BETWEEN EC.EventStart AND EC.EventEnd
        GROUP BY EC.EventID;

CREATE VIEW participant_sales AS
        SELECT P.ParticipantID AS ParticipantID, SUM(S.TotalValue) AS TotalSales
                FROM Participant AS P INNER JOIN Sale AS S
                        On P.ParticipantID = S.ParticipantID_s
        GROUP BY P.ParticipantID;

CREATE VIEW failed_reservations AS
        SELECT PSP.ProductID_psp AS ProductID
                FROM ProductSupplierPast AS PSP INNER JOIN ProductSupplierFuture AS PSF
                        ON PSF.ProductID_psf NOT IN (PSF.ProductID_psf)
                                WHERE PSF.DateOfSchedule < CURDATE();

-- view para tabelas com tabelas dependentes
CREATE VIEW employee_full AS
        SELECT E.*, GROUP_CONCAT(Email.Email, Phone.Phone) AS Contacts
    FROM Employee AS E LEFT OUTER JOIN EmployeeEmail AS Email
                ON E.EmployeeID = Email.EmployeeID_eem
        LEFT OUTER JOIN EmployeePhone AS Phone
        ON E.EmployeeID = Phone.EmployeeID_ep
        GROUP BY E.EmployeeID;

CREATE VIEW ParticipantFull AS
        SELECT P.*, GROUP_CONCAT(Email.Email, Phone.Phone) AS Contacts
                FROM Participant AS P LEFT OUTER JOIN ParticipantPhone AS Phone
                ON P.ParticipantID = Phone.ParticipantID_pp
                LEFT OUTER JOIN ParticipantEmail AS Email
                ON P.ParticipantID = Email.ParticipantID_pem
                GROUP BY P.ParticipantID;

CREATE VIEW SupplierFull AS
        SELECT S.*, GROUP_CONCAT(Email.Email, Phone.Phone) AS Contacts
                FROM Supplier AS S LEFT OUTER JOIN SupplierEmail AS Email
                ON S.SupplierID = Email.SupplierID_sem
                LEFT OUTER JOIN SupplierPhone AS Phone
                ON S.SupplierID = Phone.SupplierID_sp
                GROUP BY S.SupplierID;

CREATE VIEW SaleNoValue AS
        SELECT S.ReceiptNO AS ReceiptNO,
                    S.TotalQuantity AS TotalQuantity,
            S.DateOfSale AS DateOfSale,
            S.EmployeeID_s AS EmployeeID,
            S.ParticipantID_s AS ParticipantID
                FROM Sale AS S;
```

```
-- check product in sale (44)
CREATE VIEW ProductsInSale AS
        SELECT ReceiptNO_sp AS ReceiptNO, GROUP_CONCAT(ProductID_sp) AS ProductIDs
                FROM SaleProduct
        GROUP BY ReceiptNO_sp;

-- check all participants of a given event (46)
CREATE VIEW ParticipantsEvent AS
        SELECT EV.EventID AS EventID, GROUP_CONCAT(PA.ParticipantID) AS ParticipantIDs
                FROM EventCal AS EV INNER JOIN Sale AS S
                        ON S.DateOfSale BETWEEN EV.EventStart AND EV.EventEnd
            INNER JOIN Participant AS PA
                                ON PA.ParticipantID = S.ParticipantID_s
                        INNER JOIN SaleProduct AS SP
                                ON S.ReceiptNO = SP.ReceiptNO_sp
                        INNER JOIN Product AS PR
                                ON PR.ProductID = SP.ProductID_sp AND EV.EventName = PR.ProductName
        GROUP BY EV.EventID;

-- check suppliers of a product (50)
CREATE VIEW ProductSuppliers AS
        SELECT PSP.ProductID_psp AS ProductID, GROUP_CONCAT(PSP.SupplierID_psp) AS SupplierIDs
                FROM ProductSupplierPast AS PSP INNER JOIN ProductSupplierFuture AS PSF
                        ON PSP.ProductID_psp = PSF.ProductID_psf
        GROUP BY PSP.ProductID_psp;

-- check past suppliers of a product (51)
CREATE VIEW ProductPastSuppliers AS
        SELECT PSP.ProductID_psp AS ProductID, GROUP_CONCAT(PSP.SupplierID_psp) AS SupplierIDs
                FROM ProductSupplierPast AS PSP
        GROUP BY PSP.ProductID_psp;

-- check future suppliers of a product (52)
CREATE VIEW ProductFutureSuppliers AS
        SELECT PSF.ProductID_psf AS ProductID, PSF.SupplierID_psf  AS Suppliers
                FROM ProductSupplierFuture AS PSF
        GROUP BY PSF.ProductID_psf;

-- check all sales associated with a participant (53)
CREATE VIEW SalesParticipant AS
        SELECT S.ParticipantID_s AS ParticipantID, GROUP_CONCAT(S.ReceiptNO)
                FROM Sale AS S
        GROUP BY S.ParticipantID_s;

-- check sale values and volume in a given day (57 and 58)
CREATE VIEW DailySales AS
        SELECT S.DateOfSale AS DateOfSale, SUM(S.TotalValue) AS TotalValue, SUM(S.TotalQuantity) AS TotalQuantity
                FROM Sale AS S
        GROUP BY S.DateOfSale;

-- check who sold the most ticket in Event (72)
CREATE VIEW BestSellersEmployee AS
        SELECT EV.EventID AS EventID, S.EmployeeID_s AS EmployeeID, SUM(SP.Quantity) AS Quantity
                FROM EventCal AS EV INNER JOIN EventEmployee AS EE
                        ON EV.EventID = EE.EventID_ee
                INNER JOIN Employee AS E
                        ON EE.EmployeeID_ee = E.EmployeeID
                INNER JOIN Sale AS S
                        ON E.EmployeeID = S.EmployeeID_s
                INNER JOIN SaleProduct AS SP
                        ON S.ReceiptNO = SP.ReceiptNO_sp
                INNER JOIN Product AS P
                        ON P.ProductName = EV.EventName
        GROUP BY EV.EventID, S.EmployeeID_s
                ORDER BY SUM(SP.Quantity) DESC;

-- events someone participated in (81)
CREATE VIEW EventsParticipated AS
        SELECT S.ParticipantID_s AS ParticipantID, GROUP_CONCAT(EV.EventID, EV.EventName) AS EventID, EventName
                FROM Sale AS S INNER JOIN SaleProduct AS SP
                        ON S.ReceiptNO = SP.ReceiptNO_sp
                INNER JOIN Product AS P
                        ON SP.ProductID_sp = P.ProductID
                INNER JOIN EventCal AS EV
                        ON P.ProductName = EV.EventName
        GROUP BY S.ParticipantID_s;

-- check on employee sales (98)
CREATE VIEW EmployeeSales AS
        SELECT S.EmployeeID_s, GROUP_CONCAT(S.ReceiptNO) AS ReceiptNO
                FROM Sale AS S
        GROUP BY S.EmployeeID_s;
```

4.5 Anexo 5

```
USE mademoiselle_borges;

DELIMITER &&
CREATE PROCEDURE register_supplier (IN s_name VARCHAR(75), iban VARCHAR(50), street VARCHAR(50),
                locale VARCHAR(30), postal VARCHAR(15), email VARCHAR(75), phone VARCHAR(20))
BEGIN
    DECLARE last_ins INTEGER;


    START TRANSACTION;

    INSERT INTO Supplier (SupplierName, IBAN, Street, Locale, Postal)
    VALUES (s_name, iban, street, locale, postal);

    IF ROW_COUNT() = 0 THEN
            ROLLBACK;
    END IF;

    SELECT SupplierID INTO last_ins FROM Supplier ORDER BY SupplierID DESC LIMIT 1;

    CALL register_supplier_phone(last_ins, phone);

    IF ROW_COUNT() = 0 THEN
            ROLLBACK;
    END IF;

    CALL register_supplier_email(last_ins, email);

    IF ROW_COUNT() = 0 THEN
            ROLLBACK;
    END IF;

    COMMIT;
END &&

DELIMITER &&
CREATE PROCEDURE register_reservation_new_product (IN product_name VARCHAR(75),
    descript TEXT, baseprice DECIMAL(5,2), supplierid INTEGER, dateofschedule DATETIME,
    stock INT, dateofreservation DATETIME)
BEGIN
    DECLARE product_id INTEGER;

    START TRANSACTION;

    INSERT INTO Product (ProductName, ProductDescription, BasePrice, QuantityInStock)
    VALUES(product_name, descript, baseprice, 0);

    IF ROW_COUNT() = 0 THEN
            ROLLBACK;
    END IF;

    SELECT ProductID INTO product_id FROM Product ORDER BY ProductID DESC LIMIT 1;

    INSERT INTO ProductSupplierFuture(ProductID_psf, SupplierID_psf,
        DateOfSchedule, DateOfReservation, Quantity)
    VALUES(product_id, supplierid, dateofschedule, dateofreservation, stock);

    IF ROW_COUNT() = 0 THEN
            ROLLBACK;
    END IF;

    COMMIT;
END &&

DELIMITER &&
CREATE PROCEDURE register_reservation_exis_product (IN productid INTEGER,
    supplierid INTEGER, dateofschedule DATETIME, dateofreservation DATETIME, quantity INTEGER)
BEGIN

    START TRANSACTION;

    INSERT INTO ProductSupplierFuture(ProductID_psf, SupplierID_psf,
        DateOfSchedule, DateOfReservation, Quantity)
    VALUES(productid, supplierid, dateofschedule, dateofreservation, quantity);

    IF ROW_COUNT() = 0 THEN
            ROLLBACK;
    END IF;
    COMMIT;
END &&

DELIMITER &&

CREATE PROCEDURE register_delivery_product (IN p_id INTEGER,
    s_id INTEGER, p_dod DATETIME, p_quantity INTEGER)
BEGIN

    START TRANSACTION;
```

```
    UPDATE Product
    SET QuantityInStock = QuantityInStock + p_quantity
    WHERE ProductID = p_id;

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

    INSERT INTO ProductSupplierPast
    VALUES(p_id, s_id, p_dod, p_quantity);

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

    COMMIT;
END &&

-- Creates participants
DELIMITER &&
CREATE PROCEDURE add_prod_new_shop_new_part(IN e_id VARCHAR(10), pd_id INTEGER,
                                            part_name VARCHAR(75), part_vat VARCHAR(9),
                                            street VARCHAR(50), locale VARCHAR(30), postal VARCHAR(15), part_bd DATE,
                                            quant INTEGER, phone VARCHAR(20), email VARCHAR(75))
BEGIN
        DECLARE last_ins INTEGER;
    DECLARE last_sale_id INTEGER;
    DECLARE cur_val DECIMAL(5,2);

    START TRANSACTION;

    INSERT INTO Participant(ParticipantName, ParticipantVAT, ParticipantBirthDate,
                            Street, Locale, Postal)
    VALUES (part_name, part_vat, part_bd, street, locale, postal);

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

    SELECT ParticipantID INTO last_ins FROM Participant
    ORDER BY ParticipantID DESC LIMIT 1;

    INSERT INTO ParticipantPhone
    VALUES (last_ins, phone);

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

    IF email IS NOT NULL THEN
        INSERT INTO ParticipantEmail
        VALUES (last_ins, email);

        IF ROW_COUNT() = 0 THEN
                        ROLLBACK;
        END IF;
    END IF;

    INSERT INTO Sale(TotalValue, TotalQuantity, DateOfSale, EmployeeID_s, ParticipantID_s)
    VALUES ("0.00", "0", NULL, e_id, last_ins);

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

    SELECT ReceiptNO INTO last_sale_id FROM Sale
    ORDER BY ReceiptNO DESC LIMIT 1;

    SELECT BasePrice INTO cur_val
    FROM Product
    WHERE ProductID = pd_id;

    INSERT INTO SaleProduct(ReceiptNO_sp, ProductID_sp, CurrentValue, Quantity)
    VALUES (last_sale_id, pd_id, cur_val, quant);

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

    UPDATE Product
    SET QuantityInStock = QuantityInStock - quant
    WHERE ProductID = pd_id;

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;
    COMMIT;
END &&

DELIMITER &&
```

```
CREATE PROCEDURE add_prod_to_new_shopping_cart(IN pa_id INTEGER,
        e_id VARCHAR(10), pd_id INTEGER, quant INTEGER)
BEGIN
    DECLARE cur_stock INTEGER;
    DECLARE cur_val DECIMAL(5,2);
    DECLARE last_sale_id INTEGER;

START TRANSACTION;

    INSERT INTO Sale(TotalValue, TotalQuantity, EmployeeID_s, ParticipantID_s)
    VALUES(0, 0, e_id, pa_id);

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

    SELECT ReceiptNO INTO last_sale_id FROM Sale
    ORDER BY ReceiptNO DESC LIMIT 1;

    SELECT BasePrice INTO cur_val
    FROM Product
    WHERE ProductID = pd_id;

    INSERT INTO SaleProduct(ReceiptNO_sp, ProductID_sp, CurrentValue, Quantity)
    VALUES(last_sale_id, pd_id, cur_val, quant);

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

    UPDATE Product
    SET QuantityInStock = QuantityInStock - quant
    WHERE ProductID = pd_id;

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;
    COMMIT;
END &&

DELIMITER &&

CREATE PROCEDURE add_prod_to_shopping_cart(IN s_id INTEGER, pa_id INTEGER,
    pd_id INTEGER, quant INTEGER)
BEGIN
        DECLARE cur_stock INTEGER;
    DECLARE cur_val DECIMAL(5,2);


    START TRANSACTION;

    SELECT BasePrice INTO cur_val
    FROM Product
    WHERE ProductID = pd_id;

    INSERT INTO SaleProduct(ReceiptNO_sp, ProductID_sp, CurrentValue, Quantity)
    VALUES (s_id, pd_id, cur_val, quant);

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

    UPDATE Product
    SET QuantityInStock = QuantityInStock - quant
    WHERE ProductID = pd_id;

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;
    COMMIT;
END &&

DELIMITER &&
CREATE PROCEDURE register_sale (IN s_id INTEGER, s_dos DATETIME)
BEGIN
DECLARE no_pds INTEGER;
DECLARE pd_id INTEGER;
    DECLARE s_totval DECIMAL(5,2);
    DECLARE s_totquant INTEGER;


    START TRANSACTION;

    SELECT SUM(SP.Quantity) INTO s_totquant
    FROM SaleProduct AS SP
    INNER JOIN Sale AS S ON S.ReceiptNO = SP.ReceiptNO_sp
    WHERE S.ReceiptNO = s_id;

    SELECT SUM(SP.Quantity * SP.CurrentValue) INTO s_totval
    FROM SaleProduct AS SP
    INNER JOIN Sale AS S ON S.ReceiptNO = SP.ReceiptNO_sp
```

```
    WHERE S.ReceiptNO = s_id;

        UPDATE Sale
    SET TotalValue = s_totval, TotalQuantity = s_totquant, DateOfSale = s_dos
    WHERE ReceiptNO = s_id;

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

    COMMIT;
END &&

DELIMITER &&

CREATE PROCEDURE cancel_ongoing_sale (IN s_id INTEGER)
BEGIN
    DECLARE no_pds INT;
    DECLARE pd_id INT;
    DECLARE pd_quant_sold INT;


    START TRANSACTION;

    -- Count the number of products in the sale
    SELECT COUNT(SP.ProductID_sp) INTO no_pds
    FROM SaleProduct AS SP
    WHERE SP.ReceiptNO_sp = s_id;

    -- Loop to add products back to stock
    REPEAT
        SELECT SP.ProductID_sp, SP.Quantity INTO pd_id, pd_quant_sold
        FROM SaleProduct AS SP
        INNER JOIN Product AS P ON SP.ProductID_sp = P.ProductID
        WHERE SP.ReceiptNO_sp = s_id
        ORDER BY SP.ProductID_sp DESC
        LIMIT 1;

        UPDATE Product
        SET QuantityInStock = QuantityInStock + pd_quant_sold
        WHERE ProductID = pd_id;

        IF ROW_COUNT() = 0 THEN
                    ROLLBACK;
        END IF;

        DELETE FROM SaleProduct
        WHERE ReceiptNO_sp = s_id AND ProductID_sp = pd_id;

        IF ROW_COUNT() = 0 THEN
                    ROLLBACK;
        END IF;

        SET no_pds = no_pds - 1;

    UNTIL no_pds = 0
    END REPEAT;

    DELETE FROM Sale
    WHERE ReceiptNO = s_id;

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

    DELETE FROM Participant
    WHERE ParticipantID NOT IN (SELECT ParticipantID_s FROM Sale);

    COMMIT;
END &&

DELIMITER ;

DELIMITER &&
CREATE PROCEDURE register_new_employee (IN e_id VARCHAR(10), e_name VARCHAR(75),
                vat VARCHAR(9), bd DATE, street VARCHAR(50), locale VARCHAR(30),
                postal VARCHAR(15), manager VARCHAR(10), phone VARCHAR(20), email VARCHAR(75))
BEGIN
    START TRANSACTION;

    INSERT INTO Employee
    VALUES (e_id, e_name, vat, bd, street, locale, postal, manager);

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

        CALL register_employee_email(e_id, email);

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
```

```
    END IF;

        CALL register_employee_phone(e_id, phone);

    IF ROW_COUNT() = 0 THEN
            ROLLBACK;
    END IF;
    COMMIT;
END &&

DELIMITER &&
CREATE PROCEDURE register_employee_email (IN e_id VARCHAR(10), e_email VARCHAR(75))
BEGIN
        DECLARE check_error BOOLEAN DEFAULT FALSE;
        DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET check_error = TRUE;

        START TRANSACTION;

        INSERT INTO EmployeeEmail
            VALUES (e_id, e_email);

        IF ROW_COUNT() = 0 THEN
            ROLLBACK;
    END IF;
    COMMIT;
END &&

DELIMITER &&
CREATE PROCEDURE register_employee_phone (IN e_id VARCHAR(10), e_phone VARCHAR(20))
BEGIN
        DECLARE check_error BOOLEAN DEFAULT FALSE;

        DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET check_error = TRUE;

        START TRANSACTION;

        INSERT INTO EmployeePhone
        VALUES (e_id, e_phone);

        IF check_error = FALSE THEN
            COMMIT;
        ELSE
            ROLLBACK;
        END IF;
END &&

DELIMITER &&
CREATE PROCEDURE register_participant_email (IN p_id INTEGER, p_email VARCHAR(75))
BEGIN
        DECLARE check_error BOOLEAN DEFAULT FALSE;

        DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET check_error = TRUE;

        START TRANSACTION;

        INSERT INTO ParticipantEmail
        VALUES (p_id, p_email);

        IF check_error = FALSE THEN
            COMMIT;
        ELSE
            ROLLBACK;
        END IF;
END &&

DELIMITER &&
CREATE PROCEDURE register_participant_phone (IN p_id INTEGER, p_phone VARCHAR(20))
BEGIN
        DECLARE check_error BOOLEAN DEFAULT FALSE;

        DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET check_error = TRUE;

        START TRANSACTION;

        INSERT INTO ParticipantPhone
        VALUES (p_id, p_phone);

        IF check_error = FALSE THEN
            COMMIT;
        ELSE
            ROLLBACK;
        END IF;
END &&

DELIMITER &&
CREATE PROCEDURE register_supplier_email (IN s_id INTEGER, s_email VARCHAR(75))
BEGIN
        DECLARE check_error BOOLEAN DEFAULT FALSE;

        DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET check_error = TRUE;
```

```
        START TRANSACTION;

        INSERT INTO SupplierEmail
        VALUES (s_id, s_email);

        IF check_error = FALSE THEN
                COMMIT;
        ELSE
                ROLLBACK;
        END IF;
END &&

DELIMITER &&
CREATE PROCEDURE register_supplier_phone (IN s_id INTEGER, s_phone VARCHAR(20))
BEGIN
        DECLARE check_error BOOLEAN DEFAULT FALSE;

        DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET check_error = TRUE;

        START TRANSACTION;

        INSERT INTO SupplierPhone
        VALUES (s_id, s_phone);

        IF check_error = FALSE THEN
        COMMIT;
        ELSE
                ROLLBACK;
        END IF;
END &&

DELIMITER &&
CREATE PROCEDURE register_new_event (IN e_name VARCHAR(75),
        e_descr TEXT,
        e_beg DATETIME, e_fin DATETIME, e_capacity INTEGER,
        t_descr TEXT, t_price DECIMAL(5,2))
BEGIN
        DECLARE t_stock INT;
        START TRANSACTION;

    INSERT INTO EventCal (EventName, EventDescription, EventStart, EventEnd, Capacity)
    VALUES (e_name, e_descr, e_beg, e_fin, e_capacity);

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

    SET t_stock = e_capacity;
    IF e_capacity = 0 THEN
        SET t_stock = 94967294; -- max int
    END IF;

    INSERT INTO Product (ProductName, ProductDescription, BasePrice, QuantityInStock)
    VALUES (e_name, t_descr, t_price, t_stock);

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

    COMMIT;
END &&

DELIMITER &&
CREATE PROCEDURE assign_employee_event(IN e_id INTEGER, empl_id VARCHAR(10))
BEGIN
    START TRANSACTION;

    -- Insert into EventEmployee
    INSERT INTO EventEmployee
    VALUES (e_id, empl_id);

    IF ROW_COUNT() = 0 THEN
                ROLLBACK;
    END IF;

    -- Commit the transaction if all operations are successful
    COMMIT;
END &&
```

4.6 Anexo 6

```
USE mademoiselle_borges;

-- check all closed sales (45)
SELECT ReceiptNO
        FROM Sale
        WHERE DateOfSale IS NOT NULL;

-- check all participants of all events (47)
SELECT ParticipantID, ParticipantName
        FROM Participant;

-- check all products (49)
SELECT ProductID, ProductName
 FROM Product;

-- check all suppliers (54)
SELECT SupplierID, SupplierName
 FROM Supplier;

-- check all Employees (55)
SELECT EmployeeID, EmployeeName
        FROM Employee;

-- check all events (56)
SELECT EventID, EventName
        FROM EventCal;

-- check participant with highest volume sales associated (59)
SELECT P.ParticipantID, P.ParticipantName, SUM(S.TotalQuantity) AS Volume
        FROM Sale AS S INNER JOIN Participant AS P
                ON P.ParticipantID = S.ParticipantID_s
        GROUP BY P.ParticipantID
                ORDER BY Volume DESC -- COUNT(P.id = S.participant_id_s) DESC
LIMIT 1;

-- check which event has most volume sales (60)
SELECT EV.EventID, EV.EventName, SUM(S.TotalQuantity) AS quant
        FROM EventCal AS EV
                INNER JOIN Sale AS S
                        ON S.DateOfSale BETWEEN EV.EventStart AND EV.EventEnd
                GROUP BY EV.EventID, EV.EventName
                        ORDER BY quant DESC
LIMIT 1;

-- check event with highest rate participation (61)
SELECT EV.EventID, EV.EventName, SUM(SP.Quantity) / EV.Capacity * 100 AS rate
        FROM EventCal as EV INNER JOIN SaleProduct as SP
                INNER JOIN Product AS P
                        ON P.ProductID = SP.ProductID_sp and P.ProductName = EV.EventName
                GROUP BY EV.EventID, EV.EventName
                        ORDER BY rate DESC
                LIMIT 1;

-- check the participant with the highest value in sales (93)
SELECT P.ParticipantID, P.ParticipantName, SUM(S.TotalValue) AS totVal
        FROM Sale AS S INNER JOIN Participant AS P
                        ON P.ParticipantID = S.ParticipantID_s
        GROUP BY P.ParticipantID, P.ParticipantName
                ORDER BY totVal DESC
LIMIT 1;

-- check the event with the most value in sales (94)
SELECT EV.EventID, EV.EventName, SUM(S.TotalValue) AS totVal
        FROM EventCal AS EV INNER JOIN Sale as S
                ON S.DateOfSale BETWEEN EV.EventStart AND EV.EventEnd
        GROUP BY EV.EventID, EV.EventName
                ORDER BY totVal DESC
LIMIT 1;
```

4.7 Anexo 7

```
USE mademoiselle_borges;

-- check who manages an Employee (43)
DELIMITER &&
CREATE FUNCTION check_manager (id VARCHAR(10))
RETURNS VARCHAR(10)
DETERMINISTIC
 BEGIN
    DECLARE manager VARCHAR(10);
    SELECT EmployeeID_e INTO manager
        FROM Employee AS E
        WHERE id = E.EmployeeID;
    RETURN manager;
END &&

DELIMITER &&
CREATE FUNCTION check_participant_in_sale (id INTEGER)
RETURNS INTEGER
DETERMINISTIC
  BEGIN
     DECLARE partic INTEGER;
     SELECT ParticipantID_s INTO partic
         FROM Sale AS S
         WHERE S.ReceiptNO = id;
     RETURN partic;
END &&


-- check events in a given timespan (71)
DELIMITER &&
CREATE PROCEDURE EventsInTimespan(IN firstday DATETIME, IN lastday DATETIME)
 BEGIN
        SELECT EventID, EventName
                FROM EventCal AS EV
                        WHERE (EV.EventStart BETWEEN firstday AND lastday) AND (EV.EventEnd BETWEEN firstday AND lastday);
 END
 &&
```

4.8 Anexo 8

```
CREATE USER IF NOT EXISTS 'hen'@'localhost';
CREATE USER IF NOT EXISTS 'mii'@'localhost';
CREATE USER IF NOT EXISTS 'hom'@'localhost';
CREATE USER IF NOT EXISTS 'employee'@'localhost';
CREATE USER IF NOT EXISTS 'marketing'@'localhost';
CREATE USER IF NOT EXISTS 'guest'@'localhost';

GRANT  ALL PRIVILEGES ON mademoiselle_borges.* TO 'hen'@'localhost';
GRANT  GRANT OPTION ON mademoiselle_borges.* TO 'hen'@'localhost';
GRANT ALL PRIVILEGES ON mademoiselle_borges.* TO 'mii'@'localhost';
GRANT  GRANT OPTION ON mademoiselle_borges.* TO 'mii'@'localhost';
GRANT ALL PRIVILEGES  ON mademoiselle_borges.* TO 'hom'@'localhost';
GRANT GRANT OPTION ON mademoiselle_borges.* TO 'hom'@'localhost';

GRANT SELECT ON mademoiselle_borges.EventCal TO 'guest'@'localhost';
GRANT SELECT ON mademoiselle_borges.Product TO 'guest'@'localhost';

GRANT SELECT ON mademoiselle_borges.EventCal TO 'employee'@'localhost';
GRANT SELECT ON mademoiselle_borges.SaleNoValue TO 'employee'@'localhost';
GRANT EXECUTE ON PROCEDURE register_sale TO 'employee'@'localhost';
GRANT SELECT ON mademoiselle_borges.Participant TO 'employee'@'localhost';
GRANT EXECUTE ON PROCEDURE add_prod_new_shop_new_part TO 'employee'@'localhost';
GRANT SELECT ON mademoiselle_borges.Product TO 'employee'@'localhost';
GRANT EXECUTE ON PROCEDURE register_reservation_new_product TO 'employee'@'localhost';
GRANT EXECUTE ON PROCEDURE register_reservation_exis_product TO 'employee'@'localhost';
GRANT EXECUTE ON PROCEDURE register_delivery_product TO 'employee'@'localhost';
GRANT EXECUTE ON PROCEDURE add_prod_to_new_shopping_cart TO 'employee'@'localhost';
GRANT EXECUTE ON PROCEDURE add_prod_to_shopping_cart TO 'employee'@'localhost';
GRANT EXECUTE ON PROCEDURE cancel_ongoing_sale TO 'employee'@'localhost';
GRANT EXECUTE ON PROCEDURE register_supplier TO 'employee'@'localhost';
GRANT EXECUTE ON PROCEDURE register_supplier_email TO 'employee'@'localhost';
GRANT EXECUTE ON PROCEDURE register_supplier_phone TO 'employee'@'localhost';

GRANT SELECT ON mademoiselle_borges.EventCal TO 'marketing'@'localhost';
GRANT EXECUTE ON PROCEDURE register_new_event TO 'marketing'@'localhost';

DELIMITER &&
CREATE PROCEDURE grantRevokePermissions()
BEGIN
    IF (CURTIME() BETWEEN '07:00:00' AND '23:59:59') OR (CURTIME() BETWEEN '00:00:00' AND '02:00:00') THEN
        GRANT USAGE ON mademoiselle_borges.* TO 'guest'@'localhost';
        GRANT USAGE ON mademoiselle_borges.* TO 'marketing'@'localhost';
        GRANT USAGE ON mademoiselle_borges.* TO 'employee'@'localhost';
        GRANT USAGE ON mademoiselle_borges.* TO 'hom'@'localhost';
        GRANT USAGE ON mademoiselle_borges.* TO 'mii'@'localhost';
        GRANT USAGE ON mademoiselle_borges.* TO 'hen'@'localhost';
    ELSE
        REVOKE USAGE ON mademoiselle_borges.* FROM 'guest'@'localhost';
        REVOKE USAGE ON mademoiselle_borges.* FROM 'marketing'@'localhost';
        REVOKE USAGE ON mademoiselle_borges.* FROM 'employee'@'localhost';
        END IF;
END &&
CREATE EVENT close_open
        ON SCHEDULE EVERY 1 HOUR
        DO CALL grantRevokePermissions();

-- SHOW GRANTS FOR 'guest'@'localhost';
FLUSH PRIVILEGES;
```

4.9 Anexo 9

```
USE mademoiselle_borges;

CALL register_new_employee("ADMIN00001", "Henrique Borges",
                                                                                    "382923812", "1968-12-24", "Rua do blah",
                                                                                    "Eventopolis", "1111-111", NULL, "+1111111111
CALL register_new_employee("ADMIN00002", "Maria Ivanovna Ivanova",
                                                                                    "129293919", "1999-04-13", "Rua do blah",
                                                                                    "Eventopolis", "1111-111", NULL, "+2222222222
CALL register_new_employee("ADMIN00003", "Herr Otto Mustermann",
                                                                "198263843", "2002-07-19", "Rua do blah",
                                                                "Eventopolis", "1111-111", NULL, "+33333333333","herr@motto.et");
CALL register_new_employee("JANIT00001", "Jacinto Ivanovich",
                                                        "192382742", "1979-04-13", "Rua do blah",
                                "Eventopolis", "1111-111", NULL, "+4444444444","orig@jacinto.et");
CALL register_new_employee("JANIT00002", "Jacinto Ivanovich Sr",
                                                        "192382743", "1959-04-13", "Rua do blah",
                                "Eventopolis", "1111-111", "JANIT00001", "+555553555","sen@jacinto.et");
CALL register_new_employee("JANIT00003", "Jacinto Ivanovich Jr",
                                                        "192382744", "1999-04-13", "Rua do blah",
                                "Eventopolis", "1111-111", "JANIT00001", "+6666666666","jr@jacinto.et");
CALL register_new_employee("MARKT00001", "Miguel Mata Migalhas",
                                                        "999999999", "1974-04-25", "Avenida dos Vampiros",
                                "Eventopolis", "5555-555", NULL, "+7777777777","mimami@mar.et");
CALL register_new_employee("MARKT00002", "Ruis Luis",
                                                        "444444444", "2004-04-04", "Rua da Praça Proibida",
                                "Lon Gedaki", "4444-444", "MARKT00001", "+8888888888","uiui@mar.et");
CALL register_new_employee("SALES00001", "Artur Doffensmirtz",
                                                        "333333333", "1979-03-03", "Rua do Ornitorrinco",
                                "Eventopolis", "1111-132", "MARKT00002", "+9999999999","doff@krr.et");
CALL register_new_employee("SALES00002", "Benilde Portas de Madeira",
                                                        "222222222", "1970-02-22", "Rua do Centro nº1 , 5B",
                                "Eventopolis", "1111-221", "SALES00001", "+10101101010","bpm@sal.et");
CALL register_new_employee("SECUR00001", "José Carlos Malandro",
                                                        "111111111", "2000-11-11", "Rua do Centro nº1 , 5A",
                                "Eventopolis", "1111-221", NULL, "+20202022020","jcm@secur.et");
CALL register_new_employee("JANIT00004", "Aknostr Grust Hings",
                                                        "666666666", "1996-01-29", "Trilho da Bananeira",
                                "Hsombra", "1169-111", "JANIT00001", "+303033030303","agh@jacinto.et");
CALL register_new_employee("SECUR00002", "Alice Grande Pequena",
                                                        "777777777", "1968-07-21", "Travessa da Lua",
                                "Ham Strong City", "7777-221", "SECUR00001", "+40404044404","agp@secur.et");
CALL register_new_employee("MARKT00003", "Bernardo Fernado Ferrari",
                                                        "888888888", "1988-08-15", "Rua da Boa Corrida",
                                "Eventopolis", "8888-111", "MARKT00001", "+5050505050","bff@mar.et");
CALL register_new_employee("SALES00003", "Ines Luis Grust Hings",
                                                        "666666661", "1996-01-11", "TRilho da Bananeira",
                                "Hsombra", "1169-111", "SALES00001", "+6060660606","ilu@sal.et");
CALL register_new_employee("MARKT00004", "Yuppi May Catrindottir Alavason",
                                                        "101010101", "2000-11-11", "Rua Doutor Felis Felizardo Felisberto",
                                "Eventopolis", "1010-101", "MARKT00001", "+707707700707","ymca@mar.et");

-- Inserção de Eventos
CALL register_new_event("Christmas Booksale - Mademoiselle Borges",
                        "Discover a world of literary wonders at our Christmas Booksale , in honor of the memory of Lady Borges",
                        "2023-12-19 15:30:00", "2024-01-02 01:59:59",
                        0, "Christmas Booksale Ticket", "5.2");
CALL register_new_event ("Sueca Tournament",
                                                "Tournament of a famous card game called 'Sueca'",
                         "2024-02-02 21:30:00","2024-02-04 15:00:00",
                         50,"Ticket for the Tournament","3.0");
CALL register_new_event ("Theater Piece: 'The Hidden Madame'",
                                                "Attend a lively and entertainign theater performance about the Trust Issues of a certain Mad
                         "2023-12-15 18:30:00","2023-12-15 20:30:00",
                         150,"The Madame's Ticket","4.5");
CALL register_new_event ("Festive Procession de Sao Celestino de Rabo de Peixe",
                                                "Religious Festivity",
                         "2024-02-05 15:00:00","2024-02-06 19:30:00",
                         150,"The Religious Ticket","1.0");
CALL register_new_event ("Puppet Show: 'Red Ridding Hood'",
                                                "Delight in a colourful puppet performance that will captivate both children and adults alike
                         "2024-02-07 13:30:00","2024-02-07 20:30:00",
                         40,"The Puppet Ticket","2.0");
CALL register_new_event ("Arts and Crafts Fair",
                                                "Explore local arts and crafts , where families can discover unique handmade creations and par
                         "2024-02-08 13:00:00","2024-02-09 20:30:00",
                         80,"The Artiste Ticket","8.0");

-- Inserçao de funcionarios associados a um dado evento

CALL assign_employee_event(1, "SALES00001");
CALL assign_employee_event(1, "SALES00002");
CALL assign_employee_event(1, "SALES00003");
CALL assign_employee_event(1, "JANIT00001");
CALL assign_employee_event(1, "JANIT00002");
CALL assign_employee_event(1, "JANIT00003");
CALL assign_employee_event(1, "JANIT00004");
CALL assign_employee_event(1, "SECUR00001");
CALL assign_employee_event(1, "SECUR00002");
```

```
CALL assign_employee_event(1, "MARKT00001");
CALL assign_employee_event(1, "MARKT00002");
CALL assign_employee_event(1, "MARKT00003");
CALL assign_employee_event(1, "MARKT00004");

CALL assign_employee_event(2, "SALES00001");
CALL assign_employee_event(2, "SALES00002");
CALL assign_employee_event(2, "SALES00003");
CALL assign_employee_event(2, "JANIT00001");
CALL assign_employee_event(2, "JANIT00002");
CALL assign_employee_event(2, "JANIT00003");
CALL assign_employee_event(2, "JANIT00004");
CALL assign_employee_event(2, "SECUR00001");
CALL assign_employee_event(2, "SECUR00002");
CALL assign_employee_event(2, "MARKT00001");
CALL assign_employee_event(2, "MARKT00002");
CALL assign_employee_event(2, "MARKT00003");
CALL assign_employee_event(2, "MARKT00004");

CALL assign_employee_event(3, "SALES00001");
CALL assign_employee_event(3, "SALES00002");
CALL assign_employee_event(3, "SALES00003");
CALL assign_employee_event(3, "JANIT00001");
CALL assign_employee_event(3, "JANIT00002");
CALL assign_employee_event(3, "JANIT00003");
CALL assign_employee_event(3, "JANIT00004");
CALL assign_employee_event(3, "SECUR00001");
CALL assign_employee_event(3, "SECUR00002");
CALL assign_employee_event(3, "MARKT00001");
CALL assign_employee_event(3, "MARKT00002");
CALL assign_employee_event(3, "MARKT00003");
CALL assign_employee_event(3, "MARKT00004");

CALL assign_employee_event(4, "SALES00001");
CALL assign_employee_event(4, "SALES00002");
CALL assign_employee_event(4, "SALES00003");
CALL assign_employee_event(4, "JANIT00001");
CALL assign_employee_event(4, "JANIT00002");
CALL assign_employee_event(4, "JANIT00003");
CALL assign_employee_event(4, "JANIT00004");
CALL assign_employee_event(4, "SECUR00001");
CALL assign_employee_event(4, "SECUR00002");
CALL assign_employee_event(4, "MARKT00001");
CALL assign_employee_event(4, "MARKT00002");
CALL assign_employee_event(4, "MARKT00003");
CALL assign_employee_event(4, "MARKT00004");

CALL assign_employee_event(5, "SALES00001");
CALL assign_employee_event(5, "SALES00002");
CALL assign_employee_event(5, "SALES00003");
CALL assign_employee_event(5, "JANIT00001");
CALL assign_employee_event(5, "JANIT00002");
CALL assign_employee_event(5, "JANIT00003");
CALL assign_employee_event(5, "JANIT00004");
CALL assign_employee_event(5, "SECUR00001");
CALL assign_employee_event(5, "SECUR00002");
CALL assign_employee_event(5, "MARKT00001");
CALL assign_employee_event(5, "MARKT00002");
CALL assign_employee_event(5, "MARKT00003");
CALL assign_employee_event(5, "MARKT00004");

CALL assign_employee_event(6, "SALES00001");
CALL assign_employee_event(6, "SALES00002");
CALL assign_employee_event(6, "SALES00003");
CALL assign_employee_event(6, "JANIT00001");
CALL assign_employee_event(6, "JANIT00002");
CALL assign_employee_event(6, "JANIT00003");
CALL assign_employee_event(6, "JANIT00004");
CALL assign_employee_event(6, "SECUR00001");
CALL assign_employee_event(6, "SECUR00002");
CALL assign_employee_event(6, "MARKT00001");
CALL assign_employee_event(6, "MARKT00002");
CALL assign_employee_event(6, "MARKT00003");
CALL assign_employee_event(6, "MARKT00004");

-- Inserçao de fornecedores
CALL register_supplier ("World of Books","ET493958654897440","Paper Valley , nº234","Calecornia","2710-008-ZZZ" ,
                                                "world-books@mdl.et", "+631854-677219");
CALL register_supplier ("Lorbeer und Lorbeer Materials Lda","GE905440777332294","Teller von Lachen , nº91 , Piso 3","Inglostadt","4000-003",
                                                "materialslul@matino.ge", "+68349-333-10102");
CALL register_supplier ("Badhiya Kapaas","IND66979302668492","Traayamph Skvaayar , nº124","Goa","4485-4432",
                                                "fashionkapaas@badkap.in", "+222543296584-3");
CALL register_supplier ("AltaPelle Distributore","IT40330399569611111","Via Carlos Vallente , nº27","Piza","3-55-88895",
                                                "altpe@dist.it", "+444-8599339-2111");
CALL register_supplier ("Edital Impressao","PT96797397339295","Alameda da Vitoria , n234","Guimarães","4800-234",
                                                "editalimp@edital.pt", "+351252111116");
CALL register_supplier ("Diavges Sympan Lda","GR775588994433225566","Leoforos tou Mikrolameira , n123","Tebas","500-692-4",
                                                "diasymp@ludico.gr", "+34002-995-033-869");
CALL register_supplier ("Kreativnyye Resheniya Importer","RU0020105889044430","Arkticheskiy prospekt , n94","Ufa","38948-222-01",
                                                "kreat@imprt.ru", "+643333020-4-3111");
CALL register_supplier ("InovaPrint Solucciones Graficas","ES6854920492946841","Rua dela dama del dia , n20 , Oficina I ","Mirallá","9000-009-AA
```

```
                                                "inovaprint@inova.es", "+583920007-90");
CALL register_supplier ("Sangbeoggyuoppa Mungujeom","SCOR6695593388200060","jong-ijib geoli ,n234","Jeju","2750-008",
                                                "mungsbeog@psg.sc", "+25839443922204");
CALL register_supplier ("Páginas Vivas Publicações Inc.","BR99900299294958811","Rua Anita Carlota ,n111","Curitiba","4740-081",
                                                "paginasvivas@vivas.br", "+33335434890250-111");
CALL register_supplier ("Himno ng Kagalakan Ltda.","FP22294939205R8687853" ,"pinggan ng mga bulaklak ,n001","Manila","747484-444",
                                                "him@ltda.fp", "+3512521111121");

-- Inserção de Produtos
CALL register_reservation_new_product ("Pacote Caneta Azul Bic",
                                                                        "Pacote de 10 Canetas da Marca Bic , cor azul",
                                                "3.89","3","2023-11-30 13:00:00",35,"2023-10-30 09:00:00");
CALL register_reservation_new_product ("Borracha Branca","2 Borrachas MAPPED",1.10,
                                                                        3,"2023-11-30 13:00:00",21,"2023-10-30 09:00:00");
CALL register_reservation_new_product ("Pacote Caneta Vermelha Bic","Pacote de 10 Canetas da Marca Bic , cor vermelha",3.89,
                                                                        3,"2023-11-30 13:00:00",30,"2023-10-30 09:00:00");
CALL register_reservation_new_product ("Lápis de Cor Faber-Castell","Caixa com 36 lápis de cor vibrantes",29.99,
                                                                        3,"2023-11-30 13:00:00",13,"2023-10-30 09:00:00");
CALL register_reservation_new_product ("Livro para Colorir","Livro com imagens de locais e eventos importantes de Eventopolis para colorir ",6
                                                                        1,"2023-12-10 08:30:00",7,"2023-11-10 19:00:00");
CALL register_reservation_new_product ("Livro 'Historia de Eventopolis , do século XV até modernidade'",
                                                                        "Livro do autor Benedito Leão , onde ele escreve sobre vários eventos
                                                                        1,"2023-12-10 08:30:00",6,"2023-11-10 19:00:00");
CALL register_reservation_new_product ("Assinatura Mensal de Livros Locais",
                                                                        "Receba mensalmente um livro de um autor local ou com temática da c
                                                                        1,"2023-12-10 08:30:00",22,"2023-11-10 19:00:00");
CALL register_reservation_new_product ("Caderno Pautado","Caderno com 100 folhas pautadas",1.50,
                                                                        9,"2024-01-25 16:00:00",30,"2023-12-25 10:00:00");
CALL register_reservation_new_product ("Caderno Quadriculado","Caderno com 100 folhas quadriculadas",1.50,
                                                                        9,"2024-01-25 16:00:00",30,"2023-12-25 10:00:00");
CALL register_reservation_new_product ("Caneca 'I LOVE Eventopolis'","Caneca de ceramica com o slogan da cidade",7.50,
                                                                        2,"2023-11-30 14:45:00",20,"2023-10-30 17:00:00");
CALL register_reservation_new_product ("Jogo de Tabuleiro 'Descubra Eventopolis'",
                                                                        "Jogo interativo para explorar a história e geografia de Eventopolis
                                                                        11,"2023-12-10 13:00:00",5,"2023-11-10 11:11:11");
CALL register_reservation_new_product ("Livro de Receitas 'Sabores de Eventopolis'",
                                                                        "Receitas locais e tradicionais da região copiladas pelo enigmatico
                                                                        10,"2023-12-10 00:00:48",4,"2023-11-10 15:00:00");
CALL register_reservation_new_product ("Agenda Anual Gatos 2024",
                                                                        "Agenda com páginas para cada dia do ano com imagens de vários felin
                                                                        4,"2024-03-18 09:30:00",100,"2024-02-18 14:00:00");
CALL register_reservation_new_product ("Conjunto de Postais de Eventopolis",
                                                                        "Pacote com 10 postais ilustrados de vários monumentos da cidade , in
                                                                        5,"2023-12-10 13:00:00",23,"2023-11-10 12:45:00");
CALL register_reservation_new_product ("Guia Turístico de Eventopolis","Livro informativo com destinos , atrações e lendas da cidade",11.50,
                                                                        8,"2023-12-10 13:00:00",10,"2023-11-10 12:45:00");

CALL register_reservation_new_product ("T-shirt 'I LOVE EVENTOPOLIS'","Uma t-shirt de tamanho S com o famoso logo ",10.00,
                                                                        2,"2024-03-20 11:00:00",30,"2023-11-16 10:00:00");
CALL register_reservation_new_product ("Bloco de Notas de Couro","Bloco de notas elegante com capa de couro sintético",9.75,
                                                                        7,"2024-03-20 08:00:00",12,"2023-11-16 19:45:00");
CALL register_reservation_new_product ("Livro 'A vida secreta de Carlos Valente'","Uma Biografia escrita por Maria Alves , acerca da escandalosa
                                                                        1,"2024-03-19 08:30:00",9,"2023-11-10 19:00:00");
CALL register_reservation_new_product ("Bloco de Desenho A4","Bloco de papel para desenho tamanho A4, 50 folhas",12.95,
                                                                        9,"2024-03-19 16:00:00",19,"2023-12-25 10:00:00");

-- Inserçao de produtos encomendados pelos respetivos fornecedores

CALL register_delivery_product (7,3,"2023-11-30 13:00:00", 1);
CALL register_delivery_product (8,3,"2023-11-30 13:00:00", 1);
CALL register_delivery_product (9,3,"2023-11-30 13:00:00", 1);
CALL register_delivery_product (10,3,"2023-11-30 13:00:00", 1);
CALL register_delivery_product (11,1,"2023-12-10 08:30:00", 1);
CALL register_delivery_product (12,1,"2023-12-10 08:30:00", 1);
CALL register_delivery_product (13,1,"2023-12-10 08:30:00", 1);
CALL register_delivery_product (14,9,"2024-01-25 16:00:00", 1);
CALL register_delivery_product (15,9,"2024-01-25 16:00:00", 1);
CALL register_delivery_product (16,2,"2023-11-30 14:45:00", 1);
CALL register_delivery_product (17,11,"2023-12-10 13:00:00", 1);
CALL register_delivery_product (18,10,"2023-12-10 00:00:48", 1);
CALL register_delivery_product (19,4,"2024-03-18 09:30:00", 1);
CALL register_delivery_product (20,5,"2023-12-10 13:00:00", 1);
CALL register_delivery_product (21,8,"2023-12-10 13:00:00", 1);
CALL register_delivery_product (22,2,"2024-03-20 11:00:00", 1);
CALL register_delivery_product (23,7,"2024-03-20 08:00:00", 1);
CALL register_delivery_product (24,1,"2024-03-19 08:30:00", 1);
CALL register_delivery_product (25,1,"2024-03-19 16:00:00", 1);

-- Inserçao de carrinho de compras (Participante novo)

CALL add_prod_new_shop_new_part("SALES00001", 1,
                                                "Ana Alves Aves",NULL,"Rua Antonio Malheiro","Atchim","4406-333","2000-10-03",
                                                1, "+351101092200", "aaa@machadinha.et");
CALL register_sale (1, "2023-12-19 15:30:00");
CALL add_prod_new_shop_new_part("SALES00001", 1,
                                                "Beto Bartolomeu Broa","500402204","Sitio do Pica-pau","Amarelo","5504-999","1960-01-03",
                                                1, "+351101092201", "bbb@tvi.et");
CALL register_sale (2, "2023-12-19 15:31:00");

CALL add_prod_new_shop_new_part("SALES00002", 1,
                                                "Pavel Nguissa","330209763","Avenida da Gritaria","Onda","1039-586","2004-02-29",
```

```
                                                    1, "+24199504423", "sigmat@golo.afk");
CALL register_sale (3, "2023-12-19 15:32:00");

CALL add_prod_new_shop_new_part("SALES00002", 3,
                                                    "Eduardo Ernesto","667438290","Travessa Entre-Riachos","Puarto","7777-431","2003-08-30",
                                                    1, "+351101092204", "ee@hehe.he");
CALL register_sale (4, "2023-12-15 18:30:00");

CALL add_prod_new_shop_new_part("SALES00003", 3,
                                                    "Jasper Lindkrug","110097832","Rua Mas Marias","Gotland","6000-256","1936-05-23",
                                                    1, "+24199504425", "jaspenkrug@ger.sw");
CALL register_sale (5, "2023-12-15 18:31:00");

CALL add_prod_new_shop_new_part("SALES00001", 2,
                                                    "Carlos Coentrao Cavaco",null,"Rua do Palhaco Mudo","Palhaca","1566-839","1970-05-04",
                                                    1, "+351101092202", "ccc@cerpente.et");
CALL register_sale (6, "2024-02-02 21:30:00");

CALL add_prod_new_shop_new_part("SALES00002", 2,
                                                    "Diofantina Duarte Dias","295839290","Avenida da Real Virtualidade","Mato da Beira","0119-703"
                                                    1, "+351101092203", "ddd@nutmax.yp");
CALL register_sale (7, "2024-02-02 21:31:00");

CALL add_prod_new_shop_new_part("SALES00003", 2,
                                                    "Jay Okocha","069411194","Travessa Costa Dourada","Praia de Matosinhos","7732-410","2016-07-10
                                                    1, "+24199504424", "jjayokoctha@psg.fr");
CALL register_sale (8, "2024-02-02 21:32:00");

CALL add_prod_new_shop_new_part("SALES00002", 4,
                                                    "Fhatima Alkhashour",null,"Bairro do Gato Maltes","Pianno Valletta","4403-752","2013-12-25",
                                                    1, "+351101092205", "fhaal@tapoporto.uz");
CALL register_sale (9, "2024-02-05 15:00:00");

CALL add_prod_new_shop_new_part("SALES00003", 4,
                                                    "Gilberto Gloria Gomes Galhardo","555444333","Praca of Alliances","En-Guardinton","6659-330","
                                                    1, "+351101092206", "gegegege@togeee.ge");
CALL register_sale (10, "2024-02-05 15:01:00");

CALL add_prod_new_shop_new_part("SALES00002", 4,
                                                    "Yang Lee-sang","559302119","Bairro Tia Ying-Yang","Principado de Qin","5094-333","2174-10-31"
                                                    1, "+24199504426", "ylsneg@soc.pts");
CALL register_sale (11, "2024-02-05 15:02:00");

CALL add_prod_new_shop_new_part("SALES00003", 5,
                                                    "Blake Edwards","448395907","Rua Pantera Cor-de-Rosa","Panan-Panan","3302-544","1994-09-22",
                                                    1, "+351101092207", "hhh@aaaaa.tchin");
CALL register_sale (12, "2024-02-07 13:30:00");

CALL add_prod_new_shop_new_part("SALES00003", 5,
                                                    "Ivo Ilhas","200019776","Rua do Ivo","Ilhotas do Ivo","1111-000","1000-01-01",
                                                    1, "+351101092208", "iilhas@dunas.dv");
CALL register_sale (13, "2024-02-07 13:31:00");

CALL add_prod_new_shop_new_part("SALES00001", 5,
                                                    "Mizutsuki Yamatora",null,"Baia de Oda","Wan Pice","6039-777","1953-04-16",
                                                    1, "+24199504427", "WEAREONTHECRUISEEE@onepiece.oda");
CALL register_sale (14, "2024-02-07 13:32:00");

CALL add_prod_new_shop_new_part("SALES00003", 6,
                                                    "Anatoliyy Curie","559430210","Rua da Menina do Polonio","Eurehka","5077-496","2007-11-07",
                                                    1, "+24199504421", "ak47@boom.cs");
CALL register_sale (15, "2024-02-08 13:00:00");
CALL add_prod_new_shop_new_part("SALES00003", 6,
                                                    "Hector Hernandez","604449622","Ruelha de la Pas","Bolicau","3019-301","2001-09-10",
                                                    1, "+24199504422", "hhplus1@aaaaa.tchin");
CALL register_sale (16, "2024-02-08 13:01:00");


-- Inserção de carrinho que compras (Participantes existentes)
CALL add_prod_to_new_shopping_cart(1,"SALES00001", 10, 1);
CALL add_prod_to_shopping_cart(17, 1,14, 1);
CALL add_prod_to_shopping_cart(17, 1,24, 1);
CALL register_sale (17, "2023-12-19 15:33:00");

CALL add_prod_to_new_shopping_cart(3,"SALES00002", 23, 1);
CALL add_prod_to_shopping_cart(18, 3,11, 1);
CALL register_sale (18, "2023-12-19 15:34:00");

CALL add_prod_to_new_shopping_cart(5,"SALES00003", 7, 1);
CALL add_prod_to_shopping_cart(19, 5,8, 1);
CALL add_prod_to_shopping_cart(19, 5,17, 1);
CALL add_prod_to_shopping_cart(19, 5,12, 1);
CALL register_sale (19, "2023-12-15 18:40:00");

CALL add_prod_to_new_shopping_cart(7,"SALES00001", 16, 1);
CALL register_sale (20, "2024-02-02 21:40:00");

CALL add_prod_to_new_shopping_cart(9,"SALES00003", 20, 1);
CALL add_prod_to_shopping_cart(21, 9,21, 1);
CALL add_prod_to_shopping_cart(21, 9,25, 1);
CALL register_sale (21, "2024-02-05 15:10:00");
```

```
CALL add_prod_to_new_shopping_cart(11,"SALES00002", 18, 1);
CALL register_sale (22, "2024-02-05 15:12:00");

CALL add_prod_to_new_shopping_cart(13,"SALES00001", 9, 1);
CALL add_prod_to_shopping_cart(23, 13,22, 1);
CALL add_prod_to_shopping_cart(23, 13,19, 1);
CALL cancel_ongoing_sale(23);
```

4.10 Anexo 10

```
-- Sales report + Affluence Report
CREATE EVENT EOD_sales_reports
        ON SCHEDULE
                EVERY 1 DAY
                        STARTS '2023-12-05 02:00:00'
        DO SELECT SUM(S.TotalValue) AS Value, SUM(S.TotalQuantity) AS Volume
                        FROM Sale AS S
                        WHERE YEAR(S.DateOfSale) = YEAR(current_date()) AND
                                                MONTH(S.DateOfSale) = MONTH(current_date()) AND
                                                DAY(S.DateOfSale) = DAY(current_date())
                                INTO OUTFILE 'daily_sale_report.csv'
                                FIELDS TERMINATED BY ','
                                ENCLOSED BY '"'
                                LINES TERMINATED BY '\n';
```