



Universidade do Minho
Mestrado Integrado em Engenharia Informática
Licenciatura em Ciências da Computação

Unidade Curricular de Bases de Dados

Ano Lectivo de 2018/2019

Events Workbench

Catarina Machado, Gonçalo Faria, João Vilaça, José Fernandes

Janeiro, 2019

B **D**

Data de Recepção	
Responsável	
Avaliação	
Observações	

Events Workbench

Catarina Machado, Gonçalo Faria, João Vilaça, José Fernandes

Janeiro, 2019

Resumo

Este relatório foi desenvolvido no âmbito do desenvolvimento de uma Base de Dados para a empresa Events Workbench, que foca a sua atividade na área da gestão e promoção de eventos. Este software tem como objetivo facilitar e tornar mais eficiente a grande maioria do processo inerente à organização um evento. Desde a criação do próprio evento até ao controlo de participantes, passando, por exemplo, pela organização do local onde decorrerá ou da divulgação que será feita para o promover.

Ao longo deste relatório são apresentadas na íntegra todas as etapas do desenvolvimento desta base de dados até à implementação física da mesma.

Em primeiro lugar, foram estudadas todas características do meio onde este software será inserido. Foram analisadas as condições socioeconómicas em que a empresa surgiu, o seu método de trabalho e quais as suas motivações e objetivos para a implementação de uma base de dados, verificando se esta seria ou não economicamente viável.

Em seguida, através de métodos de análise e em conjunto com o cliente, foram levantados e aprovados os requisitos para o sistema que guiaram todo o processo de desenvolvimento. No fim desta fase, foi iniciada a modulação conceptual, desenvolvida com recurso à ferramenta “brModelo”, e que foi posteriormente aprovada pela Events Workbench. A partir desta desenvolveu-se em conformidade com as regras do mapeamento ER, o modelo lógico, na ferramenta “MySQL Workbench” também, ele validado pelo cliente. Por fim, a Base de Dados, gerada também ela através do “MySQL Workbench”, foi implementada fisicamente garantindo que todo o trabalho anterior convergia para uma solução correta e segura.

Concluído este processo, após a implementação da base de dados no cliente, foi dado como terminado este projeto.

Área de Aplicação: Desenho, Arquitetura, Desenvolvimento e Implementação de Sistemas de Bases de Dados.

Palavras-Chave: Bases de Dados, Bases de Dados Relacionais, Análise de Requisitos, Entidades, Atributos, Relacionamentos, Modelo Conceptual, Modelo Lógico, Modelo Físico, Normalização, Interrogações, Transações, *Triggers*, Índices, Vistas de Utilização, *Backup*, MySQL Workbench, SQL.

Índice

RESUMO.....	I
ÍNDICE.....	II
ÍNDICE DE FIGURAS.....	V
ÍNDICE DE TABELAS.....	VII
1. INTRODUÇÃO	1
1.1. CONTEXTUALIZAÇÃO	1
1.2. APRESENTAÇÃO DO CASO DE ESTUDO.....	2
1.3. FUNDAMENTAÇÃO DA IMPLEMENTAÇÃO DA BASE DE DADOS	3
1.4. ANÁLISE DA VIABILIDADE DO PROCESSO.....	3
1.5. ESTRUTURA DO RELATÓRIO.....	4
2. LEVANTAMENTO E ANÁLISE DE REQUISITOS	6
2.1. MÉTODO DE LEVANTAMENTO E DE ANÁLISE DE REQUISITOS ADOTADOS	6
2.2. REQUISITOS LEVANTADOS.....	7
2.2.1 REQUISITOS DE DESCRIÇÃO.....	7
2.2.2 REQUISITOS DE EXPLORAÇÃO	8
2.2.3 REQUISITOS DE CONTROLO	8
2.3. ANÁLISE DE REQUISITOS	9
3. MODELAÇÃO CONCEPTUAL.....	10
3.1. APRESENTAÇÃO DA ABORDAGEM DE MODELAÇÃO REALIZADA	10
3.2. IDENTIFICAÇÃO E CARACTERIZAÇÃO DAS ENTIDADES	11
3.3. IDENTIFICAÇÃO E CARACTERIZAÇÃO DOS RELACIONAMENTOS	12
3.4. IDENTIFICAÇÃO E CARACTERIZAÇÃO DA ASSOCIAÇÃO DOS ATRIBUTOS COM AS ENTIDADES E RELACIONAMENTOS	17
3.4.1 Domínio dos Atributos	20
3.4.2 Chaves Candidatas, Primárias e Alternativas	22
3.5. DETALHE OU GENERALIZAÇÃO DE ENTIDADES	24

3.6. APRESENTAÇÃO E EXPLICAÇÃO DO DIAGRAMA ER	25
3.7. VALIDAÇÃO DO MODELO DE DADOS COM O UTILIZADOR.....	26
4. MODELAÇÃO LÓGICA	32
4.1. CONSTRUÇÃO E VALIDAÇÃO DO MODELO DE DADOS LÓGICO	32
4.2. DESENHO DO MODELO LÓGICO	34
4.3. VALIDAÇÃO DO MODELO ATRAVÉS DA NORMALIZAÇÃO	34
4.4. VALIDAÇÃO DO MODELO COM AS INTERROGAÇÕES DO UTILIZADOR	35
4.5. VALIDAÇÃO DO MODELO COM AS TRANSAÇÕES ESTABELECIDAS	41
4.6. REVISÃO DO MODELO LÓGICO COM O UTILIZADOR.....	44
5. IMPLEMENTAÇÃO FÍSICA.....	45
5.1. SELEÇÃO DO SISTEMA DE GESTÃO DE BASE DE DADOS	45
5.2. TRADUÇÃO DO ESQUEMA LÓGICO PARA O SISTEMA DE GESTÃO DE BASES DE DADOS ESCOLHIDO EM SQL	45
5.3. TRADUÇÃO DAS INTERROGAÇÕES DO UTILIZADOR PARA SQL.....	45
5.4. TRADUÇÃO DAS TRANSAÇÕES ESTABELECIDAS PARA SQL	49
5.5. <i>TRIGGERS</i> SQL.....	53
5.6. ESCOLHA, DEFINIÇÃO E CARACTERIZAÇÃO DE ÍNDICES EM SQL	56
5.7. ESTIMATIVA DO ESPAÇO EM DISCO DA BASE DE DADOS E TAXA DE CRESCIMENTO ANUAL.....	57
5.8. DEFINIÇÃO E CARACTERIZAÇÃO DAS VISTAS DE UTILIZAÇÃO EM SQL	58
5.9. DEFINIÇÃO E CARACTERIZAÇÃO DOS MECANISMOS DE SEGURANÇA EM SQL.....	59
5.10. REVISÃO DO SISTEMA IMPLEMENTADO COM O UTILIZADOR	61
6. SISTEMA NOSQL: NEO4J	63
6.1. JUSTIFICAÇÃO DA UTILIZAÇÃO DE UM SISTEMA NoSQL.....	63
6.2. IDENTIFICAÇÃO E DESCRIÇÃO DOS OBJETIVOS DA BASE DE DADOS, EM TERMOS DE APLICAÇÕES E DE UTILIZADORES.....	64
6.3. IDENTIFICAÇÃO E EXPLICAÇÃO DO TIPO DE QUESTÕES (NECESSIDADES) QUE SERÃO REALIZADAS SOBRE O SISTEMA DE DADOS NoSQL.....	64
6.4. DEFINIÇÃO DA ESTRUTURA BASE PARA O SISTEMA DE DADOS NoSQL QUE SATISFAÇA OS REQUISITOS E AS QUESTÕES APRESENTADAS ANTERIORMENTE.....	65
6.5. IDENTIFICAÇÃO DOS OBJETOS DE DADOS NO SISTEMA SQL QUE SERÃO UTILIZADOS PARA ALIMENTAR O NOVO SISTEMA	66
6.6. MAPEAMENTO DO PROCESSO DE MIGRAÇÃO DE DADOS, DESCREVENDO O PROCESSO DE CONVERSÃO DOS VÁRIOS OBJETOS DE DADOS.....	67

6.7. EXPLICAÇÃO DO PROCESSO DE MIGRAÇÃO DE DADOS, EXPLICANDO DE FORMA DETALHADA AS SUAS PRINCIPAIS ETAPAS - EXTRAÇÃO, TRANSFORMAÇÃO E CARREGAMENTO	67
6.8. APRESENTAÇÃO E DESCRIÇÃO DA IMPLEMENTAÇÃO DO PROCESSO DE MIGRAÇÃO DE DADOS.....	68
6.9. APRESENTAÇÃO DA FORMA COMO AS QUESTÕES IDENTIFICADAS ANTERIORMENTE PODEM SER SATISFEITAS COM O NOVO SISTEMA, UTILIZANDO A LINGUAGEM DE INTERROGAÇÃO DO SISTEMA NoSQL....	70
7. CONCLUSÕES E TRABALHO FUTURO	72
REFERÊNCIAS.....	74
LISTA DE SIGLAS E ACRÓNIMOS.....	75
ANEXOS.....	76
I. ANEXO 1 – <i>SCRIPT DE INICIALIZAÇÃO DA BASE DE DADOS.....</i>	77
II. ANEXO 2 – <i>SCRIPT DE POVOAMENTO INICIAL</i>	83
III. ANEXO 3 - <i>SCRIPT DE IMPLEMENTAÇÃO DE INTERROGAÇÕES E FUNCIONALIDADES PARA O FUNCIONÁRIO.....</i>	93
IV. ANEXO 4 – <i>SCRIPT DE IMPLEMENTAÇÃO DE INTERROGAÇÕES E FUNCIONALIDADES PARA O ORGANIZADOR</i>	98
V. ANEXO 5 - <i>SCRIPT DE CRIAÇÃO DAS VISTAS QUE RESPONDEM A PARTE DAS INTERROGAÇÕES</i>	102
VI. ANEXO 6 - <i>SCRIPT DE FUNCIONALIDADES DE SIMPLIFICAÇÃO DA INTERAÇÃO COM A BD</i>	103

Índice de Figuras

Figura 1 - Relacionamento Plataforma - Divulgação.	12
Figura 2 - Relacionamento Organizador - Evento.	13
Figura 3 - Relacionamento Evento - Local.	14
Figura 4 - Relacionamento Divulgação - Evento.	15
Figura 5 - Relacionamento Evento - Participante - Divulgação.	16
Figura 6 - Modelo Conceptual.	25
Figura 7 - Modelo Conceptual Final.	26
Figura 8 - Atributos de Organizador.	26
Figura 9 - Atributos de Evento.	27
Figura 10 - Atributos de Local.	28
Figura 11 - Atributos de Divulgação.	30
Figura 12 - Atributos de Plataforma.	31
Figura 13 - Atributos de Participante.	31
Figura 14 - Modelo Lógico.	34
Figura 15 - Árvore representativa da Interrogação n.º 1.	35
Figura 16 - Árvore representativa da Interrogação n.º 2.	36
Figura 17 - Árvore representativa da Interrogação n.º 3.	37
Figura 18 - Árvore representativa da Interrogação n.º 4.	38
Figura 19 - Árvore representativa da Interrogação n.º 5.	38
Figura 20 - Árvore representativa da Interrogação n.º 6.	39
Figura 21 - Árvore representativa da Interrogação n.º 7.	40
Figura 22 - Árvore representativa da Interrogação n.º 8.	40
Figura 23 - Entidade.	41
Figura 24 - Local.	41
Figura 25 - Organizador.	42
Figura 26 - Participante.	42
Figura 27 - Evento - Local.	43
Figura 28 - Evento - Participante - Divulgação.	43
Figura 29 - Imagem do código obtido para a query n.º 1.	46
Figura 30 - Imagem do código obtido para a query n.º 2.	47
Figura 31 - Imagem do código obtido para a query n.º 3.	48

Figura 32 - Imagem do código obtido para a <i>query</i> n. ^o 4.	48
Figura 33 - Imagem do código obtido para a transação n. ^o 1.	49
Figura 34 - Imagem do código obtido para a transação n. ^o 2.	50
Figura 35 - Imagem do código obtido para a transação n. ^o 3.	50
Figura 36 - Imagem do código obtido para a transação n. ^o 4.	51
Figura 37 - Imagem do código obtido para a transação n. ^o 5.	52
Figura 38 - Imagem do código obtido para o <i>trigger</i> n. ^o 1.	53
Figura 39 - Imagem do código obtido para o <i>trigger</i> n. ^o 2.	54
Figura 40 - Imagem do código obtido para o <i>trigger</i> n. ^o 3.	55
Figura 41 - Imagem do código obtido índices participante e divulgação.	56
Figura 42 - Imagem do código obtido índice idx_tipo.	56
Figura 43 - Imagem do código obtido índice idx_evento.	56
Figura 44 - Vista de Participante.	58
Figura 45 - Vista de Geral de Eventos.	59
Figura 46 - Criação e atribuição de permissões de funcionário.	60
Figura 47 - Criação e atribuição de permissões de organizador.	60
Figura 48 - Execução do comando <i>mysqldump</i> .	61
Figura 49 - Execução do comando <i>mysqldump</i> .	61
Figura 50 - Estrutura Neo4j.	65
Figura 51 - Ilustração da base de dados NoSQL criada em Neo4j.	66
Figura 52 - Diagrama de Classes Neo4j.	69

Índice de Tabelas

Tabela 1 - Caracterização de todos os atributos existentes.	18
Tabela 2 - Caracterização dos atributos de Evento.	20
Tabela 3 - Caracterização dos atributos de Organizador.	20
Tabela 4 - Caracterização dos atributos de Participante.	21
Tabela 5 - Caracterização dos atributos de Local.	21
Tabela 6 - Caracterização dos atributos de Divulgação.	22
Tabela 7 - Caracterização dos atributos de Plataforma.	22
Tabela 8 - Caracterização dos atributos de Relacionamento Evento - Participante - Divulgação.	22
Tabela 9 - Espaço Esperado de uma Entrada de Cada Tabela.	57
Tabela 10 - Permissões de Funcionários e Organizadores.	59

1. Introdução

1.1. Contextualização

A arte e a cultura foram indiscutivelmente duas das áreas mais prejudicadas pela grave crise socioeconómica que, de 2010 até 2014, abateu Portugal. No entanto, dados da Eurostat e do INE mostram que, mesmo antes da crise, a população portuguesa foi a que menos gastou na área. Em 2006, na rubrica “Lazer, distracção e cultura”, as famílias portuguesas apenas gastaram 7,5% do seu orçamento familiar, um valor significativamente inferior à média dos países da União Europeia, 9,2%, e, em particular, à Dinamarca, o primeiro classificado, 12,3%. Uma rápida análise destes resultados deixa claro que o baixo interesse nesta área em Portugal não é meramente causada por fatores conjunturais, por exemplo o baixo rendimento per capita durante estes anos, mas sim por fatores estruturais, nomeadamente o estímulo e investimento cultural.

Hoje em dia, com a constante melhoria das condições económicas do país, é importante garantir que são aproveitadas todas as oportunidades de fomentar o interesse do público nos eventos culturais que diariamente são organizados e combater esta falta de atratividade que o povo lusitano tem pela cultura. Este aproveitamento tem de ser feito recorrendo a uma divulgação criteriosa dos mesmos, e não de forma arbitrária, para garantir a eficiente utilização dos recursos dedicados a este propósito.

É então essencial alcançar as pessoas certas, nas circunstâncias certas, com a mensagem certa. Com este objetivo em mente, surge então a Events Workbench, que se propõe a assumir um papel ativo na luta pela difusão cultural, permitindo que os organizadores dos eventos possam focar-se a 100% na significância dos momentos que criam. Esta empresa assume ainda o compromisso de estudar e trabalhar as características sociodemográficas da população para alcançar com sucesso todas as faixas etárias. Para além disso, é dada redobrada atenção obviamente para as gerações mais novas que terão um dia a capacidade de ser os verdadeiros impulsionadores de uma viragem de mentalidade em relação à importância que atribuímos a estes eventos.

Assim sendo, a Events Workbench nasceu com o objetivo de permitir que qualquer organização seja capaz de ver resolvidas as suas necessidades dentro dos serviços que esta empresa presta. É obviamente uma prioridade empresarial tentar automatizar todos os

processos, ser capaz de trabalhar em vários projetos em simultâneo sem conflitos e possibilitar os clientes a focar naquilo que verdadeiramente importa, dar e receber cultura.

Para atingir com sucesso todos estes objetivos a Events Workbench baseia-se em 3 pilares fundamentais: o “**apoio a eventos**”, o “**marketing preciso e eficiente**” e a “**gestão de participantes**”. Só a uma estrutura coesa e pensada em torno destes pilares, que garanta e facilite a interação entre eles, será capaz de prosperar.

1.2. Apresentação do Caso de Estudo

Sediada em Braga, a Events Workbench, fundada por quatro estudantes do curso de Engenharia Informática da Universidade do Minho, proporciona a organizadores de eventos serviços de Marketing e Publicidade e Gestão de Participantes.

Desde o momento em que um ou mais organizadores entram em contacto com a Events Workbench, esta compromete-se a apoiá-los em todo o processo da gestão de um evento, desde de a sua conceção até ao momento da análise de como o mesmo decorreu. A partir desse momento, a Events Workbench é responsável por encontrar e sugerir aos organizadores o melhor local para o evento, com base na análise do sucesso de eventos anteriores, e por acertar todos os detalhes, em conjunto com os mesmos, para garantir que os requisitos que os organizadores impõem para o evento são satisfeitos. É também a empresa a total responsável por gerir todas os meios de divulgação do evento, quer sejam eles, físicos, na televisão, rádio, online ou outros. Por fim, é a Events Workbench que coordena todo o processo de gestão de participantes, quer eles se registem através de uma plataforma da empresa, quer se registem através de uma plataforma de outrem.

É ainda de destacar o fator notável distingue esta empresa da sua concorrência, uma vez que, usando análise de dados, em linha com o estudo da ciência comportamental, esforça-se por compreender todos os indivíduos que constituem a audiência dos organizadores, com o intuito de ser o mais influente possível a fim de atrair o público. Esta conexão é estabelecida de modo a que a desejada audiência frequente e participe na divulgação dos eventos e organizadores aliados a esta empresa.

Esta metodologia que se pretende adotar é, agora mais que nunca, indispensável em estratégias de marketing de sucesso. A acrescida superlotação publicitária e o bombardeamento constante de informação dificultam que a divulgação seja bem sucedida. Para superar este desafio, apenas um uso visado de ferramentas de identificação de perfil comportamental, através permite a campanhas multiplataforma são capazes de atingir, de uma forma eficaz, o desejado segmento sociodemográfico.

1.3. Fundamentação da Implementação da Base de Dados

Embora a Events Workbench tenha começado a sua atividade com outro sistema de armazenamento de dados, com folhas de cálculo, este é muito rudimentar e representa vários problemas e limitações. Uma vez que a empresa pretende ser capaz de manipular e estudar os dados que vai recolhendo para melhor servir os seus clientes, é fundamental organizar e tratar estes dados de forma objetiva e eficiente. Para responder a esta necessidade, a melhor alternativa é a implementação de uma Base de Dados que facilite o armazenamento de toda a informação.

A conceção da base de dados deve ser considerada por forma a cumprir os requisitos da infraestrutura de sistema de informação que esta empresa já contém, para que a transição para o novo sistema de armazenamento e exploração de dados seja o menos dispendiosa possível. O presente sistema, permite aos funcionários da empresa armazenar a informação relativa aos clientes, eventos, campanhas de divulgação e participantes.

Para além da criação da Base de Dados para uso próprio, por forma a aumentar a transparência com os seus clientes, a Events Workbench, decidiu que a melhor decisão a tomar seria a de, já que vai expandir as suas capacidades, desenvolver uma plataforma que permita aos seus clientes consultar os participantes dos seus eventos, assim como, monitorizar em tempo real o impacto e eficácia das diferentes campanhas respetivas a estes.

Adicionalmente, para que os seus clientes possam também compreender qual foi a apreciação dos participantes dos seus eventos, a Events Workbench, pretende que, no futuro, seja possível a integração em múltiplas outras plataformas de mecanismos que permitem adicionar à sua base de dados a avaliação destes.

1.4. Análise da Viabilidade do Processo

Este projeto consiste na implementação de uma Base de Dados Relacional que possibilita estudar e relacionar os dados que a empresa possui de modo a compreender e melhorar o processo de gestão de eventos. Possuir uma Base de Dados Relacional é essencial pois permite a sua manipulação e a extração de conhecimento de uma forma metodológica, podendo assim assegurar a consistência dos dados uma vez que, em larga medida, a necessidade de redundância desaparece.

O investimento na criação de uma Base de Dados Relacional **diminuirá não só as despesas a longo prazo** como **aumentará a qualidade de serviço** que a Events Workbench oferece. Isto porque os custos logísticos associados ao uso de um sistema de ficheiros para a gestão dos dados, o presente sistema que a Events Workbench tem, são, sem dúvida, um fator limitativo no crescimento da empresa. O sistema de ficheiro atual, sendo uma abordagem descentralizada à gestão de dados, é dispendiosa de tempo, obriga a uma desnecessária repetição de tarefas de forma manual e promove a duplicação de dados, é, por exemplo,

comum ter que reescrever todos os dados do local onde o evento acontece. A redundância é dispendiosa e leva a perda da integridade dos dados. No novo sistema que propomos desenvolver, será assegurado que este problema não existirá.

Como toda a infraestrutura do sistema de informação foi feita usando uma arquitetura de desenho 3-tier, incorporar a nova base de dados neste não será extremamente dispendioso, principalmente quando comparado à necessidade de ter que refazer de novo toda esta infraestrutura, o caso se este tipo de desenho não fosse aplicado.

A nova base de dados, como será implementada num Sistema de Gestão de Base de Dados Relacional baseado no uso de transações, permitirá efetuar inúmeras sequências de operações em simultâneo sem que a consistência e integridade dos dados seja sacrificada.

Em agregado estas características da solução que nos comprometemos desenvolver contribuirão para que os problemas operacionais e administrativos adjacentes ao sistema de gestão de dados que a Events Workbench contém desapareçam. A implementação desta solução permitirá à Events Workbench estar a par da concorrência, a nível dos sistemas de informação e de gestão de dados, a fim de continuar o caminho de sucesso que tem apresentado.

1.5. Estrutura do Relatório

O presente relatório é composto por seis capítulos:

Na **Definição de Sistema**, elaboramos sobre a natureza do problema que estamos a tentar solucionar assim como o contexto em que se encontra inserido.

De seguida, em **Levantamento e Análise de Requisitos**, é apresentado o método de levantamento dos requisitos. Adicionalmente, são expostos os requisitos de descrição, exploração e controlo obtidos, assim como a análise dos mesmos.

Na **Modelação Conceptual**, é iniciada a apresentação da arquitetura da nossa solução. São descritas as entidades do problema, os respetivos atributos e os relacionamentos entre as entidades.

Os atributos são analisados e devidamente catalogados, podendo ser multivvalorados ou não, compostos ou simples, nulos ou não nulos, derivados ou não derivados. É também identificado o tipo de dados mais adequado para cada um dos atributos e o respetivo domínio de valores. As chaves candidatas, primárias e secundárias são também apresentadas. Finalmente, o modelo obtido é confrontado com uma generalização/especificação e, por último, é feita uma validação do modelo de dados resultante com o utilizador.

Em **Modelação Lógica**, numa primeira fase, é construído o modelo de dados lógico e posteriormente é validado, tendo em consideração as regras de derivação. São ainda identificadas as ocorrências de cada um dos tipos de relacionamentos presentes no modelo lógico, assim como as chaves primárias de cada um deles. Em seguida, o modelo é validado através da normalização.

Posteriormente, o modelo é também validado com as interrogações do utilizador e as transações estabelecidas e, por fim, o modelo lógico final é revisto com o utilizador.

Seguidamente, em **Implementação Física**, é apresentado o modelo de sistema de gestão de bases de dados utilizado e o modelo lógico devidamente convertido para esse sistema de gestão de bases de dados.

As interrogações outrora estabelecidas com o utilizador são agora traduzidas para SQL, assim como as transações. São implementados os *triggers* fundamentais para o programa e ainda definidos e caracterizados novos índices.

Estimou-se o espaço em disco da base de dados necessário e a taxa de crescimento anual esperada.

Em seguida, definiu-se e caracterizou-se ainda vistas de utilização e mecanismos de segurança em SQL. E, por último, o sistema implementado é revisto com o utilizador.

Numa última fase, na **Conclusão e Trabalho Futuro**, é feita uma retrospectiva do trabalho concebido e expostas as vantagens da utilização da nova base de dados. É ainda mencionado algumas funcionalidades que poderão ser implementadas no futuro, com o intuito de melhorar ainda mais a base de dados da Events Workbench.

2. Levantamento e Análise de Requisitos

2.1. Método de Levantamento e de Análise de Requisitos Adotados

Quanto à abordagem no levantamento dos requisitos, decidimos que a mais adequada a seguir seria a de agrupar todos os que dizem respeito a cada perfil de utilização de maneira a formar um único conjunto correspondente à totalidade da base de dados. Para isso recorremos a várias técnicas de *fact-finding* para tentar perceber o comportamento do sistema atual com o intuito de garantir que o resultado final satisfaz as necessidades da empresa.

As técnicas a que recorremos foram:

- **Entrevistas:** Foram feitas entrevistas a colaboradores da Events Workbench que trabalham em diferentes vertentes: distribuição, publicidade, comunicação com os organizadores e comunicação com os locais, de forma a perceber o domínio do problema, as diferentes vistas/necessidades que estas pessoas têm sobre o novo sistema e como é que estas interagem com o sistema atual.
- **Observação:** Durante alguns dias foi observado o funcionamento da empresa. Durante este intervalo de tempo foi analisado o modo como os funcionários interagiam com o sistema atual, as funcionalidades e as limitações deste e foi possível perceber o serviço prestado, podendo assim perceber como administrar e explorar o mesmo.
- **Análise do Sistema de Informação Atual:** Com a análise do sistema de informação utilizado, foi possível recolher informações descritivas do sistema, como quais os tipos de dados armazenados, a maneira como os vários tipos de dados se encontram ligados e os aspetos mais importantes a considerar para a migração de dados já existentes para o novo sistema.
- **Personas:** Foram criados alguns clientes falsos de forma a interagir com o sistema atual para ser possível explorar os vários comportamentos destes para entender como é que o sistema poderá responder a esses comportamentos e necessidades.

Depois de levantados os requisitos, os mesmos foram categorizados em requisitos de descrição, controlo e de exploração. Posteriormente, foram também apresentados ao nosso

empregador, com quem foi efetuada uma análise dos vários requisitos por forma a garantir que o problema apresentado é o que este pretendia ver solucionado.

2.2. Requisitos Levantados

2.2.1 Requisitos de Descrição

1. Quando um organizador se regista, com um funcionário da Events Workbench, ele terá de indicar o nome, a sua descrição, o seu número de telemóvel, email e o endereço;
2. Um ou mais organizadores poderão criar e editar, junto ao funcionário, eventos;
3. Aquando do registo de um evento deverão ser indicados o respetivo nome, classificação, tipo, data, duração, período de registo de participantes, preço base e uma breve descrição;
4. O evento terá de ser necessariamente de um dos seguintes tipos: Discussão, informação e formação; Religiosos e comunitários; Académicos, Políticos e de protocolo; Culturais, de lazer, desportivos e musicais; Culinária; Empresariais ou Outros;
5. Cada evento deverá estar associado obrigatoriamente ao local onde decorrerá, para que o organizador possa controlar a organização do evento e acertar pormenores com os responsáveis pelo local onde ocorrerá o evento;
6. Aquando do registo de um local deverá ser fornecida o seu nome, descrição, tipo, lotação, endereço, email e número de telemóvel;
7. O local terá de ser necessariamente de um dos seguintes tipos: Bares, Casas noturnas, Restaurantes, Hotelaria, Casa de espetáculos, Centros Culturais, Multiusos, Quintas ou Outros;
8. O evento ao ser criado deverá ainda conter informação relativa ao número máximo de participantes. Esse número nunca poderá ultrapassar a lotação do local;
9. Em eventos com vários dias deverá ser criado um único evento para cada um dos dias;
10. A cada evento poderão ser registados os seus participantes;
11. Para cada evento não pode haver mais do que um participante igual, ou seja, não pode haver participantes repetidos na lista de participantes do evento;
12. Cada participante poderá deixar uma classificação a cada evento que participou;
13. A Classificação de um evento é a média de classificações dadas pelos clientes;
14. Ao participar num evento um participante tem associado um lugar, caso este o suporte, um preço que este paga pode ser diferente do preço do evento e à sua participação pode estar associado um estado. Esse estado terá de ser um dos seguintes valores: Válido, Cancelado ou Reservado;
15. Um funcionário poderá criar uma divulgação com a respetiva associação a um evento;
16. Uma divulgação apenas pode ser partilhada quando o organizador a validar;

17. Quando da divulgação deverão ser registadas a data de início e de fim da divulgação, o custo da divulgação, o tipo e a plataforma;
18. A divulgação terá de ser necessariamente de um dos seguintes tipos: Audiovisual, Áudio, Publicação, Cartaz, Email, Brochuras, Apresentações, Promoções, Panfletos ou Outros;
19. O participante poderá indicar uma divulgação que tenha influenciado a sua participação num evento;
20. A plataforma deverá ser definida por um nome;
21. Quando se regista um participante terão de ser fornecidos o seu nome, género, NIF, data de nascimento, contactos, email e número de telemóvel.

2.2.2 Requisitos de Exploração

1. O organizador poderá ser capaz de consultar o nome, o género, os contactos e a data de nascimento dos respetivos participantes de um dado evento por este organizado;
2. O organizador poderá consultar quais as plataformas usadas para a divulgação do evento;
3. O organizador deverá ser capaz de consultar o nome, a descrição, o endereço e o tipo do local onde se realizará qualquer um dos eventos por ele organizado;
4. Um organizador poderá consultar quais os eventos por este organizado;
5. Um funcionário da empresa poderá consultar todos os participantes de um evento assim como todas as suas informações;
6. Um funcionário da empresa deverá poder filtrar eventos por data e local;
7. Um funcionário da empresa poderá consultar, para um número indicado de participantes, os que mais dinheiro gastam em eventos no geral e por tipo de eventos ou por organizador;
8. Para cada tipo de evento, um funcionário, deverá poder verificar qual o tipo de divulgação mais eficaz. Quer em quantidade de participantes registados, quer em termos de participante/custo de divulgação;
9. O organizador e um funcionário da empresa poderão consultar para cada divulgação quantos os participantes indicaram que esta os influenciou diretamente no ingresso do correspondente evento (no caso do organizador apenas os seus eventos).

2.2.3 Requisitos de Controlo

1. O organizador apenas pode consultar os aspetos do sistema que a este estão direta ou indiretamente associadas;

2. Um funcionário da empresa poderá adicionar e editar novos eventos, divulgações, locais, participantes, organizadores, plataformas, assim como, consultar informações referentes a estes;
3. Um funcionário de uma empresa poderá também associar o Organizador a Eventos e Participantes a Eventos informando a Divulgação;
4. Um funcionário da empresa não poderá alterar propriedades do esquema da base de dados.

2.3. Análise de Requisitos

Após o levantamento e qualificação dos requisitos foi necessária uma análise em agregado dos mesmos. Este processo incluiu várias discussões, não só entre os elementos da nossa equipa, como também com as partes interessadas (*stakeholders*), para que desta forma fossem resolvidos os conflitos entre os requisitos correspondentes às diferentes entidades. Centralizadas na satisfação das expectativas da Events Workbench, nestas discussões foi abordado qual o grau de importância a dar aos diferentes níveis de detalhe em determinados aspectos do sistema a desenvolver.

Um dos aspectos que mais foi discutido foi a existência ou não do conceito de bilhete/entrada no sistema. Apesar deste conceito não representar algo físico na empresa a correspondência entre vários organizadores e vários participantes é tradicionalmente manifestada nesta forma. Eventualmente, a equipa chegou à conclusão que este conceito era desnecessário.

Esta fase serviu principalmente para alinhar as diferenças vistas das partes interessadas, assim como dos elementos da equipa, de forma a mitigar complicações futuras relativas a eventuais diferenças nas interpretações dos requisitos do sistema de armazenamento de dados a desenvolver.

3. Modelação Conceptual

3.1. Apresentação da Abordagem de Modelação Realizada

Depois da recolha e da análise dos requisitos do sistema chegou a altura de começar a fase do planeamento do *design* da Base de Dados.

Com o intuito de reduzir a margem para erros e de se estabelecer um entendimento comum, ou seja, que a base de dados que será construída vá de encontro às necessidades dos seus utilizadores finais, sentimos a necessidade de utilizar um modelo, não muito técnico e fácil de ser compreendido, em que possamos comunicar com o utilizador sem ambiguidades. O modelo perfeito para esse efeito é o **Diagrama ER**.

O Diagrama ER utiliza uma abordagem *top-down* uma vez que começamos por identificar os dados relevantes do problema, as entidades, e os relacionamentos entre esses dados e, posteriormente, começamos a adicionar mais informação ao modelo com os atributos, quer nas entidades, quer nos relacionamentos entre as mesmas, e respetivos domínios. Por último, incluímos também a cardinalidade desses relacionamentos para que possam ser representadas algumas restrições.

O modelo conceptual tem como objetivo representar os dados presentes nos requisitos do utilizador. Ao construirmos este modelo estaremos repetidamente a fazer questões sobre entidades, relacionamentos e atributos, onde as respostas deverão estar corretamente documentadas nos requisitos.

Outro aspeto bastante positivo da utilização desta metodologia é o facto de ser completamente independente dos futuros detalhes de implementação.

Ao longo da construção deste modelo conceptual é fundamental que o mesmo seja constantemente testado e validado com o utilizador e com os seus requisitos, de modo a que fique assegurado que a futura base de dados seja coerente e que efetivamente resolva um problema.

3.2. Identificação e Caracterização das Entidades

Atendendo aos requisitos levantados foram identificadas as entidades apresentadas em seguida.

- **Evento**

O evento é o acontecimento que vai ocorrer e que se pretende divulgar e é representado através do seu **id**, **nome**, **descrição**, **tipo** (discussão, informação e formação; religiosos e comunitários; académicos; políticos e de protocolo; culturais, de lazer, desportivos e musicais; culinária; empresariais), a **data** em que vai ocorrer e a respetiva **duração** e o **preço** base para a entrada no evento (não é o definitivo uma vez que está sujeito a, por exemplo, promoções). Esta entidade possui ainda a informação do **número máximo de participantes**, da **data de início** e a **data final** do registo de participantes e a **classificação** do evento, que será a média de todas as classificações atribuídas pelos participantes ao determinado evento (de 0 até 10).

- **Organizador**

A entidade Organizador é a empresa, companhia, estabelecimento ou até mesmo a pessoa individual que pretende ver o seu evento divulgado. É caracterizada pelo seu **id**, **nome**, **descrição**, **endereço**, **email** e **número de telemóvel**.

- **Participante**

A entidade Participante representa a pessoa que participa no evento. Cada um dos participantes possui um **id**, **nome**, **género** (feminino ou masculino), **NIF** (para possíveis efeitos de faturação), **endereço** (morada), **email**, **número de telemóvel** e a **data de nascimento**.

- **Local**

A entidade Local representa o local e respetivas condições em que o evento vai ocorrer. Um local possui um **id**, **nome**, **endereço** (morada), **lotação**, **descrição** (alguma informação útil que seja importante salvaguardar), **tipo** (poderá ser bares, casas noturnas, restaurantes, hotelaria, casa de espetáculos, centros culturais, multiusos, quintas ou outros) e o respetivo **número de telemóvel** e **email**.

- **Divulgação**

A entidade divulgação refere-se à informação de uma determinada publicidade ao evento. Esta entidade é caracterizada pelo seu **id**, **conteúdo**, **tipo** (audiovisual, áudio, publicação, cartaz, email, brochuras, apresentações, promoções, panfletos ou outros), **custo** (o custo da divulgação), **validade** (o organizador tem que aprovar a divulgação logo poderá estar válida, pendente ou inválida) e a **data de início** e **data final** da publicidade.

- **Plataforma**

A entidade plataforma é a rede social ou meio onde a divulgação será realizada/publicada, que é representada através do seu **id** e do seu **nome**.

3.3. Identificação e Caracterização dos Relacionamentos

- **Relacionamento Plataforma – Divulgação**

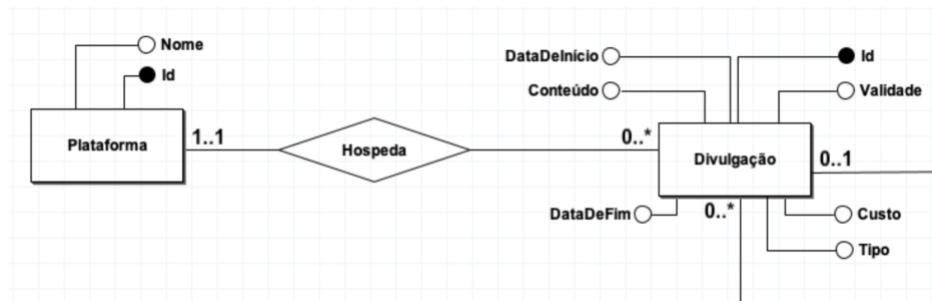


Figura 1 - Relacionamento Plataforma - Divulgação.

Relacionamento: Plataforma Hospeda Divulgação.

Descrição: Com o intuito de fazer publicidade relativa a um evento, são utilizadas as mais variadas plataformas/meios de comunicação para partilhar anúncios. Esta informação tem que ser armazenada para podermos estar constantemente a averiguar quais são as melhores estratégias de publicidade, ou seja, aquelas que têm um maior impacto no número de participantes.

Cardinalidade: Plataforma (1..1) - Divulgação (0..*).

Uma plataforma poderá hospedar várias divulgações (incluindo zero divulgações, caso não seja selecionada para o efeito), mas uma determinada divulgação apenas está presente numa só plataforma, isto porque cada divulgação é considerada única visto que poderá ter, por exemplo, um diferente custo consoante a plataforma que se insere.

Atributos: Este relacionamento não possui atributos.

- **Relacionamento Organizador – Evento**

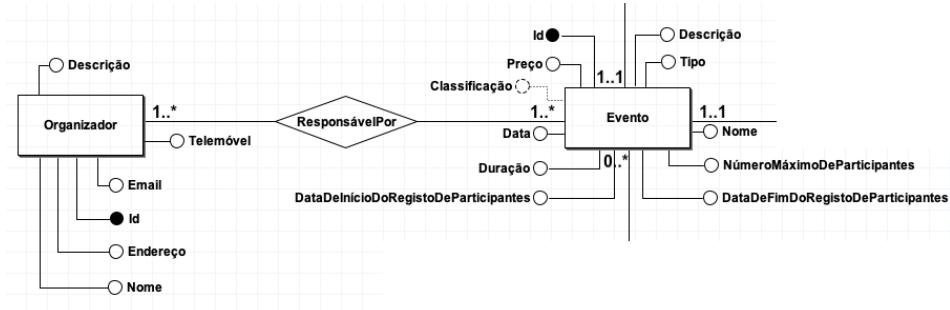


Figura 2 - Relacionamento Organizador - Evento.

Relacionamento: Organizador Responsável Por Evento

Descrição: As organizações recorrem à Events Workbench para darem o devido suporte aos seus eventos. Assim, é importante preservar esta informação para o evento ser corretamente publicitado e para o histórico de eventos realizados por x organizador poder ser consultado e analisado.

Cardinalidade: Organizador (1..*) - Evento (1..*).

O organizador é responsável por um ou mais eventos, visto que a organização tem que estar a realizar pelo menos um para recorrer à Events Workbench e que essa mesma organização poderá estar a efetivar ao mesmo tempo várias iniciativas, com o apoio da Events Workbench.

Um evento poderá ter como responsável uma ou mais organizações, uma vez que é certo que todos os eventos têm que ter pelo menos uma empresa associada e que existem eventos que têm como anfitrião várias empresas em simultâneo.

Atributos: Este relacionamento não possui atributos.

- **Relacionamento Evento – Local**

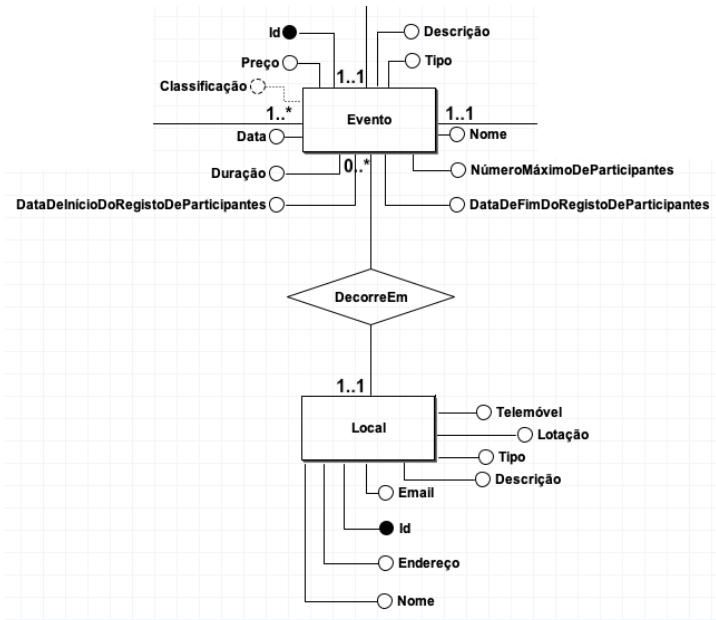


Figura 3 - Relacionamento Evento - Local.

Relacionamento: Evento Decorre Em Local

Descrição: Todos os eventos têm que ocorrer num local para poderem efetivamente acontecer.

Cardinalidade: Evento (0..*) - Local (1..1).

Um evento ocorre obrigatoriamente em um e um só local. Caso o mesmo evento se esteja a realizar em diferentes lugares ao mesmo tempo (data e hora) e com igual conteúdo, para a Events Workbench cada uma dessas ocorrências é considerada uma nova edição do evento. São considerados diferentes eventos visto que para efeitos logísticos e de divulgação esta diferença de local é inequivocamente relevante.

Num local podem estar a acontecer zero, um ou vários eventos em simultâneo.

Atributos: Este relacionamento não possui atributos.

- Relacionamento Divulgação - Evento

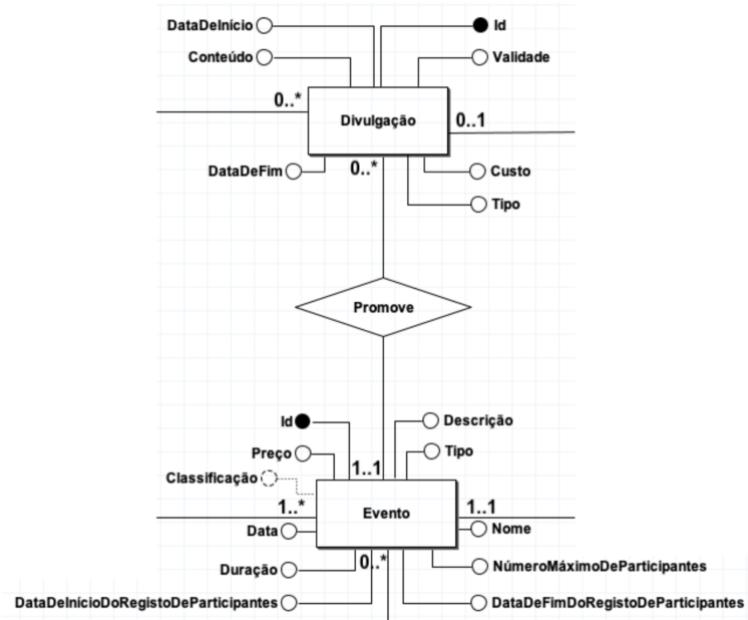


Figura 4 - Relacionamento Divulgação - Evento.

Relacionamento: Divulgação Promove Evento

Descrição: Os eventos podem ser promovidos através de divulgações

Cardinalidade: Divulgação (**0..***) - Evento (**1..1**).

Uma divulgação diz respeito a somente um evento.

Um evento pode ser promovido por zero, uma ou mais divulgações.

Atributos: Este relacionamento não possui atributos.

- Relacionamento Evento – Participante – Divulgação

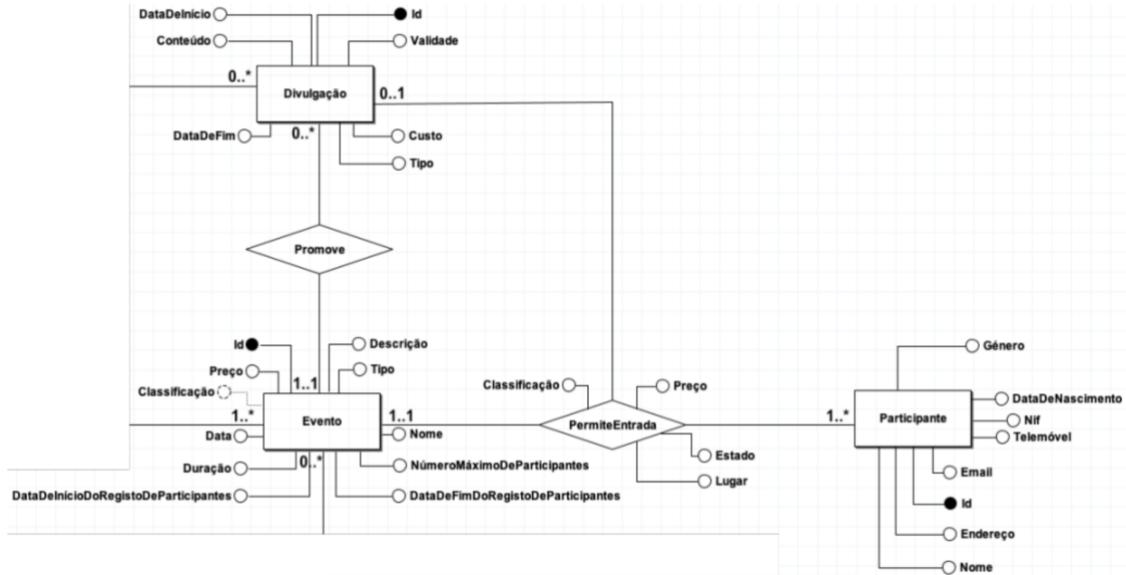


Figura 5 - Relacionamento Evento - Participante - Divulgação.

Relacionamento: Entrada de um participante num evento com a ajuda de uma divulgação.

Descrição: Este relacionamento ternário pode ser batizado como um Bilhete, mas não existe a necessidade de criar efetivamente esta entidade uma vez que os requisitos não o justificam. Juntando então estas três entidades, nomeadamente o Evento, o Participante e a Divulgação, vemos de forma clara a entrada de um participante no evento através de uma divulgação, a divulgação que desencadeou a inscrição do participante.

Cardinalidade: Evento (1..1) - Participante (1,n) - Divulgação (0,1)

A um evento e a um participante está associada uma divulgação. O participante poder ter sido persuadido com várias publicidades relativas a um evento mas apenas uma delas culmina com a inscrição do participante no evento.

De igual forma, a um participante e a uma divulgação diz respeito um, e um só, evento. Isto também está relacionado com o facto de cada divulgação publicitar apenas um evento, logo, a um participante e uma divulgação só poderá estar relacionado um evento específico.

Um evento e uma divulgação dão origem a um ou vários participantes. Influenciadas por uma divulgação relativamente a um evento, várias pessoas podem efetivar a inscrição no mesmo.

Atributos: Os atributos que este relacionamento possui são os seguintes: **classificação** (cada participante tem a oportunidade de classificar o evento, entre 0 a 10, para posteriores fins estatísticos), **estado** (o bilhete que dá acesso ao evento poderá estar válido, cancelado ou reservado), **preço** (o preço que o participante efetivamente pagou pela entrada no evento) e o **lugar** (informação do lugar reservado para o cliente, por exemplo, se for uma

sala de cinema ou teatro é necessário mencionar em que lugar o participante se deverá sentar).

3.4. Identificação e Caracterização da Associação dos Atributos com as Entidades e Relacionamentos

Entidade	Atributos	Tipo de Dados	Nulo	Composto	Multivalor	Derivado	Candidato
Evento	Id	INT	Não	Não	Não	Não	Sim
	Nome	VARCHAR(64)	Não	Não	Não	Não	Não
	Descrição	TEXT(512)	Não	Não	Não	Não	Não
	Tipo	ENUM(8)	Não	Não	Não	Não	Não
	Data	DATETIME	Não	Não	Não	Não	Não
	Duração	TIME	Não	Não	Não	Não	Não
	Preço	DECIMAL(7,2)	Não	Não	Não	Não	Não
	Número Máximo de Participantes	INT	Não	Não	Não	Não	Não
	Data Início Registo Participantes	DATETIME	Não	Não	Não	Não	Não
	Data Fim Registo Participantes	DATETIME	Não	Não	Não	Não	Não
	Classificação	DECIMAL(4,2)	Sim	Não	Não	Sim	Não
Organizador	Id	INT	Não	Não	Não	Não	Sim
	Nome	VARCHAR(64)	Não	Não	Não	Não	Não
	Descrição	TEXT(512)	Não	Não	Não	Não	Não
	Endereço	VARCHAR(128)	Não	Não	Não	Não	Não
	Email	VARCHAR(45)	Não	Não	Não	Não	Sim
	Número de Telemóvel	VARCHAR(15)	Não	Não	Não	Não	Sim
Participante	Id	INT	Não	Não	Não	Não	Sim
	Nome	VARCHAR(64)	Não	Não	Não	Não	Não
	Género	ENUM(2)	Não	Não	Não	Não	Não
	NIF	INT	Sim	Não	Não	Não	Sim
	Endereço	VARCHAR(128)	Não	Não	Não	Não	Não
	Email	VARCHAR(45)	Não	Não	Não	Não	Sim
	Número de Telemóvel	VARCHAR(15)	Não	Não	Não	Não	Sim
Local	Data de Nascimento	DATE	Não	Não	Não	Não	Não
	Id	INT	Não	Não	Não	Não	Sim
	Nome	VARCHAR(64)	Não	Não	Não	Não	Não
	Endereço	VARCHAR(128)	Não	Não	Não	Não	Não
	Lotação	INT	Não	Não	Não	Não	Não
	Descrição	TEXT(256)	Não	Não	Não	Não	Não

	Tipo	ENUM(9)	Não	Não	Não	Não	Não
	Email	VARCHAR(45)	Não	Não	Não	Não	Não
	Número de Telemóvel	VARCHAR(15)	Não	Não	Não	Não	Não
Divulgação	Id	INT	Não	Não	Não	Não	Sim
	Conteúdo	TEXT(1024)	Não	Não	Não	Não	Não
	Tipo	ENUM(10)	Não	Não	Não	Não	Não
	Custo	DECIMAL(7,2)	Não	Não	Não	Não	Não
	Validade	ENUM(3)	Não	Não	Não	Não	Não
	Data Início	DATETIME	Não	Não	Não	Não	Não
	Data Fim	DATETIME	Não	Não	Não	Não	Não
Plataforma	Id	INT	Não	Não	Não	Não	Sim
	Nome	VARCHAR(64)	Não	Não	Não	Não	Não
Relacionamento Evento - Participante - Divulgação	Classificação	INT	Sim	Não	Não	Não	Não
	Estado	ENUM(3)	Não	Não	Não	Não	Não
	Preço	DECIMAL(7,2)	Não	Não	Não	Não	Não
	Lugar	VARCHAR(32)	Não	Não	Não	Não	Não

Tabela 1 - Caracterização de todos os atributos existentes.

Os atributos guardam valores que descrevem a ocorrência das entidades e representam a parte principal dos dados guardados na base de dados.

A maioria dos atributos identificados são simples, com um único valor (*single-valued*), não são derivados e não podem ser nulos. Porém, existem algumas exceções, que passaremos a explicar em seguida.

• Evento

Todos os atributos desta entidade (id, nome, descrição, tipo, data, duração, preço, número máximo de participantes, data de início do registo dos participantes, data do final do registo dos participantes e classificação) são **simples** e possuem **um único valor**.

A classificação poderá ser **nula** caso nenhum dos participantes classifique o evento.

Caso não exista explicitamente um período de inscrição definido, ou seja, uma data de início para o registo dos participantes e uma data final para o registo dos participantes, a data de início é a data do registo do evento e a data final é a data do próprio evento.

O único atributo **derivado** é a classificação, uma vez que é uma consequência do atributo classificação do Relacionamento Evento - Participante - Divulgação.

A Classificação do Evento é calculada da seguinte forma:

$$\text{classificação} = \frac{\text{soma das classificações não nulas do Relacionamento EPD}}{\text{número de classificações não nulas do Relacionamento EPD}}$$

- **Organizador**

Nenhum dos atributos do Organizador (id, nome, descrição, endereço, email e número de telemóvel) é **derivado**, uma vez que nenhum deles pode ser deduzido através de outros atributos, e nenhum deles pode ser **nulo** visto que todas essas informações são essenciais para a identificação e comunicação com a organização. Também nenhum dos seus atributos é **multivalorado** nem **composto**.

- **Participante**

Os atributos do Participante (id, nome, género, NIF, endereço, email e número de telemóvel) são todos **single-valued**, nenhum deles é **derivado** nem **composto**.

O NIF do participante pode ser **nulo**, uma vez que não é obrigatório que todos os participantes preencham esse campo (podem não desejar uma fatura com número de contribuinte).

- **Local**

Todos os atributos do Local (id, nome, endereço, lotação, descrição, tipo, email e número de telemóvel) terão que ser não **nulos**, **single-valued**, não **derivados** e não **compostos**.

- **Divulgação**

A Divulgação possui apenas atributos (id, conteúdo, tipo, custo, validade, data de início e data do final da divulgação) não **nulos**, **simples**, **single-valued** e não **derivados**. Se não houver um custo associado à divulgação, o valor do custo deverá ser 0.

- **Plataforma**

O relacionamento entre o Evento, Participante e a Divulgação dá origem apenas a atributos (classificação, estado, preço e lugar) **simples**, **single-valued** e não **derivados**.

A classificação poderá ser **nula**, caso o participante não indique o seu grau de satisfação para com o evento e o lugar também poderá ser nulo, se o evento ocorrer num local onde não seja necessário haver lugares marcados.

- **Relacionamento Evento - Participante - Divulgação**

O relacionamento entre o Evento, Participante e a Divulgação dá origem apenas a atributos (classificação, estado, preço e lugar) **simples**, **single-valued** e não **derivados**.

A classificação poderá ser **nula**, caso o participante não indique o seu grau de satisfação para com o evento.

3.4.1 Domínio dos Atributos

Todos os atributos possuem um tipo de dados e o respetivo domínio de valores que lhe podem ser atribuídos.

- **Evento**

Atributos	Tipo de Dados	Domínio de Valores
Id	INT	Número inteiro positivo.
Nome	VARCHAR(64)	Sequência de palavras.
Descrição	TEXT(512)	Sequência de palavras.
		Discussão, informação e formação; Religiosos e comunitários; Académicos; Políticos e de protocolo; Culturais, de lazer, desportivos e musicais; Culinária; Empresariais; Outros.
Tipo	ENUM(8)	Formato ISO 8601 (YYYY-MM-DDThh:mm:ss[.mmm]).
Data	DATETIME	Formato ISO 8601 (hh:mm:ss[.mmm]).
Duração	TIME	Formato ISO 8601 (YYYY-MM-DDThh:mm:ss[.mmm]).
Preço	DECIMAL(7,2)	Número decimal positivo.
Número Máximo de Participantes	INT	Número inteiro positivo.
Data Início Registo Participantes	DATETIME	Formato ISO 8601 (YYYY-MM-DDThh:mm:ss[.mmm]).
Data Fim Registo Participantes	DATETIME	Formato ISO 8601 (YYYY-MM-DDThh:mm:ss[.mmm]).
Classificação	DECIMAL(4,2)	Número decimal entre 0 e 10.

Tabela 2 - Caracterização dos atributos de Evento.

- **Organizador**

Atributos	Tipo de Dados	Domínio de Valores
Id	INT	Número inteiro positivo.
Nome	VARCHAR(64)	Sequência de palavras.
Descrição	TEXT(512)	Sequência de palavras.
Endereço	VARCHAR(128)	Sequência de palavras.
Email	VARCHAR(45)	Sequência de caracteres alfanuméricos, seguida do carácter “@” e por fim o domínio do provedor de serviço de email.
Número de Telemóvel	VARCHAR(15)	Sequência de números guardados em formato varchar.

Tabela 3 - Caracterização dos atributos de Organizador.

- **Participante**

Atributos	Tipo de Dados	Domínio de Valores
Id	INT	Número inteiro positivo.
Nome	VARCHAR(64)	Sequência de palavras.
Género	ENUM(2)	Feminino ou Masculino.
NIF	INT	Número inteiro positivo.
Endereço	VARCHAR(128)	Sequência de palavras.
Email	VARCHAR(45)	Sequência de caracteres alfanuméricos, seguida do carácter “@” e por fim o domínio do provedor de serviço de email.
Número de Telemóvel	VARCHAR(15)	Sequência de números guardados em formato varchar.
Data de Nascimento	DATE	Formato ISO 8601 (YYYY-MM-DD).

Tabela 4 - Caracterização dos atributos de Participante.

- **Local**

Atributos	Tipo de Dados	Domínio de Valores
Id	INT	Número inteiro positivo.
Nome	VARCHAR(64)	Sequência de palavras.
Endereço	VARCHAR(128)	Sequência de palavras.
Lotação	INT	Número inteiro positivo.
Descrição	TEXT(256)	Sequência de palavras.
Tipo	ENUM(9)	Bares; Casas noturnas; Restaurantes; Hotelaria; Casa de espetáculos; Centros Culturais; Multiusos; Quintas; Outros.
Email	VARCHAR(45)	Sequência de caracteres alfanuméricos, seguida do carácter “@” e por fim o domínio do provedor de serviço de email.
Número de Telemóvel	VARCHAR(15)	Sequência de números guardados em formato varchar.

Tabela 5 - Caracterização dos atributos de Local.

- **Divulgação**

Atributos	Tipo de Dados	Domínio de Valores
Id	INT	Número inteiro positivo.
Conteúdo	TEXT(1024)	Sequência de palavras.
Tipo	ENUM(10)	Audiovisual; Áudio; Publicação; Cartaz; Email; Brochuras;

		Apresentações; Promoções; Panfletos; Outros.
Custo	DECIMAL(7,2)	Número decimal positivo.
Validade	ENUM(3)	Válida, pendente ou inválida.
Data Início	DATETIME	Formato ISO 8601 (YYYY-MM-DDThh:mm:ss[.mmm]).
Data Fim	DATETIME	Formato ISO 8601 (YYYY-MM-DDThh:mm:ss[.mmm]).

Tabela 6 - Caracterização dos atributos de Divulgação.

- **Plataforma**

Atributos	Tipo de Dados	Domínio de Valores
Id	INT	Número inteiro positivo.
Nome	VARCHAR(64)	Sequência de palavras.

Tabela 7 - Caracterização dos atributos de Plataforma.

- **Relacionamento Evento - Participante - Divulgação**

Atributos	Tipo de Dados	Domínio de Valores
Classificação	INT	Número inteiro entre 0 e 10.
Estado	ENUM(3)	Válido, cancelado ou reservado.
Preço	DECIMAL(7,2)	Número decimal positivo.
Lugar	VARCHAR(32)	Sequência de caracteres alfanuméricos.

Tabela 8 - Caracterização dos atributos de Relacionamento Evento - Participante - Divulgação.

3.4.2 Chaves Candidatas, Primárias e Alternativas

Com o intuito de simplificar o processo da explicação da escolha das chaves candidatas adicionamos à Tabela 1 uma coluna chamada “Candidato”.

Para determinarmos se um atributo pode ser uma chave candidata pensamos como esse atributo se comporta no “mundo real”. Ou seja, um atributo será chave candidata caso não exista a hipótese de existir vários valores repetidos desse mesmo atributo.

- **Evento**

Chaves Candidatas: Id.

Chave Primária: Id.

Chaves Alternativas: Nenhuma.

Uma vez que a única chave candidata é o Id do evento (valor sem significado prático, apenas aplicacional), será esta a chave primária do evento.

- **Organizador**

Chaves Candidatas: Id, número de telemóvel e e-mail.

Chave Primária: Id.

Chaves Alternativas: Número de telemóvel e e-mail.

A chave primária do Organizador é o Id uma vez que é a chave candidata que representa mais eficiente e eficazmente o Organizador. Tanto o número de telemóvel como o e-mail poderão sofrer futuras alterações.

- **Participante**

Chaves Candidatas: Id, NIF, número de telemóvel e e-mail.

Chave Primária: Id.

Chaves Alternativas: NIF, número de telemóvel e e-mail.

O NIF, apesar de ser inequivocamente único para cada participante, poderá ser nulo, logo, não poderá ser uma chave primária. Pelo motivo já mencionado (na entidade Organizador), no caso da entidade Participante também o atributo Id é o que melhor consegue representar o participante.

- **Local**

Chaves Candidatas: Id.

Chave Primária: Id.

Chaves Alternativas: Nenhuma.

Uma vez que a única chave candidata é o Id do local (valor sem significado prático, apenas aplicacional), será esta a chave primária do local. Não consideramos o e-mail nem o número de telemóvel como chaves candidatas uma vez que o mesmo responsável (empresa ou individual) pelo local poderá ter mais locais a si associadas (e utilizar os mesmos contactos).

- **Divulgação**

Chaves Candidatas: Id.

Chave Primária: Id.

Chaves Alternativas: Nenhuma.

Uma vez que a única chave candidata é o Id da divulgação (valor sem significado prático, apenas aplicacional), será esta a chave primária da divulgação.

- **Plataforma**

Chaves Candidatas: Id.

Chave Primária: Id.

Chaves Alternativas: Nenhuma.

Uma vez que a única chave candidata é o Id da plataforma (valor sem significado prático, apenas aplicacional), será esta a chave primária da plataforma.

- **Relacionamento Evento - Participante - Divulgação**

O relacionamento terá como chave primária uma chave composta pelas chaves primárias de evento (id) e participante (id).

3.5. Detalhe ou Generalização de Entidades

Para evitar descrever conceitos semelhantes, tornar o modelo conceptual mais legível e por forma a adicionar mais informação semântica, nós decidimos, usando o modelo *Enhanced Entity-Relationship*, generalizar três das nossas entidades.

Sendo que tanto o Organizador, como o Participante, como o Local contêm 4 atributos iguais, mais precisamente telemóvel, email, id, endereço e nome, foi criada uma superclasse Entidade que passa estes por herança. A associação entre esta superclasse e as suas subclasses é **não disjunta** e **obrigatória**. Mais concretamente, é **obrigatória** porque todos os membros da superclasse têm de pertencer a uma das subclasses e é **não disjunta** porque uma ocorrência da superclasse apenas pode instanciar várias ocorrências das subclasses.

3.6. Apresentação e Explicação do Diagrama ER

Depois de explicada a importância e o significado de cada uma das entidades existentes no nosso sistema, cada um dos relacionamentos entre elas e os respectivos atributos, tanto das entidades, como dos relacionamentos, apresentamos o modelo conceptual:

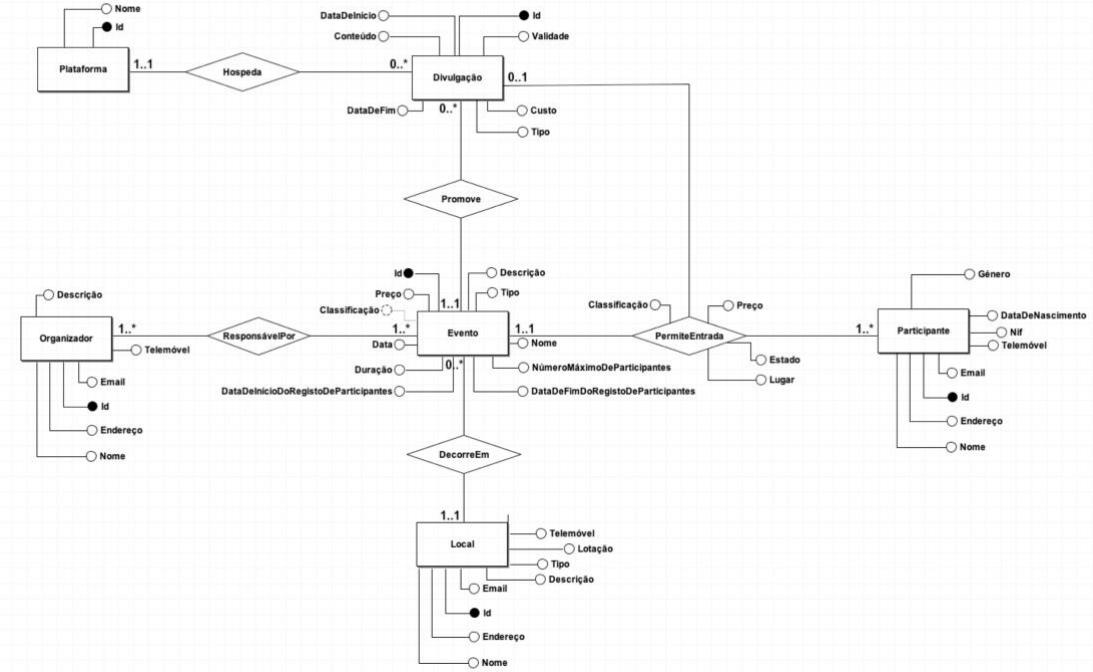


Figura 6 - Modelo Conceptual.

Assim, o funcionamento do nosso sistema começa a ganhar textura. Traduzindo o Modelo Conceptual ilustrado anteriormente para texto temos que: O Organizador é o responsável pelo Evento. Esse Evento decorre num Local. É permitida a entrada de um Participante num Evento com a ajuda de uma Divulgação. As plataformas hospedam as divulgações, divulgações essas que promovem os eventos.

Após a generalização, o diagrama ER evolui para o diagrama EER apresentado na figura seguinte (Figura 7) onde o **Organizador**, o **Participante** e o **Local** são subclasses da **Entidade**.

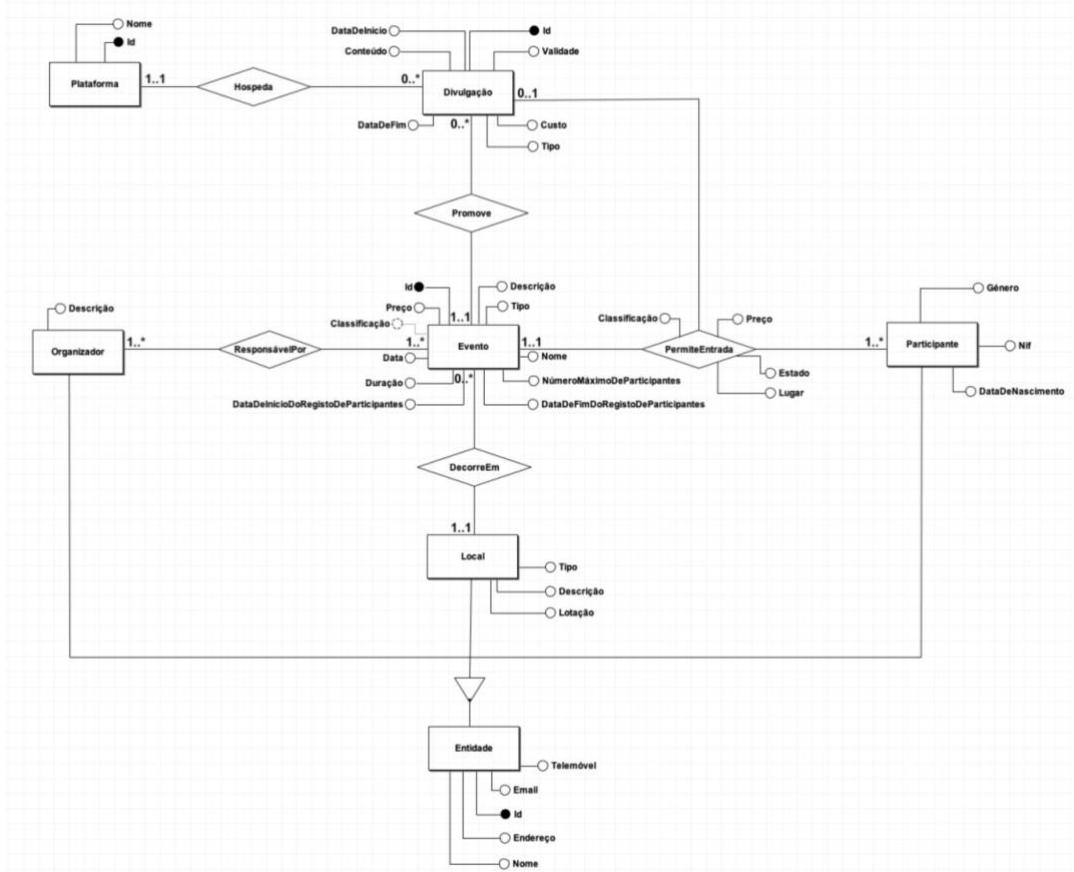


Figura 7 - Modelo Conceptual Final.

3.7. Validação do Modelo de Dados com o Utilizador

Após finalizado o modelo de dados, chegou a altura de nos reunirmos novamente com o utilizador para validarmos o nosso modelo conceptual final. Para isso, verificámos se o nosso modelo de dados consegue responder aos requisitos impostos:

1. Quando um organizador se regista, com um funcionário da Events Workbench, ele terá de indicar o nome, a sua descrição, o seu número de telemóvel, email e o endereço.

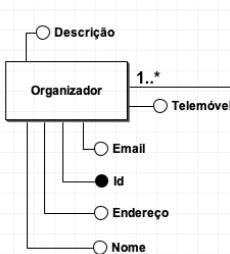


Figura 8 - Atributos de Organizador.

A entidade organizador tem associados todos os atributos referidos.

2. Um ou mais organizadores poderão criar e editar, junto ao funcionário, eventos.

Os funcionários da Events Workbench, uma vez que têm acesso total à base de dados, podem criar/editar eventos e associá-los aos respetivos organizadores, como se pode comprovar através do relacionamento da Figura 2 (Relacionamento entre o Organizador e o Evento).

3. Aquando do registo de um evento deverão ser indicados o respetivo nome, classificação, tipo, data, duração, período de registo de participantes, preço base e uma breve descrição.

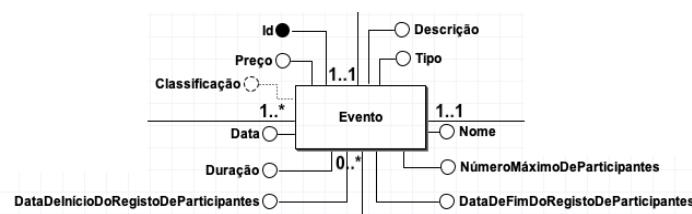


Figura 9 - Atributos de Evento.

A entidade evento tem associados todos os atributos referidos.

4. O evento terá de ser necessariamente de um dos seguintes tipos: Discussão, informação e formação; Religiosos e comunitários; Académicos, Políticos e de protocolo; Culturais, de lazer, desportivos e musicais; Culinária; Empresariais ou Outros.

O atributo tipo da entidade evento é um ENUM, com as alternativas presentes no requisito.

5. Cada evento deverá estar associado obrigatoriamente ao local onde decorrerá, para que o organizador possa controlar a organização do evento e acertar pormenores com os responsáveis pelo local onde ocorrerá o evento.

Como se pode ver através do relacionamento apresentado na Figura 3 e da respetiva cardinalidade, um evento está sempre obrigatoriamente associado a um local.

Uma vez que o Organizador tem acesso aos dados dos Eventos que organiza, como se pode ver no relacionamento da Figura 2, consequentemente terá também acesso ao local pelo qual poderá consultar todos os dados relativos ao mesmo.

- 6. Aquando da criação de um local deverá ser fornecida o seu nome, descrição, tipo, lotação, endereço, email e número de telemóvel.**

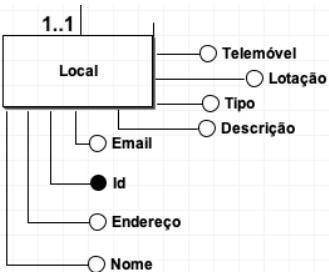


Figura 10 - Atributos de Local.

A entidade local tem associados todos os atributos referidos.

- 7. O local terá de ser necessariamente de um dos seguintes tipos: Bares, Casas noturnas, Restaurantes, Hotelaria, Casa de espetáculos, Centros Culturais, Multiusos, Quintas ou Outros.**

O atributo tipo da entidade local é um ENUM, com as alternativas presentes no requisito.

- 8. O evento ao ser criado deverá ainda conter informação relativa ao número máximo de participantes. Esse número nunca poderá ultrapassar a lotação do local.**

A entidade evento tem associados todos os atributos referidos, tal como se pode comprovar através da Figura 9.

Também a lotação do local é um atributo do Local (Figura 10), pelo que poderá ser assegurado que o número máximo de participantes não excede esse valor.

- 9. Em eventos com vários dias deverá ser criado um único evento para cada um dos dias.**

No nosso modelo de dados o Evento diz respeito a apenas uma ocorrência do evento, mesmo que possam existir várias edições do mesmo evento. Tanto o atributo data como duração, assim como o relacionamento ternário onde estão presentes o Evento e o Participante (que origina a lista de participantes do evento e o lugar que cada um dos mesmos ocupou), apenas têm em conta um único evento isolado.

10. A cada evento poderão ser registados os seus participantes.

Através do relacionamento onde está presente o Evento, o Participante e a Divulgação (Figura 5) ficam registados os participantes que participam em cada evento.

11. Num evento não pode haver mais do que um participante igual, ou seja, não pode haver participantes repetidos na lista de participantes do evento.

Através do relacionamento ternário ficam registados os participantes que estão inscritos no evento. Assim, quando um novo participante tenta inscrever-se num determinado evento pode ser averiguado se esse participante já está inscrito ou não no mesmo.

12. Cada participante poderá deixar uma classificação a cada evento que participou.

O participante poderá deixar uma classificação a cada evento que participou, que será armazenada no atributo Classificação do relacionamento ternário representado na Figura 5.

13. A Classificação de um evento é a média de classificações dadas pelos clientes.

A entidade Evento possui um atributo derivado chamado Classificação, como se pode ver na Figura 9.

14. Ao participar num evento um participante tem associado um lugar, caso este o suporte, um preço que este paga pode ser diferente do preço do evento e à sua participação pode estar associado um estado. Esse estado terá de ser um dos seguintes valores: Válido, Cancelado ou Reservado.

No relacionamento ternário existe um atributo Lugar reservado a cada participante no evento. Nesse relacionamento existe também o atributo Preço, que é o preço que o participante efetivamente pagou. Este atributo preço poderá ter um valor diferente do atributo Preço presente na entidade Evento.

O relacionamento ternário tem ainda um atributo Estado que será um ENUM, com uma das alternativas mencionadas no requisito (Válido, Cancelado ou Reservado).

15. Um funcionário poderá criar uma divulgação com a respetiva associação a um evento.

O relacionamento apresentado na Figura 4 demonstra que as divulgações de um evento ficam associadas ao evento.

16. Uma divulgação apenas pode ser partilhada quando o organizador a validar.

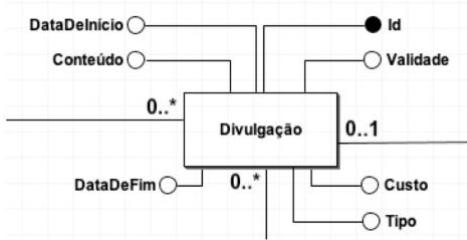


Figura 11 - Atributos de Divulgação.

Uma vez que existe um atributo na Divulgação chamado Validação, com as seguintes alternativas: Válida, Pendente ou Inválida, fica registado a validação (ou a falta dela) na Divulgação.

17. Quando da divulgação deverão ser registadas a data de início e de fim da divulgação, o custo da divulgação, o tipo e a plataforma.

A entidade divulgação tem associados todos os atributos referidos, tal como se pode ver na Figura 11.

18. A divulgação terá de ser necessariamente de um dos seguintes tipos: Audiovisual, Áudio, Publicação, Cartaz, Email, Brochuras, Apresentações, Promoções, Panfletos ou Outros.

O atributo tipo da entidade divulgação é um ENUM, com as alternativas presentes no requisito.

19. O participante terá de indicar a divulgação que o influenciou a participar num evento.

Através do relacionamento ternário entre as entidades Divulgação, Participante e Evento fica registado a divulgação que influenciou o participante a participar no evento (Figura 5).

20. A plataforma deverá ser definida por um nome.

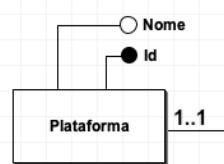


Figura 12 - Atributos de Plataforma.

A entidade plataforma tem associado o atributo referido.

21. Quando se regista um participante terão de ser fornecidos o seu nome, género, NIF, data de nascimento, contactos, email e número de telemóvel.

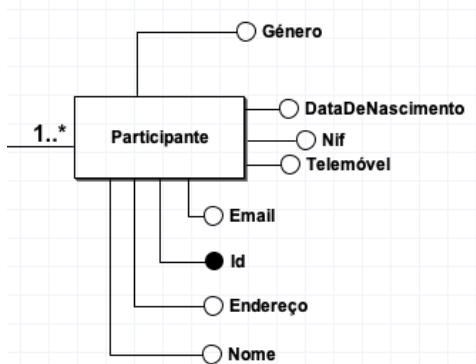


Figura 13 - Atributos de Participante.

A entidade participante tem associados todos os atributos referidos.

Assim, todos os requisitos de descrição requeridas pelo utilizador são conceptualmente possíveis.

Quanto aos requisitos de exploração, uma vez que todos os atributos criados foram de encontro com as necessidades expostas, todas consultas e verificações presentes nos requisitos de exploração podem serem satisfeitas.

Posto isto, concluímos que o modelo de dados está validado com o utilizador.

4. Modelação Lógica

4.1. Construção e Validação do Modelo de Dados Lógico

Em primeiro lugar, foram identificadas as entidades e os seus atributos.

Para terminar a construção do modelo de dados lógicos foram ainda seguidas as regras de derivação dos relacionamentos do modelo de dados relacional.

1. Relacionamento binário de grau 1:N com participação obrigatória do lado N:

São necessárias duas entidades lógicas, uma para cada entidade, e a chave primária da entidade do lado 1 tem de ser usada como atributo na entidade correspondente à entidade do lado N.

Ocorrências:

- “DecorreEm” entre Evento(N) e Local(1)
- “Promove” entre Divulgação(N) e Evento(1)
- “Hospeda” entre Plataforma(1) e Divulgação(N)

2. Relacionamento binário de grau N:M:

São sempre necessárias três entidades lógicas neste tipo de relacionamentos, uma para cada entidade e uma para o relacionamento. As chaves primárias das entidades têm de ser atributos na entidade lógica do relacionamento.

Ocorrências:

- “ResponsávelPor” entre Organizador e Evento

3. Relacionamento ternário

Na ocorrência “PermiteEntrada” entre Evento, Participante e Divulgação o valor de Divulgação pode ser nulo logo esta não faz parte da chave primária. Para além disso, devido aos nossos requisitos para um par Evento Participante só pode corresponder no máximo 1 divulgação logo as chaves primárias neste caso serão a chave primária do Evento e do Participante.

4. Relacionamento hierárquico

A superclasse dá origem a uma entidade com uma chave primária (seja denominada "K") e restantes atributos. Cada subclasse dá origem a uma entidade com os seus vários atributos e ainda com o atributo "K" que deverá ser chave primária.

Ocorrências:

- Entidade <| Organizador, Local, Participante

Entidades resultantes:

- **Plataforma** = {Id, Nome}
- **Divulgacao** = {Id, Conteudo, Tipo, Custo, Validade, DataInicio, DataFim, Plataforma_Id, Evento_Id}
- **Evento** = {Id, Nome, Descricao, Tipo, Data, Duracao, Preco, DataInicioRegistoParticipantes, DataFimRegistoParticipantes, Classificacao, NumeroMaximoDeParticipantes, Local_Entidade_Id }
- **Local** = {Entidade_Id, Lotacao, Descricao, Tipo}
- **Organizador** = {Entidade_Id, Descricao}
- **Participante** = {Entidade_Id, DataDeNascimento, Genero, Nif}
- **Entidade** = {Entidade_Id, Endereco, Nome, Email, Telemovel}
- **PermiteEntrada_Evento_Participante_Divulgacao** = {Evento_Id, Participante_Entidade_Id, Divulgacao_Id, Classificacao, Estado, Preco, Lugar}
- **Organizador_has_Evento** = {Organizador_Entidade_Id, Evento_Id}

4.2. Desenho do Modelo Lógico

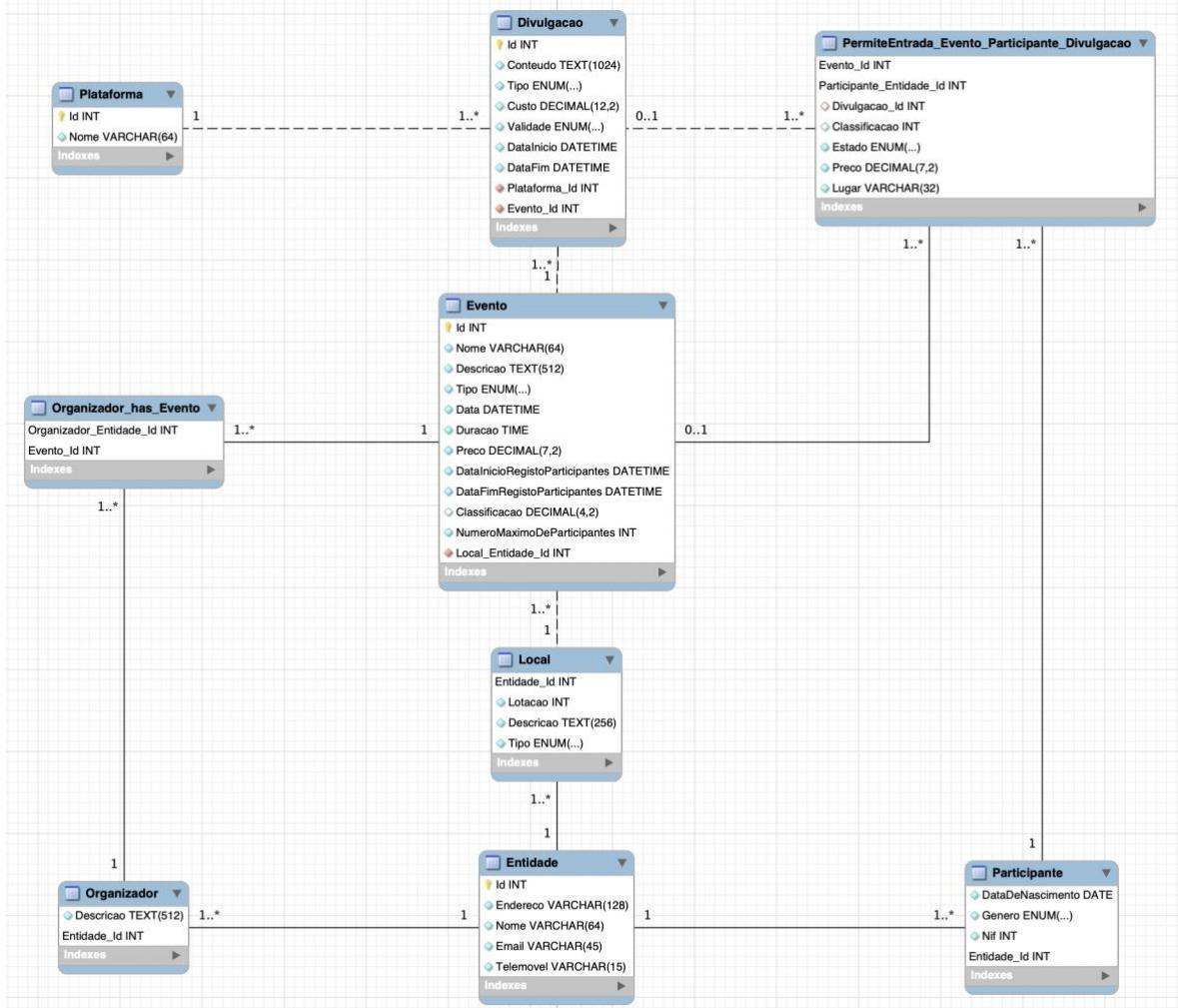


Figura 14 - Modelo Lógico.

4.3. Validação do Modelo através da Normalização

O modelo lógico encontra-se normalizado até à primeira forma normal. Um exemplo deste tipo de normalização é a tabela Plataforma. Dado que este conceito consta nos requisitos, se esta entidade fosse descartada a Divulgação teria de conter um atributo plataforma. A coluna correspondente, dado que à mesma plataforma podem corresponder várias Divulgações, iria conter grupos repetidos.

O modelo lógico encontra-se normalizado até à segunda forma normal. Um exemplo do uso desta regra é nas três tabelas Organizador, Evento e “Organizador_has_Evento”. Se estas três tabelas fossem agregadas numa só, todos os atributos seriam apenas parcialmente dependentes à chave primária identificação do organizador e identificação do evento.

Por fim, o modelo encontra-se também normalizado até à terceira forma normal. Um exemplo da aplicação desta regra é a tabela Local. Se em vez de existir esta tabela e os atributos que a compõem estivessem em Evento, dado que estes são totalmente funcionalmente dependentes de local, todos estes seriam unicamente dependentes do id de Evento por transitividade em local.

4.4. Validação do Modelo com as Interrogações do Utilizador

Para verificar se o modelo lógico desenvolvido encontra-se válido com base em algumas interrogações do utilizador seleccionamos as interrogações mais relevantes que constavam nos requisitos de exploração da base de dados e explicamos de que forma o nosso modelo lógico consegue adequadamente satisfazê-las.

1. Consultar o nome e a data de nascimento dos participantes de um dado evento.

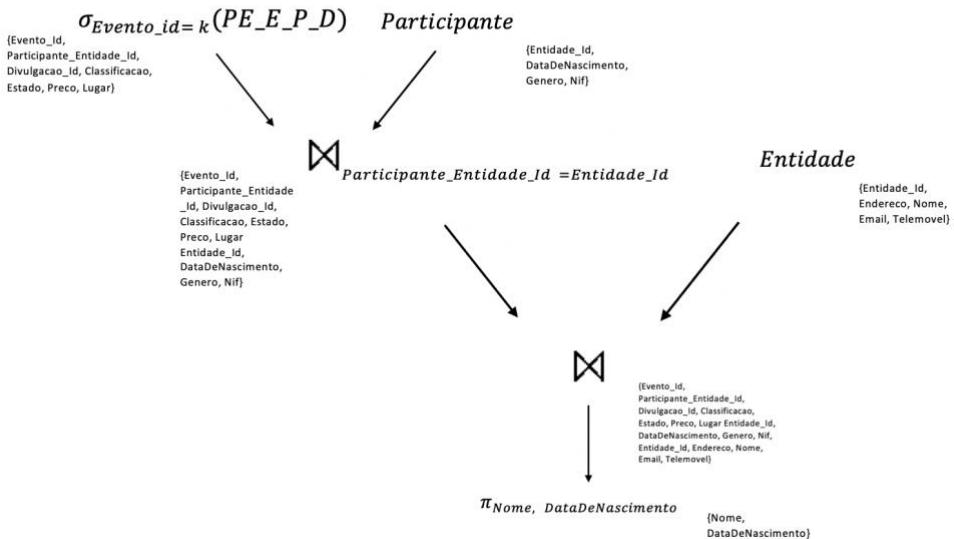


Figura 15 - Árvore representativa da Interrogação n.º 1.

Em primeiro lugar, são selecionados todos os tuplos da relação Evento_Participante_Divulgação cujo o Evento_Id é igual ao id do evento pretendido. De seguida, é feita a junção da relação resultante com a relação Participante onde Participante_Entidade_id corresponde a Entidade_id. Depois juntamos esta relação com a Entidade, onde id é correspondente a Entidade_id e projetamos os componentes nome e data de nascimento.

- 2. Consultar os nomes das plataformas utilizadas nas divulgações de um determinado evento.**

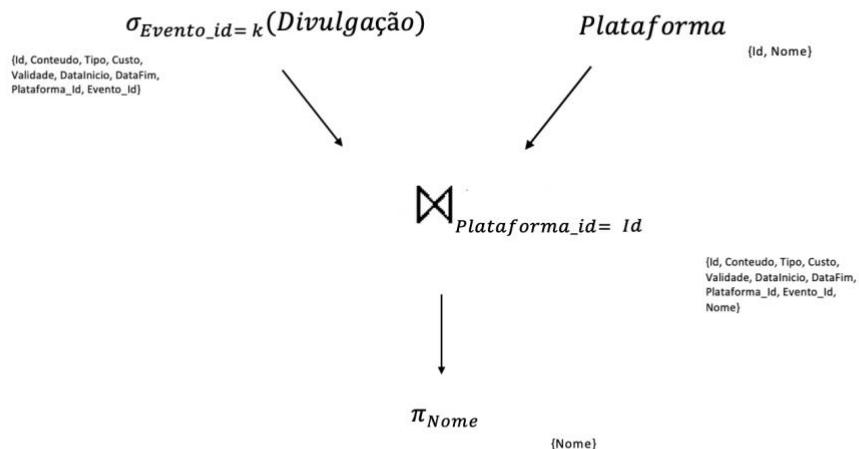


Figura 16 - Árvore representativa da Interrogação n.º 2.

Primeiramente são seleccionados os tuplos de Divulgação cujo Evento_Id é igual ao id do evento pretendido. Depois é feita a junção da relação resultante com a relação Plataforma em que Plataforma_Id corresponde a Id. Por fim, projetar o componente nome da relação resultante.

3. Consultar os ids e os nomes dos locais de todos os eventos organizados por um organizador.

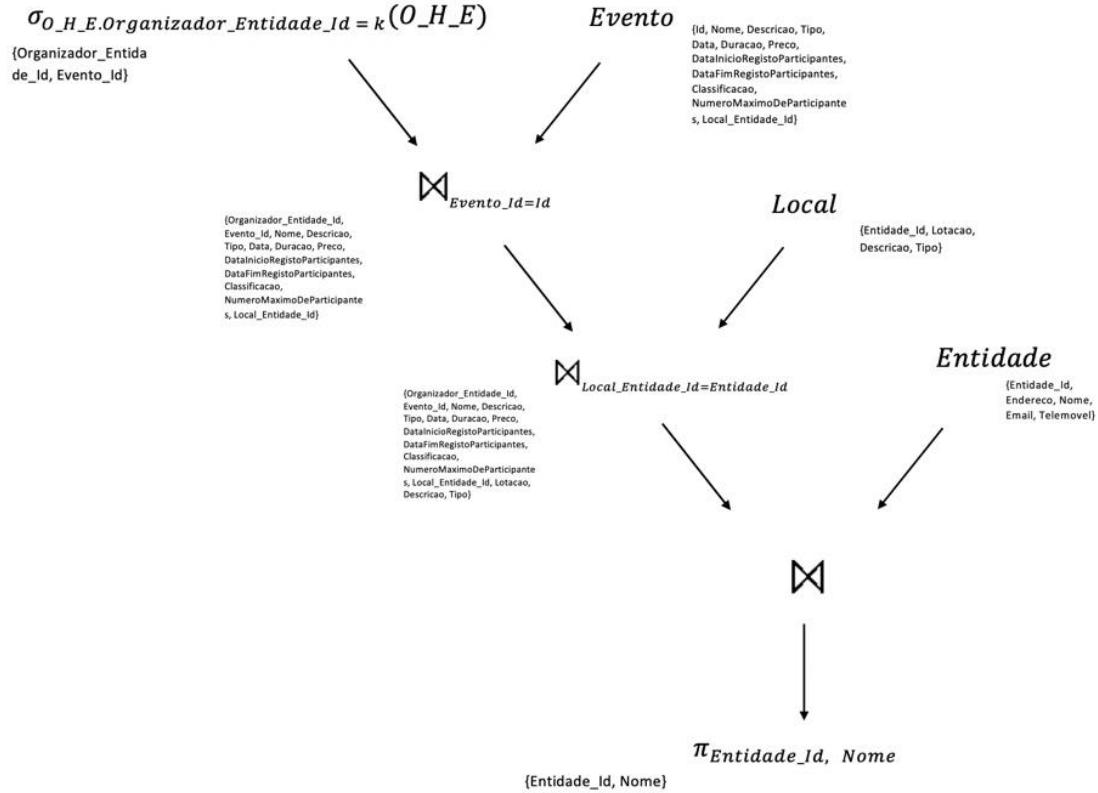


Figura 17 - Árvore representativa da Interrogação n.º 3.

Em primeiro lugar, são selecionados os tuplos da relação $Organizador_has_Evento$ cujo o $Organizador_Entidade_Id$ é igual ao id do organizador pretendido. Depois é feita a junção da relação resultante com a relação $Evento$ em que $Evento_Id$ corresponde a Id . Seguidamente é feita a junção da relação resultante com a relação $Local$ em que $Local_Entidade_Id$ corresponde a $Entidade_Id$. Após isto, a relação resultante é juntada com $Entidade$, onde $Entidade_id$ corresponde a id e, por fim, são projetados os componentes id e $nome$.

4. Consultar os nomes dos eventos organizados por um determinado Organizador.

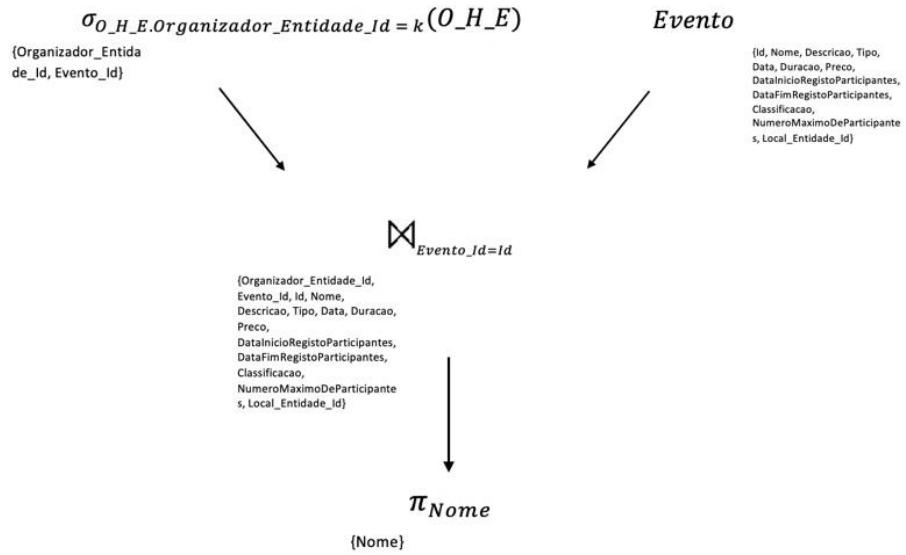


Figura 18 - Árvore representativa da Interrogação n.º 4.

Primeiramente são selecionados da relação Organizador_has_Evento os tuplos cujo componente Organizador_Entityade_Id é igual ao id do organizador pretendido. Seguidamente, é juntada a relação resultante com Evento em que Evento_Id corresponde a Id. Por fim, é aplicada uma projeção para o componente nome à relação resultante.

5. Consultar os eventos que ocorreram num determinado local e numa determinada data.

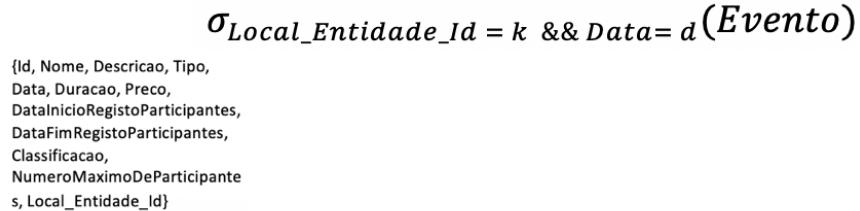


Figura 19 - Árvore representativa da Interrogação n.º 5.

São seleccionados os tuplos de Evento cujo Local_Entityade_Id é igual ao id do pretendido local e o valor na componente data é igual à data pretendida.

6. Determinar os Ids dos N participantes que gastaram mais dinheiro nos eventos de um determinado tipo e quanto gastaram.

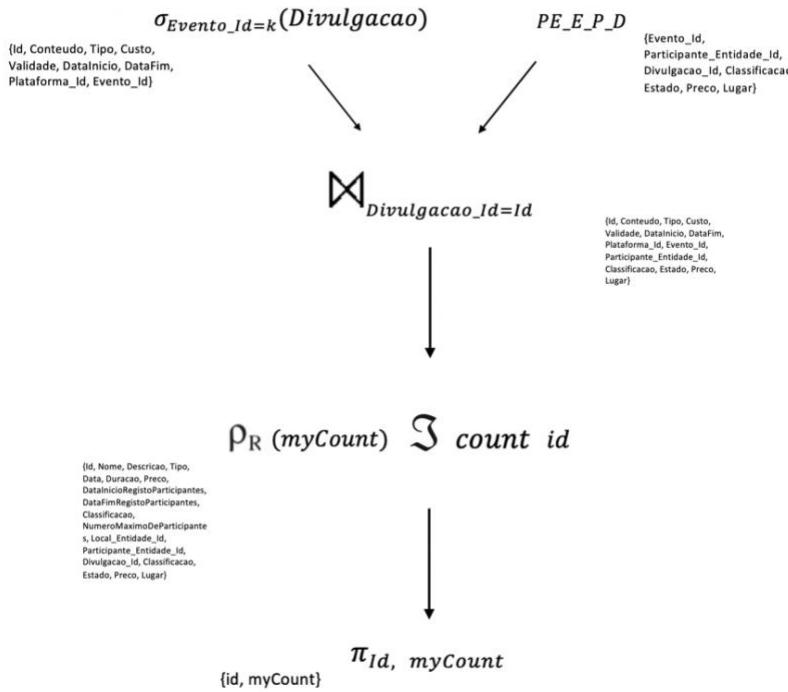


Figura 20 - Árvore representativa da Interrogação n.º 6.

Primeiramente, são selecionados os tuplos de Evento cujo a componente Tipo é igual ao tipo do evento pretendido. De seguida, é juntada a relação resultante com Evento_Participante_Divulgação em que Id corresponde a Evento_Id. Aplicar o operador de agrupamento à relação resultante na componente Participante_Entidade_Id e são agregados cada um dos grupos somando pela componente Preco. De seguida, é projetada a relação resultante pelo componente Participante_Entidade_Id e o valor somado. Por fim, são ordenados os tuplos resultantes pelo valor somado e são selecionados os primeiros N.

7. Consultar quantos participantes foram influenciados por uma determinada divulgação.

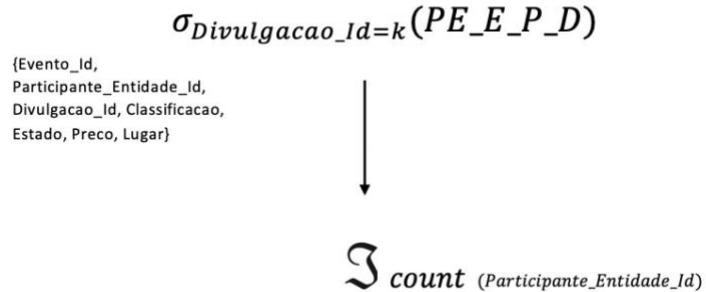


Figura 21 - Árvore representativa da Interrogação n.º 7.

São selecionados os tuplos da relação Evento_Participante_Divulgacao cujo o componente Divulgacao_Id é igual ao id da divulgação pretendido. De seguida é usado o operador de agregação que conta o número de tuplos constantes na relação.

8. Determinar qual é o Id da divulgação que mais participantes atraiu para um determinado evento.

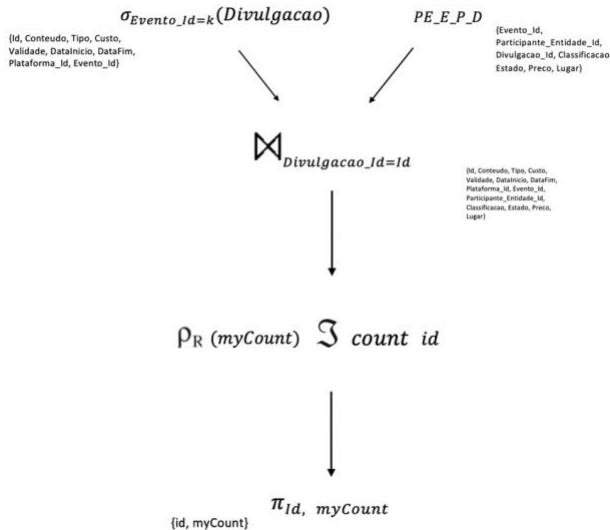


Figura 22 - Árvore representativa da Interrogação n.º 8.

Primeiramente são selecionados os tuplos da relação Divulgações em que Evento_Id é igual ao id do evento pretendido. De seguida, é juntada a relação resultante com a relação Evento_Participante_Divulgação em que Divulgacao_Id corresponde a Id. Entretanto são

agrupados os tuplos Id e através do operador de agregação são contados quantos tuplos estão em cada um dos grupos e por fim da relação resultante é projetado o Id e o valor contado.

4.5. Validação do Modelo com as Transações Estabelecidas

Foram estabelecidas algumas transações que serão posteriormente utilizadas pelos utilizadores (com as devidas permissões) da base de dados. Por se tratarem de transações, fica assegurado que cada uma destas é interpretada pelo sistema de base de dados como um bloco linear e atómico, pois as operações dentro de cada transação serão efetuadas sequencialmente nos dados. Deste modo, fica então garantido que as tabelas envolvidas na transação não serão editadas entre cada operação no bloco da transação por uma outra operação que interaja com esses mesmos dados.

A transação permite também a possibilidade de reverter tudo o que foi feito (ou seja, a base de dados não será alterada), caso alguma das operações corra mal. Assim, a esta altura do desenvolvimento do nosso modelo, esperamos que todas essas transações estabelecidas estejam válidas no nosso modelo.

1. Adicionar Local



Figura 23 - Entidade.

Para se adicionar um novo local à base de dados, ou seja, uma nova entrada na tabela Local, é necessário criar, primeiramente, uma **Entidade**, bastando para isso inserir as informações que se podem ver na Figura 23. Este procedimento é apenas tomado quando a Entidade que se pretende associar a local ainda não existe.



Figura 24 - Local.

Assim, para concluir a criação de uma entidade **Local**, é necessário adicionar a lotação máxima, uma descrição do local e o seu tipo, que será um dos seguintes tipos: bares, casas noturnas, restaurantes, hotelaria, casa de espetáculos, centros culturais, multiusos, quintas ou outros. Todos estes campos têm que ser obrigatoriamente preenchidos.

2. Adicionar Organizador

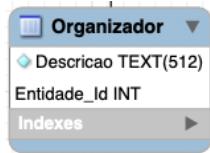


Figura 25 - Organizador.

Também no caso da inserção de um novo **Organizador** encontra-se o mesmo fenómeno encontrado na criação de local. Primeiro procede-se com a criação de uma **Entidade**, só depois a inserção em **Organizador** ao instanciar o atributo descrição.

3. Adicionar Participante



Figura 26 - Participante.

No caso do participante, também é necessário criar inicialmente a **Entidade** e, só depois, associar ao novo **Participante**, com as informações expostas na tabela.

4. Adicionar Evento

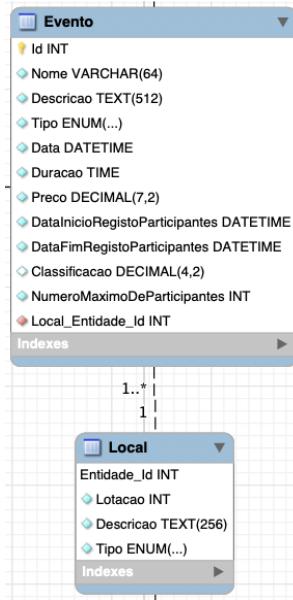


Figura 27 - Evento - Local.

Quando se pretende adicionar um **Evento** é adicionada a sua identificação (o atributo chave primária “Id”), o nome do evento, a descrição, a data e a duração do evento, o preço base, a data de início e de fim do registo dos participantes, a classificação, que é um atributo derivado calculado através da média das classificações atribuídas pelos participantes, que por *default* é nulo, o número máximo de participantes e o id do Local. Porém, quando o campo “número máximo de participantes” é preenchido deverá ser verificado se esse número não ultrapassa a capacidade máxima do local (lotação).

5. Associar um participante a um evento informando a divulgação

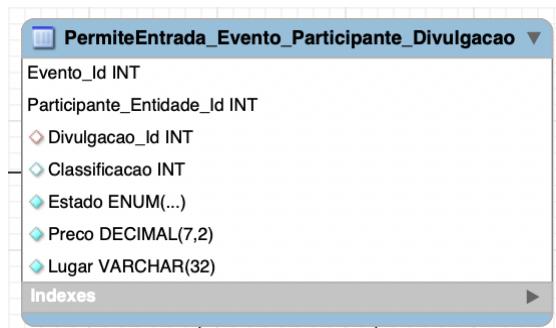


Figura 28 - Evento - Participante - Divulgação.

Quando se pretende adicionar um participante a um evento é necessário criar uma nova entrada na tabela “PermiteEntrada_Evento_Participante_Divulgacao”. Assim, as colunas respetivas ao id do Evento e ao id do Participante terão que ser preenchidas (chaves estrangeiras). Como a entrada pode não ser afetada por uma divulgação esta pode tomar valor nulo e não faz parte da chave primária. Isto também previne a existência de várias ligações de um utilizador a um evento. Terá ainda que ser verificado se o número máximo de participantes permitidos para o evento já foi atingido e, caso já tenha sido, a transação não será realizada.

O atributo classificação inicialmente será nulo, o estado poderá ser válido, cancelado ou reservado e, por *default*, será válido. O lugar que o participante deverá ocupar e o preço que ele pagou terá que ser preenchido. Quanto ao id da divulgação, terá que ser verificado se nessa divulgação o id do evento (“Evento_Id”) associado é o mesmo que o id do evento que compõe este relacionamento ternário (presente na tabela “PermiteEntrada_Evento_Participante_Divulgacao”).

Analisadas todas as transações estabelecidas com o utilizador, podemos concluir que o modelo está validado.

4.6. Revisão do Modelo Lógico com o Utilizador

Após concluir o desenvolvimento do modelo lógico foi agendada nova reunião com Events Workbench a fim de validar este modelo e obter autorização para avançar para a próxima fase do projeto.

Uma vez que este modelo é significativamente mais complexo que o modelo conceptual, o primeiro passo foi apresentar em traços gerais como analisar e interpretar este tipo de modelos, para que o cliente possa, de maneira fiável, confirmar que o modelo atende às necessidades da empresa.

Depois de verificar que todas as entidades do modelo conceptual estavam presentes no modelo lógico, o cliente verificou que todas elas tinham os atributos necessários e com o domínio certo de modo a armazenar toda a informação pretendida. Finalmente, em conjunto com o cliente, foram formuladas várias perguntas de exemplo para verificar que a estrutura do modelo era capaz de responder a todas as perguntas colocadas do modo mais simples e eficiente possível.

Cumpridas todas as exigências do cliente, ele deu autorização para avançar para a próxima fase do projeto.

5. Implementação Física

5.1. Seleção do Sistema de Gestão de Base de Dados

O Sistema de Gestão de Base de Dados Relacional escolhido para implementar a base de dados concebida foi o *MySQL*. *MySQL* é *open-source*, fácil de usar e contém uma grande comunidade ativa. Para além destas razões, o facto do sistema de armazenamento e gestão de transações do *MySQL*, denominado InnoDB, seguir os princípios **ACID**, Atomicidade, Consistência, Isolamento e Durabilidade, também constitui um forte motivador para adoção deste sistema de gestão e base de dados relacional.

5.2. Tradução do Esquema Lógico para o Sistema de Gestão de Bases de Dados escolhido em SQL

Para, de melhor forma, usufruir dos mecanismos que *MySQL* oferece nós decidimos, durante todo o processo, usar a ferramenta de desenho, desenvolvimento e administração de base de dados, *MySQL Workbench*. Dado que o nosso modelo lógico foi criado nesta ferramenta nós pudemos usufruir do *Forward Engineering* que esta oferece. Este mecanismo, tal como o seu nome elucida permite automatizar o processo *top-down* de construção de componentes de baixo nível, neste caso a implementação no sistema de gestão de base de dados, através de uma abstração de alto nível, o modelo Lógico. Tendo obtido o código resultante deste processo apenas tivemos que o executar para que a base de dados fosse criada.

5.3. Tradução das Interrogações do Utilizador para SQL

Como o sistema de gestão de base de dados optimiza automaticamente todas as seleções que inter-relacionam diferentes tabelas. Na tradução da álgebra relacional, em que são resolvidas as diferentes interrogações, para SQL apenas tivemos que especificar quais as

propriedades de correspondência das entradas das relações intervenientes pretendemos tabeladas. Para este efeito usamos predominantemente o comando *where*.

Para a implementação das interrogações parametrizadas foram usados *stored procedures*. *Stored procedures* permitem guardar blocos de código SQL na base de dados para que mais tarde estes possam ser chamados, usando o nome da *stored procedure*, em vez de serem escritos na sua totalidade. Para além desta vantagem na interpretabilidade, os *stored procedures* ao serem guardados são pré-processados, podendo assim ser otimizados para uso futuro. Adicionalmente, estes permitem centralizar a lógica de acesso à base de dados, assim como o acesso controlado a dados restritos a um dado perfil de utilização.

Suplementarmente, para as interrogações não parametrizadas foram usadas *views*. *Views* são para os utilizadores apenas outra tabela, no entanto, ao contrário das tabelas definidas na criação da base de dados, estas não se encontram guardadas como linhas e colunas de dados. Apenas quando são selecionadas é que são obtidos os tuplos correspondentes a cada uma das entradas.

- **Alguns exemplos de interrogações implementadas:**

1. O organizador deverá ser capaz de consultar o nome, a descrição, o endereço e o tipo do local onde se realizará em qualquer um dos eventos por ele organizado.

```
delimiter $$  
CREATE PROCEDURE OLocaisDeEventos(IN n_org int)  
begin  
    Select distinct  
        Ent.nome,  
        Ent.endereco,  
        Ent.email,  
        Ent.telemovel,  
        L.tipo,  
        L.descricao,  
        L.lotacao  
    From  
        Evento as E,  
        Organizador as O,  
        Organizador_has_Evento as OE,  
        Local as L,  
        Entidade as ent  
    where  
        E.id = OE.Evento_id  
        and OE.Organizador_Entidade_id = n_org  
        and L.Entidade_id = E.Entidade_id  
        and Ent.id = L.Entidade_id;  
end $$
```

Figura 29 - Imagem do código obtido para a query n.^o 1.

2. Um funcionário da empresa poderá consultar, para um número indicado de participantes, os que mais dinheiro gastam em eventos por organizador.

```
delimiter $$  
CREATE PROCEDURE FParticipanteMaisGastaOrg(In n int, In n_org int)  
begin  
    Select  
        E.*,  
        P.*,  
        sum(PEPD.Preco) as Gasto  
    From  
        Participante as P,  
        PermiteEntrada_Evento_Participante_Divulgacao as PEPD,  
        Entidade as E,  
        Evento as EV,  
        Organizador as O,  
        Organizador_has_Evento as OE  
  
    where  
        P.Entidade_id = PEPD.Participante_Entidade_Id  
        and P.Entidade_id = E.id  
        and PEPD.Evento_id = EV.id  
        and EV.id = OE.Evento_id  
        and OE.Organizador_Entidade_id = n_org  
  
    group by P.Entidade_id  
    order by sum(PEPD.Preco) DESC  
    limit n;  
end $$  
delimiter ;
```

Figura 30 - Imagem do código obtido para a query n.^o 2.

3. Os funcionários da empresa poderão consultar para cada divulgação quantos participantes indicaram que esta os influenciou diretamente no ingresso do correspondente evento.

```

CREATE VIEW FDivulgacaoInfluencia AS
Select
    D.*,
    count(PEPD.Participante_Entidade_Id) AS Influenciados
From
    Divulgacao AS D,
    PermiteEntrada_Evento_Participante_Divulgacao AS PEPD
Where
    PEPD.Divulgacao_id = D.id
Group By
    PEPD.Divulgacao_id
Order By
    count(PEPD.Participante_Entidade_Id) DESC;

```

Figura 31 - Imagem do código obtido para a query n.º 3.

4. Para cada tipo de evento, um funcionário, deverá poder verificar qual o tipo de divulgação mais eficaz em termos de custo por participante de divulgação.

```

delimiter $$
CREATE PROCEDURE FDivulgacaoEficazTipoProporcao(IN n_tipo_e INT)
begin
    Select
        D.tipo,
        sum(PEPD.Preco)/count(PEPD.Participante_Entidade_Id) AS 'CustoPorParticipante'
    From
        Divulgacao AS D,
        Entidade AS E,
        PermiteEntrada_Evento_Participante_Divulgacao AS PEPD
    Where
        E.tipo = n_tipo_e
        AND D.Evento_id = E.id
        AND PEPD.Evento_id = E.id
        AND PEPD.Divulgacao_id = D.id
    Group By D.tipo
    Order By sum(PEPD.Preco)/count(PEPD.Participante_Entidade_Id) DESC;
end $$ 
delimiter ;

```

Figura 32 - Imagem do código obtido para a query n.º 4.

5.4. Tradução das transações estabelecidas para SQL

As operações que devem ser efetuadas na implementação de cada uma das transações já haviam sido especificadas na validação do modelo com as transações estabelecidas (Secção 4.5). Nesta fase, traduzimos essas operações para SQL.

A par das interrogações, também foram utilizados *stored procedures*, para permitir que estes procedimentos podem ser mais tarde utilizados. Assim, depois de criado o *procedure* inicia-se a *transaction* e, depois, efetuamos as várias operações pretendidas.

1. Adicionar Local

```
DELIMITER $$  
CREATE PROCEDURE addLocal(IN id INT, IN endereco VARCHAR(128), IN nome VARCHAR(64),  
                         IN email VARCHAR(45), IN telemovel VARCHAR(15),  
                         IN lotacao INT, IN descricao TEXT(256),  
                         IN tipo ENUM('Bares', 'Casas noturnas', 'Casa de espetáculos', 'Hotelaria', 'Restaurantes',  
                         'Centros Culturais', 'Multiusos', 'Quintas', 'Outros'))  
BEGIN  
  
    DECLARE EXIT HANDLER FOR SQLEXCEPTION  
    BEGIN  
        ROLLBACK;  
    END;  
  
    START TRANSACTION;  
    -- criar Entidade  
    INSERT INTO Entidade (id, endereco, nome, email, telemovel)  
    VALUES (id, endereco, nome, email, telemovel);  
    -- criar Local  
    INSERT INTO Local (entidade_id, lotacao, descricao, tipo)  
    VALUES (id, lotacao, descricao, tipo);  
  
    COMMIT;  
  
END $$  
DELIMITER ;
```

Figura 33 - Imagem do código obtido para a transação n.º 1.

Quando se tenciona adicionar um Local sem se ter inicialmente a Entidade na base de dados é necessário primeiro inserir a Entidade e só depois o Local.

O resultado será uma nova entrada na tabela Local e na tabela Entidade, caso ambas as operações executem com êxito, ou nenhuma nova entrada na base de dados, caso alguma delas corra mal.

2. Adicionar Organizador

```
DELIMITER $$  
CREATE PROCEDURE addOrganizador(IN id INT, IN endereco VARCHAR(128), IN nome VARCHAR(64),  
                                IN email VARCHAR(45), IN telemovel VARCHAR(15), IN descricao TEXT(512))  
BEGIN  
    DECLARE EXIT HANDLER FOR SQLEXCEPTION  
    BEGIN  
        ROLLBACK;  
    END;  
  
    START TRANSACTION;  
    -- criar Entidade  
    INSERT INTO Entidade (id, endereco, nome, email, telemovel)  
    VALUES (id, endereco, nome, email, telemovel);  
    -- criar Organizador  
    INSERT INTO Organizador(entidade_id, descricao)  
    VALUES (id, descricao);  
  
    COMMIT;  
  
END $$  
DELIMITER ;
```

Figura 34 - Imagem do código obtido para a transação n.º 2.

Quando se tentiona adicionar um Organizador sem se ter inicialmente a Entidade na base de dados, o procedimento é igual ao anterior. Primeiro adiciona-se a Entidade e só posteriormente o Organizador.

3. Adicionar Participante

```
DELIMITER $$  
CREATE PROCEDURE addParticipante(IN id INT, IN endereco VARCHAR(128), IN nome VARCHAR(64),  
                                IN email VARCHAR(45), IN telemovel VARCHAR(15),  
                                IN datadenascimento DATE, IN genero ENUM('Feminino', 'Masculino'), IN nif INT)  
BEGIN  
    DECLARE EXIT HANDLER FOR SQLEXCEPTION  
    BEGIN  
        ROLLBACK;  
    END;  
  
    START TRANSACTION;  
    -- criar Entidade  
    INSERT INTO Entidade (id, endereco, nome, email, telemovel)  
    VALUES (id, endereco, nome, email, telemovel);  
    -- criar Participante  
    INSERT INTO Participante (entidade_id, datadenascimento, genero, nif)  
    VALUES (id, datadenascimento, genero, nif);  
  
    COMMIT;  
  
END $$  
DELIMITER ;
```

Figura 35 - Imagem do código obtido para a transação n.º 3.

No caso de se adicionar um Participante o raciocínio é o anteriormente utilizado, primeiro adiciona-se a Entidade e depois adiciona-se o Participante.

4. Adicionar Evento

```
DELIMITER $$  
CREATE PROCEDURE addEvento(IN id INT, IN nome VARCHAR(64), IN descricao TEXT(512),  
    IN tipo ENUM('Discussão, informação e formação', 'Religiosos e comunitários', 'Académicos',  
    'Políticos e de protocolo', 'Culturais, de lazer, desportivos e musicais', 'Culinária', 'Empresariais', 'Outros'),  
    IN data DATETIME, IN duracao TIME, IN preco DECIMAL(7,2), IN dataInicioRegistoParticipantes DATETIME,  
    IN dataFimRegistoParticipantes DATETIME, IN classificacao DECIMAL(4,3), IN numeroMaximoDeParticipantes INT,  
    IN local_Entidade_Id INT)  
BEGIN  
    DECLARE maxlotacao INT;  
    DECLARE `_rollback` BOOL DEFAULT 0;  
    DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `_rollback` = 1;  
  
    START TRANSACTION;  
        -- guardar lotação máxima do local numa variável  
        SET maxlotacao = (SELECT Local.lotacao  
            FROM Local  
            WHERE Local.Entidade_Id = local_Entidade_Id);  
        -- verificar se o local possui a lotação máxima necessária  
        IF (maxlotacao < numeroMaximoDeParticipantes) THEN  
            SET `_rollback` = 1;  
        END IF;  
        -- criar Evento  
        INSERT INTO Evento (id, nome, descricao, tipo, data, duracao, preco, dataInicioRegistoParticipantes,  
            dataFimRegistoParticipantes, classificacao, numeroMaximoDeParticipantes, local_Entidade_Id)  
        VALUES (id, nome, descricao, tipo, data, duracao, preco, dataInicioRegistoParticipantes,  
            dataFimRegistoParticipantes, classificacao, numeroMaximoDeParticipantes, local_Entidade_Id);  
  
        IF `_rollback` THEN  
            ROLLBACK;  
        ELSE  
            COMMIT;  
        END IF;  
    END TRANSACTION;  
END $$  
DELIMITER ;
```

Figura 36 - Imagem do código obtido para a transação n.º 4.

Quando se adiciona um novo Evento é necessário garantir que o Local onde vai ocorrer o Evento possui a lotação necessária para reunir o número máximo de participantes estabelecidos para o Evento em questão. Caso esta restrição não seja satisfeita, o Evento não é registado na base de dados.

5. Associar um participante a um evento informando a divulgação

```

DELIMITER $$

CREATE PROCEDURE addParticipanteEventoDivulgacao(IN participante_entidade_id INT, IN evento_id INT, IN divulgacao_id INT,
                                                IN classificacao INT, IN estado ENUM('Válido', 'Cancelado', 'Reservado'),
                                                IN preco DECIMAL(7,2), IN lugar VARCHAR(32))

BEGIN
    DECLARE evento_divulgacao INT;
    DECLARE nr_partic平antes_maximo INT;
    DECLARE nr_partic平antes_no_evento INT;
    DECLARE `_rollback` BOOL DEFAULT 0;
    DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `_rollback` = 1;

    START TRANSACTION;
        -- guardar variavel com o id do evento publicitado na divulgacao
        SET evento_divulgacao = (SELECT d.evento_id
                                  FROM Divulgacao d
                                  WHERE d.id = divulgacao_id);
        -- numero maximo de participantes permitido evento
        SET nr_partic平antes_maximo = (SELECT e.NumeroMaximoDeParticipantes
                                         FROM Evento e
                                         WHERE e.id = evento_id);
        -- numero de participantes inscritos no evento
        SET nr_partic平antes_no_evento = (SELECT COUNT(pe.evento_id)
                                         FROM PermiteEntrada_Evento_Participante_Divulgacao pe
                                         WHERE pe.evento_id = evento_id);
        -- verificar se o evento associado à divulgação é o mesmo evento que se prende adicionar ao relacionamento ternário
        IF (evento_divulgacao <> evento_id) THEN
            SET `_rollback` = 1;
        END IF;
        -- verificar se o número de participantes máximo já foi atingido
        IF (nr_partic平antes_maximo = nr_partic平antes_no_evento) THEN
            SET `_rollback` = 1;
        END IF;
        -- criar Evento-Participante-Divulgacao
        INSERT INTO PermiteEntrada_Evento_Participante_Divulgacao (participante_entidade_id, evento_id,
                                                                     divulgacao_id, classificacao, estado, preco, lugar)
        VALUES (participante_entidade_id, evento_id, divulgacao_id, classificacao, estado, preco, lugar);

        IF `_rollback` THEN
            ROLLBACK;
        ELSE
            COMMIT;
        END IF;
    END TRANSACTION;
END $$

DELIMITER ;

```

Figura 37 - Imagem do código obtido para a transação n.º 5.

Para se poder criar uma nova entrada na tabela “PermiteEntrada_Evento_Participante_Divulgacao”, que indica que um participante participou num evento influenciado por uma determinada divulgação, é necessário garantir que o evento publicitado por essa divulgação é o mesmo evento em que o participante esteve. É também necessário que o número máximo de participantes ainda não tenha sido atingido. Só deste modo poderá ser registado este procedimento, pois basta que pelo menos uma destas restrições não seja satisfeita para a inserção não ser concretizada.

5.5. Triggers SQL

No nosso problema temos um atributo derivado, chamado Classificação, que está presente na tabela Evento. Este atributo é a média das classificações dadas pelos participantes aos eventos (que se encontram na tabela “PermiteEntrada_Evento_Participante_Divulgacao”). A utilização de um *trigger* permite-nos que sempre que um desses valores seja atualizado, o atributo derivado Classificação é também atualizado, automaticamente.

Existem três hipóteses onde há necessidade do atributo derivado ser atualizado:

1. Numa entrada já existente da tabela “PermiteEntrada_Evento_Participante_Divulgacao”, o valor da classificação ser atualizado:

```
CREATE TRIGGER after_classificacao_update
AFTER UPDATE ON PermiteEntrada_Evento_Participante_Divulgacao
FOR EACH ROW
BEGIN

DECLARE nr_classificacoes INT;
DECLARE ex_media DECIMAL(4,2);
DECLARE old_somaclassificacao_total INT;
DECLARE result INT;
SET nr_classificacoes = (SELECT COUNT(classificacao)
                        FROM PermiteEntrada_Evento_Participante_Divulgacao
                        WHERE evento_id = NEW.evento_id AND (classificacao IS NOT NULL));
SET ex_media = (SELECT classificacao
                FROM Evento
                WHERE id = NEW.evento_id);

IF OLD.classificacao IS NOT NULL THEN
    IF NEW.classificacao IS NOT NULL THEN
        SET old_somaclassificacao_total = nr_classificacoes * ex_media;
        SET result = (old_somaclassificacao_total - OLD.classificacao + NEW.classificacao) / nr_classificacoes;
    ELSE
        IF nr_classificacoes <> 0 THEN
            SET old_somaclassificacao_total = (nr_classificacoes + 1) * ex_media;
            SET result = (old_somaclassificacao_total - OLD.classificacao) / nr_classificacoes;
        ELSE
            SET result = null;
        END IF;
    END IF;
ELSE
    IF ex_media IS NOT NULL THEN
        SET old_somaclassificacao_total = (nr_classificacoes - 1) * ex_media;
    ELSE
        SET old_somaclassificacao_total = 0;
    END IF;
    SET result = (old_somaclassificacao_total + NEW.classificacao) / nr_classificacoes;
END IF;

UPDATE Evento
SET Evento.Classificacao = result
WHERE Evento.id = NEW.evento_id;

END $$

DELIMITER ;
```

Figura 38 - Imagem do código obtido para o *trigger* n.º 1.

Quando um Participante adiciona uma classificação ao Evento em que participou, a classificação total desse evento deverá ser automaticamente atualizada. Assim, sempre que existir uma **atualização** na tabela “PermiteEntrada_Evento_Participante_Divulgacao” o presente trigger é ativado (*after update*).

No corpo do *trigger* tivemos que ter em atenção tanto se a antiga classificação (antes de sofrer a alteração) era nula, como se a classificação do evento (derivada) era nula ou passou a ser. Depois de analisadas todas as alternativas possíveis o *trigger* resultante é o que se encontra na Figura 38.

2. É adicionada uma nova entrada à tabela “PermiteEntrada_Evento_Participante_Divulgacao”, com o valor da classificação diferente de nulo:

DELIMITER \$\$

```
CREATE TRIGGER after_classificacao_insert
AFTER INSERT ON PermiteEntrada_Evento_Participante_Divulgacao
FOR EACH ROW
BEGIN

DECLARE nr_classificacoes INT;
DECLARE ex_media DECIMAL(4,2);
DECLARE old_somaclassificacao_total INT;

IF NEW.classificacao IS NOT NULL THEN

    SET nr_classificacoes = (SELECT COUNT(classificacao)
                                FROM PermiteEntrada_Evento_Participante_Divulgacao
                                WHERE evento_id = NEW.evento_id AND (NEW.classificacao IS NOT NULL));
    SET ex_media = (SELECT classificacao
                    FROM Evento
                    WHERE id = NEW.evento_id);

    IF ex_media IS NOT NULL THEN
        SET old_somaclassificacao_total = (nr_classificacoes - 1) * ex_media;
    ELSE
        SET old_somaclassificacao_total = 0;
    END IF;

    UPDATE Evento
    SET Evento.Classificacao = (old_somaclassificacao_total + NEW.classificacao) / nr_classificacoes
    WHERE Evento.id = NEW.evento_id;

END IF;
END $$

DELIMITER ;
```

Figura 39 - Imagem do código obtido para o *trigger* n.º 2.

Este *trigger* será ativado sempre que houver uma nova inserção (*after insert*) na tabela “PermiteEntrada_Evento_Participante_Divulgacao”.

Mais uma vez todas as restrições foram analisadas e o código resultante encontra-se na Figura 39.

3. É eliminada uma entrada da tabela “PermiteEntrada_Evento_Participante_Divulgacao”, com o valor da classificação diferente de nulo:

DELIMITER \$\$

```

CREATE TRIGGER after_classificacao_delete
AFTER DELETE ON PermiteEntrada_Evento_Participante_Divulgacao
FOR EACH ROW
BEGIN

DECLARE nr_classificacoes INT;
DECLARE ex_media DECIMAL(4,2);
DECLARE result INT;
DECLARE old_somaclassificacao_total INT;

IF OLD.classificacao IS NOT NULL THEN
    SET nr_classificacoes = (SELECT COUNT(classificacao)
                             FROM PermiteEntrada_Evento_Participante_Divulgacao
                             WHERE evento_id = OLD.evento_id AND (classificacao IS NOT NULL));
    SET ex_media = (SELECT classificacao
                    FROM Evento
                    WHERE id = OLD.evento_id);
    IF nr_classificacoes <> 0 THEN
        SET old_somaclassificacao_total = (nr_classificacoes + 1) * ex_media;
        SET result = (old_somaclassificacao_total - OLD.classificacao) / nr_classificacoes;
    ELSE
        SET result = null;
    END IF;
    UPDATE Evento
    SET Evento.Classificacao = result
    WHERE Evento.id = OLD.evento_id;
END IF;
END $$

DELIMITER ;

```

Figura 40 - Imagem do código obtido para o trigger n.º 3.

Por fim, também quando alguma entrada na tabela é apagada (*after delete*) existe a necessidade da média final ser atualizada (atributo derivado Classificação).

Neste caso, é necessário ter em consideração se deixaram de existir atributos com valor definido (todas as entradas iguais a nulo) para poder ser colocado o atributo derivado também a nulo. O resultado final está representado na Figura 40.

5.6. Escolha, Definição e Caracterização de Índices em SQL

Apesar de a opção de *Forward Engineering* criar índices por predefinição, para as chaves primárias por exemplo, achamos necessária a criação de mais alguns índices de forma a melhorar o tempo de execução das *queries*.

Alguns dos índices que definimos são:

A criação de dois índices para as chaves estrangeiras (Participante_Entidade_Id e Divulgacao_Id) na tabela Permite_Entrada_Evento_Participante_Divulgacao. Não criamos índice para a Evento_Id porque o índice gerado pela opção de *Forward Engineering* é um *Multiple-Column Indexes*, devido a ter três chaves primárias, o qual suporta as procura por Evento_Id, uma vez que este se encontra na primeira posição do índice gerado.

```
CREATE INDEX idx_participante ON Permite_Entrada_Evento_Participante_Divulgacao (Participante_Entidade_Id);
CREATE INDEX idx_divulgacao ON Permite_Entrada_Evento_Participante_Divulgacao (Divulgacao_Id);
```

Figura 41 - Imagem do código obtido índices participante e divulgação.

Ao criar estes dois índices não só aumentamos a eficiência das procura por na tabela por Participante_Entidade_Id e por Divulgacao_Id como também aumentamos a eficiência dos *Joins*, uma vez que maioria dos *joins* com esta tabela envolvem estes atributos.

Outro dos índices que criamos foi um índice para o tipo dos locais.

```
CREATE INDEX idx_tipo ON Local (Tipo);
```

Figura 42 - Imagem do código obtido índice idx_tipo.

Este índice irá melhorar a procura de locais de um determinado tipo, operação que esperamos que seja utilizada com frequência pelos funcionários.

E por fim, criamos um índice para a tabela de Divulgação em relação à chave estrangeira Evento_Id.

```
CREATE INDEX idx_evento ON Divulgacao (Evento_Id);
```

Figura 43 - Imagem do código obtido índice idx_evento.

Este índice não só irá melhorar as procura das divulgações de um determinado evento mas também irá melhorar a performance das operações de *Join* da Tabela Evento com a tabela Divulgação que dizem respeito ao atributo Evento_Id. Operações que nós esperamos efetuar várias vezes para os diversos Organizadores e para os funcionários.

5.7. Estimativa do Espaço em Disco da Base de Dados e Taxa de Crescimento Anual

O cálculo das estimativas de espaço baseia-se no capítulo 11.8 do *MySQL 8.0 Reference Manual*.

Os valores foram calculados somando o número de bytes necessários para cada atributo.

Tabela	Espaço Esperado de uma Entrada (em Bytes)
Plataforma	68
Divulgação	1048
Evento	647
Organizador_has_Evento	8
Organizador	516
Entidade	256
Participante	41
Local	264
Permite_Eentrada_Evento_Participante_Divulgacao	73

Tabela 9 - Espaço Esperado de uma Entrada de Cada Tabela.

Por cada entrada criada na tabela Evento esperamos criar 2 entradas na tabela Organizador_Has_Evento, 3 entradas na tabela Divulgação em média 50 entradas na tabela Permite_Eentrada_Evento_Participante_Divulgacao. Estimamos assim que a criação de um Evento irá resultar num espaço ocupado em tabelas de $2*8 + 1048*3 + 647 + 73*50$ bytes = 7.5 kbytes.

A empresa espera ajudar criação de cerca de 200 a 300 eventos no próximo ano o que irá representar um crescimento da base de dados em 1.5 Mbytes a 2.5 Mbytes.

As tabelas de Organizadores e Local não irão sofrer um crescimento significativo uma vez que a empresa tende a trabalhar regularmente com os mesmos Organizadores e com os mesmos lugares. Estima-se que num ano estas tabelas apresentam um aumento de de 5 a 10 entradas o que representa um aumento do tamanho da tabela de $5*(516+256)$ bytes = 3.9 kbytes a $10*(516+256)$ = 7.7 kbytes na tabela de Organizadores e de $5*(264+256)$ bytes = 2.6 kbytes a $10*(264+256)$ = 5.2 kbytes na tabela de locais.

Sempre que é adicionado um participante é criada uma entrada nas tabelas Entidade e Participante o que ocupa por volta de $256+41= 297$ bytes na base de dados.

No pior dos casos será preciso criar um novo participante por cada entrada criada na tabela Permite_Eentrada_Evento_Participante_Divulgacao, cerca $200 \times 50 = 10000$ a $300 \times 50 = 15000$ novas entradas o que iria ocupar $10000 \times 297 = 2.7 \text{ Mbytes}$ a $15000 \times 297 = 4.5 \text{ Mbytes}$.

O crescimento dos participantes não irá crescer a uma taxa um bocado mais baixa uma vez que a empresa espera que uma parte dos participantes vá a mais do que um evento.

Podemos assim estimar que a base de dados irá assim crescer no máximo, se as estimativas que nos foram dadas forem verdadeiras, por volta de $4.5 + 2.5 \text{ MBytes} = 7 \text{ Mbytes}$ por ano.

5.8. Definição e Caracterização das vistas de utilização em SQL

- **Vista de Participante**

Dados de participante que os organizadores poderão consultar.

```
CREATE VIEW viewParticipante AS
    SELECT Id, Nome, Genero, Email, Telemovel, DataDeNascimento
    FROM Participante AS P
    INNER JOIN Entidade AS E
    ON P.Entidade_Id=E.Id;
```

Figura 44 - Vista de Participante.

- **Vista de Geral de Eventos**

Avaliação económica do evento passado, os dados retornados são o nome do Evento, a data em que se realizou, a receita máxima que poderia ter sido obtido através da venda de bilhetes (correspondente ao número de bilhetes que poderiam ter sido vendidos vezes o preço base de cada bilhete), as receitas totais obtidas, resultado do preço que os participantes pagaram para entrar no evento, o custo total com divulgação, e o balanço, a diferença entre o custo e as receitas.

```

CREATE VIEW viewEvents AS

SELECT
    Nome,
    E.Data,
    E.NumeroMaximoDeParticipantes * E.Preco AS 'Receita Máxima Possível',
    SUM(B.Preco) AS 'Receitas Totais',
    SUM(Custo) AS 'Custo Total',
    SUM(B.Preco) - SUM(Custo) AS 'Balanço'

FROM Evento AS E

INNER JOIN Divulgacao AS D
    ON E.Id=D.Evento_Id

INNER JOIN PermiteEntrada_Evento_Participante_Divulgacao AS B
    ON E.Id=B.Evento_Id

WHERE B.Estado=1

GROUP BY E.Id;

```

Figura 45 - Vista de Geral de Eventos.

5.9. Definição e Caracterização dos Mecanismos de Segurança em SQL

Em qualquer projeto é necessário garantir que sejam cumpridos todos os requisitos de segurança necessários para assegurar a proteção dos dados. Assim é essencial que a base de dados seja ela própria o primeiro mecanismo de defesa contra qualquer tipo de ações mal-intencionadas.

Assim a primeira medida foi, por exemplo, a criação de vários utilizadores que irão interagir de forma regular com a base de dados com permissões segundo a seguinte tabela. Existirá ainda um administrador que terá todos os privilégios.

	Funcionários				Organizadores			
Tabela	S	I	U	D	S	I	U	D
Evento	x	x	x	x	x			
Organizador	x	x	x	x	x			
Plataforma	x	x	x	x	x			
Divulgação	x	x	x	x	x		x	
Local	x	x	x	x	x			
Participante	x	x	x	x				
Organizador_has_Evento	x	x	x	x	x			
Permite_Entrada_Evento_Participante_Divulgacao	x	x	x	x	x			

Tabela 10 - Permissões de Funcionários e Organizadores.

S - Select; I - Insert; U - Update; D - Delete

```

CREATE USER 'funcionario'@'localhost' IDENTIFIED BY 'password';

GRANT SELECT, INSERT, UPDATE, DELETE ON EventsWorkbench.* TO 'funcionario'@'localhost' ;

GRANT EXECUTE ON PROCEDURE FParticipanteEvento TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE FEventoEntreDatas TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE FEventoEmLocal TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE FParticipanteMaisGastaTotal TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE FParticipanteMaisGastaTipoEvento TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE FParticipanteMaisGastaOrg TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE FDivulgacaoEficazTipoBruto TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE FDivulgacaoEficazProporcao TO 'funcionario'@'localhost';

GRANT EXECUTE ON PROCEDURE addLocal TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE addOrganizador TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE addParticipante TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE addEvento TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE addParticipanteEventoDivulgacao TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE addClassificacao TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE ValidarDivulgacao TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE AlterarEstadoParticipacao TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE CriarPlataforma TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE CriarDivulgacao TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE addLocalComEntidadeExistente TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE addOrganizadorComEntidadeExistente TO 'funcionario'@'localhost';
GRANT EXECUTE ON PROCEDURE addParticipanteComEntidadeExistente TO 'funcionario'@'localhost';

GRANT SELECT ON viewEvents TO 'funcionario'@'localhost';

```

Figura 46 - Criação e atribuição de permissões de funcionário.

```

CREATE USER 'organizador'@'%' IDENTIFIED BY 'password';

GRANT SELECT
ON Evento
TO 'organizador'@'%';

GRANT SELECT
ON Organizador
TO 'organizador'@'%';

GRANT SELECT
ON Plataforma
TO 'organizador'@'%';

GRANT SELECT, UPDATE
ON Divulgacao
TO 'organizador'@'%';

GRANT SELECT
ON Local
TO 'organizador'@'%';

GRANT SELECT
ON Organizador_has_evento
TO 'organizador'@'%';

GRANT SELECT
ON PermiteEntrada_Evento_Participante_Divulgacao
TO 'organizador'@'%';

GRANT EXECUTE ON PROCEDURE OParticipantesDeEvento TO 'organizador'@'%';
GRANT EXECUTE ON PROCEDURE OPlataformasDeEvento TO 'organizador'@'%';
GRANT EXECUTE ON PROCEDURE OLocaisDeEventos TO 'organizador'@'%';
GRANT EXECUTE ON PROCEDURE OEventos TO 'organizador'@'%';
GRANT EXECUTE ON PROCEDURE ODivulgacao TO 'organizador'@'%';
GRANT EXECUTE ON PROCEDURE ODivulgacaoInfluencia TO 'organizador'@'%';

GRANT SELECT ON viewParticipante TO 'organizador'@'%';
GRANT SELECT ON viewEvents TO 'organizador'@'%';

```

Figura 47 - Criação e atribuição de permissões de organizador.

Outro exemplo de uma medida de segurança a ser utilizada é a criação regular de backups dos dados do sistema recorrendo ao comando *mysqldump*:

```
▲ ~/EventsWorkbench ls
▲ ~/EventsWorkbench mysqldump --user=root EventsWorkbench > "EventsWorkbench.backup.$(date +"%Y%m%d_%H%M%S").sql"
▲ ~/EventsWorkbench ls
EventsWorkbench.backup.20181126_004032.sql

▲ ~/EventsWorkbench head EventsWorkbench.backup.20181126_004032.sql
-- MySQL dump 10.13 Distrib 8.0.13, for macos10.14 (x86_64)
--
-- Host: localhost      Database: EventsWorkbench
-- -----
-- Server version     8.0.13

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
SET NAMES utf8mb4 ;

```

Figura 48 - Execução do comando *mysqldump*.

Este comando será agendado através de um *cron job* para ser executado todos os dias às 2:00:00 UTC o que irá permitir manter um registo seguro dos dados, e assegurará que a Events Workbench está prevenida para qualquer eventualidade em relação a estes dados.

```
* 2 * * * mysqldump --user=root EventsWorkbench > "EventsWorkbench.backup.$(date +"%Y%m%d_%H%M%S").sql"
```

Figura 49 - Execução do comando *mysqldump*.

5.10. Revisão do Sistema Implementado com o Utilizador

Após a conclusão da implementação da base de dados foi agendada uma reunião final com a Events Workbench de forma a validar a base de dados e possivelmente dar como concluído o desenvolvimento do sistema.

A revisão da implementação consistiu principalmente em 5 fases.

Na primeira fase mostramos que a base de dados conseguia responder às várias interrogações às quais nós nos tínhamos comprometido a poder responder com o cliente.

Na segunda fase demonstramos as várias operações de inserção de dados no sistema e como é que o sistema lidava com inserções incorretas de dados.

Na terceira fase mostramos ao cliente as estimativas de espaço ocupado da base de dados e da taxa de crescimento esperada.

Na quarta fase consistiu em explicar ao cliente os vários mecanismos de segurança implementados ao nível da base de dados.

A fase final consistiu numa discussão com o cliente sobre as várias dúvidas que a apresentação poderia ter deixado acerca da implementação.

Esclarecidas todas as dúvidas que poderiam existir sobre a implementação o cliente deu autorização para avançar para a instalação do sistema.

6. Sistema NoSQL: Neo4j

6.1. Justificação da utilização de um sistema NoSQL

Apesar de ser uma empresa muito recente, devido à forte aposta da Events Workbench na digitalização dos seus dados, em particular no desenvolvimento de uma base de dados relacional, que se tem mostrado de elevada qualidade e utilidade, a empresa tem vindo a reforçar a sua posição de mercado e, consequentemente, cada vez mais organizadores têm-na contactado para colaborar na organização dos seus eventos.

Esta nova situação tem trazido alguns problemas, com cada vez mais eventos, o número de consultas de informação tem crescido exponencialmente e, cada vez mais, a necessidade de usar um sistema de base de dados complementar, com maior escalabilidade, melhor “*performance*” e que seja especialmente dedicado à obtenção de respostas rápidas a “*queries*” (interrogações).

Assim surge a utilização de um sistema NoSQL, nomeadamente uma “*Graph Database*”, recorrendo ao motor Neo4j, focado em 3 aspectos essenciais: a “*Performance*”, principalmente em bases de dados com uma enorme quantidade de relacionamentos entre os dados, tal como é o caso da Events Workbench; a Flexibilidade: que permite que muito rapidamente este sistema de base de dados seja alterado para fazer face às necessidades dos organizadores e clientes, visto que poderá ser expandido para uso das próprias plataformas de divulgação externas; e a Agilidade: em grande parte devido também à sua flexibilidade, este sistema permite acompanhar de maneira subtil quaisquer possíveis alterações aos requisitos de negócio, permitindo que rapidamente a bases de dados NoSQL esteja a par das necessidades das empresas e permita fornecer aos utilizadores aquilo que eles precisam, ajudando a Events Workbench a faturar.

6.2. Identificação e descrição dos objetivos da base de dados, em termos de aplicações e de utilizadores

O grande objetivo da implementação deste sistema NoSQL é, sem dúvida, possibilitar a realização de interrogações que envolvem a junção de várias relações de uma forma extremamente rápida. Estas interrogações são correspondentes às interrogações que outrora tinham sido implementadas em SQL que, devido a necessidade de melhoria na qualidade do serviço dos organizadores, terão agora lugar em Neo4j onde será tirado proveito das funcionalidades deste sistema de travessias em grafos. Adicionalmente, alguns dos dados não serão normalizados, no sentido de base de dados relacionais, dado que estes não serão alvo de alterações até se repetir o processo de migração.

6.3. Identificação e explicação do tipo de questões (necessidades) que serão realizadas sobre o sistema de dados NoSQL

Para este sistema de dados e de acordo com os objetivos previamente descritos do mesmo, serão disponibilizados um subconjunto dos dados da base de dados relacional previamente apresentada. Nestes dados serão efetuadas interrogações que obtêm a listagem dos participantes que mais gastaram nos eventos de uma dada organização, a listagem do ganho por participante correspondente a uma divulgação de um dado tipo, a determinação de qual a divulgação que mais participantes atraiu para um dado evento, a determinação de quantos participantes uma dada divulgação influenciou e de forma semelhante a obtenção dos participantes de um dado evento.

6.4. Definição da estrutura base para o sistema de dados NoSQL que satisfaça os requisitos e as questões apresentadas anteriormente

Para satisfazer os requisitos para o sistema de dados NoSQL totalmente dedicado a consulta de informações de eventos por parte dos participantes sugere-se a seguinte estrutura:

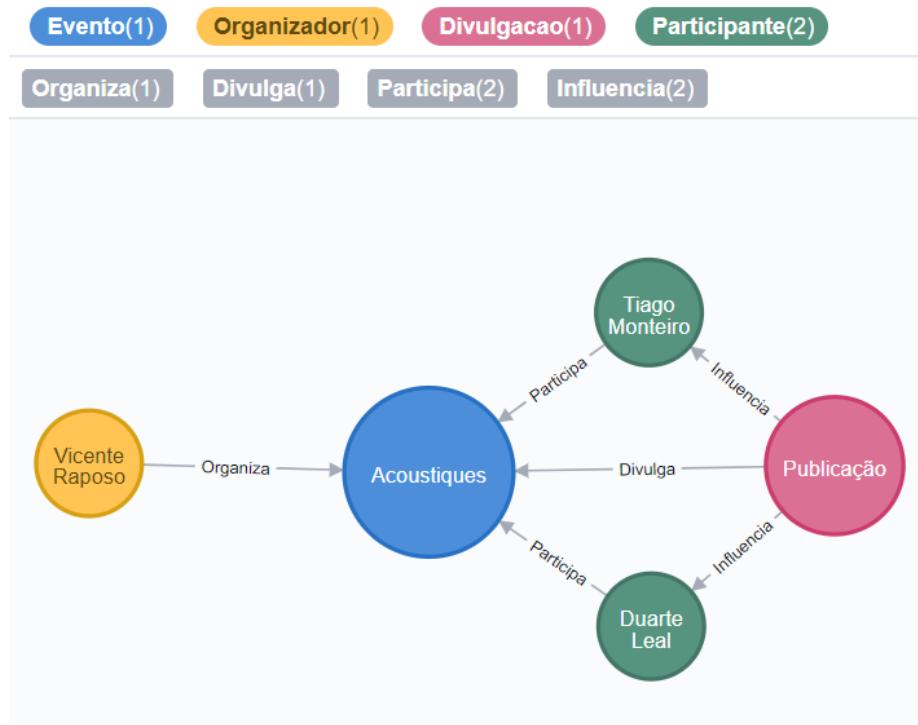


Figura 50 - Estrutura Neo4j.

Os nodos principais são os nodos azuis de “Evento” ao qual estarão relacionados todos os outros tipos de nodos (Organizador, Participante, Divulgação).

A cada evento estarão ligados pelo relacionamento “ORGANIZADO” desde 0 até N organizadores, 0 até N divulgações através do relacionamento “DIVULGA” e 0 até N participantes através de “PARTICIPA”. Por fim cada participante poderá estar ligado a 0 a N divulgações através de “INFLUENCIA”.

Cada organizador e participante poderão estar ligados a vários eventos.

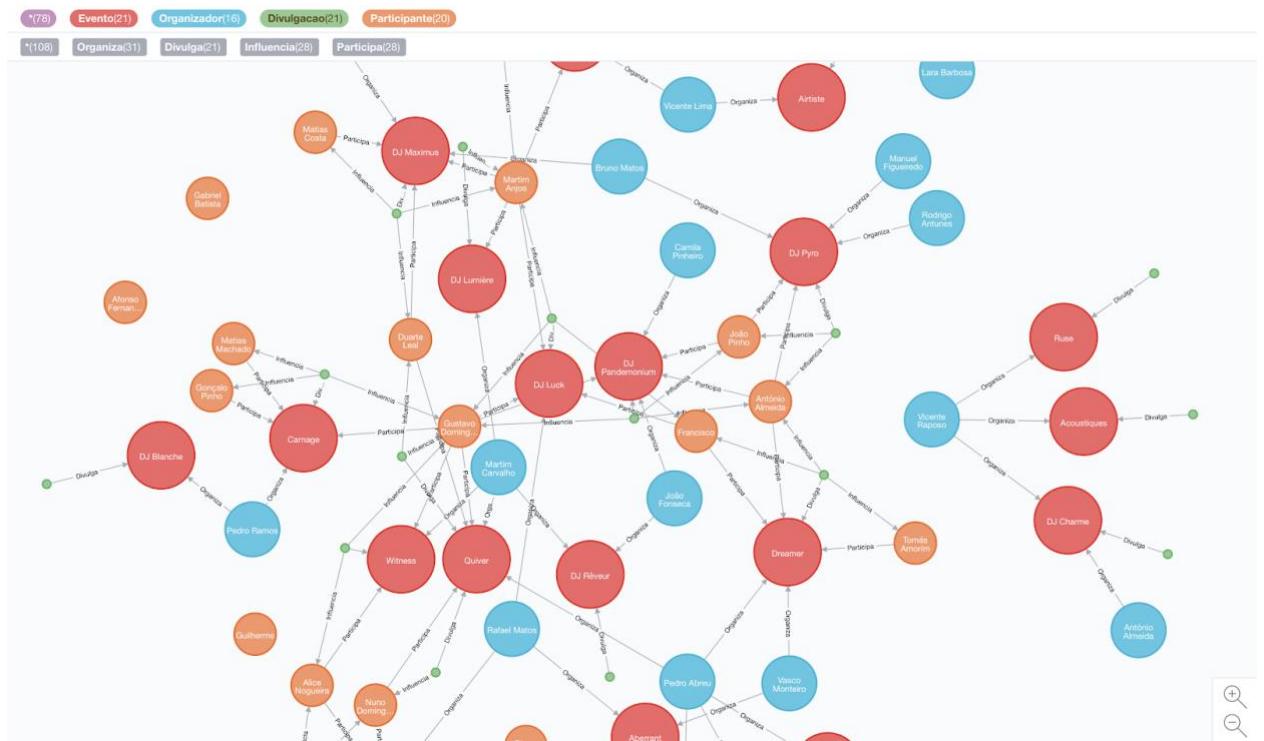


Figura 51 - Ilustração da base de dados NoSQL criada em Neo4j.

6.5. Identificação dos objetos de dados no sistema SQL que serão utilizados para alimentar o novo sistema

Para a base de dados serão utilizados objetos de dados de 7 tabelas do sistema relacional: Evento, Organizador, Divulgacao, Participante, Entidade, Organizador_has_Evento e PermiteEntrada_Evento_Participante_Divulgacao.

Cada um dos nodos representa uma tabela no modelo relacional. Os nodos “Evento” são gerados a partir da tabela Evento, os “Organizador” através da Organizador e Entidade, os “Divulgacao” através de Divulgacao e os “Participante” através de Participante e Entidade.

Os relacionamentos “Organiza” são gerados a partir de Organizador_has_Evento, os “Participa” e “Influencia” são gerados a partir de PermiteEntrada_Evento_Participante_Divulgacao e o “Divulga” é gerado a partir de Divulgação.

6.6. Mapeamento do processo de migração de dados, descrevendo o processo de conversão dos vários objetos de dados

O processo de migração de dados terá duas fases: o carregamento dos nodos e o carregamento dos relacionamentos entre esses nodos.

O **carregamento dos nodos** será feito da seguinte forma:

- As entradas da tabela Evento foram convertidas para nodos do tipo Evento com os atributos id, preço e nome;
- As entradas resultantes da junção de Entidade com Organizador foram convertidas para nodos do tipo Organizador com os atributos id, nome e email;
- As entradas de Divulgação foram convertidas para nodos com os atributos id, preço e tipo;
- As entradas resultantes da junção de Entidade foram convertidas para nodos com os atributos id, nome, e-mail, telemóvel, género, NIF e data de nascimento.

O **carregamento dos relacionamentos** será feito da seguinte forma:

- A tabela ParticipaEntrada_Evento_Participante_Divulgacao será transformada em dois tipos de relacionamento. O “Participa” que será criado com os ids do Participante e do Evento com direção de Participante para Evento e que irá guardar o preço que o cliente pagou. O “Influencia” que será criado caso o Divulgacao_id não seja nulo com os ids do Participante e da Divulgação com direção de Divulgação para Participante;
- A tabela Organizador_has_Evento será utilizada para criar os relacionamentos “Organiza” com direção do Organizador para o Evento;
- E, por fim, os nodos de Divulgacao terão uma ligação para o evento que divulgam que será criada através da chave estrangeira Evento_Id com direção Divulgacao para Evento.

6.7. Explicação do processo de migração de dados, explicando de forma detalhada as suas principais etapas - extração, transformação e carregamento

Dado que a nova base de dados não seria alvo de alterações futuras, para além da periódica migração, não foram consideradas questões de normalização de dados. Por esta razão, os nodos do tipo Participante e Organizador contém a junção da informação contida em Organizador ou Participante e também da correspondente Entidade. Adicionalmente, todas as

relações base que foram migradas para a nova base de dados foram alvo de projeções, dado que nem todos os atributos constantes nestas eram necessários para as interrogações NoSQL.

Os dados foram obtidos realizando interrogações e criando cursores nas tabelas resultantes. Os tipos vértices criados em Neo4j foram Participante, Organizador, Evento e Divulgação. As arestas foram as correspondentes aos relacionamentos entre estas entidades na base de dados relacional, mais precisamente Divulga, Influencia, Organiza e Participa.

O processo de conversão é feito aos bocados. É sempre feita a extração parcial das entradas, a sua transformação, e, por fim, o seu carregamento na base de dados. A extração é parcial para ultrapassar as limitações de memória que o programa pode ter, desta forma o programa funciona para bases de dados de “qualquer” tamanho.

6.8. Apresentação e descrição da implementação do processo de migração de dados

Para a migração da base de dados foi criada uma solução de software em java. O programa desenvolvido usa duas implementações da interface *jdbc*, que é uma interface que visa generalizar o processo de ligação a uma de base dados em java, feitas pelas equipas de desenvolvimento do MySQL e do Neo4j.

O processo de extração, transformação e carregamento de dados foi realizado concorrentemente por forma a mitigar o tempo de espera associado à comunicação com as duas bases de dados. Com objetivo de desenvolver um sistema de migração de *mysql* para *neo4j* modular e funcional criamos duas hierarquias de classes.

A hierarquia que instancia a extração dos dados na base de dados relacional não só disponibiliza métodos para a extração dos tuplos relativos às entidades como também para os tuplos correspondentes aos relacionamentos. A hierarquia feita para a adição de novos vértices e arestas em Neo4j é ligeiramente mais sofisticada, por esta razão é apresentada na Fig. 52 o diagrama de classes relativo a esta.

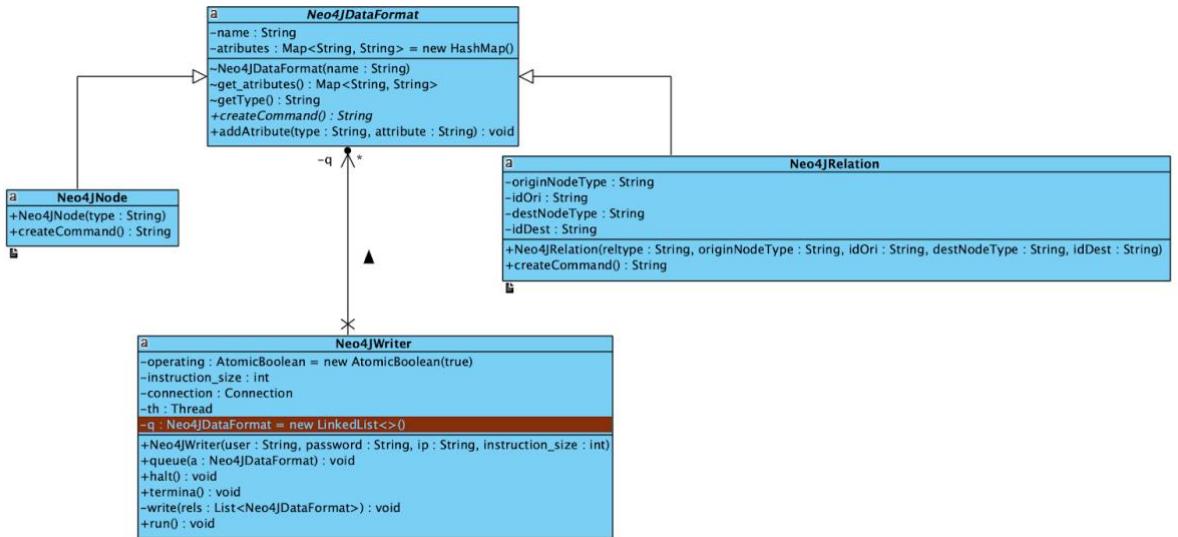


Figura 52 - Diagrama de Classes Neo4j.

A classe *Neo4jDataFormat* é uma classe abstrata que tem como método abstrato *createCommand*. Este método deve devolver a *String* correspondente à declaração em *cypher* que cria o respetivo objeto em Neo4j. As classes *Neo4jNode* e *Neo4jRelation* são as classes que representam, respetivamente, os vértices e arestas em Neo4j.

Para efetuar a ligação à base de dados em Neo4j foi criada a classe *Neo4jWrite* que, para além dessa funcionalidade, contém *run*, um método que é executado concorrentemente quando são extraídos e transformados os dados provenientes de MySQL. Este método consecutivamente converte conjuntos de *Neo4jDataFormat*, com um tamanho pré-definido, para *String* através de *createCommand* e envia-os para Neo4j.

Para correr o programa é necessário passar-lhe os argumentos na seguinte ordem:
`<sqluser> <sqlpassword> <sqlip:sqlport> <neo4juser> <neo4jpassword>`
`<neo4jip:sqlport>.` Exemplo: "root 123456 localhost:3306 neo4j 123456 localhost:7687".

6.9. Apresentação da forma como as questões identificadas anteriormente podem ser satisfeitas com o novo sistema, utilizando a linguagem de interrogação do sistema NoSQL

Qualquer uma das questões identificadas anteriormente pode facilmente ser traduzida para Neo4j. Decidimos demonstrar apenas as que mais sobrecarregam a base de dados, ou seja, as que recorrem a *inner joins*, especialmente quando aplicados à tabela que possui o relacionamento triplo, visto que estas são as travessias que justificam a utilização do sistema NoSQL. As *queries* que fazem somente leituras dos atributos das tabelas não sobrecarregam a base de dados MySQL pelo que a performance das mesmas já é satisfatória, mesmo perante um enorme volume de dados.

1. Consultar o nome e a data de nascimento dos participantes de um dado evento.

```
MATCH (p:Participante) - [:Participa] ->
(e:Evento{nome:<eventonome>})
RETURN p.nome, p.DataDeNascimento;
```

2. Consultar quantos participantes foram influenciados por uma dada divulgação.

```
MATCH (d:Divulgacao{id:<divid>}) - [:Influencia] ->
(p:Participante) with d, count(p) as numero_participantes
RETURN numero_participantes;
```

3. Determinar qual é o Id da divulgação que mais participantes atraiu para um determinado evento.

```
MATCH (d:Divulgacao) - [:Influencia] -> (p:Participante)
MATCH (d) - [:Divulga] -> (e:Evento{nome:<eventonome>})
with d, count(p) as numero_participantes
RETURN d.id
ORDER BY numero_participantes DESC
LIMIT 1;
```

4. Um funcionário da empresa poderá consultar, para um número indicado de participantes, os que mais dinheiro gastam em eventos por organizador.

```
MATCH (o:Organizador)
WHERE o.id = <orgid>
MATCH (o)-[:Organiza]->(e:Evento)
MATCH (p:Participante)-[pk:Participa]->(e)
RETURN p, sum(pk.preco)
ORDER BY sum(pk.preco) DESC
LIMIT <n>;
```

- 5. Para cada tipo de evento, um funcionário, deverá poder verificar qual o tipo de divulgação mais eficaz em termos de custo por participante de divulgação.**

```
MATCH (d:Divulgacao)
WHERE d.tipo = <divtipo>
MATCH (d)-[:Influencia]->(p:Participante)
MATCH (d)-[:Divulga]->(c:Evento)
MATCH (p)-[pk:Participa]->(c)
RETURN d, (sum(pk.preco)-d.preco)/count(p);
```

7. Conclusões e Trabalho Futuro

A Base de dados desenvolvida vai, incontestavelmente, impulsionar a ascensão da Events Workbench. Pois, pela primeira vez, esta tirará proveito de uma tecnologia de gestão, exploração e armazenamento de dados à altura da sua estimulante missão empresarial. Podendo desta forma estar adequadamente preparada para os desafios operacionais que enfrentará no caminho a alcançar os seus objetivos.

Assim, assumindo um papel central e inovador, este software representa uma nova e melhorada estratégia de desenvolvimento de infraestrutura de sistemas de informação baseada nos dados. Foi então essencial garantir, à partida, a consistência, integridade, segurança, concorrência, acessibilidade e facilidade de manutenção da Base de dados.

O cumprimento integral destas necessidades é o resultado de um longo processo de estudo onde foi aplicado o método de concepção de Base de dados apresentado e discutido no livro *Database Systems, A Practical Approach to Design, Implementation, and Management*. Este método é apresentado como antídoto ao falhanço no desenvolvimento de software, mais precisamente de sistemas de armazenamento de dados, por meio da definição de um ciclo de vida para o desenvolvimento destes sistemas. Este ciclo visa decompor todo o caso de estudo em partes digeríveis, tendo como prioridade a totalidade dos requisitos, por forma a ir de encontro aos critérios de sucesso estabelecido pelas partes interessadas.

A aplicação deste método de desenvolvimento permite que, a qualquer momento, sejam implementadas novas funcionalidades de um modo simples e seguro, assegurando que uma expansão futura desta Base de Dados que incorpore, por exemplo, modos de guardar os dados referentes ao tráfego, a conteúdo criado e hiperligações acedidas nas plataformas pelos participantes não comprometam a integridade e a normalização da Base de dados. Estes são alguns dos possíveis requisitos para futuro visto que estes dados permitiriam a possíveis futuras camadas aplicacionais extrair ainda mais informação para análise psico-demográfica podendo proporcionar ainda mais ferramentas à Events Workbench.

Dada a elegância do método adotado, compreendida nos esforços da Events Workbench de adaptação à crescente utilização dos seus serviços, a migração, resultante da execução da solução de software desenvolvida, foi suave e não desencadeou complicações.

A sua existência introduziu novas possibilidades de escalabilidade nos sistemas de informação da Events Workbench sem, simultaneamente, introduzir as limitações caracteristicamente associadas, pois a base de dados NoSQL criada é unicamente complementar.

A adição desta alternativa base de dados, apresenta um novo conjunto de possibilidades aos desenvolvedores da Events Workbench, pois agora, poderão muito mais eficientemente interrogar uma versão, não necessariamente atualizada, da base de dados relacional.

Posto isto, ao longo deste relatório, foram apresentadas todas as condições do sistema em que a base de dados se iria inserir, as necessidades do mesmo e métodos utilizados. Primeiramente, foi abordado contexto que envolve a Events Workbench, detalhamos e analisamos os requisitos referentes à base de dados que esta pretende, apresentamos uma modelação para esta desligada de implicações físicas, de seguida, transformamos esse modelo num lógico, implementamos este num sistema de gestão de base de dados relacional e, por fim, migramos parte da base de dados desenvolvida para Neo4j, um sistema de gestão de base de dados NoSQL, dedicado ao processamento eficiente de interrogações.

Referências

Connolly, T. and Begg, C. (2005). Database Systems, A Practical Approach to Design, Implementation, and Management. 4th ed. Addison-Wesley.

Mysql. (2018). MySql. [Online]. [26 November 2018]. Available from: <https://dev.mysql.com/doc/refman/8.0/en/>.

Lista de Siglas e Acrónimos

BD	Base de Dados
NIF	Número de Identificação Fiscal
SQL	<i>Structured Query Language</i>
EPD	Evento - Participante - Divulgação
ER	<i>Entity–Relationship</i>
EER	<i>Enhanced Entity–Relationship</i>
O_H_E	Organizador_Has_Evento
PE_E_P_D	PermiteEntrada_Evento_Participante_Divulgacao
ACID	Atomicidade, Consistência, Isolamento e Durabilidade
JBDC	<i>Java Database Connectivity</i>

Anexos

I. Anexo 1 – *Script de Inicialização da Base de Dados*

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZE
RO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

-----
-- Schema EventsWorkbench
-----

-----
-- Schema EventsWorkbench
-----

CREATE SCHEMA IF NOT EXISTS `EventsWorkbench` DEFAULT CHARACTER SET
utf8 ;
USE `EventsWorkbench` ;

-----
-- Table `EventsWorkbench`.`Plataforma`
-----

CREATE TABLE IF NOT EXISTS `EventsWorkbench`.`Plataforma` (
`Id` INT NOT NULL,
`Nome` VARCHAR(64) NOT NULL,
PRIMARY KEY (`Id`))
ENGINE = InnoDB;

-----
-- Table `EventsWorkbench`.`Entidade`
-----
```

```

CREATE TABLE IF NOT EXISTS `EventsWorkbench`.`Entidade` (
  `Id` INT NOT NULL,
  `Endereco` VARCHAR(128) NOT NULL,
  `Nome` VARCHAR(64) NOT NULL,
  `Email` VARCHAR(45) CHARACTER SET 'utf8' NOT NULL,
  `Telemovel` VARCHAR(15) NOT NULL,
  PRIMARY KEY (`Id`),
  UNIQUE INDEX `Email_UNIQUE` (`Email` ASC) VISIBLE,
  UNIQUE INDEX `Telemovel_UNIQUE` (`Telemovel` ASC) VISIBLE)
ENGINE = InnoDB;

```

```

-- -----
-- Table `EventsWorkbench`.`Local`

CREATE TABLE IF NOT EXISTS `EventsWorkbench`.`Local` (
  `Entidade_Id` INT NOT NULL,
  `Lotacao` INT NOT NULL,
  `Descricao` TEXT(256) NOT NULL,
  `Tipo` ENUM('Bares', 'Casas noturnas', 'Casa de espetáculos',
  'Hotelaria', 'Restaurantes', 'Centros Culturais', 'Multiusos',
  'Quintas', 'Outros') NOT NULL,
  PRIMARY KEY (`Entidade_Id`),
  CONSTRAINT `fk_Local_Entidade1`
    FOREIGN KEY (`Entidade_Id`)
    REFERENCES `EventsWorkbench`.`Entidade` (`Id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

```

-- -----
-- Table `EventsWorkbench`.`Evento`

CREATE TABLE IF NOT EXISTS `EventsWorkbench`.`Evento` (
  `Id` INT NOT NULL,
  `Nome` VARCHAR(64) NOT NULL,
  `Descricao` TEXT(512) NOT NULL,
  `Tipo` ENUM('Discussão, informação e formação', 'Religiosos e
  comunitários', 'Académicos', 'Políticos e de protocolo', 'Culturais',
```

```

de lazer, desportivos e musicais', 'Culinária', 'Empresariais',
'Outros') NOT NULL,
`Data` DATETIME NOT NULL,
`Duracao` TIME NOT NULL,
`Preco` DECIMAL(7,2) NOT NULL DEFAULT 0.0,
`DataInicioRegistoParticipantes` DATETIME NOT NULL,
`DataFimRegistoParticipantes` DATETIME NOT NULL,
`Classificacao` DECIMAL(4,2) NULL,
`NumeroMaximoDeParticipantes` INT NOT NULL,
`Local_Entidade_Id` INT NOT NULL,
PRIMARY KEY (`Id`),
INDEX `fk_Evento_Local1_idx` (`Local_Entidade_Id` ASC) VISIBLE,
CONSTRAINT `fk_Evento_Local1`
    FOREIGN KEY (`Local_Entidade_Id`)
    REFERENCES `EventsWorkbench`.`Local` (`Entidade_Id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

```

```

-- -----
-- Table `EventsWorkbench`.`Divulgacao`

CREATE TABLE IF NOT EXISTS `EventsWorkbench`.`Divulgacao` (
`Id` INT NOT NULL,
`Conteudo` TEXT(1024) NOT NULL,
`Tipo` ENUM("Audiovisual", "Áudio", "Publicação", "Cartaz", "Email",
"Brochuras", "Aresentações", "Promoções", "Panfletos", "Outros") NOT
NULL,
`Custo` DECIMAL(12,2) NOT NULL DEFAULT 0.0,
`Validade` ENUM('Pendente', 'Válida', 'Inválida') CHARACTER SET
'utf8' NOT NULL DEFAULT 'Pendente',
`DataInicio` DATETIME NOT NULL,
`DataFim` DATETIME NOT NULL,
`Plataforma_Id` INT NOT NULL,
`Evento_Id` INT NOT NULL,
PRIMARY KEY (`Id`),
INDEX `fk_Divulgação_Plataforma_idx` (`Plataforma_Id` ASC) VISIBLE,
INDEX `fk_Divulgação_Evento_idx` (`Evento_Id` ASC) VISIBLE,
CONSTRAINT `fk_Divulgação_Plataforma`
    FOREIGN KEY (`Plataforma_Id`)

```

```

    REFERENCES `EventsWorkbench`.`Plataforma` (`Id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
CONSTRAINT `fk_Divulgação_Evento1`
FOREIGN KEY (`Evento_Id`)
REFERENCES `EventsWorkbench`.`Evento` (`Id`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- -----
-- Table `EventsWorkbench`.`Participante`

CREATE TABLE IF NOT EXISTS `EventsWorkbench`.`Participante` (
`DataDeNascimento` DATE NOT NULL,
`Genero` ENUM('Feminino', 'Masculino') NOT NULL,
`Nif` INT NOT NULL,
`Entidade_Id` INT NOT NULL,
PRIMARY KEY (`Entidade_Id`),
UNIQUE INDEX `Nif_UNIQUE` (`Nif` ASC) VISIBLE,
CONSTRAINT `fk_Participante_Entidade1`
FOREIGN KEY (`Entidade_Id`)
REFERENCES `EventsWorkbench`.`Entidade` (`Id`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- -----
-- Table
`EventsWorkbench`.`PermiteEntrada_Evento_Participante_Divulgacao`

CREATE TABLE IF NOT EXISTS
`EventsWorkbench`.`PermiteEntrada_Evento_Participante_Divulgacao` (
`Evento_Id` INT NOT NULL,
`Participante_Entidade_Id` INT NOT NULL,
`Divulgacao_Id` INT NULL,
`Classificacao` INT NULL,
`Estado` ENUM('Válido', 'Cancelado', 'Reservado') NOT NULL DEFAULT
'Válido',

```

```

`Preco` DECIMAL(7,2) NOT NULL DEFAULT 0.0,
`Lugar` VARCHAR(32) NOT NULL,
PRIMARY KEY (`Participante_Entidade_Id`, `Evento_Id`),
INDEX           `fk_Evento_has_Participante_Participantel_idx`(`Participante_Entidade_Id` ASC) VISIBLE,
INDEX   `fk_Evento_has_Participante_Eventol_idx`(`Evento_Id` ASC)
VISIBLE,
INDEX  `fk_Evento_has_Participante_Divulgaçao1_idx`(`Divulgacao_Id` ASC) VISIBLE,
CONSTRAINT `fk_Evento_has_Participante_Evento1`
FOREIGN KEY (`Evento_Id`)
REFERENCES `EventsWorkbench`.`Evento`(`Id`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `fk_Evento_has_Participante_Participantel`
FOREIGN KEY (`Participante_Entidade_Id`)
REFERENCES `EventsWorkbench`.`Participante`(`Entidade_Id`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `fk_Evento_has_Participante_Divulgaçao1`
FOREIGN KEY (`Divulgacao_Id`)
REFERENCES `EventsWorkbench`.`Divulgacao`(`Id`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

```

```

-- -----
-- Table `EventsWorkbench`.`Organizador`

CREATE TABLE IF NOT EXISTS `EventsWorkbench`.`Organizador` (
`Descricao` TEXT(512) NOT NULL,
`Entidade_Id` INT NOT NULL,
PRIMARY KEY (`Entidade_Id`),
CONSTRAINT `fk_Organizador_Entidade1`
FOREIGN KEY (`Entidade_Id`)
REFERENCES `EventsWorkbench`.`Entidade`(`Id`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

```

-- -----
-- Table `EventsWorkbench`.`Organizador_has_Evento`
-- -----
CREATE TABLE IF NOT EXISTS `EventsWorkbench`.`Organizador_has_Evento`
(
    `Organizador_Entidade_Id` INT NOT NULL,
    `Evento_Id` INT NOT NULL,
    PRIMARY KEY (`Organizador_Entidade_Id`, `Evento_Id`),
    INDEX `fk_Organizador_has_Evento_Evento1_idx`(`Evento_Id` ASC)
VISIBLE,
    INDEX `fk_Organizador_has_Evento_Organizador1_idx`(`Organizador_Entidade_Id` ASC) VISIBLE,
    CONSTRAINT `fk_Organizador_has_Evento_Organizador1`
        FOREIGN KEY (`Organizador_Entidade_Id`)
        REFERENCES `EventsWorkbench`.`Organizador`(`Entidade_Id`)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION,
    CONSTRAINT `fk_Organizador_has_Evento_Evento1`
        FOREIGN KEY (`Evento_Id`)
        REFERENCES `EventsWorkbench`.`Evento`(`Id`)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION)
ENGINE = InnoDB;

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;

```

II. Anexo 2 – Script de Povoamento Inicial

```
USE EventsWorkbench;

INSERT INTO Entidade (Id,Endereco, Nome,Email,Telemovel)
VALUES
(1,'Braga','Rodrigo Gonçalves','Rodrigo_Gonçalves@mail.pt','939898677'),
(2,'Portimão','Gonçalo Pinho','Gonçalo_Pinho@mail.pt','938253925'),
(3,'Funchal','Martim Anjos','Martim_Anjos@mail.pt','938564942'),
(4,'Porto','Francisco Simões','Francisco_Simões@mail.pt','937841037'),
(5,'Funchal','Duarte Leal','Duarte_Leal@mail.pt','931678810'),
(6,'Faro','Tomás Amorim','Tomás_Amorim@mail.pt','932249721'),
(7,'Portimão','Gabriel Freitas','Gabriel_Freitas@mail.pt','936066676'),
(8,'Amadora','Gabriel Batista','Gabriel_Batista@mail.pt','936731590'),
(9,'Guimarães','Duarte Rodrigues','Duarte_Rodrigues@mail.pt','938266154'),
(10,'Odivelas','Afonso Fernandes','Afonso_Fernandes@mail.pt','938119672'),
(11,'Setúbal','Tiago Monteiro','Tiago_Monteiro@mail.pt','931460285'),
(12,'Agualva-Cacém','Alice Nogueira','Alice_Nogueira@mail.pt','934254312'),
(13,'Amadora','Guilherme Moura','Guilherme_Moura@mail.pt','930019252'),
(14,'Faro','Matias Machado','Matias_Machado@mail.pt','936325104'),
(15,'Porto','Nuno Domingues','Nuno_Domingues@mail.pt','938706919'),
(16,'Braga','Matias Costa','Matias_Costa@mail.pt','939480483'),
```

(17, 'Almada', 'Gustavo
 Domingues', 'Gustavo_Domingues@mail.pt', '936217069'),
 (18, 'Funchal', 'Diogo
 Soares', 'Diogo_Soares@mail.pt', '938893867'),
 (19, 'Rio Tinto', 'João Pinho', 'João_Pinho@mail.pt', '937894606'),
 (20, 'Amadora', 'António
 Almeida', 'António_Almeida@mail.pt', '931686372'),
 (21, 'Agualva-Cacém', 'Pedro
 Abreu', 'Pedro_Abreu@mail.pt', '930417425'),
 (22, 'Funchal', 'João
 Fonseca', 'João_Fonseca@mail.pt', '934495742'),
 (23, 'Faro', 'Vasco
 Monteiro', 'Vasco_Monteiro@mail.pt', '933949413'),
 (24, 'Rio
 Ramos', 'Pedro_Ramos@mail.pt', '937709443'),
 (25, 'Porto', 'Rodrigo
 Antunes', 'Rodrigo_Antunes@mail.pt', '931360166'),
 (26, 'Viseu', 'Martim
 Carvalho', 'Martim_Carvalho@mail.pt', '939297741'),
 (27, 'Viseu', 'Vicente
 Raposo', 'Vicente_Raposo@mail.pt', '936236486'),
 (28, 'Funchal', 'Manuel
 Neves', 'Manuel_Neves@mail.pt', '939298188'),
 (29, 'Lisboa', 'Lara Barbosa', 'Lara_Barbosa@mail.pt', '931480990'),
 (30, 'Barreiro', 'Manuel
 Figueiredo', 'Manuel_Figueiredo@mail.pt', '939706401'),
 (31, 'Faro', 'Rafael Matos', 'Rafael_Matos@mail.pt', '933073763'),
 (32, 'Porto', 'Bruno Matos', 'Bruno_Matos@mail.pt', '931617835'),
 (33, 'Vila Nova de Gaia', 'Camila
 Pinheiro', 'Camila_Pinheiro@mail.pt', '930756374'),
 (34, 'Faro', 'Dinis Matias', 'Dinis_Matias@mail.pt', '936271712'),
 (35, 'Vila Nova de Gaia', 'Vicente
 Lima', 'Vicente_Lima@mail.pt', '937310040'),
 (36, 'Leiria', 'Jumpstart
 Bistro', 'jumpstart_bistro@mail.pt', '933381367'),
 (37, 'Évora', 'Wakey Wakey
 Shop', 'wakey_wakey_coffee_shop@mail.pt', '935929449'),
 (38, 'Amadora', 'Thinking
 Joint', 'thinking_cup_joint@mail.pt', '939643171'),
 (39, 'Rio Tinto', 'Bean
 Cafeteria', 'bean_bag_cafeteria@mail.pt', '937240500'),

(40, 'Portimão', 'Big	Mugs
Coffee', 'big_mugs_coffee@mail.pt', '930112602'),	
(41, 'Barreiro', 'Tranquil	Chocolate
Cafeteria', 'tranquil_chocolate_cafeteria@mail.pt', '937048652'),	
(42, 'Faro', 'Unique	Eden
Cafe', 'unique_eden_cafe@mail.pt', '939899812'),	
(43, 'Lisboa', 'Havana	Beach
Diner', 'havana_beach_diner@mail.pt', '932606694'),	
(44, 'Portimão', 'Gentle	Breeze
Cafe', 'gentle_breeze_cafe@mail.pt', '936736945'),	
(45, 'Portimão', 'Peaceful	Harvest
Shop', 'peaceful_harvest_coffee_shop@mail.pt', '937647104'),	Coffee
(46, 'Rio	Tinto', 'Symphony
Plaza', 'symphony_plaza@mail.pt', '937777053'),	
(47, 'Odivelas', 'Marvel	
Plaza', 'marvel_plaza@mail.pt', '931936164'),	
(48, 'Queluz', 'Market	
Plaza', 'market_plaza@mail.pt', '938865840'),	
(49, 'Rio	Tinto', 'Grand
Plaza', 'grand_palace_plaza@mail.pt', '939517860'),	Palace
(50, 'Rio	Tinto', 'Conquest
Square', 'conquest_square@mail.pt', '931696920'),	
(51, 'Portimão', 'Sensation	
Plaza', 'sensation_plaza@mail.pt', '939038301'),	
(52, 'Funchal', 'Freedom	
Plaza', 'freedom_plaza@mail.pt', '935428399'),	
(53, 'Odivelas', 'Vineyard	
Square', 'vineyard_square@mail.pt', '937058934'),	
(54, 'Funchal', 'Augury	
Plaza', 'augury_plaza@mail.pt', '934098233'),	
(55, 'Leiria', 'Commemoration	
Square', 'commemoration_square@mail.pt', '930315040'),	
(56, 'Amadora', 'Neptune	
Shore', 'neptune_shore@mail.pt', '932075300'),	
(57, 'Funchal', 'Enchanted	
Bank', 'enchanted_bank@mail.pt', '935354837'),	
(58, 'Rio Tinto', 'Red Sands', 'red_sands@mail.pt', '938980807'),	
(59, 'Funchal', 'The	
Hummingbird', 'the_hummingbird@mail.pt', '933415322'),	
(60, 'Leiria', 'The	Distant
Strand', 'the_distant_strand@mail.pt', '931440805'),	

```

(61,'Lisboa','Exisle Bank','exisle_bank@mail.pt','935961919'),
(62,'Leiria','Middlesleche
Strand','middlesleche_strand@mail.pt','933516585'),
(63,'Barreiro','Bainmouth
Sands','bainmouth_sands@mail.pt','938250139'),
(64,'Almada','Munrial
Edge','munrial_edge@mail.pt','930278633'),
(65,'Viseu','Coatidon
Strand','coatidon_strand@mail.pt','935439954');

```

```

INSERT INTO Local (Entidade_id,Descricao,Tipo,Lotacao)
VALUES      (36,'Jumpstart Bistro',1,55),
(37,'Wakey Wakey Coffee Shop',1,91),
(38,'Thinking Cup Joint',1,57),
(39,'Bean Bag Cafeteria',1,78),
(40,'Big Mugs Coffee',1,97),
(41,'Tranquil Chocolate Cafeteria',1,66),
(42,'Unique Eden Cafe',1,88),
(43,'Havana Beach Diner',1,54),
(44,'Gentle Breeze Cafe',1,58),
(45,'Peaceful Harvest Coffee Shop',1,91),
(46,'Symphony Plaza',1,72),
(47,'Marvel Plaza',1,82),
(48,'Market Plaza',1,98),
(49,'Grand Palace Plaza',1,100),
(50,'Conquest Square',1,73),
(51,'Sensation Plaza',1,75),
(52,'Freedom Plaza',1,63),
(53,'Vineyard Square',1,54),
(54,'Augury Plaza',1,79),
(55,'Commemoration Square',1,68),
(56,'Neptune Shore',1,64),
(57,'Enchanted Bank',1,82),
(58,'Red Sands',1,68),
(59,'The Hummingbird',1,83),
(60,'The Distant Strand',1,90),
(61,'Exisle Bank',1,89),
(62,'Middlesleche Strand',1,55),
(63,'Bainmouth Sands',1,81),
(64,'Munrial Edge',1,95),

```

```

(65,'Coatidon Strand',1,73);

INSERT INTO Organizador (Entidade_id, Descricao)
VALUES      (20,'António Almeida'),
            (21,'Pedro Abreu'),
            (22,'João Fonseca'),
            (23,'Vasco Monteiro'),
            (24,'Pedro Ramos'),
            (25,'Rodrigo Antunes'),
            (26,'Martim Carvalho'),
            (27,'Vicente Raposo'),
            (28,'Manuel Neves'),
            (29,'Lara Barbosa'),
            (30,'Manuel Figueiredo'),
            (31,'Rafael Matos'),
            (32,'Bruno Matos'),
            (33,'Camila Pinheiro'),
            (34,'Dinis Matias'),
            (35,'Vicente Lima');

INSERT INTO Participante (Entidade_id,nif,DataDeNascimento,Genero)
VALUES (1,253572677,'1991-05-01',1),
        (2,233703579,'1995-11-17',1),
        (3,207767246,'1985-11-16',1),
        (4,224432618,'1998-04-20',1),
        (5,233736164,'1980-08-07',1),
        (6,261208828,'1997-03-19',1),
        (7,291049787,'1996-12-17',1),
        (8,264410652,'1988-06-09',1),
        (9,205293639,'1985-01-06',1),
        (10,230597752,'1984-05-14',1),
        (11,297766014,'1985-08-05',1),
        (12,273758048,'1980-11-12',2),
        (13,252152474,'1989-09-12',1),
        (14,281791621,'1985-04-09',1),
        (15,267106516,'1987-10-17',1),
        (16,209461642,'1983-05-05',1),
        (17,280648047,'1992-12-07',1),
        (18,298994994,'1989-02-19',1),
        (19,223752496,'1984-01-18',1),
        (20,208354015,'1981-12-05',1);

```

```

INSERT INTO Plataforma (id,nome)
VALUES      (1,'Facebook'),
            (2,'Intagram'),
            (3,'Twitter'),
            (4,'GoogleAds'),
            (5,'Youtube');

INSERT INTO Evento
(Id,Nome,Descricao,Tipo,data,DataInicioRegistroParticipantes,DataFimRegistroParticipantes,Duracao,Preco,NumeroMaximoDeParticipantes,Local_Entidade_Id)
VALUES      (1,'Quiver','Concerto do Quiver',5,'2017-08-23','2017-08-10','2017-08-23',CURRENT_TIME(),29.99,39,36),
            (2,'DJ Maximus','Concerto do DJ Maximus',5,'2017-06-09','2017-05-26','2017-06-08',CURRENT_TIME(),35.99,38,37),
            (3,'Dreamer','Concerto do Dreamer',5,'2017-10-08','2017-09-24','2017-10-07',CURRENT_TIME(),36.99,39,38),
            (4,'DJ Luck','Concerto do DJ Luck',5,'2017-02-08','2017-01-26','2017-02-08',CURRENT_TIME(),33.99,46,39),
            (5,'DJ Pandemonium','Concerto do DJ Pandemonium',5,'2017-05-18','2017-05-03','2017-05-16',CURRENT_TIME(),38.99,43,40),
            (6,'DJ Pyro','Concerto do DJ Pyro',5,'2017-09-25','2017-09-08','2017-09-21',CURRENT_TIME(),29.99,33,41),
            (7,'Impulse','Concerto do Impulse',5,'2017-01-13','2016-12-30','2017-01-12',CURRENT_TIME(),27.99,34,42),
            (8,'Fusion','Concerto do Fusion',5,'2017-06-20','2017-06-06','2017-06-19',CURRENT_TIME(),22.99,37,43),
            (9,'Witness','Concerto do Witness',5,'2017-06-24','2017-06-07','2017-06-20',CURRENT_TIME(),29.99,32,44),
            (10,'Carnage','Concerto do Carnage',5,'2017-04-12','2017-03-26','2017-04-08',CURRENT_TIME(),21.99,46,45),
            (11,'DJ Lumière','Concerto do DJ Lumière',5,'2017-06-10','2017-05-27','2017-06-09',CURRENT_TIME(),36.99,32,46),
            (12,'DJ Blanche','Concerto do DJ Blanche',5,'2017-12-22','2017-12-08','2017-12-21',CURRENT_TIME(),36.99,49,47),
            (13,'DJ Tornade','Concerto do DJ Tornade',5,'2017-11-25','2017-11-10','2017-11-23',CURRENT_TIME(),27.99,44,48),
            (14,'Acoustiques','Concerto do Acoustiques',5,'2017-06-08','2017-05-25','2017-06-07',CURRENT_TIME(),34.99,39,49),

```

```

(15,'Ruse','Concerto      do      Ruse',5,'2017-06-19','2017-06-
05','2017-06-18',CURRENT_TIME(),37.99,30,50),
(16,'DJ Rêveur','Concerto do DJ Rêveur',5,'2017-05-13','2017-
04-29','2017-05-12',CURRENT_TIME(),28.99,33,51),
(17,'Airtiste','Concerto do Airtiste',5,'2017-09-19','2017-09-
02','2017-09-15',CURRENT_TIME(),36.99,38,52),
(18,'Visage','Concerto    do   Visage',5,'2017-10-13','2017-09-
26','2017-10-09',CURRENT_TIME(),38.99,43,53),
(19,'DJ Charme','Concerto do DJ Charme',5,'2017-05-26','2017-
05-09','2017-05-22',CURRENT_TIME(),21.99,40,54),
(20,'DJ Témoin','Concerto do DJ Témoin',5,'2017-08-11','2017-
07-25','2017-08-07',CURRENT_TIME(),32.99,46,55),
(21,'Aberrant','Concerto do Aberrant',5,'2017-03-19','2017-03-
02','2017-03-15',CURRENT_TIME(),34.99,45,56);

```

```

INSERT INTO Organizador_has_evento (Evento_id,Organizador_Entidade_id)
VALUES
(1,21),
(2,34),
(3,23),
(4,31),
(5,22),
(6,25),
(7,35),
(8,31),
(9,26),
(10,24),
(11,26),
(12,24),
(13,21),
(14,27),
(15,27),
(16,22),
(17,35),
(18,21),
(19,20),
(20,28),
(21,31),
(21,23),
(6,32),

```

```

(19,27),
(6,30),
(16,26),
(20,21),
(1,26),
(2,32),
(5,33),
(3,21);

INSERT INTO
Divulgacao(Id, Conteudo, Tipo, Custo, DataInicio, DataFim, Evento_id, Plataforma_id)
VALUES
    (1, 'Concerto' do 'Quiver', 6, 290.99, '2017-08-10', '2017-08-23', 1, 1),
    (2, 'Concerto' do DJ 'Maximus', 8, 305.99, '2017-05-26', '2017-06-08', 2, 4),
    (3, 'Concerto' do 'Dreamer', 1, 306.99, '2017-09-24', '2017-10-07', 3, 4),
    (4, 'Concerto' do DJ 'Luck', 7, 330.99, '2017-01-26', '2017-02-08', 4, 4),
    (5, 'Concerto' do DJ 'Pandemonium', 1, 308.99, '2017-05-03', '2017-05-16', 5, 5),
    (6, 'Concerto' do DJ 'Pyro', 3, 290.99, '2017-09-08', '2017-09-21', 6, 3),
    (7, 'Concerto' do 'Impulse', 10, 270.99, '2016-12-30', '2017-01-12', 7, 2),
    (8, 'Concerto' do 'Fusion', 1, 220.99, '2017-06-06', '2017-06-19', 8, 5),
    (9, 'Concerto' do 'Witness', 7, 209.99, '2017-06-07', '2017-06-20', 9, 5),
    (10, 'Concerto' do 'Carnage', 2, 201.99, '2017-03-26', '2017-04-08', 10, 1),
    (11, 'Concerto' do DJ 'Lumière', 9, 36.99, '2017-05-27', '2017-06-09', 11, 1),
    (12, 'Concerto' do DJ 'Blanche', 3, 369.99, '2017-12-08', '2017-12-21', 12, 4),
    (13, 'Concerto' do DJ 'Tornade', 1, 278.99, '2017-11-10', '2017-11-23', 13, 5),
    (14, 'Concerto' do 'Acoustiques', 3, 3400.99, '2017-05-25', '2017-06-07', 14, 4),

```

```

(15,'Concerto      do      Ruse',7,3798.99,'2017-06-05','2017-06-
18',15,4),
(16,'Concerto    do   DJ  Rêveur',1,2889.99,'2017-04-29','2017-05-
12',16,3),
(17,'Concerto    do   Airtiste',1,3896.99,'2017-09-02','2017-09-
15',17,2),
(18,'Concerto    do   Visage',1,38896.99,'2017-09-26','2017-10-
09',18,5),
(19,'Concerto    do   DJ  Charme',5,2178.99,'2017-05-09','2017-05-
22',19,4),
(20,'Concerto    do   DJ  Témoin',2,3278.99,'2017-07-25','2017-08-
07',20,1),
(22,'Quiver  Strikes  Back',6,390.99,'2017-08-10','2017-08-
23',1,3),
(23,'Concerto    do   Rei',1,306.99,'2017-09-24','2017-10-
07',3,1),
(24,'Concerto  de  Ourives',6,334.99,'2017-09-24','2017-10-
07',4,4);

```

#####

```

INSERT INTO PermiteEntrada_Evento_Participante_Divulgacao (Evento_Id,
Participante_Entidade_Id, Divulgacao_Id, Classificacao, Preco, Lugar)
VALUES (1,5,1,4,54,'L15'),
(1,15,22,3,54,'L12'),
(1,17,1,6,54,'L8'),
(2,5,2,5,52,'L41'),
(2,3,2,3,52,'L30'),
(3,6,3,6,24,'L15'),
(3,4,3,4,24,'L14'),
(3,20,3,5,24,'L13'),
(3,3,23,3,23,'L11'),
(4,3,4,7,22,'L40'),
(4,17,4,6,22,'L14'),
(4,4,4,3,22,'L5'),
(4,1,24,9,23,'L13'),
(5,20,5,6,32,'L36'),
(5,17,5,5,32,'L41'),
(5,19,5,3,32,'L11'),
(6,20,6,4,32,'L21'),
(6,19,6,3,32,'L24'),

```

(7,3,7,3,10,'L46'),
(7,9,7,5,10,'L21'),
(7,1,7,4,10,'L42'),
(8,15,8,6,12,'L27'),
(8,12,8,4,12,'L45'),
(9,12,9,5,12,'L44'),
(9,17,9,7,12,'L17'),
(10,17,10,7,10,'L35'),
(10,14,10,4,10,'L36'),
(10,2,10,7,10,'L28'),
(11,3,11,7,45,'L1'),
(11,20,11,6,45,'L18'),
(11,19,11,3,45,'L11'),
(12,4,12,3,44,'L47'),
(12,6,12,4,44,'L23'),
(12,17,12,4,44,'L29'),
(13,2,13,7,20,'L38'),
(13,20,13,5,20,'L25'),
(13,7,13,4,20,'L31'),
(14,11,14,7,15,'L14'),
(14,5,14,3,15,'L48'),
(15,10,15,3,15,'L32'),
(15,7,15,7,15,'L36'),
(15,1,15,5,15,'L8'),
(16,4,16,4,12,'L10'),
(16,9,16,6,12,'L9'),
(16,16,16,4,12,'L1'),
(17,9,17,7,12,'L31'),
(17,10,17,5,12,'L24'),
(17,4,17,4,12,'L47'),
(18,11,18,7,39,'L49'),
(18,2,18,5,39,'L4'),
(18,12,18,6,39,'L5'),
(19,8,19,3,39,'L26'),
(19,15,19,6,39,'L31'),
(19,7,19,7,39,'L0'),
(20,6,20,6,90,'L21'),
(20,19,20,6,90,'L7'),
(20,17,20,3,90,'L11');

III. Anexo 3 - Script de Implementação de Interrogações e Funcionalidades para o Funcionário

```
delimiter $$

CREATE PROCEDURE FParticipanteEvento(IN n_evento int)
begin
    Select distinct
        E.id as Id,
        P.genero as Genero,
        P.nif as Nif,
        P.datadenascimento as 'Data de Nascimento',
        Ent.nome as Nome, Ent.endereco as 'Endereço',
        Ent.email as Email, Ent.telemovel as 'Número de Telefone'
    From
        Evento as E,
        PermiteEntrada_Evento_Participante_Divulgacao as PEPD,
        Participante as P,
        Entidade as Ent
    where
        E.id = n_evento
        and PEPD.Evento_id = E.id
        and P.Entidade_id = PEPD.Participante_Entidade_id
        and Ent.id = P.Entidade_id;
end $$

delimiter ;

delimiter $$

CREATE PROCEDURE FEventoEntreDatas(IN d_start DATETIME, IN d_end DATETIME)
begin
    Select *
    From
```

```

        Evento as E
    where
        E.data between d_start and d_end;
    end $$

delimiter ;

delimiter $$

CREATE PROCEDURE FEventoEmLocal(IN n_local INT)
begin
    Select E.*
    From
        Evento as E
    where
        E.Local_Entidade_Id = n_local;
    end $$

delimiter ;

delimiter $$

CREATE PROCEDURE FParticipanteMaisGastaTotal(In n int)
begin
    Select
        E.*,
        P.*,
        sum(PEPD.Preco) as Gasto
    From
        Participante as P,
        PermiteEntrada_Evento_Participante_Divulgacao as PEPD,
        Entidade as E
    where
        P.Entidade_id = PEPD.Participante_Entidade_Id
        and P.Entidade_id = E.id
    group by P.Entidade_id
    order by sum(PEPD.Preco) DESC
    limit n;
end $$

delimiter ;

delimiter $$

CREATE PROCEDURE FParticipanteMaisGastaTipoEvento(In n int, In
n_tipo_e int)
begin
    Select

```

```

        E.*,
        P.*,
        sum(PEPD.Preco) as Gasto
From
        Participante as P,
        PermiteEntrada_Evento_Participante_Divulgacao as PEPD,
        Entidade as E,
        Evento as EV
where
        P.Entidade_id = PEPD.Participante_Entidade_Id
        and P.Entidade_id = E.id
        and PEPD.Evento_id = EV.id
        and EV.tipo = n_tipo_e

        group by P.Entidade_id
        order by sum(PEPD.Preco) DESC
        limit n;
end $$

delimiter ;

delimiter $$

CREATE PROCEDURE FParticipanteMaisGastaOrg(In n int, In n_org int)
begin
    Select
        E.*,
        P.*,
        sum(PEPD.Preco) as Gasto
From
        Participante as P,
        PermiteEntrada_Evento_Participante_Divulgacao as PEPD,
        Entidade as E,
        Evento as EV,
        Organizador as O,
        Organizador_has_Evento as OE

where
        P.Entidade_id = PEPD.Participante_Entidade_Id
        and P.Entidade_id = E.id
        and PEPD.Evento_id = EV.id
        and EV.id = OE.Evento_id
        and OE.Organizador_Entidade_id = n_org

```

```

        group by P.Entidade_id
        order by sum(PEPD.Preco) DESC
        limit n;
    end $$

delimiter ;

delimiter $$

CREATE PROCEDURE FDivulgacaoEficazTipoBruto(In n_tipo_e int)
begin
    Select
        D.tipo,
        count(PEPD.Participante_Entidade_Id) as Influenciados
    From
        Divulgacao as D,
        Entidade as E,
        PermiteEntrada_Evento_Participante_Divulgacao as PEPD
    where
        E.tipo = n_tipo_e
        and D.Evento_id = E.id
        and PEPD.Evento_id = E.id
        and PEPD.Divulgacao_id = D.id
        group by D.tipo
        order by count(PEPD.Participante_Entidade_Id) DESC;
    end $$

delimiter ;

delimiter $$

CREATE PROCEDURE FDivulgacaoEficazProporcao(In n_tipo_e int)
begin
    Select
        D.tipo,
        sum(PEPD.Preco)/count(PEPD.Participante_Entidade_Id)      as
'CustoPorParticipante'
    From
        Divulgacao as D,
        Entidade as E,
        PermiteEntrada_Evento_Participante_Divulgacao as PEPD
    where
        E.tipo = n_tipo_e
        and D.Evento_id = E.id

```

```
        and PEPD.Evento_id = E.id
        and PEPD.Divulgacao_id = D.id
        group by D.tipo
        order                                by
sum(PEPD.Preco)/count(PEPD.Participante_Entidade_Id) DESC;
end $$

delimiter ;
```

IV. Anexo 4 – *Script de Implementação de Interrogações e Funcionalidades para o Organizador*

```
delimiter $$

CREATE PROCEDURE OParticipantesDeEvento(IN n_org int, IN n_evento int)
begin
    Select
        Ent.nome,
        Ent.endereco,
        Ent.email,
        Ent.teleovel
    From
        Evento as E,
        Organizador as O,
        Organizador_has_Evento as OE,
        PermiteEntrada_Evento_Participante_Divulgacao as PEPD,
        Participante as P,
        Entidade as Ent
    where
        E.id = n_evento
        and E.id = OE.Evento_id
        and OE.Organizador_Entidade_id = n_org
        and PEPD.Evento_id = E.id
        and P.Entidade_id = PEPD.Participante_Entidade_id
        and Ent.id = P.Entidade_id;
    end $$

delimiter ;

delimiter $$

CREATE PROCEDURE OPlataformasDeEvento(IN n_org int, IN n_div int)
begin
```

```

Select distinct
P. *
From
    Evento as E,
    Organizador as O,
    Organizador_has_Evento as OE,
    Divulgacao as D,
    Plataforma as P
where
    E.id = OE.Evento_id
    and OE.Organizador_Entidade_id = n_org
    and D.Evento_id = E.id
    and D.id = n_div
    and D.Plataforma_id = P.id;
end $$

delimiter ;

delimiter $$

CREATE PROCEDURE OLocaisDeEventos(IN n_org int)
begin
    Select distinct
        Ent.nome,
        Ent.endereco,
        Ent.email,
        Ent.telemovel,
        L.tipo,
        L.descricao,
        L.lotacao
    From
        Evento as E,
        Organizador as O,
        Organizador_has_Evento as OE,
        Local as L,
        Entidade as ent
    where
        E.id = OE.Evento_id
        and OE.Organizador_Entidade_id = n_org
        and L.Entidade_id = L.Entidade_id
        and Ent.id = L.Entidade_id;
end $$

delimiter ;

```

```

delimiter $$

CREATE PROCEDURE OEventos(IN n_org int)
begin
    Select
        E. *
    From
        Evento as E,
        Organizador as O,
        Organizador_has_Evento as OE
    where
        E.id = OE.Evento_id
        and OE.Organizador_Entidade_id = n_org;
end $$

delimiter ;

delimiter $$

CREATE PROCEDURE ODivulgacao(IN n_div INT, IN n_org INT)
begin
    Select
        D. *
    From
        Divulgacao as D,
        Evento as E,
        Organizador_has_Evento as OE
    where
        D.id = n_divulgacao
        and D.Evento_id = E.id
        and OE.Evento_id = E.id
        and OE.Organizador_Entidade_id = n_org;
end $$

delimiter ;

delimiter $$

CREATE PROCEDURE ODivulgacaoInfluencia(IN n_div INT, IN n_org INT)
begin
    Select
        D.*,
        count(PEPD.Participante_Entidade_Id) as Influenciados
    From
        Divulgacao as D,

```

```
    Evento as E,
    Organizador_has_Evento as OE,
    PermiteEntrada_Evento_Participante_Divulgacao as PEPD
where
    D.id = n_div
        and PEPD.Divulgacao_id = D.id
        and D.Evento_id = E.id
        and PEPD.Evento_id = E.id
        and OE.Evento_id = E.id
        and OE.Organizador_Entidade_id = n_org
        group by PEPD.Divulgacao_id;
end $$

delimiter ;
```

V. Anexo 5 - Script de Criação das Vistas que respondem a parte das Interrogações

```
CREATE VIEW FDivulgacaoInfluencia as
    Select
        D.*,
        count(PEPD.Participante_Entidade_Id) as Influenciados
    From
        Divulgacao as D,
        PermiteEntrada_Evento_Participante_Divulgacao as PEPD
    where
        PEPD.Divulgacao_id = D.id
    group by
        PEPD.Divulgacao_id
    order by
        count(PEPD.Participante_Entidade_Id) DESC;
```

```
CREATE VIEW FContagemTipoDivulgacao as
    Select
        D.tipo,
        count(D.tipo) as Contagem
    From
        Divulgacao as D,
        PermiteEntrada_Evento_Participante_Divulgacao as PEPD
    where
        PEPD.Divulgacao_id = D.id
    group by
        D.tipo
    order by
        count(D.tipo) DESC;
```

VI. Anexo 6 - Script de Funcionalidades de Simplificação da Interação com a BD

```
DELIMITER $$

CREATE PROCEDURE addClassificacao(
    IN id_particante INT,
    IN id_evento INT,
    IN apreciation DECIMAL(4, 2)
)
BEGIN
    UPDATE
        PermiteEntrada_Evento_Participante_Divulgacao as PEPD
    SET
        PEPD.Classificacao = apreciation
    WHERE
        PEPD.Evento_id = id_evento
        and PEPD.Entidade_Participante_id = id_particante
END $$

DELIMITER ;

DELIMITER $$

CREATE PROCEDURE ValidarDivulgacao(IN id_divulgacao INT)
BEGIN
    UPDATE
        Divulgacao as D
    SET
        D.Validade = 2
    WHERE
        D.id = id_divulgacao
END $$

DELIMITER ;

DELIMITER $$

CREATE PROCEDURE AlterarEstadoParticipacao()
```

```

        IN id_evento INT,
        IN id_participante INT,
        IN estado_novo ENUM('Válido', 'Cancelado', 'Reservado')
    )
BEGIN
    UPDATE
        PermiteEntrada_Evento_Participante_Divulgacao as PEPD
    SET
        PEPD.Estado = estado_novo
    WHERE
        EPD.Participante_Entidade_id = id_participante
        and PEPD.Evento_id = id_evento

    END $$

DELIMITER ;

DELIMITER $$

CREATE PROCEDURE CriarPlataforma(
    IN id INT,
    IN nome VARCHAR(64)
)
BEGIN
    INSERT INTO
        Plataforma(id, nome)
    VALUES(id, nome)

END $$

DELIMITER ;

DELIMITER $$

CREATE PROCEDURE CriarDivulgacao(
    IN d_id INT,
    IN d_conteudo TEXT(1024),
    IN d_custo DECIMAL(12, 2),
    IN d_tipo ENUM(
        "Audiovisual",
        "Áudio",
        "Publicação",
        "Cartaz",
        "Email",
        "Brochuras",

```

```

        "Aresentações",
        "Promoções",
        "Panfletos",
        "Outros"
    ) ,
    IN d_DataInicio DATETIME,
    IN d_DataFim DATETIME,
    IN d_plataforma_id INT,
    IN d_Evento_id INT
)
BEGIN
    INSERT INTO
        Divulgacao(
            id,
            conteudo,
            tipo,
            custo,
            DataInicio,
            DataFim,
            plataforma_id,
            Evento_id
        )
    VALUES (
        d_id,
        d_conteudo,
        d_tipo,
        d_custo,
        d_DataInicio,
        d_DataFim,
        d_plataforma_id,
        d_Evento_id
    )
END $$

DELIMITER;

# 1. Adicionar Local
DELIMITER $$

CREATE PROCEDURE addLocalComEntidadeExistente(IN id INT, IN lotacao
INT, IN descricao TEXT(256),

```

```

        IN tipo ENUM('Bares', 'Casas noturnas', 'Casa de
espetáculos', 'Hotelaria', 'Restaurantes',
                           'Centros
Culturais', 'Multiusos', 'Quintas', 'Outros'))
BEGIN
    INSERT INTO Local (entidade_id, lotacao, descricao, tipo)
    VALUES (id, lotacao, descricao, tipo);
END $$

DELIMITER ;

# 2. Adicionar Organizador
DELIMITER $$

CREATE PROCEDURE addOrganizadorComEntidadeExistente(IN id INT, IN
descricao TEXT(512))
BEGIN
    INSERT INTO Organizador(entidade_id, descricao)
    VALUES (id, descricao);
END $$

DELIMITER ;

# 3. Adicionar Participante
DELIMITER $$

CREATE PROCEDURE addParticipanteComEntidadeExistente(IN id INT, IN
datadenascimento DATE, IN genero ENUM('Feminino', 'Masculino'), IN nif
INT)
BEGIN
    INSERT INTO Participante (entidade_id, datadenascimento,
genero, nif)
    VALUES (id, datadenascimento, genero, nif);
END $$

DELIMITER ;

```