Universidade do Minho
Licenciatura em Ciências da Computação

Unidade Curricular de Bases de Dados

Ano Lectivo de 2023/2024

Gestão e Divulgação do Calendário de Eventos de uma Cidade

Diogo Aires, Pedro Martins, Henrique Faria

setembro, 2023

BD

Data de Recepção	
Responsável	
Avaliação	
Observações	

City Dreams - Organização e Divulgação de Eventos

Diogo Aires, Pedro Martins, Henrique Faria

setembro, 2023

Resumo

O presente trabalho tem como objetivo planear e criar uma Base de Dados Relacional para uma empresa que se dedica à organização e divulgação de eventos. Em particular, estamos a desenvolver um sistema para gerir e promover o calendário de eventos de uma cidade.

Neste relatório, vamos explicar a definição do projeto, contextualizá-lo, discutir a motivação por trás da sua implementação e estabelecer os objetivos. Vamos também analisar se o projeto é viável e apresentar um plano detalhado para a sua execução.

Explicaremos como recolhemos os requisitos para o sistema e descreveremos os diferentes tipos de utilizadores envolvidos no processo. Além disso, mostraremos como fizemos a modelação conceptual e lógica da base de dados.

De seguida, apresentaremos um plano para a implementação física do sistema, com exemplos de código SQL. Finalmente, concluiremos o relatório com um resumo das principais conclusões.

Área de Aplicação: Desenho e arquitetura da Base de Dados.

Palavras-Chave: SQL, MySQL, Base de Dados Relacional, Modelo Conceptual, Modelo lógico, Modelo Físico.

Índice

Índice de Figuras

Índice de Tabelas

1. Definição do Sistema

1.1. Contexto de aplicação e fundamentação do sistema

Nos últimos anos, eventos como concertos em festas e festivais em grandes e pequenas cidades (e até mesmo em vilas) têm visto um aumento significativo em popularidade. Isso resultou numa demanda crescente por uma organização mais sofisticada desses eventos. Para atender a essa demanda, várias empresas de organização de eventos surgiram para oferecer serviços especializados.

A City Dreams foi uma dessas empresas. Criada pelos sócios Diogo Aires, Pedro Martins e Henrique Faria, em meados de 2019. Sediada em Braga, a empresa tem vindo a evoluir de forma notável, contando com múltiplos funcionários. O seu portfólio inclui desde concertos memoráveis a festas temáticas extravagantes, e o sucesso da City Dreams é impulsionado pela sua capacidade de proporcionar experiências únicas aos clientes. O sucesso da City Dreams foi tão grande, que o próximo passo será a organização de um festival jovem a nível nacional (em Braga). No entanto, à medida que o volume de eventos e a lista de clientes aumentaram, tornou-se evidente que era necessário uma base de dados para acompanhar o ritmo e garantir a continuação da excelência operacional.

À medida que a popularidade desses eventos cresceu, o volume de trabalho envolvido na organização e na gestão de informações associadas a estes também aumentou consideravelmente. Isso inclui detalhes sobre clientes, locais, datas, orçamentos e muito mais. Manter o controlo de todas essas informações manualmente tornou-se um desafio cada vez maior.

Portanto, a necessidade de implementar uma base de dados tornou-se evidente. Um sistema desse tipo permitiria à empresa gerenciar eficazmente todas as informações relacionadas aos eventos, facilitando o acesso, a organização e a recuperação de dados importantes. Além disso, melhoraria a eficiência operacional, ajudando a empresa a lidar com um fluxo de trabalho crescente de maneira mais eficaz e precisa.

1.2. Motivação e Objetivos do Trabalho

A City Dreams, nos seus primórdios, utilizava um sistema de arquivos para o armazenamento de dados que lia e guardava dados consoante necessário. Este sistema funciona com utilizações em pequena escala, mas com um aumento exponencial da lista de clientes, eventos/festivais e contactos a complexidade é demasiado grande para ser viável. O uso de uma base de dados tornou-se na melhor opção para um crescimento estruturado da empresa e para uma melhor gestão de recursos humanos e dos seus respectivos eventos.

1.3. Análise e Viabilidade do Processo

Com a implementação deste sistema na empresa espera-se que esta consiga atingir os objetivos pretendidos no serviço de organização do seu festival, assim como conseguir oferecer um serviço de qualidade ao cliente.

Depois de uma reunião para analisar a proposta, concluímos que num momento inicial será feita apenas uma base de dados relativa à prestação de serviços a municípios e a gestão de custos de forma a garantir financiamento para esta fase. Num futuro próximo será construída a componente de gestão de recursos para a criação de um evento a nível nacional.

Espera-se no primeiro ano recuperar 50% do investimento inicial.

1.4. Recursos e Equipa de Trabalho

Para implementar esta base de dados será necessário um computador para o escritório. Este computador irá servir de ponto de acesso a um servidor que armazena a base de dados com o objetivo de ler e modificar dados.

A equipa de trabalho é constituída por 3 técnicos que irão desenvolver a base de dados no decorrer dos próximos 3 meses.

1.5. Plano de Execução

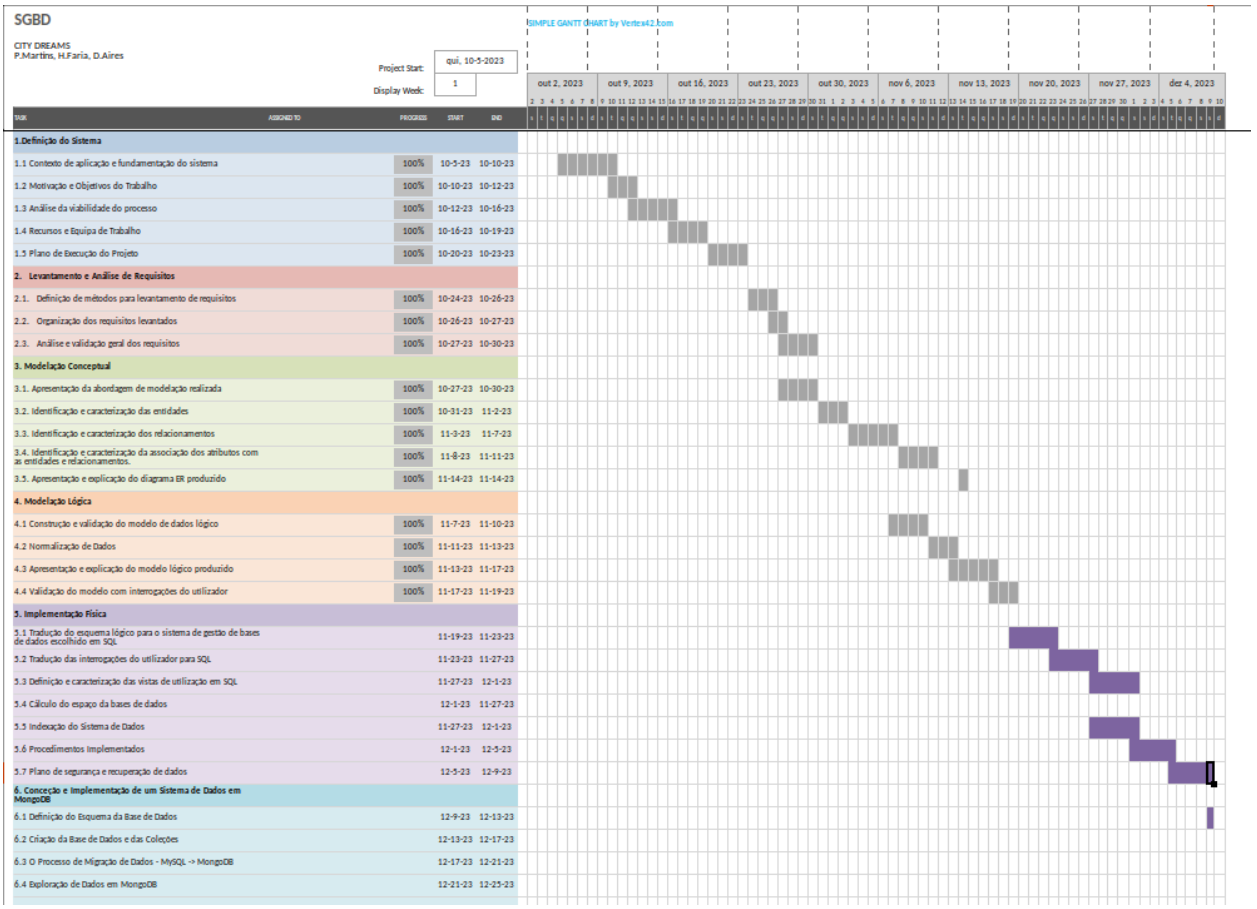

Figura 1 : Plano de Execução do Trabalho

2. Levantamento e Análise de Requisitos

2.1. Método de Levantamento e de Análise de Requisitos

A principal forma de levantamento de requisitos foi através de três reuniões com todos os funcionários da empresa. Uma vez que a empresa não é de grandes dimensões foi possível juntar todos os funcionários no final de um dia de trabalho.
Após a ocorrência de alguns incidentes na organização de eventos ,decidimos fazer questionários de forma a perceber o que estava mal, e de que forma poderíamos melhorar o nosso serviço.
Efetuamos então uma averiguação do resultado dos questionários, e assim, através dos mesmos conseguimos definir quais os requisitos que serão necessários para a construção da nossa BD.

Com a análise das faturas já existentes foi possível concluir que dados são essenciais para a organização de um serviço.

2.2. Levantamento de Requisitos

De acordo com os métodos descritos acima levantaram-se os seguintes requisitos que são necessários para a criação desta base de dados.

Com o propósito de facilitar a organização dos nossos requisitos, tendo em conta a estrutura estabelecida pela linguagem SQL, procedemos à organização dos requisitos levantados da seguinte forma: requisitos de descrição, requisitos de exploração e requisitos de controlo.

2.2.1 Requisitos de Descrição

1. Cada cliente deve ter um id, uma forma de identificação e um contacto;
2. Cada pedido tem de possuir a data e local desejados, bem como o tipo do evento e o preço máximo que o cliente está disposto a pagar pelo mesmo. Caso pretenda pedidos mais elaborados existe a opção de acrescentar uma descrição. Por pedido haverá um id;
3. Cada evento tem um id, uma data e local final onde irá decorrer o evento, sendo que o local terá uma rua e um código postal, haverá também um preço final tal como uma hora de início e de fim e o número de pessoas que estão na festa;
4. Cada funcionário terá um id, um contacto, um nif, um nome e um tipo para saber quais as tarefas que tem de executar;
5. Cada cliente pode fazer múltiplos pedidos, mas um pedido só pode estar associado a um cliente;
6. Cada pedido tem um e um só evento associado e um evento tem um e um só pedido associado;
7. Cada evento pode ter vários funcionários a trabalhar e os funcionários poderão trabalhar em mais do que um evento desde que os mesmos não possuam horas sobrepostas com outros eventos.

2.2.2 Requisitos de Exploração

1. Listar todos os clientes;
2. Listar os eventos por ordem das datas mais recentes em que vão acontecer;
3. Deve disponibilizar uma lista decrescente dos funcionários que trabalharam em mais eventos;
4. Apresentar o pedido feito por um determinado cliente;
5. Mostrar uma lista com os cliente que tenham o maior valor gasto com os seus eventos;
6. Listar o tempo de duração de cada evento

2.2.3 Requisitos de Controlo

1. Os funcionários apenas poderão consultar a hora de início e hora final de cada evento em que estarão alocados, assim como a data e local do evento;
2. Apenas os 3 donos da empresa terão acesso total à informação existente na base de dados;
3. Os clientes terão acesso à data e local final, bem como o preço final a pagar e a hora de início e fim do evento;
4. Apenas os clientes podem alterar os requisitos dos seus pedidos (preço máximo, local data, tipo e descrição), bem como o seu contacto e identificação;
5. Apenas os funcionários poderão alterar as suas informações tais como o contacto, nome, nif.

2.3. Análise e Validação dos Requisitos

Neste projeto, seguimos um processo de levantamento de requisitos que envolveu a categorização em três grupos: requisitos de descrição, de exploração e de controlo. Na fase inicial, concentramo-nos na identificação dos elementos essenciais necessários para o funcionamento da aplicação. Em seguida, na fase de levantamento de requisitos de exploração, examinamos as informações que poderiam ser extraídas dos diversos elementos e relacionamentos presentes na base de dados. Por fim, na última fase de levantamento, que se refere aos requisitos de controlo, definimos as restrições necessárias para a administração da base de dados, garantindo assim a eficiência do sistema.

Após essa análise detalhada dos três grupos de requisitos, a nossa equipa avaliou a sua viabilidade e validou a sua aplicação para construir uma base de dados que atendesse às necessidades específicas do projeto.

3. Modelação Conceptual

3.1. Apresentação da Abordagem da Modelação Realizada

Considerando os requisitos levantados e validados anteriormente, vamos agora proceder à explicação do processo de planeamento da estrutura da Base de Dados implementada. Para alcançar isto, vamos construir um Diagrama ER que representará as diversas entidades, incluindo os atributos que as compõem e os relacionamentos existentes entre as mesmas. Sendo assim, vamos para apresentar os diferentes componentes desta modelação, simulando o processo que decorreu. Ou seja, começamos pela identificação e caracterização de cada entidade, seguida pela caracterização e identificação de cada relacionamento e, por fim, o diagrama obtido utilizando o software brModelo.

3.2. Identificação e Caracterização das Entidades

- **Cliente:** Esta entidade corresponde aos diversos clientes da empresa. Contém informações sobre os nomes/empresas associadas e os seus respectivos contactos.
- **Pedido:** Esta entidade corresponde aos diversos pedidos feitos por cada cliente. Contém informações como o tipo de evento que se pretende realizar, o local e data pretendido, o preço máximo que o cliente se dispõe a gastar e pode ou não conter uma descrição para pedidos mais elaborados.
- **Evento:** Esta entidade corresponde aos eventos que foram criados de acordo com a viabilidade dos pedidos que foram sendo realizados. Contém informações acerca da data e local que ficaram definidos, assim como o tipo de evento que se pretende realizar, hora em que começa e hora em que acaba, a quantidade de pessoas que vão ou se espera que compareçam dependendo do tipo de evento em causa e também um preço final de todos os custos do evento.

- **Funcionário:** Esta entidade corresponde aos funcionários existentes na empresa. Contém informações sobre os nomes, os contactos, os nifs e que tipo de serviços realizam.

3.3. Identificação e Caracterização dos Relacionamentos

- **Relacionamento Cliente-Pedido**

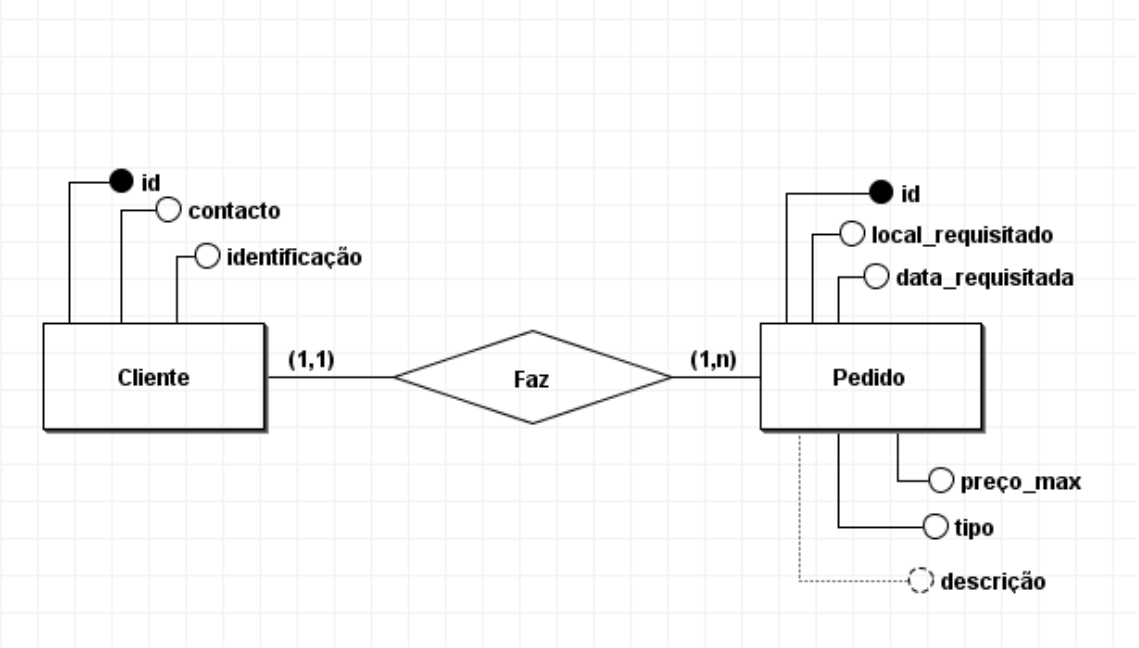

Figura 2 : Relacionamento Cliente-Pedido

Cada cliente pode fazer 1 ou N pedidos, e cada pedido só pode ser feito por 1 cliente, logo temos uma cardinalidade de 1 para N.

- **Relacionamento Pedido-Evento**

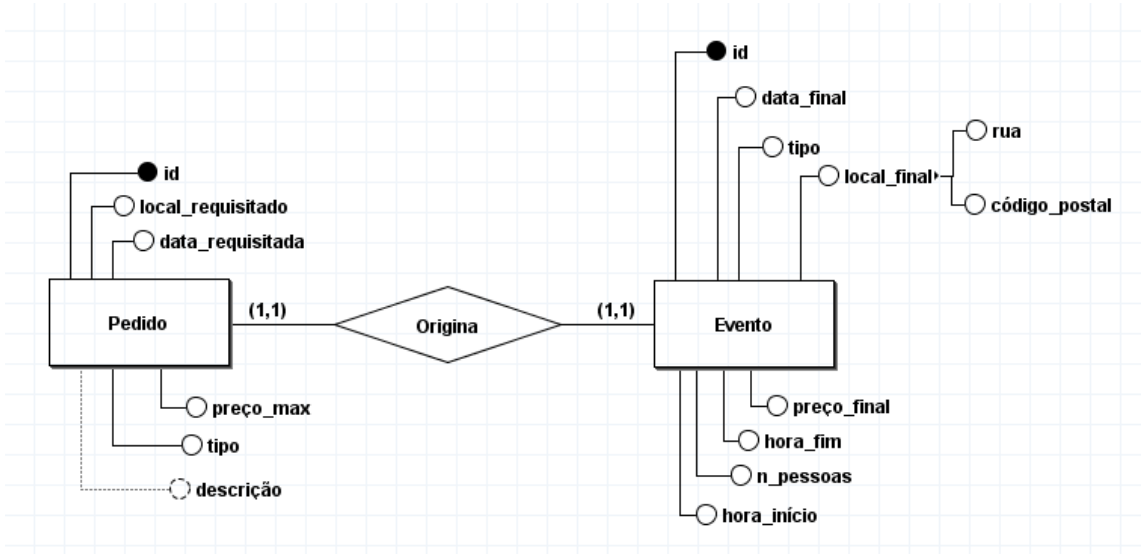

Figura 3 : Relacionamento Pedido-Evento

Cada pedido origina 1 evento, e cada evento é originado por 1 pedido, logo temos uma cardinalidade de 1 para 1.

- **Relacionamento Evento-Funcionário**

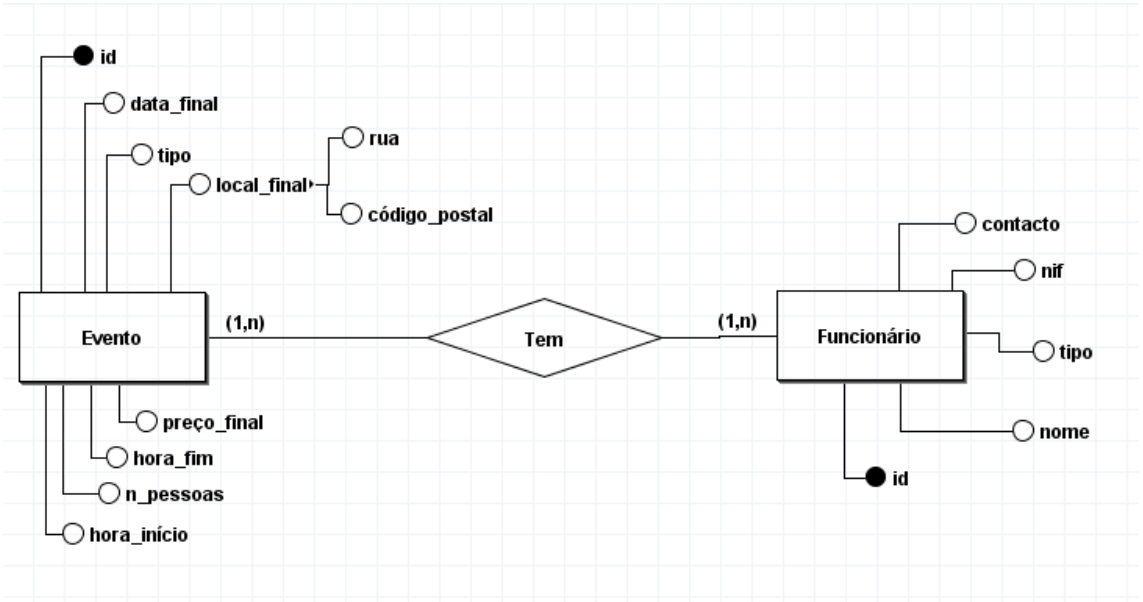

Figura 4 : Relacionamento Evento-Funcionário

Cada evento tem 1 ou N funcionários, e cada funcionário pode estar em 1 ou N eventos, logo temos uma cardinalidade de N para M.

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

- **Atributos da Entidade Cliente:**

Atributos	Tipo	Descrição
ID Cliente	Chave	Número único que identifica o cliente
Contacto	Simples	Contacto do cliente
identificação	Simples	Nome do cliente/empresa

Tabela 1 : Atributos Cliente

- **Atributos da Entidade Pedido**

Atributos	Tipo	Descrição
ID Pedido	Chave	Número único que identifica o pedido
local_requisitado	Simples	Local escolhido por parte do cliente para a criação do evento
data_requisitada	Simples	Data escolhida por parte do cliente para a criação do evento
tipo	Simples	Que tipo de evento o cliente pretende fazer
preço_max	Simples	Budget do cliente que se pode gastar no evento
descrição	Simples	Para pedidos mais detalhados do cliente relativamente a um determinado evento

Tabela 2 : Atributos Pedido

- **Atributos da Entidade Evento**

Atributos	Tipo	Descrição
ID Evento	Chave	Número único que identifica o evento
tipo	Simples	tipo de evento
local_final/código_postal	Composto	sequência numérica do endereço em que vai acontecer o evento
local_final/rua	Composto	nome da rua em que vai acontecer o evento
data_final	Simples	data em que ficou registado o evento
hora_ínicio	Simples	hora de ínicio do evento
hora_fim	Simples	hora final do evento
n_pessoas	Simples	quantas pessoas vão ou se espera que compareçam no evento
preço_final	Simples	custos finais do evento

Tabela 3 : Atributos Evento

- **Atributos da Entidade Funcionário**

Atributos	Tipo	Descrição
ID Funcionário	Chave	Número único que identifica o funcionário
nome	Simples	Nome do funcionário
contacto	Simples	Contacto do funcionário
tipo	Simples	Que tipo de serviços presta
nif	Simples	Número de identificação fiscal do funcionário

Tabela 4 : Atributos Funcionário

3.5. Diagrama ER

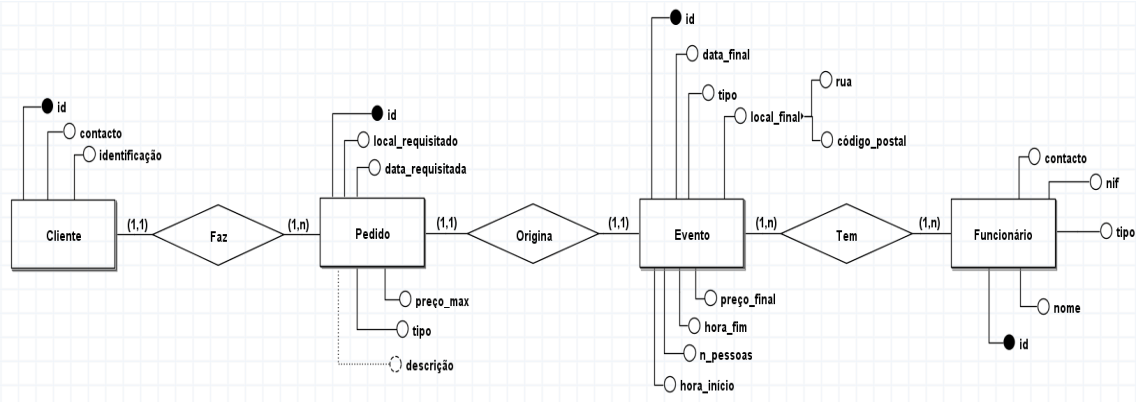

Figura 5 : Diagrama ER

4. Modelação Lógica

4.1. Normalização dos Dados

A normalização tem como principal objetivo evitar a redundância de dados, proporcionando um maior rendimento do modelo, uma vez que este processo evita as anomalias provocadas pela inserção, exclusão e alteração de registos na base de dados

Para avaliar a normalização do sistema devemos verificar que as 3 formas normais são satisfeitas, tendo em conta a análise das tabelas das diferentes entidades.

id	identificacao	contacto
1	João Silva	123456789
2	Maria Santos	987654321
3	Pedro Almeida	555666777
4	Ana Oliveira	111222333
5	Carlos Pereira	999888777
6	Sofia Rodrigues	444555666
7	Miguel Costa	777888999
8	Inês Ferreira	666555444
9	Rui Sousa	222333444
10	Lúcia Martins	888999000

Tabela 5 - Tabela Cliente

id	local_requisitado_rua	local_requisitado_codigo_postal	data_requisitada	preco_max	tipo	descricao	Cliente_id
1	Rua A	12345	2023-01-10 08:00:00	50100	Concerto	NULL	1
2	Rua B	54321	2023-02-15 14:30:00	3400	Festa	NULL	2
3	Rua C	67890	2023-03-20 10:45:00	5000	Conferencia	NULL	3
4	Rua D	48217	2023-04-05 09:00:00	150.75	Normal	NULL	4
5	Rua E	24680	2023-05-12 11:20:00	6000	Palestra	NULL	5
6	Rua F	90356	2023-06-25 16:00:00	500	Aniversario	NULL	6
7	Rua F	90356	2023-06-25 16:00:00	3000	Orgia	NULL	7
8	Rua G	73980	2023-08-19 09:30:00	300.25	Urgente	NULL	8
9	Rua I	24873	2023-09-30 10:00:00	50	Normal	NULL	9
10	Rua J	60194	2023-10-14 15:45:00	180	Urgente	NULL	10

Tabela 6 - Tabela Pedido

id	nome	contacto	tipo
1	Carlos Silva	987654321	Gerente
2	a Santos	123456789	Assistente
3	Miguel Pereira	555666777	Assistente
4	Sofia Ferreira	111222333	Assistente
5	João Almeida	999888777	Assistente

Tabela 7 - Tabela Funcionario

id	data_final	tipo	local_final_rua	local_final_codigo_postal	preco_final	hora_final	n_pessoas	hora_inicio	Pedido_id	Pedido_Cliente_id
1	2023-01-20 17:00:00	Concerto	Rua A	12345	50000	20:00:00	100	17:00:00	1	1
2	2023-02-10 20:30:00	Festa	Rua B	54321	3000	23:00:00	50	20:30:00	2	2
3	2023-03-15 18:00:00	Conferência	Rua C	67890	2000	23:00:00	200	18:00:00	3	3
5	2023-05-18 17:30:00	Palestra	Rua E	24680	5000	20:00:00	30	17:00:00	5	5
6	2023-06-25 16:00:00	Aniversario	Rua F	90356	422	17:00:00	10	16:00:00	6	6
7	2023-06-25 16:00:00	Orgia	Rua F	90356	2150	17:00:00	7	16:00:00	7	7

Tabela 8 - Tabela Evento

Evento_id	Evento_Pedido_id	Evento_Pedido_Cliente_id	Funcionario_id
1	1	1	2
5	5	5	2
1	1	1	3
3	3	3	3
1	1	1	4
2	2	2	4
3	3	3	4
5	5	5	4
1	1	1	5
5	5	5	5

Tabela 9 - Tabela Evento_tem_Funcionario

A Primeira Forma Normal (1FN) enuncia que os atributos devem ser atómicos, ou seja, nas tabelas não podem existir valores repetidos, nem podem existir atributos multivalorados. Com uma simples análise às tabelas do nosso modelo, reparamos que tal não acontece, tornando válida a 1FN.

A Segunda Forma Normal (2FN), depende da satisfação da 1FN, citando também que os atributos normais devem depender apenas da chave primária da tabela.

Tomemos como exemplo, o Contacto que está presente na Entidade Cliente, este atributo depende somente da chave primária, Id. Verificando-se a mesma correlação com os restantes atributos do modelo, validando a 2FN.

Por último, a Terceira Forma Normal (3FN) que também está sujeito à validade das 2 formas anteriores, impõe a verificação da existência de vínculos entre os diferentes atributos, isto é, se existe algum atributo que pode ser gerado a partir de outro. No caso do nosso modelo, nenhum atributo é proporcional ou possível de obter a partir de outro, cumprindo assim a 3FN.

4.2. Desenho do Modelo Lógico

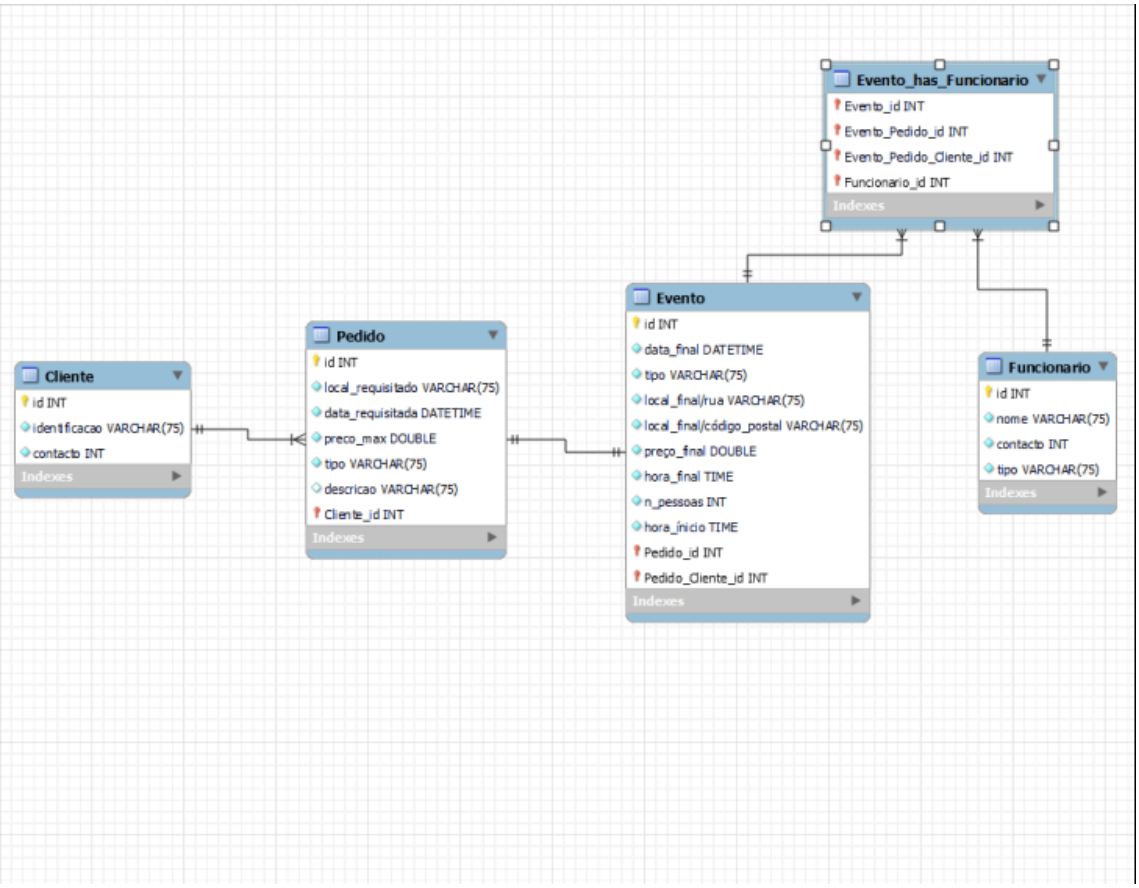

Figura 6 : Diagrama Lógico

4.3. Validação do Modelo Lógico com Interrogações do Utilizador

- Listar todos os clientes;

π Cliente.id, Cliente.identificacao, Cliente.contacto(Cliente)

- Listar os eventos e os respetivos clientes por ordem das datas mais recentes em que vão acontecer;

π Cliente.identificacao, Evento.tipo, Evento.data_final (((Cliente)
⋈
Cliente.id = Pedido.Cliente_id (Pedido)) ⋈ Evento.Pedido_id = Pedido.id (Evento))

- Deve disponibilizar uma lista decrescente dos funcionários que trabalharam em mais eventos;

π Funcionario.nome, Participacoes(γ Funcionario.id,
COUNT(Evento_tem_Funcionario.Funcionario_id) -> Participacoes (((Evento_tem_Funcionario)
⋈
Evento_tem_Funcionario.Funcionario_id = Funcionario.id (Funcionario))))

- Apresentar o pedido feito por um determinado cliente;

π Cliente.identificacao (σ Cliente.id = 5 (Cliente))

- Mostrar uma lista com os clientes que tenham o maior valor gasto com os seus eventos;

π Cliente.identificacao, Total_Gasto
(γ Cliente.id, SUM(Evento.preco_final) -> Total_Gasto ((Evento)
⋈
Evento.Pedido_id = Pedido.id ((Pedido) ⋈ Pedido.Cliente_id = Cliente.id (Cliente))))

- Listar o tempo de duracao de cada evento;

π Evento.id, Evento.tipo, TIMEDIFF(hora_final, hora_inicio) -> Duracao (Evento)

5. Implementação Física

5.1. Tradução do esquema lógico para o sistema de gestão de bases de dados escolhido

O sistema de gestão de bases de dados utilizado para a implementação deste projeto foi o MySQL, visto que este foi fortemente recomendado. Este consiste num sistema de gerenciamento de bases de dados relacional de código aberto, que tem a linguagem SQL como interface.

Ao longo da realização do projeto em questão foram utilizadas as ferramentas MySQL assim como MySQL Workbench que graças à simples manipulação de dados que estas garantem, tornaram o processo de criação de criação de uma base de dados mais acessível, o que, por sua vez, enriqueceu a nossa experiência.

As tabelas criadas:

- Cliente:

```
CREATE TABLE IF NOT EXISTS Cliente (
  id INT NOT NULL,
  identificacao VARCHAR(75) NOT NULL,
  contacto INT NOT NULL,

  PRIMARY KEY (id),
  UNIQUE KEY contacto(contacto));
```

Figura 26 - Criação Tabela Entidade Cliente

- Pedido

```
CREATE TABLE IF NOT EXISTS Pedido (
  id INT NOT NULL,
  local_requisitado_rua VARCHAR(75) NOT NULL,
  local_requisitado_codigo_postal VARCHAR(75) NOT NULL,
  data_requisitada DATETIME NOT NULL,
  preco_max DOUBLE NOT NULL,
  tipo VARCHAR(75) NOT NULL,
  descricao VARCHAR(75) NULL,
  Cliente_id INT NOT NULL,

  PRIMARY KEY (id, Cliente_id),
  FOREIGN KEY (Cliente_id) REFERENCES Cliente(id));
```

Figura 27 - Criação Tabela Entidade Pedido

- Funcionario:

```
CREATE TABLE IF NOT EXISTS Funcionario (
  id INT NOT NULL,
  nome VARCHAR(75) NOT NULL,
  contacto INT NOT NULL,
  tipo VARCHAR(75) NOT NULL,
  PRIMARY KEY (id),
  UNIQUE KEY (contacto));
```

Figura 28 - Criação Tabela Entidade Funcionario

- Evento:

```
CREATE TABLE IF NOT EXISTS Evento (
  id INT NOT NULL,
  data_final DATETIME NOT NULL,
  tipo VARCHAR(75) NOT NULL,
  local_final_rua VARCHAR(75) NOT NULL,
  local_final_codigo_postal VARCHAR(75) NOT NULL,
  preco_final DOUBLE NOT NULL,
  hora_final TIME NOT NULL,
  n_pessoas INT NOT NULL,
  hora_inicio TIME NOT NULL,
  Pedido_id INT NOT NULL,
  Pedido_Cliente_id INT NOT NULL,
  PRIMARY KEY (id, Pedido_id, Pedido_Cliente_id),
  FOREIGN KEY (Pedido_id, Pedido_Cliente_id) REFERENCES Pedido (id, Cliente_id));
```

Figura 29 - Criação Tabela Entidade Evento

- Evento_Tem_funcionario:

```
CREATE TABLE IF NOT EXISTS Evento_tem_Funcionario (
  Evento_id INT NOT NULL,
  Evento_Pedido_id INT NOT NULL,
  Evento_Pedido_Cliente_id INT NOT NULL,
  Funcionario_id INT NOT NULL,
  PRIMARY KEY (Evento_id, Evento_Pedido_id, Evento_Pedido_Cliente_id, Funcionario_id),
  FOREIGN KEY (Evento_id, Evento_Pedido_id, Evento_Pedido_Cliente_id) REFERENCES Evento (id, Pedido_id, Pedido_Cliente_id),
  FOREIGN KEY (Funcionario_id) REFERENCES Funcionario (id));
```

Figura 30 - Criação Tabela Entidade Evento_Tem_funcionario

5.2. Tradução das interrogações do utilizador para SQL (alguns exemplos)

- Listar todos os Clientes

```
SELECT * FROM cliente;
```

Figura 7 : Query 1

Produzindo a respetiva tabela (resultado):

id	identificacao	contacto
1	João Silva	123456789
2	Maria Santos	987654321
3	Pedro Almeida	555666777
4	Ana Oliveira	111222333
5	Carlos Pereira	999888777
6	Sofia Rodrigues	444555666
7	Miguel Costa	777888999
8	Inês Ferreira	666555444
9	Rui Sousa	222333444
10	Lúcia Martins	888999000

Figura 8 - Resultado da Query 1

- Listar os eventos e os respetivos clientes por ordem das datas mais recentes em que vão acontecer

```
SELECT Cliente.identificacao, Evento.tipo, Evento.data_final FROM Evento
JOIN Pedido ON Evento.Pedido_id = Pedido.id
JOIN Cliente ON Pedido.Cliente_id = Cliente.id
ORDER BY Evento.data_final;
```

Figura 9 - Querie 2

Produzindo a respetiva tabela (resultado):

identificacao	tipo	data_final
João Silva	Concerto	2023-01-20 17:00:00
Maria Santos	Festa	2023-02-10 20:30:00
Pedro Almeida	Conferência	2023-03-15 18:00:00
Carlos Pereira	Palestra	2023-05-18 17:30:00
Sofia Rodrigues	Aniversario	2023-06-25 16:00:00
Miguel Costa	Orgia	2023-06-25 16:00:00

Figura 10 - Resultado da Query 2

- Disponibilizar uma lista decrescente dos funcionários que trabalharam em mais eventos

```
SELECT Funcionario.nome, COUNT(Evento_tem_Funcionario.Funcionario_id) AS Participacoes FROM Evento_tem_Funcionario
JOIN Funcionario ON Evento_tem_Funcionario.Funcionario_id = Funcionario.id
GROUP BY Funcionario.nome
ORDER BY COUNT(Evento_tem_Funcionario.Funcionario_id) DESC;
```

Figura 11 - Query 3

Produzindo a respetiva tabela (resultado):

nome	Participacoes
Sofia Ferreira	4
Ana Santos	2
Miguel Pereira	2
João Almeida	2

Figura 12 - Resultado da Query 3

- Apresentar o pedido feito por um determinado cliente

```
SELECT Cliente.identificacao FROM Cliente
WHERE (Cliente.id = 5);
```

Figura 13 - Query 4

Produzindo a respetiva tabela (resultado):

identificacao
Carlos Pereira

Figura 14 - Resultado da Query 4

- Mostrar uma lista com os clientes que tenham o maior valor gasto com os seus eventos

```
SELECT Cliente.identificacao, SUM(Evento.preco_final) AS Total_Gasto FROM Evento
JOIN Pedido ON Evento.Pedido_id = Pedido.id
JOIN Cliente ON Pedido.Cliente_id = Cliente.id
GROUP BY Cliente.identificacao
ORDER BY SUM(Evento.preco_final) DESC;
```

Figura 15 - Query 5

Produzindo a respetiva tabela (resultado):

identificacao	Total_Gasto
João Silva	50000
Carlos Pereira	5000
Maria Santos	3000
Miguel Costa	2150
Pedro Almeida	2000
Sofia Rodrigues	422

Figura 16 - Resultado da Query 5

- Listar o tempo de duração de cada evento

```
SELECT Evento.id, Evento.tipo, TIMEDIFF(hora_final, hora_inicio) AS Duracao FROM Evento;
```

Figura 17 - Query 6

Produzindo a respetiva tabela (resultado):

id	tipo	Duracao
1	Concerto	03:00:00
2	Festa	02:30:00
3	Conferência	05:00:00
5	Palestra	03:00:00
6	Aniversario	01:00:00

Figura 18 - Resultado da Query 6

5.3. Definição e caracterização das vistas de utilização em SQL (alguns exemplos)

- Listar todos os eventos com detalhes dos Funcionários:

```
CREATE VIEW View_Eventos_Funcionarios AS
SELECT ef.Evento_id, ef.Evento_Pedido_id, ef.Evento_Pedido_Cliente_id, ef.Funcionario_id,
       f.nome AS Funcionario_Nome, f.contacto AS Funcionario_Contacto, f.tipo AS Funcionario_Tipo
FROM Evento_tem_Funcionario ef
INNER JOIN Funcionario f ON ef.Funcionario_id = f.id;
```

Figura 19 - View 1

Produzindo o respetivo resultado:

```
SELECT * FROM View_Eventos_Funcionarios;
```

Figura 20 - Chamada da View 1

Evento_id	Evento_Pedido_id	Evento_Pedido_Cliente_id	Funcionario_id	Funcionario_Nome	Funcionario_Contacto	Funcionario_Tipo
1	1	1	2	Ana Santos	123456789	Assistente
5	5	5	2	Ana Santos	123456789	Assistente
1	1	1	3	Miguel Pereira	555666777	Assistente
3	3	3	3	Miguel Pereira	555666777	Assistente
1	1	1	4	Sofia Ferreira	111222333	Assistente
2	2	2	4	Sofia Ferreira	111222333	Assistente
3	3	3	4	Sofia Ferreira	111222333	Assistente
5	5	5	4	Sofia Ferreira	111222333	Assistente
1	1	1	5	João Almeida	999888777	Assistente
5	5	5	5	João Almeida	999888777	Assistente

Figura 21 - Resultado da View 1

- Listar todos os eventos com detalhes dos clientes

```
CREATE VIEW View_Eventos_Cliente AS
SELECT e.id AS Evento_ID, e.data_final, e.tipo AS Evento_Tipo, e.local_final_rua, e.local_final_codigo_postal,
       e.preco_final, e.hora_final, e.n_pessoas, e.hora_inicio,
       p.id AS Pedido_ID, p.local_requisitado_rua, p.local_requisitado_codigo_postal, p.data_requisitada, p.preco_max, p.tipo AS Pedido_Tipo,
       c.id AS Cliente_ID, c.identificacao, c.contacto
FROM Evento e
INNER JOIN Pedido p ON e.Pedido_id = p.id AND e.Pedido_Cliente_id = p.Cliente_id
INNER JOIN Cliente c ON p.Cliente_id = c.id;
```

Figura 22 - View 2

Produzindo o respetivo resultado:

```
SELECT * FROM View_Eventos_Cliente;
```

Figura 23 - Chamada da View 2

Evento_ID	data_final	Evento_Tipo	local_final_rua	local_final_codigo_postal	preco_final	hora_final	n_pessoas	hora_inicio	Pedido_ID	local_requisitado_rua
1	2023-01-20 17:00:00	Concerto	Rua A	12345	50000	20:00:00	100	17:00:00	1	Rua A
2	2023-02-10 20:30:00	Festa	Rua B	54321	3000	23:00:00	50	20:30:00	2	Rua B
3	2023-03-15 18:00:00	Conferência	Rua C	67890	2000	23:00:00	200	18:00:00	3	Rua C
5	2023-05-18 17:30:00	Palestra	Rua E	24680	5000	20:00:00	30	17:00:00	5	Rua E
6	2023-06-25 16:00:00	Aniversario	Rua F	90356	422	17:00:00	10	16:00:00	6	Rua F

local_requisitado_codigo_postal	data_requisitada	preco_max	Pedido_Tipo	Cliente_ID	identificacao	contacto
12345	2023-01-10 08:00:00	150	Concerto	1	João Silva	123456789
54321	2023-02-15 14:30:00	3400	Festa	2	Maria Santos	987654321
67890	2023-03-20 10:45:00	5000	Conferencia	3	Pedro Almeida	555666777
24680	2023-05-12 11:20:00	6000	Palestra	5	Carlos Pereira	999888777
90356	2023-06-25 16:00:00	500	Aniversario	6	Sofia Rodrigues	444555666

Figuras 24,25 - Resultado da View 2

5.4. Cálculo do espaço da base de dados (inicial e taxa de crescimento anual)

Para obtermos o espaço que é ocupado pela Base de Dados, é necessário perceber o espaço ocupado por cada entidade.

Sabemos que (espaço de cada tipo):

- INT (4 bytes)
- DATE (3 bytes)
- VARCHAR(N) (2*n+1 bytes)
- TINYINT (1 byte)
- TEXT(N) (N+2, N<2^16)
- DECIMAL (8 bytes)
- DATETIME (8 bytes)

Assim:

Cliente

ATRIBUTOS	**TIPO DE DADOS**	**TAMANHO** (bytes)
id	INT	4
identificacao	VARCHAR(75)	75*2+1 = 151
contacto	INT	4

Tabela 10 - Tamanho Entidade Cliente

Evento

ATRIBUTOS	**TIPOS DE DADOS**	**TAMANHO** (bytes)
id	INT	4
data_final	DATETIME	8
tipo	VARCHAR(75)	2*75+1 = 151
local_final_rua	VARCHAR(75)	2*75+1 = 151
local_final_codigo_postal	VARCHAR(75)	2*75+1 = 151
preco_final	DOUBLE	8
hora_final	TIME	3
n_pessoas	INT	4
hora_inicio	TIME	3
pedido_id	INT	4
Pedido_Cliente_id	INT	4

Tabela 11 - Tamanho Entidade Evento

Funcionário

ATRIBUTOS	**TIPOS DE DADOS**	**TAMANHO** (bytes)
id	INT	4
nome	VARCHAR(75)	2*75+1 = 151
contacto	INT	4
tipo	VARCHAR(75)	2*75+1 = 151

Tabela 12 - Tamanho Entidade Funcionário

Pedido

ATRIBUTOS	**TIPOS DE DADOS**	**TAMANHO** (bytes)
id	INT	4
local_requisitado_rua	VARCHAR(75)	2*75+1 = 151
local_requisitado_codigo_postal	VARCHAR(75)	2*75+1 = 151
data_requisitada	DATETIME	8
preco_max	DOUBLE	8
tipo	VARCHAR(75)	2*75+1 = 151
descricao	VARCHAR(75)	2*75+1 = 151
Cliente_id	INT	4

Tabela 13 - Tamanho Entidade Pedido

Evento_Tem_funcionario

ATRIBUTOS	**TIPOS DE DADOS**	**TAMANHO** (bytes)
Evento_id	INT	4
Evento_Pedido_id	INT	4
Evento_Pedido_Cliente_id	INT	4
Funcionario_id	INT	4

Tabela 14 - Tamanho Entidade Evento_Tem_Funcionario

Inicialmente considerando que a empresa tem cerca de 10 clientes, 10 pedidos, 5 funcionários, 6 eventos e 10 funcionários alocados para determinados eventos, definindo assim um total de 10 * (4 + 151 + 4) + 10 * (4+ 8 + 151 + 151 + 151 + 8 + 3 + 4 + 3 + 4 + 4) + 5 * (4 + 151 + 4 + 151) + 6 * (4 + 151 + 151 + 8 + 8 + 151 + 151 + 4) + 10 * (4 + 4 + 4 + 4), dando um resultado final de 11978 bytes.

Espera se a cada ano um crescimento de pelo menos 2 funcionários, 10 novos pedidos, 10 novos clientes, 10 novos funcionários e a alocação de funcionário para eventos ser igual ao ano anterior + o número de funcionários novos * metade do número de novos eventos (para simplificar as contas), o que significa que por ano precisaríamos de uma média de

(11978) + (10 * (4 + 151 + 4) + 10 * (4+ 8 + 151 + 151 + 151 + 8 + 3 + 4 + 3 + 4 + 4) + 10 * (4 + 151 + 4 + 151) + 10 * (4 + 151 + 151 + 8 + 8 + 151 + 151 + 4) + (10 * 2 + 5)* (4 + 4 + 4 + 4)) = 28258 bytes por ano.

5.5. Indexação do Sistema de Dados

Em SQL , a utilização de índices numa Base de Dados permite que o tempo de procura de informação seja reduzido, no entanto, leva a um maior consumo de espaço de armazenamento. O SQL , por definição, gera automaticamente índices para cada objeto criado que contém uma Primary Key, denominados por índices implícitos. Contudo, além destes, é possível criar novos índices, conhecidos como índices explícitos.

- Tabela Cliente:

```
CREATE INDEX idx_cliente_id ON Cliente (id);
```

- Tabela Pedido:

```
CREATE INDEX idx_pedido_id ON Pedido (id);
CREATE INDEX idx_pedido_data_preco ON Pedido (data_requisitada, preco_max)
```

- Tabela Evento:

```
CREATE INDEX idx_evento_id ON Evento (id)
CREATE INDEX idx_evento_data_preco ON Evento (data_final, preco_final);
```

- Tabela Funcionario:

```
CREATE INDEX idx_funcionario_id ON Funcionario (id);
```

5.6. Procedimentos Implementados (alguns exemplos)

- Listar informações relativas a um evento específico

```
DELIMITER //

CREATE PROCEDURE GetEventDetails(
    IN evento_id INT,
    IN evento_pedido_id INT,
    IN evento_pedido_cliente_id INT
)
BEGIN
    SELECT *
    FROM Evento
    WHERE id = evento_id
        AND Pedido_id = evento_pedido_id
        AND Pedido_Cliente_id = evento_pedido_cliente_id;
END //

DELIMITER ;
```

Figura 31 - Procedure 1

Para os parâmetros (1,1,1):

```
CALL GetEventDetails(1, 1, 1);
```

Figura 32 - Chamada da Procedure 1

Produzindo o respetivo resultado:

id	data_final	tipo	local_final_rua	local_final_codigo_postal	preco_final	hora_final	n_pessoas	hora_inicio	Pedido_id	Pedido_Cliente_id
1	2023-01-20 17:00:00	Concerto	Rua A	12345	50000	20:00:00	100	17:00:00	1	1

Figura 33 - Resultado da Procedure 1

- Contar os eventos de um funcionário

```
DELIMITER //

CREATE PROCEDURE CountEventsForFuncionario(
    IN f_id INT
)
BEGIN
    SELECT COUNT(*) AS TotalEvents
    FROM Evento_tem_Funcionario
    WHERE Funcionario_id = f_id;
END //

DELIMITER ;
```

Figura 34 - Procedure 2

Para os parâmetros (2):

```
CALL CountEventsForFuncionario(1);
```

Figura 35 - Chamada da Procedure 2

Produzindo o respetivo resultado:

TotalEvents
2

Figura 36 - Resultado da Procedure 2

- Atualizar o preço de um pedido

```
DELIMITER //

CREATE PROCEDURE UpdatePrecoMax(
    IN pedido_id INT,
    IN novo_preco_max DOUBLE
)
BEGIN
    UPDATE Pedido
    SET preco_max = novo_preco_max
    WHERE id = pedido_id;
END //

DELIMITER ;
```

Figura 37 - Procedure 3

Para os parâmetros (1, 150.00):

```
CALL UpdatePrecoMax(1, 150.00);
```

Figura 38 - Chamada da Procedure 3

Produzindo o respetivo resultado:

```
select * from Pedido where id = 1
```

Figura 39 - Chamada para visualização de resultado

Antes:

id	local_requisitado_rua	local_requisitado_codigo_postal	data_requisitada	preco_max	tipo	descricao	Cliente_id
1	Rua A	12345	2023-01-10 08:00:00	50100	Concerto	NULL	1

Depois:

id	local_requisitado_rua	local_requisitado_codigo_postal	data_requisitada	preco_max	tipo	descricao	Cliente_id
1	Rua A	12345	2023-01-10 08:00:00	150	Concerto	NULL	1

Figura 40,41 - Resultado da Procedure 3 (antes e depois)

5.7. Funções

Dedução do valor que se tem a pagar a um determinado funcionário pelas horas prestadas em todos os eventos em que esteve.

```
DELIMITER //

CREATE FUNCTION CalculateTotalSalaryForFuncionario(f_id INT) RETURNS DECIMAL(10,2)
DETERMINISTIC
BEGIN
    DECLARE total_salary DECIMAL(10,2);

    SELECT SUM(TIMESTAMPDIFF(SECOND, e.hora_inicio, e.hora_final) / 3600 * 6)
    INTO total_salary
    FROM Evento e
    INNER JOIN Evento_tem_Funcionario ef ON e.id = ef.Evento_id
        AND e.Pedido_id = ef.Evento_Pedido_id
        AND e.Pedido_Cliente_id = ef.Evento_Pedido_Cliente_id
    WHERE ef.Funcionario_id = f_id;

    RETURN total_salary;
END //

DELIMITER ;

SELECT CalculateTotalSalaryForFuncionario(2) AS TotalSalary;
```

Figura 42 - Função de pagamentos

Produzindo o respetivo resultado:

	TotalSalary
▶	36.00

Figura 43 - Resultado da Função

5.8. Plano de segurança e recuperação de dados

Uma vez que a base de dados é relativamente pequena, podem se fazer backups diários.

O plano de segurança, contra a potencial perda de dados ou, até mesmo, contra a existência de erros que possam corromper a base de dados, consiste na realização de backups frequentes do projeto.

No MySQL recorremos às seguintes funcionalidades: Data export -> selecionar a base de dados -> Self contained file -> Start export e, após realizados esses passos, a base de dados encontra-se guardada num script SQL.

Assim, caso seja necessário efetuar a recuperação da base de dados, seguimos os seguintes passos: Data Import/restore -> Selecionar a script pretendida -> Start import e, depois disto, fica concluída a recuperação da base de dados

No que toca à segurança de dados, utilizamos uma flag ‘isDeleted’, que nos permite “apagar” um determinado registo/dado sem apagar definitivamente esse mesmo da base de dados, apenas marcando como estando apagado.
Os timestamps de criação e atualização são úteis para debug, isto é, aquando da necessidade de se fazer uma manutenção do sistema, é possível, saber quando os registos foram feitos.

6. Conclusões e Trabalho Futuro

A criação e implementação da Base de Dados Relacional para a empresa de organização e divulgação de eventos, City Dreams, apresenta uma solução robusta e estruturada para gerir eficientemente todas as informações relacionadas aos clientes, pedidos, eventos e funcionários. Ao longo do processo, seguimos uma abordagem detalhada desde a definição do projeto até a implementação física no sistema de gestão de bases de dados MySQL.

O levantamento e análise de requisitos foram conduzidos de forma abrangente, envolvendo reuniões com a equipa da empresa e a aplicação de questionários para identificar necessidades e melhorias. A modelação conceptual e lógica foi cuidadosamente realizada, culminando num Diagrama ER e num modelo lógico normalizado que atendem às três formas normais.

A implementação física no MySQL foi guiada pelos esquemas lógicos, resultando na criação de tabelas e vistas de utilização. A validação do modelo lógico foi realizada através de consultas SQL, demonstrando a capacidade do sistema em responder às necessidades específicas da empresa.

Além disso, apresentamos exemplos de procedimentos implementados, como consultas, atualizações e contagens, proporcionando funcionalidades úteis para o utilizador. A segurança e recuperação de dados foram abordadas com a implementação de backups diários e o uso de flags para marcar registros como apagados, permitindo uma abordagem segura para a remoção de dados.

Trabalho Futuro:

O sistema pode ser expandido e melhorado com as seguintes considerações para o futuro:

Implementação da Componente de Gestão de Recursos: Como mencionado no relatório, está planeada a construção da componente de gestão de recursos para a criação de eventos a nível nacional. Esta expansão poderia incluir uma gestão mais detalhada de recursos como equipamentos, fornecedores e patrocinadores.

Melhorias na Interface do Utilizador: Desenvolver uma interface mais amigável e intuitiva para os utilizadores interagirem com a base de dados, facilitando consultas e atualizações.

Integração de Relatórios e Análises: Adicionar funcionalidades de relatórios e análises para fornecer insights detalhados sobre o desempenho dos eventos, gastos de clientes, entre outros.

Implementação de Restrições de Acesso: Reforçar a segurança implementando restrições de acesso mais granulares, concedendo permissões específicas com base nos papéis dos utilizadores.

Otimização de Desempenho: Avaliar e otimizar o desempenho da base de dados à medida que ela cresce, considerando índices, particionamento e outras técnicas.

O desenvolvimento contínuo dessas áreas contribuirá para uma gestão mais eficiente e aprimorada dos eventos pela City Dreams.

7. Bibliografia

BD Base de Dados

DW Data Warehouse

OLTP On-Line Analytical Processing

Connolly, T., Begg, C., 2014. Database Systems: A Practical Approach to Design, Implementation, and Management. Addison-Wesley, Global Edition.

Belo, O., 2021. Bases de Dados Relacionais: Implementação com MySQL. FCA, Editora de Informática. Gouveia, F., 2021.

MySQL, 2021. Reference Manual. [online] Available at: https://dev.mysql.com/doc/refman/8.0/en/