UNIVERSIDADE FEDERAL DE SANTA MARIA

CURSO DE CIÊNCIA DA COMPUTAÇÃO

Relatório Técnico de Implementação em Banco de Dados

Disciplina de Fundamentos de Banco de Dados

Docente: João carlos Damasceno Lima

Carlos Eduardo Velozo
Davi de Castro Machado

Santa Maria, RS

Introdução:

Apresentação de um relatório técnico sobre o desenvolvimento do projeto final da disciplina Fundamentos de Banco de Dados, destacando as experiências e aprendizados obtidos ao utilizar uma das ferramentas sugeridas e uma linguagem a nossa escolha para a implementação de um banco de dados. O documento explora os desafios enfrentados, as soluções adotadas e as contribuições práticas da ferramenta para o desenvolvimento do sistema, proporcionando uma análise crítica do processo.

Descrição do Projeto Escolhido:

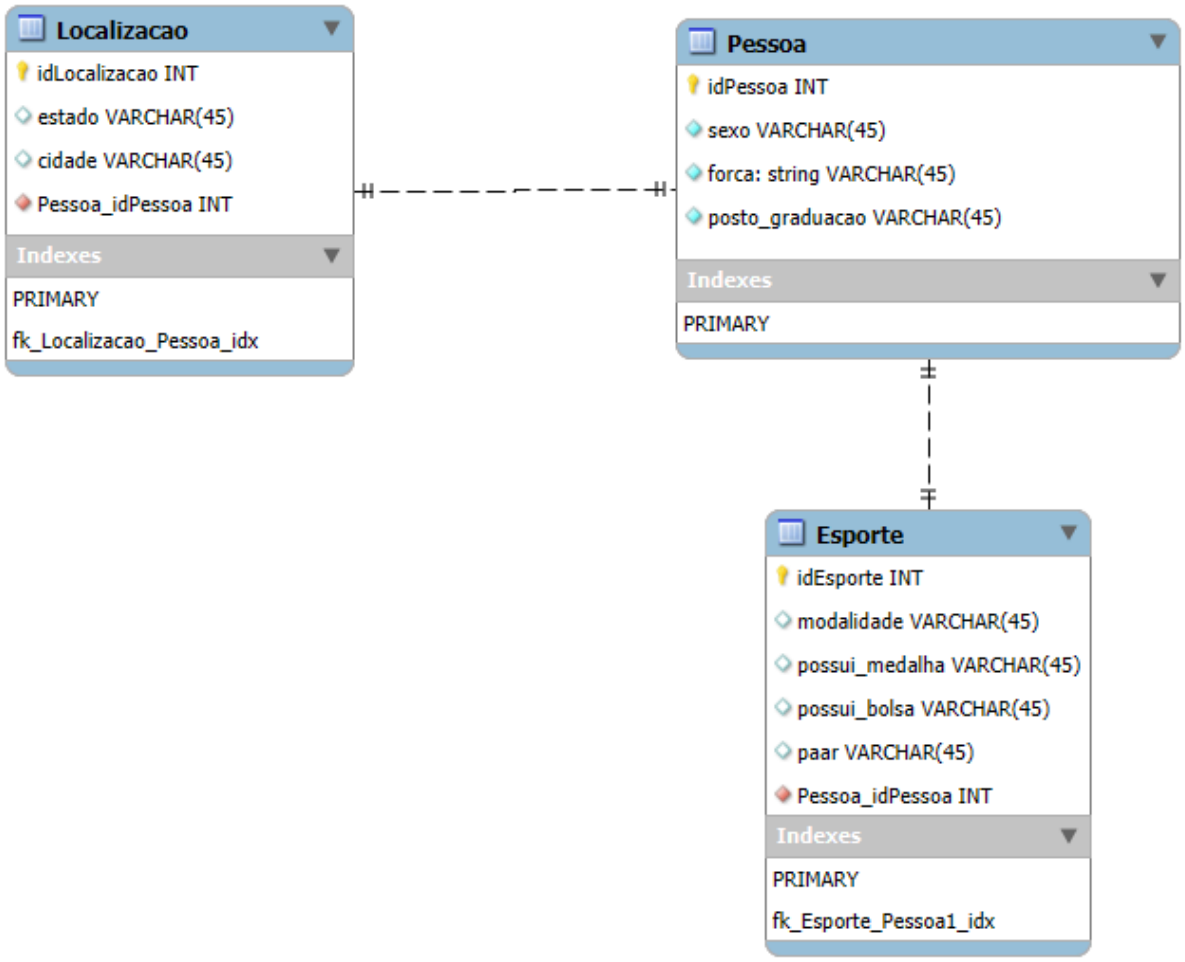
O Programa Atletas de Alto Rendimento (PAAR) foi criado em 2008 com o objetivo de fortalecer a equipe militar brasileira em competições esportivas de alto nível. O programa é uma parceria entre o Ministério da Defesa, o Comitê Olímpico do Brasil (COB), o Ministério da Cidadania, clubes esportivos, além de confederações e federações esportivas, formando uma engrenagem que contribui significativamente para o sucesso do Projeto Olímpico Brasileiro.

Por meio do PAAR, atletas militares recebem apoio e condições para competirem em alto nível, representando o Brasil em competições nacionais e internacionais, enquanto também cumprem funções dentro das Forças Armadas.

Para o projeto da disciplina Fundamentos de Banco de Dados, escolhemos o PAAR como estudo de caso para modelar um banco de dados, utilizando o último conjunto de dados disponível de setembro de 2024.

Diagrama de Classe

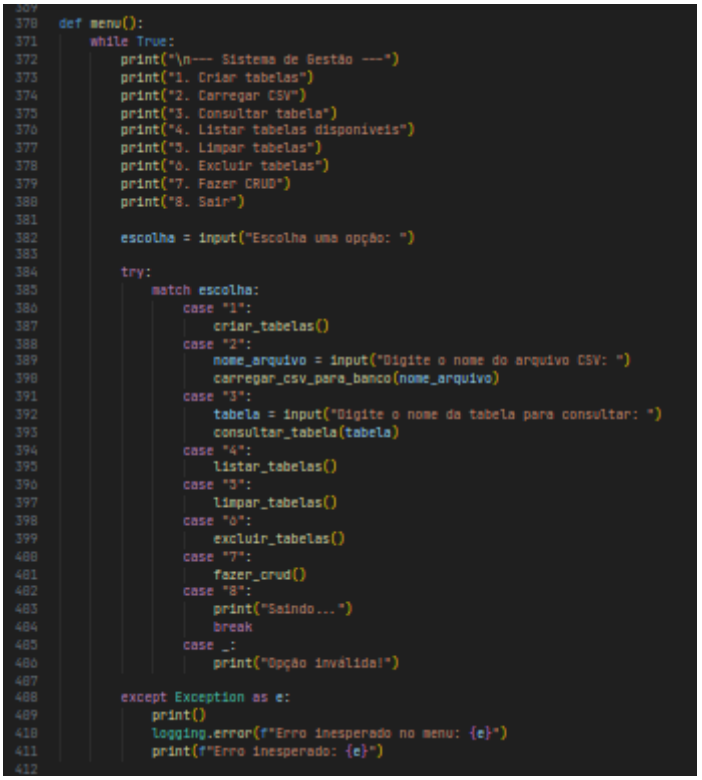
Com base no estudo de caso do Programa Atletas de Alto Rendimento, foram identificados e organizados os seguintes dados para a elaboração do Diagrama Entidade-Relacionamento (DER). O modelo contempla três tabelas principais, cada uma projetada para armazenar informações específicas relacionadas ao programa:



Essas tabelas foram implementadas e modeladas no MySQL Workbench, resultando em um DER que reflete a estrutura lógica do programa PAAR. O modelo permite um gerenciamento eficiente dos dados e facilita a análise das informações para fins de acompanhamento.

Funções implementadas:

Utilizamos da linguagem python para criação e organização das tabelas do conjunto de dados escolhido. O programa se baseia no formato SQL. A seguir estão as principais funções implementadas:



```python
def menu():
    while True:
        print("\n--- Sistema de Gestão ---")
        print("1. Criar tabelas")
        print("2. Carregar CSV")
        print("3. Consultar tabela")
        print("4. Listar tabelas disponiveis")
        print("5. Limpar tabelas")
        print("6. Excluir tabelas")
        print("7. Fazer CRUD")
        print("8. Sair")

        escolha = input("Escolha uma opção: ")

        try:
            match escolha:
                case "1":
                    criar_tabelas()
                case "2":
                    nome_arquivo = input("Digite o nome do arquivo CSV: ")
                    carregar_csv_para_banco(nome_arquivo)
                case "3":
                    tabela = input("Digite o nome da tabela para consultar: ")
                    consultar_tabela(tabela)
                case "4":
                    listar_tabelas()
                case "5":
                    limpar_tabelas()
                case "6":
                    excluir_tabelas()
                case "7":
                    fazer_crud()
                case "8":
                    print("Saindo...")
                    break
                case _:
                    print("Opção inválida!")

        except Exception as e:
            print()
            logging.error(f"Erro inesperado no menu: {e}")
            print(f"Erro inesperado: {e}")
```

Esse é o menu principal de nosso script, aqui temos várias opções de escolha como por exemplo:

- Fazer CRUD: que levará a um submenu com escolhas de inclusão, remoção, alteração e leitura. Todas essas funcionalidades já foram implementadas.
- Listar Tabelas: chama uma função que retorna print do nome das tabelas disponíveis.
- Carregar Tabelas a partir de um arquivo csv: chama a função *process_csv* que normaliza e filtra inconsistências do arquivo para aí sim carregá-lo nas tabelas.
- Excluir Tabelas, Limpar Tabelas, entre outros.

Normalização de dados csv

A normalização se deu com o uso dos seguintes métodos:

- Remoção de tags html.
- Remoção de acentos e substituição de ‘ç’ por ‘c’ por exemplo.
- Conversão para UTF-8.
- Uso de ‘;’ como delimitador.

Função *process_csv* :

```python
def process_csv(input_file, output_file):
    """
    Processa um arquivo CSV, remove tags HTML, normaliza texto (acentuação, 'ç' para 'c')
    e ajusta delimitadores (de ponto e vírgula para o delimitador desejado).
    """
    def remove_html_tags(text):
        """Remove tags HTML de uma string."""
        return re.sub(r'<[^>]+>', '', text)

    def normalize_text(text):
        """Remove acentuações e substitui 'ç' por 'c'."""
        if not text:
            return None
        normalized = unicodedata.normalize('NFD', text)
        without_accents = ''.join(char for char in normalized if unicodedata.category(char) != 'Mn')
        return without_accents.replace('ç', 'c').replace('Ç', 'C')

    def converter_para_utf8(nome_arquivo):
        """Converte um arquivo CSV para UTF-8."""
        arquivo_normalizado = os.path.join(os.getcwd(), "normalized_" + nome_arquivo)

        try:
            dados = pd.read_csv(nome_arquivo, sep=";", encoding="latin1")
            # Normalizar colunas
            dados.columns = dados.columns.str.strip().str.replace(r'\s+', ' ', regex=True)
            dados.to_csv(arquivo_normalizado, index=False, encoding="utf-8")
            logging.info(f"Arquivo '{nome_arquivo}' convertido para UTF-8 com sucesso.")
            return arquivo_normalizado
        except Exception as e:
            logging.error(f"Erro ao converter arquivo: {e}")
            raise

    # Converter o arquivo original para UTF-8
    converter_para_utf8(input_file)

    # Processar e salvar o arquivo normalizado
    with open(input_file, mode='r', encoding='latin-1') as infile, \
         open(output_file, mode='w', encoding='utf-8', newline='') as outfile:

        reader = csv.reader(infile, delimiter=';')  # Ajuste para ponto e vírgula no CSV original
        writer = csv.writer(outfile, delimiter=';')  # Preserva o delimitador na saída

        # Processar linha por linha
        for row in reader:
            # Normalizar o conteúdo de cada célula
            cleaned_row = [normalize_text(remove_html_tags(cell)) for cell in row]
            writer.writerow(cleaned_row)

    logging.info(f"Arquivo '{input_file}' processado e salvo como '{output_file}'.")

```

Configurações do banco de dados

Para utilizar o código será necessário que você altere os valores presentes no início do script para que possa então acessar um banco de dados de seu domínio.

```python
# Configurações do banco de dados
DB_CONFIG = {
    "host": os.environ.get("DB_HOST", "localhost"),
    "user": os.environ.get("DB_USER", "kiri"),
    "password": os.environ.get("DB_PASSWORD", "darius15"),
    "database": os.environ.get("DB_NAME", "fdb1")
}

```

Utilizamos do MySQL WorkBench para a criação do script que implementa as *Views* e *Triggers.* A seguir estão as funções:

```sql
CREATE VIEW vw_pessoa_localizacao AS
SELECT
    p.id AS pessoa_id,
    p.sexo,
    p.forca,
    p.posto_graduacao,
    l.estado,
    l.cidade
FROM Pessoa p
JOIN Localizacao l ON p.id = l.id;

CREATE VIEW vw_pessoa_esporte AS
SELECT
    p.id AS pessoa_id,
    p.sexo,
    p.forca,
    p.posto_graduacao,
    e.modalidade,
    e.possui_medalha,
    e.possui_bolsa,
    e.paar
FROM Pessoa p
JOIN Esporte e ON p.id = e.id;
```

```sql
CREATE VIEW vw_esporte AS
SELECT
    e.id AS esporte_id,
    e.modalidade,
    e.possui_medalha,
    e.possui_bolsa,
    e.paar
FROM Esporte e;
```

As views criadas no banco de dados do PAAR têm a finalidade de facilitar a consulta e análise dos dados. A view *vw_pessoa_localizacao* combina informações de pessoas com sua localização, detalhando atributos como sexo, força militar, posto/graduação, estado e cidade. A view *vw_pessoa_esporte* relaciona dados pessoais com esportes, incluindo modalidade esportiva, conquistas de medalhas e recebimento de bolsas. Por fim, a view *vw_esporte* apresenta informações específicas sobre esportes, como modalidade, medalhas, bolsas e vínculo com o programa PAAR. Essas estruturas otimizam o acesso a informações integradas para análises detalhadas.

```
DELIMITER //

CREATE TRIGGER trg_validate_modalidade
BEFORE INSERT ON Esporte
FOR EACH ROW
BEGIN
    IF NEW.modalidade NOT IN (
        'Apneia', 'Atletismo', 'Basquete', 'Boxe', 'Canoagem Slalom',
        'Canoagem Velocidade', 'Ciclismo MTB', 'Escalada Esportiva', 'Esgrima',
        'Futebol', 'Ginastica Artistica', 'Golfe', 'Judo', 'Levantamento de Peso',
        'Lifesaving', 'Lutas Associadas (Wrestling)', 'Maratona', 'Maratonas Aquaticas',
        'Nado Sincronizado', 'Natacao', 'Orientacao', 'Paraquedismo', 'Pentatlo Militar',
        'Pentatlo Moderno', 'Pentatlo Naval', 'Pesca Submarina', 'Taekwondo', 'Tiro',
        'Tiro com Arco', 'Triatlo', 'Vela', 'Voleibol', 'Volei de Praia'
    ) THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Modalidade inválida. Insira uma modalidade válida!';
    END IF;
END;

//

DELIMITER ;
```

A função do gatilho *trg_validate_modalidade* é validar as inserções na tabela Esporte, garantindo que apenas modalidades esportivas pré-definidas sejam registradas. As modalidades escolhidas foram baseadas nos conjuntos de dados fornecidos pelo PAAR. Antes de cada operação de inserção, o gatilho verifica se o valor da coluna modalidade está na lista de modalidades aceitas, como Atletismo, Futebol, Judô, entre outras. Caso a modalidade inserida não esteja na lista, o gatilho interrompe a operação e retorna uma mensagem de erro: "Modalidade inválida. Insira uma modalidade válida!". Essa validação automática assegura a consistência e integridade dos dados relacionados às modalidades esportivas do PAAR.

Facilidades/Dificuldades no desenvolvimento do trabalho:

No início do desenvolvimento do trabalho, enfrentamos desafios relacionados ao uso do MySQL por meio da ferramenta MySQL Workbench, já que essa foi nossa primeira experiência tanto com modelagem de dados quanto com a criação de scripts em SQL. A familiarização com a interface e os comandos exigiu tempo e esforço, o que inicialmente dificultou o progresso. Contudo, à medida que exploramos a ferramenta e aplicamos os conceitos aprendidos, fomos superando as dificuldades e ganhando confiança no uso do ambiente.

Durante o desenvolvimento do código em python percebemos alguns desafios não tão visíveis, algumas nuances a respeito de uso de ‘id’, tratamento de valores nulos e regras de entrada do usuário. Além disso, é de certa forma desafiador lidar com duas linguagens distintas ao mesmo tempo, alternando entre python e SQL, o que exige muita concentração do programador.

Por fim, um desafio mais geral relacionado ao trabalho foi em relação ao código ficar extenso e de difícil reutilização, com funções muito semelhantes porém distintas e com peculiaridades únicas.