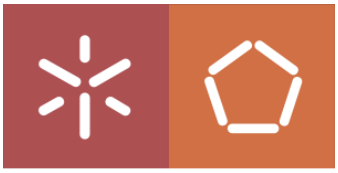

Universidade do Minho
Escola de Engenharia

Departamento de informática

UC: Introdução aos algoritmos, à programação e à base de dados

Mestrado de Bioinformática

Implementação de uma base de dados

Bruno Sá PG48932

Luís Ferreira PG49840

Miguel Valério PG49845

1 Introdução

1.1 Contextualização, motivação, objetivos

Na medicina e investigação, é importante termos alguma forma de reunirmos toda a informação detalhada que constitui um organismo, de forma a termos um acesso mais facilitado e direto a todos os pontos chaves pertinentes da nossa espécie de estudo. Desta forma é possível resolver diferentes problemas que pretendamos solucionar, presentes na própria espécie ou fazendo ligação com problemas que ocorrem em outros seres vivos. Com isto, surgiram os ficheiros Genbank, que representam um banco de dados de sequências genéticas do NIH, em há armazenamento de toda a informação pertinente para investigar um organismo.

Para obtermos o Genbank, podemos utilizar a plataforma Pubmed, que corresponde a um motor de busca da National Library of Medicine (NLM), que reúne registos da base de dados MEDLINE (principal base produzida pela NLM) e registos únicos da PubMed. Esta plataforma vai ser responsável por mostrar vários resultados para o organismo em estudo, tais como o Genbank da espécie pretendida ou artigos relacionados com a mesma. Podemos então utilizar estes ficheiros Genbank de forma a relacionar diferentes sequenciamentos de DNA, em que pode ser benéfico para reconhecer, detetar e desenvolver tratamentos para doenças genéticas ou em pesquisas que estejam presentes organismos infeciosos, em que o sequenciamento pode levar a tratamentos para doenças contagiosas.

Como tal, o nosso objetivo do trabalho vai ser criar uma base de dados relativamente ao ficheiro Genbank e ao PubMed do organismo pretendido, em que vai ser possível retirar informação pertinente de forma rápida, permitindo também responder a perguntas complexas sobre possíveis problemas em causa.

2. Tabelas de atributos e relacionamentos

A criação das seguintes tabelas vai ter como objetivo facilitar a organização e planificação da informação presente no ficheiro Genbank, para depois ser usada na criação dos diferentes modelos.

Atributos					
Entidade/ relacionamento	**Nome**	**Tipo e domínio**	**Multivalor**	**Nulos**	**Chave primária**
Genbank	ID_version_seq	String(50)	-	N	S
	Organism	String(200)	-	N	N
	Acession	String(50)	-	N	N
	Definition	String(200)	-	S	N
	KeyWords	String(200)	-	S	N
	ID_locus	String(50)	-	N	N
Locus	ID	String(50)	-	N	S
	Size	Int	-	N	N
	Molecular_type	String(50)	-	N	N
	Genbank_division	String(3)	-	N	N
	Modification_date	String(20)	-	N	N
Sequences	ID_version_seq	String(50)	-	N	S
	Sequence	Long Text	-	N	N
	Count A	Int	-	S	N
	Count C	Int	-	S	N
	Count T	Int	-	S	N
	Count G	Int	-	S	N
Features	ID_version_genbank	String(50)	-	N	S
	CDS_count	Int	-	N	N
	Gene	Int	-	S	N
	Regulatory	Int	-	S	N
	Exons	Int	-	S	N
	Poly_A_site	Int	-	S	N
	Misc_feature	Int	-	S	N
	mRNA	Int	-	S	N
	Location_span	String(45)		N	N
CDS	ID_protein_cds	String(50)	-	N	S

	Translacion seq	String (10000)	-	N	N
	Localization	String(2000)	-	N	N
	ID_version_features	String(50)	-	N	N
	Protein	String(2000)	-	N	N
Genbank_reference	ID_version_connect	String(45)	-	N	S
	Journal_connect	String(200)	-	N	S
Authors	Names	String(45)	-	N	S
References	Journal_ID	String(200)	-	N	S
	Pubmed	String(50)	-	S	N
	Title	String(200)	-	S	N
	Consortium	String(200)	-	S	N
	Remark	String(200)	-	S	N
Pubmed _Info	ID_journal	String(200)	-	N	S
	Title	String(200)	-	N	N
	DOI	String(50)	-	N	N
	Affiliation	String(200)	-	N	N
	Abstract	String(2000)	-	N	N
Reference_Authors	Reference_journal_key	String(200)	-	N	S
	Authors_key	String(45)	-	N	S

Tabela 1 Criação das diferentes entidades e os seus atributos relativamente à base de dados.

Entidade	#	Relacionamento	Entidade	#
Genbank	1…1	Contém	Locus	1…1
Genbank	1…1	Contém	Sequences	1…1
Genbank	N…N	Genbank_reference	reference	N…N
Genbank	1…1	Contém	Features	1…1
Features	1…N	Contém	CDS	N…1
reference	N…N	reference_authors	Authors	N…N
reference	1…1	Contém	Pubmed info	1…1

Tabela 2 Relacionamento entre as diferentes entidades da base de dados.

3. Construção dos diferentes modelos associados à base de dados

a) Modelo conceptual

Após análise e estruturação dos dados retirados do ficheiro Genbank, formando as entidades, atributos e respetivas relações entre si, passamos à construção do modelo conceptual. Para isso escolhemos o software TerraER, devido ao seu conhecimento prévio das aulas práticas e da sua fácil utilização.

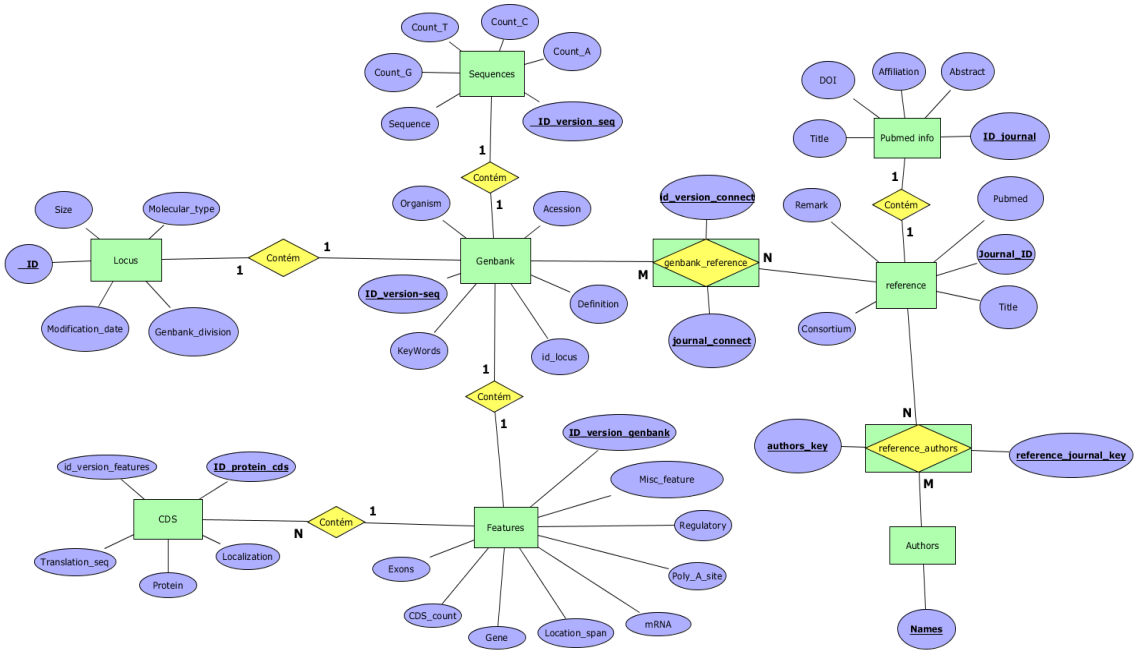

Figura 2 Esquema conceptual de um ficheiro Genbank.

b) Modelo lógico

Para realizarmos este modelo, vamos utilizar o software MySQL Workbench, sendo que para a construção da base de dados baseamo-nos na estruturação do modelo conceptual, de forma a facilitar a montagem das entidades e os seus relacionamentos.

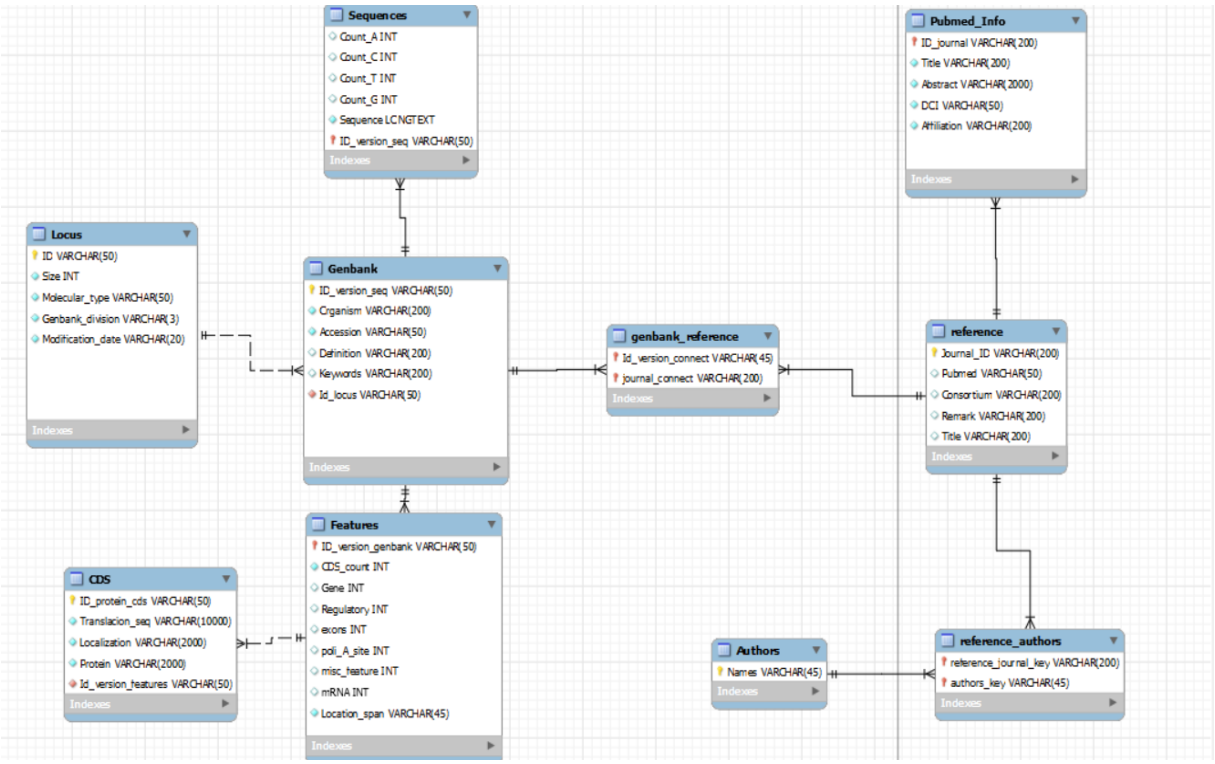

Figura 3 Esquema lógico de um ficheiro Genbank.

c) Modelo físico

O modelo físico vai corresponder à base de dados mas na forma de script, podendo ser retirada através do modelo lógico com o uso da opção forward engineering. De forma a ser possível expor o modelo físico, vamos retirar um excerto da script em que é possível visualizar a constituição de duas entidades e as caraterísticas dos seus atributos.

```
-- -----------------------------------------------------
-- Table `Base Dados IAP`.`Genbank`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `Base Dados IAP`.`Genbank` ;

CREATE TABLE IF NOT EXISTS `Base Dados IAP`.`Genbank` (
  `ID_version_seq` VARCHAR(50) NOT NULL,
  `Organism` VARCHAR(200) NOT NULL,
  `Accession` VARCHAR(50) NOT NULL,
  `Definition` VARCHAR(200) NULL,
  `Keywords` VARCHAR(200) NULL,
  `Id_locus` VARCHAR(50) NOT NULL,
  PRIMARY KEY (`ID_version_seq`),
  INDEX `fk_Genbank_1_idx` (`Id_locus` ASC) VISIBLE,
  CONSTRAINT `fk_Genbank_1`
    FOREIGN KEY (`Id_locus`)
    REFERENCES `Base Dados IAP`.`Locus` (`ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);
```

```
-- -----------------------------------------------------
-- Table `Base Dados IAP`.`Sequences`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `Base Dados IAP`.`Sequences` ;

CREATE TABLE IF NOT EXISTS `Base Dados IAP`.`Sequences` (
  `Count_A` INT NULL,
  `Count_C` INT NULL,
  `Count_T` INT NULL,
  `Count_G` INT NULL,
  `Sequence` LONGTEXT NOT NULL,
  `ID_version_seq` VARCHAR(50) NOT NULL,
  PRIMARY KEY (`ID_version_seq`),
  CONSTRAINT `fk_Sequences_1`
    FOREIGN KEY (`ID_version_seq`)
    REFERENCES `Base Dados IAP`.`Genbank` (`ID_version_seq`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);
```

Figura 4 Esquema físico de uma base de dados correspondente a um ficheiro Genbank.

4. Povoamento das tabelas do modelo físico

4.1) Ligação à base de dados

Utilizamos o módulo mysql.connector de forma a ligarmos a nossa base de dados local ao python.

```
import mysql.connector as SQLC

DataBase = SQLC.connect(
    host ="127.0.0.1",
    user ="root",
    password ="login123",
    database = "Base Dados IAP",
    auth_plugin="mysql_native_password"
)
DataBase.autocommit= True
cur = DataBase.cursor()
```

Figura 5 Script correspondente à ligação da base de dados local.

4.2) Acesso ao NCBI

Através do identificador único (accession) vamos obter todos os dados acerca do mesmo, sem necessidade de um ficheiro local. Para isto foram utilizados diversos submódulos pertencentes ao módulo BioPython, entre os quais o Entrez, que nos permitiu aceder à base de dados do NCBI e aceder à informação relativa ao nosso organismo. Foi também utilizado o SeqIO que nos permitiu ler a informação recolhida do NCBI no formato Genbank.

```
id_genbank_input = input("ID NCBI")
handle = Entrez.efetch(db='nucleotide',retmode='txt',id=id_genbank_input, rettype = "gb")
record = SeqIO.read(handle, "gb")
```

Figura 6 Script utilizada para obter a informação relativa ao ficheiro Genbank através do input da referência.

4.3. Povoamentos das tabelas

4.3.1) Povoamento da tabela Locus

O seguinte script corresponde ao povoamento da tabela Locus onde utilizamos o módulo Biopython de forma a termos um acesso rápido e direto à informação pretendida.

```
# Locus
ID = record.name
Size = len(record.seq)
Molecular_type = record.annotations["molecule_type"] + " " + record.annotations["topology"]
Genbank_division = record.annotations["data_file_division"]
Modification_date = record.annotations["date"]

try:
        cur.execute(f"""
                INSERT INTO Locus (ID, Size, Molecular_type, Genbank_division, Modification_date)
                VALUES (%s, %s, %s, %s, %s)
                """,
                    (ID, Size, Molecular_type, Genbank_division, Modification_date))

except:
        print("Erro Povoação Locus")
```

Figura 7 Script correspondente ao povoamento da tabela Locus recorrendo ao biopython e inserção das variáveis definidas na base de dados.

4.3.2) Povoamento da tabela Pubmed_info

O seguinte script corresponde ao povoamento da tabela Pubmed_info onde utilizamos as referências do PubMed, anteriormente obtidas através do ficheiro GB, acedendo às mesmas de forma a retirar a informação pretendida e recorrendo novamente ao Entrez (aceder à base de dados do Pubmed) e a expressões regulares (recolher informação pretendida).

É importante salientar que foram utilizadas listas de forma a organizar melhor a informação, visto que cada ficheiro Genbank pode conter inúmeras referências do Pubmed.

```
Abstract = []                       # mudar para cada referenias
DOI = []
Affiliation = []
for c in Pubmed:
    if c != "None":
        Entrez.email = 'sa.bruno.2001@gmail.com'
        # handle = Entrez.efetch(db='pubmed',retmode='xml',id="15496913")
        handle = Entrez.efetch(db='pubmed',retmode='xml',id=c)
        resultado = Entrez.read(handle)
        q = str(resultado["PubmedArticle"][0])

        if re.search(r"AbstractText.:\s+\['([\w\s,\.\(\)%-:]+)", q):
            o = re.search(r"AbstractText.:\s+\['([\w\s,\.\(\)%-:]+)", q)
            Abstract.append(o.group(1))
        else:
            Abstract.append("None")

        if re.search(r"'pubmed'}.,\s+StringElement.'([\d\w\./]+)" , q):
            p = re.search(r"'pubmed'}.,\s+StringElement.'([\d\w\./]+)" , q)
            DOI.append(p.group(1))
        else:
            DOI.append("None")

        if re.search(r"'Affiliation':\s+'([,\s\d\w\./]+)" , q):
            t = re.search(r"'Affiliation':\s+'([,\s\d\w\./]+)" , q)
            Affiliation.append(t.group(1))
        else:
            Affiliation.append("None")
    if c == "None":
            Abstract.append("None")
            DOI.append("None")
            Affiliation.append("None")
```

Figura 8 Criação das diferentes estruturas de dados de forma auxiliar o processo de povoamento da tabela Pubmed.

```
pubmed_count= 0
for c in Pubmed :
        if c != "None":
                cur.execute("""
                        INSERT INTO Pubmed_Info (ID_journal, Title, Abstract, DOI, Affiliation)
                        VALUES (%s, %s, %s, %s, %s)
                        """,
                        (Journal[pubmed_count], Title[pubmed_count], Abstract[pubmed_count], DOI[pubmed_count], Affiliation[pubmed_count]))
                pubmed_count += 1
        else:
                pubmed_count += 1
```

Figura 9 Script correspondente à inserção das variáveis anteriormente definidas na base de dados.

4.3.3) Povoamento das restantes tabelas

Visto que o povoamento das restantes tabelas apresentaram uma estrutura semelhante na obtenção de nova informação (biopyton/expressões regulares), não achamos necessário sobrecarregar o relatório com informação repetida, visto que esta se encontra exemplificada anteriormente e complementada no nosso Notebook presente no Git, que é possível de ser acedido através do link https://github.com/BrunoAGSa/Projeto_Algoritmos-Introducao/blob/main/IAP/Final/Final.ipynb

5. Desenvolvimento e implementação, descrição de funções, procedimentos e triggers SQL

Uma vez que utilizamos a linguagem Python para auxiliar o processo de povoamento das tabelas, decidimos que não havia necessidade de utilizar ou integrar triggers e funções diretamente na nossa base de dados, visto que a interação com a mesma decorreu através do Python, permitindo assim um melhor controlo e gestão sobre os mesmos.

6. Resolução de problemas recorrendo à base de dados

Para resolvermos problemas de informação cujo conteúdo se encontra presente nos ficheiros Genbank dos nossos organismos, utilizamos uma base de dados relacional SQL, que nos vai permitir relacionar informações de diferentes origens através de elementos comuns entre as mesmas. No caso da nossa base de dados, através do nosso ficheiro Genbank é possível aceder à informação contida no Pubmed (por exemplo).

Abaixo encontram-se dois exemplos Query que nos vao facilmente permitir obter informações acerca dos diferentes elementos presentes na nossa base de dados.

```
select DOI from Pubmed_Info
join reference on Pubmed_Info.ID_journal = reference.Journal_ID
join genbank_reference on reference.Journal_ID = genbank_reference.journal_connect
join Genbank on genbank_reference.Id_version_connect = Genbank.ID_version_seq
where Organism = "Ectocarpus siliculosus";
```

Figura 10 Query utilizada para obter o DOI do artigo Pubmed relacionado com o organismo pretendido.

```
select Organism from Genbank
join Features on Genbank.ID_version_seq = Features.ID_version_genbank
join CDS on Features.ID_version_genbank = CDS.Id_version_features
where Protein = '[''DnaA-like replication initiation protein'']'
```

Figura 11 Query utilizada para obter o organismo associado a uma proteína.

7. Análise crítica

Ao longo do desenvolvimento da nossa base de dados, deparamo-nos com várias dificuldades na implementação das relações e na implementação do modelo lógico, que foram resolvidas com auxílio do docente.

Uma possível melhoria deste projeto no futuro, iria ser complementar as informações já existentes, com informação proveniente de outras bases, tal como acontece atualmente com a base de dados do Pubmed. Um exemplo para este objetivo, em que recorreríamos a outras base de dados, tais como a Uniprot, iria ser através da utilização do atributo id_proteína já presente na tabela CDS, que iria funcionar como ligação entre ambas.