SAPIENZA
UNIVERSITÀ DI ROMA

Integrazione di dati open access di espressione proteica in un database di dati genetici

Facoltà di Ingegneria dell'Informazione, Informatica e Statistica
Dipartimento di Informatica
Corso di laurea in Informatica

Patrizio Renelli
Matricola 1937842

Relatore
Maurizio Mancini

Correlatore
Enrico Tronci

A.A. 2023-2024

Integrazione di dati open access di espressione proteica in un database di dati
Relazione di Tirocinio. Sapienza – Università di Roma

renelli.1937842@studenti.uniroma1.it

Alla mia famiglia

Sommario

Abstract

La discussione di questa relazione si concentrerà sul lavoro svolto per individuare dati di studi proteomici e genomici di quello che spesso viene considerato il "Male del Secolo" cioè il tumore.

Per definizione il tumore è una proliferazione non controllata di cellule che hanno la capacità di infiltrarsi nei normali organi e tessuti dell'organismo alterandone la struttura e il funzionamento, questo significa che con uno screening continuo e mirato si potrebbero individuare queste anomalie e iniziare tempestivamente le adeguate terapie.

Lo screening tumorale per eccellenza viene effettuato attraverso il prelievo di campioni biologici che vengono analizzati in specifici laboratori e i dati successivamente analizzati da medici specializzati in oncologia.

Per una più veloce e accurata analisi si punta a creare un modello di machine learning che possa analizzare i dati biologici dei pazienti e individuare quelli potenzialmente dannosi.

Per addestrare il modello e renderlo affidabile e completo, è necessario avere un'importante mole di dati, che derivano da repository pubbliche contenenti dati anonimizzati di analisi di altri pazienti. Nel nostro caso andremo ad analizzare il lavoro svolto per ottenere, organizzare e archiviare dati non omogenei derivanti dai repository "Genomic Data Commons" (GDC) e "Proteomic Data Commons"(PDC) offerti dal National Cancer Institute (NIH) del governo degli Stati Uniti.

Capitolo 1 Introduzione

1.1 Contesto

Negli ultimi anni stiamo assistendo ad una continua evoluzione della tecnologia a nostra disposizione, sia in ambito informatico che in quello delle tecnologie usate per l'analisi dei campioni biologici.

Lo sviluppo delle tecnologie di analisi e la loro libera circolazione attraverso dataset pubblici, hanno permesso di migliorare e rendere più rapida la ricerca in ambito biomedico.

Grazie alla vasta disponibilità dei dati offerti da questi nuovi dataset pubblici, è possibile andare a realizzare modelli di machine learning affidabili e addestrati su una quantità di dati non esigua.

I modelli di machine learning potranno essere addestrati su analisi di tipo proteomiche e genomiche, consentendo una maggiore qualità nei risultati e un'affidabilità elevata.

Negli ultimi anni vediamo un aumento sempre crescente del numero dei tumori, passando da 376.600 del 2020 a 395.000 del 2023. L'aumento del numero dei tumori per fortuna però non coincide con l'aumento di morti legate al tumore, infatti stiamo assistendo ad una forte riduzione della mortalità associata, grazie a continue campagne di screening che ci consentono di individuare un tumore o una possibile predisposizione e iniziare tempestivamente le adeguate terapie anti tumorali. Proprio per processare i dati provenienti dai continui e crescenti screening, sarà fondamentale l'utilizzo di strumenti tecnologici avanzati tra cui proprio i modelli di machine learning.

1.2 Motivazione

La disponibilità e libera circolazione di dati sulle analisi dei campioni e di terapie con i relativi esiti, ci consente di poter individuare pattern di geni o proteine che possono rappresentare "campanelli d'allarme" per la formazione di tumori. Inoltre è possibile capire quali sono le terapie più adeguate e risolutive per ciascun tipo di mutazione, evitando sprechi di tempi per la sperimentazione di terapie specifiche per il paziente.

Inoltre negli ultimi anni abbiamo assistito al perfezionamento e alla semplificazione delle tecniche di sequenziamento e analisi di espressioni geniche e proteiche, che permettono di identificare dei biomarcatori che identificano l'attività genica e proteica all'interno delle cellule e dei tessuti.

1.3 Scopo e obiettivi della tesi

Questa tesi ha come obiettivo quello di estrarre e archiviare i dati di espressioni geniche e proteiche disponibili nei data repository forniti dal National Institutes of Health (NIH) del governo americano.

I data repository analizzati sono:

- Genomic Data Commons (GDC) (https://portal.gdc.cancer.gov) contenente principalmente dati di studi genomici e in minor parte dati di studi proteomici
- Proteomic Data Commons (PDC) (https://proteomic.datacommons.cancer.gov) incentrato maggiormente sullo studio di dati proteomici

Il lavoro svolto durante il tirocinio si è focalizzato in larga parte sullo sviluppo di codice che permettesse di individuare, scaricare e archiviare localmente i dati relativi ai studi proteomici offerti dai due repository, con particolare attenzione a quelli offerti da PDC.

Nel corso delle prossime pagine, esamineremo in dettaglio il lavoro svolto per raggiungere questo obiettivo, discutendo delle implicazioni dell'utilizzo di questi dati per la medicina di precisione. Questo lavoro rappresenta un contributo alla comprensione dei biomarcatori correlati alle malattie e al potenziale impatto positivo che possono avere sulla salute umana.

Capitolo 2 Background

La diversità di tutti gli organismi viventi è generata dall'informazione biologica contenuta nel DNA. Il DNA è acido desossiribonucleico che è dato dalla combinazione di tante unità di base ripetute che si chiamano nucleotidi. Ogni nucleotide contiene una molecola di acido fosforico, uno zucchero e un composto organico chiamato base azotata.

Tutte le informazioni codificate sono strutturate come una doppia elica di DNA che viene chiamato genoma. Il genoma rappresenta l'insieme di tutte le informazioni biologiche depositate nella sequenza di DNA che sono necessarie alla generazione e al mantenimento di ogni organismo vivente.

Nel nostro studio sarà molto importante il proteoma cioè l'insieme delle proteine in una cellula che a differenza del genoma non è costante nel tempo.

Le istruzioni per la sintesi delle proteine è contenuta in particolari geni, esse vengono costituite da venti diversi amminoacidi. La formazione di una specifica proteina inizia quando alcune informazioni presenti in un gene vengono trascritte in una sequenza di RNA messaggero, RNA viene quindi usato per tradurre il messaggio dei geni, corrispondente ad una sequenza di acidi nucleici in una catena di amminoacidi.

2.1 Biomarcatori

Un biomarcatore è un indicatore di processi fisiologici, patologici o di risposte biologiche a terapie mediche. Se misurati prima di una determinata terapia, i biomarcatori possono essere impiegati per selezionare i partecipanti a una sperimentazione clinica in modo da riscontrare con precisione e rapidità eventuali malattie (anche ancora non scaturite ma con una predisposizione genetica). Se misurati successivamente alla terapia possono predire o individuare problemi di sicurezza nella terapia e rilevare effetti positivi della terapia. Inoltre aiutano a ridurre l'incertezza nella valutazione di efficienza di una terapia, fornendo previsioni quantificabili sulle sue performance e consentendo di calibrare appropriatamente le dosi.

Sono stati definiti diversi sottotipi di biomarcatori in base al loro utilizzo:

- biomarcatori prognostici aiutano a distinguere i pazienti per grado di rischio di insorgenza o alla progressione di una determinata malattia,

- biomarcatori predittivi aiutano a distinguere i pazienti sulla base della loro probabilità di risposta a particolari terapie rispetto al non sottoporsi a terapie,

- biomarcatori diagnostici aiutano a distinguere un individuo in base alla presenza o assenza di uno specifico stato fisico, fisiologico o di una malattia,

- biomarcatori farmacodinamici aiutano a rilevare la risposta biologica di un paziente sottoposto ad un determinato intervento terapeutico, utili a calibrare appropriatamente le dosi e capire quale terapia è più funzionale al caso

2.2 Proteoma

Si è stimato che l'uomo possieda all'incirca 20.000 geni, invece il contenuto proteico supera il milione e varia in base alla parte del corpo dove il campione viene estratto e dalle varie fasi del ciclo vitale. Il proteoma indica quindi l'insieme di tutti i possibili prodotti proteici espressi in una cellula, un tessuto o un organismo complesso.

2.3 Proteomica

La proteomica consiste nell'identificazione e nell'analisi di proteine, considerando la loro struttura, funzione, attività, quantità e interazioni molecolari

La proteomica si sviluppa su tre diversi livelli:

- proteomica di espressione, per l'identificazione di pattern di espressione, usato per distinguere i pazienti sani da quelli malati,
- proteomica strutturale, per la caratterizzazione delle proteine a livello strutturale e sfrutta tecniche quali cristallografia e NMR,
- proteomica funzionale, comprende lo studio delle interazioni tra proteine, lo studio delle interazioni tra una proteina ed i suoi substrati e lo studio delle funzioni specifiche delle proteine

Tra i dati più importanti per gli studi proteomici troviamo il rapporto tra un'aliquota presa in analisi e un campione di riferimento degli ioni peptidici associati al gene che viene solitamente espressione in log2.

I peptidi rappresentano una singola catena lineare di residui di amminoacidi (composti organico) di modeste dimensioni (20/30 amminoacidi).
Lo studio dei peptide ci consente di dire se dei geni sono dedotti quando essi hanno almeno due identificazioni petidiche non condivise (unshared peptides).

Con "distinct peptides" esprimiamo invece il numero di peptidi distinti, intendendolo come il numero di peptido, indipendentemente dalla sua lunghezza, che esistono in una sola proteina di un determinato proteoma analizzato, non escludendo la possibilità che lo stesso peptide appaia più volte nella stessa proteina.

Inoltre troviamo informazioni sullo "spectral count" cioè una strategia per quantificare le concentrazioni proteiche nelle miscele proteiche pre-elaborate analizzate utilizzando cromatografia liquida e la spettrometria di massa. La spettrometria di massa si basa sulla possibilità di separare una miscela di ioni in funzione del loro rapporto massa/carica attraverso campi magnetici.

2.4 Funzionamento Machine Learning

Il machine learning cioè lo sviluppo di algoritmi e modelli che consentono ai computer di identificare dai dati determinati pattern per poi effettuare previsioni o decisioni basandosi sul loro apprendimento e affinando nel tempo le loro scelte.

Lo scoglio più complesso del machine learning, oltre la scrittura del codice degli algoritmi, consiste nel reperire i dati su cui effettuare l'addestramento dei modelli.

Per un utente comune i 42 TB di dati offerti da PDC possono sembrare una mole di dati enorme, ma nel mondo dell'intelligenza artificiale rappresentano una goccia nell'acqua, ma che nell'ambito della ricerca del cancro rappresentano una buona base di dati da cui partire, vista la difficoltà di reperire questa tipologia di dati.

I dati offerti dai due repository (PDC, GDC) offrono dati totalmente anonimizzati, aspetto particolarmente importante in ambito sanitario, visto il diritto della privacy.

Inoltre tutti i dati sono offerti in modo libero, andando a risolvere la questione di possibili problemi legali causati dal copyright.

L'aspetto dell'utilizzo di articoli, dati e informazioni coperti da copyright è oggi al centro di molti dibattiti pubblici, dove i detentori dei diritti chiedono un compenso per l'utilizzo dei loro beni utilizzati per l'addestramento dei modelli di intelligenza artificiale. Quest'ultima problematica ha portato alla creazione di modelli di intelligenza artificiale progettati per creare "dati artificiali" da utilizzare per l'addestramento di altri modelli, in modo da usare esclusivamente dati robusti, anonimizzati e liberi.

Il machine learning si basa su delle fasi ben precise:

1. Raccolta dei dati: consiste nel raccogliere un dataset abbastanza ampio e variegato da usare per addestrare i nostri modelli,
2. Preparazione dei dati: bisogna cercare di armonizzare i dati, eliminando dati duplicati, inconsistenti e tutti i dati che potrebbero alterare il corretto apprendimento dei nostri modelli, come ad esempio parametri totalmente fuori da un range comune, che potrebbero a causa del loro peso alterare l'apprendimento,
3. Scelta del modello: una volta che abbiamo i dati dobbiamo decidere quale modello utilizzare, che andrà ad identificare la strategia che si vuole adottare, come ad esempio alberi decisionali, reti neurali artificiali, regressione lineare ed ecc,
4. Addestramento del modello: consiste nella fase fondamentale in cui il modello attraverso degli algoritmi otterrà i dati e inizierà il suo apprendimento, andando ad affinare il suo addestramento ad ogni ciclo,
5. Valutazione del modello: dopo aver addestrato il modello sarà necessario verificare l'efficacia del nostro modello, utilizzando un dataset differente da quello di addestramento andremo a verificare se i risultati ottenuti dalle previsioni/decisioni del modello sono in linea con quelle da noi attese,

6. Ottimizzazione del modello: se la fase di valutazione del modello non è stata soddisfacente sarà necessario ottimizzare e modificare il modello in base alle nostre attese, andando ad iterare la fase di valutazione e ottimizzazione fino ad ottenere un risultato soddisfacente
7. Rilascio del modello: se il modello soddisfa i nostri requisiti allora sarà possibile rilasciare il nostro modello e utilizzarlo con dataset diversi da quello di addestramento

Questo progetto di tirocinio ha come obiettivo quello di assolvere ai compiti di:

- individuare delle fonti di dati
- raccogliere i dati necessari ad un futuro modello di machine learning
- preparare i dati in modo da rimuovere duplicati, inconsistenze e dati superflui per l'addestramento del modello

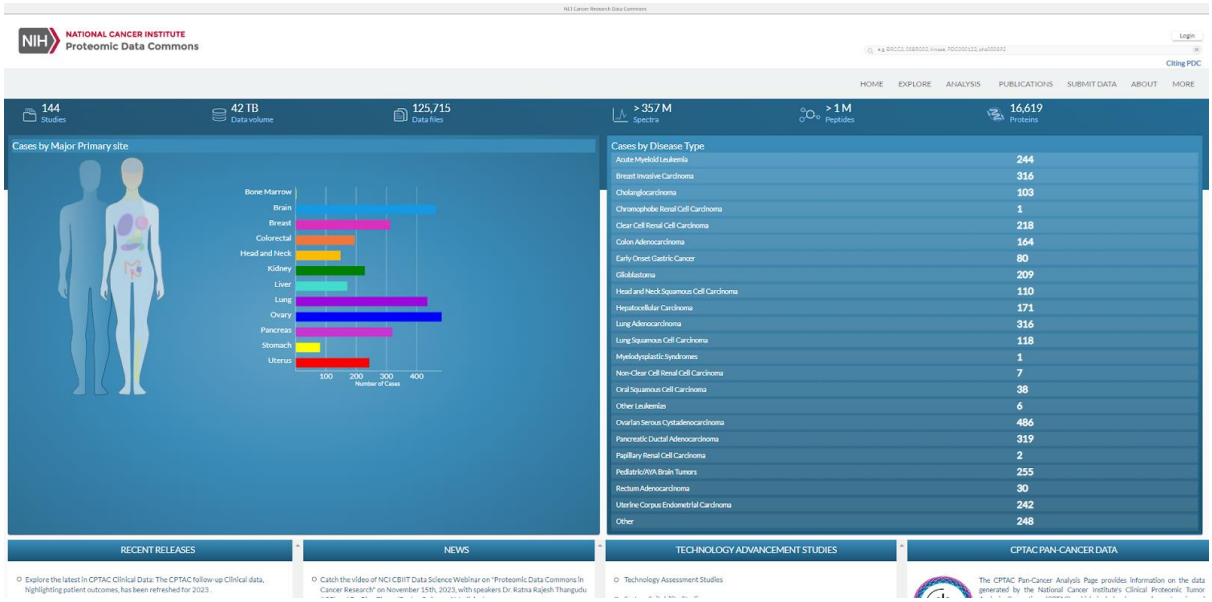
Capitolo 3 Repository e Dati

In questo capitolo andremo ad analizzare i repository e i dati che essi ci offrono, andando a descrivere e analizzare la struttura del database che utilizzeremo per la loro archiviazione

3.1 Repository

Come fonte principale dei dati relativamente ai geni e di conseguenza delle proteine ad essi associati, che serviranno per addestrare i modelli di machine learning utilizziamo il repository "Proteomic Data Commons" (PDC).

Questo repository è popolato e gestito dal National Institutes of Health (NIH) del dipartimento per la Salute e i servizi alla persona degli Stati Uniti.



(Homepage sito PDC - https://proteomic.datacommons.cancer.gov/)

Dalla homepage possiamo già renderci conto della mole di dati offerti da questo repository.

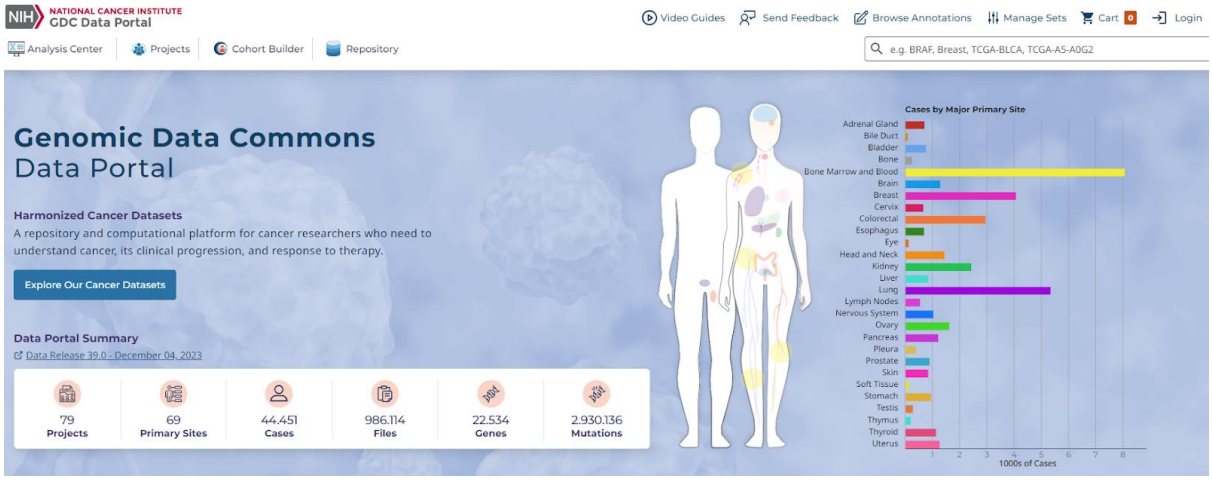
I dati che possiamo rapidamente individuare sono:

- 144 studi registrati
- 42 TB come dimensione totale del dataset offerto
- 16.619 proteine registrate

La decisione di scegliere questo dataset è intrinseca nei dati sopra riportati, poiché grazie alla vasta mole di dati offerta (anche se non tutti da noi accessibile) consente un migliore apprendimento da parte dei modelli di machine learning, inoltre il dataset viene aggiornato in modo da integrare i nuovi studi pubblicati, fornendo in questo modo la possibilità di affinare maggiormente i modelli.

Come fonte secondaria di dati di studi proteomici si è utilizzato il repository "Genomic Data Commons" (GDC) anche questo repository è offerto dal National Institutes of Health (NIH).

Questo secondo dataset è focalizzato maggiormente sugli studi genomici ma contiene in seconda battuta anche dati di studi proteomici.



(Homepage sito GDC- https://portal.gdc.cancer.gov/)

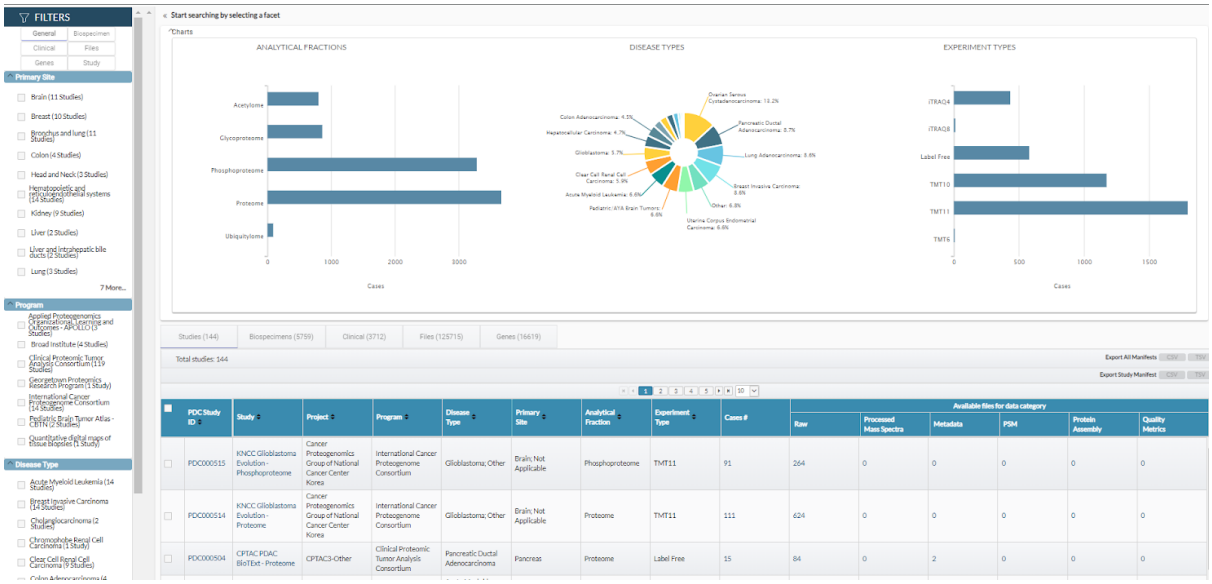
Dalla homepage possiamo individuare rapidamente che GDC offre dati relativi a:

- 79 progetti
- 69 primary site
- 44.451 cases
- 22.534 genes

3.2 Struttura generale dei dati offerti dai repository

I dati che si possono visualizzare sul sito PDC riguardano principalmente le seguenti entità di maggiore interesse e fondamentali:

- Study: oggetti che identificano i vari studi sui dati genomici e proteomici;
- Case: oggetti che identificano un paziente, collegando i suoi dati al contesto di uno specifico studio;
- Primary site: oggetti che identificano i site su cui si focalizza uno studio o da cui si estrae un campione;
- Disease: oggetti che identificano una malattia specifica;
- Biospecimen: oggetti che rappresentano dei campioni biologici estratti da un paziente;
- Aliquot: oggetti che identificano porzioni di una parte di campione prelevato da un paziente;
- Sample: oggetti che identificato un campione di materiale biologiche ottenuto per effettuare la ricerca, questi campioni possono essere estratti sia da organismi in viventi sia da organismi le cui attività vitali sono terminate;
- Gene: oggetti che identificano i geni individuati nei vari studi;
- Protein: oggetti che identificano le proteine associate ad un determinato gene

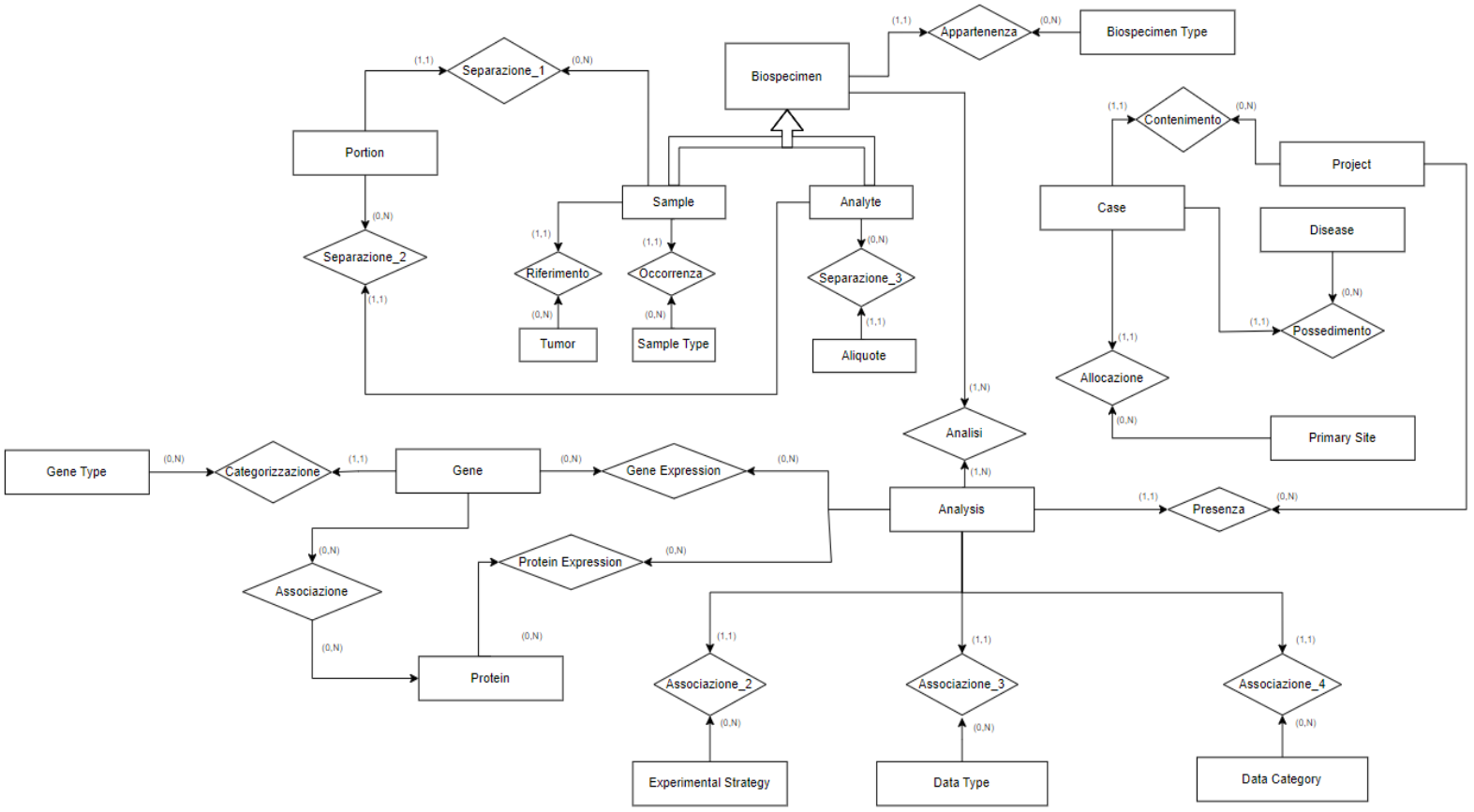


(Esempio di Tabella grafica per l'analisi dei dati disponibili su PDC)

Da GDC invece si è cercato di estrarre la medesima tipologia di dati estratto da PDC in modo da poter armonizzare i dati archiviati nel database e poterli usare in modo intercambiabili all'interno dei modelli.

3.3 Modello concettuale database

Un modello concettuale è una rappresentazione astratta dei dati, delle entità, delle relazioni e delle regole che descrivono uno specifico dominio. Si concentra sull'organizzazione logica dei dati piuttosto che sui dettagli di come verranno memorizzati all'interno delle tabelle del database. Il modello concettuale è fondamentale nella fase di progettazione dei database, poiché aiuta a definire in modo chiaro e comprensibile la struttura dei dati e le relazioni tra le diverse entità, riuscendo a definire eventuali gerarchie, semplificazioni o specificazioni possibili nella successiva fase di ristrutturazione. Il modello concettuale proposto rappresenta la prima fase della progettazione del futuro database utilizzato per archiviare le informazioni biomediche che andremo ad elaborare all'interno del nostro progetto.



(Modello concettuale di base)

3.4 Dizionario delle entità e relazioni

3.4.1 Dizionario delle entità

ENTITÀ	DESCRIZIONE	ATTRIBUTI	PRIMARY KEY
PROJECT	Contiene i dati riguardo i progetti/studi biomedici con un identificativo univoco e un nome. Ogni progetto può avere associati una serie di file, casi, e altre informazioni specifiche del progetto	project_id, name	project_id
CASE	Contiene i dati relativi ad un paziente specifico nel contesto di un progetto specifico	case_id, ethnicity, gender, race, vital_status	case_id
BIOSPECIMEN	Contiene i dati relativi ai campioni di materiale biologico	id	id
DISEASE	Contiene i dati sulla malattia del caso preso in questione	disease_id, type	disease_id
PRIMARY SITE	Contiene i dati sul sito preso in analisi del caso in questione	site_id, site	site_id
SAMPLE TYPE	Tipologia di un campione	type_id, type	type_id
TUMOR	Contiene i dati sul tumore studiato in campione	tumor_code_id, code, descriptor	tumor_code_id
BIOSPECIMEN TYPE	Contiene informazioni sulle tipologie dei campioni biologici (Sample, Slide, Portion, Analyte o Aliquot)	type_id, type	type_id

ANALYSIS	Contiene i dati sui file contenente i dati sulle espressioni genomiche e proteomiche nel dominio GDC	file_id, filename, file_size, created_datetime, updated_datetime	file_id
EXPERIMENTAL STRATEGY	Contiene informazioni sulla strategia sperimentale usata per ciascuna analisi	strategy_id, strategy	strategy_id
DATA TYPE	Contiene informazioni sul tipo dei dati contenuti nei file di analisi	type_id, type	type_id
DATA CATEGORY	Contiene informazioni sulle categorie dei dati contenuti nei file di analisi	category_id, category	category_id
GENE	Contiene informazioni sui singoli geni individuati	gene_id, gene_name	gene_id
GENE TYPE	Contiene i dati sulle tipologie dei geni	type_id, type	type_id
PROTEIN	Contiene i dati sulle proteine estratte dalle analisi e associate a dei geni	agid, lab_id, catalog_num, set_id, peptide_tget, gene_id, aliquot,unshred_peptides, distinct_peptides, log2_ratio, unsh_log2_ratio, project_id, spactral_count	agid

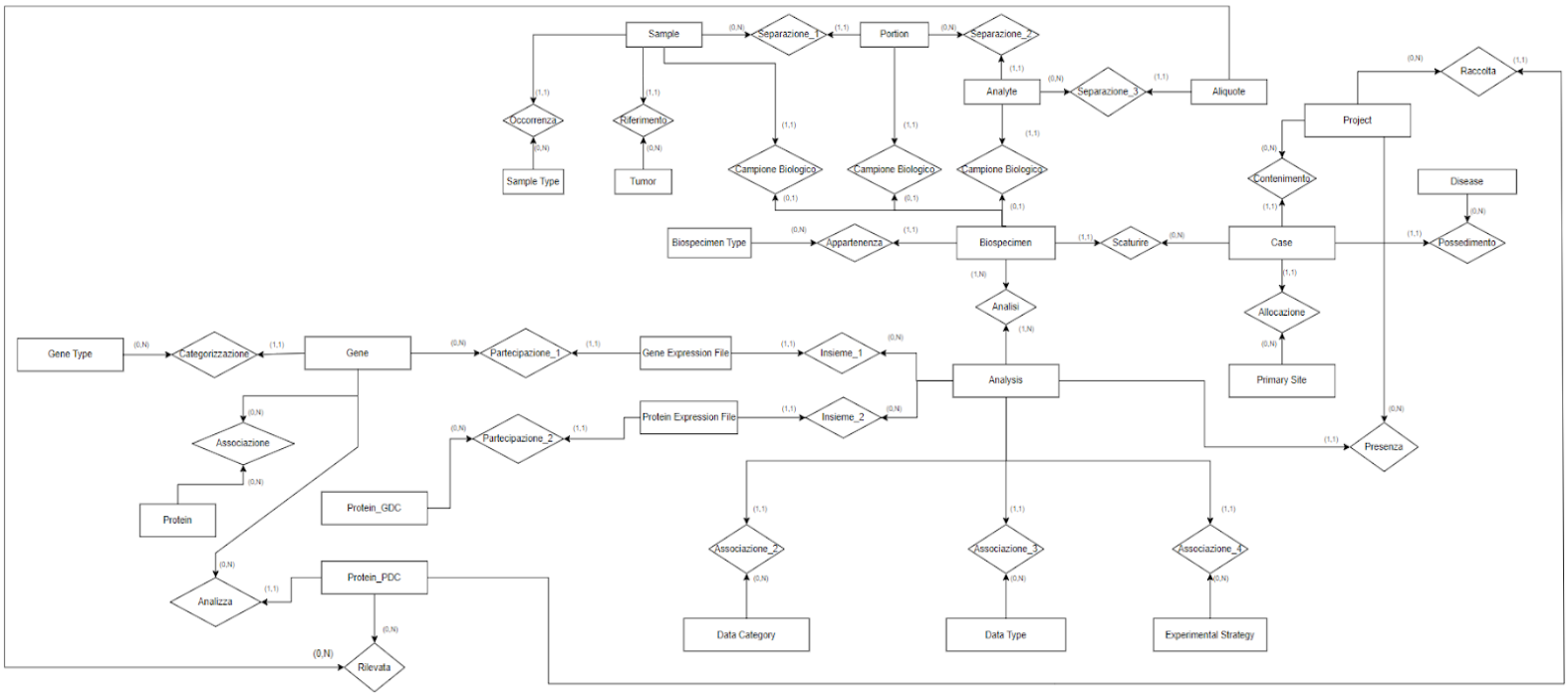
ANALYTE	Contiene i dati su un prodotto sfuso liquido, viene estratto da un campione di materiale organico	Ereditati da Biospecimen	Ereditati da Biospecimen
ALIQUOTE	Contiene i dati sulla separazioni in due o più parte di un campione	id	id
SAMPLE	Contiene i dati relativi ai campioni di materiale biologico, includendo più componenti	Ereditati da Biospecimen	Ereditati da Biospecimen
PORTION	Contiene i dati delle sottocomponenti dei campioni	Ereditati da Biospecimen	Ereditati da Biospecimen

3.4.2 Dizionario delle relazioni

RELAZIONI	DESCRIZIONE	ENTITA' IN RELAZIONE
CONTENIMENTO	Associa ogni case ad un project	Project, Case
PRESENZA	Associa ogni file ad un project	Project, Analysis
POSSEDIMENTO	Associa ogni case ai suoi disease	Case, Disease
ALLOCAZIONE	Associa ogni case ad un primary site	Case, Primary Site
OCCORRENZA	Associa un campione biologico al suo type	Sample, Type
RIFERIMENTO	Associa un sample al tumor a cui fa riferimento	Sample, Tumor

SEPARAZIONE_1	Associa un sample alle portion in cui è stato scomposto	Sample, Portion
SEPARAZIONE_2	Associa delle portion agli analyte	Portion, Analyte
SEPARAZIONE_3	Associa un'analyte alle sue aliquot	Analyte, Aliquot
APPARTIENE	Associa ad un campione biologico il suo type	Biospecimen, Biospecimen Type
ANALISI	Associa ad un'analisi i campioni biologici di cui effettua l'analisi	Biospecimen, Analysis
ASSOCIAZIONE_2	Associa una strategia di sperimentazione ad ogni analisi	Experimental Strategy, Analysis
ASSOCIAZIONE_3	Associa un data type ad ogni analisi	Data type, Analysis
ASSOCIAZIONE_4	Associa un data category ad ogni analisi	Data category, Analysis
Gene Expression	Associa ad ogni analisi i dati sui geni analizzati	Gene, Analysis
Protein Expression	Associa ad ogni analisi i dati sulle proteine analizzati	Protein, Analysis
Associazione	Associa ad ogni gene le proteine individuate durante lo studio	Gene, Protein
Categorizzazione	Associa ad ogni gene un gene type	Gene, Gene Type

3.5 Modello concettuale ristrutturato



(Modello concettuale ristrutturato)

3.6 Dizionario delle entità e relazioni ristrutturato

3.6.1 Dizionario delle entità

ENTITÀ	DESCRIZIONE	ATTRIBUTI	PRIMARY KEY
Biospecimen	Contiene i dati relativi ai campioni di materiale biologico	Id	Id
Analyte	Contiene i dati su un prodotto sfuso liquido, viene estratto da un campione di materiale organico	Analyte_id, Concentration	Analyte_id
Biospecimen Type	Contiene informazioni sulle tipologie dei campioni biologici (Sample, Slide, Portion, Analyte o Aliquot)	Type_id, Type	Type_id
Aliquote	Contiene i dati sulla separazioni in due o più parte di un campione	Aliquote_id, Concentration	Aliquote_id
Disease	Contiene i dati sulla malattia del caso preso in questione	Disease_id, Type	Disease_id
Case	Contiene i dati relativi ad un paziente specifico nel contesto di un progetto specifico	Case_id, Ethnicity, Gender, Race, Vital_status,	Case_id
Sample	Contiene i dati relativi ai campioni di materiale biologico, includendo più componenti	Sample_id	Sample_id

Sample Type	Tipologia di un campione	Type_id, Type	Type_id
Primary Site	Contiene i dati sul sito preso in analisi del caso in questione	Site_id, Site	Site_id
Project	Contiene i dati riguardo i progetti/studi biomedici con un identificativo univoco e un nome.	Project_id, Name	Project_id
Data Type	Contiene informazioni sul tipo dei dati contenuti nei file di analisi	Type_id, Type	Type_id
Experimental Strategy	Contiene informazioni sulla strategia sperimentale usata per ciascuna analisi	Strategy_id, Strategy	Strategy_id
Data Category	Contiene informazioni sulle categorie dei dati contenuti nei file di analisi	Category_id, Category	Category_id
Protein_GDC	Contiene i dati sulle proteine estratte dalle analisi e associate a dei geni nel dominio di GDC	Agid, Lab_id, Catalog_number, Set_id, Peptide_target	Agid
Protein	Contiene informazioni sulle proteine estratte	Protein	Protein
Protein_PDC	Contiene i dati sulle proteine estratte dalle analisi delle varie aliquote nel dominio di PDC	Label, Spectral_count, Distinct_peptides, Unshared_peptides, Log2_ratio, Unshared_log2	Gene, Aliquot, Project

		_ratio	
Gene	Contiene informazioni sui singoli geni individuati	Gene_id, Name	Gene_id
Gene Type	Contiene i dati sulle tipologie dei geni	Type_id, Type	Type_id
Portion	Contiene i dati delle sottocomponenti dei campioni	Portion_id	Portion_id
Tumor	Contiene i dati sui tumori individuati	Tumor_code_id, Code, Description	Tumor_code_id
Analysis	Contiene i dati sui file contenente i dati sulle espressioni genomiche e proteomiche nel dominio GDC	File_id, Filename, File_size, Created_datetime, Updatedtime	File_id
Gene Expression File	Contiene i dati estratti dai file delle espressioni genomiche nel dominio GDC	Tpm, Fpkm, Fpkm_uq, Unstranded, Straded_first, Straded_second	Analysis, Gene
Protein Expression File	Contiene i dati estratti dai file delle espressioni proteomiche nel dominio GDC	Expression	Analysis, Protein

3.6.2 Dizionario delle relazioni

RELAZIONI	DESCRIZIONE	ENTITA' IN RELAZIONE
Occorrenza	Associa a ciascun sample un tipo specifico tipo	Sample, Sample Type
Riferimento	Associa a ciascun sample un tumore a cui fa riferimento	Sample, Tumor
Campione Biologico	Associa un campione biologico al biospecimen da cui viene estratto	Sample/Portion/Analyte, Biospecimen
Separazione_1	Associa ad ogni portion un sample di riferimento	Sample, Portion
Separazione_2	Associa ad ogni analyte la portion di riferimento	Portion, Analyte
Separazione_3	Associa ad ogni aliquote l'analyte di riferimento	Analyte, Aliquote
Appartenenza	Associa un biospecimen al suo tipo specifico	Biospecimen, Biospecimen Type
Scaturire	Associa ad ogni biospecimen il suo case di riferimento	Case, Biospecimen
Contenimento	Associa ad ogni case il progetto in cui è coinvolto	Case, Project
Raccolta	Associa ad ogni proteina campione relativo alle proteine estratte da PDC il progetto che l'ha individuata	Protein_PDC, Project

Possedimento	Associa ad ogni case la malattia in studio	Disease, Case
Allocazione	Associa ad ogni case il sito preso in analisi	Case, Primary Site
Analisi	Associa ogni campione biologico ai suoi studi	Biospecimen, Analysis
Presenza	Associa ogni analisi al progetto di appartenenza	Analysis, Project
Insieme_1	Associa ad ogni analisi i file della sperimentazione genomica	Analysis, Gene Expression File
Insieme_2	Associa ad ogni analisi i file della sperimentazione proteomica	Analysis, Protein Expression File
Partecipazione_1	Associa ad ogni file di uno studio genomico il gene di riferimento	Gene Expression File, Gene
Partecipazione_2	Associa ad ogni file di uno studio proteomico la proteina di riferimento	Protein Expression File, Prorein_GDC
Categorizzazione	Associa ad ogni gene il suo gene type	Gene, Gene Type
Associazione	Associa ogni proteina al gene di riferimento	Gene, Protein
Analizza	Associa ad ogni studio di PDC i geni rilevati	Gene, Protein_PDC
Rilevata	Associa ad ogni studio di PDC le aliquote analizzate	Protein_PDC, Aliquote
Associazione_2	Associa ad ogni analisi la categoria di dati	Data Category, Analysis

Associazione_3	Associa ad ogni analisi il tipo di dati	Data type, Analysis
Associazione_4	Associa ad ogni analisi la strategia di sperimentazione	Experimental Strategy, Analysis

3.7 Tavola dei volumi

CONCETTO	TIPO	VOLUME
Biospecimen	Entità	610.000
Analyte	Entità	160.000
Biospecimen Type	Entità	5
Aliquote	Entità	310.000
Disease	Entità	230
Case	Entità	25.000
Sample	Entità	70.000
Sample Type	Entità	25
Primary Site	Entità	70
Project	Entità	300
Data Type	Entità	30
Experimental Strategy	Entità	15

Data Category	Entità	10
Protein_GDC	Entità	500
Protein	Entità	135.000
Protein_PDC	Entità	200.000.000
Gene	Entità	80.000
Gene Type	Entità	60
Portion	Entità	100.000
Tumor	Entità	40
Analysis	Entità	35.000
Gene Expression File	Entità	1.500.000.000
Protein Expression File	Entità	5.000.000
Occorrenza	Relazione	70.000
Riferimento	Relazione	70.000
Campione Biologico	Relazione	330.000
Separazione_1	Relazione	70.000
Separazione_2	Relazione	160.000

Separazione_3	Relazione	310.000
Appartenenza	Relazione	610.000
Scaturire	Relazione	610.000
Contenimento	Relazione	25.000
Raccolta	Relazione	200.000.000
Possedimento	Relazione	25.000
Allocazione	Relazione	25.000
Analisi	Relazione	610.000
Presenza	Relazione	35.000
Insieme_1	Relazione	1.500.000.000
Insieme_2	Relazione	5.000.000
Partecipazione_1	Relazione	1.500.000.000
Partecipazione_2	Relazione	5.000.000
Categorizzazione	Relazione	80.000
Associazione	Relazione	11.000.000
Analizza	Relazione	200.000.000

Rilevata	Relazione	200.000.000
Associazione_2	Relazione	35.000
Associazione_3	Relazione	35.000
Associazione_4	Relazione	35.000

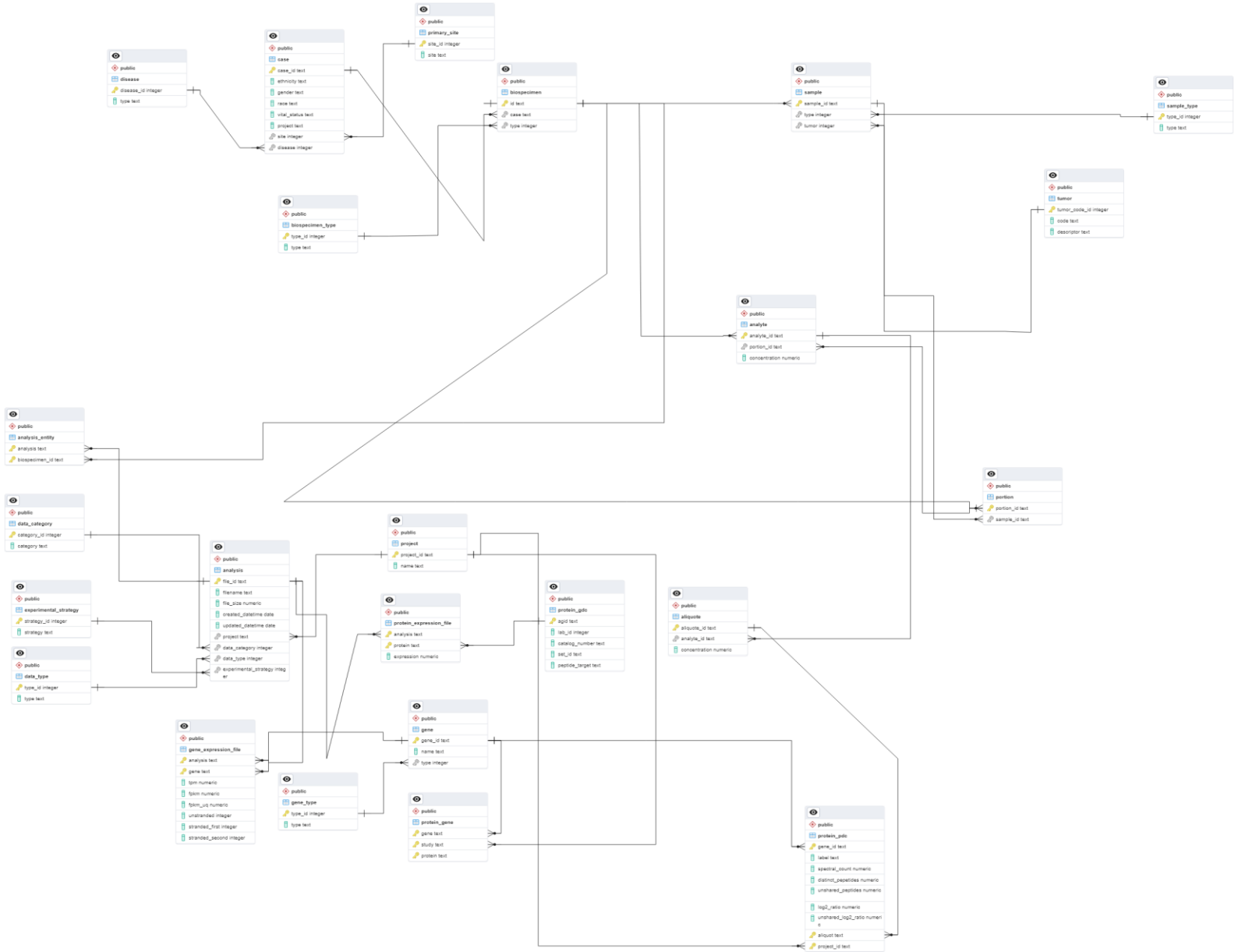
3.8 Schema logico

Aliquote (aliquote_id, analyte_id, concentration) Foreign Key analyte_id references analyte(analyte_id)
Analysis (file_id, filename, file_size, created_datetime, updated_datetime, project, data_category, data_type, experimental_strategy) Foreign Key data_category references data_category(category_id) Foreign Key data_type references data_type(type_id) Foreign Key project references project(project_id) Foreign Key experimental_strategy references experimental_strategy(strategy_id)
Analysis_entity (analysis, biospecimen_id) Foreign Key analysis references analysis (file_id) Foreign Key biospecimen_id references biospecimen (id)
Analyte (analyte_id, portion_id, concentration) Foreign Key analyte_id references biospecimen (id) Foreign Key portion_id references portion(portion_id)
Biospecimen (id, case, type) Foreign Key case references case(case_id) Foreign Key type references biospecimen_type(type_id)
Biospecimen_type (type_id, type)
Case (case_id, ethnicity, gender, race, vital_status, project, site, disease)

Foreign Key disease references disease(disease_id) Foreign Key site references primary_site(site_id) Foreign Key project references project(project_id)
Data_Category (category_id, category)
Data_Type (type_id, type)
Disease (disease_id, type)
Experimental_Strategy (strategy_id, strategy)
Gene (gene_id, name, type) Foreign Key type references gene_type(type_id)
Gene_Expression_File (analysis, gene, tpm, fpkm, fpkm_uq, unstranded, stranded_first, stranded_second) Foreign Key file references analysis(file_id) Foreign Key genereferences gene(gene_id)
Gene_Type (type_id, type)
Portion (portion_id, sample_id) Foreign Key portion_id references biospecimen(id) Foreign Key sample_id references sample(sample_id)
Primary_Site (site_id, site)
Project (project_id, name)

Protein_Expression_File (analysis, protein, expression) Foreign Key analysis references analysis(file_id) Foreign Key protein references protein_gdc(agid)
Protein_GDC (agid, lab_id, catalog_number, set_id, peptide_target)
Protein_Gene (gene, study, protein) Foreign Key gene references gene(gene_id) Foreign Key study references project(project_id)
Protein_PDC (gene_id, label, spectral_count, distinct_pepetides, unshared_paptides, log2_ratio, unshared_log2_ratio, aliquot, project_id) Foreign Key aliquote references aliquote(aliquote_id) Foreign Key gene_id references gene(gene_id) Foreign Key project_id references project(project_id)
Sample (sample_id, type, tumor) Foreign Key tumor references tumor(tumor_code_id) Foreign Key type references sample_type(type_id) Foreign Key sample_id references biospecimen(id)
Sample_Type (type_id, type)
Tumor (tumor_code_id, code, descriptor)

3.9 Schema Relazionale



(Schema relazionale)

Capitolo 4 Implementazione

In questo capitolo si andrà ad analizzare e descrivere le tecnologie usate e l'implementazione realizzata per la raccolta dei dati necessari per i modelli di machine learning

4.1 Tecnologie usate

Ogni parte specifica del progetto è stata sviluppata utilizzando tecnologie e metodologie differenti. Possiamo individuare nel progetto 2 diverse tipologie di operazioni:

1. ottenimento dei dati provenienti dalle API dei repository PDC e GDC e dei dati provenienti dallo scraping web
2. interazione con il database

Per il primo punto si è utilizzato il linguaggio di programmazione Python con diverse librerie di supporto in modo da poter effettuare le seguenti operazioni:

- request verso le API dei repository PDC e GDC
- scraping web dei siti PDC e GDC
- estrazione e manipolazione dei dati necessari
- interazione con il database che comprende il salvataggio, l'estrazione e l'aggiornamento dei dati

Per il secondo punto si è utilizzato PostgresSQL per effettuare:

1. creazione del database
2. definizione delle tabelle
3. salvataggio dei dati
4. estrazione dei dati
5. aggiornamenti dei dati

4.2 Librerie e moduli utilizzati

In questo paragrafo vogliamo esaminare le varie librerie e tecnologie utilizzate per implementare la parte di scripting e web scraping realizzata in Python.

Tra le librerie usate nel codice Python possiamo trovare:

- Requests: consente di effettuare le richieste HTTP verso le API per ottenere i dati necessari,
- Json: consente di convertire i dati ottenuti dalle API in formato JSON e lavorare con questa tipologia di dato,
- Psycopg2: consente di instaurare una connessione con il database PostgresSQL e successivamente di effettuare tutte le operazioni sul db di cui necessitiamo,
- Pandas: consente di manipolare e analizzare i dati ottenuti dalle API, andando a creare delle strutture dati specifiche,
- Os: consente di interagire con il sistema operativo, per la gestione di file e l'esecuzione di comandi,
- Datetime: consente di ottenere e manipolare la data e l'orario,
- Pathlib: consente la manipolazione e costruzione di percorsi adatti ai filesystem,
- Time: consente di ottenere e manipolare data/orario e ci mette a disposizione la funzione sleep() utilizzata per inserire una pausa durante l'esecuzione del programma,
- BeautifulSoup: consente di estrarre i dati da file HTML creando alberi di analisi che consentono l'analisi del codice HTML ottenuto attraverso uno scraping web,
- Selenium: consente l'automatizzazione di un browser web per effettuare scraping web

Una nota da fare è su l'utilizzo della libreria Selenium che ci consente di simulare l'utilizzo da parte di un utente di un browser web, andando a sfruttare un WebDriver.

Un WebDriver offre di simulare attraverso il vero e proprio browser web le azioni che un utente potrebbe andare a svolgere, includendo funzionalità di navigazione, input, esecuzione di JavaScript e così via.

4.3 Funzioni di gestione del database

Nella fase di realizzazione del progetto si è deciso di creare delle funzioni apposite di connessione, gestione e interazione col database in modo da tenere distinte le funzione di interazione col database dalle funzioni che manipolano i dati in analisi contenuti in memoria.

Per la parte di gestione del database possiamo trovare le seguenti funzioni fondamentali

databaseConstruction

Si occupa di verificare l'esistenza delle tabelle tabelle presenti nel database, in caso contrario leggerà il contenuto di un file contenente lo schema sql precedentemente elaborato e dopo aver elaborato il suo contenuto si occuperà di creare le tabelle necessarie all'esecuzione e richiamare la funzione fillingBasicTable()

databaseCreation

Si occupa di creare una connessione con il server PostgresSQL, creare il database e controllare se esistono dei backup del database da cui recuperare i dati, in caso positivo effettuerà il caricamento dei dati attraverso la funzione reloadData(), in caso negativo richiamerà la funzione databaseConstruction() per costruire la struttura delle tabelle del database

databaseConnection

Si occupa di creare una connessione con il database PostgresSQL, andrà a controllare l'effettiva esistenza del database, se esso non esiste invocherà la funzione databaseCreation() per richiederà in modo automatico la creazione del database

filingBasicTable

Si occupa di riempire le tabelle di base necessarie all'esecuzione degli script, al momento l'unica tabella da riempire prima dell'esecuzione è la tabella Biospecimen_Type

saveDatabase

Si occupa di creare un backup della struttura e dei file contenuti all'interno del database, per farlo sfrutta la pg_dump offerta da Postgres. La funzione inoltre si occupa di controllare e mantenere costante il numero di backup effettuati, limitandone le copie a due

reloadData

Si occupa di recuperare i dati di cui si era effettuato il backup, scegliendo sempre la versione più aggiornata in modo da ripristinare la quantità di dati maggiore possibile

Per la parte di interazione con il database si sono definite una serie di funzioni specifiche per verificare l'esistenza di un determinato elemento nel database, fare un inserimento di un elemento o ottenerlo.

Nello specifico si sono create le seguenti funzioni

checkExistBiospecimen()

Controlla se un biospecimen con un determinato id è già presente nel database

insertNewBiospecimen()

Inserisce un nuovo biospecimen nel database

checkExistAliquote()

Controlla se un aliquote con un determinato aliquote_id è già presente nel database

insertNewAnalyte()

Inserisce una nuova analyte nel database

insertNewPortion()

Inserisce una nuova portion nel database

insertNewAliquote()

Inserisce una nuova aliquote nel database

checkExistProject()

Controlla se un project con un determinato project_id è già presente nel database

getProjectId()

Ottiene il project_id di un determinato project già presente nel database partendo dal nome del progetto

getLog2Ratio()

Ottiene il valore del log2_ratio di una tupla individuata dal gene, aliquota e project_id di riferimento dalla tabella protein_pdc

getUnsharedLog2Ratio()

Ottiene il valore del unshared_log2_ratio di una tupla individuata dl gene, aliquota e project_id di riferimento dalla tabella protein_pdc

insertNewProject()

Inserisce un nuovo progetto nel database

checkExistCase()

Controlla se un case con un determinato case_id è già presente nel database

insertNewCase()

Inserisce un nuovo case nel database

insertNewSample()

Inserisce un nuovo sample nel database

checkExistFile()

Controlla se un file con un determinato file_id è già presente nel database

searchSampleTypeId()

Ottiene il valore del type_id di una tupla individuata dal type dalla tabella sample_type

insertNewSampleType()

Inserisce un nuovo sample type nel databse

checkExistTumor()

Controlla se un tumore con un determinato tumor_code_id è già presente nel database

insertNewGeneProteinStudy()

Inserisce una nuova tupla nella tabella protein_gene associando un gene e una proteina

checkExistProtein_PDC()

Controlla se una tuple identificata dal gene_id, aliquot e project_id è già presente nella tabella protein_pdc

insertNewTumor()

Inserisce un nuovo tumor nel database

checkExistGene()

Controlla se un gene con un determinato gene_id è già presente nel database

insertNewGene()

Inserisce un nuovo tumor nel database

getPrimarySite()

Si occupa di ottenere il site_id di un site dal suo nome, se non presente nel database invoca la funzione insertNewPrimarySite e lo ottiene

insertNewPrimarySite()

Inserisce un nuovo primary site nel database

getDisease()

Si occupa di ottenere il disease_id di un disease dal suo type, se non presente nel database invoca la funzione insertNewDisease e lo ottiene

insertNewDisease()

Inserisce un nuovo disease nel database

getExperimentalStrategy()

Si occupa di ottenere lo strategy_id di una experimental_strategy dalla sua strategy, se non presente nel database invoca la funzione insertNewExperimentalStrategy e lo ottiene

insertNewExperimentalStrategy()

Inserisce una nuova experimental strategy nel database

getDataCategory()

Si occupa di ottenere la category_id di una data_category dalla sua category, se non presente nel database invoca la funzione insertNewDataCategory e la ottiene

insertNewDataCategory()

Inserisce una nuova data category nel database

getDataType()

Si occupa di ottenere il type_id di una data_type dal suo type, se non presente nel database invoca la funzione insertNewDataType e lo ottiene

insertNewDataType()

Inserisce una nuova data type nel database

getGeneType()

Si occupa di ottenere il type_id di un gene_type dal suo type, se non presente nel database invoca la funzione insertNewGeneTypee lo ottiene

insertNewGeneType()

Inserisce un nuovo gene type nel database

4.4 Funzioni di download

In questo capitolo si andranno ad illustrare le funzione utilizzate per l'effettivo download e manipolazione dei dati offerti dal repository PDC

getProgram()

Questa funzione specifica ci consente di ottenere i dati sui vari studi disponibili.

Attraverso l'API "studyCatalog" ottiene i dati sui vari progetti che ci consentiranno di ottenere successivamente i dati sui pazienti, aliquote, geni e proteine. Dopo aver convertito i dati, ricevuti attraverso una get, estrae le informazioni fondamentali dei progetti come:

- study_id
- study_submitter_id
- submitter_id_name
- pdc_study_id

La funzione crea un oggetto della classe study, specificamente creato per contenere i dati estratti, che viene aggiunto ad un array di tutti gli studi presenti in PDC che verrà restituito.

getCases(study_id, study_submitter_id, cursor per effettuare la query nel database, connessione che verrà utilizzata per effettuare il commit delle query)

Questa funzione specifica ci consente di ottenere i dati sui vari pazienti (case).

Verrà utilizzata l'API "paginatedCaseDemographicsPerStudy" che necessita però dei parametri:

- study_id: per identificare il progetto di cui si vogliono ottenere i case associati
- offset: per indicare il numero di case da saltare nell'interrogazione al repository
- limit: per indicare il numero massimo di case da estrarre dal repository

Da una prima interrogazione otterremo il numero di case da estrarre che utilizzeremo come valore per il campo limit nella request all'API sopra indicata.

Dopo aver ottenuto i dati sui vari case andremo a verificare per ciascuno se il 'case_submitter_id' è presente nel database sfruttando la funzione checkExistCase().

Dopo aver verificato che il case non è inserito all'interno del database andiamo ad estrarre i dati demografici di ciascun paziente, andando anche ad ottenere il 'disease_id' e il 'site_id' attraverso relativamente le funzioni getDisease() e getPrimarySite(). Dopo aver ottenuto tutti i dati relativi al case, andremo ad effettuare l'insert nel database attraverso l'apposita funzione insertNewCase().

getSample(pdc_study_id, cursor, connessione)

Questa funzione specifica ci consente di ottenere i dati sui vari sample.

Verrà utilizzata l'API "paginatedCasesSamplesAliquots" che necessita però dei parametri:

- pdc_study_id: per identificare il progetto di cui si vogliono ottenere i case associati
- offset: per indicare il numero di case da saltare nell'interrogazione al repository
- limit: per indicare il numero massimo di case da estrarre dal repository

Da una prima interrogazione otterremo il numero di sample da estrarre che utilizzeremo come valore per il campo limit nella request all'API sopra indicata.

All'interno della response ottenuta avremo una serie di dati relativamente ai sample, altri campioni biologici e al tumore associato.

Estrarremo inizialmente i dati relativi al tumore di riferimento, andando a controllare la loro completezza e se il tumore sia già presente all'interno del database attraverso la funzione checkExistTumor(), se essa ci ritornerà False e i dati relativi al tumore sono completi, andremo ad inserire i dati del nuovo tumore all'interno del database.

Dopo aver raccolto i dati sui sample andremo a verificare la loro correttezza, andando a verificare attraverso la funzione searchSampleTypeId() se il sample_type_id sia presente nel database, in caso negativo andremo ad inserirlo attraverso la funzione insertNewSampleType(). Può succedere in alcuni casi che il sample_type_id abbia il valore None associato, sarà quindi necessario andare a verificare se il sample_type sia già presente nel database e in caso positivo otterremo l'id specifico.

Per ogni sample nel risultato della request andremo ad effettuare attraverso le funzioni insertNewBiospecimen() e insertNewSample() l'inserimento di un nuovo biospecimen e di un nuovo sample all'interno del database.

Successivamente per ogni aliquote associata al sample andremo ad inserire nel database una nuova portion, un nuovo analyte e una nuova aliquote attraverso le funzioni insertNewPortion(), insertNewAnalyte() e insertNewAliquote().

Le aliquote saranno fondamentali per le successive funzioni.

getGenes(cursor, connessione)

Questa funzione specifica ci consente di ottenere i dati sui vari gene e le proteine associate a ciascuno.

Verrà utilizzata l'API "getPaginatedGenes" per ottenere i dati su tutti i geni con le relative proteine analizzate.

Per ciascun gene name andremo a costruire l'url necessario per richiamare l'API "geneSpectralCount" che ci consente di ottenere alcune informazioni dei geni.

Tra queste informazioni però abbiamo il il campo "pdc_study_id" che ci indica il project di riferimento attraverso lo standard pdc, che è differente a quello utilizzato per archiviare i dati all'interno del nostro database. Sarà quindi necessario attraverso l'API "allPrograms" ottenere tutti i "pdc_study_id" con il relativo id usato nel nostro database in modo da poterlo convertire.

Successivamente visto che all'interno del nostro database i geni sono identificati dal loro id che è nello standard Ensembl, che rappresenta lo standard più utilizzato nell'ambito genico, sarà necessario attraverso l'API offerta da NCBI l'Ensembl Id specifico per ogni gene analizzato.

Andiamo ad ottenere attraverso la funzione getGeneType() il gene type relativo al singolo gene, avendo quindi tutti i dati necessari, attraverso la funzione insertNewGene() andiamo a salvare i dati sul nuovo gene nel database.

Dopo aver salvato i dati relativi ai geni andiamo ora a salvare i dati sulle singole proteine, andando ad inserire attraverso la funzione insertNewGeneProteinStudy() l'associazione tra i singoli geni, le proteine e il progetto dove questi dati sono stati rilevati.

getProteinInfo(cursor, connessione)

Questa funzione specifica ci consente di ottenere maggiori dati sui vari dati, e di conseguenza sulle sue proteine associate.

Attraverso un interrogazione al database andremo ad ottenere i dati sui vari geni che abbiamo già salvato, per ciascuno di esso andremo a richiamare la funzione GetGeneProInformation() il cui funzionamento sarà illustrato successivamente.

Dopo aver terminato l'esecuzione della funzione sopra indicata attraverso l'API "allPrograms" otterremo i dati sui vari progetti, in particolare ci interessa ottenere i "pdc_study_id" che utilizzeremo come chiave nella creazione di un dizionario avente per valori i corrispondenti "study_submitter_id".

Successivamente questo dizionario verrà utilizzato per passare i valori necessari alla funzione getLog2RatioInfo() il cui funzionamento vedremo successivamente.

getGeneProInformation(gene_name, gene_id, cursor, connessione)

Questa funzione specifica ci consente di ottenere maggiori dati sui vari dati attraverso il web scraping effettuato con Selenium.

Per ogni gene_name passato la funzione costruirà un url specifico e attraverso il metodo get() di selenium aprirà il link grazie all'ausilio del chromedriver usato per simulare un utente che utilizza il web browser Chrome.

Dopo un'attesa di cinque secondi necessaria per consentire il caricamento della pagina web richiesta, si cercherà attraverso il metodo find_element_by_xpath() il bottone 'Continue' presente all'interno di un pop-up di disclaimer che viene visualizzato ogni volta che ci si collega ad una pagina PDC, in modo da poterlo chiudere simulando un click di un utente virtuale attraverso il metodo click().

Una volta arrivati sulla pagina web richiesta andremo ad ottenere il codice HTML della pagine attraverso l'attributo "page_source" del driver usato per aprire la pagina.

Manipolato il contenuto della pagina creiamo un array contenente le aliquote presenti nella pagina e andiamo ad iterare dieci volte, in modo da scorrere le dieci righe delle corrispondenti aliquote-valori presenti al massimo in ciascuna pagina, durante questa fase andiamo a verificare la posizione dell'etichetta "Label" in modo da identificare la corretta posizione degli elementi da estrarre successivamente.

Avendo ora trovato la posizione corretta degli elementi otteniamo i valori necessari ad effettuare una successiva insert/update nella tabella "protein_pdc", andando ad ottenere il "project_id" attraverso la funzione getProjectId().

Dopo aver verificato l'esistenza dei vari valori necessari all'inserimento controlla se la tupla con le sue specifiche chiavi già esiste all'interno del database, in caso affermativo effettua un UPDATE della tupla, in caso negativo effettua una INSERT .

Conclusa l'analisi di tutte le righe presenti nella pagina andrà a ricercare attraverso il metodo find_element() il bottone necessario a caricare le 10 tuple successive, in modo da analizzare e immagazzinare anche queste nuove informazioni. Prima di di iniziare la nuova estrazione però si attende un tempo semi casuale tra 0.95s e 4.95s in modo da ridurre la possibilità da parte del sito di richiedere tramite captcha la verifica dell'utente, per assicurarsi che non sia un bot automatizzatto a richiedere le pagine web del sito.

query_pdc(pdc_study_id, data_type)

Questa funzione specifica ci consente di ottenere maggiori i dati dei log2_ratio dei vari geni. Per ciascun "pdc_study_id" e "data_type" (che può essere "log2_ratio" oppure "unshared_log2_ratio") costruisce un json apposito da passare all'interno di una request che ci consentirà di ottenere per ciascun gene tutte le aliquote a esso associate all'interno del progetto indicato, con i relativi valori dei "log2_ratio" oppure "unshared_log2_ratio". Restituirà il json dei dati ottenuti.

getEnsemblId(gene_name)

Questa funzione specifica ci consente di ottenere maggiori l'Ensembl gene id partendo da un gene name. Per ciascun gene_name passato alla funzione andrà a richiedere tramite una get al sito Ensembl il gene id e andrà a scrivere in un apposito file la corrispondenza gene_name : gene_id. Ritornerà la corrispondenza individuata.

getLog2RatioInfo(pdc_study_id, study_submitter_id, cursor, connessione)

Questa funzione ci consente di ottenere il valore del log2_ratio e dell'unshared_log2_ratio individuato per ogni gene, in ogni aliquota di ogni progetto individuato.

Dopo aver ottenuto i dati attraverso la funzione query_pdc(), andremo a costruire una struttura di 2 dimensioni, in modo da avere una tabella in cui navigare per estrarre i dati, come ad esempio la lista dei gene_name, la cui presenza dei suoi singoli elementi dovrà essere verificata all'interno di un file di caching dedicato utilizzato per salvare la traduzione gene_name : gene_id.

La scelta di utilizzare un file locale di caching nasce dal fatto che per effettuare la traduzione attraverso Ensembl di ogni gene name richiederebbe troppo tempo, in questo modo andremo a effettuare questa traduzione solo la prima volta che si lanciano gli script di download, tutte le volte successive si andrà prima a controllare se un gene_name è presente all'interno del file di caching, se è presente ottiene il relativo gene_id e procede nell'esecuzione, in caso non sia salvato nel file allora procederà ad effettuare la richiesta della traduzione attraverso la funzione getEnsemblId() e salverà la nuova corrispondenza nel file di caching. Usando questa tecnica di memorizzazione locale riduciamo di molto i tempi di esecuzione di questa funzione.

La funzione andrà quindi a verificare che i dati necessari a creare le nove tuple, rispettino i vincoli delle foreign key espressi nel database, per effettuarlo sfrutta le funzioni checkExistGene(), checkExistProject() e checkExistAliquote() e verifica che il valore estratto per il log2_ratio/unshared_log2_ratio sia presente e nel formato corretto.

Finita questa fase di validazione dei dati andremo a creare il base al type (log2_ratio o unshared_log2_ratio), le query necessarie all'eventuale UPDATE o INSERT.

Per verificare se una tupla è già presente all'interno del database useremo la funzione checkExistProtein_PDC(), in caso positivo andremo a verificare se il valore già presente è uguale a zero (valore usato come placeholder dalla funzione getGeneProInformation()) oppure None, in questi due casi effettueremo un UPDATE della tupla già presente nel database. Nel caso la funzione checkExistProtein_PDC() dia esito negativo andremo ad effettuare una INSERT nel database utilizzando i dati a nostra disposizione.

Capitolo 5 Risultati finali

Nei capitoli precedenti abbiamo esaminato la progettazione concettuale e logica del database usato per la gestione dei dati relativi alle espressioni geniche/proteiche in ambito biomedico, esaminando nel dettaglio il codice delle funzioni implementate per l'estrazione, elaborazione e archiviazione dei dati. In questo capitolo esamineremo i risultati ottenuti dall'esecuzione delle funzioni sopra illustrate, i limiti tecnici e i possibili futuri sviluppi.

5.1 Dati processati

Nello sviluppo del progetto abbiamo visto quindi come raccogliere tutti i dati offerti dal repository pubblico PDC.

Tra questi dati troviamo delle informazioni fondamentali come:

- Geni con associate le relative proteine individuate,
- Campioni biologici quali biospecimen, sample, aliquote che rappresentano i vari campioni biologi estratti dai pazienti,
- Origine dei campioni biologici che ci consentono di avere i dati sui vari pazienti per conoscere caratteristiche come etnia, genere, stato vitale ed ecc,
- Tumori individuati che abbiamo associato a ciascun campione biologico individuato e categorizzati per malattia

Dai test effettuati riusciamo correttamente ad associare a ciascun progetto i suoi pazienti e per ciascuno di esso i propri campioni biologici. Dopo aver raccolto tutti i geni disponibili e averli associati alle loro proteine, individuiamo per ciascun campione biologico i geni che vi sono individuati, andando poi ad estrarre informazioni aggiuntive sui geni.

5.2 Limiti delle API

Attualmente le API che ci consentono di estrarre i dati dal repository sono molto limitate ed ancora oggi in fase di sviluppo. Questa è stata la causa che ha portato alla necessità di utilizzare soluzioni non sempre affidabili e stabili, come l'utilizzo di Selenium per effettuare il web scraping ed ottenere "manualmente" alcuni dei dati che oggi non sono ancora disponibili tramite le API ufficiali di PDC. Inoltre i dati non sempre sono propriamente normalizzati e standardizzati, potendo utilizzare come esempio la necessità di effettuare la conversione dei geni id offerti dalle API in uno standard pubblico come quello Ensembl.

Inoltre la mancanza di API specifiche, hanno causato al momento la mancanza di dati necessari per riempire tabelle quali protein_gdc, protein_expression_file ed altre, che possono essere però facilmente popolate attraverso degli script specifici scaricando i dati dal repository GDC. La struttura del database utilizzata in questo progetto infatti è stata pensata per essere compatibile anche con la raccolta di dati genomici e proteomici provenienti da GDC che era stata precedentemente sviluppata, comportando ad esempio il mantenimento di tabelle che attualmente non vengono utilizzate.

5.2 Futuri sviluppi e possibilità di utilizzo dei dati estratti

Lo step successivo nello sviluppo di questo progetto sarebbe quello di creare un'infrastruttura adeguata che consenta di schedulare in modo automatico e costante l'esecuzione degli script sopra descritti, in modo da poter aggiornare costantemente i dati raccolti e ampliare sempre di più la quantità di dati archiviata.

In futuro quando le nuove API saranno rilasciate da PDC sarà possibile acquisire nuovi dati che potranno essere usati per migliorare e rendere più affidabili i modelli di machine learning.

Infatti i dati raccolti, manipolati e archiviati nello specifico database dagli script sopra descritti possono essere usati come fonte di addestramento per modelli di machine learning dedicati alla rivelazione di pattern specifici. Come detto precedentemente i modelli di machine learning e la loro applicazione per l'analisi di nuovi campioni biologici consentirà di effettuare uno screening più accurato a rapido, riducendo i tempi di analisi e diagnosi che potrebbero consentire di anticipare eventuali cure per i pazienti malati e di calibrare in modo più correttamente le dosi di eventuali farmaci da assumere.

Bibliografia

- Proteomica: https://www.airc.it/cancro/informazioni-tumori/ricerca-di-base/proteomica
- Proteoma: https://www.biopills.net/proteomica/
- Biomarcatori: https://www.aifa.gov.it/-/i-biomarcatori-strumento-prezioso-per-lo-sviluppo-di-nuovi-farmaci
- PDC: https://pdc.cancer.gov/
- PDC API: https://pdc.cancer.gov/data-dictionary/publicapi-documentation
- PDC Data Dictionary: https://proteomic.datacommons.cancer.gov/pdc/data-dictionary
- NCBI: https://www.ncbi.nlm.nih.gov/datasets/docs/v2/reference-docs/rest-api/#post-/gene
- GDC: https://gdc.cancer.gov/
- GDC API: https://docs.gdc.cancer.gov/API/Users_Guide/Search_and_Retrieval/
- GDC Python Integration: https://docs.gdc.cancer.gov/API/Users_Guide/Python_Examples/
- Ensembl: https://www.ensembl.org/index.html
- Geni: https://www.biopills.net/proteine-struttura-funzione-e-relazione-con-i-geni/
- Uncovering Distinct Peptide Charging Behaviors in Electrospray Ionization Mass Spectrometry Using a Large-Scale Dataset: https://pubmed.ncbi.nlm.nih.gov/38095561/