EpMiner:
Scoperta di Pattern Frequenti e Pattern Emergenti

Caso di studio di Metodi Avanzati di Programmazione
AA 2020-2021

Realizzato da

Capone Simone 717216
s.capone7@studenti.uniba.it

Giacovazzo Christian 716696
c.giacovazzo1@studenti.uniba.it

Giannini Mariangela 680636
m.giannini16@studenti.uniba.it

INDICE

1. Introduzione

1.1 Algoritmo Apriori

L'algoritmo Apriori è stato proposto da R. Agrawal e R. Srikant nel 1994.
Esso è progettato per operare su database contenenti transazioni. Ogni transazione è vista come un insieme di elementi. Data una soglia ***C***, l'algoritmo Apriori identifica gli insiemi di elementi che sono sottoinsiemi di almeno ***C*** transazioni nel database.

Apriori utilizza un approccio "dal basso verso l'alto", in cui i sottoinsiemi frequenti vengono estesi un elemento alla volta e i gruppi di candidati vengono testati rispetto ai dati. L'algoritmo termina quando non vengono trovate ulteriori estensioni riuscite.

Apriori utilizza la ricerca in ampiezza (*breadth-first search*) e una struttura ad albero Hash per contare in modo efficiente i set di elementi candidati. Genera insiemi di elementi candidati di lunghezza ***k*** da insiemi di elementi di lunghezza ***k-1***. Quindi sfoltisce i candidati che hanno un sottomodello poco frequente. L'insieme candidato contiene così tutti gli insiemi di elementi frequenti di lunghezza k. Successivamente, esegue la scansione del database delle transazioni per determinare gli insiemi di elementi frequenti tra i candidati.

Lo pseudo codice per l'algoritmo è riportato di seguito. Viene impiegata la notazione della teoria degli insiemi. Ad ogni passo, si assume che l'algoritmo generi gli insiemi candidati dagli insiemi di elementi del livello precedente.

```
frequentPatternDiscovery(DTarget,minS)→FP
begin
        FP = Ø
        L_1 = {1 item che compaiono in minS x |D| transazioni di DTarget}
        K=2
        while L_k-1 ≠ Ø do
        begin
                C_k = {candidati generati da L_k-1 aggiungendo un nuovo item}
                L_k = Ø
                for each (p ϵ C_k) do
                        if (supporto(p, DTarget) >= minS) then
                                L_k = L_k U p
                FP = FP U L_k
                K = K + 1

        end

        return FP

end
```

```
EPDiscovery(DBackground,FP,minGr)→EP

begin
        EP = Ø
        for each (p ϵ FP) do
                begin
                        if (growrate(p,DBacKground) >= minGR) then
                                EP = EP U p

                end

        return EP

end
```

1.2 Limiti

L’algoritmo Apriori soffre di una serie di inefficienze. La generazione di candidati genera un numero elevato di sottoinsiemi, infatti l'algoritmo tenta di caricare il gruppo di candidati, con il maggior numero possibile di sottoinsiemi prima di ogni scansione del database. L'esplorazione dal basso verso l'alto del sottoinsieme trova qualsiasi sottoinsieme massimale S solo dopo tutti i **2^|S| - 1** suoi sottoinsiemi propri.

L'algoritmo esegue la scansione del database troppe volte, il che riduce le prestazioni complessive. Per questo motivo, l'algoritmo presuppone che il database sia permanentemente in memoria.

Inoltre, sia la complessità temporale che spaziale di questo algoritmo sono molto elevate: **O(2^|D|)**, quindi esponenziale, dove |D| è il numero totale di elementi presenti nel database.

2. Introduzione al progetto

2.1 Descrizione del progetto

Il software realizzato utilizza l'algoritmo Apriori, descritto della sezione precedente, elaborando dati estratti da una tabella presente in un database di tipo MySQL.
Il progetto, espansione di quello svolto a lezione, consiste in un'applicazione distribuita di tipo Client/Server con interazione attraverso una interfaccia grafica.

Il server si occupa di ricevere le richieste di uno o più client, i quali posso effettuare le seguenti operazioni:

- Ricerca di un nuovo pattern nel database
- Caricamento di un pattern salvato in archivio nel database.

In entrambi i casi, il client dovrà specificare nei criteri di ricerca:

- Minimo supporto
- Minimo grow rate
- Nome delle tabelle target e background

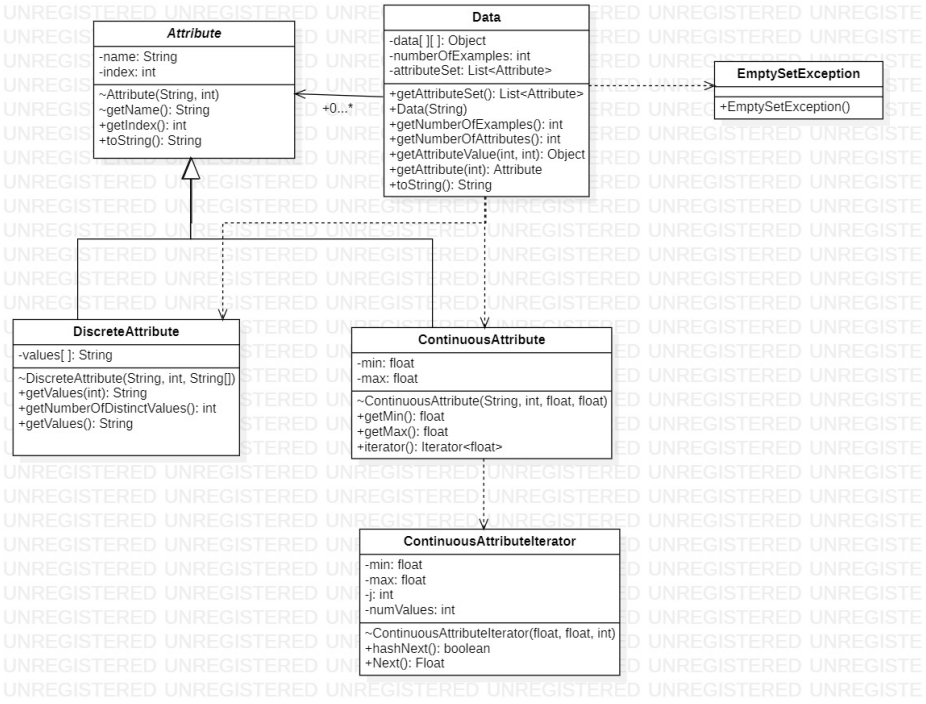
Il server mostra inoltre informazioni sul client connesso: le operazioni da esso richieste e il loro esito.
L'estensione è stata sviluppata usando la tecnologia JavaFX; inoltre, è stato utilizzato SceneBuilder per la creazione dell'interfaccia grafica e CSS per migliorare la user experience.[1]
Nel progetto sono presenti entrambe le versioni, sia quella fruibile attraverso console, sia quella utilizzabile con l'interfaccia grafica. Nel presente documento verrà trattata solo l'estensione.
Nella sezione 3 sono riportati anche i diagrammi UML per i package e per le classi.
Inoltre, nella cartella "*Javadoc*" è stata allegata la Javadoc creata direttamente dall'IDE di sviluppo (IntelliJ).
Nella sezione 5 del documento sono riportati esempi di esecuzione.

3. Diagrammi UML

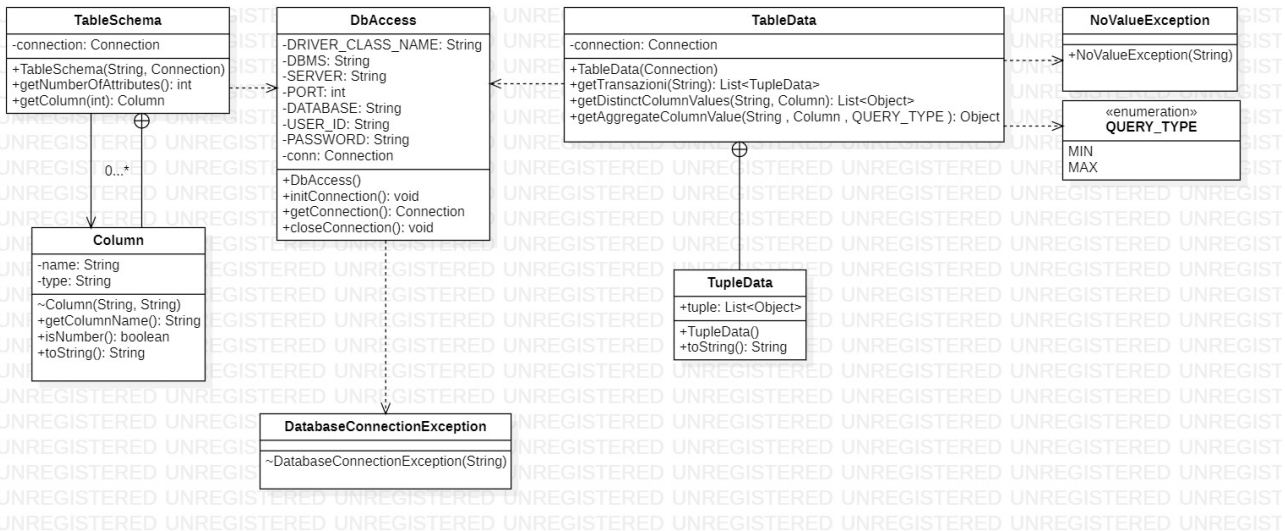
Per la realizzazione dei diagrammi è stato utilizzato il software *StarUML*.

3.1 UML Packages

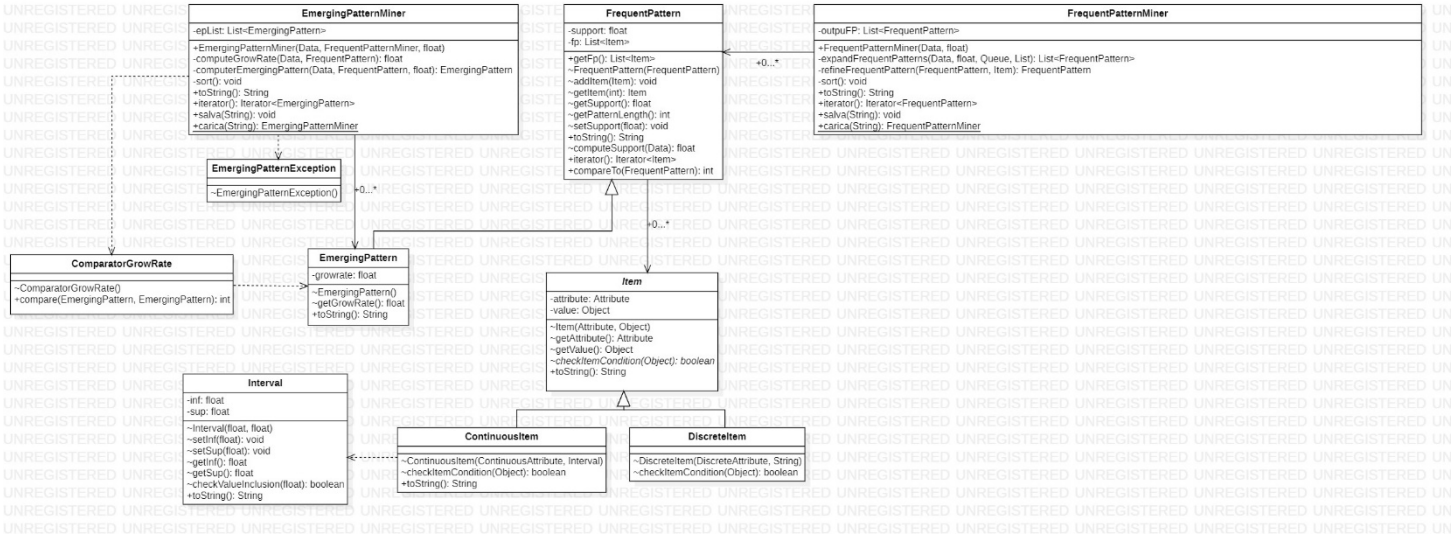
Package data



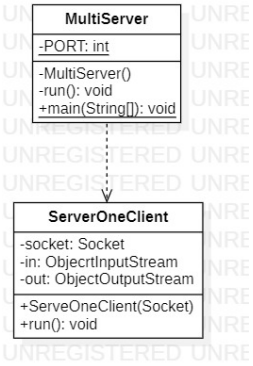
Package database



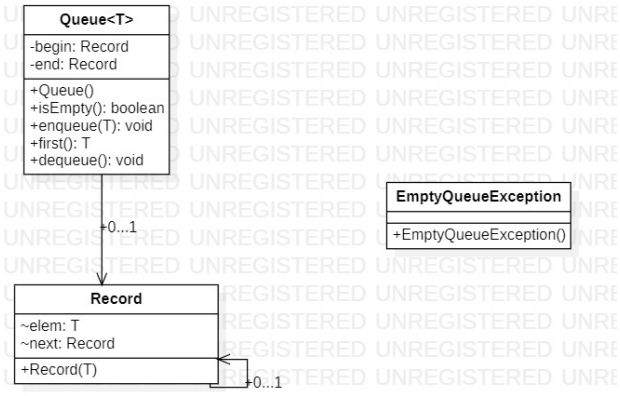
Package mining



Package server

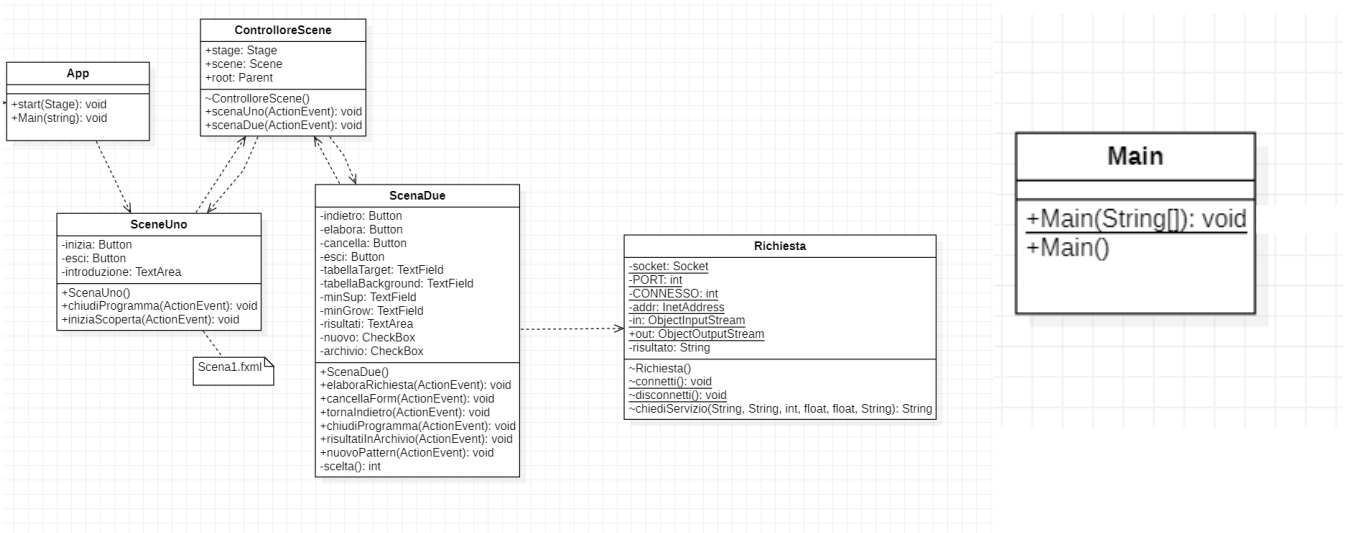


Package utility

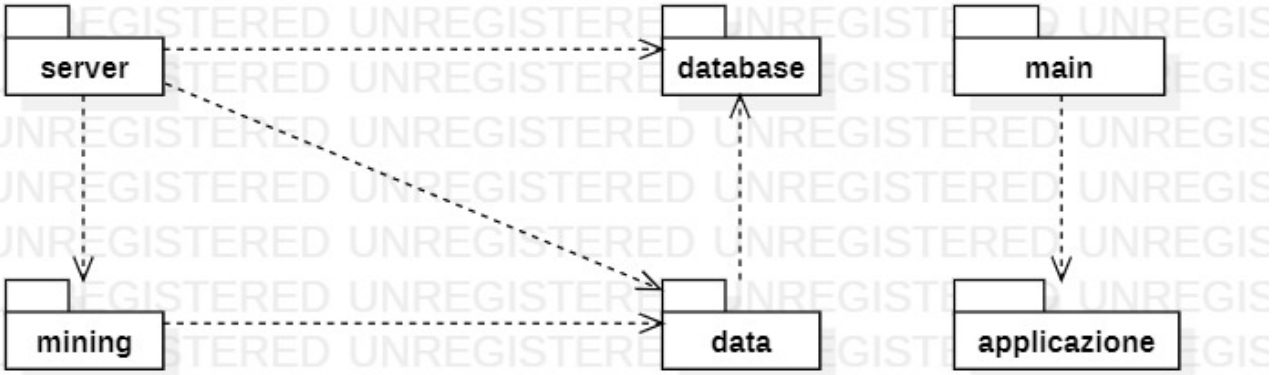


Package applicazione

Package main



Interazione Packages



3.2 UML Classi

Per garantire una maggiore leggibilità, nel diagramma delle classi EpMinerServer_esteso sono stati omessi metodi e attributi.

Diagramma completo delle classi EpMinerServer_esteso

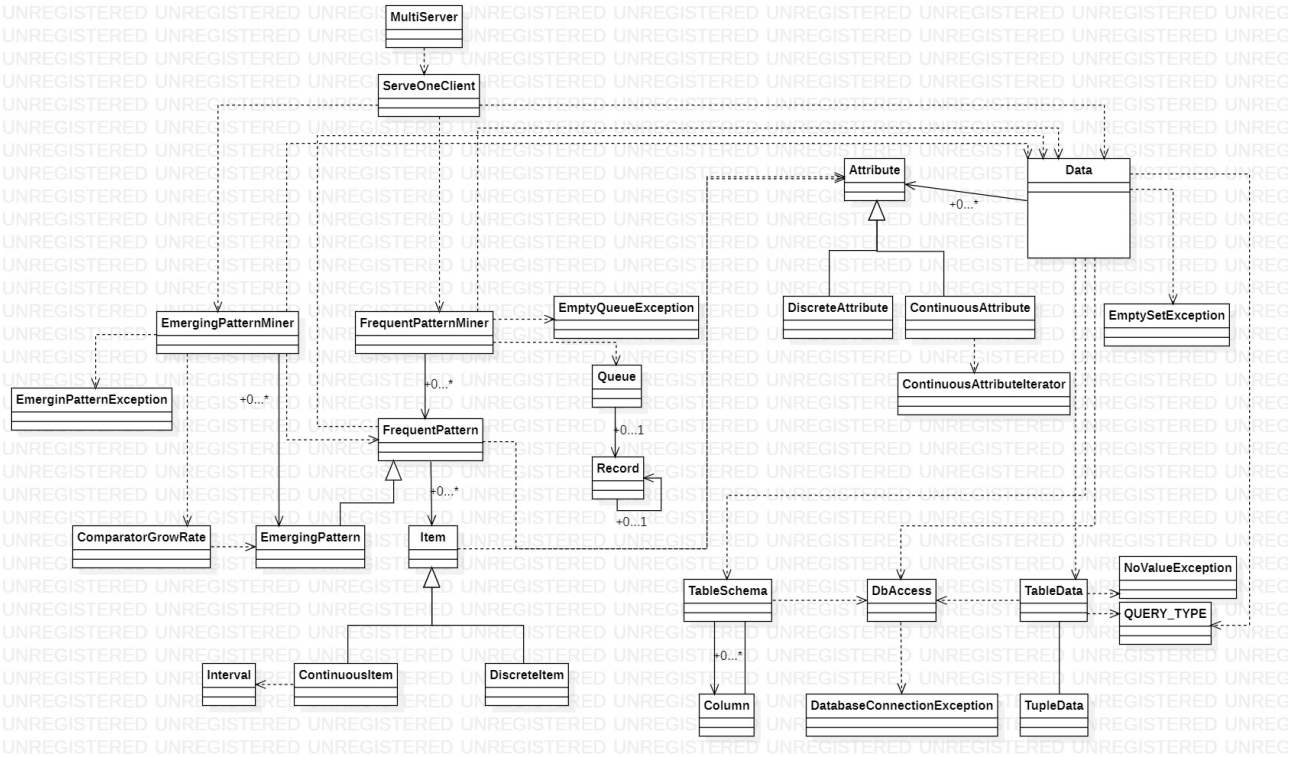
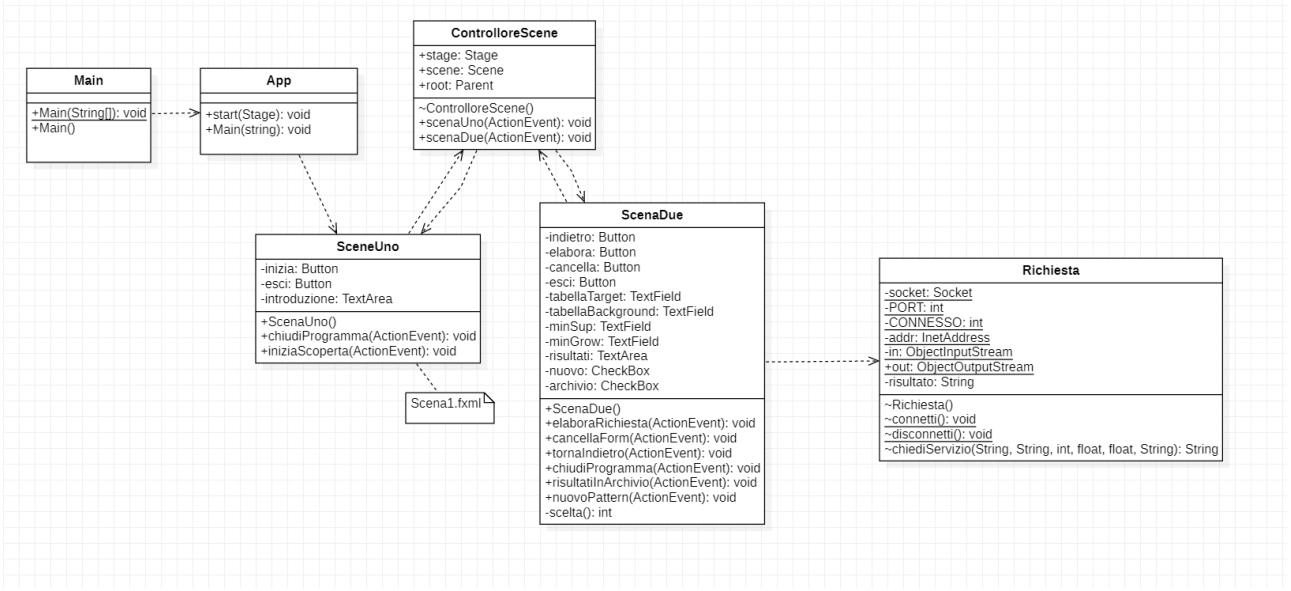


Diagramma delle classi EpMinerClient_esteso



4. Guida all'installazione

4.1 Installazione Server

Per il corretto funzionamento del progetto lato server è necessario:

- installare mySQL 8.0;
- installare Java Runtime Environment (JRE) versione 16;
- linkare al progetto relativo al server il file "*mysql-connector-java-8.0.17*", responsabile della connessione al database mySQL.[2]
  Tale connettore è reperibile nella cartella "*SQL Connector e script*";
- avviare il server mySQL;
- eseguire lo script mySQL presente nella cartella "*SQL Connector e script*".
  Tale script inizializza il database con tabelle e tuple di esempio[3].

Per un più immediato utilizzo, la porta 3306 è preimpostata per accedere al database mySQL.

4.1 Installazione Client

Per il corretto funzionamento del progetto lato client è necessario:

- installare Java Runtime Environment (JRE) versione 16;
- avviare il server. In caso contrario ci sarà un messaggio di errore.

Per un più immediato utilizzo, l'indirizzo IP al quale il client si connette è preimpostato a quello locale 127.0.0.1 con porta 8080.

5. Guida Utente

Nella cartella principale è presente una sotto cartella "*EpMiner_Archive*", nella quale verranno salvati (e caricati) in file *.bin* tutti i pattern trovati. Spostare tale cartella sul Desktop. In essa sono presenti già dei file a scopo di esempio.
Le tabelle di esempio presenti nello script mySQL si chiamano "*playtennistarget*" e "*playtennisbackground*".

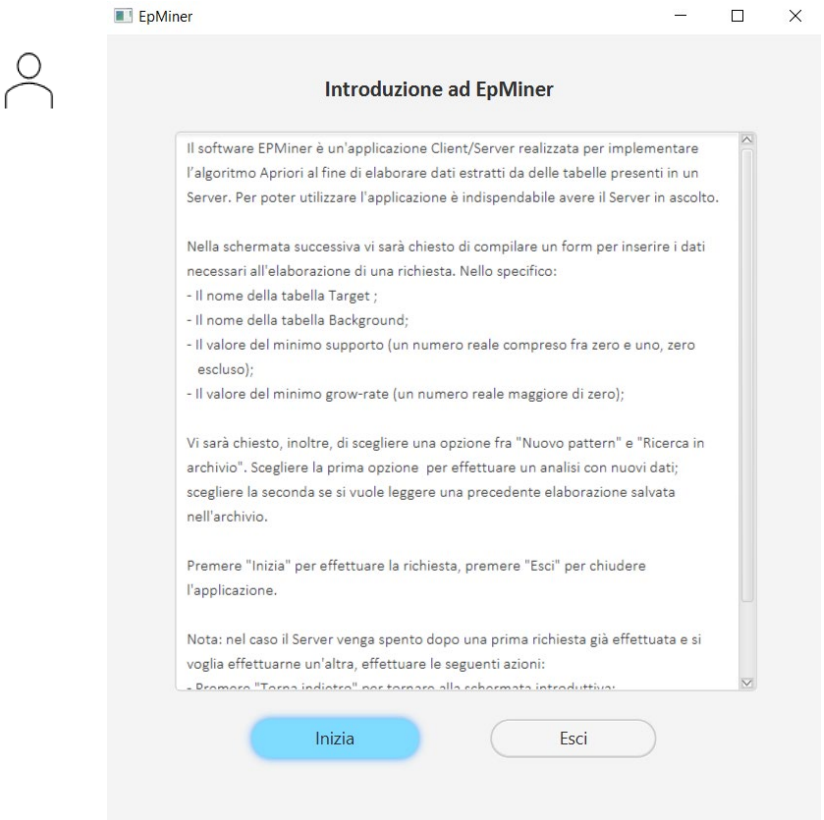
5.1 Guida alla interazione da GUI

Nella cartella "*Eseguibile*" eseguire il file "*Server.bat*" e il file "*EpMinerClient_esteso.bat*". Si apriranno due distinte schermate: una per il server e una interfaccia per il client.

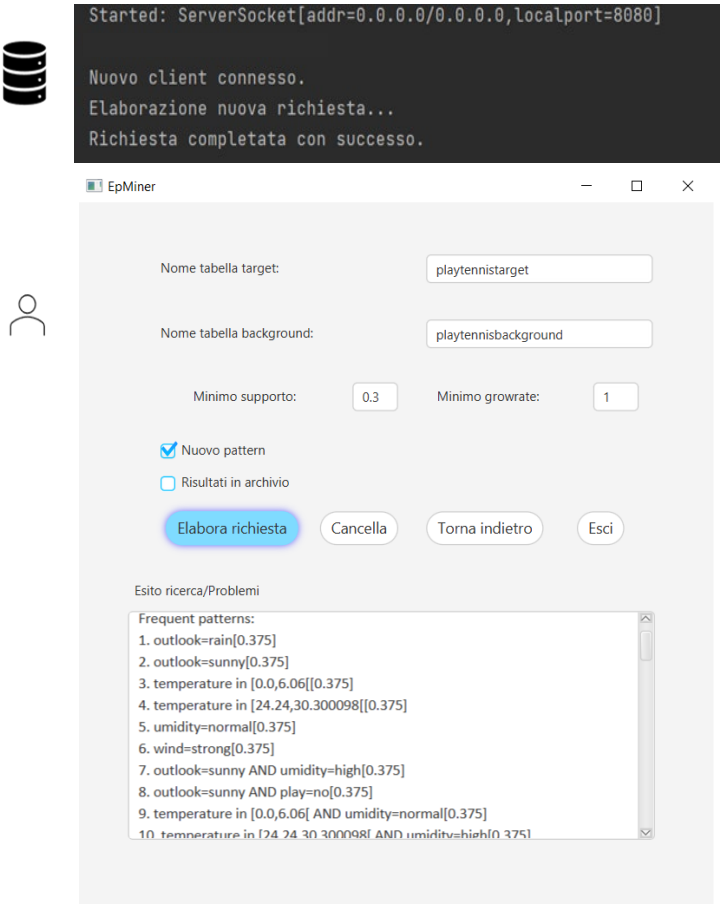
L'interazione lato server è pressoché identica alla versione base del progetto. L'interazione lato client invece cambia completamente: tutti i comandi precedentemente inseriti tramite terminale ora vanno invece inseriti in apposite caselle di testo nella interfaccia.

1) Avvio Server:

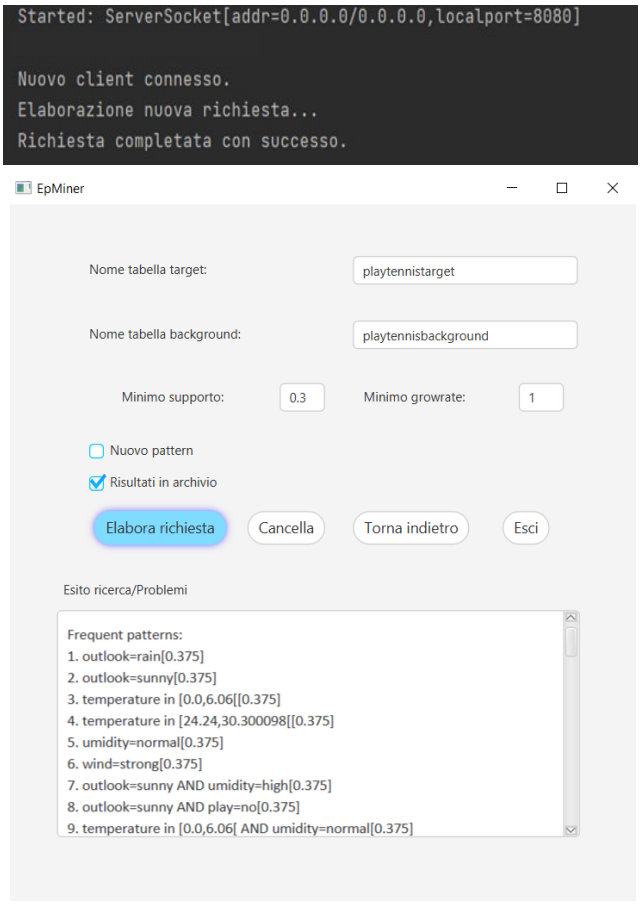
2) Avvio Client:



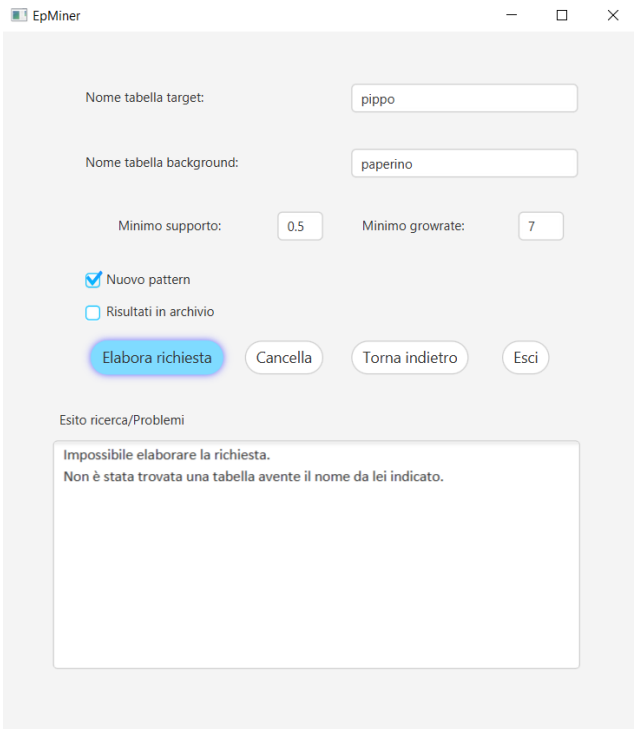
3) Nuova scoperta:

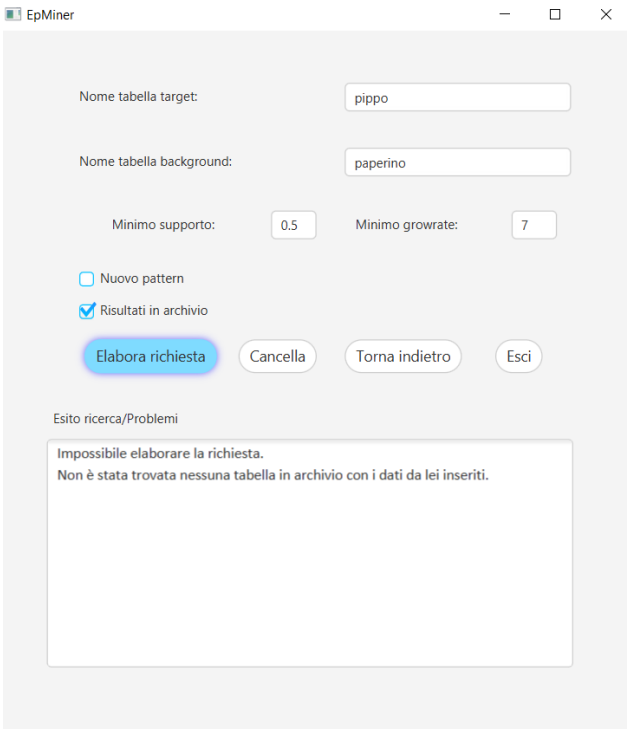


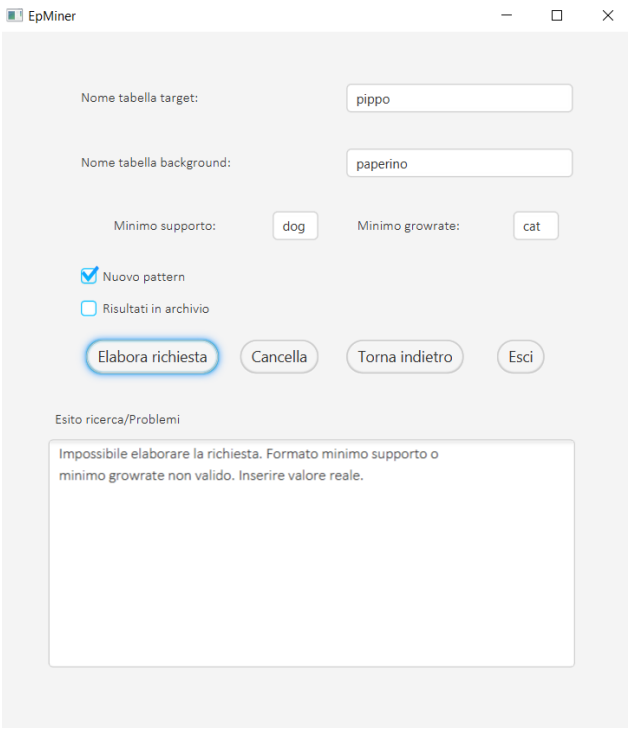
4) Risultati in archivio:

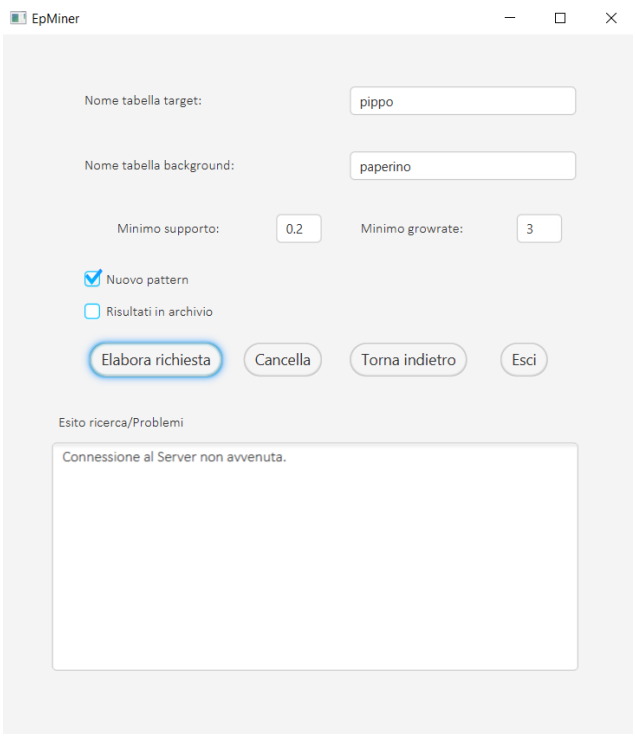


5) Casi particolari:









Cliccando su "*Torna indietro*", ci si sconnette dal server e si torna alla schermata iniziale. Cliccando su "*Esci*" si chiude l'applicazione. Cliccando su "*Cancella*" si ripuliscono tutte le caselle di testo.

6. Note

[1] *JavaFX è disponibile nella cartella "JavaFX". Nel caso in cui non si riesca ad aprire il progetto "EpMinerClient_esteso", creare un nuovo progetto JavaFX e copiare al suo interno la cartella "src" di "EpMinerClient_esteso".*

[2] *Nel caso in cui il file jar non funzionasse a causa del connettore non linkato correttamente, è possibile importare quest'ultimo come libreria esterna al progetto, ed avviare il server direttamente da IDE.*

[3] *In alternativa si può aprire il file con un editor di testo e copiare il contenuto nella Shell MySQL.*