EPMiner

1 Introduzione del progetto

EPMiner è un'applicazione, con interfaccia grafica, che permette, mediante l'utilizzo dell'algoritmo Apriori, di scoprire pattern frequenti e pattern emergenti partendo da due tabelle differenti.
Con il termine **pattern frequenti** si intende l'insieme di item o pattern che occorrono con una frequenza minima chiamata **minimo supporto**, mentre con il termine **pattern emergenti** si intende l'insieme di pattern o item che occorrono con una frequenza minima e che hanno un **grow rate** o tasso di crescita superiore rispetto al passato.

2 Algoritmo Apriori

L'algoritmo utilizzato per la scoperta dei pattern è l'algoritmo Apriori. L'algoritmo Apriori è un classico algoritmo di ricerca delle associazioni. È utilizzato per la generazione dei pattern frequenti, per approssimazioni successive, a partire dagli itemset con un solo elemento. In sintesi, il presupposto teorico su cui si basa l'algoritmo parte dalla considerazione che, se un insieme di oggetti (itemset) è frequente, allora anche tutti i suoi sottoinsiemi sono frequenti, ma se un itemset non è frequente, allora neanche gli insiemi che lo contengono sono frequenti.

Un ambito dove questo algoritmo trova grande applicabilità è il market/basket problem. Per ricavare le associazioni viene impiegato un approccio bottom up, dove i sottoinsiemi frequenti sono costruiti aggiungendo un item per volta (generazione dei candidati); i gruppi di candidati sono successivamente verificati sui dati e l'algoritmo termina quando non ci sono ulteriori estensioni possibili.

I dati in input del problema sono i seguenti:

- un database di transizioni target DTarget
- un database di transizioni di background Dbackground
- un valore minimo di supporto ($0 < \text{minS} \leq 1$)
- un valore minimo di grow rate ($\text{minGr} \geq 1$)

L'obiettivo è trovare i pattern che siano frequenti (cioè con supporto maggiore o uguale a minS) in DTarget e emergenti (grow rate maggiore o uguale a minGr) rispetto a Dbackground. L'algoritmo si divide in due fasi: nella prima fase si scoprono i pattern di lunghezza k a partire dai pattern frequenti di lunghezza k-1, utilizzando la tabella DTarget. Nella seconda fase, si scoprono i pattern emergenti, calcolando i grow rate dei pattern frequenti in DTarget e in Dbackground e, confrontando i valori ottenuti, si selezionano i pattern con grow rate superiore alla soglia prefissata.

Di seguito viene riportato lo pseudo-codice dell'algoritmo:

```
frequentPatternDiscovery(DTarget,minS)→FP
begin
   FP= ∅
   L1= {1-item che compaiono in minS×|D| transazioni di DTarget}
   K=2
   while LK-1 ≠∅ do
   begin
        CK=  candidati generati da Lk-1 aggiungendo un nuovo item
        LK=∅
        for each (p ∈ Ck) do
               if (supporto(p, DTarget) >= minS) then
                     LK=LK∪p
        FP=FP∪Lk
        K=K+1
   end
   return FP
end
```

Figura 1: Pseudo-codice per la scoperta di pattern frequenti.

```
EPDiscovery(DBackground,FP,minGr)→ EP
begin
   EP= ∅
   for each (p ∈ FP) do
       begin
             if (growrate(p,DBacKground) >= minGR) then
                   EP=EP∪ p
       end
return EP
end
```

Figura 2: Pseudo-codice per la scoperta di pattern emergenti.

3 Architettura del progetto

Il progetto EPMiner è stato sviluppato con l'IDE Intellij IDEA. Esso presenta una architettura di tipo client-server, in cui il client può mandare richieste al server per i seguenti servizi:

- scoprire dei nuovi pattern nel database
- caricare pattern salvati nell'archivio

Il client deve specificare i valori di minimo supporto, grow rate e le tabelle di target e background.

Il server elabora le richieste accedendo al database tramite il DBMS MySql. Il server viene avviato sulla porta 8080, pertanto il client deve essere avviato con i parametri "localhost" e la porta 8080. È possibile avviare server e client con parametri diversi.

4 Versione estesa del progetto

Per la versione estesa del progetto il gruppo ha realizzato un'interfaccia grafica lato client utilizzando le librerie offerte da **JavaFX**. Abbiamo deciso di non implementare l'interfaccia grafica lato server in quanto, per il server, non viene richiesta nessuna interazione diretta con l'utente. L'interfaccia grafica è stata implementata interamente tramite codice Java, senza l'utilizzo di tool grafici, ad esempio SceneBuilder ed è stato aggiunto un foglio di stile **CSS** per personalizzare lo stile grafico dell'applicazione.

5 Guida all'installazione

Di seguito sono riportate le linee guida da seguire per il corretto funzionamento del server:

1. installare la Java Runtime Environment, versione 16 o superiore
2. installare il DBMS MySql, versione 5.7 o superiore
3. dopo aver avviato il server mysql, eseguire lo script "mysqlScript" presente nel seguente percorso: "EPMiner\Versione estesa\Server\mysqlScript.sql"

Le linee guida da seguire per il funzionamento del client sono:

1. installare la Java Runtime Environment, versione 16 o superiore
2. avviare il server prima di avviare il client

6 Guida utente

6.1 Avvio del server tramite batch

Per il corretto funzionamento del programma è necessario avviare prima il server. Per avviare il server bisogna eseguire il file serverBatch.bat, presente nel seguente percorso: "EPMiner\Versione estesa\Server\serverBatch.bat", tramite uno dei seguenti metodi:

- fare doppio clic con il mouse sul file batch
- eseguire il file batch tramite riga di comando (es. cmd o PowerShell)

In questo modo il server verrà avviato senza parametri sulla porta 8080. Se si ha necessità di avviare il server con **parametri** si può modificare il file serverBatch tramite un qualsiasi editor di testo, aggiungendo i parametri come segue:

```
cd build/libs

java -jar Server-1.0-SNAPSHOT.jar 45621
PAUSE
```

In questo modo il server verrà avviato sulla porta 45621 (se libera).

6.2 Avvio del client tramite batch

Per avviare il client bisogna eseguire il file clientBatch.bat presente nel seguente percorso: "EPMiner\Versione estesa\Client\clientBatch.bat", tramite uno dei seguenti metodi:

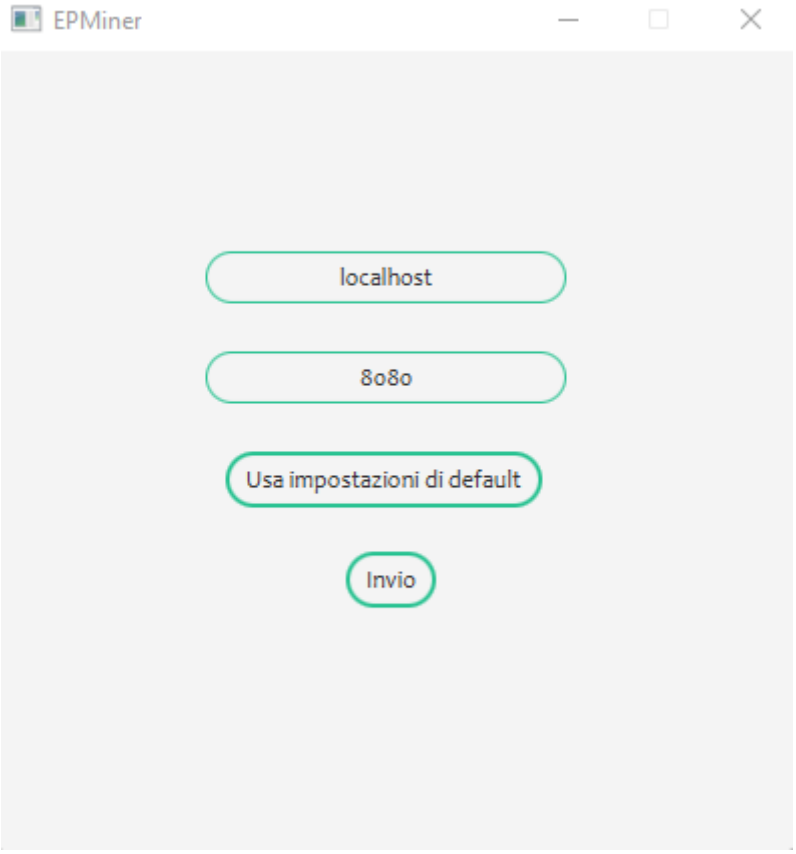
- fare doppio clic con il mouse sul file batch
- eseguire il file batch tramite riga di comando (es. cmd o PowerShell)

Nella versione estesa del progetto, viene chiesto all'utente di inserire l'indirizzo del server e il numero di porta del processo, quindi non bisogna modificare il file clientBatch.

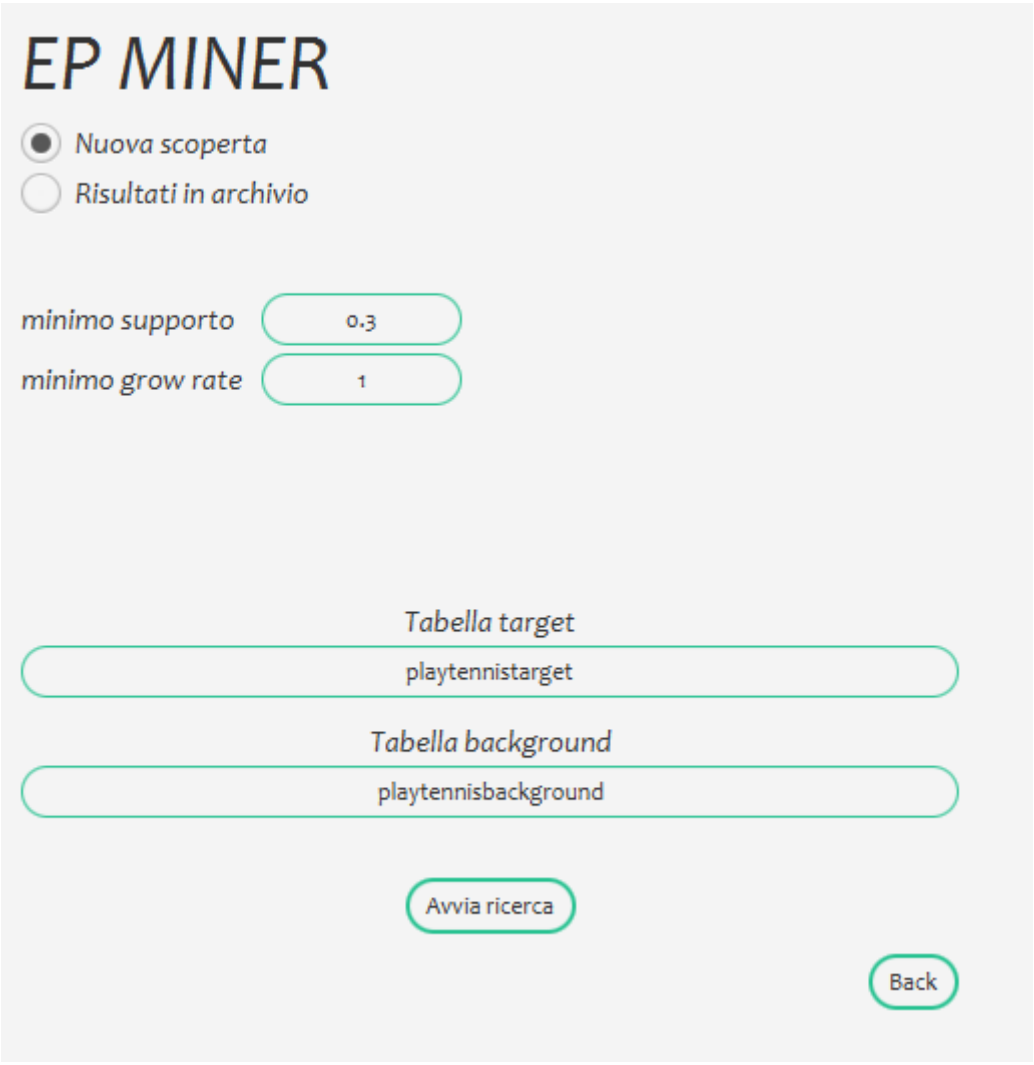
6.3 Esempi di test

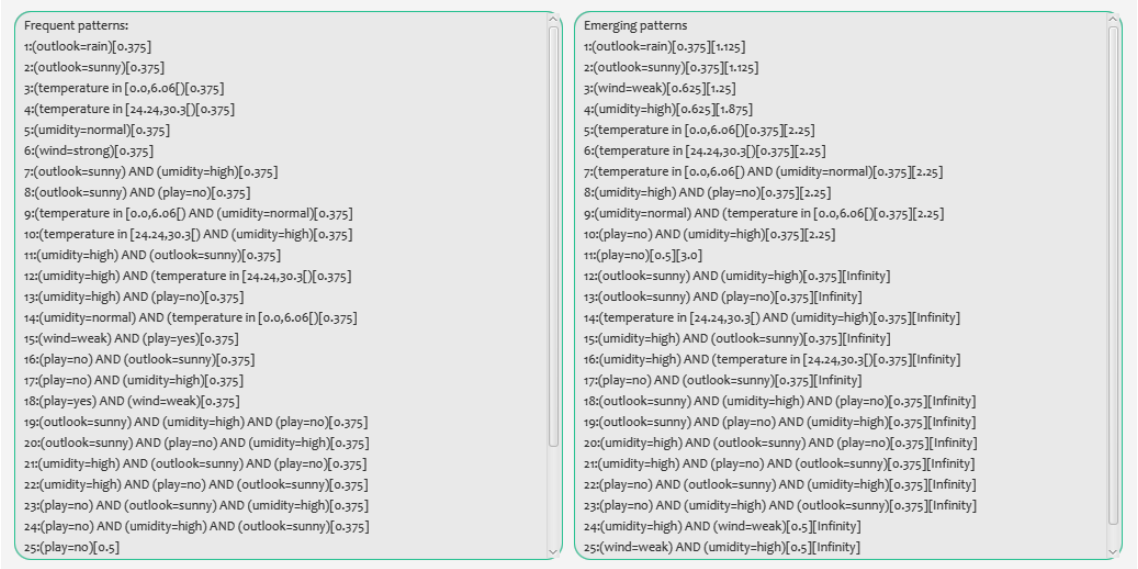
Di seguito sono riportati esempi di test.

All'avvio del client viene visualizzata una schermata per l'inserimento dei dati relativi all'indirizzo e porta del server per poter effettuare la connessione. Usando l'opzione "Usa impostazioni di default" i campi verranno automaticamente compilati con i parametri di default del server.

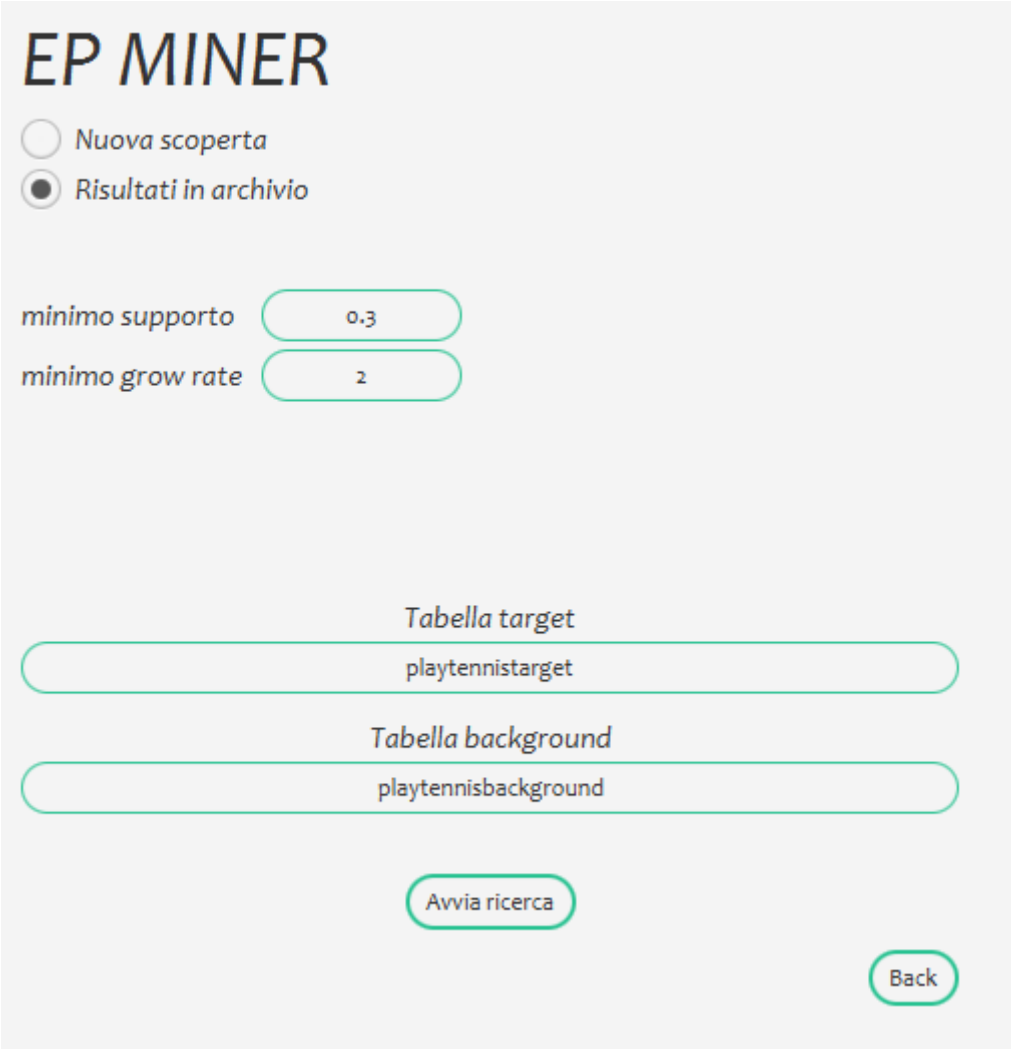


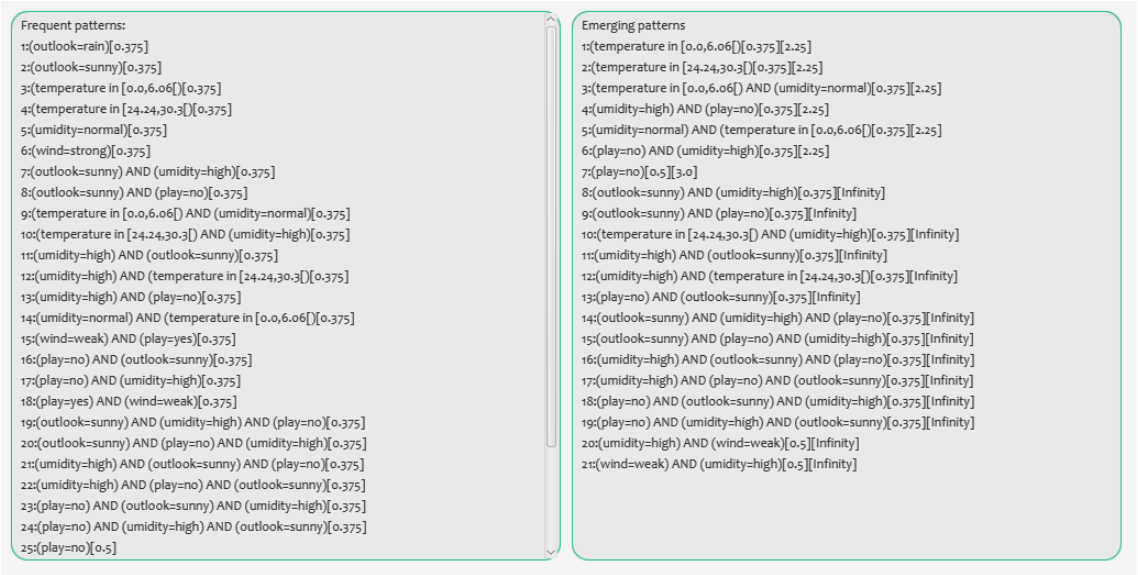
Se la connessione è avvenuta con successo, verrà visualizzata la schermata principale del programma in cui è possibile usufruire dei servizi offerti dal server.
Un esempio di test di una richiesta "Nuova Scoperta":



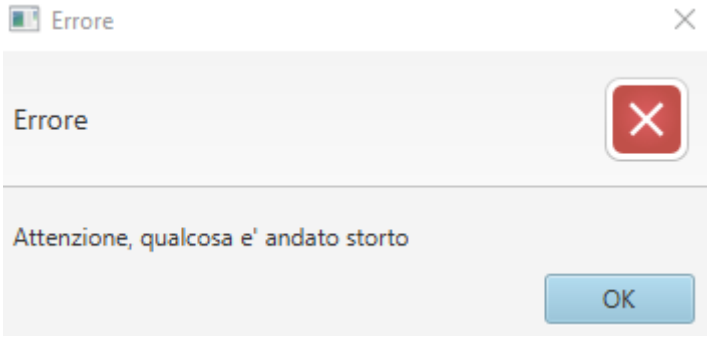


Un esempio di test di una richiesta "Risultati in archivio":





Se la richiesta non viene eseguita con successo, verrà visualizzato un messaggio di errore:



7 Diagrammi UML

Diagramma delle classi del package Mining:

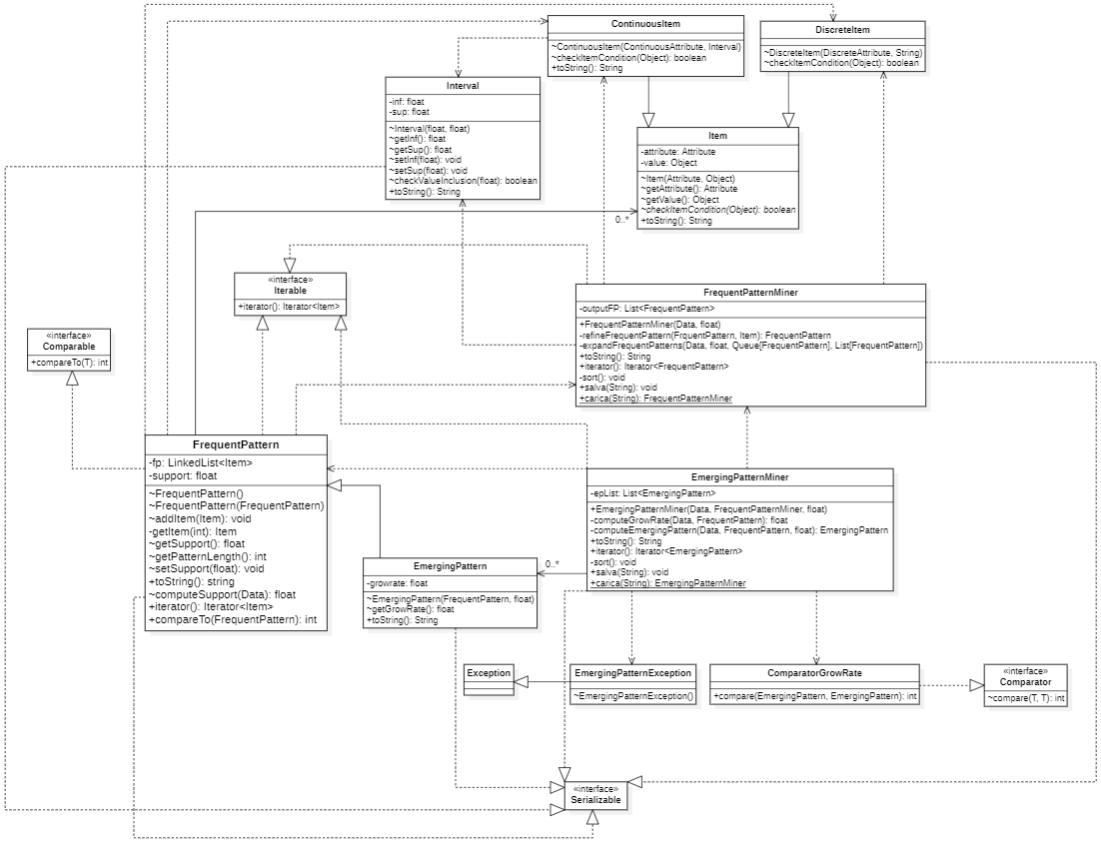


Diagramma delle classi del package Data:

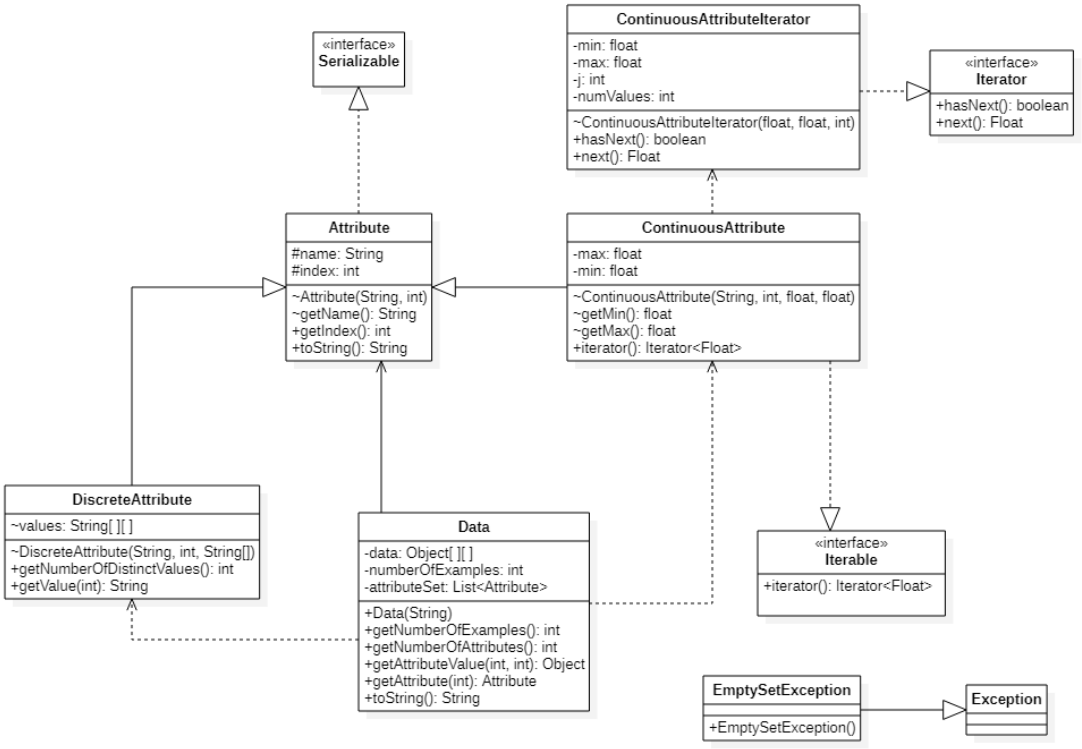


Diagramma delle classi del package Utility:

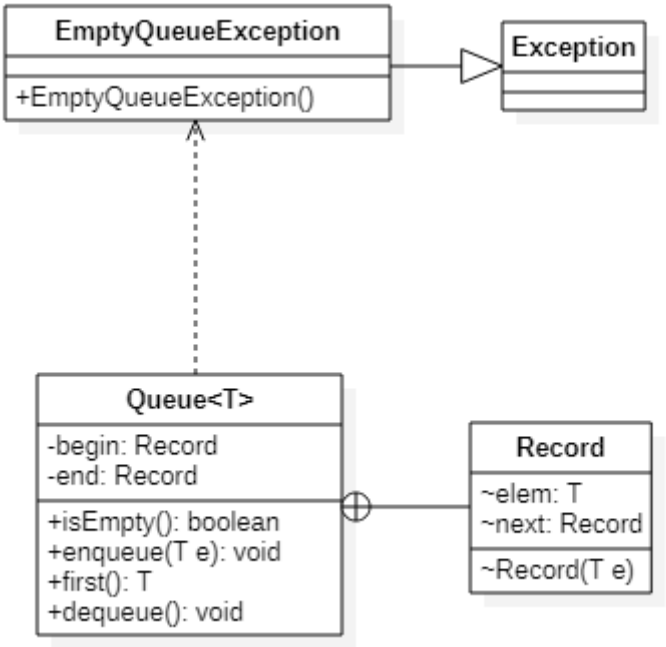


Diagramma delle classi del package Database:

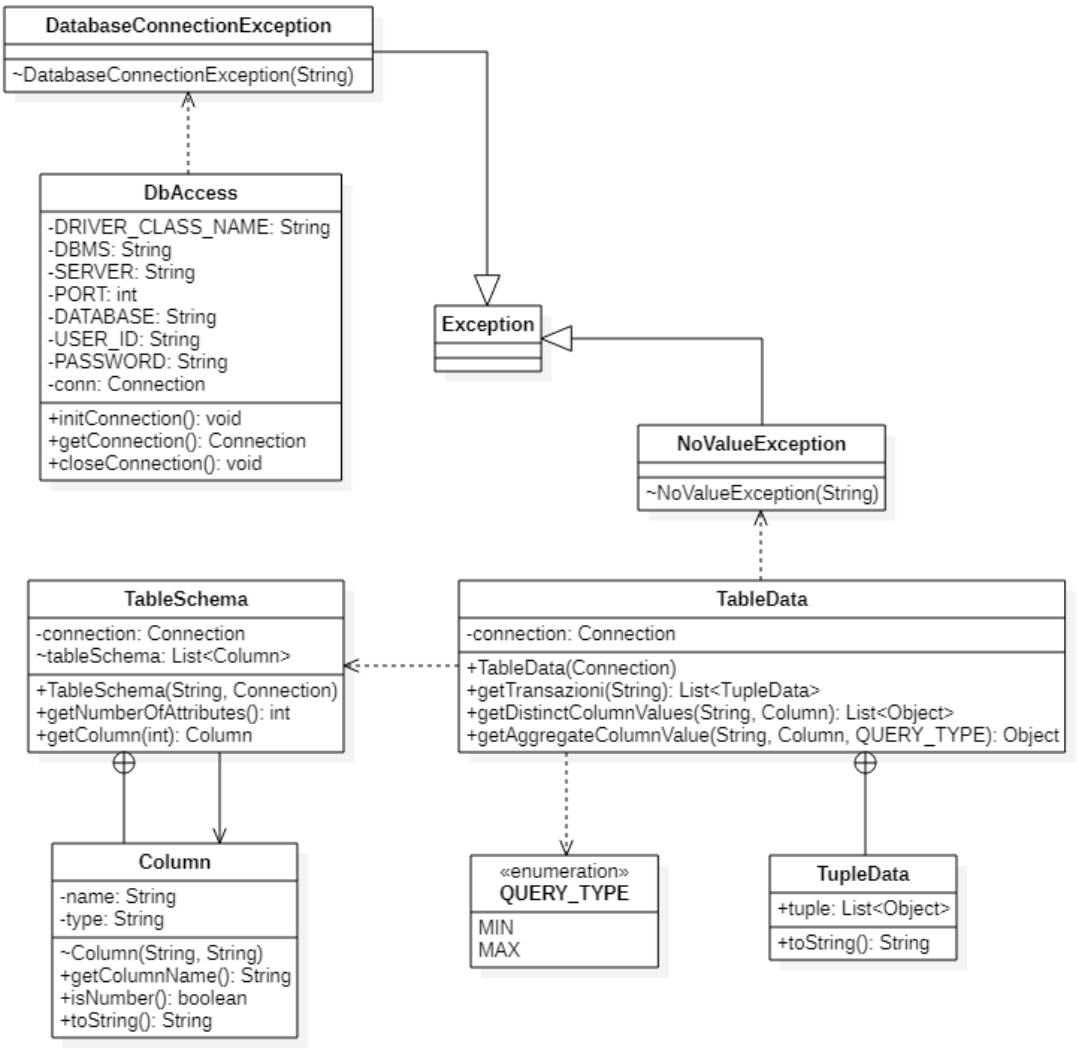


Diagramma delle classi Client:

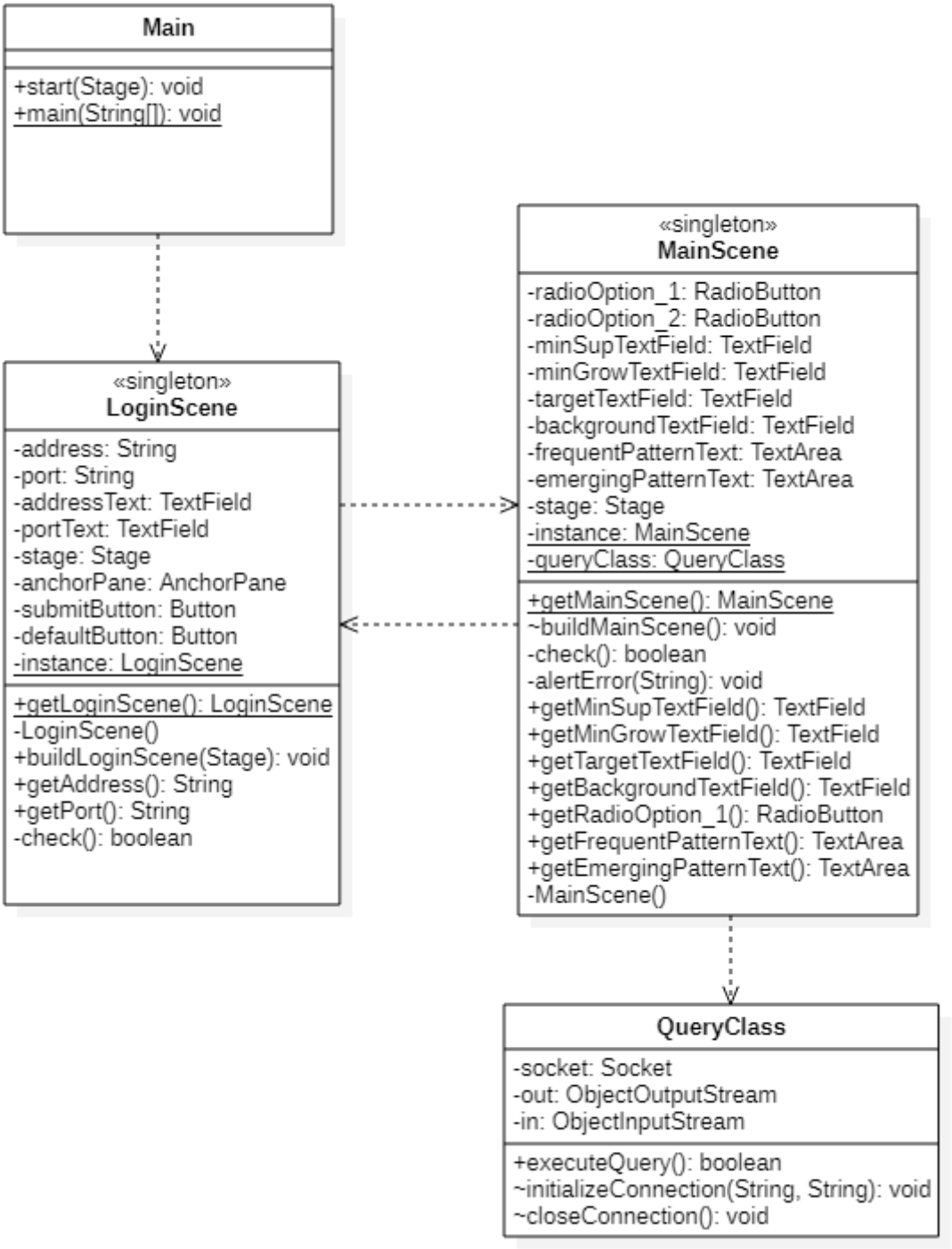


Diagramma del package Client:

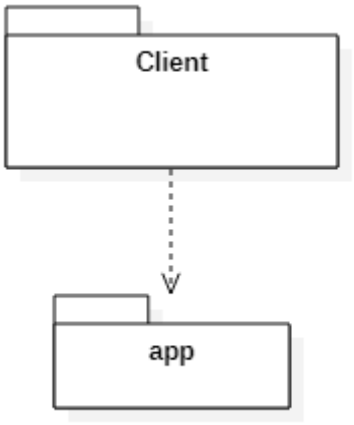
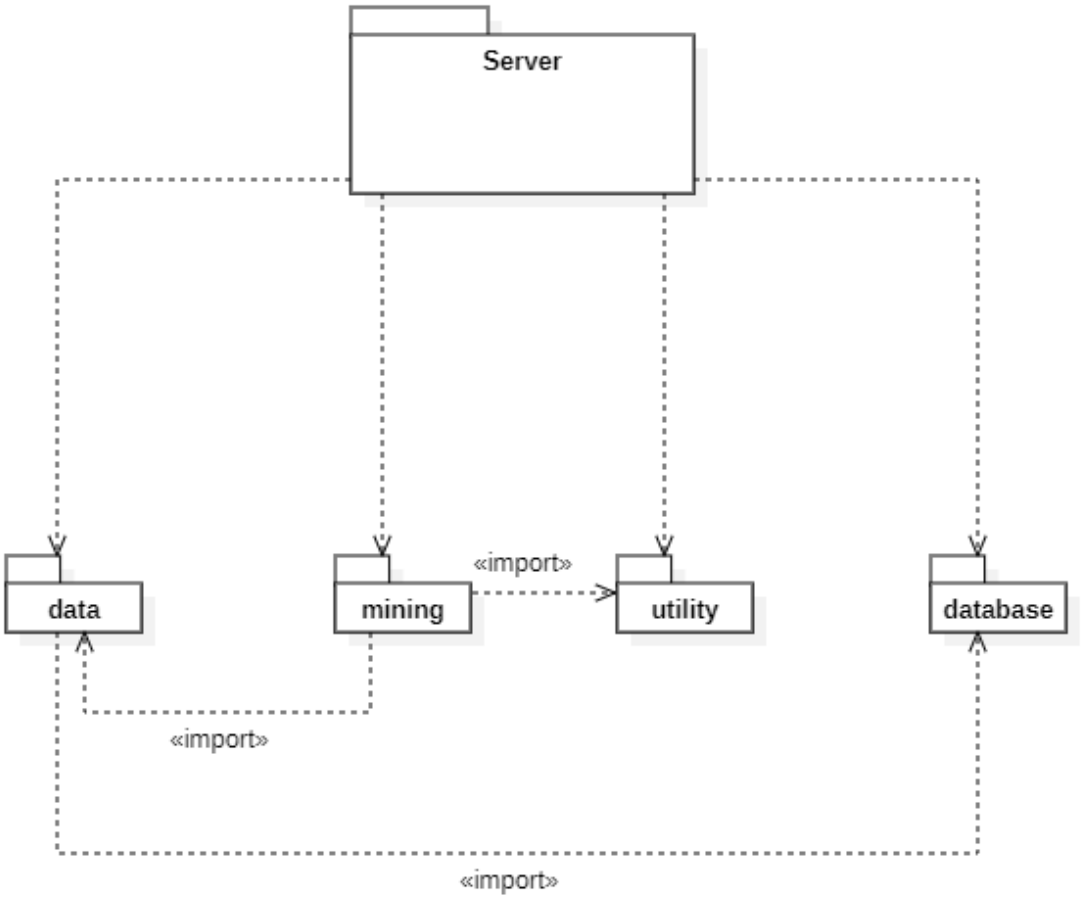

Client
app

Diagramma del package Server:



8 Javadoc

È possibile visualizzare la documentazione del client e del server tramite le pagine presenti nei seguenti percorsi:
"EPMiner\Versione estesa\Client\build\docs\javadoc\allclasses-index"
"EPMiner\Versione estesa\Server\build\docs\javadoc\allclasses-index"

9 Studenti del gruppo

Il progetto è stato realizzato da:

- Marco Angelo Lillo - MAT: 717683 - m.lillo21@studenti.uniba.it
- Daniele Cecca - MAT: 718588 - d.cecca1@studenti.uniba.it
- Ferrulli Francesco - MAT: 716836 - f.ferrulli14@studenti.uniba.it

[1]

[1] È possibile scaricare il progetto EPMiner dalla repository https://github.com/Ferru2000/EPMiner