piDIANA

Pathways Integration for the DIANA project

Progetto per il corso di Bioinformatics aa 2016/17

Francesco Manfredi, Davide Micarelli

Indice

Presentazione

PiDiana è una utility per l'**aggiunta di dati su pathways e malattie al database a grafo del progetto DIANA**. La utility è stata sviluppata da Francesco Manfredi e Davide Micarelli come progetto finale per il corso di Bioinformatics della Professoressa Antinisca Di Marco.

Per l'esecuzione è necessario disporre di un'interprete Python, di una connessione ad internet e delle credenziali di accesso al database del progetto DIANA. È possibile impostare un proprio database che simuli quello di produzione con la struttura minima per eseguire dei test attraverso uno script fornito insieme alla utility.

Utilizzo

PiDiana si presenta in forma di un piccolo numero di moduli Python, il principale dei quali è il modulo core.py.

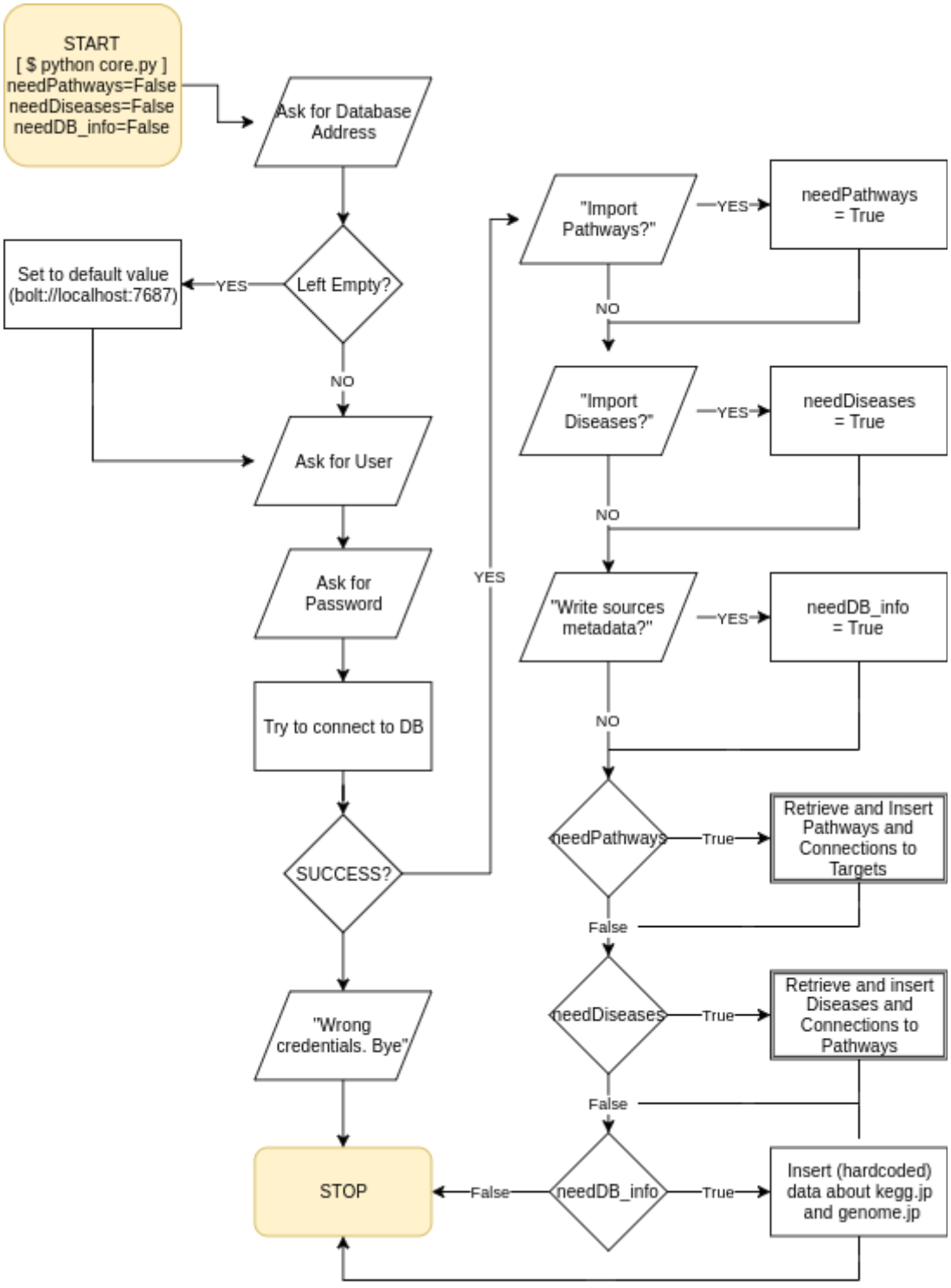
Per avviare la procedura guidata è sufficiente dare il seguente comando in un terminale:

```
$ python core.py
```

Lo script core si occupa di richiedere all'utente le informazioni necessarie per connettersi al database a grafo e quali delle operazioni si intende effettuare o saltare. Raccolte queste informazioni procede ad operare raccogliendo i dati dalle fonti ed inserendoli nel database, fornendo nel frattempo un riscontro all'utente sul numero di entità inserite e su eventuali casi particolari (es: pathways privi di classe o malattie che non risultano collegate ad alcun pathway).

Le operazioni di inserimento di pathways e diseases richiedono un tempo piuttosto lungo a causa del fatto che per ogni entità inserita è necessario determinare a quali altre entità va collegata. Il problema è stato mitigato in parte con la creazione preventiva di indici sulle proprietà utilizzate per le ricerche (:Target(geneid), :Pathway(entry) e :Disease(entry)).

Flow Chart


START
[$ python core.py]
needPathways=False
needDiseases=False
needDB_info=False
Ask for Database Address
Left Empty?
YES
Set to default value (bolt://localhost:7687)
NO
Ask for User
Ask for Password
Try to connect to DB
SUCCESS?
YES
"Wrong credentials. Bye"
STOP
"Import Pathways?"
YES
needPathways = True
NO
"Import Diseases?"
YES
needDiseases = True
NO
"Write sources metadata?"
YES
needDB_info = True
NO
needPathways
True
Retrieve and Insert Pathways and Connections to Targets
False
needDiseases
True
Retrieve and insert Diseases and Connections to Pathways
False
needDB_info
True
Insert (hardcoded) data about kegg.jp and genome.jp
False

Setup di un ambiente di test in locale

Insieme a piDIANA sono forniti uno script ed un pacchetto dati per inserire in un database Neo4j i dati di partenza necessari per poter effettuare dei test utili (si tratta di circa 20000 nodi Target che sono l'unico collegamento tra i nuovi dati ed i dati già presenti).

Per eseguirlo è sufficiente dare il seguente comando:

```
$ python downloadSomeTarget.py
```

L'intera esecuzione richiede pochi secondi grazie all'esecuzione in batch da 200 statements per volta.

Considerazioni sull'opportunità di una modalità update

Inizialmente si era immaginato di poter creare una modalità di esecuzione di aggiornamento dei dati che controllasse velocemente la presenza di nuovi dati e differenze tra le sorgenti ed il database di DIANA ed in caso di necessità procedesse ad inserire solo i nuovi dati.

In fase di sviluppo ci si è resi conto che eseguire un confronto sui dati ed aggiornarli condizionatamente al rilevamento di differenze sarebbe stato in ogni caso molto più esoso rispetto a cancellare completamente i dati inseriti da questo script e riinserirli nuovamente.

Tecnologie Utilizzate

Tutta la business logic è implementata con moduli Python senza l'utilizzo di librerie specifiche. Non sono quindi presenti particolari dipendenze per l'esecuzione.

Il codice è stato scritto in maniera il più possibile compatibile con le versioni 2.7 e 3.5 di Python e dovrebbe funzionare con entrambe, ma si consiglia in ogni caso l'esecuzione con Python 3.5.

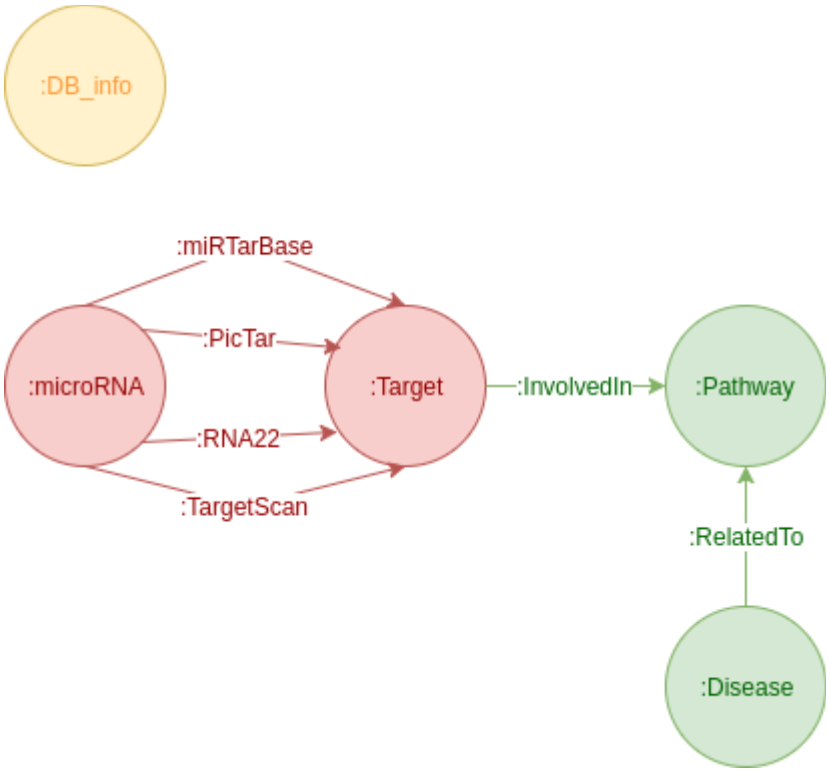
Il database a grafo utilizzato è Neo4j.

Il driver di connessione al database è neo4j.v1, disponibile come package per Python.

La comunicazione tra il driver neo4j ed il server avviene tramite protocollo bolt.

Schema dei dati

I dati inseriti da questa utility vanno a collegarsi ai nodi Target già presenti nel database DIANA secondo lo schema seguente:



Nel grafo mostrato i nodi e gli archi rossi sono i dati di partenza, quelli verdi sono quelli inseriti come nuova struttura ed il nodo giallo rappresenta le informazioni sulle sorgenti dati, alcune già presenti, altre inserite.

Segue una descrizione schematica delle proprietà attribuite ai nuovi dati inseriti.

Entità	Proprietà	Significato
nodo**:Pathway**	**entry**: un codice univoco che lo identifica nel database di provenienza **name**: nome del pathway **class**: categorizzazione standard del pathway **fullData**: url al quale reperire informazioni esaustive	Un pathway che può coinvolgere certe proteine ed il cui funzionamento anomalo può causare malattie.
nodo:**Disease**	**entry**:un codice univoco che identifica la malattia nella sorgente di provenienza **name**: nome della malattia	Una malattia che è in qualche modo collegata ad uno o più pathways.
arco:**InvolvedIn**	-	Un **Target** può essere coinvolto in un **Pathway**

arco:**RelatedTo**	-	Una **malattia** può essere correlata ad un **pathway.**

Sorgenti dei dati

Tutti i dati sono stati raccolti dal sito della Kyoto Encyclopedia of Genes and Genomes (http://www.kegg.jp) attraverso la API messa a disposizione all'indirizzo http://rest.kegg.jp. Nome e link a questo sito sono stati inseriti in un nodo :DB_info come quelli già presenti nel database.

Un secondo nodo :DB_info è stato inserito per il sito di **GenomeNet** (di cui KEGG è una parte) perché può essere sfruttato per raggiungere una **visualizzazione grafica dei pathways** aggiungendo la proprietà entry all'indirizzo "http://www.genome.jp/kegg-bin/show_pathway?map=".