UNIVERSITA`

DEGLI STUDI DI MESSINA

Dipartimento di Scienze Matematiche e Informatiche Scienze Fisiche e Scienze della Terra

## Corso di Laurea in Informatica

RELAZIONE - BASI DI DATI II

**“ANALISI DELLA RETE POLITICA DI DONALD TRUMP”**

Sonia Trifiletti Federica Zuccarello

Prof. Antonio Celesti

Anno Accademico 2018-2019

# Indice

[Indice 2](#_TOC_250031)

[Problematica affrontata 4](#_TOC_250030)

[Introduzione 4](#_TOC_250029)

[I DBMS NoSQL 4](#_TOC_250028)

[Soluzioni DB utilizzate 5](#_TOC_250027)

[Neo4j 5](#_TOC_250026)

[MongoDB 5](#_TOC_250025)

[Strumenti Utilizzati 6](#_TOC_250024)

[Progettazione 7](#_TOC_250023)

[Data Model 7](#_TOC_250022)

[Neo4j: Modello di dati grafico 8](#_TOC_250021)

[Tecnica di Modellazione di un Grafo 9](#_TOC_250020)

[MongoDB: Modello di dati documentale 9](#_TOC_250019)

[Tecnica di Modellazione di un Documento 10](#_TOC_250018)

[Implementazione 12](#_TOC_250017)

[Importazione in Neo4j 12](#_TOC_250016)

[Importazione in MongoDB 13](#_TOC_250015)

[Implementazione Query 14](#_TOC_250014)

[Query 1 14](#_TOC_250013)

[Query 2 15](#_TOC_250012)

[Query 3 15](#_TOC_250011)

[Query 4 16](#_TOC_250010)

[Query 5 16](#_TOC_250009)

[Query 6 17](#_TOC_250008)

[Query 7 17](#_TOC_250007)

[Generatore Dati 18](#_TOC_250006)

[Esperimenti 19](#_TOC_250005)

[Risultati Neo4j - Dataset di 1000 e 10000 19](#_TOC_250004)

[Confronto dei risultati 20](#_TOC_250003)

[Risultati MongoDB - Dataset di 1000 e 10000 22](#_TOC_250002)

[Confronto Dataset da 1000 per i due DB 23](#_TOC_250001)

[Conclusioni 25](#_TOC_250000)

# Problematica affrontata

## Introduzione

Il 15 gennaio BuzzFeed ha rilasciato un ampio set di dati sulle connessioni di Donald Trump, incluse persone, organizzazioni e la natura delle loro rela- zioni. In questo progetto mostriamo la visualizzazione della rete politica di Donald Trump basata su quel set di dati. In questo tipo di contesto la solu- zione migliore per l’analisi dei dati `e sicuramente l’utilizzo di un DB NoSql in cui l’informazione piu` preziosa risiede nelle relazioni tra gli oggetti piut- tosto che nei dati stessi, Neo4j (il piu` famoso Graph Database) esemplifica perfettamente questo modello di offerta. Per dimostrare la totale bont`a del metodo scelto, si `e deciso di implementare la stessa problematica utilizzando un altro DB Nosql, MongoDB (Document Database), che segue una filosofia opposta, cio`e debolmente orientata alle relazioni.

## I DBMS NoSQL

Negli ultimi anni, con l’avvento dei *Big Data* `e incredibilmente aumentata la popolarit`a delle tecnologie di immagazzinamento di informazioni conosciute con il nome di **NoSQL** acronimo che sta per *Not only SQL*.

I NoSQL Database Management System sono sistemi software che consento- no di immagazzinare e organizzare i dati senza fare affidamento sul model- lo relazionale. Ne esistono tantissimi e possono avere le caratteristiche piu` disparate, vediamo quali sono i piu` importanti:

* Key-Value store.
* Document-oriented.
* Column store.
* Graph DBMS.

## Soluzioni DB utilizzate

### Neo4j

**Neo4j** `e un famoso database grafico, cio`e un database utilizzato per model- lare i dati sotto forma di grafo, qui i nodi di un grafo rappresentano le entit`a mentre le relazioni descrivono l’associazione di questi nodi.

E` DBMS open source transazionale, prodotto dalla software house Neo Tech-

nology, sviluppato completamente in Java, robusto, scalabile e ad alte pre-

stazioni.

E` dotato di:

* transazioni ACIDe,
* alta disponibilit`a,
* tecniche di memorizzazione per miliardi di nodi e relazioni,
* alta velocit`a di interrogazione tramite attraversamenti,
* linguaggio di interrogazione dichiarativo e grafico (**Cypher**).

E` un DBMS schema-less, ci`o sta a significare che i suoi dati non devono

attenersi ad alcuna struttura di riferimento prefissata, inoltre non possiede una politica di accesso controllata.

### MongoDB

**MongoDB** `e un database NoSQL orientato ai documenti, che nasce nel 2007 in California come servizio da utilizzare nell’ambito di un progetto piu` ampio, ma che presto `e diventato un prodotto indipendente ed open-source. Esso memorizza i documenti in JSON, piu` precisamente in formato **BSON** (*concettualmente simile a un oggetto JSON ma codificato in forma binaria)*. Il documento `e fondamentalmente un albero che pu`o contenere molti dati, anche annidati. I documenti sono raggruppati in **collezioni** che possono essere anche eterogenee, non esiste quindi uno schema fisso per i documenti, tra le collezioni non ci sono relazioni o legami garantiti.

Le caratteristiche chiave di MongoDB sono:

alta disponibilit`a, dal momento che la replicazione di un database pu`o avvenire in modo molto semplice,

*•*

scalabilit`a orizzontale, ossia la possibilit`a di distribuire le collezioni in cluster di nodi, in modo da supportare grandi quantit`a di dati senza influire pesantemente sulle performance.

*•*

* alta velocit`a, a scapito delle propriet`a ACIDe.

## Strumenti Utilizzati

*DataTrumpWorld.csv* : file csv contenente i dati relativi alla rete poli- tica di Trump utilizzati per la stesura di questo progetto

*•*

* *Browser Neo4j* : Interfaccia grafica di Neo4j
* *Neo4j py*: API Python per Neo4j
* *MongoDB Compass*: Interfaccia grafica di Mongo DB
* *PyMongo*:API Python per MongoDB
* *PHP Mongo*: API PHP per MongoDB
* *Sublime Text* : editor di testo
* *Eclipse Java*: ambiente di sviluppo per programmi Java

*XAMPP* : una piattaforma software open source costituita dal server web Apache

*•*

* *Mozilla Firefox* : browser usato per sviluppo e test

# Progettazione

## Data Model

Nell’ambito delle basi di dati, il modello dei dati rappresenta un insieme di strumenti concettuali, detto “formalismo”, che consta di tre componenti essenziali:

* insieme di strutture dati con opportuni operatori
* notazione per specificare i dati tramite le strutture dati del modello
* insieme di operazioni per manipolare i dati.

Qualsiasi modello dei dati deve risolvere due principali quesiti:

* come rappresentare le entit`a e i loro attributi
* come rappresentare le associazioni.

Per descrivere bene questi aspetti sopra citati `e stato costruito un modello concettuale, in particolare **E-R**, per rappresentare la gestione degli oggetti piu` importanti e le loro associazioni.(fig. 1)

Chiaramente lavorando con Database Nosql tale grafico avr`a il solo obiettivo di mostrare in maniera generale il comportamento dei dati ma non risulter`a totalmente vincolante ai fini della stesura del modello logico di ciascun DB utilizzato.

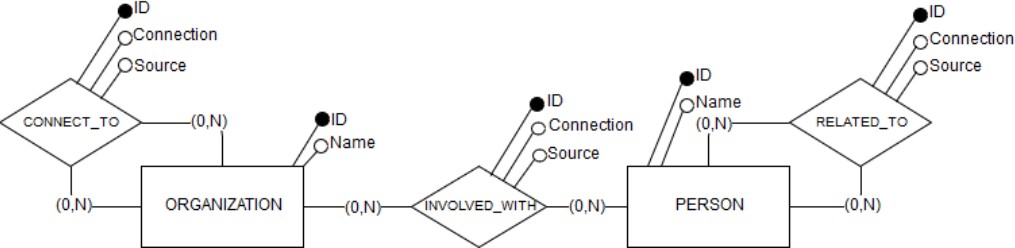


Figura 1: Modello Concettuale, E-R

Da un’attenta analisi del Dataset messo a disposizione dalla traccia ci si

`e resi conto che le entit`a coinvolte potevano essere suddivise direttamente in base al loro tipo perch´e presenti in solo due forme: *ORGANIZATION* e *PERSON*, per tale ragione questi due oggetti diventano le entit`a fondamentali dell’E-R aventi due attributi: **ID** (identificatore) e Name. Esistono tre tipi di associazioni descritte in base alle entit`a coinvolte:

1. *CONNECT TO*, descrive relazioni tra due organization (associazione ricorsiva molti a molti)
2. *INVOLVED WITH*, descrive relazioni tra un organization e un person (associazione binaria molti a molti)
3. *RELATED TO*, descrive relazioni tra due person (associazione ricorsiva molti a molti)

Tutte le relazioni hanno gli stessi attributi: **ID** (identificatore), Connec- tion (natura delle loro relazioni), Source (Sito web con esemplificazione della relazione).

## Neo4j: Modello di dati grafico

Neo4j Graph Database memorizza tutti i suoi dati sotto forma di grafo, perci`o

`e naturale chiedersi cosa sia un grafo.

Un grafo `e un insieme di nodi connessi da relazioni. I grafi rappresentano le entit`e con i nodi, e il modo nel quale queste entit`a si rapportano con il mondo, con le relazioni. I dati del nodo e delle relazioni sono rappresentate in termini di propriet`a (coppie chiave-valore).

Dunque i principali elementi costitutivi di un modello di dati grafico sono:

* I nodi
* Le relazioni
* Le Propriet`a

### Tecnica di Modellazione di un Grafo

Partendo dal nostro schema E-R, costruito precedentemente, effettuiamo una vera e propria traduzione ad un modello a grafo che cerchi di descrivere al meglio il dominio del problema.

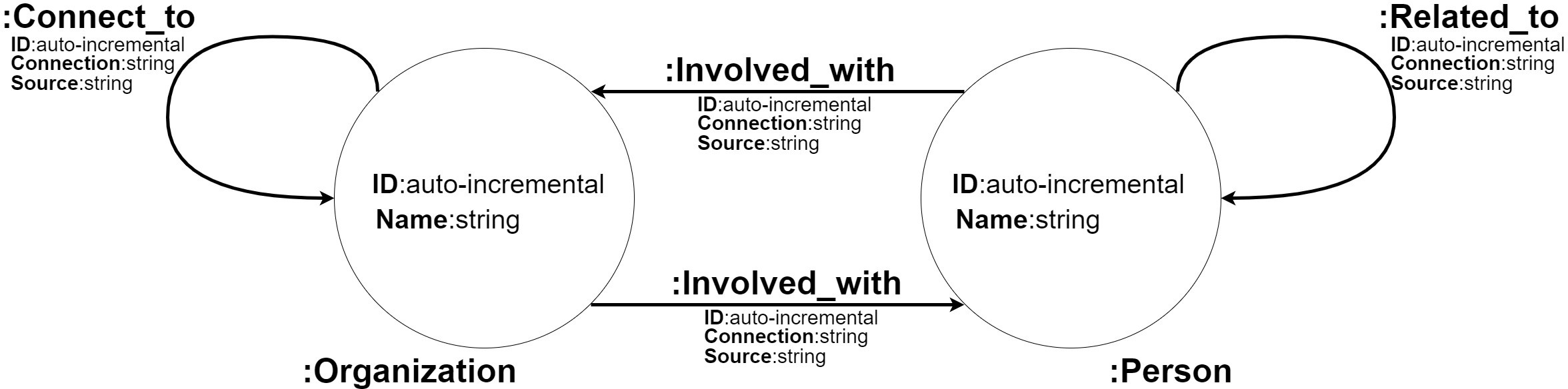


Figura 2: Modello Grafico generalizzato

Osservando questo modello si intuisce bene la natura del database. I cerchi rappresentano i nodi che, ovviamente in questo contesto, sono i due oggetti: *Organization* e Person e le frecce rappresentano le relazioni cio`e: *Connect to*, *Involved with*, *Related to*, le relazioni hanno direzioni unidirezionale e bidi- rezionale. Entrambi i nodi e le relazioni contengono propriet`a: *ID* che `e un valore incrementale univoco e *Name*, *Connection*, *Source* che sono valori di tipo stringa.

## MongoDB: Modello di dati documentale

In MongoDB l’unit`a fondamentale di memorizzazione delle informazioni `e rappresentata dal documento. Tale termine non ha una definizione precisa e univoca, in quanto un documento `e un qualsiasi insieme di informazioni concettualmente coerenti raggruppate secondo una data codifica.

I documenti vengono raccolti in collezioni (*collection*) le quali rappresentano una suddivisione concettuale dei dati in modo analogo alle tabelle dei da- tabase relazionali; a differenza di quest’ultime, in MongoDB una collezione non ha vincoli strutturali e pertanto i documenti presenti all’interno possono avere campi diversi, per tipo e valore. A loro volta, le collection risiedono nel database che rappresenta l’universo del sistema di cui si vogliono gestire i dati.

|  |  |
| --- | --- |
| **Modello Relazione** | **Modello documentale** |
| DataBase | Database |
| Tabella | Collezione |
| Riga | Documento |
| Colonna | Campo |
| Chiave Primaria | ObjectId fornito da mongodb stesso |

Figura 3: Confronto della terminologia RDBMS con MongoDB

### Tecnica di Modellazione di un Documento

MongoDB non permette alcune operazioni fondamentali tipiche dei database relazionali, quali join tra collection. La mancanza di controlli di integrit`a referenziale tra i dati obbliga il progettista a modellare lo schema logico con modalit`a molto diverse da quelle messe in atto durante lo sviluppo di soluzioni SQL-based.

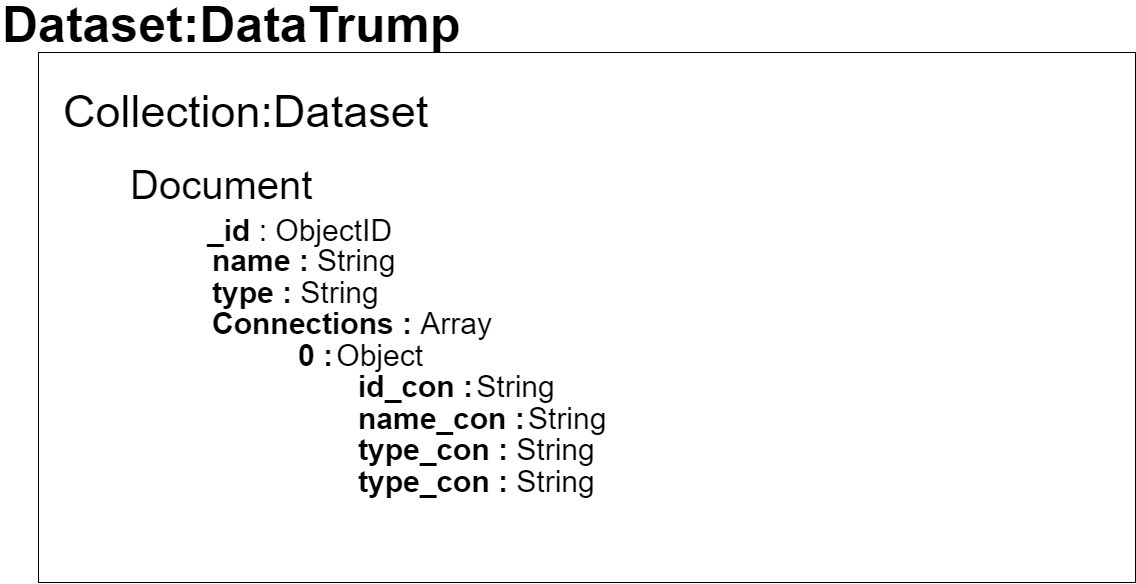


Figura 4: Modello documentale generalizzato

Il documento in esame permette di mostrare la soluzione implementativa utilizzata per gestire la mancanza di relazioni. Innanzitutto all’interno di ogni documento `e presente:

1. il campo speciale **ObjectID** (“ id”) per l’identificazione univoca del documento
2. il campo *“name”* contenente il nome dell’entit`a
3. il campo *“type”* che pu`o assumere il valore di Organization o Person
4. il campo *“Connections”* di tipo array contentente documenti annidati per descrivere le associazioni.
   1. il sotto-campo *‘id con‘* contenente gli idObject delle entit`a colle- gate
   2. il sotto-campo *‘name con‘* contenente i nomi delle entit`a collegate d.3)il sotto-campo *‘type con‘* contenente il tipo di legame

d.4)il sotto-campo *‘source con‘* contenente i siti web delle entit`a col- legate

Questo approccio mantiene tutti i dati relativi in ogni singolo documento, il che rende piu` facile il recupero e la manutenzione. Lo svantaggio `e che se il documento incorporato continua a crescere troppo di dimensioni, pu`o influire sulle prestazioni di lettura / scrittura.

# Implementazione

Completato lo step progettuale si passa alla fase implementativa. Il primo aspetto fondamentale `e effettuare una corretta importazione dei dati per entrambi i DB.

## Importazione in Neo4j

Come gi`a detto nell’introduzione, Neo4j supporta come linguaggio dichiara- tivo ufficiale **Cypher**, creato dal team di Neo4j allo scopo di interagire con il database in modo molto semplice, `e possibile infatti scrivere le interroga- zioni direttamente da riga di comando dell’interfaccia di Neo4j. Di seguito viene proposto il codice utilizzato per l’importazione dei dati dal file CSV a disposizione.

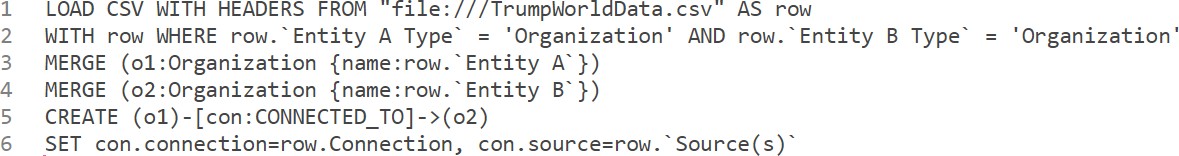


Figura 5: Importazione delle organizzazioni e le loro connessioni

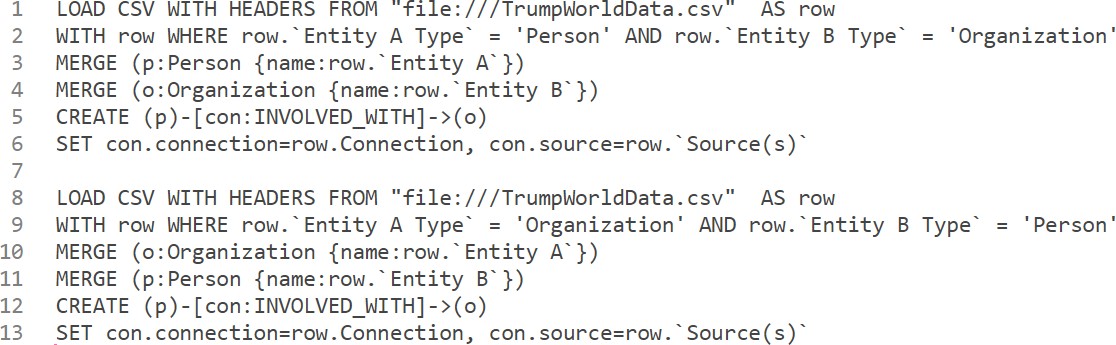


Figura 6: Importazione delle organizzazioni e delle persone connesse

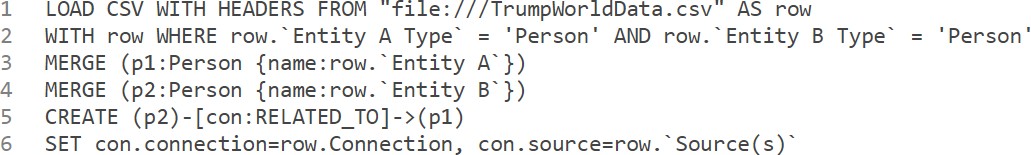


Figura 7: Importazione delle organizzazioni e delle persone connesse

## Importazione in MongoDB

Sebbene MongoDB disponga di una propria Shell per l’invio di interrogazioni e comandi e per restituire i risultati, per i piu` importanti e utilizzati linguaggi di programmazione sono state realizzate delle librerie (**Driver**, come si dice nell’ambito della gestione di database) per l’interfacciamento con MongoDB in modo semplice e intuitivo. Per gli script di importazione `e stato scelto *l’API PHP* mentre per la scrittura delle query *l’API Python*. Per cercare di ottimizzare al meglio i dati sono stati creati 4 script relativi ad ogni step:

1. *“manipulation dataset.php”*, dal file CSV originale vengono creati 3 nuovi sotto-dataset (*“CONNECT TO.php”*, *“INVOLVED WITH.php”*, *“RELATED TO.php”*) secondo il modello descritto nello schema con- cettuale.
2. *“import data.php”*, permette di inserire i dati dei 3 dataset nella col- lection di MongoDB, per ogni documento viene generatore il proprio ObjectID.



1. *“recuperoIDMONGO.php”*, recupera gli ObjectID da i documenti di MongoDB generati e li salva in un file CSV (*recuperaID*) con i nomi associati.
2. *“updateIDMONGO.php”*, permette di individuare le connessioni tra le entit`a a causa della mancanza del costrutto join in MongoDB.
3. *appendConnection.php*, per inserire nel campo Connections tutte le entit`a che sono associate



## Implementazione Query

Una volta inseriti i dati vengono eseguite per entrambi i DB 7 query. Per ogni interrogazione viene riportato l’intero codice scritto in **Cypher** (immagine su) e in **MongoDBShell** (immagine giu`).

### Query 1

Visualizzazione del nome di 10 nodi random.





### Query 2

Ricerca le Organizzazioni che hanno interagito maggiormente e restituisce in ordine decrescente il conto totale delle connessioni stabilite con i relativi collegamenti:

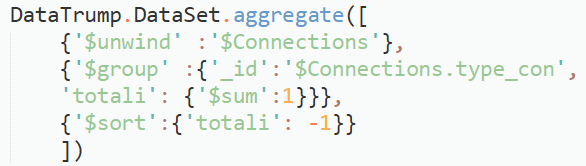




### Query 3

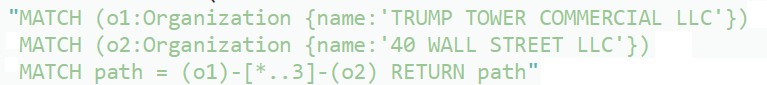
Restituisce il numero di volte che una connessione si presenta nel dataset

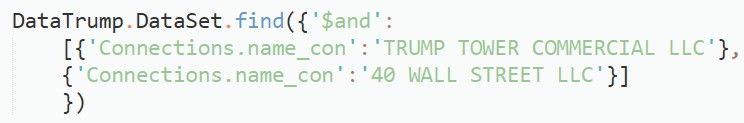




### Query 4

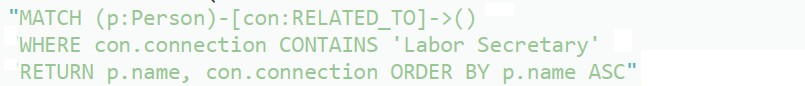
Rispetto alle organizzazioni Trump Tower Commercial LLC e 40 Wall Street LLC che sono quelle piu` popolari, quali organizzazioni sono comuni ad en- trambe





### Query 5

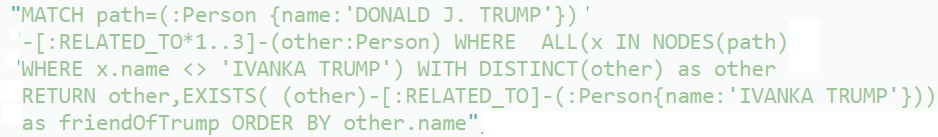
Chi ha nominato Trump come capo di Gabinetto

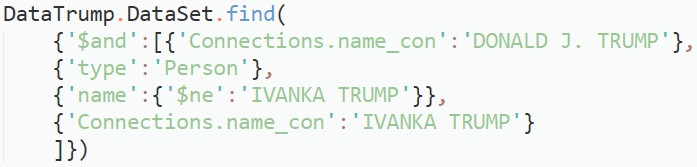




### Query 6

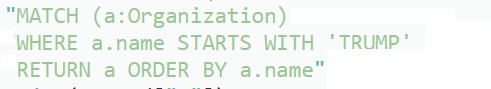
Mostra le persone che stanno nelle cerchie di DONALD J. TRUMP scar- tando quei percorsi che passano per IVANKA TRUMP (figlia), prendendo le persone distinte, e restituisce la persona trovata e un valore booleano che indica se `e anche amico di IVANKA TRUMP. Tutto in ordine alfabetico.





### Query 7

Ritorna tutte le organizzazioni dove i rispettivi nomi iniziano per TRUMP





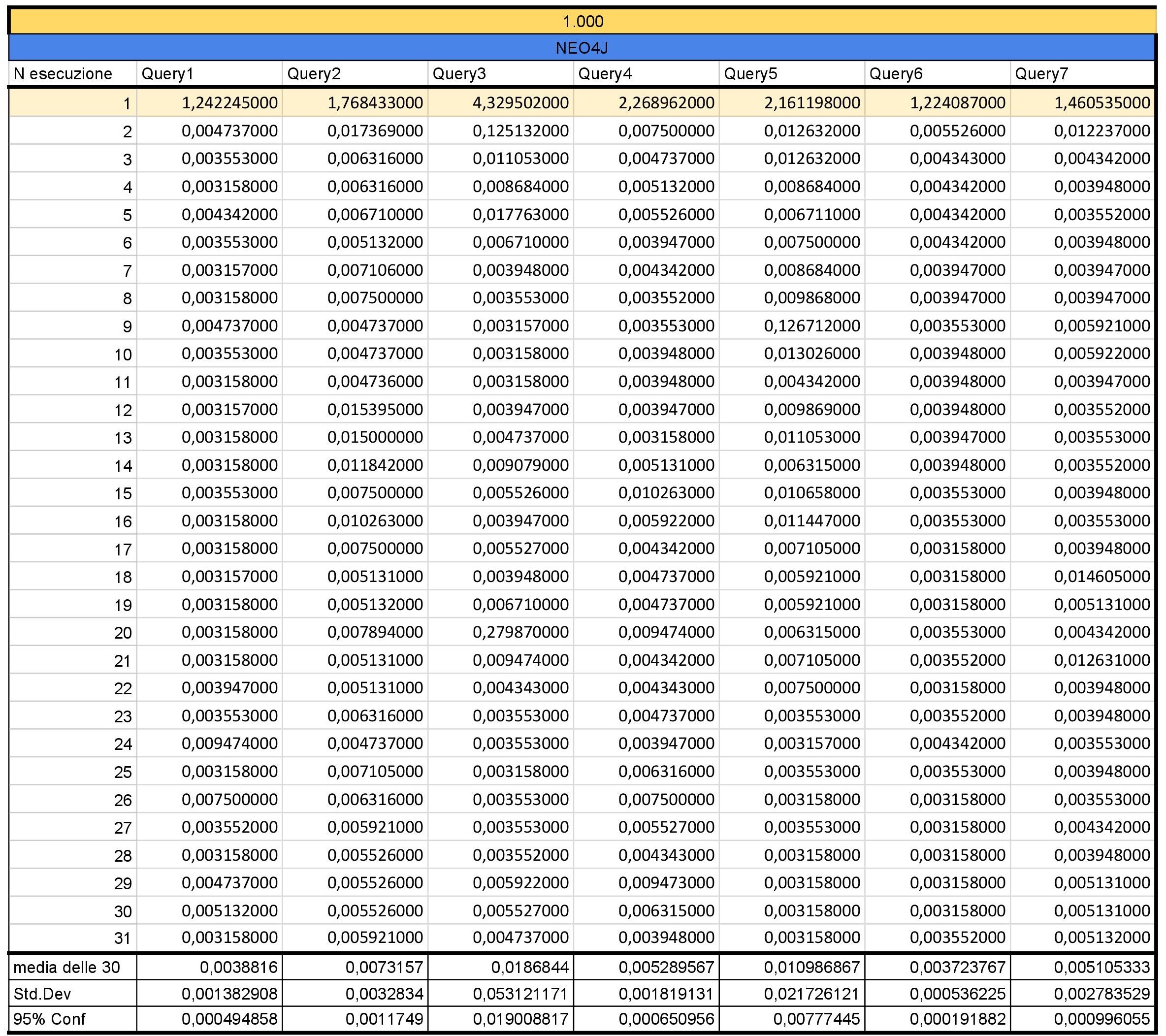
## Generatore Dati

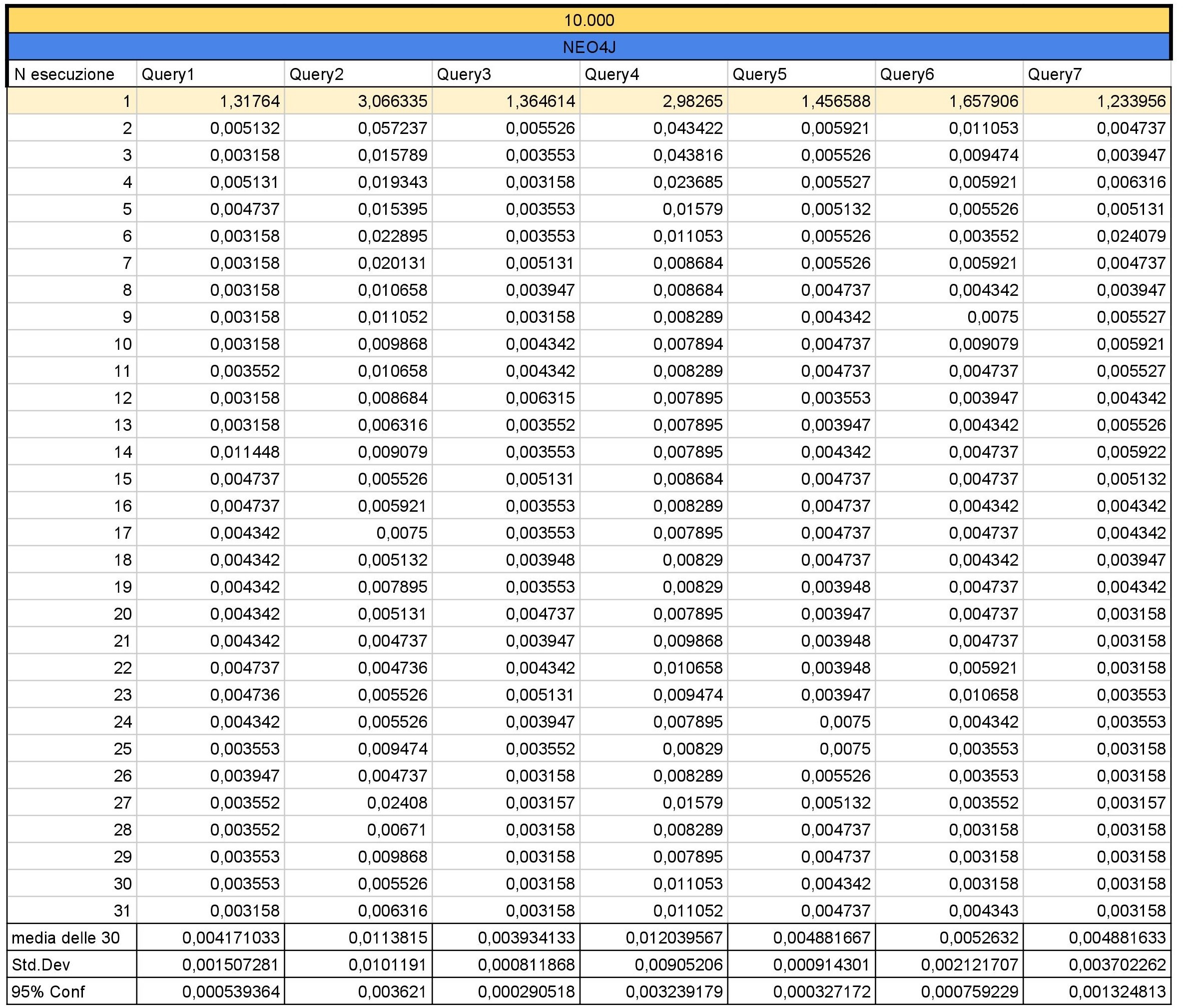
Nella fase successiva quella di testing, per verificare la maggiore efficienza di una delle due soluzione di DataBase, verranno calcolati i tempi di risposta delle query su dimensioni di dataset differenti. Il file CSV `e composto da circa 3000 record, per ottenere set di dati di grandezze maggiori `e stato im- plementato un vero e proprio generatore di dati fittizzi. L’idea `e stata quella di cercare online nomi e cognomi comuni americani da attribuire all’entit`a persona e aziende americane per le organization riportate come elenco in un semplice file testuale. Tramite delle funzioni casuali sono stati creati nuovi record che riprendono esattamente il modello del CSV originale.

# Esperimenti

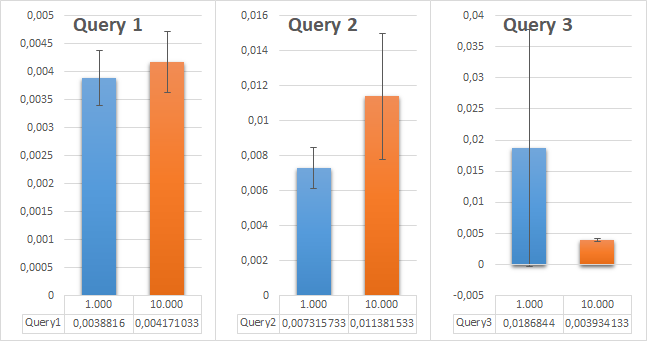
Una volta generati i diversi dataset e importanti nei DB non resta che lanciare le query e farsi restiture i tempi di esecuzione. A causa di meccanismi di chaching, il tempo di esecuzione della prima query sar`a sempre maggiore dei tempi delle seguenti query se non viene variato l’input. Per questo motivo ogni interrogazione verr`a ripetuta 31 volte e verr`a considerato il valore medio con intervalli di confidenza al 95%.

## Risultati Neo4j - Dataset di 1000 e 10000

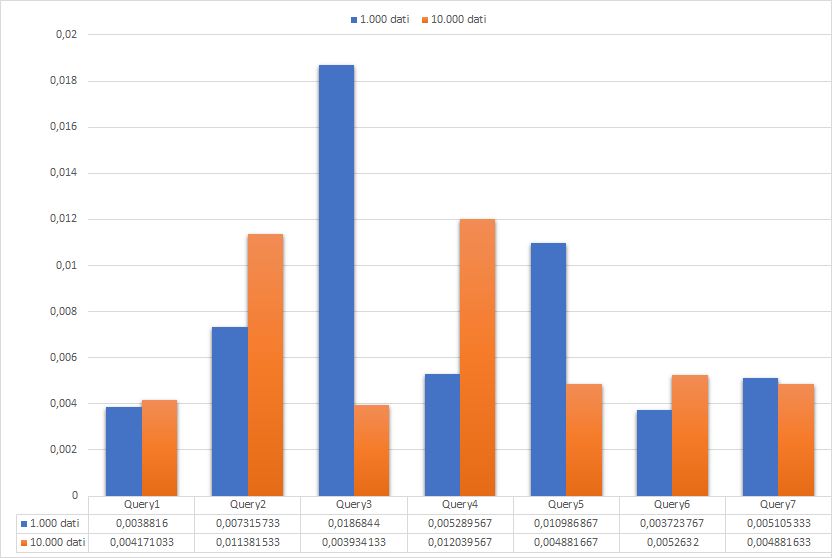




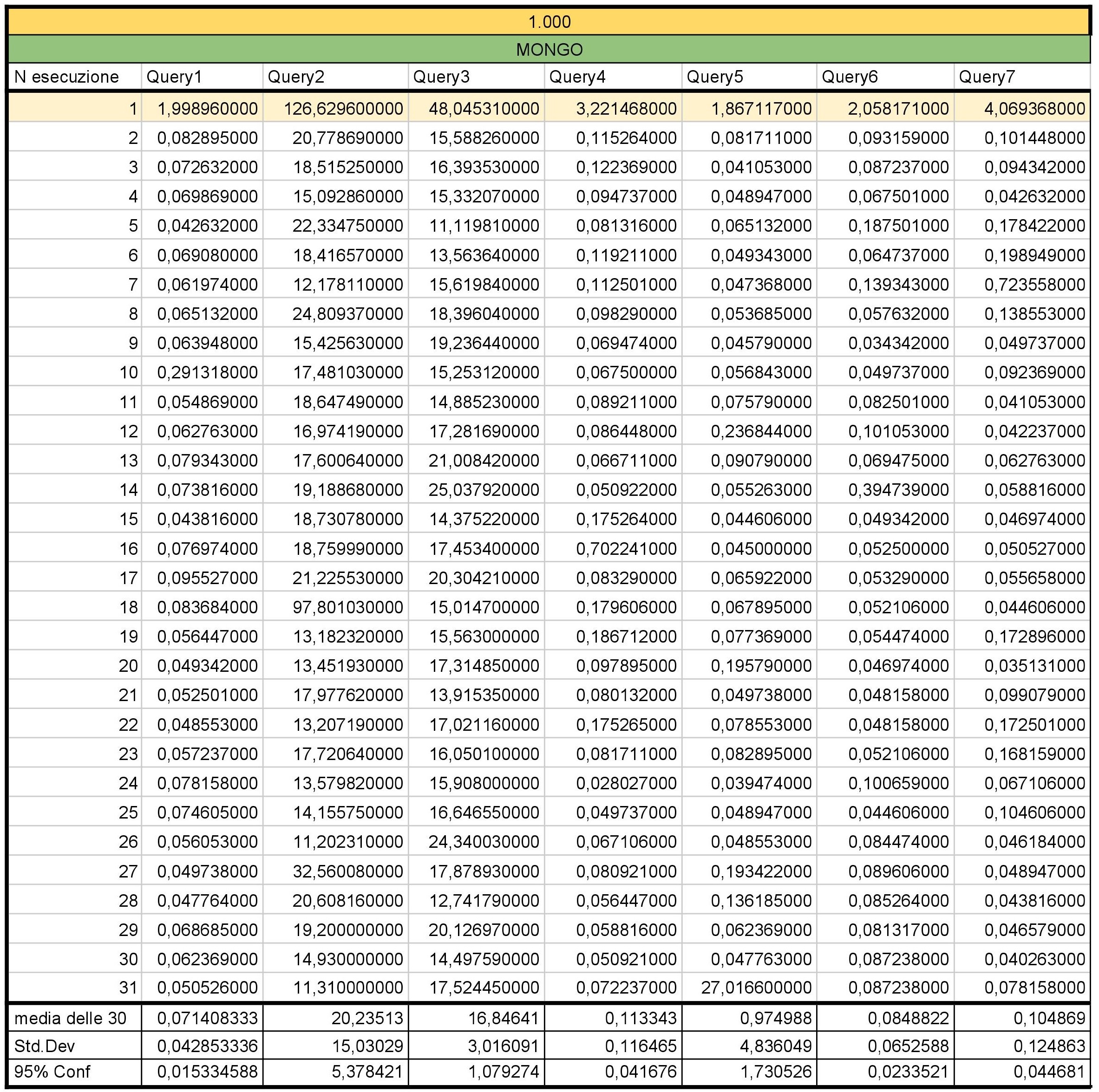
## Confronto dei risultati



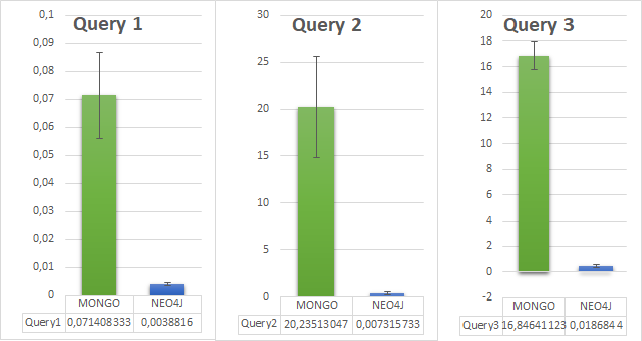


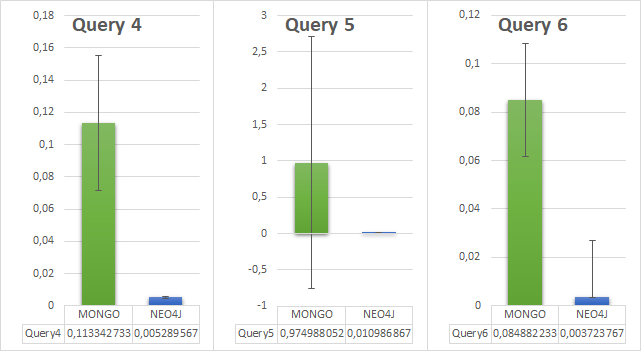


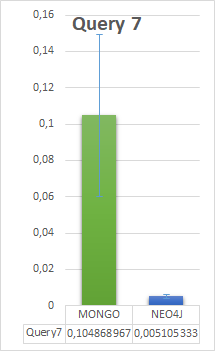
## Risultati MongoDB - Dataset di 1000 e 10000

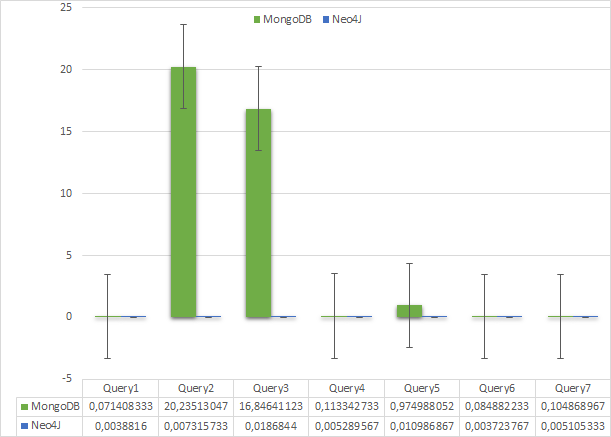


## Confronto Dataset da 1000 per i due DB









# Conclusioni

Questo progetto ha cercato di dimostare la potenza del DB Neo4j nei casi in cui le relazioni tra i dati sono l’informazione piu` importante, per accentuare

ancora di piu` questo aspetto, sono stati messi a confronto i risultati delle

stesse query con un altro DB che invece non implementa affato il concetto di associazione tra dati, MongoDB. A tal fine, `e stata condotta un’accurata indagine attraversol ’utilizzo di due dataset con grandezze diverse: 1000 e 10000 dati. Come ci aspettavamo, al crescere della grandezza del dataset, alcune query realizzate con neo4j hanno dato tempi di risposta ancora piu` bassi.