DB Benchmarks su DBMS PostGreSQL

Contenuto

[Introduzione 1](#_Toc463980895)

[Struttura del progetto 1](#_Toc463980896)

[Installazione 4](#_Toc463980897)

[STRUMENTI DI TEST 4](#_Toc463980898)

[Struttura del progetto MAVEN 5](#_Toc463980899)

[Sorgenti 6](#_Toc463980900)

[Casi di test 6](#_Toc463980901)

[Possibili evoluzioni 9](#_Toc463980902)

# Introduzione

E’ stata realizzata un’applicazione JAVA che esegue dei test benchmark su un DB JDBC compliant, nel caso specifico POSTGRESQL 9.4. L’applicazione si connette via JDBC al Database, crea una tabella di esempio chiamata TEST\_TABLE ed esegue operazioni di UPDATE massivi e select per PK monitorando il tempo minimo, massimo e medio per ogni statement. Ad ogni lancio dell’applicazione la tabella viene droppata e ricreata.

# Struttura del progetto

E’ stato creato un package **it.test.jdbcPostgreSQL** contenente le classi **App** (che contiene il main dell’applicazione) e **DBOperation**.

La classe **DBOperation** contiene i metodi utilizzati per i benchmark di INSERT e SELECT su un DB JDBC compliant (nel nostro caso POSTGRESQL 9.4 installato localmente sulla macchina nella quale sono stati eseguiti i test)

I metodi pubblici esposti dalla classe sono

|  |  |
| --- | --- |
| **Nome\_metodo** | **descrizione** |
| ***createTable*** | Esegue la creazione di una tabella di DB sulla quali eseguire i benchmark |
| ***DropTable*** | Utilizzato per droppare la tabella e ricrearla ad ogni lancio dell'applicazione. |
| ***InsertIntoTable*** | Utilizzato per eseguire i benchmark degli statement di INSERT con commit ogni X statement. Update sequenziali e a blocchi. |
| ***SelectFromTableTime*** | Utilizzato per eseguire i benchmark degli statement  di SELECT per PK della tabella creata in precedenza. |
| ***CloseConn*** | Utilizzato per chiudere la connessione al DB |

La classe **App** contiene il main dell'applicazione ed esegue i benchmark di INSERT e SELECT su un DB PostGreSQL via JDBC.

L'applicazione esegue i seguenti step:

1) carica da un file di properties (***config.properties***) gli estremi di connessione al DB e i parametri per il benchmark. In particolare vengono recuperati parametri per la costruzione dell'URL di connessione via JDBC, ovvero

a) Il nome del DB

b) Lo User di default per la connessione con privilegi di admin sul DB

c) la pwd di connessione al DB

d) l'hostname del DBserver

e) il prefisso dell'URL di connessione (es. jdbc:postgresql://)

f) il nome del class driver JDBC per connettersi al DB (es. org.postgresql.Driver)

g) Numero massimo di statement di INSERT per ogni COMMIT

h) Numero massimo di righe da inserire in tabella durante il test

I parametri a), b) e c) possono essere forniti anche dalla linea di comando nel momento del lancio dell'applicazione seguendo il suddetto ordine. Nel caso non vengano specificati da linea di comando, i relativi valori di default vengono recuperati dal file di properties (***config.properties***)

2) Viene effettuata la connessione al DB utilizzando i parametri recuperati al punto precedente.

3) Se la connessione avviene con successo, viene istanziato un oggetto della classe **DBOperation** per eseguire le seguenti operazioni di benchmark

3.1) Invocazione del metodo ***createTable*** dell'oggetto per creare la tabella di test sul DB chiamata TEST\_TABLE. Se la tabella esiste già viene droppata per ricominciare un nuovo test da zero. La tabella creata come esempio ha 4 colonne

- ID (PK di tabella) di tipo INTEGER

- SURNAME VARCHAR(255)

- NAME VARCHAR2(255)

- CREATED\_DATE DATE

3.2) Se la creazione della tabella di test (TEST\_TABLE) è avvenuta con successo al punto precedente, viene invocato il metodo ***InsertIntoTable*** dell'oggetto della classe ***DBOperation*** istanziato in precedenza. Il metodo viene invocato passando in input

a) la connessione creata (oggetto di tipo Connection)

b) il nome della tabella sulla quale effettuare i benchmark (TEST\_TABLE nel nostro caso)

c) il numero di ***insert statement*** prima di ogni commit

d) il numero massimo di righe da inserire in tabella

e) un parametro boolean (***abSequentialUpdate***) che indica se effettuare le INSERT in modalità sequenziale, ovvero una per ogni iterazione del ciclo di INSERT STATEMENT; oppure se effettuare le insert a blocchi in modalità batch, con commit ogni X righe.

Il metodo stampa a video i tempi in nanosecondi e msec (minimo, medio e massimo) delle due modalità sopra elencate. Vengono effettuati i commit su DB ogni X statement. Nel caso della modalità di INSERT a blocchi di statement (modalità batch) il tempo mimino, medio e massimo sono calcolati per ogni esecuzione del batch di inserimento a pacchetti.

3.3) Se la creazione della tabella TEST\_TABLE è avvenuta con successo al punto 3.1) viene invocato il metodo ***SelectFromTableTime*** dell'oggetto della classe ***DBOperation*** precedentemente istanziato. Il metodo prevede in input

a) la connessione creata (oggetto di tipo Connection)

b) il nome della tabella sulla quale effettuare i benchmark (TEST\_TABLE nel nostro caso)

c) il numero di select consecutive per PK da eseguire (nel nostro caso pari al numero delle righe inserite in tabella al punto 3.2)

Il metodo stampa a video i tempi in nanosecondi e msec (minimo, medio e massimo) delle select eseguite su DB.

4) Se esiste una connessione valida creata al punto 1) viene chiusa.

**Strumenti utilizzati:**

DBMS PostGreSQL 9.4

Java JDK 8

Maven per la creazione del progetto

# Installazione

E’ stato creato un progetto MAVEN contenente la struttura dell’applicazione. E’ stato creato il package **it.test.jdbcPostgreSQL** contenente la main class **App**. Nella directory ***src/main/resources*** del progetto è stato inserito il file di configurazione **config.properties** che contiene i parametri di setup dell’applicazione, in particolare

***jdbcURL***=jdbc:postgresql:// 🡪 prefisso URL di connessione via ODBC al DB PostgreSQL

***dbpassword***=nadav200 🡪 pwd di connessione al DB

***database***=postgres 🡪 Nome del DB installato sul DB server

***dbuser***=postgres 🡪 User per la connessione al DB con privilegi di ADMIN

***host***=127.0.0.1:5432 🡪 host del DB server (nel caso in oggetto il DB è stato installato localmente)

***ClassDriverJDBC*** = org.postgresql.Driver (class driver JDBC)

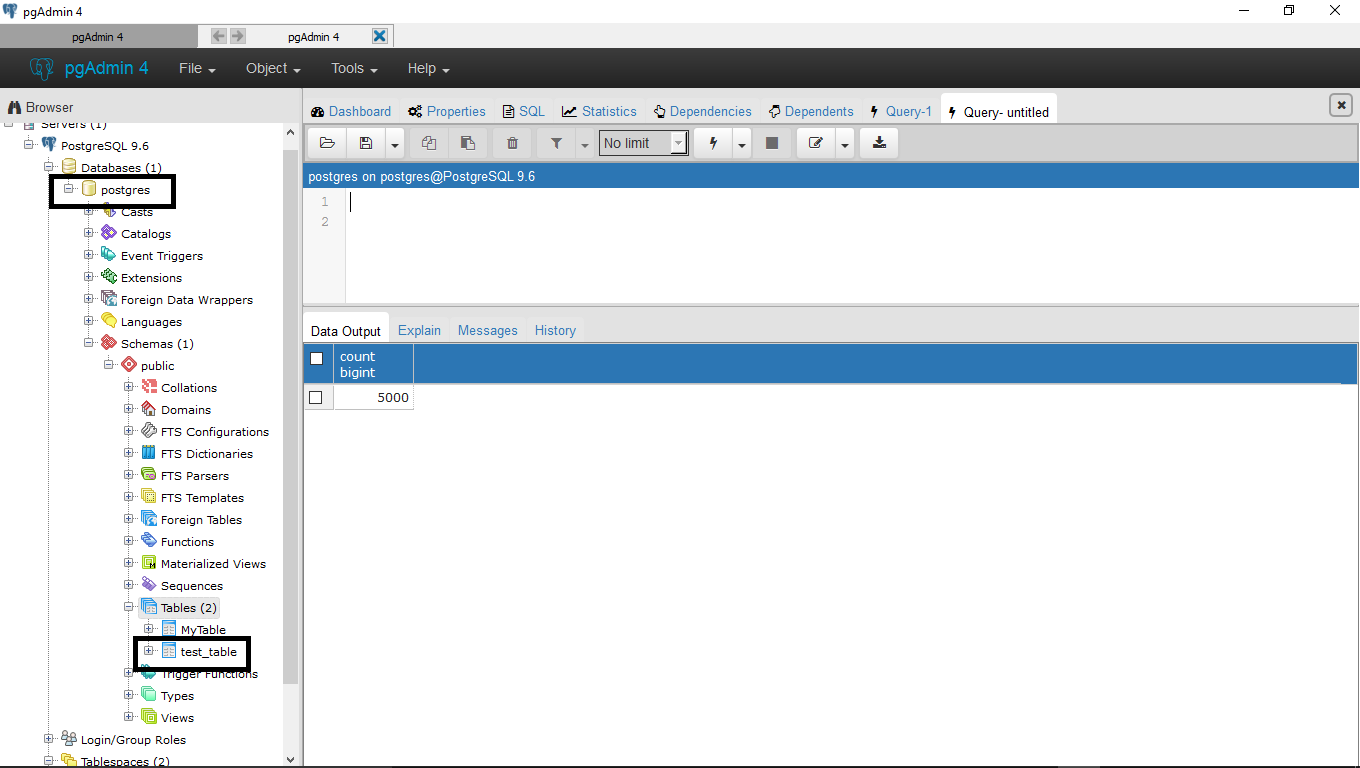
***maxRowsPerCommit*** = 5000 (numero massimo di statement per ogni commit; es. commit ogni 5000 righe inserite)

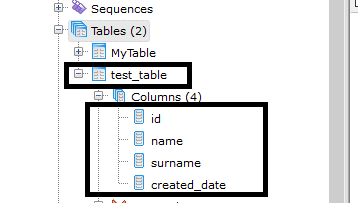
***maxRowsInserted***= 1000000 (numero massimo di righe da inserire per il test, il numero massimo consentito è stato limitato a 2000000 di righe.)

Nella cartella ***src/main/resources/JDBC*** è stato inserito il jar **postgresql-9.4.1211.jar** relativo al driver JDBC di POSTGRESQL. Il jar va copiato in una cartella compresa nel classpath dell’applicazione. Nel caso della macchina con Windows 10 utilizzata per i test è stato copiato nella cartella **C:\Program Files\Java\jre1.8.0\_102\lib\ext** (dove sono presenti anche i file del JDK 8).

# STRUMENTI DI TEST

Una volta installato PostGreSQL 9.4 sulla macchina utilizzata per i test, è stato creato il DB name **postgres.** Si è utilizzato il browser pgAdmin 4 per visualizzare la struttura del DB e controllare la tabella creata dall’applicazione

****

****

Per lanciare la compilazione e l’esecuzione dell’applicazione si può utilizzare la seguente riga di comando a partire dalla directory principale di installazione (ovvero quella che contiene il file della struttura del progetto POM.XML)

java -cp target/jdbcPostgreSQL-1.0-SNAPSHOT.jar it.test.jdbcPostgreSQL.App

# Struttura del progetto MAVEN

**/jdbcPostgreSQL**

**src/main/java/it/test/** **jdbcPostgreSQL/App.java)** (main class)

**target/** **jdbcPostgreSQL-1.0-SNAPSHOT.jar** (versione compilata dell’applicazione)

**pom.xml** (contiene le informazioni del progetto)

**src/main/resources/config.propertie**s (file di configurazione dell’applicazione)

**src/main/resources/JDBC/postgresql-9.4.1211.jar** (jar driver JDBC PostgreSQL)

**doc**/**DB Benchmarks su DBMS PostGreSQL.docx (**questo documento**)**

**html/index-all.html (java doc)**

java -cp target/jdbcPostgreSQL-1.0-SNAPSHOT.jar **it.test.jdbcPostgreSQL.App**

# Sorgenti

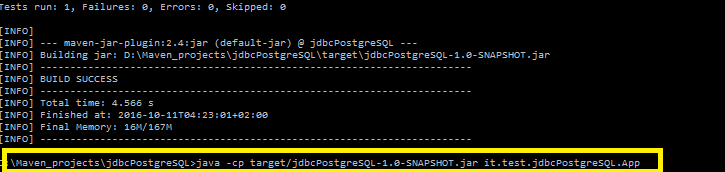
In allegato il codice sorgente JAVA, il main è contenuto nella classe **App** e le operazioni di test sul DB vengono esposte nella classe **DBOperation.**



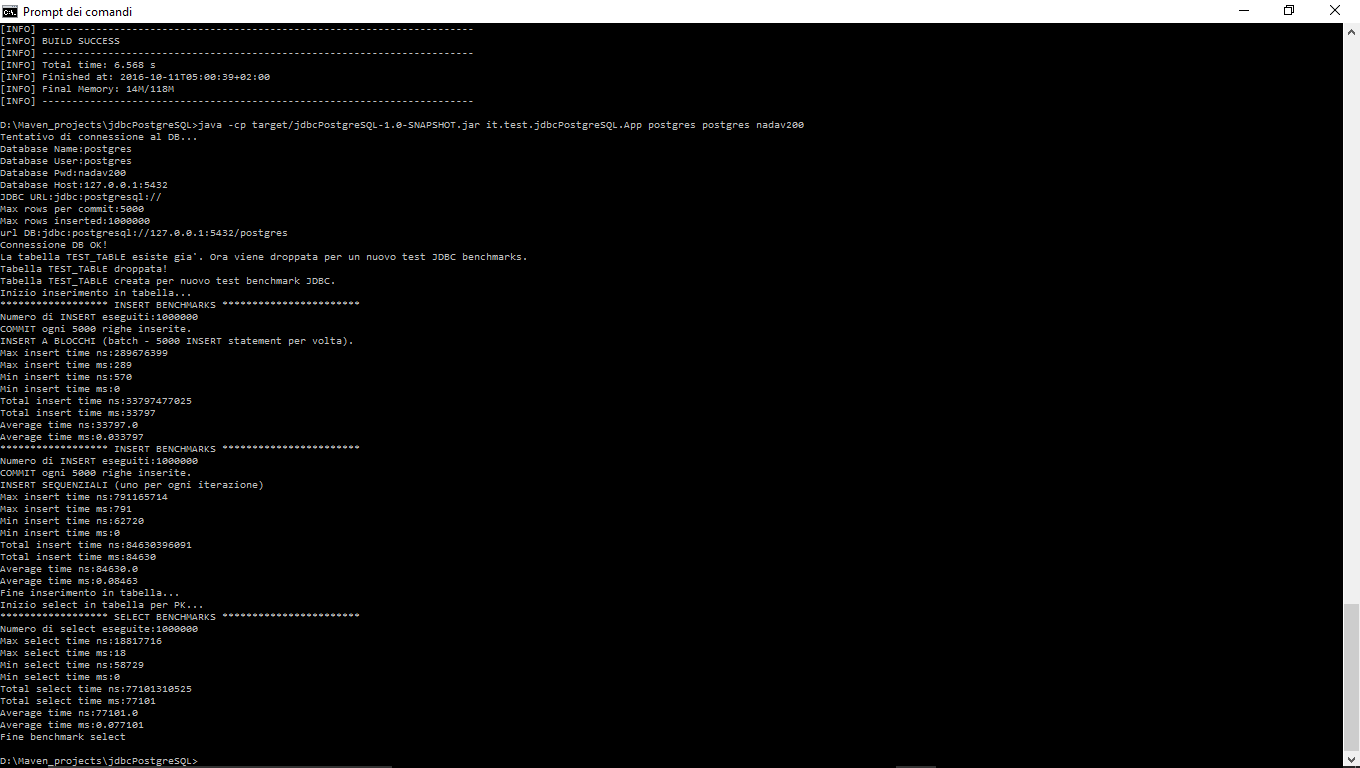
# Casi di test

1. Lancio applicazione senza parametri nella riga di comando. In questo caso i parametri di connessione **database, dbUser e dbPassword** vengono recuperati dal file di configurazione config.properties.





Con impostazioni che prevedono 1000000 di righe inserite con commit ogni 5000 righe e 1000000 di select per PK l’output dell’applicazione sulla macchina di test utilizzata è riportato di seguito



D:\Maven\_projects\jdbcPostgreSQL>java -cp target/jdbcPostgreSQL-1.0-SNAPSHOT.jar it.test.jdbcPostgreSQL.App

Tentativo di connessione al DB...

Database Name:postgres

Database User:postgres

Database Pwd:nadav200

Database Host:127.0.0.1:5432

JDBC URL:jdbc:postgresql://

Max rows per commit:5000

Max rows inserted:1000000

url DB:jdbc:postgresql://127.0.0.1:5432/postgres

Connessione DB OK!

La tabella TEST\_TABLE esiste gia'. Ora viene droppata per un nuovo test JDBC benchmarks.

Tabella TEST\_TABLE droppata!

Tabella TEST\_TABLE creata per nuovo test benchmark JDBC.

Inizio inserimento in tabella...

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* INSERT BENCHMARKS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Numero di INSERT eseguiti:1000000

COMMIT ogni 5000 righe inserite.

INSERT A BLOCCHI (batch - 5000 INSERT statement per volta).

Max insert time ns:289676399

Max insert time ms:289

Min insert time ns:570

Min insert time ms:0

Total insert time ns:33797477025

Total insert time ms:33797

Average time ns:33797.0

Average time ms:0.033797

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* INSERT BENCHMARKS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Numero di INSERT eseguiti:1000000

COMMIT ogni 5000 righe inserite.

INSERT SEQUENZIALI (uno per ogni iterazione)

Max insert time ns:791165714

Max insert time ms:791

Min insert time ns:62720

Min insert time ms:0

Total insert time ns:84630396091

Total insert time ms:84630

Average time ns:84630.0

Average time ms:0.08463

Fine inserimento in tabella...

Inizio select in tabella per PK...

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* SELECT BENCHMARKS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Numero di select eseguite:1000000

Max select time ns:18817716

Max select time ms:18

Min select time ns:58729

Min select time ms:0

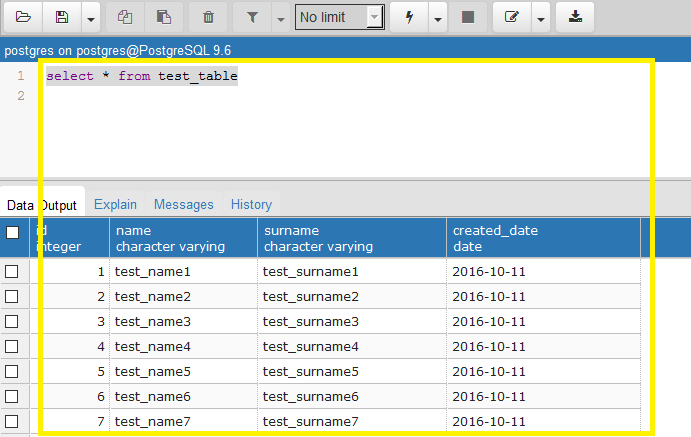
Total select time ns:77101310525

Total select time ms:77101

Average time ns:77101.0

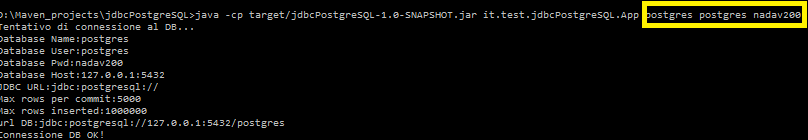
Average time ms:0.077101

Fine benchmark select



L’applicazione prevede 3 parametri facoltativi per il lancio dalla linea di comando (dbName, DbUser, dbPassword). In mancanza di questi parametri vengono recuperati i default specificati nel file di configurazione **src/main/resources/config.propertie**s.

Esempio di lancio con i 3 parametri



# Possibili evoluzioni

Per rendere l’applicazione general purpose e quindi non legata alla costruzione di una tabella di test con una struttura fissa, si può pensare di portare fuori gli statement di creazione, select e insert e caricarli da file.