stratebi
open business intelligence

VoltDB – Up & Running

VoltDB

Índice

- Introducción
  - ¿Qué es?
  - Conceptos clave
- Instalación
- Ejemplo – Hola Mundo
- Procedimientos Almacenados

¿Qué es?

- Base de datos en memoria y distribuida.
- No hay Base de datos como tal, hay catálogos que contienen tablas en memoria.
- Estos catálogos se replican a través de los nodos, donde el nodo maestro mantiene el tracking de los datos.

Mantenimiento de Nodos

- VoltDB ha pensado en el caso de que se den caídas de los nodos. En caso de caída, otro nodo mantiene la consistencia de datos.
- Snapshots: en caso de querer guardar el sistema, se pueden crear Snapshots que pueden ser restaurados con posterioridad, para evitar tener que hacer un nuevo catálogo.

Tipos de Distribución de Nodos

- Replicación: nos permite "replicar" las tablas a través de los nodos, para acceder más rápidamente a los datos.
- Partición: dado una clave primaria de una tabla, esta es divida a través de los nodos teniendo en cuenta esa clave. La clave debe ser un solo campo de la tabla.

Requisitos Previos

- Ubuntu 64 Bits
- RAM: 1GB

Instalación en Single-Node

- Paso 1. Descargar la versión debian:

$wget http://voltdb.com/downloads/technologies/server/voltdb_3.4-1_amd64.deb

- Paso 2. Instalar VoltBD:

$ sudo dpkg –i voltdb_3.4-1_amd64.db

Preparación previa

- Crear directorio y cambiarnos a el:

```
$ mkdir helloworld
$ cd helloworld
```

El esquema del catálogo

- Crear esquema BBDD “helloworld.sql”:

```
CREATE TABLE HELLOWORLD(
    HELLO VARCHAR(15),
    WORLD VARCHAR(15),
    DIALECT VARCHAR(15) NOT NULL,
    PRIMARY KEY (DIALECT)
);
PARTITION TABLE HELLOWORLD ON COLUMN DIALECT;
```

Compilar esquema y operar:

- Compilar esquema:

$ voltdb compile –o helloworld.jar helloworld.sql

- Creación del catálogo:

$ voltdb créate catalgo helloworld.jar

- Operaciones con el catálogo:

$ INSERT INTO helloworld VALUES (‘hello’,’world’,’English’);

$ SELECT * FROM helloworld WHERE dialect=’English’;

VoltDB utiliza JAVA para comunicarse

- Los procedimientos almacenados están escritos en Java y nos sirven para comunicarnos con VoltDB, ya sea para leer o insertar datos en las tablas.

- Añadir las siguientes líneas al "helloworld.sql":

```
CREATE PROCEDURE FROM CLASS insert;
CREATE PROCEDURE FROM CLASS select;
PARTITION PROCEDURE insert ON TABLE HELLOWORLD
    COLUMN DIALECT;
PARTITION PROCEDURE select ON TABLE HELLOWORLD
    COLUMN DIALECT;
```

Procedimientos almacenados que usaremos

- Vamos a ver dos ejemplos sencillos de inserción y selección a través de los procedimientos almacenados. Simplemente crearemos los archivo .java siguientes:
  - insert.java
  - select.java
  - client.java

Insert.java

```
import org.voltdb.*;


public class insert extends VoltProcedure{
public final SQLStmt sql = new SQLStmt(
        "INSERT INTO HELLOWORLD VALUES (?,?,?);"
);
public VoltTable[] run( String languaje,
                        String hello,
                        String world)
    throws VoltAbortException{
        voltQueueSQL(sql, hello, world, languaje);
        voltExecuteSQL();
        return null;
    }
}
```

Select.java

```
import org.voltdb.*;


public class select extends VoltProcedure{
public final SQLStmt sql = new SQLStmt(
        "SELECT HELLO, WORLD FROM HELLOWORLD " +
        " WHERE DIALECT = ?;"
);
public VoltTable[] run( String languaje)
    throws VoltAbortException{
        voltQueueSQL(sql, languaje);
        return voltExecuteSQL();
    }
}
```

Client.java

```
import org.voltdb.*;
import org.voltdb.client.*;

public class Client {

    /**
     * @param args
     * @throws Exception
     * @throws
     */
    public static void main(String[] args) throws Exception {
        // TODO Auto-generated method stub
        org.voltdb.client.Client myApp;
        myApp = ClientFactory.createClient();
        myApp.createConnection("localhost");
```

Client.java - continuación

```
myApp.callProcedure("insert", "English","Hello","World");
        myApp.callProcedure("insert", "French","Bonjour","Monde");
        myApp.callProcedure("insert", "Spanish","Hola","Mundo");
        myApp.callProcedure("insert", "Danish","Hej","Verden");
        myApp.callProcedure("insert", "Italian","Ciao","Mondo");


        final ClientResponse response = myApp.callProcedure("select", "Spanish");

if(response.getStatus()!=ClientResponse.SUCCESS){
    System.err.println(response.getStatusString());
    System.exit(-1);
}
```

Client.java - continuación

```
final VoltTable results[]=response.getResults();
        if(results.length==0||results[0].getRowCount()!=1){
            System.out.printf("I can't say Hello in that languaje.\n");
            System.exit(-1);
        }

        VoltTable resultTable=results[0];
        VoltTableRow row = resultTable.fetchRow(0);
        System.out.printf("%s, %s!\n", row.getString("hello"),
                          row.getString("wolrd"));
    }

}
```

Variable de Entorno y generación de .class

- Ahora establecemos la variable de entorno:

```
echo CLASSPATH="./:/home/javier/Escritorio/voltdb-3.4/lib/*:/home/javier/Escritorio/voltdb-3.4/voltdb/*"
```

- Creamos los .class:

```
javac client.java
javac insert.java
javac select.java
```

Compilar el archiv .sql en un jar

voltdb compile --classpath="./" -o helloworld.jar helloworld.sql

Archivo deployment.xml

- Creamos el archivo deployment.xml:

```
<?xml version="1.0"?>
<deployment>
 <cluster hostcount="1"
      sitesperhost="2"
 />
 <httpd enabled="true">
   <jsonapi enabled="true" />
 </httpd>
</deployment>
```

La creación del catálogo

create catalog helloworld.jar deployment deployment.xml host localhost

Ejecución

- En un nuevo terminal, vamos al directorio donde hemos estado operando y ejecutamos:

$ java Client

Salida

- Devolverá algo parecido a lo siguiente:

javier@javier-Parallels-Virtual-Platform:~/helloworld$ java Client

log4j:WARN No appenders could be found for logger (HOST).

log4j:WARN Please initialize the log4j system properly.

log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.

Hola, Mundo!

Preguntas

stratebi
open business intelligence