RogueWave SOFTWARE

Accelerating Great Code

USING CASSANDRA WITH SOURCEPRO DB

INTRODUCTION TO CASSANDRA

Apache Cassandra™ is a distributed database designed to manage large amounts of structured and unstructured data across many servers. Originally created at Facebook and put into open source in 2009, Cassandra's distribution design is based on Amazon's Dynamo and its data model on Google's Bigtable. Unlike relational databases, Cassandra can store structured, unstructured, and semi-structured data. Also, unlike SQL, Cassandra does not support operations like SQL Joins; thus data tends to be highly denormalized[1].

The primary container of data in Cassandra is a keyspace, similar to a database in RDBMS. Inside a keyspace are one or more column families, which are like relational tables but provide more flexibility in structure. Column families can have one to many thousands of columns.

USING CASSANDRA WITH SOURCEPRO DB

SourcePro DB provides an object-oriented interface that abstracts away the complexity of writing database applications. A SourcePro DB installation consists of the DB Interface Module and one or more DB Access Modules. Each DB Access Module provides access to a specific database API.

For querying Cassandra, use the SourcePro DB ODBC Access Module, which connects to Cassandra using the DataStax ODBC driver.

Install the DataStax ODBC driver

Skip this section if you already have the DataStax ODBC driver installed.

1. Download and install the DataStax ODBC driver: http://www.datastax.com/download-drivers (The driver version used for this example is 1.0).
2. Configure an ODBC DSN for Cassandra. Detailed instructions, including screenshots, are available at: http://www.datastax.com/dev/blog/datastax-odbc-cql-connector-apache-cassandra-datastax-enterprise.

Example on querying Cassandra using SourcePro DB

This example demonstrates using Cassandra with SourcePro DB, and performs the following operations:

- Creates a census column family with the column's name, id, age, address, and zip.
- Inserts data into the table using the RWDBInserter class.
- Queries the inserted data using the RWDBSelector class for names of people with the zip code 97330 and ordered by id.
- Drops the census table.

These operations are performed similarly to the way in which you would query a relational database, with some minor exceptions.

[1] "Introduction to Apache Cassandra", DataStax, 2012.

```
#include <rw/db/db.h>
#include <rw/tools/cstring.h>

// Forward declarations
RWDBSchema createCensusSchema();

void insertIntoCensusTbl(const RWDBSchema& aSchema, const RWDBTable& censusTab, const
RWDBConnection& conn);

void selectFromCensusTbl(const RWDBConnection& conn, const RWDBTable& censusTab)


int main() {
    // Connect to Cassandra using configured DSN
    RWDBDatabase db = RWDBManager::database("ODBC", "<CASSANDRA DSN NAME>", "<USER>",
"<PASSWORD>", "<KEYSPACE>", "");
    RWDBConnection conn = db.connection();
    RWDBSchema  aSchema = createCensusSchema();
    RWCString tabName("test.census");  //1
    db.createTable(tabName, aSchema, conn);
    RWDBTable censusTab = db.table(tabName);
    censusTab.tag(""); //2
    insertIntoCensusTbl(aSchema, censusTab, conn);
    selectFromCensusTbl(conn, censusTab);
    censusTab.drop();
    return 0;
}

RWDBSchema createCensusSchema() {
    RWDBSchema  aSchema;
    RWDBColumn nameColumn, idColumn, ageColumn, addressColumn, zipColumn;
    nameColumn.name("name").type(RWDBValue::String).storageLength(50).nullAllowed(false);
    aSchema.appendColumn(nameColumn);
    idColumn.name("id").type(RWDBValue::UnsignedLong).nullAllowed(false);
    aSchema.appendColumn(idColumn);
    ageColumn.name("age").type(RWDBValue::UnsignedLong);
    aSchema.appendColumn(ageColumn);
    addressColumn.name("address").type(RWDBValue::String).storageLength(50).
nullAllowed(true);
    aSchema.appendColumn(addressColumn);
    zipColumn.name("zip").type(RWDBValue::UnsignedLong);
    aSchema.appendColumn(zipColumn);
    RWDBPrimaryKey pkey;
    pkey.appendColumn(aSchema[4]); //3
    pkey.appendColumn(aSchema[1]); //3
    aSchema.primaryKey(pkey);
    return aSchema;
}
```

```
void insertIntoCensusTbl(const RWDBSchema& aSchema, const RWDBTable& censusTab, const
RWDBConnection& conn) {
    RWDBInserter ins = censusTab.inserter(aSchema); //4
    ins << "Cary Silverstone" << 1001 << 12 << "1200 Market St." << 97330;
    ins.execute(conn);
    ins << "Bob Smith" << 1002 << 15 << "100 Main St." << 97030;
    ins.execute(conn);
    ins << "Alan Turing" << 1003 << 55 << "12 2nd st." << 97330;
    ins.execute(conn);
    ins << "Alex Campbell" << 1004 << 23 << "17 Champa St." << 80330;
    ins.execute(conn);
    ins << "Sandy Doe" << 1005 << 40 << "114 39th St." << 97330;
    ins.execute(conn);
}

void selectFromCensusTbl(const RWDBConnection& conn, const RWDBTable& censusTab) {
    RWDBSelector sel = conn.database().selector();
    sel << censusTab["name"].clearTable(); //5
    sel << censusTab["address"].clearTable();
    sel << censusTab["id"].clearTable();
    sel.from(censusTab);
    sel.where(censusTab["zip"].clearTable() == 97330);
    sel.orderBy(censusTab["id"].clearTable());
    RWDBTable table = sel.execute(conn).table();
    RWDBReader rdr = table.reader(conn);
    RWCString name, address;
    size_t id;
    std::cout << "Ids, names and addresses of people who live in zipcode 97330 ordered by
id:"
              << std::endl << std::endl;
    while (rdr()) {
        rdr >> name >> address >> id;
        std::cout << id << std::endl
                  << name << std::endl
                  << address << std::endl
                  << std::endl;
    }
}
```

Rogue Wave provides software development tools for mission-critical applications. Our trusted solutions address the growing complexity of building great software and accelerates the value gained from code across the enterprise. The Rogue Wave portfolio of complementary, cross-platform tools helps developers quickly build applications for strategic software initiatives. With Rogue Wave, customers improve software quality and ensure code integrity, while shortening development cycle times.