DATABASES AND INFORMATION SYSTEMS
AUTUMN 2017

Project Report

Distributed Database System

Divya Raghunathan	150110086
Madhav Goel	150110017
Vishwajeet Singh	150050046
Varshith S	150050084

November 24, 2017

Contents

1 Introduction

1.1 Motivation

The world is expanding at an unprecedented rate with increasing connectivity and expanding information systems. This calls for rapidly scalable infrastructure without the loss of abstraction to ensure forward compatibility and ease of use.

1.2 Core Idea

We provide a framework that provides an interface to a group of databases running on different servers, under the abstraction of a single database.
With this framework in place, users would be able to scale quickly by adding servers to the group of servers already in place, and be able to use the entire group of databases under the same abstraction with the same interface, without any extra software setup.

2 Implementation and Functionalities supported

We build a horizontally fragmented homogeneous distributed database system and provide an interface to the system under a single-database abstraction.

2.1 Functionality Supported

An operation is distributed among the child sites and the final result is output by the application server.

2.1.1 Database Operations

The following operations are supported to modify and query the database:

- Create, drop, insert, update, delete operations
- Basic select queries involving one relation. Range queries are not supported.
- Basic aggregates: count, sum, avg (involving groupings)
- Joins using foreign tables

2.1.2 Distributed Database Operations

The following operations are supported to setup the distributed database:

- master: Adding a master server to store the metadata information related to database
- add: Adding a postgres server to already existing servers
- del: Deletes the postgres server from the group of servers already present
- execute: Actual database operations are run across a subset of servers
- run: Script file support to run the above commands from a file
- debug: Setting the debug mode off/on

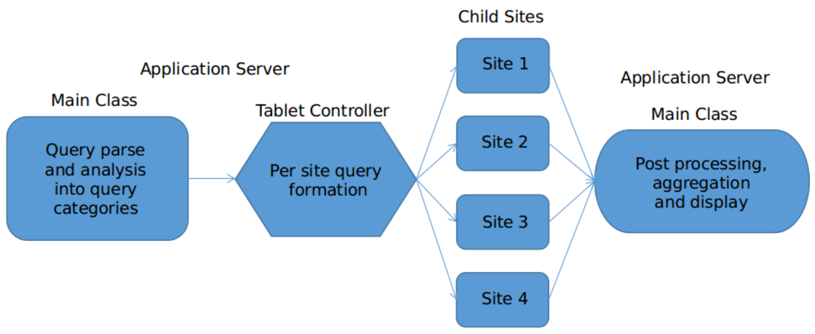

Figure 2.1: Query Processing Overview

2.2 Architecture of the System

Application Server

- The application server receives all database operations through a command line interface.
- It creates threads to execute the operations on the appropriate child sites as given by the tablet controller.
- The results of the query on each site are collected and combined at the application server.
- The application server maintains information about the child sites (host, port, database and username). It also maintains metadata, such as the relations in the schema and the primary key attributes of each relation.

Metadata Server

- The metadata server runs postgres database and stores the metadata related to the actual database.
- Stores the relations created by executing create statement, information related to added servers, tablet mapping and tuple count estimates.

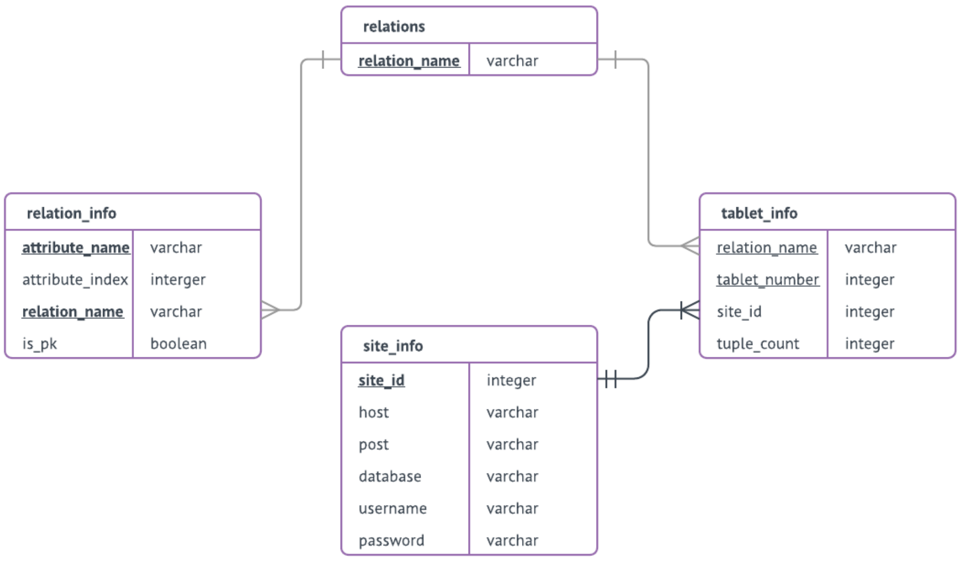

Figure 2.2: Metadata Schema

Child Sites

- The child sites run postgres databases and store the relation tuples.
- Relations are horizontally fragmented across child sites, so that each site has the same schema.
- Tuples of a relation are stored in units of tablets, which are implemented as separate tables with table name <relation_name>_<tablet_id>.
- On receiving an operation, the application server sends the operation to be executed on all child sites or a subset of child sites. Schema operations are executed on all child sites. Certain types of queries are executed on a subset of sites, given by the tablet controller.
- One of the sites is declared the master site. This site stores references to schemas in the other sites for processing join queries.

Tablet Controller

- The tablet controller maintains a tablet id to site mapping for all tablets.
- The number of tablets is taken to be a constant, fixed by the application server.
- The tablet controller determines the tablet into which a particular tuple must be inserted by computing a hash over the primary key attributes and taking modulus over the number of tablets.

- Given a query, the tablet controller determines the subset of sites on which to run the query, and the query which is to be run on each site.

2.3 Query Handling

Create The relation's primary key attributes are extracted and stored in the Tablet Controller as metadata to assist tablet id computation. The tablet_id-to-site mapping is created and stored. The corresponding tablets are created on the respective sites according to the mapping.

Insert The primary key attributes of the tuple are extracted, hashed to find the tablet id and inserted in the tablet on the host site.

Select (without aggregation) Queries involving the primary key attributes are run on the site containing the tablet that tuple would belong to. The query is run on all sites in cases where only one relation is involved.

Select (with aggregation) Aggregates min, max, count, sum are run on all sites and the required post processing done before display. Aggregate avg involves an additional count(*) evaluation, done through addition of a count(*) term in the original query, and the required post processing.

Select (involving joins) One of the sites is chosen to be a master site. This has access to other sites' schema and tables using ftables. When a query that involves joins is issued, the tables are assimilated from tablets and the queries executed on the master site.

** Other queries are routed to all sites.*

3 Future Work

- Efficient joins
- Data load balancing using tablet relocations
- Foreign key constraints
- Decomposition Algorithms

4 Frameworks

python Python is the programming language used for building the interface, data operations and database management.

pg_query pg_query is used to construct the parse tree of a query

psycopg2 psycopg2 is used to connect to child sites from the application server

5 Distribution

Vishwajeet CLI, load balancing utilities, Queries with aggregation, Metadata Management

Divya Inserts, Query Processing, Metadata Management, load balancing utilities

Varshith CLI, Inserts, Query execution, Joins and Foreign Schemas

Madhav Queries with aggregation, Joins and Foreign Schemas

6 References

Programming

- https://pypi.python.org/pypi/psycopg2
- https://pypi.python.org/pypi/pg_query

General

- https://people.eecs.berkeley.edu/ wong/wong_pubs/wong58.pdf
- https://github.com/kmwenja/siafu
- https://www.citusdata.com/blog/2015/02/24/how-to-build-distributed-db/
- https://github.com/citusdata/citus/tree/2e0916e15a10f1a6b4b47fb72f6bb3564f8b3003
- http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.90.9001&rep=rep1&type=pdf