

 README.md

ServerLogs

Project for Post-Graduate class Database Management Systems

Authors

- [Papadopoulos Christos](#)
- [Papamichalopoulos Marios](#)

Tools Used

- Spring Boot
- React
- PostgreSQL 11.6
- IntelliJ IDEA
- Webpack
- Various NPM dependencies (package.json)

How to run

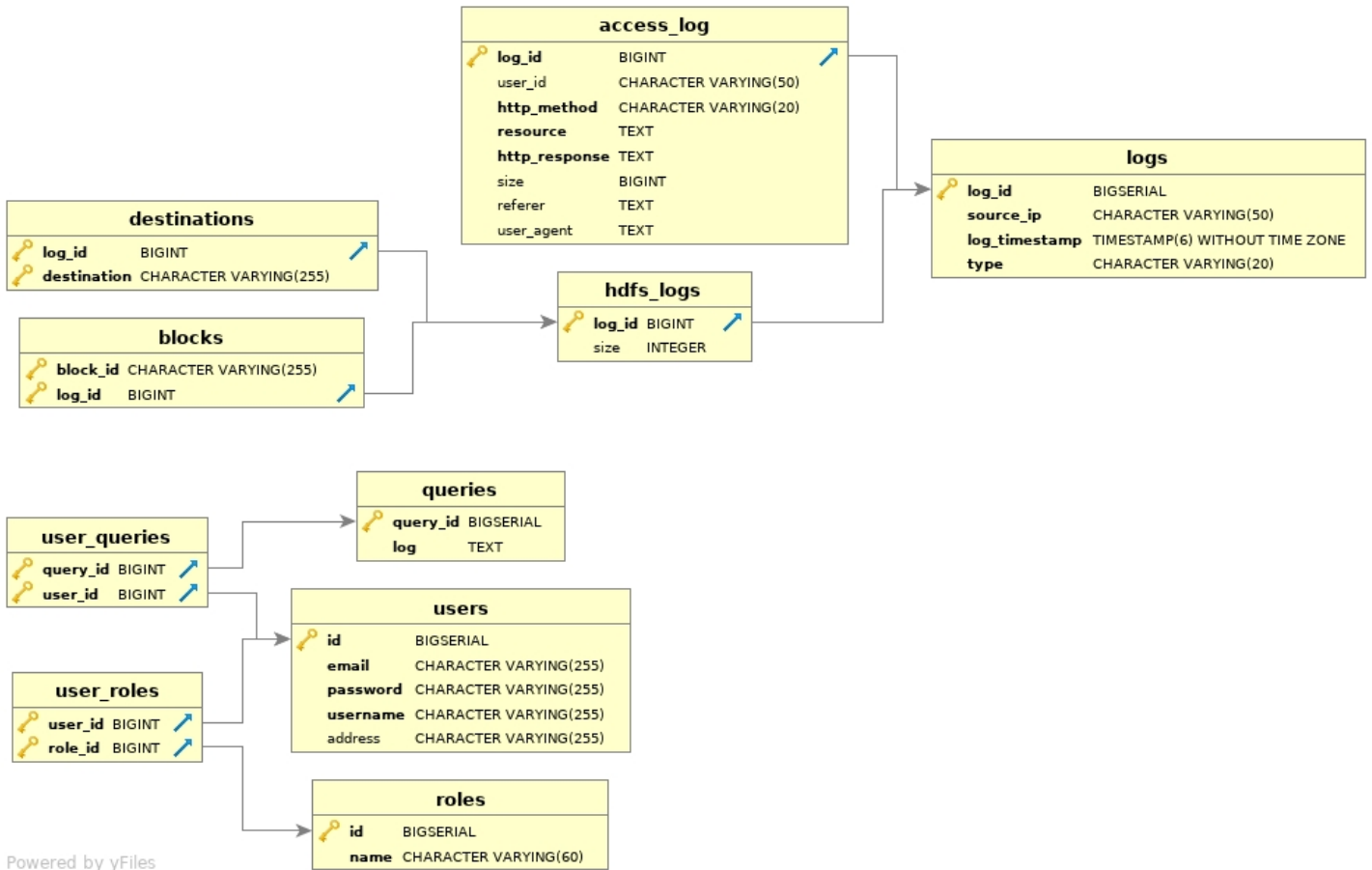
- Import project as Maven project in IntelliJ
- Extract the `logs.tar.gz` from the `logs` directory
- Use the python script `csv_parser.py` within the directory to parse the logs to CSV format.
- Import CSV files using pgAdmin 4.
- Run in the project folder the commands:

```
sudo npm install
sudo npm run build
```

- Start from IntelliJ and you should be able to browse it from:

`https://localhost:8080/`

Final Schema



Powered by yFiles

The main relation in our database is the logs relation. It contains all the information present in every type of log and its primary key is an auto-incremented value which acts as the id for each log. This attribute is also referenced by every other relation in our schema and it allows us to have the database in second normal form (2NF) since there are no values dependent on a subset of any candidate key (the only candidate key here is the minimal superkey log_id). As for 3NF normalization we allowed some tables to remain non normalized (such as access table where some fields might be dependent on other non-candidate key fields) to speed up queries.

We also experimented with various indexes and index types. The most efficient index is a b-tree on the timestamp column on log_id as it speeds up ranged queries like function1 by quite a lot. This can be seen by the times displayed below:

⌚ SystemDB/postgres@Postgres

Query Editor Query History

1 **SELECT** log_db.function1(
2 '2004-10-10 10:10:10',
3 '2019-10-10 10:10:10'
4)

Data Output Scratch Pad Explain Notifications Messages

function1
record

1	(access,3353...
2	(Receiving,17...
3	(Served,4287...
4	(delete,21784)
5	(Received,709...
6	(replicate,7002)

✔ Successfully run. Total query runtime: 7 secs 817 msec. 6 rows affected.

Query Editor Query History

1 **SELECT** log_db.function1(
2 '2004-10-10 10:10:10',
3 '2019-10-10 10:10:10'
4)

Data Output Scratch Pad Explain Notifications Messages

function1
record

1	(access,3353...
2	(Receiving,17...
3	(Served,4287...
4	(delete,21784)
5	(Received,709...
6	(replicate,7002)

✔ Successfully run. Total query runtime: 1 secs 967 msec. 6 rows affected.

Some other indices like on http methods for the last functions did not give us any significant performance gains. Function5 was better with index on referer:

Query Editor Query History

1 **SELECT** log_db.function5()

Data Output Scratch Pad Explain Notifications Messages

	function5 character varying	
1		<div>Successfully run. Total query runtime: 11 secs 307 msec. 657 rows affected.</div>
2	/	
3	91.143.107.26	
4	almhuetten-raith.at	
5	almhuetten-raith.at/	
6	android-app://com.goo...	
7	android-app://com.goo...	
8	android-app://org.teleg...	
9	finder.cool	

Query Editor

Query History

1

`SELECT log_db.function5()`

Data Output

Scratch Pad

Explain

Notifications

Messages

	function5 character varying	
1		<div>Successfully run. Total query runtime: 7 secs 299 msec. 657 rows affected.</div>
2	/	
3	91.143.107.26	
4	almhuetten-raith.at	
5	almhuetten-raith.at/	
6	android-app://com.goo...	
7	android-app://com.goo...	
8	android-app://org.teleg...	
9	finder.cool	

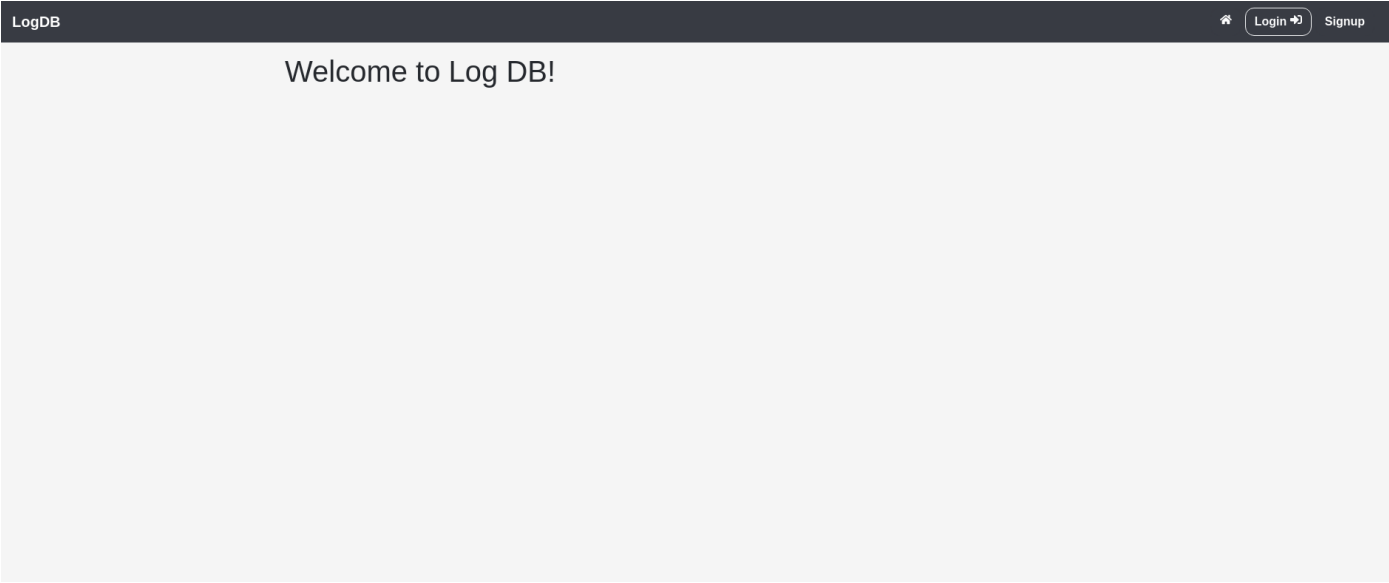
Functions

Their implementation can be seen through the file schema:

```
src/main/postgresql/log_db.sql
```

Sample Snapshots

- Homepage



- Sign Up

LogDB

Home

Login ↗

Signup

Signup

liszt

liszt@piano.man

Liszt Street 404

Submit

- Log In

LogDB

Home

Login ↗

Signup

You have successfully registered in our system. Please login.

Login

liszt

Submit

LogDB

Procedure 1

Procedure 2

Procedure 3

Search IP

Home

+

Logout ↗

You have successfully logged in.

Welcome to Log DB!

- Insert any type of Log

LogDBProcedure 1Procedure 2Procedure 3Search IP

Choose the type of log to insert.

Access

Received

Receiving

Served

Replicate

Delete

LogDBProcedure 1Procedure 2Procedure 3Search IP

Insert Replicate Log

Timestamp:

mm/dd/yyyy, --:--:-- --

Source IP:

Block ID:

For multiple destination IPs write each one of them separated with commas, e.g. dest_ip1,dest_ip2,....,dest_ipn

Destination IP(s):

Submit

localhost:6419

7/8

- Execute server procedures

LogDBProcedure 1Procedure 2Procedure 3Search IP

Procedure 1

1. Find the total logs per type that were created within a specified time range and sort them in a descending order. Please note that individual files may log actions of more than one type.

From:

mm/dd/yyyy, --:--:-- --

To:

mm/dd/yyyy, --:--:-- --

Submit

Procedure 1 Results

No results yet.

LogDBProcedure 1Procedure 2Procedure 3Search IP

Procedure 1

1. Find the total logs per type that were created within a specified time range and sort them in a descending order. Please note that individual files may log actions of more than one type.

From:

12/12/2005, 06:15:00 PM

To:

12/12/2015, 06:30:00 PM

Submit

Procedure 1 Results

Type	Total
receiving	1723232
served	428726
delete	21784
received	7097
replicate	7002
access	2

Github Link

[ServerLogs](#)