Deployment and Analysis of a Distributed PostgreSQL Database System Using the Pagila Dataset

Huynh Minh Thinh
Department of Information Technology 2
Posts and Telecommunications Institute of Technology at HCM city
Ho Chi Minh City, Vietnam
D22CQCN01-N
n22dccn082@student.ptithcm.edu.vn

***Abstract*—The abstract does not only mention the paper, but is the original paper shrunken to approximately 200 words. It states the purpose, reports the information obtained, gives conclusions, and recommendations. In short, it summarizes the main points of the study adequately and accurately. It provides information from every major section in the body of the report in a dense and compact way. Past tense and active voice is appropriate when describing what was done. If there is any, it includes key statistical detail.**

Depending on the format you use, the abstract may come on the title page or at the beginning of the main report.

I. Introduction

In today's digital world, a single database can struggle to keep up with the demands of a successful application. As more users connect and more data is collected, a traditional database can become a bottleneck, slowing everything down or even crashing entirely. Worse, if that single server fails, everything can be lost. A popular and effective solution to this problem is to move away from a single database and instead use a "distributed" system that spreads the work across multiple computers.

This project puts that solution into practice. I set out to build and analyze a small network—or cluster—of four PostgreSQL databases working together. Using PostgreSQL's own streaming replication feature, I configured one main "primary" server and three "hot-standby" replicas. This kind of setup is an industry standard for making sure a system stays online (high availability) and can handle many users reading data at once (read scalability). To test the cluster with realistic information, I loaded it with the Pagila sample dataset, which is designed to look like the database for a video rental store.

This report will walk you through the entire project. It starts by explaining the core problem of scaling a database and how replication helps solve it. I'll then cover the step-by-step process of setting up the virtual machines, configuring the main server and its replicas, and loading the Pagila data. After that, I will analyze how well this setup would support a busy web application, and I'll finish by summarizing what I learned and suggesting ideas for the future.

II. Problem Definition

This will be a revised version of the problem definition in your proposal.

III. Proposed Solutions

This may be a modified version of your proposal depending on previously carried out research or any feedback received.

A. Your first solution

Describe your first solution here.

B. Your second solution

Describe your second solution here.

C. Your third solution

Describe your third solution here.

1) Subsubsection Heading Here: Use the subsubsection command with caution—you probably won't need it at, but I'm including it this an example.

IV. Criteria for Assessing Solutions

This may be a modified version of your proposal depending on previously carried out research or any feedback received.

V. Implementation Methodology

A. System Environment and Network Configuration

I built my four-node cluster using a mix of my main computer and some virtual ones. My physical machine was the primary server, and I ran the three standby servers as virtual machines (VMs) right on that same computer.

To do this, I used KVM/QEMU for the virtualization part, mostly because it's already built into Linux and works well. The virt-manager program was what I used to handle the VMs.

To get all these servers talking to each other, I put them all on their own private network (192.168.122.0/24). This kept the important database traffic separate from everything else. A crucial step was giving each server a static IP address. I did this so the standby servers would never lose track of where

the primary server was, which is essential for replication to work without issues.

All the specific details, like the IP addresses I used for each server, are listed in Table I.

TABLE I
NODE CONFIGURATION FOR THE DISTRIBUTED CLUSTER

Role	Hostname	Operating System	Static IP Address
Primary	Lenovo	EndeavourOS (Host)	192.168.122.1
Standby 1	node1	Ubuntu Server 24.04	192.168.122.102
Standby 2	node2	Ubuntu Server 24.04	192.168.122.103
Standby 3	node3	Ubuntu Server 24.04	192.168.122.104

B. Standby Node Preparation

To save time and make sure all the servers were identical, I didn't build each of the three standby nodes from scratch. Instead, I made a reusable template, which some people call a "golden image."

My starting point for the template was a clean installation of Ubuntu Server 24.04. I got it fully updated and then installed the essential software. The most important piece, naturally, was PostgreSQL version 17—it had to be an exact match to the primary server. I also threw in a couple of basic networking tools like openssh-server and iputils-ping to make life easier later on.

Once the template was good to go, I simply cloned it three times to create my three standby servers. This left me with three identical VMs, which was a great start, but they each needed their own unique identity before they could work together in the cluster. This meant making a few unique tweaks to each one:

1) **Gave it a new name:** I changed the hostname of each server using the hostnamectl command so I could tell them apart. For example, this is how I set the name for the first standby:

Listing 1. Setting the hostname for node1

```
sudo hostnamectl set-hostname node1
```

2) **Set its IP address:** Next, I edited the Netplan configuration file (found in '/etc/netplan/') to give each server its own static IP.
3) **Checked the connection:** Finally, I did a quick network check. I used the 'ping' command from each server to make sure it could reach all the other nodes in the cluster, especially the primary.

The very last thing I did on each standby server was to stop the PostgreSQL service from running with sudo systemctl stop postgresql. This was important because their data folders needed to be empty and ready for the next big step: copying all the data over from the primary server using the pg_basebackup tool.

VI. ANALYSIS AND INTERPRETATION

In this section you will mainly analyze your data in terms of your assessment criteria; e.g., do the data suggest that a particular solution is "cost effective" "environmentally acceptable", "technically feasible" or "affordable"?

Be logical and selective when analyzing/interpreting your research data. For example, if a proposed solution is proven to be far too expensive to realistically implement in your context, is there any value in discussing whether it is "culturally viable" or "technically sustainable"? Perhaps in this case you can focus more attention on solutions that your research suggests are more valid. Do not just throw huge quantities of raw data at your reader and leave them to interpret it. Present enough to transparently support any conclusions you draw and make sure that you offer justifications for your analysis.

Be honest and reflective while discussing your data. Your data might be too limited or unclear to interpret with accuracy—explain this, perhaps suggesting how this shortcoming could be addressed. Admitting the above will help you draw more honest and worthwhile conclusions.

Remember that research is an imperfect and ongoing process that should be open to question and verification. Therefore, unless convinced by the absolute strength of your evidence, you should be tentative in your language choice when interpreting/analyzing research results. Selectively use *hedging* (language which indicates a lack of certainty) to modify the tone of your analysis and any conclusions that result from this.

Here are some examples that show differing degrees of certainty:

- it appears that ...
- it can be tentatively concluded that ...
- it is almost certain that ...
- perhaps the evidence indicates ...
- this seems to point to the fact that ...
- this could be interpreted as evidence of ...
- without doubt its application would prove beneficial for ...

Finally, don't introduce any new content (e.g., research methods or solutions) within this section—this will prove confusing for the reader. The reader should clearly understand that you are, based on specific criteria, interpreting the results of your research in order to test the viability of various solutions to remedy a particular problem. The sole function of this part of the report is to openly discuss your research findings in order to set up your conclusions/recommendations.

A reference to Table II.

VII. CONCLUSIONS AND RECOMMENDATIONS

Conclusion shows what knowledge comes out of the report. As you draw a conclusion, you need to explain it in terms of the preceding discussion. You are expected to repeat the most important ideas you have presented, without copying. Adding a table/chart summarizing the results of your findings might be helpful for the reader to clearly see the most optimum solution(s).

	Growth Media				
Strain	1	2	3	4	5
GDS1002	0.962	0.821	0.356	0.682	0.801
NWN652	0.981	0.891	0.527	0.574	0.984
PPD234	0.915	0.936	0.491	0.276	0.965
JSB126	0.828	0.827	0.528	0.518	0.926
JSB724	0.916	0.933	0.482	0.644	0.937
Average Rate	0.920	0.882	0.477	0.539	0.923

TABLE II
SOME IMPRESSIVE NUMBERS

It is likely that you will briefly describe the comparative effectiveness and suitability of your proposed solutions. Your description will logically recycle language used in your assessing criteria (section IV): "Solution A proved to be the most cost effective of the alternatives" or "Solution B, though a viable option in other contexts, was shown to lack adaptability". Do not have detailed analysis or lengthy discussions in this section, as this should have been completed in section X.

As for recommendations, you need to explain what actions the report calls for. These recommendations should be honest, logical and practical. You may suggest that one, a combination, all or none of your proposed solutions should be implemented in order to address your specific problem. You could also urge others to research the issue further, propose a plan of action or simply admit that the problem is either insoluble or has a low priority in its present state.

The recommendations should be clearly connected to the results of the report, and they should be explicitly presented. Your audience should not have to guess at what you intend to say.

APPENDIX A
WHAT GOES IN THE APPENDICES

The appendix is for material that readers only need to know if they are studying the report in depth. Relevant charts, big tables of data, large maps, graphs, etc. that were part of the research, but would distract the flow of the report should be given in the Appendices.

APPENDIX B
FORMATTING THE APPENDICES

Each appendix needs to be given a letter (A, B, C, etc.) and a title. LaTeXwill do the lettering automatically.

REFERENCES

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

[2] D. Horowitz, *End of Time*. New York, NY, USA: Encounter Books, 2005. [E-book] Available: ebrary, http://site.ebrary.com/lib/sait/Doc?id=10080005. Accessed on: Oct. 8, 2008.

[3] D. Castelvecchi, "Nanoparticles Conspire with Free Radicals" *Science News*, vol.174, no. 6, p. 9, September 13, 2008. [Full Text]. Available: Proquest, http://proquest.umi.com/pqdweb?index=52&did=1557231641&SrchMode=1&sid=3&Fmt=3&VInst=PROD&VType=PQD&RQT=309&VName=PQD&TS=1229451226&clientId=533. Accessed on: Aug. 3, 2014.

[4] J. Lach, "SBFS: Steganography based file system," in *Proceedings of the 2008 1st International Conference on Information Technology, IT 2008, 19-21 May 2008, Gdansk, Poland.* Available: IEEE Xplore, http://www.ieee.org. [Accessed: 10 Sept. 2010].

[5] "A 'layman's' explanation of Ultra Narrow Band technology," Oct. 3, 2003. [Online]. Available: http://www.vmsk.org/Layman.pdf. [Accessed: Dec. 3, 2003].