Contents

[Introduction 3](#_Toc101261680)

[A distributed database is a collection of several different databases distributed among multiple computers. Discuss why a company would want a distributed database. 3](#_Toc101261681)

[A distributed database is a collection of several different databases distributed among multiple computers. Discuss why a company would want a distributed database. 3](#_Toc101261682)

[A distributed database is a collection of several different databases distributed among multiple computers. Discuss why a company would want a distributed database. 3](#_Toc101261683)

[A distributed database is a collection of several different databases distributed among multiple computers. Discuss why a company would want a distributed database. 3](#_Toc101261684)

[Bibliography 3](#_Toc101261685)

# Introduction

The concept of distributed database is one that I am quite familiar with. For my work, I am currently in the process of improving my application’s database performance, and one of my ideas is through distributed databases.

# A distributed database is a collection of several different databases distributed among multiple computers. Discuss why a company would want a distributed database.

A company might consider using a distributed database for a few reasons: data availability, security, scalability, and performance. “Access to data is a critical feature of an efficient, progressive and ultimately self-correcting scientific ecosystem” (Altman, et al., p. 1). Within the growing Internet of Everything, seemingly exponential amounts of data are being uploaded and distributed every day. People have access to so many resources with the stroke of keypad. Along with this unfathomed amount of data comes the expectation of real-time result. We want our data now. Companies, such as Netflix, are aware of our need for data and have therefore adopted the distributed database model. Distributed databases store data in separate locations or on separate databases or computers (Megha, p. 1). This decoupling of data provides a few advantages. Since data is decoupled, the users have to sort through less data. This reduces the query time to receive the data, enhancing performance. Also, by storing different data in different regions, users in those regions will receive their data more quickly since they do not have to sort through an entire superset of data. This is similar to database partitioning, which is distributing data, but within a single database. I have use database partitioning to achieve performance increases within my role as a software engineer. Partitioning creates database tables with subsets of data based on the specified schema. This does two things: reduces the amount of data and blocks unauthorized access. These can be translated of you distributed databases. A company might choose a distributed database to block users from having access to an entire superset of data that they do not need access to. If each of these databases are distributed geographically and the company has multiple locations, this helps the companies work more efficiently by seeing only the applicable data for that location. Distributed database architecture also works well for scalability. If a company were to add a new location, it would a smaller database server to store only the new location’s data, instead of storing the entire company’s data. Only storing local data does have a disadvantage; outside data cannot easily be shared, which is why data replication must be considered between the distributed databases.

# Productivity increases as rapid response times are achieved. Discuss what is considered an acceptable system response time for interactive applications.

Interactive response time plays a key role in

The question of acceptable system response times is a rather open-ended question, simply due to the diverse systems that users interact with. Users might interact with a web interface, API, database, standalone application, etc. The system might be simple and perform one or two tasks, or it might perform ten or eleven tasks and return a larger amount of data. For an interactive web application, I would consider an acceptable response time to be somewhere between 500ms to 1000ms. I base this on the amount of data needed to be loaded on the page. For an API, I would consider an acceptable response time to be between 100ms to 400ms. The applications I create require response times less than 200ms. Sometimes this is difficult due to the amount of data, so I must look within the application to determine how to reduce the response times. This is an instance where distributed databases would benefit the system, by reducing the amount of data the application sorts through. With regard to manipulating an interactive application, the framework that the application is built using also comes into play. When comparing Anguler and Blazor web development frameworks, Nilsson (2021) found that Angular was the more favorable framework due to overall performance.

# A fully centralized data processing facility is centralized in many senses of the word. Discuss the key characteristics of centralized data processing facilities.

# Equipment and communication redundancies are common in today's data centers. Discuss the major types of equipment and communication redundancies found in today's data centers.

# Bibliography