

Ethan Shellem-Prince Report Database and Application

by Ethan Shellem-Prince

Submission date: 08-Jan-2023 07:03PM (UTC+0000)

Submission ID: 193979286

File name: 31725_Ethan_Shellem-Prince_Ethan_Shellem-Prince_Report_Database_and_Application_464571_1047972249.docx (76.1K)

Word count: 1686

Character count: 9182

The Health is Wealth Hospital

Database Solutions

Student Name: Ethan Shellem-Prince

Student Number: 2109506

Module: Databases and Application Development

Module Number: ACCA5026



Abstract

The purpose of this investigation is to improve query performance for The Health is Wealth Hospital by identifying the most important transactions and adding secondary indexes as necessary. The focus is on data queries that are critical or frequently executed in the database. The indexes we have implemented will be based on the needs of these specific queries, with the goal of optimizing their performance.

Secondly, we will present a fragmentation schema for the database system for the Accident and Emergency Department. We will identify the fragmentation strategy applied to each table and provide reasons for the choices we make.

Table of Contents

What is a data query	3
How will the use of Secondary Indexes benefit you	3
Important Queries	3
Possible Indexes that will help you (with explanation)	3
Requirements and Checks to use the database efficiently_.....	4
What is Database Fragmentation	5
Fragmentation strategies for the A&E Department Database	5
Conclusion	6
References	7

What is a database query?

In order to improve database queries we must understand what a database query consists of. A Database query is a request made from the user to the database to fetch or amend any data that is stored within the database.

The queries you make are done using SQL (Structured Query Language). There are many different types of queries that can be completed. For example,

SELECT – The select query allows you to retrieve data from your database.

DELETE – The delete query allows you to remove any unwanted data in the database.

UPDATE – The update query allows you to amend data that already exists in the database.

INSERT – The insert query allows you to add new records into the database.

How will the use of Secondary Indexes benefit you?

Secondary Indexes are additional data structures that we create to assist with accessing data faster. The indexes will allow you to pin point data within specific rows and tables without the need to scan the whole database. This will not only speed up the process dramatically but it will also bypass all unwanted data causing less faults/problems to occur.

By the addition of secondary indexes, we can improve the performance and efficiency of queries that target specific rows. This is very useful with large tables that have data stored in random orders. This is explained further in the article below,

The secondary index is an index type that helps to reduce the size of mapping by introducing another level of indexing. At the initial stage, it selects a range for the columns. Therefore, the mapping size of the first level becomes smaller. Then, this index method reduces each range into smaller ranges. Generally, the primary memory stores the first level mappings to fetch addresses faster. Furthermore, the secondary memory stores the mapping of the second level and the actual data. [1]

Important queries

As you are a very large organization with many different transactions, there is always a need for improvements. We have listed below the queries we think you would benefit from improving.

- 1) Retrieving Patient Information - Database users may want to retrieve information about patients, such as their name, age, bed number, and current medications.

This will allow users to easily fetch key data on the patient with ease. The data may need to be accessed quickly due to critical injury, medical problems or just general data forms.

- 2) Retrieving staff shift information – Database users might want to retrieve information about staff shifts, such as the start and end times, and the staff members working in each shift.

This will become very useful when planning resource by the management staff. Users will be able to clearly see what staff are working what shift with the added extra of seeing there start and end times.

- 3) Retrieving bed supervision information - Users might want to retrieve information about which beds are being supervised by which nurses during a shift.

Users will be able to see which nurse is supervising what bed during there shift, This would become very useful when looking at which nurse should be assisting certain patients.

- 4) Assigning beds to patients - Users might want to assign a bed to a patient and update the bed_id column in the patient table.

Database users will want to update what bed the patient has been placed on. This will help other staff such as nurses and doctors easily locate certain patients if needed.

- 5) Adding prescriptions to patients - Users might want to add a new prescription to a patient and insert a row into the prescription table.

Doctors may add on medication to certain patients due to a change of medical needs. This function will allow users to insert new prescriptions into the database table to therefor be assigned to patients.

- 6) Discharging patients - Users might want to discharge a patient and update the bed_id column in the patient table to a NULL value.

Database users will want to reduce the amount of unwanted data in the database, This will allow users to amend the bed table to unassign a bed to a patient that is no longer in need of a bed.

Requirements and Checks to use the database efficiently

In order for the above improvements to work we have implemented each of the 6 examples into the SQL Code for the Database design. The Code will allow database users to carry out queries for the above statements while attaining quick return information. The new changes will also allow you to amend and change data as explained in the descriptions above.

When adding and remove data into the database such as 'Discharging patients' please ensure you remove the correct bed and assign it to a patient that does not already have a bed. We have implemented different checks into the SQL code to stop simple mistakes from

occurring. For example we have implemented a check to ensure each patient currently in the hospital has at least one or more prescription subscribed to them at any given time.

Another check we have added is to ensure at least 2 Receptionists, 3 Nurses and 3 Doctors are all assigned to every shift, this will make it easier for staff to fill insert new records into the upcoming shifts.

What is Database fragmentation

Database fragmentation is the process of dividing a larger database tables up into smaller pieces called fragments. There are many different types of fragmentation strategies that we can use. These include,

- horizontal fragmentation
- vertical fragmentation
- hybrid fragmentation.

The best fragmentation strategy for a particular table many vary, There are many different dependencies such as the data distribution, access patterns and many other factors. As stated in the web article below the fragments are stored at various sites,

Fragmentation is a process of dividing the whole or full database into various subtables or sub relations so that data can be stored in different systems. The small pieces of sub relations or subtables are called fragments. These fragments are called logical data units and are stored at various sites. It must be made sure that the fragments are such that they can be used to reconstruct the original relation (i.e, there isn't any loss of data). [2]

Fragmentation strategies for the A&E Department Database

Table 'staff'

This table could be horizontally fragmented by staff role, with each fragment containing the staff members of a particular role. This would allow you to easily retrieve all the staff members of a particular role, and it would also allow you to scale the database horizontally by adding more fragments on different servers.

Table 'shifts'

The Shifts table could be horizontally fragmented by date, with each fragment containing the shifts that fall on a particular day. This would allow you to easily retrieve all the shifts on a particular day, and it would also allow you to scale the database horizontally by adding more fragments on different servers.

Table 'staff_shift'

The staff_shift table could be horizontally fragmented by staff role, with each fragment containing the staff-shift assignments for a particular role. This would allow you to easily retrieve all the staff-shift assignments for a particular role, and it would also allow you to scale the database horizontally by adding more fragments on different servers.

Table 'patient'

This table could be horizontally fragmented by bed type, with each fragment containing the patients assigned to beds of a particular type. This would allow you to easily retrieve all the patients assigned to beds of a particular type, and it would also allow you to scale the database horizontally by adding more fragments on different servers.

Table 'prescription'

The prescription table could be horizontally fragmented by patient, with each fragment containing the prescriptions for a particular patient. This would allow you to easily retrieve all the prescriptions for a particular patient, and it would also allow you to scale the database horizontally by adding more fragments on different servers.

For the remaining tables, 'medication' and 'shift_nurse' we don't suggest the use of fragmentation. The tables are quite small and don't have any repeating groups so they may not have any benefit from the use of fragmentation.

Conclusion

In conclusion, this investigation has identified the most important transactions that are undertaken in the database, We then suggested different secondary indexes to improve query performance for The Health is Wealth Hospital. The indexes implemented were based on the needs of specific, critical or frequently executed queries, with the aim of optimizing the overall performance of the database.

Additionally, we presented a fragmentation schema for the database system for the Accident and Emergency Department, where we identified the fragmentation strategy applied to each table and provided reasons for our choices.

References

[Lithmee, "What is the Difference Between Primary and Secondary Index," 25 July 2019. [Online].
1 Available: <https://pediaa.com/what-is-the-difference-between-primary-and-secondary-index/#:~:text=The%20secondary%20index%20is%20an%20index%20type%20that,mapping%20size%20of%20the%20first%20level%20becomes%20smaller..> [Accessed 02 01 2023].

[T.Singhwap, "Fragmentation in Distributed DBMS," 22 November 2022. [Online]. Available:
2 <https://www.geeksforgeeks.org/fragmentation-in-distributed-dbms/>. [Accessed 5 January 2023].
]

Ethan Shellem-Prince Report Database and Application

GRADEMARK REPORT

FINAL GRADE

63/100

GENERAL COMMENTS

Instructor

Hi Ethan,

You got 63% which is good however there are few areas that needs improvement as listed below:

1.
I would recommend you to start attempting your questions as soon as its released. You have missed the sections pertaining to rollback and commit ,secondary indexes identification and fragmentation for the given context.
2.
Ensure that the report is detailed and specific. Always include the screen dumps as your practical work including the modifications you suggest for secondary indexing and fragmentation.
3.
The steps showing the normalization process upto the 3NF could have been a good justification for the created tables.
4.
Referencing and Citations are vital in report writing. I would recommend you to cite atleast a minimal number of credible references such as Books, Journals and conference proceedings that are relevant to the tasks.

I hope you will find these comments useful.

Should you have any questions let me know

All the
Best

Regards

Divya

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8
