Project Reflection

1. Introduction

In this project, my primary focus revolved around managing and optimizing a MariaDB database, specifically catering to a Point of Sale (POS) system. The project's wide characteristics included a broad spectrum of tasks ranging from the initial installation and updating of MariaDB software to the rapid development of tables, execution of ETL processes, creation of views, and handling transactions within the POS database. The significance of these activities is underscored by their real-life implications. The journey through this project not only involved conventional relational database management tasks but delved into the realm of stored procedures, shedding light on their practical applications in enhancing database functionality. A crucial phase of this exploration involved with the complexities of triggers—understanding their mechanisms and understanding their integral role in maintaining data integrity and automating responses to specific events.

As the narrative unfolds, the project transitions into the concept of NoSQL databases, introducing MongoDB. Installation procedures and operational insights into MongoDB are explained, leading to an exploration of the fundamental concepts underpinning NoSQL databases. Notably, the project involves into the utilization of JSON documents, showing their role in structuring and organizing data within the MongoDB framework. This introduction serves as a compass, guiding the reader through the database management, from the foundational setup of MariaDB for a POS system to the complexity of triggers and the advent of NoSQL databases. Headings inserted throughout the text give viewers hints about what's to come and an overview for predicting the complexities of transactions, stored procedures, triggers, clustering and NoSQL databases.

2. What I learned from each milestone

Milestone 1 – Infrastructure

In milestone 1 just like building blocks we started at the foundation level i.e create an AWS Instance and set some firewall rules to allow network connections. After that I configured Linux and made sure that the software is up to date. Then create and configured the user accounts and install the MariaDB, post that I had to create user accounts in the MariaDB and configure the open ssh to allow key based authentication. After doing all the steps for the first time it took around 2 hrs to understand what was going on. Then I started creating tables by using the ER diagram in the infrastructure pdf. Post creating the tables, I ran them and I was flooded with the errors. The 1st error was the order of the tables that I created, customer table at first followed by other tables in the ER diagram. My 2nd error is the ORDER table as the table name itself is a keyword. I had to search what was wrong with the name, then found the reason. The 3rd error was the composite key.

Then I saw the Hints pdf and went through it and realized my mistakes and redid my whole script from the beginning of the city table followed by the product which has no foreign keys and good to start with and the Zerofill in the city table. Then corrected my Orderline table by defining the composite key in the right way. In this milestone I learnt to be precise, I tried and worked out a lot of new things like using AWS instance which I didn't know how to use and got to know the installation and setup. I did learn to start defining tables which have no foreign keys in it, about why Zerofill is used and how to name a table even though it is a keyword, by using backticks or using the database name in front of it.

Milestone 2 - ETL

In this milestone the next steps are to load the raw data. I did face some challenges in this too. ETL is Extract Transform and Load which means we need to extract the data, by this it means we get a lot of data which won't be in right format let's say we have duplicate names, date of birth format, and null data. For this milestone DR. G gave us datasets in the formats of CSV files we had to import them into database. To work this I pushed my IP to get the files into the database in /home/dgomillion/. After this I started to script, to achieve this we need to create temporary tables. We can create temp tables with the temp key word but it will expire as soon the session ends to avoid this we create normal tables which has temp in its name so that we can drop them at last of the script to with hold the data in temp tables until our work is finished. With naming convention as temp in it we create additional 6 tables. Now we need to extract the data, to import the data into we use LOAD data local infile command to transfer the data from /home/dgomillion/ to our respective tables. After successfully importing the data we need to transform the data.

We need to change the empty spaces to Null, and birth date formats are changed by using STR_To_Date. And some people have the date of birth as 00-00-0000 which not valid so we need to update them. After finishing everything I started to run the script and then verified tables to check whether my data is formatted correctly. Here I have found some data which was wrongly formatted in Product table there should be no "$" and "," characters in the table so I have gone through the script and wrote an update command to remove the characters. After verifying the data, I started to load them from my temp tables to my original tables. After doing everything I have dropped my temp tables. In this milestone I learned how ETL works as a data engineer it is important that we have to understand that all the data which we get is not true and all the data is not in correct format so we need to ensure that all the data should be changed. And I did learn about the commands like LOAD data local infile and how it works. I did another mistake where I forgot to count the product ID and do the Group by in OrderLine table. So, then I went through the clarifications pdf and realised my mistake and updated it too.

Milestone 3 – Views and Indexes

In this milestone I got to learn about views and materialized views and indexes. The concept of these views is to ensure that the users can not change the data, in fact it allows users to narrow down and show only what he needs to see. Meanwhile materialized Views are the updated version of the views where it takes the snapshot (Oracle calls them earlier) of the results. These are built in MariaDB we need to create the table and then we need to select the required data. Whereas Indexes are used to fetch and look for data faster. As Dr. G mentioned in the class that indexes such PRIMARY INDEX ensures to not to have any other duplicate primary key.

While doing this milestone I have created the views and executed them to know how it works and I created a view to get customer names combining the last name followed by first name. Next I created another view to get customer details which should retrieve me the details of the customer, and with the next view I created product buyers in which I need to retrieve data from two tables by using the concept of joins in which I used left join from product → orderline → order → customer and then group it with the id and name of the products. This results the details of the customers who bought the brought.

Now to create the materialized views we need to create a view first which we already did. After the creation of the view we can insert this into materialized view table to store retrieve them when ever we need. These are helpful when there is a need for faster reads. Then we have Indexes where we can create them on whatever column we need to retrieve by using the table name and the column name. Even though we can't see the indexes if we intend to see them we have to run SHOW INDEX FROM followed by table name. So, I did learn another lesson in this milestone apart from views is that case sensitive even though I did all the views correctly I got a 0 in this milestone that's because adding a single extra letter in the table name of CustomerNames. Then I remembered DR. G saying that you have to be careful about case sensitivity and I learned it the hard way. So, my take away from this milestone is that you need to make sure that you should be careful while defining the names or another.

Milestone 4 – Transactions and Prepared statements

In this milestone we get to learn how real time transactions happen especially in banking systems. Ever wondered how the money in one account is transferred from one account to another account safe and secured way. Transactions are the concept that was implemented in this process. So, to initiate the transaction we use the command start transaction or begin, and we were advised to write a command SET autocommit = 0; so that the transaction won't be committed unless we give the commit command or roll back command to execute those command which are inside transactions. Commit command will make the transaction changes permanently at the end of a transaction whereas the ROLLBACK command will make sure to undo all the changes which were given in between the transactions and revert the changes. To check how the transactions work we need to connect to putty and open 2 sessions and set them side by side to notice the difference when the transactions happen. Another command is TRUNCATE which is similar to delete the records but the truncate command deletes the data in row by row. We did work on this command by using in between transactions and tried to rollback it. But it didn't work out so make sure the data to be restored we had to use all the 3 scripts and then commit.

The other concept is Prepared Statements, this is a concept where we define a command and then execute it to work. So, this concept is used when there is a lot data which has to be entered into the database. To work this, we use "PREPARE Structure Name" and followed by command, to execute this whole statement we "EXECUTE Structure Name" and followed by the values. For deleting the prepared statement, we use DEALLOCATE keyword. After working on prepared statements, I learned that for a single row of data it might be lengthy and tricky compared to insert statement whereas using for multiple rows of inserting data will be appropriate. Another concept called SQL Injection, here it is the concept about how the malicious injection might happen to the database.

Milestone 5 – Stored Procedures

In this milestone, the concept is stored procedures where it works with set of actions that need to be taken when there is a database that has to be clumped which makes it easier and faster. Stored procedures mostly work when we have databases in which the data need not to be moved to a different server and back to the persistence layer. To achieve this, we had created procedures. Firstly, we need to create a new procedure and altered some columns in the tables and created some procedures to replace all null data. For a procedure we need use he keyword DELIMITER and start write the commands and close it with a DELIMITER. Then I created another procedure where I need to delete the data from materialized views which were created from the earlier milestone and insert the new data without using DROP, TRUNCATE as we need the transaction to be safe.

After doing this milestone I get how the stored procedures work and when and where it can be helpful and the syntax which has to be used for the procedures to work perfectly. Apart from the stored procedures concept I did learn another lesson. While I was working with this milestone my AWS session was stopped after 4 hours then I should have opted to start the lab again instead of that I clicked reset unknowingly. So, this made my whole instance data got deleted and restored to new instance in which all my previous milestone data was lost. On a plus side I had time to create a new instance and updated the software just like I did in milestone 1. So, this milestone did remind me that I should have a back up of my codes which I did and helped me not to get a 0 in the milestone.

Milestone 6 – Triggers

In this milestone the concept is about triggers. Triggers are similar to the stored procedures but the key difference is that triggers automatically execute them selves when the data is inserted, updated or deleted. In this milestone I started working on since the beginning of the week as it is huge. Since there is an implicit transaction covering every statement with the before and after triggers. From this milestone I did learn how the triggers behave. When ever I went to a grocery store I used to look at billing screen, when ever the billing person scan the product the amount on screen tends to add up it will distinguish between products and the number of products we added let's say 2 bottles of a soda. So, while looking at it looks so fast and easy to update and delete some unwanted items. But after working on this milestone I understood that the concept of triggers is used in this.

So, while working on triggers we had to call the stored procedures which were created in the previous milestone. Then we need to create a new table called sales tax which comes with the Zip code and tax rates. Then we created a new temp tables for the sales tax and did the ETL process and loaded the data in the original sales tax table. After this process I had to alter some tables and created the triggers. We used triggers when there is price update on a product which we used AFTER Keyword. In the next trigger I did the update before inserting data into tables for a new unit price. In another trigger I did use all the 3 keywords like the insert, update and delete to work when ever there is a change in the OrderLine table. I used another trigger to set the quantity 1 wherever it I have NULL. Triggers can also be used when ever there is quantity constraint such as if the customer ordered the product which is greater than the available quantity. All these updates are done easily but I faced some difficulty while rounding up the sales tax and calculating them and updating the results in materialized views which understood how to do in the class.

Milestone 7 – Clustering

In this milestone I did learn about clustering. Clustering means creating multiple servers and sync the data and work them together. Cluster are used in many ways some use this method to partition and the tasks in which that particular server work on specific role. In some clustering some work independently based on the business requirements. In this particular milestone I learned about simple replication in which I created 2 servers. To work this, we named 1st Server as PRIMRY(MASTER) and accessed it by doing all the reads and writes. The 2nd server works as SECONDARY(Slave) in which the only sole purpose of this server is to retrieve the data i.e for Reads. The main moto of this milestone is to secure the data when there is issue with the primary server. In case if the server 1 got crashed we have a backup plan of server 2 in which our data is safe and ready to use.

So, in this I created 2 new instances in AWS and did all the updates and installed MariaDB on them and in the 1st server. I configured the replication to work from primary to secondary by using the private IP address because it is unique and wont change. I made sure that port 3306 is open between the primary and secondary but not to rest of the world to ensure data protection. Then updated my cnf file on the primary first by creating the replication user with a new user name and password and granted the ability to replicate the data from primary to secondary server. Then I did the same steps in the my cnf file on the secondary server by giving a new sever id which is unique. And repeated the same steps which I did on primary server and it is mandatory and important that too set the server to read_only as we needed it to have replicated data but not the ability to write in it. Then create the dgomillion user on the primary to replicate to the 2nd server. After all this I transferred all my scripts and ran them on primary and it took some time to replicate on the secondary server.

Then I verified by adding some new data on server 1 and then I tried to delete the new entry on server 1 from server 2. By doing this the 2nd server didn't allow me to do the update as it is restricted by my that the server sole purpose to read-only. After working on this milestone learned that it is important to have multiple servers as a backup. I did learn that it takes some time to replicate the data from primary to secondary and this might cause some stale data due the replication time and it is called as slave lag.

Milestone 8 – Peer to Peer Clustering

In this milestone, I learned about the next phase of clustering. The purpose of this milestone is to allowing the secondary servers with granting the read and write permissions This type of setting will be done purely based on the requirement of the business so, the older clustering I used when there are many read requests whereas in this method to get the consistent data and to avoid slave lag we will be using as well the 3 servers to sync and allow replication. This will enable to have multiple read and write replicas too. It can also be used in a situation where the 1st node in the cluster will be the actual changes in table and the other node can be used to running triggers and stored procedures and gives the original data. In this cluster we will be using virtually synchronous replication which means that the delay in replication wont cause inconsistent data.

Galera is used because of its unique nature of peer to peer clustering of relational databases. In this peer to peer clustering i created 3 instance and named them A, B, C where A is the primary server and the other 2 will be my secondary or replicas. After the creation install the required MariaDB software and update it. Then created the dgomillion Linux user and configured the public key for the authentication, pushed the zip files of data to all the 3 instances and moved the scripts and unzipped the csv files in all the 3 nodes. Then updated my cnf file on all the 3 nodes to configure peer to peer replication using al the 2 private IP address of the nodes in gcomm:// part. Post creating the dgomillion user on any one of the nodes and allowing it to replicate to rest of them. Then I did some changes which was meant to be done to acknowledge the actual working and usage of peer to peer replication by altering the data in one node and verifying it from another node.

Milestone 9 – JSON

In this milestone, I learned about how to write into json document and how do we json document and its purpose. To import the relational data into a NoSQL database we use certain types of format. Mongodb is the well know NoSQL database to convert the data we use json format. This milestone I had convert my relational data into json document and later imported that into Mongodb. For this milestone I used json_object and json_arrayag functions. While doing this the default location of the desired files will be in /var/lib/mysql/pos. While creating the json documents for the customer 1 it should include all the details customer and if has any null should be removed and combine all the data into a single line then the returned output should be named into the Cust1.json. for I had to use only json_object and json_arrayag which I found difficulties to work with but after lot of changes I got to know how they work and how the output should be.

In a similar way I did all the required data and converted them to json documents but I found some difficulty while doing this milestone is that I had change my path every time when the output was correct I had to back to ec2 user then remove all the code and rewrite the code and run it and change the path and check the json data whether it the expected data or not. And also, in the last json file that is Cust2.json I couldn't run the code due time constraint and submitted it. Then I found reason why I could do that and changed it. So, I did learn that I shouldn't wait till the last minute.

Milestone 10 – MongoDB

In this milestone, I got to know MongoDB works. It is the most used NoSQL database. Since it is the document database it accepts JSON documents. Since we worked on JSON documents in the earlier milestone it is the time in which I had to import them to this NoSQL database. But before that I had to install the database and configure it. While doing process I installed the software first and tried to install repo files but it didn't work. So, I had to uninstall the mongodb software and then I installed the repo files first then I installed the software then it worked with out any error. And I also got to where and why I had to replace $releaserver with the number 9 which defines the release version of the software.

Then I used import commands to import the json documents into the mongodb software. To verify the documents I gave commands which and are used to retrieve the data. Through this I got the data so fast and quick but due to denormalization I did get stale data. But that's ok because we need the data in a faster way so we had to use the repetitive data in json documents. Through this milestone I learned anew database interface and how it works. Through all this milestone I learned a ton knowledge which I can use them in the real time world applications.

3. The most difficult Section

The most difficult section i would say would be the last 2 milestones i.e JSON documents and MongoDB. It is that because, for the earlier milestones we had a lot of time and the interaction with MariaDB got adapted and i knew what needs to be used to manipulate the whereas to convert the relational data to a NoSQL format took a lot of time to understand. While defining the JSON documents even though the creation of the documents is easy but to understand how use the JSON command (OBJECT, ARRAYAG) and when to use is the difficult part. After working on the queries for a time i understood what's going. The difficulty is not because of the concept it was the time which i took because i have no previous experience working with JSON documents. So, i would say this is one of difficult section.

Another section was the MongoDB milestone. Even in this milestone i have no prior experience working with this. I felt difficulty because while working on this milestone there was miscalculation where the count of products in an order is 0. I found this when i trying to achieve a how to find fraudulent orders. So, i had to go back to json.sql change the calculation and remove the older json documents in MariaDB and run the script. Now i had to remove previous imported json documents in mongo dB and then do the importing again and then i had to run the command on mongo dB to get the correct data. I guess there might be an easy way but working on json and mongo is the first time i felt difficult in while working these milestones.

4. The most surprising thing I learned

The surprising thing which i find is that how the NoSQL databases work. The presentation in the class helped me to understand how the databases work. I also got know the types of databases are their functionality. My presentation was about Redis database. While looking at the purpose of Redis database i was so surprised how the Redis works i was particularly intrigued with the concept called sorted sets which works in Redis. Redis is a key value database, this is used in real time applications. It is most popularly used in the score boards or leader boards in online multi player game. This concept of sorted sets has many surprising elements in it. I was not sure about what a in memory storage is but later on while going deep into the functionality on Redis, i understood how the in-memory database works. Now i can strongly explain about the Redis database and concepts of another NoSQL database too.

The concept of sharding and the working of sharding made me surprise because i haven’t heard that before and listening and going through the concepts of sharding was a surprise to me.

5. My favourite Section

My favourite section would be that transactions and triggers. While doing transactions i got know a lot of subject which i was interested and i got hands on experience in this topic. I tried a lot of scenarios to see how the transactions work. In that milestone i did learn about prepared statements. More over in that milestone it describes how and where to use the prepared statements. Even though triggers are not loved by everyone i find it most interesting and favourite because it helped me understand the concept how it triggers the data to manipulate it. I was always considered how triggers are used in real life scenarios, while working this milestone i understood where to use triggers and how effective it can be in a relational database management. Throughout this course i also noticed that RDBMS will follow all the ACID properties and transactions safe.

6. Why I think this project was useful

I must say it helped me a lot. Yes, this course project was helpful that not only i grasped the subject but having hands on experience gave me confidence on the subject. Instead having a theory this course project will allow any person to understand what’s actually he is undergoing. Every concepts of this course are useful. I looked at this project as Super Market (H.E.B or Walmart) database which comprises with the details of customer, product, orders, orderline, city, sales tax etc. Yes, i would say it was worth your time to build such a tremendous project where students won’t feel it as a class project moreover it makes them feel that they are working a real-world database. Yes, it would help me the future, i can confidently talk about the databases and he concepts and can discuss about how it works with anyone.

Course Reflection

1. What this class is really about

In the beginning of the course I thought it was advanced database. While the course started with creating the tables and how the tables should be designed. Then going through the ETL I thought it was just cleaning the data and setting it in a right way. Later when concept of views started I understood that why views are used slowly this concept diverted on how it can be used in the business layer where a lot people are considered. Then moving on to transactions it explained how real-life money transactions happen and how important it is to maintain the ACID properties. Then the stored procedures and triggers were involved and saw their mechanisms. The later on topics delve deeper into the clustering and peer to peer replication with how to maintain a backup and how important it is to have them, apart from these based upon your business layer plays a crucial role.

Now I'm at the end of this course and i understood what Advanced Database Management System really is. It is no just about creating tables or cleaning it, it was about how to choose the right database to use. Now my opinion has changed through the course and it really helped to understand what to decide when it comes choosing the right database. It was never about the learning about database it was all about when/why to choose a NoSQL database, when/why to choose a transaction safe database, when/why/which to choose the best persistence layer. This course taught that me how manage the data based upon the real time based on the real time requirement.

2. The best part about participating in class

The best participation was the perusal comments because, I had the chances to express my thoughts on the video even though I couldn't talk in the class. In this I have given a lot of my opinions on video where other students can learn too. I too got to a lot their ideas and how they think and loot the concept. Even though we won't be talking and knowing about the person in class but this give a path to understand people and their views on the subject. Some of them who have prior work experience or students who worked in internships on this kind of databases and how this course revolves around these topics. This platform also made sure that the comments which we post might be useful or not and apart from this we can discuss one on one which other students on the topic additionally. There is another option where i can takes notes instead of write it some where you can actually note down the comments which you like and think that would be helpful.

Th next Participation which I like is the presentation about NoSQL. Which allowed students to work on a single goal. This kind of participation is useful because one-person cant knows everything about the database so by combining 3 different mindsets can be helpful and share their ideas and opinions ensures to involve everyone in the project rather than a single person working for presentation. This course taught me many things about the concepts, participation, and even this paper reflection which I'm writing is also helpful. This unleashes the concepts which i covered and while doing this work I did forgot about some topics which worked on and now it just got reminded again. By this end of the reflection essay it resembles that how much i have learnt about the course. So, I have learnt a lot in this course work which will definitely cherish and contribute to may future.

3. How to select the best persistence layer

In this section I’m going talk about the types of relational and NoSQL databases and look at what they going to offer us. We have different types databases which can be used based upon our requirements. They are File system, Relational Database, Key Value database, Document Database, Column-Family Database, Graph Databases.

Relational databases

In this database it follows schema. Majority of the relational databases have schema. This data can be stored in tabular schema. The relational data bases can be used when we need scalability and strong ACID properties. It can be suggested for Banking systems, Grocery Stores or for a Medical Drug Store, e-commerce platforms, Human resources Management System, customer Relationship Management systems. We can use MySQL, PostgreSQL, Microsoft SQL Server and Oracle Database as it comes with most of the features.

Files

We use this as a persistence layer it depends on the nature of the data and the system’s requirement. It can be suggested for a smaller-Scale application with simpler data structure like csv formatted files. For a complex data binary file system might be useful. The choices we can have for this system is Plain Text Files, Binary Files.

Key-Value Datastores

In this type of database, they are faster because of in-memory data caching. It is a NoSQL database that means it is a schema less and the data will be in key and comes with value. To access the data key is important. The queries are simple when compared to relational model. The data can be stored in the form of blobs They can be used when we require high speed access. To use this, we can use Redis, Apache Cassandra, Amazon DynamoDB, Memcached and Riak. It can be used when there is a need for Caching systems, session storage and real time updates and analysis.

Document databases

This type of database can be suggested to be used when we are dealing with semi structured or unstructured data, they have factors such as schema flexibility, scalability, and ease of development. It supports quasi transactions. It is suggested for event logging, Blogging. It is not suitable for complex transactions. It has built in auto sharding. In this u can use xml and any software other language and combine them to use. This can used by downloading these MongoDB, CouchDB, Elastic Search, Firebase Fire store. This can be used for Content management, product catalogue and mobile applications.

Column-Family Databases

This database can be used when we want design and handle large volumes of data with high writes. We can choose when the we need factors like scalability, fault tolerance and data model suitability. It has implicit schema which means it can have different schema. clustering can be done and for compaction SST will be used. The writes are bit complex for replication as it needs to be written to logfile→Memtable→Replicate→Successful. This type of database offers a feature called super columns where it has multiple column under one column each and can be indexed. Atomic writes are possible that row level writes use CQL (Cassandra Query Language) and it doesn’t support joins and transactions. Apache Cassandra, HBase, Amazon SimpleDB and Google Bigtable are some software which are column-Family. It can be used for time series data in IoT applications, can handle large scale sparse data.

Graph Databases

In graph database it is a NoSQL database it typically stores the data in the form of nodes and then it uses edge to connect to another node. It is faster to query, nodes know about incoming and outgoing of data. It follows the properties of ACID, replication is easier than sharding because graph exists on a single server. The nodes or edges can be updated at a same time with a transaction. The sharding can be done on the application layer. This database is built on top of relational database so transactions are also possible. Good for finding the routes. Neo4j, Amazon Neptune, ArangoDB, Janus Graph are some software which works on graph database. Social media, fraud detection, network analysis can be done with this database.

4. Elevator pitch

Hello, as data engineer i have a comprehensive background in advanced database management system. In my recent project, I explored the relational databases, implementing and optimizing MariaDB for a real-world Point of Sale system. I spearheaded tasks from software installation to the development by using views, transaction, stored procedures and triggers, ensuring robust data integrity within the database. Beyond the concepts of relational databases, I effortlessly integrated NoSQL databases, specifically MongoDB, into our ecosystem. This involved understanding and implementing JSON documents, figuring out details of document-oriented storage, and enhancing our data management capabilities. The project showcased my ability in making decisions to pick the best persistence layer for diverse scenarios, whether it be the familiarity of SQL in relational databases or the flexibility of NoSQL in handling unstructured data. My expertise extends across a spectrum of database technologies—files, relational databases, key-value stores, document stores, column-family databases, and graph databases. This versatility enables me to strategically align the choice of a persistence layer with the unique demands of each project. I bring a comprehensive understanding of both traditional and cutting-edge database systems, ensuring optimal solutions tailored to the dynamic needs of modern applications.