

Final Project Report

on

**Home Office Operations Management**



**DATABASE FOUNDATIONS FOR BUSINESS ANALYTICS (BUAN 6320)**

Under the guidance of Prof Kannan Srikanth

BUAN 6320 - FALL 2022

Prepared by:

|  |  |
| --- | --- |
| NAME | NET ID |
| Shivali Mavi | Sxm220124 |
| Manisha Varma | mxv220022 |
| Harshini Parepally | hxp220004 |
| Shree Lavalika Kalepu | sxk220056 |
| Sohan Rao Kakkerala | sxk220132 |

Submitted on: 12/12/2022

**Individual Contribution:**

|  |  |
| --- | --- |
| *Name* | *Contribution* |
| Shivali Mavi | 100% |
| Manisha Varma | 100% |
| Harshini Parepally | 100% |
| Shree Lavalika Kalepu | 100% |
| Sohan Rao Kakkerala | 100% |

Table of Contents

|  |
| --- |
| Introduction  1.1 Objective      1.2 Project Overview      1.3 Project Deliverables      1.4 Features & Benefits |
| Database Design  2.1 Creation of Tables  2.2 ER Diagram |
| Queries  3.1 Joins  3.2 Triggers  3.3 Stored Procedures  3.4 Stored Functions |
| Conclusion and Future Scope |
| References |

**INTRODUCTION**

In the past, for employees working from home the employers believed that they would be too easily distracted at home and there will be a huge gap in the communication among peers. A few years ago, remote working was very uncommon, however, employees today are more connected than ever which completely changed the dynamic for work from home after Covid 19 Pandemic.

Remote working had become a necessity to keep up with the fast pace of technology and business. 10,000 workers were polled by the University of Chicago's Becker Friedman Institute for Economics, which found that they believed working from home was equivalent to working in an office in terms of productivity. In fact, 30% of the respondents said working from home increased their engagement and productivity. The same study team estimated that from the middle of March 2020 to the middle of September 2020, commute times were cut by 62.4 million hours per day, resulting in total time savings of over 9 billion hours.

It is safe to say that the remote work environment is replacing the standard norm of traditional office management systems as more and more people around the globe are [acclimatizing](https://www.wordhippo.com/what-is/another-word-for/acclimatizing.html) to the ongoing trend of working from home or even during vacations just to have a change in atmosphere. This also increased the need of a system which manages, tracks and analyses the day-to-day operations in order to ensure smooth conduct of business in a remote set-up.

**OBJECTIVE**

To ensure a smooth functioning of an organisation, create a value for the customer and be universally productive we need a system to tactfully administrate and conduct daily operations and activities of the organisation. This is achieved by an operations system which focuses on the effectiveness of the business processes and its services.

With rapid shift in working environment, we need a flexible operation system which can work well for both in-office work environment as well as remote work. Some of the objectives for a management system to systematically operate the business are as mentioned below:

1. Streamlining the workflow: Every resource in an office setting holds specific tasks and activities which are required to fulfil the goals and visions of an organisation. Seamless execution of workflow which optimises the tasks, eliminate necessary steps and improve the overall system is essential.
2. Effective Resource Utilisation: Utilising resources effectively and to deal with unplanned circumstances such as underutilisation, overutilisation and wastage which drastically impacts the functioning and efficiency of an organisation we need a precise system.
3. Coordinating resources: The major factor which helps in the overall growth of a business is the coordination amongst the resources which prevents the uncertainties and disconnect between them.
4. Swift Delivery: The operation system is also responsible for making sure the deliverables expected are fulfilled in a timely manner. This includes planning, scheduling, monitoring and controlling the operations. The system is also responsible for ensuring that all the activities are performed by business are done correctly which can be carried out by identifying and prioritising tasks to be done by each department.

**Features of The System**

1. **Centralized Database System:** The purpose of centralised database system is to make sure that there is a single point to access all the data. This allows easy retrieval of information and find out the needs of the business services. Some characteristics of our project includes:

* Documents linked to services are kept in one place (contracts, project plans, meeting notes, policies, etc.).
* Querying of document metadata through a search engine.
* Creating documents using templates.
* Document auto-reference to relevant tasks, projects, etc.

1. **Service Planning and Optimization** is the process to identify the best way to achieve our desired goals. It involves identification, evaluation and implementation of strategies and resource available to help the organisation in achieving its goals and objectives. Our system includes below features to implement this:

* Establishing service-related processes (approval workflows, corporate policies to follow, etc.).
* Management of ongoing service improvement (discovering best practices in service, getting input from staff, and voting on suggestions).
* Service level agreement management.

1. **Project Tasks and Service Request Management:** is a process of managing the activities that are needed to complete an organisation’s services and meet the business requirements. Some of the aspects are as follows:

* Assignment of tasks and requests as well as tracking of progress.
* Automated scheduling of employees' working hours in accordance with given tasks.
* A customizable task/request schedule viewer for teams and staff (with a calendar view, Gantt chart view, etc.).

1. **Time and Expense Management** can be a major issue for a company. It is important to have an efficient system in place that will help save the overall cost, increase efficiency and productivity. The cost of the businesses is determined by number of employees, equipment, supplies and other resources needed to run the business. This is a very important aspect to consider when planning its operations so as to reduce the cost and keep the business running smoothly.

* Customizable cost claim forms and timesheets.
* Employees' billable and non-billable hours are tracked throughout time.
* Expenditure management (submitting receipts for transportation expenses, etc.).
* Automated system for approving employee timesheets and expenditure reports.
* Automated invoice creation, including tracking of approval statuses (approved/pending, etc.) based on authorized billable hours and expenses.

1. **Project and Service Analytics** is a comprehensive approach that identifies the problem in an organisation and develop an effective, efficient and cost-effective system for managing and delivering services. Our system includes below properties to ensure the same:

* Impact of change analysis (shows how changes, e.g., unexpected delays, affect service schedule or project progress).
* Using real-time project/service delivery monitoring to base revenue forecasts (can be drilled down by service types or projects, clients, business units, etc.).
* Reports on resource use.
* Risk assessment for projects and services using AI (for instance, by contrasting client demand with a company's available resources).

1. **Supply Management:** This is a set of tools which manages resources effectively that meets customer requirements. It includes – inventory control, scheduling tracking and reporting which helps in managing production process and services. Our system has following features for the same:

* Tracking of supply quantity and position using barcodes.
* Setting up reorder thresholds and low-stock supply notifications (can be tuned for different locations).
* Formulation and approval of supply purchase orders.
* Tracking order status and reviewing order history.
* Choosing preferred vendors for orders.
* Demand and supply forecasting (based on historical demand data and seasonal demand fluctuations).

**Benefits**

1. **Task progress:** The product feature helps track the task progress assigned to an employee or individual. This helps in tracking the issues, resource allocation, staffing, estimated time for the completion of work.
2. **Monitoring the issues:** With our system we can record technical issues that are raised as tickets by the employees which are stored as data with a complete explanation of the issue, which generally can help in analyzing the data for future troubles and save time.
3. **Shift Planning:** With increase in the recruitment of the employees, it is a problem for staffing them with already existing employees. As most companies are opting or leaning towards a hybrid working system, it is essential to have an efficient seating system and resource allocation. This management system will help plan and track the shifts.
4. **Data Backups:** Backing up data from a single location is simple, however in a hybrid setup employees operate from several locations worldwide. Our product will help in simplifying the backup process.
5. **Data Security:** Data is housed in a secure central location, and many access privileges can be assigned to multiple people. Our system keeps track of the members or employees that have the access to or were given access for any of the reasons.
6. **Data Accessibility:** One of the main benefits of our product is that the same business data can be made available to any employee/individual at any time and from any location. This system allows multiple users to access information that is accessible remotely and twenty-four hours a day, seven days a week.

**Database Design**

The first step to designing any database in SQL is to identify what to include and what not to include. The subsequent steps involve deciding how the included items relate to each other and then setting up tables accordingly.

The initial step in designing a database is deciding which aspects of the system are important enough to include in the model.

Firstly, we created a list of all the objects we could think of relevant to our product Home Office Operation Management Systems.

After developing reasonably complete set of objects, we moved on to the next step: deciding how these objects relate to each other. Some of the objects are major entities that are crucial to give the results we want. Other objects are subsidiary to those major entities.

Major entities translate into database tables. Each major entity has a set of *attributes* — the table columns. For instance, in our database, we have the ‘Employees\_T’ table, in  that we have included key details such as EmployeeID, FirstName, LastName, Gender, Age, Email, PhoneNumber, Address, PostalCode and other relevant information. Each attribute of an employee becomes a column (and a column heading) in the Employees\_T table.

**Creation of Tables**

We created the tables defined in our database using SQL’s Data Definition Language (DDL).

Following are the tables with their key attributes:

**Employees\_T**  🡪

Columns in the table:

|  |
| --- |
| EmpID - PK |
| FirstName |
| LastName |
| Gender |
| Age |
| Email |
| PhoneNumber |
| StreetAddress |
| City |
| PostalCode |
| State |
| Country |

**Department\_T**   🡪

Columns in the table:

|  |
| --- |
| DeptID - PK |
| DeptName |
| DeptDirector |

**Team\_T  🡪**

Columns in the table:

|  |
| --- |
| TeamID – PK |
| TeamName |
| TeamLead |
| DeptID - FK |

**Work\_Mode\_T  🡪**

Columns in the table:

|  |
| --- |
| ModeID - PK |
| ModeName |

**Job\_T  🡪**

Columns in the table:

|  |
| --- |
| Position |
| Compensation |
| HireDate |
| EmpID - FK |
| TeamID - FK |
| ModeID - FK |
| EmployeeType |
| TimeType |
| PRIMARY KEY (EmpID, TeamID) |

**Task\_T  🡪**

Columns in the table:

|  |
| --- |
| TaskID - PK |
| TaskName |
| Description |
| Reporter |
| DueDate |
| CreatedDate |
| TeamID - FK |

**Task\_Status\_LookUP\_T**  🡪

Columns in the table:

|  |
| --- |
| StatusID - PK |
| StatusDesc |

**Task\_Management\_T**  🡪

Columns in the table:

|  |
| --- |
| TaskID - FK |
| AssigneeID - FK |
| StatusID -FK |
| StartDate |
| UpdateDate |

**Timesheets\_T  🡪**

Columns in the table:

|  |
| --- |
| TimesheetsID - PK |
| EmpID |
| StartDate |
| Monday |
| Tuesday |
| Wednesday |
| Thursday |
| Friday |
| Saturday |
| Sunday |
| TotalHours |
| ApprovalManager - FK |
| Status |
| Notes |

**ShiftPlanning\_T  🡪**

Columns in the table:

|  |
| --- |
| ShiftID - PK |
| EmpID - FK |
| DeskNumber |
| TimesheetsID - FK |
| MondaySchedule |
| TuesdaySchedule |
| WednesdaySchedule |
| ThursdaySchedule |
| FridaySchedule |

**LeaveType\_T** 🡪

Columns in the table :

|  |
| --- |
| TypeID - PK |
| LeaveCategory |

**Leaves\_T**  🡪

Columns in the table:

|  |
| --- |
| EmpID - FK |
| TotalLeaves |
| RemainingLeaves |
| Status |

**Asset\_Status\_Lookup\_T** 🡪

Columns in the table:

|  |
| --- |
| AssetStatusID - PK |
| AssetStatusName |

**Asset\_Lookup\_T  🡪**

Columns in the table:

|  |
| --- |
| AssetTypeID - PK |
| AssetName |

**Emp\_AssetManagement\_T   🡪**

Columns in the table:

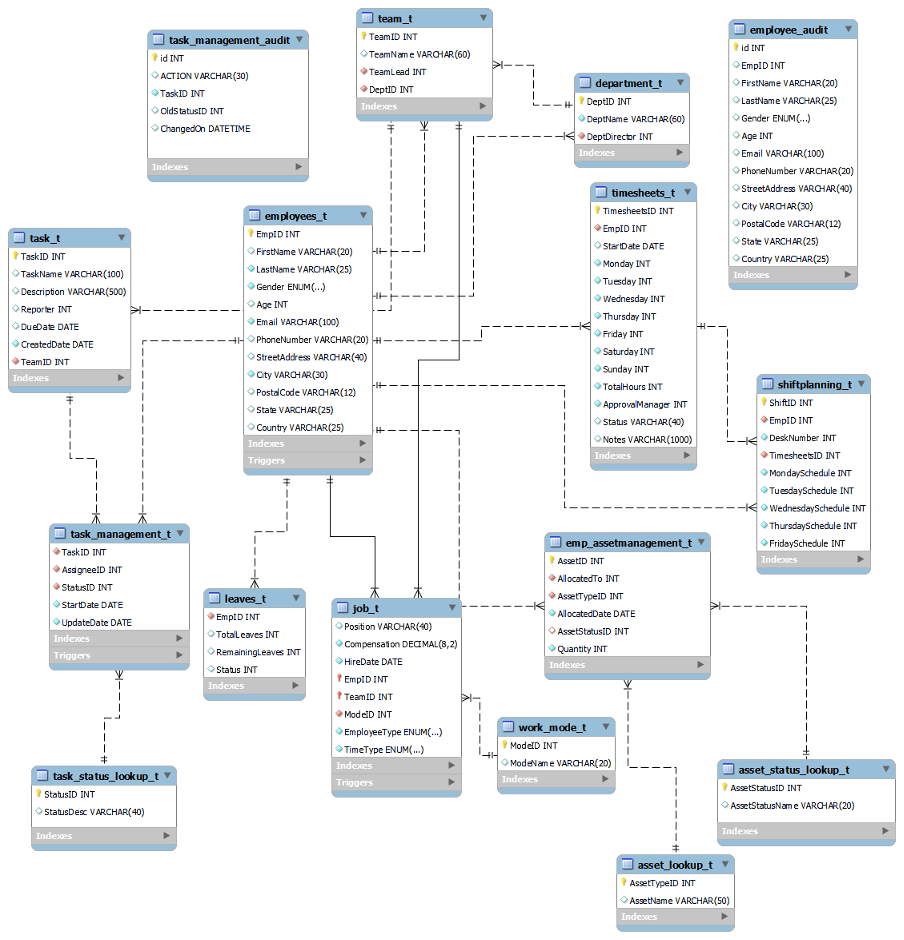
|  |
| --- |
| AssetID - PK |
| AllocatedTo - FK |
| AssetTypeID - FK |
| AllocatedDate |
| AssetStatusID - FK |
| Quantity |

**Entity Relationship Diagram**

ER Diagram stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities, attributes and relationships.

ER Diagrams contain different symbols that use rectangles to represent entities, ovals to define attributes and diamond shapes to represent relationships.

Following is the Entity Relationship Diagram (ERD) for our database:



For a table to serve as an integral part of a relational database, we link that table to at least one other table in the database, using a common column. There are three types of relationships that can exist between two entities.

An entity-relationship (ER) diagram can be created based on these three types, which are listed below:

1. **one-to-one relationship**: In relational database design, a one-to-one (1:1) relationship exists when zero or one instance of entity A can be associated with zero or one instance of entity B, and zero or one instance of entity B can be associated with zero or one instance of entity A. (abbreviated 1:1)
2. **one-to-many relationship**: (abbreviated 1:N) In relational database design, a one-to-many (1:N) relationship exists when, for one instance of entity A, there exists zero, one, or many instances of entity B; but for one instance of entity B, there exists zero or one instance of entity A.
3. **many-to-many relationship**: In relational database design, a many-to-many (M:N) relationship exists when, for one instance of entity A, there exists zero, one, or many instances of entity B; and for one instance of entity B, there exists zero, one, or many instances of entity A. (abbreviated M:N)

In our database pertaining to Home Office Operation Management system, there only exists *one-to-many* relationships. We decided to implement one-to-many relationships in our database as it is the most common relational database design and is at the heart of good design as it ensures referential integrity and efficiency.

For instance, to establish one-to-many relationships in our database, we used EmpID which is a primary key in the ‘Employees\_T’ table as referential foreign key in the ‘timesheets\_t’, ‘job\_t’, ‘shiftplanning\_t’, and ‘leaves\_t’ tables namely.

**Queries**

To retrieve the required data from our database, relate the information in different tables and perform various database operations, we have used several queries using clauses, functions, and programs.

We can calculate the number of employees from each team who are scheduled to work on a particular day of the week from the office using joins on ShiftPlanning, Employees and Job Tables.

We have computed a query using joins again to find the number of tasks which are either in-progress, assigned, completed or so on for each team. For this we have used task, task management, task status and team table.

To calculate the number of assets assigned to a department or team we are using join on asset management, department, and team table. We have also used order by clause here to get the team using the highest number of assets.

**Joins**

Data is kept in several tables that are connected to one another in relational databases like SQL Server, Oracle, MySQL, and others via a common key value. As a result, it is frequently necessary to combine data from two or more tables into one results table. The SQL JOIN clause in SQL Server makes this simple to do.

Based on logical connections between the tables, the SQL clause JOIN is used to query and access data from various tables.

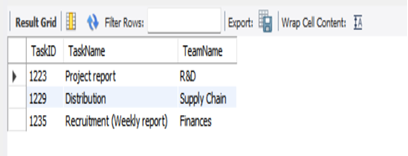
In other words, JOINS specify how SQL Server should choose entries from a different table using information from a different source.

SQL Server supports many kinds of different joins including INNER JOIN, SELF JOIN, CROSS JOIN, and OUTER JOIN. In fact, each join type defines the way two tables are related in a query. OUTER JOINS can further be divided into LEFT OUTER JOINS, RIGHT OUTER JOINS, and FULL OUTER JOINS.

* SQL INNER JOIN creates a result table by combining rows that have matching values in two or more tables.
* SQL LEFT OUTER JOIN includes in a result table unmatched rows from the table that is specified before the LEFT OUTER JOIN clause.
* SQL RIGHT OUTER JOIN creates a result table and includes in it all the records from the right table and only matching rows from the left table.
* SQL SELF JOIN joins the table to itself and allows comparing rows within the same table.
* SQL CROSS JOIN creates a result table containing paired combination of each row of the first table with each row of the second table.

The Joins Used for the project are:

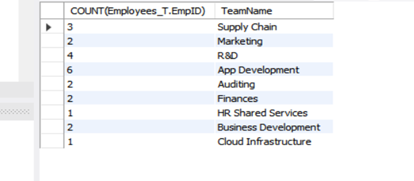
* To pull up all the in-progress tasks for each team (teamID will be passed) between Task\_T, task mgmt, task status and team.



In this we are joining the Task, Task Management, Task Status and Team tables which help the employer to know/check the task status assigned to each team by entering the team Id.

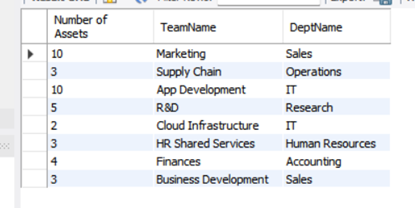
* How many employees from each team are scheduled to work on Monday (day) a week from the office? (shift planning, employee, Job\_T)

This basic query involves joining Shift Planning, Employee and Job tables which provides us the information of the employees present on a particular day of the week from a team.



* Number of assets assigned to a particular department or team

This join gives us the insights about no of assets assigned to a team or department which help the management to keep track of the assets and to analyze if more assets need to be added to the inventory or to be repaired by the asset status it shows in the result.



**Triggers**

A trigger is a stored program invoked automatically in response to an event such as insert, update, or delete that occurs in the associated table. For example, you can define a trigger that is invoked automatically before a new row is inserted into a table. MySQL supports triggers that are invoked in response to the INSERT, UPDATE or DELETE event.

The SQL standard defines two types of triggers: row-level triggers and statement-level triggers.

A row-level trigger is activated for each row that is inserted, updated, or deleted.  For example, if a table has 100 rows inserted, updated, or deleted, the trigger is automatically invoked 100 times for the 100 rows affected.

A statement-level trigger is executed once for each transaction regardless of how many rows are inserted, updated, or deleted.

MySQL supports only row-level triggers. It doesn’t support statement-level triggers.

**Advantages of Triggers:**

* Integrity of data − With the help of MySQL trigger we can check the integrity of data in the table. In other words, MySQL triggers are the alternative way to check the integrity of data.
* Useful for catching errors − MySQL triggers can catch errors in business logic in the database layer.
* Alternative way to run scheduled tasks − Actually by using MySQL triggers we do not have to wait to run the scheduled tasks because the triggers are invoked automatically ‘before’ or ‘after’ a modification is done to the data in the table.
* Auditing − Actually MySQL triggers are very useful for the purpose of auditing the changes made in the table.
* Prevention of invalid transactions− MySQL triggers are very useful in the prevention of invalid transactions.
* Logging of event− MySQL triggers can log an event and can also store the information on the access of table.

**The Triggers Used for the project are:**

**BEFORE INSERT trigger**: To check if an employee is eligible for the year end leaves by checking the total number of remaining leaves and then updating the status in the table.   
If it’s more than five then they are eligible to take long term leaves during the year end.

**AFTER INSERT trigger**: In this trigger, after a new employee is added to the organisation, their records will be updated as per their profile in the respective tables.

**BEFORE UPDATE trigger**: This trigger stores the previous status of task, after a task is moved from one state to another and before updating the task management table.

**AFTER DELETE trigger**: This trigger will be prompted to remove employee details from jobs, asset management and leaves table when an employee leaves the organisation.

**Stored Procedures**

A collection of SQL statements is referred to as a stored procedure in SQL and is kept together in a database. It can execute one or more DML operations on the database and return value, if any, depending on the statements in the procedure and the arguments you pass. As a result, it enables reuse by allowing you to pass the same statements more than once.

Stored processes offer several significant advantages, including:

* Reusable: As previously indicated, running a stored procedure allows numerous users and programs to use and reuse it.
* Simple to change: Using the ALTER TABLE command, you can easily change the statements in a stored procedure whenever you need to.
* Security: By preventing users from directly accessing the table, stored procedures let you improve the security of an application or database.
* Low network traffic: The server merely transmits the procedure name, lowering network bandwidth, rather than the entire query.
* Performance is improved since a plan for the stored procedure is established on the first use and stored in the buffer pool for speedy execution the next time.

To serve the purpose of data integrity and reusability we have 2 procedures.

1. Procedure to **update** the leaves table with the remaining number of leaves. This procedure is to update the leaves table once the leave is granted to a particular employee, based on the employee id
2. Procedure to **automate** the insertion of values into the job table that tracks the position, compensation, team details of each employee. This procedure helps the employer to keep track and saves the time and resources in inserting or updating the latest employee records in the job table.

**Stored Functions**

A stored function is a special kind stored program that returns a single value. Typically, you use stored functions to encapsulate common formulas or business rules that are reusable among SQL statements or stored programs.

Different from a stored procedure, you can use a stored function in SQL statements wherever an expression is used. This helps improve the readability and maintainability of the procedural code.

* To create a stored function, you use the CREATE FUNCTION statement.
* Specify the name of the stored function that you want to create after CREATE FUNCTION keywords.
* List all parameters of the stored function inside the parentheses followed by the function name. By default, all parameters are the IN parameters. You cannot specify IN, OUT or INOUT modifiers to parameters
* Specify the data type of the return value in the RETURNS statement, which can be any valid MySQL data types.
* Specify if a function is deterministic or not using the DETERMINISTIC keyword.

A deterministic function always returns the same result for the same input parameters whereas a non-deterministic function returns different results for the same input parameters.

If you don’t use DETERMINISTIC or NOT DETERMINISTIC, MySQL uses the NOT DETERMINISTIC option by default.

Write the code in the body of the stored function in the BEGIN END block. Inside the body section, you need to specify at least one RETURN statement. The RETURN statement returns a value to the calling programs. Whenever the RETURN statement is reached, the execution of the stored function is terminated immediately.

To serve the purpose of data integrity and reusability we have 2 Functions.

A **function** to determine the number of days taken to complete a task.

* This function helps in determining the total number of days taken to complete a particular task given to an employee and the team id of the employee is assigned to.

A **function** to calculate the remaining leaves of an employee.

* The function helps in calculating the remaining leaves of a particular employee and let the employer decide whether to grant any future leaves for the employee.

**Conclusion and Future Scope**

The target audience for our product are the companies or businesses that are adapting to the remote working / hybrid working environment.

The home office operation management system is designed using MySQL addressing the needs of the current shift towards the hybrid/ remote working culture.

Our database system will allow the organizations to analyze the data on a real-time basis for the future benefits of the organization and track work efficiency reports through simple queries.

Our clients can use operational reporting to track project status, monitor goals vs actual progress, and compare ideal, available, actual & billed hours. Business managers can analyze project overheads, work distribution across employees, budgeting and resource allocation and monitor the timeline compliance.

We have identified a few essential improvements to pave the way for continuous innovation

Going forward there are two aspects on which our product can improvise which are access control and cloud back up.

According to a study by Gartner, 75% of databases will be migrated to the cloud by the end of 2022. Cloud database management systems can store data in any of the cloud storage tiers in different types of data models, including relational, non-relational, and more using AWS virtual platform. During the pandemic, many organizations offered remote working opportunities, and cloud-based systems allowed employees to access their work-related datasets at anytime from anywhere.

The other aspect that extremely integral to any database management system is data security. The recent rise of cyber threats and data breaches has inspired an even greater emphasis on data security in database management systems, especially those in the cloud.

To ensure better data integrity and security we look forward to develop access control mechanism such that the access of database lies with the concerned team heads and integrity of important information is maintained.

**REFERENCES**

* <https://www.lucidchart.com/pages/er-diagrams>
* <https://intellicus.com/operational-reporting-and-why-businesses-need-it/>
* <https://www.mckinsey.com/capabilities/operations/our-insights/new-operations-management-systems-for-a-digital-world>
* <https://www.tutorialspoint.com/What-are-the-advantages-disadvantages-and-restrictions-of-using-MySQL-triggers>
* <https://www.mysqltutorial.org/mysql-triggers/>
* <https://www.mysqltutorial.org/mysql-stored-procedure-tutorial.aspx>
* <https://www.investopedia.com/terms/o/operations-management.asp>
* <https://apploye.com/virtual-office-management/>