Aashir Shaikh - A106

Satyam Karde - A116

Lakshya Sharma - A122

**DBMS Mini Project**

**Introduction**

A Portfolio Management System is a software application designed to help investors, traders, and fund managers monitor and manage their investment portfolios. It allows them to track their investment holdings, monitor their performance, and make informed decisions about buying, selling, and managing their assets. Our DBMS project aims to build a Portfolio Management System that provides a centralised platform to manage all investments made by an individual or an organisation.

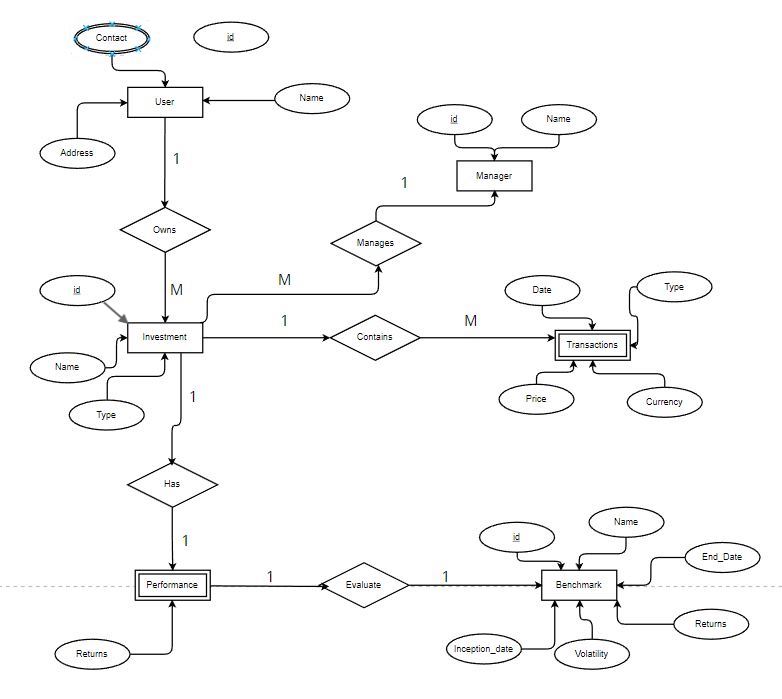
Management of multiple investments, financial instruments, and portfolios can be overwhelming for investors, making it difficult for them to keep track of their investment’s performance and overall portfolio value. This is where a Portfolio Management System comes into play, providing investors with a comprehensive platform to manage their investments, portfolios, and financial instruments in one place.

The portfolio management system (PMS) offers several solutions to investors and portfolio managers to efficiently manage their investment portfolios. The system consists of six main entities: User, Manager, Investment, Transaction, Performance, and Benchmark.

The User entity represents the investors who can create an account on the system, view their portfolio, and make investment decisions. The Manager entity represents the portfolio managers who can manage the investments of multiple users. The Investment entity represents the investment options available to the users, such as stocks, mutual funds, and bonds. The Transaction entity represents the transactions made by the users, such as buying or selling investments. The Performance entity represents the performance of the investments, and the Benchmark entity represents the benchmark against which the performance of the investments is compared.

The PMS offers solutions such as investment tracking, portfolio optimization, risk management, and performance analysis. Users can track their investments, view their portfolio diversification, and optimise their portfolio for better returns. Managers can manage the investments of multiple users, monitor the performance of their portfolios, and generate reports. The PMS also offers risk management solutions such as tracking risk exposure and managing risk factors. The performance analysis solution allows users and managers to compare the performance of their investments against the benchmark, and make data-driven investment decisions.

**ER diagram**

****

Entities:- User, Investment, Performance (weak entity) , Benchmark, Transaction(Weak entity)

Relationships:-

i) The one-to-many relationship between User and Investment implies that a user can own multiple investments, while each investment can only be owned by one user. This relationship enables the portfolio management system to keep track of the investments owned by each user, which helps in monitoring the performance of the portfolio over time.

ii) The one-to-one relationship between Investment and Performance implies that each investment has one and only one performance record associated with it. This relationship allows the portfolio management system to track the performance of each investment over time and evaluate its return on investment.

iii) The one-to-one relationship between Performance and Benchmark implies that each performance is evaluated against one benchmark. This relationship allows the portfolio management system to compare the performance of each investment against a relevant benchmark and determine its relative performance.

iv) The one-to-many relationship between Investment and Transaction implies that each investment can have multiple transactions associated with it. This relationship allows the portfolio management system to keep track of all the transactions related to each investment, including the purchase price, sale price, and any other relevant transaction details. This information is crucial for monitoring the performance of each investment and evaluating the overall portfolio performance.

**Conversion to relational schema**

Manager (id: int, name: string, benchmarks: string, foreign key)

User (id: int, name: string, contact: string, location: string)

Investment (name: string, type: string, sector: string, price: int, date\_of\_purchase: date)

Performance (investment\_name: string, portfolio: string, returns: int, benchmark\_returns: int, foreign key)

Benchmark (name: string, inception\_date: date, end\_date: date, benchmark\_returns: int, volatility: int)

Transaction (date: date, type: string, investment\_name: string, price: int, currency: string)

**Normalization**

We converted our schema from its initial 1NF form to the highest possible normal form which is 3NF after removing partial dependencies, transitive dependency and has atomic values for each entity.

To convert a relation from 1NF to 3NF, we identified and removed any transitive dependencies. This can be done by decomposing the original relation into smaller relations. We first identified the functional dependencies in the original relation, and then we created new relations for each subset of attributes that are functionally dependent on a candidate key. In other words, we created new relations that satisfy the rules of 2NF.

Next, we analysed the dependencies between attributes in the new relations to identify any transitive dependencies. As we found any transitive dependencies, we created new relations for each subset of attributes that are functionally dependent on another non-candidate key attribute.

The process of decomposition continued until each resulting relation satisfied the rules of 3NF. The resulting relations are then connected by creating foreign key relationships between them. The goal of this process is to normalize our database, eliminate data redundancy and ensure data integrity by avoiding data anomalies that can occur due to transitive dependencies.-

User (ID: int, Name: varchar, Contact: varchar, Location: varchar)

Investment (ID: int, Name: varchar, Type\_ID: int, Sector\_Ind\_ID: int, Price: int, Date\_of\_purchase: date)

Investment\_Type (ID: int, Name: varchar)

Sector\_Industry (ID: int, Name: varchar)

Transaction (ID: int, Date: date, Type\_ID: int, Investment\_ID: int, Price: int, Currency\_ID: int)

Currency (ID: int, Name: varchar)

Performance (ID: int, Investment\_ID: int, Portfolio: varchar, Returns: int)

Benchmark (ID: int, Name: varchar, Inception\_date: date, End\_date: date, Return: int, Volatility\_ID: int)

Benchmark\_Volatility (ID: int, Benchmark\_ID: int, Date: date, Volatility: int)

**All Queries for the Database**

1. The average price of all investments:

SELECT AVG(Price) AS AveragePrice

FROM Investment;

1. The total price of all investments:

SELECT SUM(Price) AS TotalPrice

FROM Investment;

1. The ratio of investment in each sector:

SELECT si.Name ,SUM(i.Price) AS Total\_Investment,

SUM(i.Price) / (SELECT SUM(Price) FROM Investment) AS Investment\_Ratio

FROM Investment i

JOIN Sector\_Industry si ON i.Sector\_Ind\_ID = si.ID

GROUP BY si.Name;

1. The average price of investments for each type:

SELECT Type, AVG(Price) AS AveragePrice

FROM Investment

GROUP BY Type;

1. Retrieve the benchmark return for a specific date range:

SELECT Benchmark.Name, SUM(Benchmark.Return) as Total\_Return

FROM Benchmark

WHERE Benchmark.Inception\_date BETWEEN '2022-01-01' AND '2022-12-31'

GROUP BY Benchmark.Name;

1. Retrieve all investments in a specific sector:

SELECT Investment.ID, Investment.Name, Sector\_Industry.Name

FROM Investment

INNER JOIN Sector\_Industry ON Investment.Sector\_Ind\_ID = Sector\_Industry.ID

WHERE Sector\_Industry.Name = ‘Any\_industry’;

1. Calculate the total return on investment for a specific portfolio:

SELECT Portfolio, SUM(Returns) as Total\_Returns

FROM Performance

WHERE Investment\_ID IN (

SELECT ID

FROM Investment

WHERE Investment.Name LIKE '%TCS%'

)

GROUP BY Portfolio;

1. Calculate the total value of a portfolio:

SELECT Portfolio, SUM(Price) as Total\_Value

FROM Investment

WHERE Investment.ID IN (

SELECT Investment\_ID

FROM Performance

WHERE Portfolio = 'My Portfolio'

)

GROUP BY Portfolio;

1. Retrieve the Top performing investments in a specific sector:

SELECT Investment.Name, Performance.Returns

FROM Investment

INNER JOIN Performance ON Investment.ID = Performance.Investment\_ID

WHERE Investment.Sector\_Ind\_ID = (

SELECT ID

FROM Sector\_Industry

WHERE Sector\_Industry.Name = 'Information Technology'

)

ORDER BY Performance.Returns DESC;

1. Rollback transactions:

SAVEPOINT A;

INSERT INTO Investment (Name, Type\_ID, Sector\_Ind\_ID, Price, Date\_of\_purchase)

VALUES ('ABC Corp', 1, 2, 1000, '2022-01-01');

ROLLBACK TO A;

SELECT \* FROM Investment;

1. Retrieve the managers who have invested in a specific sector/industry:

SELECT DISTINCT Manager.Name

FROM Manager

JOIN Investment ON Manager.ID = Investment.ID

JOIN Sector\_Industry ON Investment.Sector\_Ind\_ID = Sector\_Industry.ID

WHERE Sector\_Industry.Name = 'Banking';

1. Retrieve the total value of all transactions in a specific currency:

SELECT SUM(Transaction.Price)

FROM Transaction

JOIN Currency ON Transaction.Currency\_ID = Currency.ID

WHERE Currency.Name = 'USD';

1. Top-performing investments in each sector, based on their return:

SELECT si.Name AS Sector, i.Name AS Investment, p.Returns

FROM Performance p

INNER JOIN Investment i ON p.Investment\_ID = i.ID

INNER JOIN Sector\_Industry si ON i.Sector\_Ind\_ID = si.ID

WHERE p.Returns = (SELECT MAX(Returns) FROM Performance WHERE Investment\_ID = i.ID)

ORDER BY si.Name;

1. Maximum benchmark return for each investment:

SELECT i.Name, MAX(b.Return) AS Max\_Return

FROM Investment i

JOIN Benchmark b ON b.ID = i.ID

GROUP BY i.Name;

1. Investments whose total return is greater than the average return of all investments:

SELECT i.Name, SUM(p.Returns) AS Total\_Returns

FROM Investment i

JOIN Performance p ON p.Investment\_ID = i.ID

GROUP BY i.Name

HAVING SUM(p.Returns) > (SELECT AVG(Returns) FROM Performance);