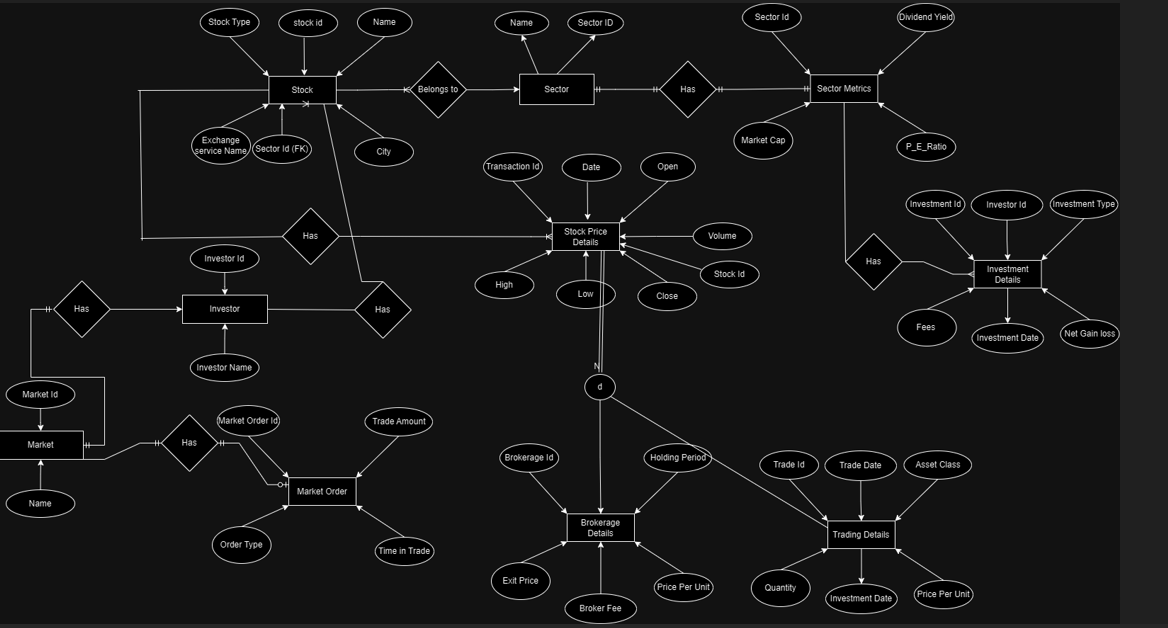
**Name: Shahzaib Ali**

**Roll Number: 22i0576  
Section: AI-B**

**Report**

**DDL and ERD Explanation:**

****

**Sector**:

* Sector contains sectors within which stocks can belong.
* Columns:
  + Sector\_ID: Primary key, uniquely identifying each sector.
  + Name: Name of the sector, must be unique.
* In the ERD, this entity is represented by the Sector entity and is linked to Stock and Sector\_Metrics.

**Stock**:

* Stock holds information on different stocks.
* Columns:
  + Stock\_ID: Primary key for stocks.
  + Name, City, Exchange\_Service\_Name: Basic details about each stock.
  + Sector\_ID: Foreign key, links each stock to a sector.
  + Stock\_Type: Enumerated type defining the category of the stock (e.g., 'dividend', 'blue chip').
* This table is connected to Sector through the foreign key, linking each stock to a specific sector.

**Stock\_Price\_History**:

* Stores the historical price data for each stock.
* Columns:
  + Stock\_History\_ID: Primary key for price history entries.
  + Stock\_ID: Foreign key linking to the Stock table.
  + Date, Open, High, Low, Close, Volume: Details of stock prices on a specific date.
* This corresponds to the Stock Price History entity in the ERD, showing the daily price history of each stock.

**Stock\_Metrics**:

* Holds specific metrics for each stock, such as dividend and historical price records.
* Columns:
  + Stock\_Metrics\_ID: Primary key for stock metrics.
  + Stock\_ID: Foreign key linking to Stock.
  + Dividend, All\_Time\_High, 52\_Week\_High, 52\_Week\_Low, Insider\_Ownership: Metrics that give additional insight into the stock.
* This table is represented in the ERD as part of the Stock Metrics entity and is connected to Stock.

**Sector\_Metrics**:

* Stores metrics specific to each sector.
* Columns:
  + Sector\_Metrics\_ID: Primary key for sector metrics.
  + Sector\_ID: Foreign key linking to Sector.
  + Dividend\_Yield, Market\_Cap, P\_E\_Ratio: Key financial metrics for a sector.
* In the ERD, this corresponds to the Sector Metrics entity, which provides overall metrics for each sector.

**Investor**:

* Represents individual investors in the database.
* Columns:
  + Investor\_ID: Primary key for each investor.
  + Name: The investor's name.
* This table corresponds to the Investor entity in the ERD.

**Investment Table**:

* Details individual investments made by investors.
* Columns:
  + Investment\_ID: Primary key for each investment.
  + Investor\_ID: Foreign key linking to the Investor table.
  + Investment\_Type: Type of investment (e.g., 'Short', 'Long').
  + Fees, Net\_Gain\_Loss, Investment\_Date: Financial details related to the investment.
* This corresponds to the Investment entity in the ERD, representing specific investments by each investor.

**Trade**:

* Stores details of each trade transaction.
* Columns:
  + Trade\_ID: Primary key for each trade.
  + Trade\_Date, Asset\_Class, Quantity, Price\_Per\_Unit: Basic details of the trade, such as the date, asset type, and quantity.
* This table represents the Trade entity in the ERD, detailing individual trade activities.

**Trade\_Stock\_Relation**:

* Associates trades with specific stocks.
* Columns:
  + Trade\_Stock\_ID: Primary key for trade-stock relationships.
  + Trade\_ID: Foreign key linking to Trade.
  + Stock\_ID: Foreign key linking to Stock.
* This table creates a many-to-many relationship between Trade and Stock, representing trades that involve specific stocks.

**Market Table**:

* Represents different markets where trades can occur.
* Columns:
  + Market\_ID: Primary key for each market.
  + Market: Name of the market, must be unique.
* This table corresponds to the Market entity in the ERD.

**Market\_Order**:

* Represents orders placed in the market.
* Columns:
  + Market\_Order\_ID: Primary key for each order.
  + Market\_ID: Foreign key linking to Market.
  + Trade\_Amount, Order\_Type, Time\_in\_Trade: Details related to the market order, such as order type (limit or market).
* This entity corresponds to the Market Order entity in the ERD, showing individual market orders placed by investors.

**Brokerage**:

* Holds details about the brokerage services.
* Columns:
  + Brokerage\_ID: Primary key for each brokerage record.
  + Broker\_Fee, Holding\_Period, Exit\_Price, Price\_Per\_Unit: Financial details related to brokerage services.
* This table represents the Brokerage entity in the ERD, detailing broker fees and related information.

**Challenges Faced:**

**1. Ensuring Data Integrity Across Relationships**

* **Challenge**: With multiple interdependent tables (e.g., Stock, Sector, Investment, and Trade), maintaining data consistency and enforcing relationships between entities can become complex.
* **Solution**: Using **foreign key constraints** ensures that records in related tables (e.g., Stock\_ID in Stock\_Price\_History referencing Stock\_ID in Stock) are consistent. These constraints enforce referential integrity by preventing orphaned records and ensuring that all relationships align with business rules.

**2. Handling Many-to-Many Relationships**

* **Challenge**: In cases like linking stocks to trades, a many-to-many relationship exists where each trade can involve multiple stocks, and each stock can be involved in multiple trades.
* **Solution**: This is addressed by creating a **junction table** (Trade\_Stock\_Relation) to break down the many-to-many relationship into two one-to-many relationships. This table efficiently manages associations between trades and stocks by storing the Trade\_ID and Stock\_ID for each combination.

**3. Defining Appropriate Data Types for Financial Data**

* **Challenge**: Financial data, like stock prices and volumes, require precise data types to avoid rounding errors or data inaccuracies. Additionally, large data volumes need to be handled correctly to avoid storage issues.
* **Solution**: Decimal data types (e.g., DECIMAL(10, 2) for prices and BIGINT for Volume) were chosen to provide both precision and the ability to store large values. This ensures accurate calculations for financial records.

**4. Dealing with Data Duplication**

* **Challenge**: Redundant data (e.g., repeating Sector or Investor names) can increase storage needs and make it harder to manage updates.
* **Solution**: The schema uses **normalization**, where tables like Investor, Sector, and Market are stored separately with unique keys. This reduces data redundancy, makes updates easier, and prevents inconsistencies.

**Differences Between ERD and schema:**

| **Aspect** | **EERD Image** | **Schema Diagram** |
| --- | --- | --- |
| **Level of Detail** | High-level conceptual | Implementation details with data types |
| **Entity Representation** | Entities, relationships, attributes | Tables and columns with data types |
| **Specialization** | Shows hierarchies visually | Implemented using tables or ENUM |
| **Relationships** | Visual lines with cardinality | Foreign keys with constraints |
| **Constraints** | Disjoint/overlapping visually indicated | Enforced with primary, foreign keys, and constraints |
| **Aggregation** | Represented visually | Not directly supported, foreign keys used instead |
| **Data Types** | Not shown | Explicitly defined (e.g., DECIMAL, VARCHAR) |
| **Derived Attributes** | Conceptual | Needs explicit implementation or calculation |