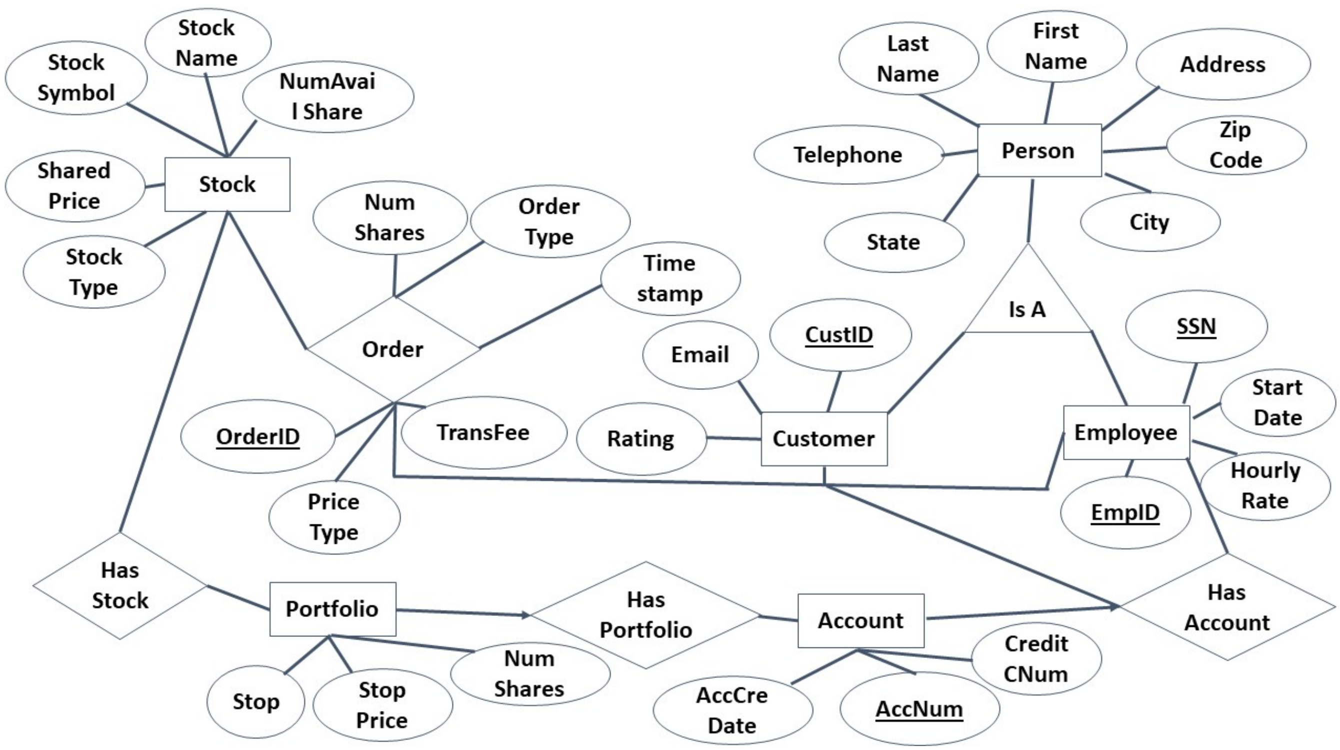
Final Assignment

Max Points: 100

1. **Entity-Relationship (E-R) Diagram of the complete database scheme.**



2. **Lucid description of the relational database scheme for your online stock trading database, based on your ER diagram. Include a discussion of the reasoning behind your design decisions (e.g., how did you normalize?).**

<u>Solution</u> –

The entity 'Person' acts as a general class that encompasses both Customers and Employees, streamlining data by consolidating common characteristics. This design ensures uniqueness in identification through specific primary keys, CusId for Customers and EmpId for Employees. The account management system mandates a one-to-one correspondence between customers and accounts, with each customer possessing at least one account, yet allowing for multiple accounts per customer. This is visually represented in the schema by bold arrows and lines indicating participation and key constraints in relationships involving accounts and customers.

An account is identified by a unique primary key and is linked to a stock portfolio, which may vary in composition without any restrictions. The relationship between Portfolio and Account is highlighted by a bold arrow, signifying a key constraint where each portfolio is linked to only one account.

Stocks within this system can be a part of none, one, or several portfolios, showcasing flexibility in allocation. Each stock is uniquely identified by a symbol, and both customers and employees can place orders for stocks without any obligation to do so. The Order entity captures these transactions, with each Order identified by a unique OrderId.

Distinct sets of credentials are maintained for customers and employees, emphasizing the importance of unique usernames as primary login keys. Entities like Transact and ConditionalPriceHistory are linked to Orders through foreign keys, maintaining referential integrity. Each transaction in Transact is uniquely identified, while Conditional Price History uses a composite key of OrderId, PriceType, and Timestamp to distinctively identify each entry.

StockHistory, another entity, records changes in stock prices and is linked to specific stocks through the StockSymbol as part of its primary key, along with timestamp and share price details. This allows tracking of price fluctuations over time. Each StockHistory entry can include numerous Stocks, offering a comprehensive view of stock performance.

3. **A list of all functional dependencies in the relational database scheme.**

<u>Solution</u> -

i. ConditionalPriceHistory

{OrderId, PriceType, TimeStamp} → {CurSharePrice, StopPrice}

ii. Portfolio

{AccNum, StockSymbol} → {NumShares, Stop, StopPrice}

iii. StockPriceHistory

{StockSymbol, TimeStamp} → {SharePrice}

iv. Transact

{Id} → {OrderId, Transfee, TimeStamp}

v. Customer

{CusId} → {LastName, FirstName, Address, City, State, ZipCode, Telephone, Email, Rating, Email}

vi. Employee

{EmpId} → {SSN, LastName, FirstName, Address, City, State, ZipCode, Telephone, Start Date, HourlyRate, Position}

{SSN} → {LastName, FirstName, Address, City, State, ZipCode, Telephone, StartDate, HourlyRate, Position, EmpId}

vii. Stock

{StockSymbol} → {StockName, StockType, SharePrice, NumAvailShares}

{StockName} → {StockType, SharePrice, NumAvailShares}

viii. Account

{AccNum} → {AccCreDate, CreditNum, CusId}

ix. Order

{OrderId} → {StockSymbol, OrderType, NumShares, CusAccNum, TimeStamp, PriceType, StopPrice, StopDiff, CurSharePrice, EmpId, Recorded, Completed}

{StockSymbol, TimeStamp, CusAccNum, EmpId} → {OrderType, NumShares, PriceType, StopPrice, StopDiff, CurSharePrice, Recorded, Completed}

4. Description of integrity constraints including referential integrity.

Integrity constraints are essential rules that ensure the maintenance of high-quality data. They play a critical role in ensuring that operations like data insertion and updating preserve the data's accuracy, thereby protecting the database from unintentional corruption.

The types of integrity constraints include:

Domain Constraints: These define acceptable value ranges for attributes, based on their data types such as string, character, integer, time, date, currency, etc. Each attribute in a database is associated with a specific domain datatype, ensuring that its value falls within the defined range. For example, in the STOCK table, attributes like name, type, and symbol are varchar, while SharePrice is a float integer. All tables in the database adhere to these domain constraints.

Entity Integrity Constraints: This constraint stipulates that the primary key of a table cannot be null. This is crucial because primary keys uniquely identify each row in a table, and a null primary key would render this identification impossible. In the database under discussion, this constraint is met as no primary keys in any table are null.

Referential Integrity Constraints: These constraints are established between two tables, ensuring that every foreign key value in one table (Table 1) either matches a primary key value in another table (Table 2) or is null. In this database, referential integrity is maintained as each foreign key value correlates with a primary key in its related table, with none of the foreign keys being null.

Key Constraints: Keys are used to uniquely identify entities within an entity set of a database. While there can be multiple keys for an entity set, only one is designated as the primary key. This key must be unique and not null. In this database, the key constraints are observed, as each entity in the tables has one primary key, exemplified by the stockSymbol in the STOCK table being the primary key.