Database design document
Group – 5
Topic : Stock market Portfolio Management System

Description of business problems being addressed by our data base:

Database Management: The primary problem is to build and maintain a database for an online stock trading platform. This includes managing and organizing data related to companies, their listed stocks, trader information, orders, and trade history.

Stock Tracking: The platform aims to provide a solution for traders to easily keep track of companies and the stocks they have available for trading. This involves real-time monitoring of stock prices, trading volumes, and other relevant data.

User Registration and Trading: Enabling traders to register on the platform and perform stock trading activities, including buying and selling stocks. This involves user authentication, order placement, and trade execution.

Stock Listing for Companies: Providing a mechanism for companies to list their stocks on the platform. Companies should be able to submit details of their stocks, making it easier for them to reach a wider audience of potential investors and traders.

Portfolio Management: Traders should have the capability to manage their portfolios, which are collections of stocks they own. This involves tracking the stocks they hold, their current value, and overall portfolio performance.

Convenience for Traders: The platform aims to make it easier for traders to buy and sell stocks of the companies they are interested in at their own convenience. This addresses the need for a user-friendly and accessible trading platform.

Below are the Entities of Final ER-Diagram

Person
Trader
Trader Login
Employee
Employee Login
Portfolio
Portfolio_stock
Company
Order
Order_stock
Stock
Wishlist
Wishlist_stock

Brief Description identifying the changes made to the Initial ERD:

High-Level Overview:

The final ER diagram has a more detailed structure with a greater number of entities and relationships compared to the Initial ER diagram. This indicates a more comprehensive data model. The initial ER diagram is a streamlined version of the final ERD, having fewer entities and attributes.

Attributes:

Trader: The 'license_number' attribute from the first diagram is not present in the second.

Employee: The second ER diagram retains only the 'date_of_joining' attribute, eliminating attributes such as 'mobile_no', 'email', 'address', 'city', 'state', 'zipcode', etc., that were present in the first diagram.

Portfolio: Attributes are consistent across both diagrams.

Portfolio_stock: In the second ER diagram, 'purchase_price' has been renamed to 'average_price'.

Stock: 'stock_category_ID' is an attribute in the first diagram but is absent in the second.

Order: Both diagrams maintain the same attributes.

Order_stock: Both diagrams maintain the same attributes.

Company: Both diagrams maintain the same attributes.

Relationships:

The relationships in the first diagram, particularly those involving the 'Person' entity, indicate a more comprehensive understanding of how the various entities are interconnected.

Advantages of Final ERD over Initial ERD:

Comprehensiveness: The first diagram offers a more comprehensive view of the system, capturing a greater number of entities and their relationships.

Flexibility: By including a 'Person' entity, the first diagram can easily extend to accommodate different roles beyond just 'Trader' and 'Employee'.

Security Concerns: The first diagram differentiates between logins for traders and employees, which could be crucial for implementing role-based access controls.

User Experience: Features like 'Wishlist' indicate a more user-centered approach in the first diagram, allowing traders to save and track stocks they're interested in.

Richness of Data: The first diagram captures more attributes for entities, which can offer more detailed insights and functionality. For instance, having attributes like 'mobile_no', 'email', etc., for employees can be crucial for communication purposes.

Relationships and Key Design Decisions:

Person Entity:

Trader has a relationship with Person. Each trader is a person. This is a one-to-one relationship where the primary key person_ID in the Person entity relates to a trader.

Employee has a relationship with Person. Each employee is a person. This is a one-to-one relationship where the primary key person_ID in the Person entity relates to an employee.

Login Entity:

Trader_Login has a relationship with Login. Each trader login belongs to a specific login. This is a one-to-one relationship.

Employee_Login has a relationship with Login. Each employee login belongs to a specific login. This is a one-to-one relationship.

Portfolio Entity:

Trader has a relationship with Portfolio. A trader can have one portfolio. This is a one-to-one relationship where the primary key trader_ID in the Trader entity relates to a portfolio.

Portfolio_Stock Entity:

Portfolio has a relationship with Portfolio_Stock. A portfolio can have multiple stocks. This is a one-to-many relationship where the primary key portfolio_ID in the Portfolio entity relates to multiple portfolio stocks.

Stock has a relationship with Portfolio_Stock. A stock can be in multiple portfolios. This is a many-to-many relationship facilitated by the Portfolio_Stock entity.

Stock Entity:

Company has a relationship with Stock. A company can have multiple stocks. This is a one-to-many relationship where the primary key company_ID in the Company entity relates to multiple stocks.

Wishlist Entity:

Trader has a relationship with Wishlist. A trader can have one wishlist. This is a one-to-one relationship where the primary key trader_ID in the Trader entity relates to a wishlist.

Wishlist_Stock Entity:

Wishlist has a relationship with Wishlist_Stock. A wishlist can have multiple stocks. This is a one-to-many relationship where the primary key wishlist_ID in the Wishlist entity relates to multiple wishlist stocks.

Stock has a relationship with Wishlist_Stock. A stock can be in multiple wishlists. This is a many-to-many relationship facilitated by the Wishlist_Stock entity.

Order Entity:

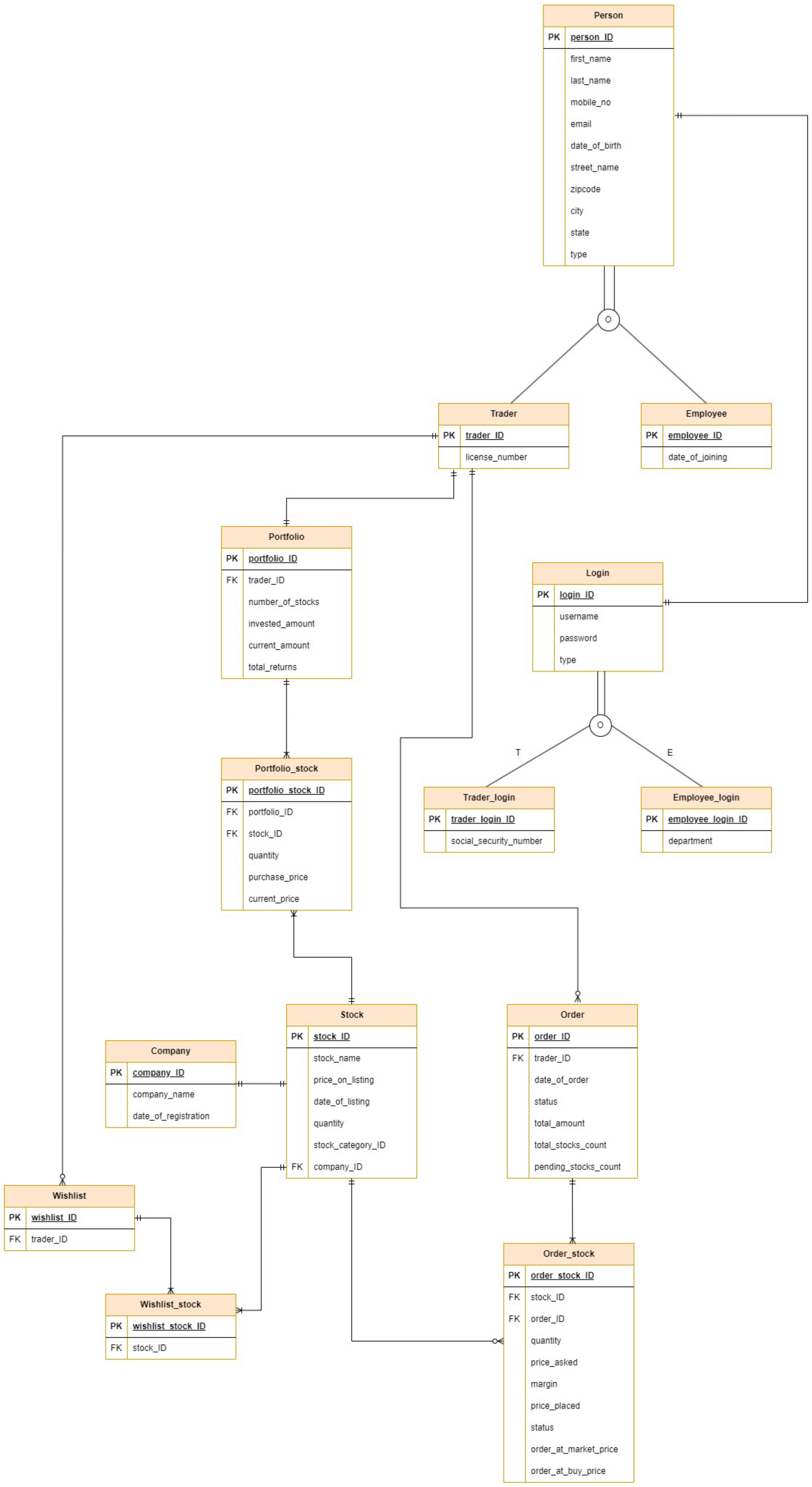
Trader has a relationship with Order. A trader can place multiple orders. This is a one-to-many relationship where the primary key trader_ID in the Trader entity relates to multiple orders.

Order_Stock Entity:

Order has a relationship with Order_Stock. An order can have multiple stocks. This is a one-to-many relationship where the primary key order_ID in the Order entity relates to multiple order stocks.

Stock has a relationship with Order_Stock. A stock can be in multiple orders. This is a many-to-many relationship facilitated by the Order_Stock entity.

FINAL ER – DIAGRAM


Person
PK person_ID
first_name
last_name
mobile_no
email
date_of_birth
street_name
zipcode
city
state
type
O
Trader
PK trader_ID
license_number
Employee
PK employee_ID
date_of_joining
Portfolio
PK portfolio_ID
FK trader_ID
number_of_stocks
invested_amount
current_amount
total_returns
Login
PK login_ID
username
password
type
O
T
E
Trader_login
PK trader_login_ID
social_security_number
Employee_login
PK employee_login_ID
department
Portfolio_stock
PK portfolio_stock_ID
FK portfolio_ID
FK stock_ID
quantity
purchase_price
current_price
Stock
PK stock_ID
stock_name
price_on_listing
date_of_listing
quantity
stock_category_ID
FK company_ID
Order
PK order_ID
FK trader_ID
date_of_order
status
total_amount
total_stocks_count
pending_stocks_count
Company
PK company_ID
company_name
date_of_registration
Wishlist
PK wishlist_ID
FK trader_ID
Wishlist_stock
PK wishlist_stock_ID
FK stock_ID
Order_stock
PK order_stock_ID
FK stock_ID
FK order_ID
quantity
price_asked
margin
price_placed
status
order_at_market_price
order_at_buy_price