TRADING PLATFORM

PROJECT NUMBER 12

MADE BY :-

HEMESH GUPTA (2020A7PS1688P)

AMIT GUPTA (2020A7PS1688P)

1. System requirement specification (SRS)

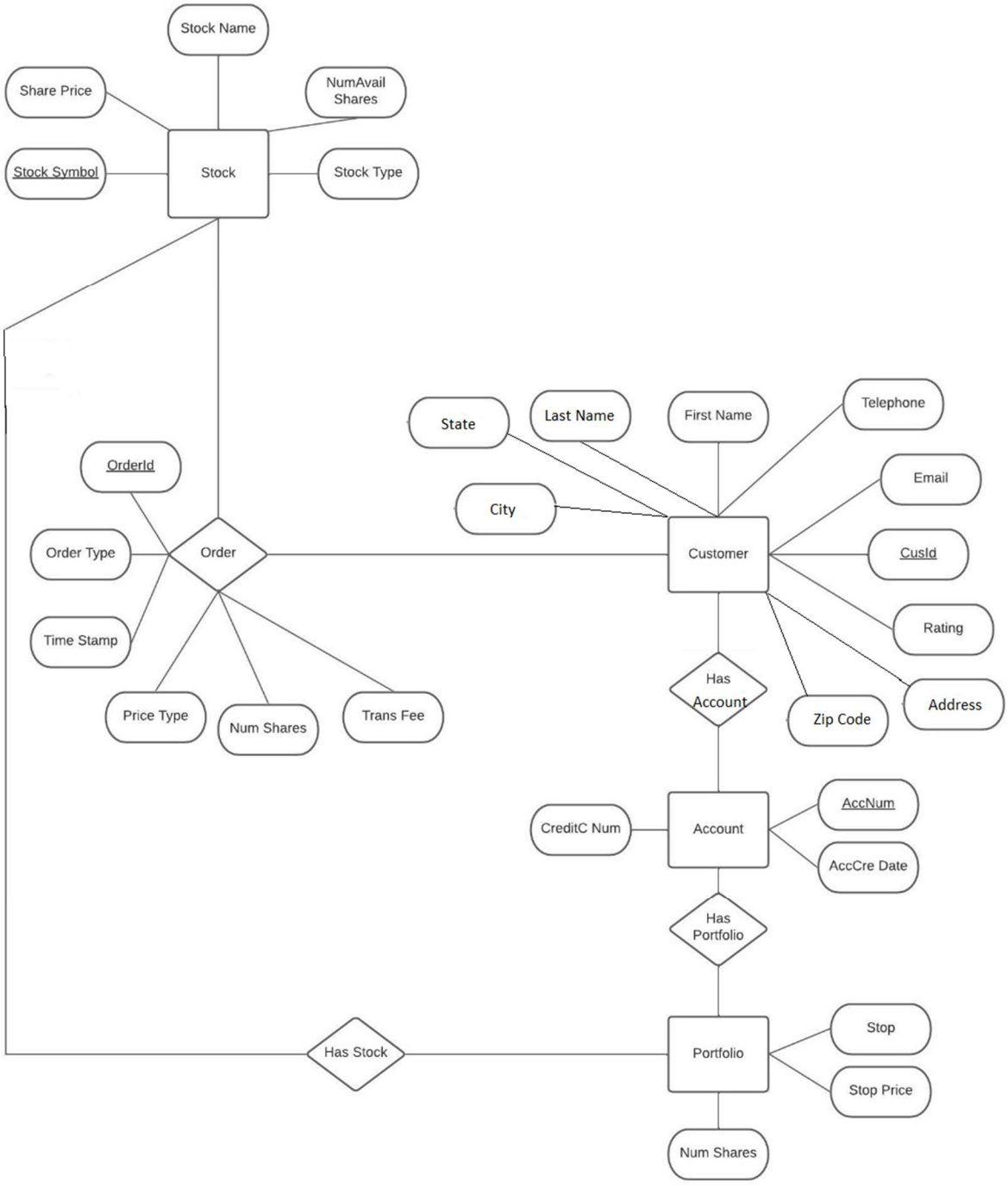
Purpose: The purpose of the project was to develop a platfrom where people can buy and sell stocks and other derivates. The platform contains the information of the customers and all of their transactions, also the all the information about the stocks is also maintained.

System requirements : To check the functionality of the platform the following software are required:-

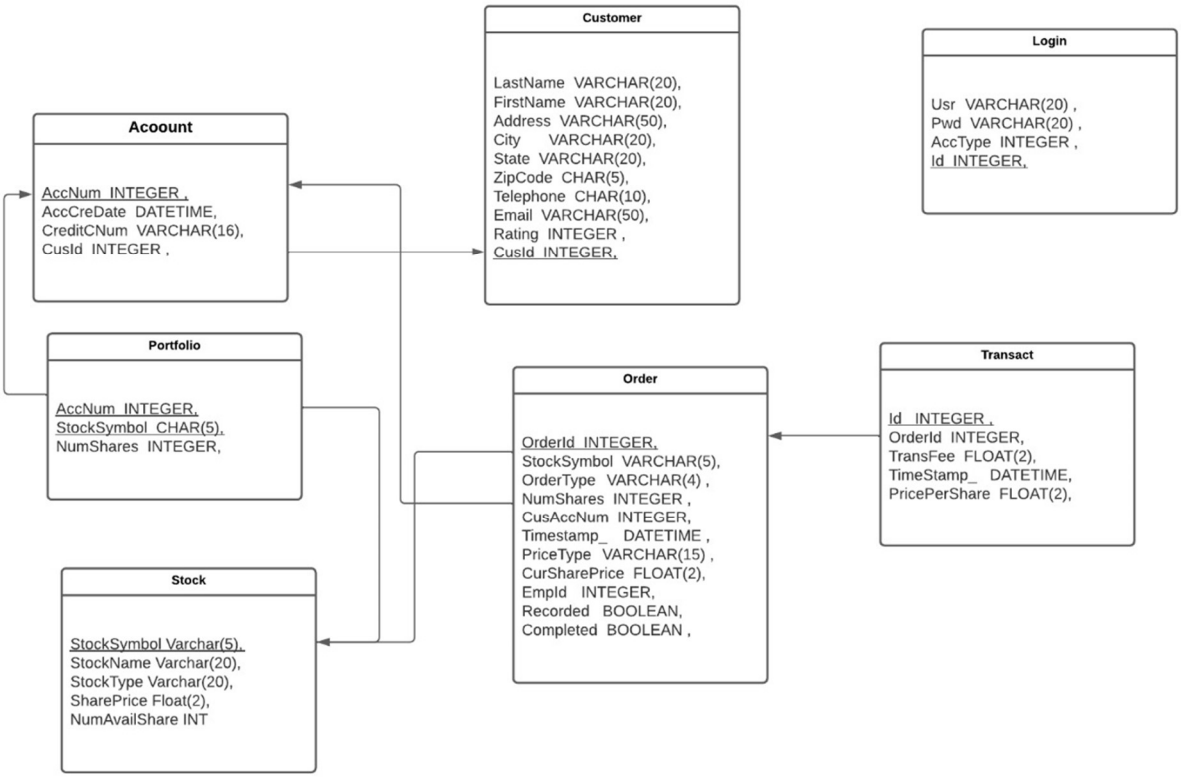
- MySQL
- MySQL workbench.

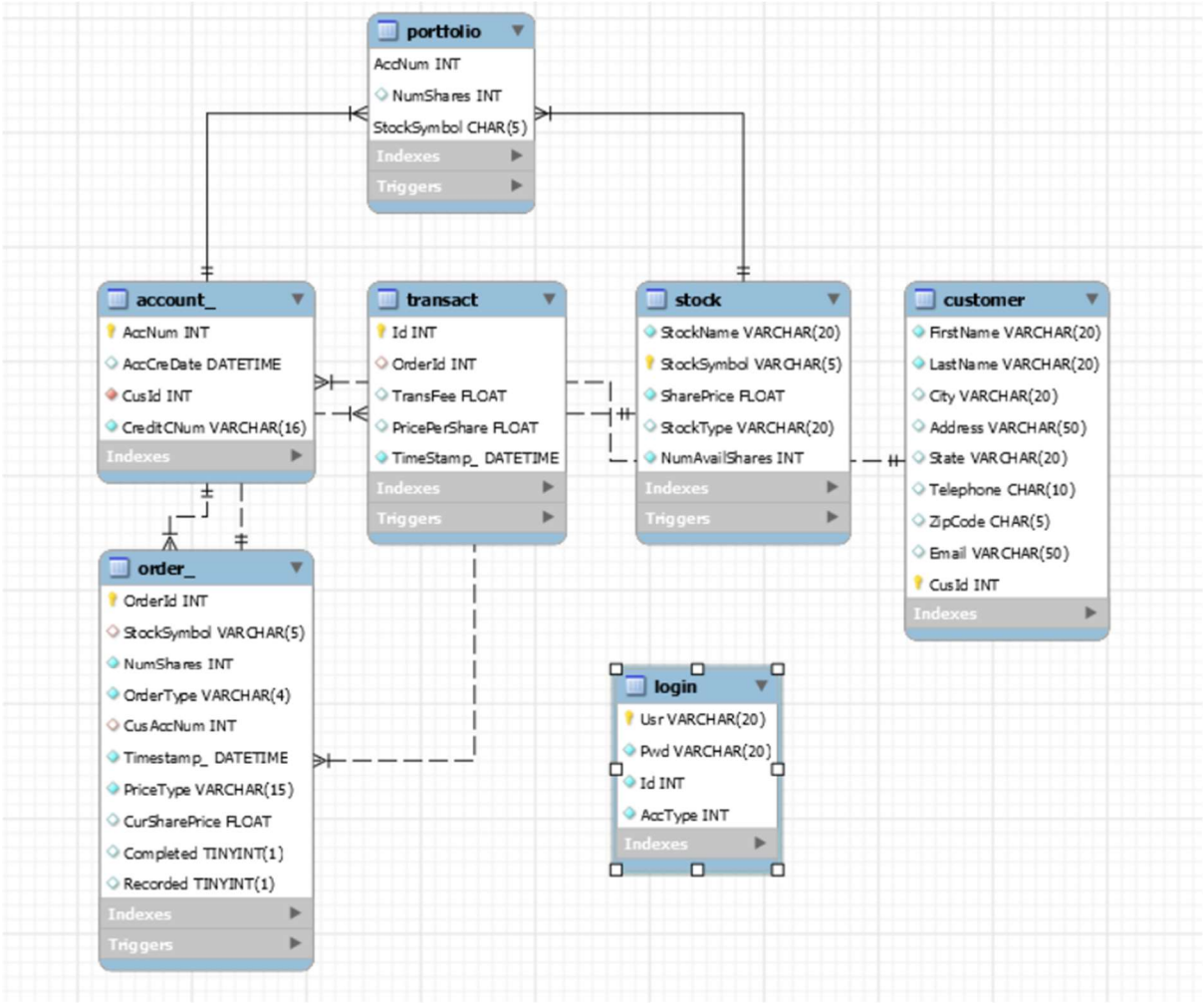
2. System modelling:-

a) E-R diagram



b) Schema design




porttolio
AccNum INT
NumShares INT
StockSymbol CHAR(5)
Indexes
Triggers
account_
AccNum INT
AccCreDate DATETIME
CusId INT
CreditCNum VARCHAR(16)
Indexes
transact
Id INT
OrderId INT
TransFee FLOAT
PricePerShare FLOAT
TimeStamp_ DATETIME
Indexes
Triggers
stock
StockName VARCHAR(20)
StockSymbol VARCHAR(5)
SharePrice FLOAT
StockType VARCHAR(20)
NumAvailShares INT
Indexes
Triggers
customer
FirstName VARCHAR(20)
LastName VARCHAR(20)
City VARCHAR(20)
Address VARCHAR(50)
State VARCHAR(20)
Telephone CHAR(10)
ZipCode CHAR(5)
Email VARCHAR(50)
CusId INT
Indexes
order_
OrderId INT
StockSymbol VARCHAR(5)
NumShares INT
OrderType VARCHAR(4)
CusAccNum INT
Timestamp_ DATETIME
PriceType VARCHAR(15)
CurSharePrice FLOAT
Completed TINYINT(1)
Recorded TINYINT(1)
Indexes
Triggers
login
Usr VARCHAR(20)
Pwd VARCHAR(20)
Id INT
AccType INT
Indexes

c) Data normalization

1NF : In our database we tried to keep all the values single valued, but still there were some attributes like telephone, email etc. which seems to be multivalued but the customer can enter only one value in them. Moreover all the attributes in our database are non composite (Simple).

Therefore the database is 1NF since all the attributes in all the tables are single valued and not composite.

2NF : In our database most of the tables have only one prime attributes. Tables – Portfolio, StockPriceHistory, and ConditionalPriceHistory have primary keys which comprise of two attributes but these tables does not have any partial dependency. So we can say that the database is in 2NF also.

3NF : For a database to be in 3NF there shouldn't be any transitive dependency. We have tried to remove all the transitive dependencies in all the tables except customer(where email can determine all the other attributes too, since emails are unique) and account(where CreditCNumber can also determine all the other attributes, since it is unique). Except these all the other are normalised to 3NF.

Further redundancy reduction : The relation hasAccount between customer and account is of type one to many relationship so instead of creating a new table for hasAccount we added the attribute CustId in Account which takes a foreign reference to the custId in customer.

The relation hasPortfolio between account and portfolio is also of type one to many. So we added the attribute AccNum in Portfolio which takes a foreign reference from Account.

Lastly the hasStock between Portfolio and Stock is also of type one to many. So we added the attribute StockSymbol in Portfolio table which takes a foreign reference from the table Stock.

d) List of tables required

1. Stock
2. Customer
3. Login
4. Account
5. Order
6. Transact
7. Portfolio

e) Additional requirements

1. Triggers:-

- SharePriceValid - Make sure share price is positive.
- AvailSharesValid - Make sure that there is not a negative number of shares available.
- NumSharesValid - -- Make sure the Order being added is for a valid number of shares.
- GetPrices - Change the prices on an inserted Order.
- OrderTypes - Make sure the order type is in the domain allowed.
- PriceTypes - Make sure that the price type is allowed.
- GetPrices2 - Set price and fee in Transact table.
- DoTransact - Executes a transaction for an order. Checks to make sure that the number of shares being bought or sold is correct.
- UpdatePortfolio - Adds or subtracts shares from the portfolio following an order transaction.
- AddToAccount - Adds a new portfolio entry to account following a transaction.
- UpdateStockQuantity - Updates the number of available shares of a stock after an order is executed.

2. Procedures:-

- setSharePrice - Sets the share price of a stock.
- listAllStocks - Produces a list of all stocks and how many orders areplaced for each one.
- getNumOrders - Produces a list of all stocks and how many orders are placed for each one.
- showStockRevenue - Shows the total revenue produced by each stock.
- showCustomerRevenue - Shows the total revenue produced by each customer.
- recordOrder - Records an order, meaning it is ready to be executed.
- addCustomer - Adds a customer to the database.
- deleteCustomer - Delete a customer from db
- getCustomerStockHoldings - Get customers current stock holdings.
- getCustomerOrdersHistory - A history of all current and past orders a customer has placed.
- getStockUsingType - Stocks available of a particular type.
- getStockUsingKeyword - Stocks available with a particular keyword or set of keywords in the stock name.
- getMostRecentOrderInfo - Most-recent order info of a customer