CHAPTER 1

INTRODUCTION

When you buy a stock, you're buying a piece of the company. When a company needs to raise money, it issues shares. This is done through an initial public offering (IPO), in which the price of shares is set based how much the company is estimated to be worth, and how many shares are being issued. The company gets to keep the money raised to grow its business, while the shares (also called stocks) continue to trade on an exchange, such as the New York Stock Exchange (NYSE). Traders and investors continue to buy and sell the stock of the company on the exchange, although the company itself no longer receives any money from this type of trading. The company only receives money from the IPO.

The first step in the process of creating a stock exchange information system is defining the general user requirements that will be set to the information systems. Beside the standard technical and technological aspects that are given attention during the development of the stock exchange Analysis system, it is essential that other important problems that IT managers encounter during the realization of the system be solved too.

The initial offering of stocks and bonds to investors is done in the primary market and subsequent trading is done in the secondary market. A stock exchange is often the most important component of a stock market. Supply and demand in stock markets is driven by various factors that, as in all free markets, affect the price of stocks. The stock exchange helps companies generate capital. As a primary market, it provides an avenue for them to sell new shares and bonds to investors.

Stocks are issued by companies to raise cash, and the stock then continues to trade on an exchange. Overall stocks have risen over the long-term, which makes owning shares attractive. There are also additional perks such as dividends (income), profit potential and voting rights. Share prices also fall, though, which is why investors typically choose to invest in a wide array of stocks, only risking a small percentage of their capital on each one.

In our project, we analysed a year's worth of stock portfolio for a selected group of companies and apply moving averages and Markov Chains to the data in hopes to predict the stock prices for the near future. The first thing we did was to apply moving averages to create an approximate evaluation of the data. In order to find moving averages, we first had to apply a moving average with an increment of three. This involves taking the sum of three days of stock and then dividing it by three. However, one can only do this starting at day three, because there was enough data to actually create a moving average. We eventually had a data set that included a moving average price and a closing price.

We then needed to find a difference data set to apply Markov Chains to. We took the difference of the closing price and the moving average price. These differences were going to be what we applied Markov Chains to. However, we first had to group the differences into four blocks. The reason why we did this was to create more accurate observations that it makes it more exact when we analyse the data. We then create a transition matrix. The entries in the matrix represented how many times the data points go from one block to another.

This leads to 16 observations of data. For example, the first row of entries the matrix represents the number of times the data goes from the first block and stays in the first block, the number of times the data goes from the first block to the second block, the number of times the data goes from the first block to the third block, and finally the number of times the data goes from the first block to the fourth block. All entries needed to be in decimal form, so the total number of observation points divided each entry.

With the matrix, we could now apply Markovian properties to our data. In other words, using Markovian properties we created a system of equations with the unknown variables being our steady states that we are aiming to obtain. These equations are sums of probabilities multiplied by our unknown variables. We then aimed to solve the system of equations to find our steady state probabilities.

With our steady state probabilities, we were now able to predict where each immediate stock price can fall into an interval. These probabilities are now good indicators of where the stock prices will fall. We then observed the new data and made some observations. We were able to predict what the possible price range for a given day could be. In conclusion, applying Markov Chains is an effective way to predict stock prices.

CHAPTER 2

PROBLEM STATEMENT.

2.1 OBJECTIVES OF THE PROJECT:

- To develop a system that should be able to create a portfolio for a company database with taking all its real time stock values.
- The aim of this paper is to develop a prototype of an application that is able to monitor the trading floor of the global Stock Exchange as well as the stocks.
- The system should be able fetch the data of any company with only by its company name.
- The system should hold the information about the user and stock purchased by the user and his amount he invested on it.
- The system should Using the stochastic process called Markov-Chains, we sought out to predict the immediate future stock prices for a few given companies.
- The system deals with the creation of a webpage and linking to MYSQL Server and fetching the real time stock data using a APIs to from Bombay/Indian stock market and other global market.
- The fetched data is used to create a graphical representation of the growth rate of the stock for every period for helping the investor by giving a clear picture of market value and growth.
- The system should perform in real time.
- The system should show the opening and closing price of next day stock value using Markov-chains.
- The system should have a query box for executing SQL queries on the designed database.
- The system should be able to keep in its database all work done in the local system.
- The application should be developed based on PHP and MySQL.
- The web app should be developed to run across all platforms.

2.2 DATABASE ASSUMPTIONS:

- Company shares are stored in a table for each stock.
- Stock has a Sid as a Primary Key, Sname, Owner, Face-value and Total_shares.
- Users has Uid as a Primary Key, Uname , Address, Balance, Age.
- Every user buys the share of a company where its is stored in a another table where it has Uid, Sid, nos(number of shares), Investment.
- When a user buys the share the stock company stores the remaining shares in table where it consists of details of Total_shares remaining for other users to buy.
- Each company shares are in separate table with company name as table name.
- Each company stock has a integer id, date, open, open_mov_av, opened, close, close_mov_av, close_diff, high, low.
- When the stock price is changed in the market the databases fetches the new value and stores in the respective company share table.
- In the case that the there is a change in the price of any stock before the set time the system should automatically refresh itself.
- Message table consists of id, email, message.
- The system must be able to show the users the information on the stock exchange.
- The stored values of stock price are selected for Markov-chain analysis purpose.

2.3 REQUIREMENT SPECIFICATION:

SOFTWARE:

1. XAMPP Server (Apache, MySQL, php).
2. Sublime text / Microsoft visual studio.
3. Google chrome for testing.

HARDWARE:

1. Core i3
2. 4 GB RAM
3. 500 GB Hard Disk

CHAPTER 3

DATABASE DESIGN.

3.1 BRIEF DESCRIPTION OF REALATIONAL MODEL.

The data and relationships will be represented by a set inter-related tables in a relational model. Every table will consist of column and a row where column refers to the attribute of an entity and row refers to the record.

3.2 ER-DIAGRAM.

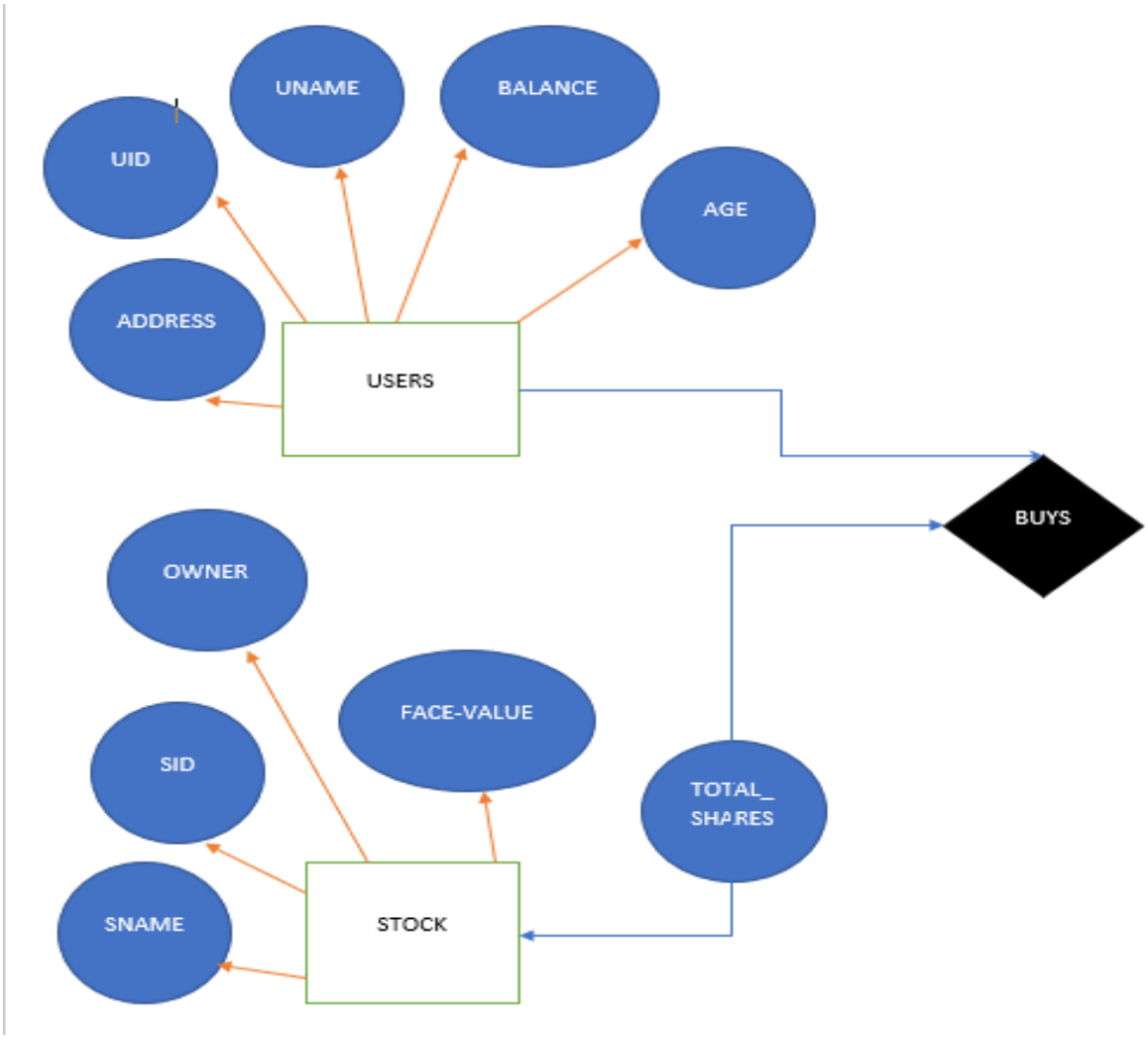

FIG-3.2.1 ER Diagram.

3.3 REALATIONAL-SCHEMA:

A relation schema is essentially the schema for a table. In a relational database (what people typically mean when they say database) each take can be referred to as a "relation" . Hence a relational schema is the design for the table. It includes none of the actual data, but is like a blueprint or design for the table, so describes what columns are on the table and the data types. It may show basic table constraints but not how it relates to other tables.

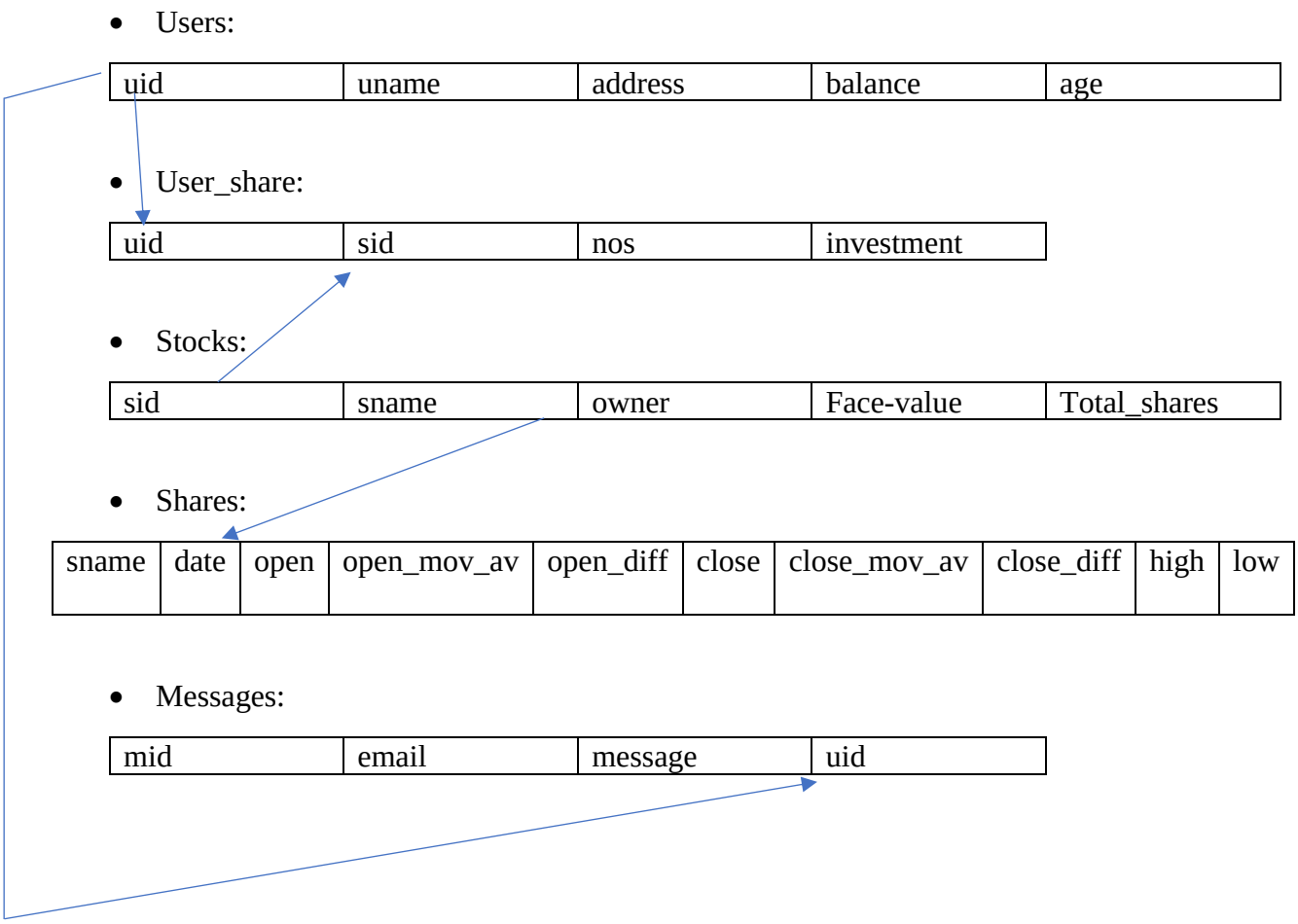

Fig-3.3.1 SCHEMA.

CHAPTER 4

IMPLEMENTATION

4.1 TABLE CREATION

- create table stock (sid int primary key, sname varchar(10), owner varchar(10), face-value number(10, 2), total_shares int);
- create table users (uid int primary key, uname varchar(10), address varchar(10), balance number(10, 2), age int);
- create table users_share (uid int, sid int, primary key(uid, sid), foreign key(uid) references users(uid), foreign key(sid) references stock(sid), nos int, investment);
- create table <COMPANY_NAME> (id int(11) not null auto_increment, date varchar(10) default null, open decimal(5,3) default null, open_mov_av decimal(10,8) default null, open_diff decimal(11,9) default null, close decimal(4,2) default null, close_mov_av decimal(10,8) default null, close_diff decimal(11,9) default null, high decimal(6,4) default null, low decimal(6,4) default null, primary key (`id`));
- create table messages (id int(11) not null auto_increment, name varchar(32) not null,email varchar(64) not null, message text not null, uid int, foreign key(uid) refernces users(uid), primary key (`id`));

4.2 QUERIES ON DDL COMMAND

- alter table stock add column percentage int;
- alter table users change age doj date;
- alter table stock drop percentage;

4.3 AGGREGATE FUNCTION

- select count(*) from apple;
- select max(investment) from user_share;
- select avg(investment) from user_share where sid=101;

4.4 QUERIES ON JOINTS

- select uname, sname from users inner join stock on users.uid=stock.sid;
- select uname from users outer join user_share on user_share.uid=users.uid;
- select owner from stock inner join user_share on user_share.sid=stock.sid;

4.5 NESTED AND CORE-RELATED QUERIES

- select sname, owner from stock where sid in (select sid from user_share where uid<=205);
- select * from users where uid in (select uid from user_share where sid in (select sid from stock where sname='microsoft'));
- select * from users where uid in (select uid from user_share where sid in (select sid from stock where total_shares>1500 and sid>103));

4.6 VIEWS

- create view ck as select sname, uname from users u, stock s, user_share us where u.uid=us.uid and s.sid=us.sid;
- create view jk as select total_shares , nos from stock s, user_share us where s.sid=us.sid;

4.7 TRIGGERS

- **Trigger T1:**

➢ Create table number_of_rows(cnt_user int, cnt_stock int);
➢ Insert into number_of_rows values(10, 5)
➢ Delimiter //
CREATE TRIGGER t1 AFTER INSERT ON users
FOR EACH ROW
BEGIN UPDATE number_of_rows SET cnt_user=cnt_user+1;
➢ END;//
➢ Delimiter ;

- **Trigger T2:**

➢ Delimiter //
CREATE TRIGGER t2 AFTER INSERT ON stock
FOR EACH ROW
BEGIN UPDATE number_of_rows SET cnt_stock=cnt_stock+1;
➢ END;//
➢ Delimiter ;

- **Trigger T3:**

➢ Create table remaining (total_shares int, nos int, left_shares int);
➢ INSERT INTO remaining values (total_shares, nos)
SELECT total_shares, nos FROM jk;
➢ Delimiter //
➢ CREATE TRIGGER t3 AFTER INSERT ON user_share
FOR EACH ROW
BEGIN INSERT INTO remaining (total_shares, nos)
SELECT total_shares, nos FROM jk;
➢ END;//
➢ Delimiter ;

➢ CHAPTER 5

SNAPSHOTS

- select * from microsoft where id<10;

select * from microsoft where id<10;

id	date	open	open_mov_av	open_diff	close	close_mov_av	close_diff	high	low
1	02-05-2014	40.310			39.690			40.3400	39.6600
2	01-05-2014	40.240			40.000			40.3600	39.9500
3	30-04-2014	40.400	40.31666667	0.08333333	40.400	40.03000000	0.37000000	40.5000	40.1699
4	29-04-2014	41.100	40.58000000	0.52000000	40.510	40.30333333	0.20666667	41.1900	40.3900
5	28-04-2014	40.140	40.54666667	-0.40666667	40.870	40.59333333	0.27666667	41.2900	40.0900
6	25-04-2014	40.290	40.51000000	-0.22000000	39.910	40.43000000	-0.52000000	40.6800	39.7500
7	24-04-2014	39.740	40.05666667	-0.31666667	39.860	40.21333333	-0.35333333	39.9700	39.3000
8	23-04-2014	39.990	40.00666667	-0.01666667	39.690	39.82000000	-0.13000000	39.9900	39.4700
9	22-04-2014	39.960	39.89666667	0.06333333	39.990	39.84666667	0.14333333	40.1400	39.8300

Fig-5.1 SNAPSHOT 1

- select * from users where uid in (select uid from user_share where sid in (select sid from stock where sname='microsoft'));

uid	uname	balance	address	age
205	alok	789000000	australia	19
209	tim	567788900	canada	90

Fig-5.2 SNAPSHOT 2

- select sname, owner from stock where sid in (select sid from user_share where uid<=205);

sname	owner
apple	steve-jobs
facebook	mark-zuckerburg
genelec	namratha
microsoft	bill-gates

Fig-5.3 SNAPSHOT 3

- select avg(investment) from user_share where sid=101;

avg(investment)
5223162.6667

Fig-5.4 SNAPSHOT 4

- select sid, count(*) from user_share group by sid;

sid	count(*)
101	3
102	2
103	2
104	2
105	2

Fig-5.5 SNAPSHOT 5

- select * from ck;

sname	uname
apple	anmol
apple	rahul
facebook	alex
facebook	camila
genelec	rita
genelec	regina
ibm	arun
ibm	jim-jung
microsoft	alok
microsoft	tim

Fig-5.6 SNAPSHOT 6

CHAPTER 6

CONCLUSION

Thus, we have built this web app in view of all those investors who are at a loss as to whether or not they should make any further investments in a particular share. Since the working algorithm of this app is a mathematical model, the accuracy of this app will be greatly enhanced. This app will be providing mathematical figures which will make it convenient for an investor to come to a decision and that too without the hassle of focusing all their grey cells in an attempt to take note of the expert advices available on various forms of media.

This app will be providing predictions at the mere click of a finger. All the user needs to be concerned about is the stock that he/she wants to invest in. Although no prediction can be considered to be perfect, this app strives to achieve near perfection while making predictions thereby reducing the chances of incurring loss for an investor.

Our work thus far has implemented the use of Markov chains to predicting stock prices. Using the difference between forecast prices and actual prices, we have calculated the possible steady state, or probability of the future of the difference price. We checked our app to make predictions for selected stocks and the predictions were found to be nearly accurate. Our app also provides current news feed related to selected stocks, in addition to the prediction, as we have linked it to Yahoo Finance. We are using historical data, from Yahoo Finance, of the companies which are listed in NASDAQ. Our web app is scheduled for a CRON job to run a PHP script in order to update the data (based on which the predictions are made).

Finally, this webpage app developed using php and Xampp SQL server works on real-time to fetch the data from the stock market using API and helps us to determine the current price of stock and its historical data and helping the users to decide weather to invest or not. Thus, application also holds the details of user's information along with their respective stock shares, investment, balance, age address etc.

BIBLIOGRAPHY

- http://en.wikipedia.org/wiki/BSE_SENSEX.
- http://en.wikipedia.org/wiki/Bombay_Stock_Exchange.
- Investopedia (2015), Portfolio Management Retrieved from http://www.investopedia.com/terms/p/portfoliomanagement.asp.
- Hal McIntyre, Megan Johnson, and Deborah Fortuna, Software Testing for Financial Services Firm, The Summit Group, New York, June 2003.
- IDEF: Standard IDEF0 i IDEFIX, www.idef.com
- Acharya, V. V., & Pedersen, L. H. (2005). Asset pricing with liquidity risk. Journal of Financial Economics, 77(2), 375-410.
- Hendershott, T., & Moulton, P. C. (2011). Automation, speed, and stock market quality: The NYSE's hybrid. Journal of Financial Markets, 14(4), 568-604.
- https://en.wikipedia.org/wiki/Markov_chain
- Predicting Stock Price (Approved Paper) by Shuchi S. Mitra and Michael J. Ruggieri
- https://en.wikipedia.org/wiki/Application_programming_interface