Michael Howerter

Douglas Masini

Nicolas Werner

Crypto Tracking:

An Application to be created by "Project Group 3"

Project Group 3 will be creating a web-based cryptocurrency tracking database system that stores, analyzes, and presents information about several cryptocurrencies such as Bitcoin, Ethereum, and Ripple.

Contents

4.1

A. **Main Functions**

This application will be a web-based interface to the cryptocurrency database. Its primary function will be to store information about popular currencies and utilize that data to provide users the following features:

- The ability to view general information about individual cryptocurrencies such as:
  - Current exchange rate
  - Average exchange rate
  - Ledger for currency blocks
- Searching for archived information about cryptocurrencies
  - Past exchange rates on a given day or date range
- Comparisons between cryptocurrencies
  - Differences in price
  - Current exchange rate between the currencies
  - Using satoshi or gwei levels to model differences between currencies
- Forecasting of one or multiple cryptocurrencies
  - Estimating future price(s) based on current trends
- Graphs which allow a user to select information to display over a defined period of time about one or many currencies
  - Providing a graph-based view of information from the comparisons or general information over a set period of time
- Heat map showing popularity of a given currency against a calendar
  - Popularity would be based on either volume of transactions in a set period of time or volume of coin available on the market
- User accounts/profiles that allow users to select favorite currencies and track certain information via a dashboard inside the application
  - Would require storing separate user information in the database

B. **Interaction of Functions**

- Viewing general information about a currency
  - User will select an available currency from a list
  - The selected currency and date will be queried to return general information to the user
- Searching archived information
  - User will select a currency from a list and a date or date range in the past
  - The two user inputs will be used in a query to return past general information or a series of general information outputs for the selected currency and date
- Comparisons between currencies
  - The user will select two or more currencies

    - These currencies will be used in a series of queries and formulas to return information such as exchange rate between the two currencies, exchange rates back to USD, and differences between general information
    - These inputs will also be used to generate a graph of the information to provide a direct visual comparison between the currencies
        - Users will be able to modify the graph by selecting a different time period (e.g. daily, weekly, monthly, yearly), different information to chart (e.g. current price, average price, exchange rate, etc)
- Forecasting the currencies
    - The user will select one or multiple currencies to forecast
    - Using the information produced by the general and archived information query, an estimated price for a set period will be returned
    - I.e. User selects bitcoin, the current price of bitcoin will be returned along with the predicted price tomorrow or next week
- The graphing functionality will be entirely dependent upon the query information from the other queries. This information will be used to provide the user a graphical view of the data
- The heat map will use a query for the currency to chart use of the currency against dates. Utilizing this usage and date information, a color-coded calendar (or date range) will be presented to show gains or losses in popularity
- User information will be created via a 'register' functionality
    - This information along with a login will be stored in a user's portion of the database
    - The user's profile will be linked to particular currencies so that the information about those currencies is presented in a dashboard at login time

C. **Key Queries**

These queries will be used to retrieve key pieces of information about each cryptocurrency. This information will then be used to perform calculations and derive new information about each currency

- What is the current price of {currency} for today
- What is the current price of {currency} for {date/date range}
- What is the current value of {currency} in satoshi levels or gwei levels?
- What is the average {price/volume/etc} for {currency}

D. **Necessary Data Collection**

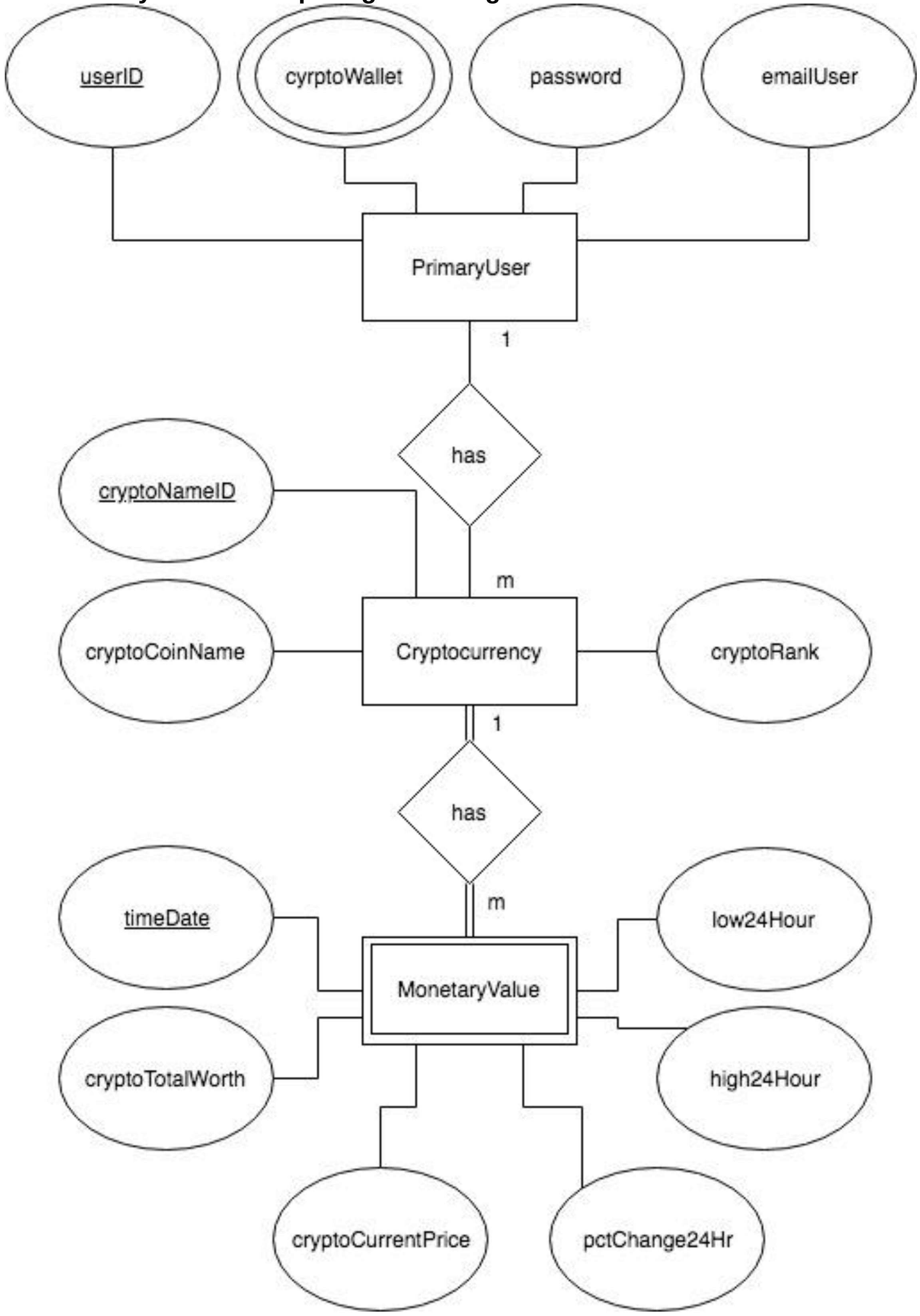
The cryptocurrency tracking database system will exhibit inheritance and have the ability to derive useful values from the stored data points. Cryptocurrency dating back to 2014 at the latest will supply well over the 100,000 data point minimum required. The top level of our data base system will be derived from the overall worth of the cryptocurrency, and thus will catalog all the current cryptocurrencies in that order. Each cryptocurrency will have a list of dates that span to the length of life from the cryptocurrency dating as far back to 2014. The information that will be gathered from publicly available information from https://www.cryptocompare.com/api/.

E. **Necessary Software**

We will be using the internet for our user interface. UF CISE Oracle will be used as our database.

4.2

A. Entity-Relationship Diagram Design



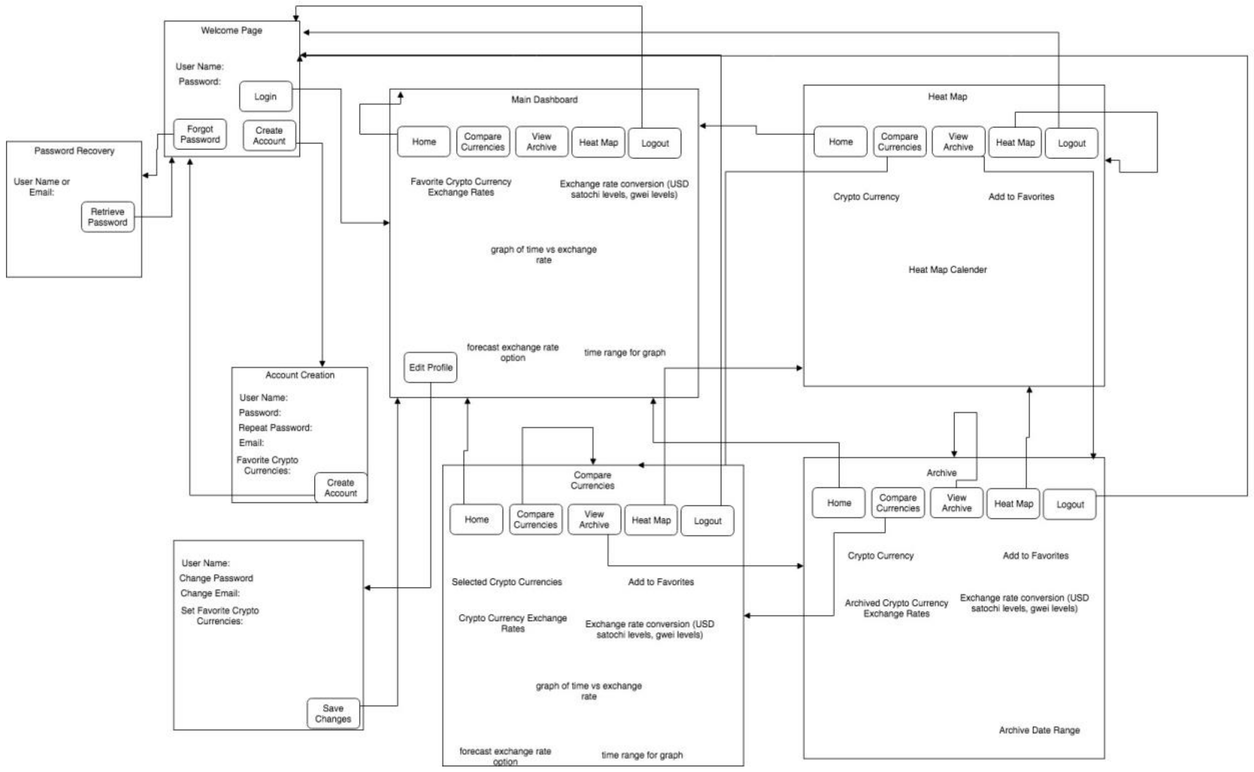
B. **Explanation of ER Diagram**

The ER diagram detailed above describes the schema for the database that underlies the core functionality of this project. A fundamental piece of the project specification included a multi-user capability. As the users would be distinct from and capable of owning cryptocurrencies, it was decided to create a unique entity set for storing information about the users. Each user will be uniquely identified by a userID but will also have an email and a password. Finally, users will contain a multi-valued attribute called CryptoWallet meant to represent an individual user's cryptocurrency wallet. Each user may have multiple wallets to serve such functions as distinct accounts (like a checking v.s. savings account), storage of different currencies, etc.

The next major component of the database would be the cryptocurrencies and relevant information about each cryptocurrency. It was possible to view cryptocurrencies as a single entity however as there would be some information that remains fairly static on a long-term basis and other information that would be reasonable expected to change frequently (either daily or hour to hour) the design decision was made to split the attributes pertaining to a cryptocurrency into two distinct entity sets. Therefore, cryptocurrency will be used to store the predominantly static information about each currency uniquely identified by a cryptoNameID (similar to a guid for cryptocurrencies) as well as the full currency name, and its current rank (in popularity) against the other currencies. This entity will be a strong entity set to the MonetaryValue entity set which is uniquely identified by combining the cryptoID and a Date/Time stamp. This weak entity set will provide the financial information such as daily high, daily low, and current price that is expected to change each time the information is 'refreshed'. Due to the records in MonetaryValue being identified via a date/time format, there will be many entries related to a single cryptocurrency with each monetary value participating in the relationship.

Finally, it is expected that users will own a cryptocurrency or multiple cryptocurrency. Thus, a relationship was created between the cryptocurrency entity and users called "has" to indicate ownership of a cryptocurrency by a user. Each user can own multiple cryptocurrencies.

C. **User Interface Design**



4.3

A. **Database Schema**

PrimaryUser(userID : integer, userPassword : string, userEmail : string, cryptoWallet : string)

CrytpCurrency(cryptoNameID : integer, cryptoCoinName : string, cryptoRank : integer)

MonetaryValue(timeDate : Date, cryptoTotalWorth : double, cryptoCurrentPrice : double, pctChange24Hour : float, high24Hour : double, low24Hour : double, cryptoNameID : integer)

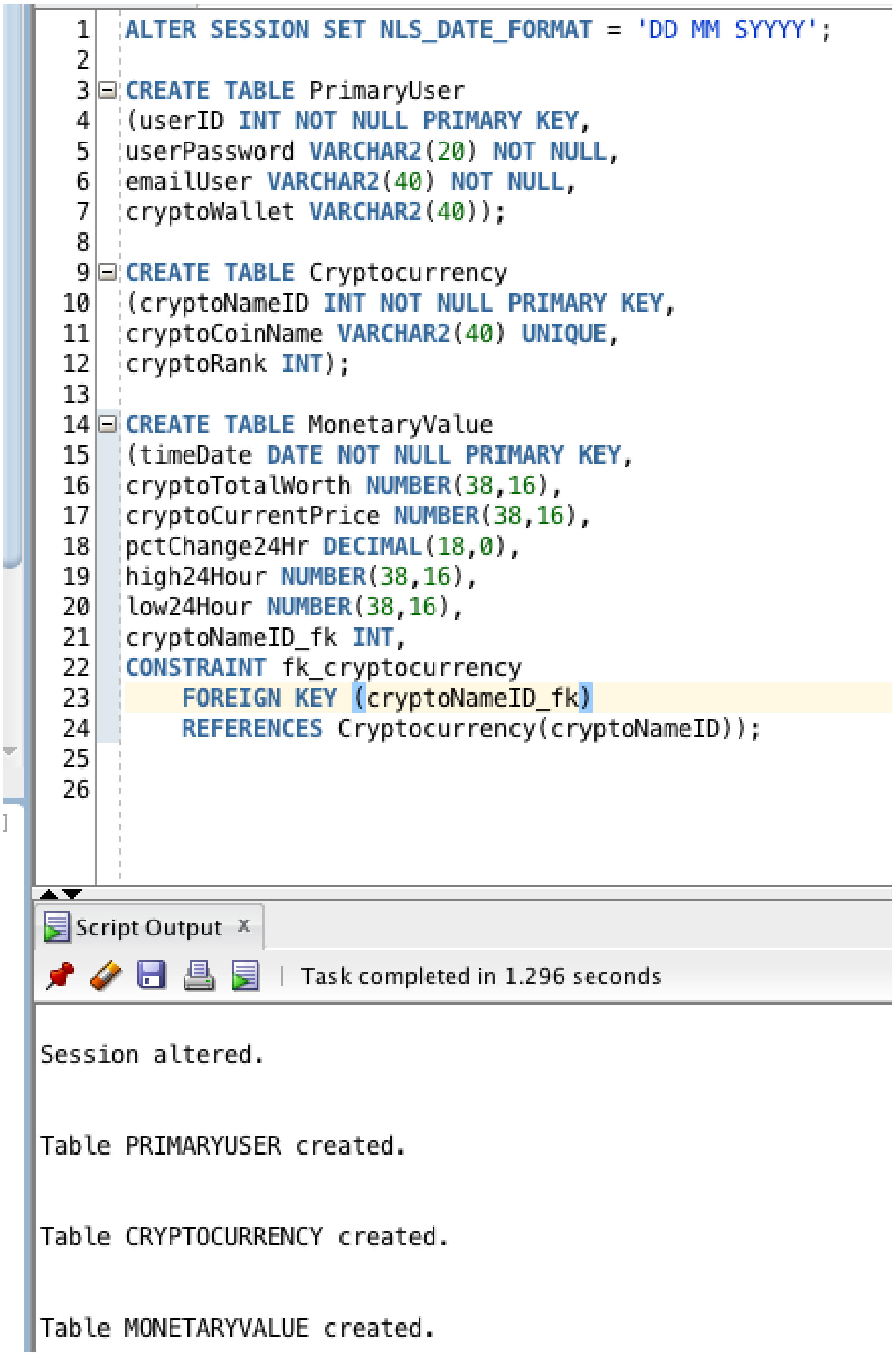
```
ALTER SESSION SET NLS_DATE_FORMAT = 'DD MM SYYYY';

CREATE TABLE PrimaryUser
(userID INT NOT NULL PRIMARY KEY,
userPassword VARCHAR2(20) NOT NULL,
emailUser VARCHAR2(40) NOT NULL,
cryptoWallet VARCHAR2(40));

CREATE TABLE Cryptocurrency
(cryptoNameID INT NOT NULL PRIMARY KEY,
cryptoCoinName VARCHAR2(40) UNIQUE,
cryptoRank INT);

CREATE TABLE MonetaryValue
(timeDate DATE NOT NULL PRIMARY KEY,
cryptoTotalWorth NUMBER(38,16),
cryptoCurrentPrice NUMBER(38,16),
pctChange24Hr DECIMAL(18,0),
high24Hour NUMBER(38,16),
low24Hour NUMBER(38,16),
cryptoNameID_fk INT,
CONSTRAINT fk_cryptocurrency
   FOREIGN KEY (cryptoNameID_fk)
   REFERENCES Cryptocurrency(cryptoNameID));
```

B**. Oracle Screenshots**



```
ALTER SESSION SET NLS_DATE_FORMAT = 'DD MM SYYYY';

CREATE TABLE PrimaryUser
(userID INT NOT NULL PRIMARY KEY,
userPassword VARCHAR2(20) NOT NULL,
emailUser VARCHAR2(40) NOT NULL,
cryptoWallet VARCHAR2(40));

CREATE TABLE Cryptocurrency
(cryptoNameID INT NOT NULL PRIMARY KEY,
cryptoCoinName VARCHAR2(40) UNIQUE,
cryptoRank INT);

CREATE TABLE MonetaryValue
(timeDate DATE NOT NULL PRIMARY KEY,
cryptoTotalWorth NUMBER(38,16),
cryptoCurrentPrice NUMBER(38,16),
pctChange24Hr DECIMAL(18,0),
high24Hour NUMBER(38,16),
low24Hour NUMBER(38,16),
cryptoNameID_fk INT,
CONSTRAINT fk_cryptocurrency
    FOREIGN KEY (cryptoNameID_fk)
    REFERENCES Cryptocurrency(cryptoNameID));
```

```
Session altered.


Table PRIMARYUSER created.


Table CRYPTOCURRENCY created.


Table MONETARYVALUE created.
```