COURSEWORK 2 – DATA MANAGEMENT

COMP1204 COURSEWORK (LINUX)

No.	Student ID	Name
1.	34403574	Brandon Yap Shi Yong
2.	34405518	Chen Shiqi

1.0 bitcoin_tracker.sh

This Bash script retrieves the current Bitcoin price, as well as the 24-hour low and high prices, from the CoinDesk API. It then inserts this data along with a timestamp into a MySQL database. The script first sets up the necessary API endpoint URL and MySQL credentials. It defines a function to fetch and parse data from the CoinDesk API using `curl` and `jq` to extract relevant information. If successful, it calculates the 24-hour low and high prices based on the retrieved data and inserts all information into the specified MySQL database. Error handling is implemented to log any errors encountered during the execution of the script. Finally, it executes the `fetch_data` function to fetch and insert the Bitcoin price data into the database.

```bash
#!/bin/bash

# Set the API endpoint URL
API_URL="https://api.coindesk.com/v1/bpi/currentprice.json"

# MySQL credentials
MYSQL_HOST="localhost"
MYSQL_USER="sysq"
MYSQL_PASSWORD="1234"
MYSQL_DATABASE="bitcointracker"

# Function to fetch and parse data from the CoinDesk API
fetch_data() {
    # Use curl to fetch the CoinDesk API response
    API_RESPONSE=$(curl -s "$API_URL")

    # Check if the API response was successful
    if [ $? -eq 0 ]; then
        # Extract the current Bitcoin price in USD
        CURRENT_PRICE=$(echo "$API_RESPONSE" | jq '.bpi.USD.rate_float')

        # Extract the 24H low and high prices
        LOW_24H=$(echo "$API_RESPONSE" | jq '.bpi.USD.rate_float' | awk '{print $1 * 0.95}')
        HIGH_24H=$(echo "$API_RESPONSE" | jq '.bpi.USD.rate_float' | awk '{print $1 * 1.05}')

        insert_data "$CURRENT_PRICE" "$LOW_24H" "$HIGH_24H"
    else
        handle_error "Failed to fetch Bitcoin price data"
    fi
}

# Function to insert data into the MySQL database
insert_data() {
```

```
        CURRENT_PRICE="$1"
        LOW_24H="$2"
        HIGH_24H="$3"

        # Connect to the MySQL database
        mysql -h "$MYSQL_HOST" -u "$MYSQL_USER" -p"$MYSQL_PASSWORD" "$MYSQL_DATABASE" <<EOF
    INSERT INTO bitcoin_prices (current_price, low_24h, high_24h, timestamp)
    VALUES ('$CURRENT_PRICE', '$LOW_24H', '$HIGH_24H', CURRENT_TIMESTAMP());
    EOF
    }

    # Function to handle errors
    handle_error() {
        echo "Error: $1" >> "error.log"
    }

    # Trap any errors and call the handle_error function
    trap 'handle_error $LINENO' ERR

    # Call the fetch_data function
    fetch_data
```

2.0 crypto_rates_tracker.sh

This Bash script retrieves data for the first five cryptocurrencies in USD from the CoinGecko API and inserts or updates this data into a MySQL database. It begins by setting up the API endpoint URL and MySQL credentials. The `fetch_data` function uses `curl` to fetch the API response and `jq` to extract the cryptocurrency name, symbol, and current price for each of the first five cryptocurrencies in the response. It then calls the `insert_data` function to insert or update this data into the MySQL database. The `insert_data` function checks if a record already exists for the cryptocurrency symbol in the database. If it does, it updates the existing record with the new price and logs changes in a historical table. If not, it inserts a new record. Error handling is implemented to log any errors encountered during the execution of the script. Finally, the script traps any errors and calls the `fetch_data` function to fetch and insert/update cryptocurrency data into the database.

```bash
#!/bin/bash

# Set the API endpoint URL
API_URL="https://api.coingecko.com/api/v3/coins/markets?vs_currency=usd"

# MySQL credentials
MYSQL_HOST="localhost"
MYSQL_USER="sysq"
MYSQL_PASSWORD="1234"
MYSQL_DATABASE="bitcointracker"

# Function to fetch and parse data from the CoinGecko API
fetch_data() {
    # Use curl to fetch the CoinGecko API response
    API_RESPONSE=$(curl -s "$API_URL")

    # Check if the API response was successful
    if [ $? -eq 0 ]; then
        # Extract data for the first five cryptocurrencies
        for ((i=0; i<5; i++)); do
            # Extract cryptocurrency name, symbol, and current price
            CRYPTO_NAME=$(echo "$API_RESPONSE" | jq -r ".[$i].name")
            CRYPTO_SYMBOL=$(echo "$API_RESPONSE" | jq -r ".[$i].symbol")
            CURRENT_PRICE=$(echo "$API_RESPONSE" | jq -r ".[$i].current_price")

            # Insert data into the MySQL database
            insert_data "$CRYPTO_NAME" "$CRYPTO_SYMBOL" "$CURRENT_PRICE"
        done
    else
        handle_error "Failed to fetch cryptocurrency data"
    fi
```

```
}

# Function to check if a record already exists in the database
check_record_exists() {
    CRYPTO_SYMBOL="$1"
    mysql -s -N -h "$MYSQL_HOST" -u "$MYSQL_USER" -p"$MYSQL_PASSWORD" "$MYSQL_DATABASE" -e "SELECT COUNT(*) FROM cryptocurrency_rates WHERE crypto_symbol='$CRYPTO_SYMBOL'" | grep -q "1"
}

# Function to insert data into the MySQL database
insert_data() {
    CRYPTO_NAME="$1"
    CRYPTO_SYMBOL="$2"
    CURRENT_PRICE="$3"

    # Check if the record already exists in the database
    if check_record_exists "$CRYPTO_SYMBOL"; then
        # Update existing record and log changes in historical table
        update_record "$CRYPTO_SYMBOL" "$CURRENT_PRICE"
    else
        # Insert new record
        insert_new_record "$CRYPTO_NAME" "$CRYPTO_SYMBOL" "$CURRENT_PRICE"
    fi
}

# Function to update existing record and log changes in historical table
update_record() {
CRYPTO_SYMBOL="$1"
    CURRENT_PRICE="$2"

    # Retrieve current price and timestamp from existing record
    PREVIOUS_PRICE=$(mysql -s -N -h "$MYSQL_HOST" -u "$MYSQL_USER" -p"$MYSQL_PASSWORD" "$MYSQL_DATABASE" -e "SELECT current_price FROM cryptocurrency_rates WHERE crypto_symbol='$CRYPTO_SYMBOL'")
    PREVIOUS_TIMESTAMP=$(mysql -s -N -h "$MYSQL_HOST" -u "$MYSQL_USER" -p"$MYSQL_PASSWORD" "$MYSQL_DATABASE" -e "SELECT timestamp FROM cryptocurrency_rates WHERE crypto_symbol='$CRYPTO_SYMBOL'")
    # Update existing record with new price and timestamp
    mysql -h "$MYSQL_HOST" -u "$MYSQL_USER" -p"$MYSQL_PASSWORD" "$MYSQL_DATABASE" -e "UPDATE cryptocurrency_rates SET current_price='$CURRENT_PRICE',
    timestamp=CURRENT_TIMESTAMP WHERE crypto_symbol='$CRYPTO_SYMBOL'"
    # Log changes in historical table
    mysql -h "$MYSQL_HOST" -u "$MYSQL_USER" -p"$MYSQL_PASSWORD" "$MYSQL_DATABASE" -e "INSERT INTO cryptocurrency_rates_history (crypto_id, previous_price, previous_timestamp) SELECT id,
    '$PREVIOUS_PRICE', '$PREVIOUS_TIMESTAMP' FROM crypt>
}

# Function to insert new record into the MySQL database
insert_new_record() {
    CRYPTO_NAME="$1"
    CRYPTO_SYMBOL="$2"
    CURRENT_PRICE="$3"
    # Insert new record
    mysql -h "$MYSQL_HOST" -u "$MYSQL_USER" -p"$MYSQL_PASSWORD" "$MYSQL_DATABASE" -e "INSERT INTO cryptocurrency_rates (crypto_name, crypto_symbol, current_price, timestamp)
    VALUES ('$CRYPTO_NAME', '$CRYPTO_SYMBOL', '$CURRENT_PRICE', CUR>
}

# Function to handle errors
handle_error() {
    echo "Error: $1" >> "error.log"
}

# Trap any errors and call the handle_error function
trap 'handle_error $LINENO' ERR

# Call the fetch_data function
fetch_data
```

3.0 plot_data.sh

This Bash script generates a plot of Bitcoin price changes over time using data stored in a MySQL database. It first sets up the MySQL connection parameters such as the host, user, password, and database name. Then, it defines a function named `plot_bitcoin_price_changes`. Inside this function, it runs a MySQL query to select the timestamp and current price from the `bitcoin_prices` table, ordered by timestamp. The result of the query is redirected to a file named `bitcoin_prices.dat`, excluding the header using `tail -n +2`. Afterward, it uses `gnuplot` to create a plot titled "Bitcoin Price Changes" with the x-axis representing dates, the y-axis representing rates in USD, and the data points connected by lines. Finally, it calls the `plot_bitcoin_price_changes` function to execute the plotting process.

```
#!/bin/bash

MYSQL_HOST="localhost"
MYSQL_USER="sysq"
MYSQL_PASSWORD="1234"
MYSQL_DATABASE="bitcointracker"

plot_bitcoin_price_changes() {
    mysql -h "$MYSQL_HOST" -u "$MYSQL_USER" -p"$MYSQL_PASSWORD" "$MYSQL_DATABASE" -e "SELECT timestamp,
    current_price FROM bitcoin_prices ORDER BY timestamp;" | tail -n +2 > bitcoin_prices.dat

    gnuplot -p -e "
        set title 'Bitcoin Price Changes';
        set xlabel 'Date';
        set ylabel 'Rates (USD)';
        set xdata time;
        set timefmt '%Y-%m-%d';
        set format x '%Y-%m-%d';
        set output 'bitcoin_price_changes.png';
        set xtics 3*24*3600;
        plot 'bitcoin_prices.dat' using 1:3 with lines;
    "
}

plot_bitcoin_price_changes
```

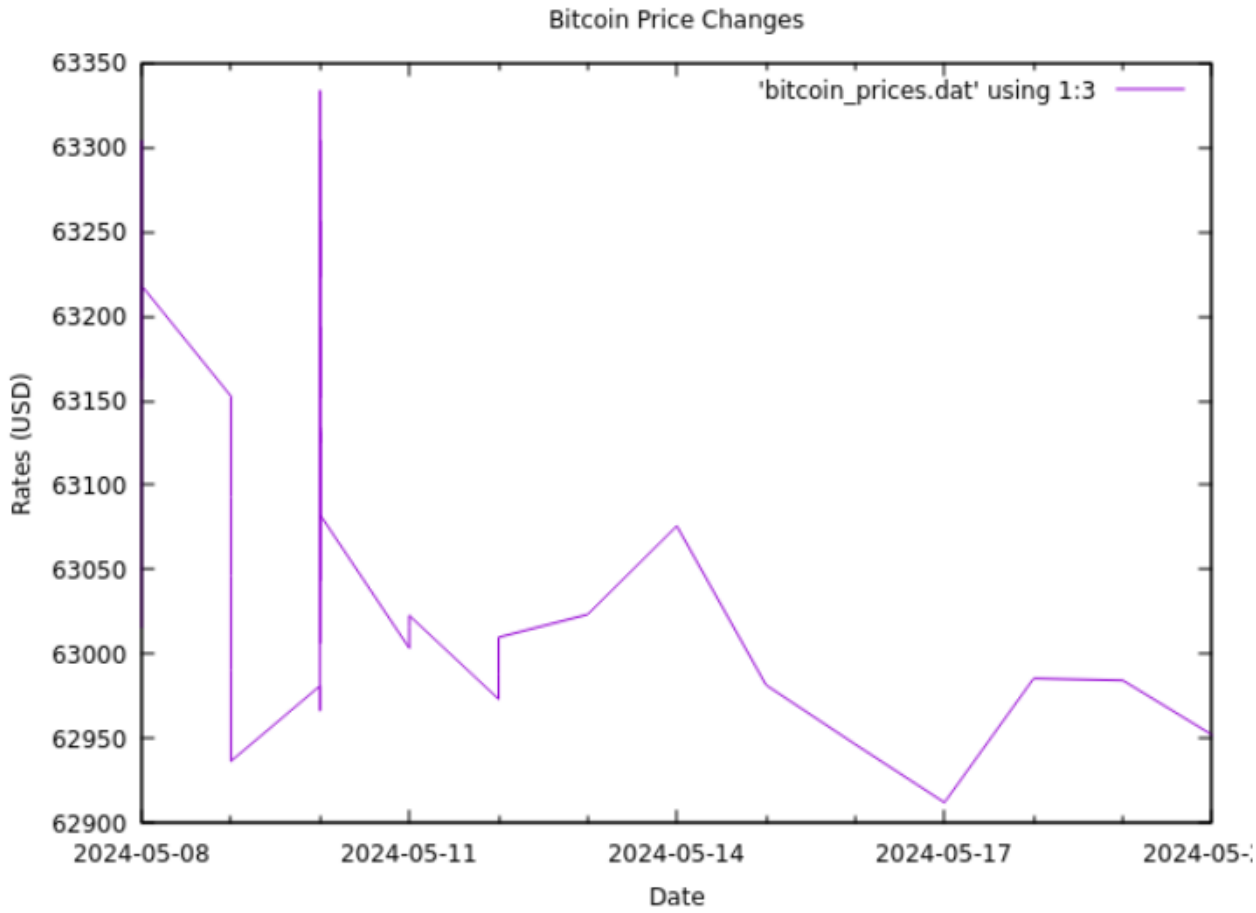

Figure 3.1

4.0 Crontab entry

```
0 * * * * /home/pi/scripts/bitcoin_tracker.sh
0 * * * * /home/pi/scripts/crypto_rates_tracker.sh
```

This code specifies the timing of the task. In this case, the code will execute two scripts, `bitcoin_tracker.sh` and `crypto_rates_tracker.sh`, every hour by adding the new records to the SQL database.

5.0 MySQL

SQL queries to create tables:

```
30      CREATE TABLE `bitcoin_prices` (
31        `id` int(11) NOT NULL,
32        `current_price` decimal(10,2) DEFAULT NULL,
33        `low_24h` decimal(10,2) DEFAULT NULL,
34        `high_24h` decimal(10,2) DEFAULT NULL,
35        `timestamp` timestamp NOT NULL DEFAULT current_timestamp()
36      ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

100     CREATE TABLE `cryptocurrency_rates` (
101       `id` int(11) NOT NULL,
102       `crypto_name` varchar(50) NOT NULL,
103       `crypto_symbol` varchar(10) NOT NULL,
104       `current_price` decimal(18,8) NOT NULL,
105       `timestamp` timestamp NOT NULL DEFAULT current_timestamp()
106     ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

125     CREATE TABLE `cryptocurrency_rates_history` (
126       `id` int(11) NOT NULL,
127       `crypto_id` int(11) DEFAULT NULL,
128       `previous_price` decimal(18,6) DEFAULT NULL,
129       `previous_timestamp` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE current_timestamp()
130     ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

6.0 GitHub

Use of Git for version control. Commits over 10.