**How to load stock data into mysql:**

First, login into IB TWS platform on desktop.

Next, open “get\_ib\_historical\_data.py” file to pull data from Interactive Brokers into CSV files. Need to make sure contract list, duration, bar size, and end date are updated for new load. CSV files are uploaded to: C:\ProgramData\MySQL\MySQL Server 8.0\Uploads. When program updated, run program to see files in above folder.

Next, us “working\_csv\_to\_mysql\_V3.py” file to load csv files into mysql. The connection detail to Mysql is below and already captured in program:

#mysql configuration items for connection

host = '127.0.0.1'

user = 'root'

password = 'EptL@Rl!1'

database = 'Stock\_Prices'

table = '5\_min\_prices'

After running this file, and confirming that data is in mysql (see section further below), run trading program to backtest “Working Trading ModelvXX.py”. Make sure to adjust dates to account for new data if you wish test recently downloaded data.

**Access mysql:**

User name is ‘root’, and password is 'EptL@Rl!1'.

Cheat Sheet to access tables and check that data was loaded:

**mysql> show databases;**

+--------------------+

| Database |

+--------------------+

| information\_schema |

| menagerie |

| mysql |

| performance\_schema |

| sakila |

| stock\_prices |

| sys |

| teststocksdb |

| vsearchlogdb |

| world |

+--------------------+

10 rows in set (0.00 sec)

mysql> **use stock\_prices;**

Database changed

**mysql> show tables;**

+------------------------+

| Tables\_in\_stock\_prices |

+------------------------+

| 5\_min\_prices |

+------------------------+

1 row in set (0.00 sec)

**mysql> SELECT COUNT(\*) FROM 5\_min\_prices;**

+----------+

| COUNT(\*) |

+----------+

| 1494155 |

+----------+

1 row in set (0.80 sec)

**mysql> select \*from 5\_min\_prices ORDER BY date DESC LIMIT 1;**

**This returns last row of table:**

+--------+---------------------+---------+---------+---------+---------+--------+---------------------+---------------------+------------------------+

| ticker | date | open | high | low | close | volume | created | source | key |

+--------+---------------------+---------+---------+---------+---------+--------+---------------------+---------------------+------------------------+

| XRT | 2018-09-28 14:55:00 | 51.0200 | 51.0600 | 51.0100 | 51.0500 | 3959 | 2018-09-30 17:17:18 | Interactive Brokers | XRT 20180928 14:55:00 |

+--------+---------------------+---------+---------+---------+---------+--------+---------------------+---------------------+------------------------+

1 row in set (1.93 sec)

**mysql> select date from 5\_min\_prices ORDER BY date ASC LIMIT 1;**

**This returns first row of table:**

**Output into csv file:**

mysql> SELECT \* FROM 5\_min\_prices WHERE date >= '2018-07-01' and date <='2018-08-30' INTO OUTFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/erikquery.csv';

**Search by date:**

mysql> SELECT \* FROM 5\_min\_prices WHERE date = '2018-09-28 14:55:00';

+--------+---------------------+----------+----------+----------+----------+--------+---------------------+---------------------+------------------------+

| ticker | date | open | high | low | close | volume | created | source | key |

+--------+---------------------+----------+----------+----------+----------+--------+---------------------+---------------------+------------------------+

| AGG | 2018-09-28 14:55:00 | 105.5500 | 105.5500 | 105.5100 | 105.5300 | 2513 | 2018-09-30 15:41:40 | Interactive Brokers | AGG 20180928 14:55:00 |

| DBA | 2018-09-28 14:55:00 | 16.9000 | 16.9100 | 16.8900 | 16.9100 | 419 | 2018-09-30 16:55:18 | Interactive Brokers | DBA 20180928 14:55:00 |

| EMB | 2018-09-28 14:55:00 | 107.8300 | 107.8700 | 107.8100 | 107.8100 | 2471 | 2018-09-30 15:52:44 | Interactive Brokers | EMB 20180928 14:55:00 |

| EWH | 2018-09-28 14:55:00 | 23.9400 | 23.9800 | 23.9300 | 23.9800 | 3199 | 2018-09-30 16:59:54 | Interactive Brokers | EWH 20180928 14:55:00 |

| EWW | 2018-09-28 14:55:00 | 51.1900 | 51.2000 | 51.1400 | 51.1700 | 2383 | 2018-09-30 17:04:07 | Interactive Brokers | EWW 20180928 14:55:00 |

| EWZ | 2018-09-28 14:55:00 | 33.7000 | 33.7800 | 33.6800 | 33.7700 | 9890 | 2018-09-30 16:17:46 | Interactive Brokers | EWZ 20180928 14:55:00 |

| HYG | 2018-09-28 14:55:00 | 86.4500 | 86.4500 | 86.4300 | 86.4500 | 13186 | 2018-09-30 17:08:42 | Interactive Brokers | HYG 20180928 14:55:00 |

| IAU | 2018-09-28 14:55:00 | 11.4300 | 11.4300 | 11.4200 | 11.4300 | 1700 | 2018-09-30 15:32:45 | Interactive Brokers | IAU 20180928 14:55:00 |

| LQD | 2018-09-28 14:55:00 | 114.9300 | 114.9300 | 114.9000 | 114.9300 | 1634 | 2018-09-30 16:13:09 | Interactive Brokers | LQD 20180928 14:55:00 |

| MCD | 2018-09-28 14:55:00 | 167.1400 | 167.3500 | 167.1300 | 167.2800 | 1541 | 2018-09-30 16:51:01 | Interactive Brokers | MCD 20180928 14:55:00 |

| SHY | 2018-09-28 14:55:00 | 83.0900 | 83.1000 | 83.0900 | 83.1000 | 283 | 2018-09-30 17:23:03 | Interactive Brokers | SHY 20180928 14:55:00 |

| SPY | 2018-09-28 14:55:00 | 290.4700 | 290.6700 | 290.4100 | 290.6100 | 59200 | 2018-09-30 12:26:56 | Interactive Brokers | SPY 20180928 14:55:00 |

| TIP | 2018-09-28 14:55:00 | 110.6200 | 110.6300 | 110.6000 | 110.6200 | 956 | 2018-09-30 15:37:03 | Interactive Brokers | TIP 20180928 14:55:00 |

| VNQ | 2018-09-28 14:55:00 | 80.5900 | 80.6800 | 80.5800 | 80.6700 | 3353 | 2018-09-30 15:57:08 | Interactive Brokers | VNQ 20180928 14:55:00 |

| XHB | 2018-09-28 14:55:00 | 38.4300 | 38.4400 | 38.4000 | 38.4200 | 1620 | 2018-09-30 17:27:30 | Interactive Brokers | XHB 20180928 14:55:00 |

| XLK | 2018-09-28 14:55:00 | 75.2600 | 75.3200 | 75.2400 | 75.3200 | 8824 | 2018-09-30 17:21:55 | Interactive Brokers | XLK 20180928 14:55:00 |

| XLU | 2018-09-28 14:55:00 | 52.6400 | 52.6900 | 52.6300 | 52.6800 | 17305 | 2018-09-30 16:01:50 | Interactive Brokers | XLU 20180928 14:55:00 |

| XLV | 2018-09-28 14:55:00 | 95.0600 | 95.1300 | 95.0300 | 95.1100 | 4831 | 2018-09-30 17:13:00 | Interactive Brokers | XLV 20180928 14:55:00 |

| XOM | 2018-09-28 14:55:00 | 85.1100 | 85.1600 | 84.9900 | 85.0200 | 7887 | 2018-09-30 16:08:51 | Interactive Brokers | XOM 20180928 14:55:00 |

| XRT | 2018-09-28 14:55:00 | 51.0200 | 51.0600 | 51.0100 | 51.0500 | 3959 | 2018-09-30 17:17:18 | Interactive Brokers | XRT 20180928 14:55:00 |

+--------+---------------------+----------+----------+----------+----------+--------+---------------------+---------------------+------------------------+

20 rows in set (1.50 sec)

**mysql> describe 5\_min\_prices;**

+---------+---------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+---------+---------------+------+-----+---------+-------+

| ticker | varchar(32) | NO | | NULL | |

| date | datetime | NO | | NULL | |

| open | decimal(19,4) | YES | | NULL | |

| high | decimal(19,4) | YES | | NULL | |

| low | decimal(19,4) | YES | | NULL | |

| close | decimal(19,4) | YES | | NULL | |

| volume | bigint(20) | YES | | NULL | |

| created | datetime | NO | | NULL | |

| source | varchar(32) | NO | | NULL | |

| key | varchar(64) | NO | PRI | NULL | |

+---------+---------------+------+-----+---------+-------+

10 rows in set (0.00 sec)

#Add new column to existing table:

alter table 5\_min\_prices add column time datetime;

#Populate new column with data from an existing date column

Update 5\_min\_prices set time = date;

#Delete all records in time column of 5\_min\_prices table (by setting all records to NULL):

UPDATE 5\_min\_prices SET time = Null;

#Delete column from table

ALTER TABLE 5\_min\_prices DROP COLUMN time;

#Format new column into just time:

SELECT \* FROM 5\_min\_prices WHERE DATE\_FORMAT(date, “%H:%i”)=”08:55”;

SELECT \* FROM 5\_min\_prices WHERE DATE\_FORMAT(date, “%H:%i”) between ”08:55” and “09:05”;

SELECT date,open,high,low,close,volume FROM 5\_min\_prices WHERE ticker = “VIX” AND date between "2016-01-01" and “2016-01-08” AND DATE\_FORMAT(date, “%H:%i”) between ”08:30” and “14:55” ORDER BY date ASC;

Query String for Mysql file:

self.result = self.conn.execute("SELECT date,open,high,low,close,volume FROM {} WHERE ticker = '{}' AND date between '{}' and '{}' AND DATE\_FORMAT(date,'{}') between '{}' and '{}' ORDER BY date ASC".format(self.p.table,symbol\_string,self.p.fromdate.strftime("%Y-%m-%d"),self.p.todate.strftime("%Y-%m-%d"),"%%H",self.p.sessionstart.strftime("%H"),self.p.sessionend.strftime("%H")))

Hit CTRL and C to kill running mysql query

SELECT \* FROM 5\_min\_prices WHERE date >= '2017-04-03' and date <='2017-04-20' and ticker = 'SPY';

Select from a few different tickers at once:

select \* from 5\_min\_prices where ticker IN('SPY','XLU','TICK-NYSE') and date >= '2017-10-02' and date <='2017-12-15' INTO OUTFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/erikprices.csv';

To see the index for a specific table use SHOW INDEX:

SHOW INDEX FROM 5\_min\_prices;

+--------------+------------+-----------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+

| Table | Non\_unique | Key\_name | Seq\_in\_index | Column\_name | Collation | Cardinality | Sub\_part | Packed | Null | Index\_type | Comment | Index\_comment | Visible | Expression |

+--------------+------------+-----------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+

| 5\_min\_prices | 0 | PRIMARY | 1 | key | A | 1982158 | NULL | NULL | | BTREE | | | YES | NULL |

| 5\_min\_prices | 1 | idx\_stock | 1 | ticker | A | 4633 | NULL | NULL | | BTREE | | | YES | NULL |

| 5\_min\_prices | 1 | idx\_stock | 2 | date | A | 1982158 | NULL | NULL | | BTREE | | | YES | NULL |

| 5\_min\_prices | 1 | idx\_stock | 3 | open | A | 1982158 | NULL | NULL | YES | BTREE | | | YES | NULL |

| 5\_min\_prices | 1 | idx\_stock | 4 | high | A | 1982158 | NULL | NULL | YES | BTREE | | | YES | NULL |

| 5\_min\_prices | 1 | idx\_stock | 5 | low | A | 1982158 | NULL | NULL | YES | BTREE | | | YES | NULL |

| 5\_min\_prices | 1 | idx\_stock | 6 | close | A | 1982158 | NULL | NULL | YES | BTREE | | | YES | NULL |

+--------------+---

CREATE INDEX:

CREATE INDEX indx\_stock ON *5\_min\_prices(ticker,date,open,high,low,close,volume)*

DROP INDEX:

DROP INDEX *idx\_stock* ON *5\_min\_prices;*

Check for duplicates:

SELECT date, COUNT(\*) AS CountOf FROM 5\_min\_prices WHERE ticker = “DBA” GROUP BY date HAVING COUNT(\*) > 1;

SHOW INDEX from 5\_min\_prices;

List of Ticker in Database as of 4/18/2020

| mysql> SELECT ticker, MAX(date), MIN(date) from 5\_min\_prices GROUP BY ticker;

+-----------+---------------------+---------------------+

| ticker | MAX(date) | MIN(date) |

+-----------+---------------------+---------------------+

| AGG | 2019-04-05 14:55:00 | 2014-09-30 08:30:00 |

| DBA | 2019-04-05 14:55:00 | 2014-10-02 08:30:00 |

| EMB | 2019-04-05 14:55:00 | 2017-08-03 08:30:00 |

| EWH | 2019-04-05 14:55:00 | 2014-09-30 08:30:00 |

| EWW | 2019-04-05 14:55:00 | 2014-09-30 08:30:00 |

| EWZ | 2019-04-05 14:55:00 | 2014-09-30 08:30:00 |

| HYG | 2019-04-05 14:55:00 | 2014-09-30 08:30:00 |

| IAU | 2019-04-05 14:55:00 | 2014-09-30 08:30:00 |

| LQD | 2019-04-05 14:55:00 | 2014-09-30 08:30:00 |

| MCD | 2019-04-05 14:55:00 | 2014-09-30 08:30:00 |

| SHY | 2019-04-05 14:55:00 | 2017-08-03 08:30:00 |

| SPY | 2019-04-05 14:55:00 | 2012-10-01 08:30:00 |

| TICK-NYSE | 2019-07-02 14:55:00 | 2014-07-07 08:30:00 |

| TIP | 2019-04-05 14:55:00 | 2014-09-30 08:30:00 |

| TRIN-NYSE | 2019-07-02 14:55:00 | 2014-07-07 08:30:00 |

| VIX | 2019-07-02 15:10:00 | 2014-07-02 08:30:00 |

| VNQ | 2019-04-05 14:55:00 | 2014-09-30 08:30:00 |

| XHB | 2019-04-05 14:55:00 | 2014-09-30 08:30:00 |

| XLK | 2019-04-05 14:55:00 | 2014-09-30 08:30:00 |

| XLU | 2019-04-05 14:55:00 | 2014-09-30 08:30:00 |

| XLV | 2019-04-05 14:55:00 | 2014-09-30 08:30:00 |

| XOM | 2019-04-05 14:55:00 | 2014-09-30 08:30:00 |

| XRT | 2019-04-05 14:55:00 | 2014-09-30 08:30:00 |

+-----------+---------------------+---------------------+

23 rows in set (4.01 sec)

#CREATE TABLE:

CREATE TABLE econ (

id INT(6) UNSIGNED AUTO\_INCREMENT PRIMARY KEY,

date DATETIME NOT NULL,

econ VARCHAR(32) NOT NULL,

close DECIMAL(19,4) NULL

);

CREATE INDEX indx\_econ ON *economics (date,econ,close);*

Modify fieldname in table:

ALTER TABLE econ

MODIFY close DECIMAL(19,2) NULL;

ALTER TABLE economics CHANGE econ name varchar(32) NOT NULL;

CREATE TABLE fundamentals (

id INT(6) UNSIGNED AUTO\_INCREMENT PRIMARY KEY,

date DATETIME NOT NULL,

eps\_estimate NUMERIC (19,4) NOT NULL,

eps\_actual NUMERIC (19,4) NOT NULL,

eps\_diff NUMERIC (19,4) NOT NULL,

source VARCHAR(32) NOT NULL);

CREATE INDEX indx\_fund ON fundamentals*(ticker,datetime,eps\_estimate,eps\_actual,eps\_diff);*

SHOW INDEX from fundamentals;

DROP INDEX *indx\_fund* ON *fundamentals;*

SELECT IFNULL(eps\_estimate, 'N/A') AS eps\_estimate, IFNULL(eps\_actual,'N/A') AS eps\_actual, IFNULL(eps\_diff, 'N/A') AS eps\_diff FROM fundamentals;