```sql
-- C.Advanced Challenge
-- 1.In addition to the built-in aggregate functions, explore ways to calculate other key statistics about the data, such as the median or variance.
SELECT stock_symbol, stock_price
FROM stocks_tbl
ORDER BY stock_price
LIMIT 1
OFFSET (SELECT COUNT(*)
        FROM stocks_tbl) / 2
;

-- 2.Let’s refactor the data into 2 tables - stock_info to store general info about the stock itself (ie. symbol, name) and stock_prices to store the collected data on price (ie. symbol, datetime, price).
-- Creation of new stock_info table that stores the general info about the stock, like the symbol and the name, with stock_symbol as the primary key. Records under this new table will be inserted from the original stocks_tbl table.
CREATE TABLE "stock_info" (
    "stock_symbol"  TEXT NOT NULL,
    "stock_name"    TEXT,
    PRIMARY KEY("stock_symbol")
);

SELECT count(*)
FROM stock_info;

SELECT *
FROM stock_info;

-- Insert records to stock_info table from the original stocks_tbl table, but we are only inserting unique records. With that said, there should only be 5 records to be inserted to stock_info.
INSERT INTO stock_info (stock_symbol, stock_name)
SELECT DISTINCT stock_symbol, stock_name FROM stocks_tbl;

SELECT *
FROM stock_info;

-- Creation of new stock_prices table that stores the collected data on price about the stock. Columns will include the stock_symbol, stock_price, dttm_stamp, and effective_date, with stock_symbol as the primary key. Records under this new table will be coming from the original stocks_tbl table and will be inserted upon the creation of the new table.
SELECT * FROM stocks_tbl;

CREATE TABLE stock_prices AS
SELECT stock_symbol, stock_price, dttm_stamp, effective_date
FROM stocks_tbl;

SELECT count(*)
FROM stock_prices;

SELECT *
FROM stock_prices;

-- 3.Now, we do not need to repeat both symbol and name for each row of price data. Instead, join the 2 tables in order to view more information on the stock with each row of price.
SELECT *
FROM stock_info
JOIN stock_prices
    ON stock_info.stock_symbol = stock_prices.stock_symbol;

SELECT stock_info.stock_symbol, stock_info.stock_name, count(*)
FROM stock_info
JOIN stock_prices
    ON stock_info.stock_symbol = stock_prices.stock_symbol
GROUP BY stock_info.stock_symbol, stock_info.stock_name;

-- 4.Add more variables to the stock_info table and update the data (e.g., sector, industry, etc).
-- We will be executing an ALTER clause to add some columns to an existing table stock_info, such as stock_exchange and stock_sector. We will also be including some information about the company.
SELECT *
FROM stock_info;

ALTER TABLE stock_info
ADD "stock_exchange" TEXT;
```

```sql

ALTER TABLE stock_info
ADD "stock_sector" TEXT;

ALTER TABLE stock_info
ADD "company_ceo" TEXT;

ALTER TABLE stock_info
ADD "company_founded" date;

ALTER TABLE stock_info
ADD "company_headquarters" TEXT;

ALTER TABLE stock_info
ADD "company_website" TEXT;

ALTER TABLE stock_info
ADD "company_employes" INTEGER;

SELECT * FROM stock_info;

-- We will be executing an UPDATE clause to update the value of the newly added columns.
UPDATE stock_info
SET stock_sector = "Transportation",
    company_ceo = "Dara Khosrowshahi",
    company_founded = "2009-03-01",
    company_headquarters = "San Francisco, California, United States",
    company_website = "uber.com",
    company_employes = 32200
WHERE stock_symbol = "UBER";

UPDATE stock_info
SET stock_exchange =
    CASE
        WHEN stock_symbol = "UBER" THEN "NYSE"
        ELSE "NASDAQ"
    END;

SELECT *
FROM stock_info;
```