DATABASE CREATION AND TESTING

SUBMISSION 1

DAN
GRIZLI777

Contents

Creating the database

Creating the database was performed in Microsoft SQL Server by right clicking the databases folder in the tree and selecting new database. Within the new database wizard I gave the new database the name “Upskilled_NYSE_DB”.

Creating the tables

New tables were created for DailyPrices and StockNames by right clicking the tables folder located within the new database and selecting new table for each.

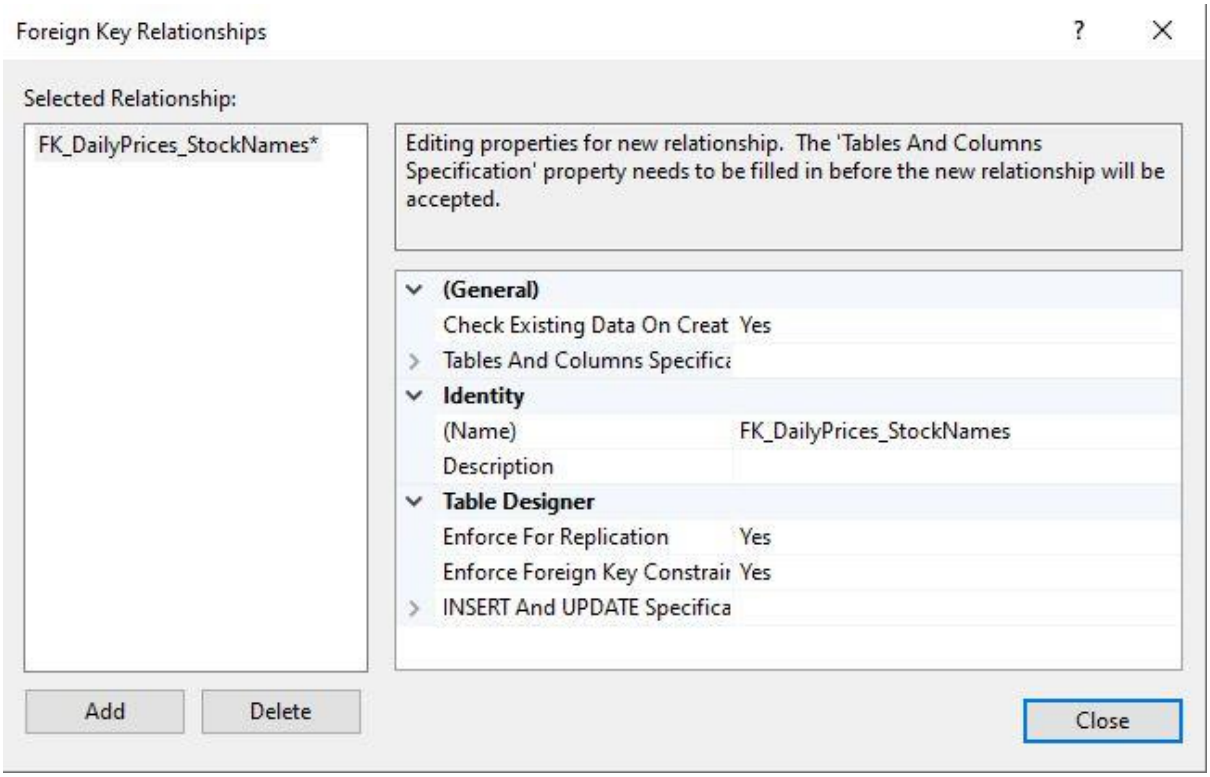
The following design and attributes were input as follows, the first one for DailyPrices:

Column Name	Data Type	Allow Nulls
StockSymbol	char(3)	☐
Date	date	☐
StockPriceOpen	float	☑
StockPriceClose	float	☑
StockPriceLow	float	☑
StockPriceHigh	float	☑
StockPriceAdjClose	float	☑
StockVolume	int	☑
		☐

And the second one for StockNames:

Column Name	Data Type	Allow Nulls
StockSymbol	char(3)	☐
StockName	varchar(100)	☑
StockExchange	char(4)	☑
		☐

Once both tables had been saved and finalized, I added the one to many relationship between the foreign key StockSymbol field in DailyPrices and the primary key StockSymbol field in StockNames using the relationships wizard.



Importing data and populating fields

We imported the data into SQL Server by using the Import Flat File wizard, accessible via right clicking on the database and navigating into tasks. Each .csv file was imported into a new table via the wizard.

Once we had the data in the database we needed to move it from the new tables into our DailyPrices and StockNames tables.

We wrote the following TSQL code to perform the operation for

StockNames:

```
INSERT INTO StockNames (StockSymbol, StockName, StockExchange)

SELECT stock_symbol, stock_name, stock_exchange FROM NYSE_stock_names;
```

DailyPrices:

```
INSERT INTO DailyPrices (Date, StockSymbol, StockPriceOpen, StockPriceClose,
StockPriceLo, StockPriceHigh, StockPriceAdjClose, StockVolume)

SELECT date, stock_symbol, stock_price_open, stock_price_close, stock_price_low,
stock_price_high, stock_price_adj_close, stock_volume FROM NYSE_daily_prices_A;
```

The StockNames table was chosen to populate first due the foreign key constraint in DailyPrices requiring this to be so. The table was populated without error.

StockSymbol	StockName	StockExchange
AA	Automatic Aromas Intl	NYSE
ABB	Abbacaad	NYSE
ABK	American Beech and Keise	NYSE
ABM	Automated Business Machines	NYSE
ACH	American Career Helpers	NYSE
ACM	Anderson Charles and Mitchell	NYSE
ADI	Andreeli Systems	NYSE
ADY	Aardvaarky	NYSE
AEA	American External Accessories	NYSE
AEB	AEB Metals	NYSE

We encountered a problem when populating the DailyPrices table due to there being a missing StockSymbol in the StockNames data. To find the missing entry, we used an EXCEPT statement on the two data tables which returned “AOL” to be missing from the StockNames table. After adding an entry for “AOL” into the StockNames table, we were able to populate the DailyPrices table without error.

StockSymbol	Date	StockPriceOpen	StockPriceClose	StockPriceLow	StockPriceHigh	StockPriceAdj...	StockVolume
AA	2000-01-03	83	80.94	80.37	83.56	32.52	3103200
AA	2000-01-04	80.94	81.31	80.31	81.81	32.67	4469600
AA	2000-01-05	81.31	86	81	86.5	34.55	6243200
AA	2000-01-06	86	84.87	84.81	86.37	34.1	8989400
AA	2000-01-07	84.87	84.62	84.56	86.87	34	9069400
AA	2000-01-10	84.62	84.37	84.37	87.25	33.9	7671600
AA	2000-01-11	84.37	83.87	82	84.5	33.7	4463200
AA	2000-01-12	83.87	83.12	82.75	83.87	33.4	3747200

Test Plan

The following procedures were tested using T-SQL to ensure that the database performs the required functions, and disallows certain data entry if the conditions are not met.

The T-SQL code performing the tests will be included for completeness.

Entries can be added to StockNames table

```
INSERT INTO dbo.StockNames (StockSymbol, StockName, StockExchange)
VALUES ('FFF', 'Five Five Four', 'NYSE');

SELECT StockSymbol, StockName, StockExchange
```

```
FROM dbo.StockNames
WHERE StockSymbol = 'FFF';
```

This test returned the new entry correctly and verifies that entries are able to be made into the StockNames table.

Entries can be added to DailyPrices table

```
INSERT INTO DailyPrices (Date, StockSymbol, StockPriceOpen, StockPriceClose,
StockPriceLow, StockPriceHigh, StockPriceAdjClose, StockVolume)
VALUES (GETDATE(), 'FFF', 34.4, 36.7, 36.7, 32.1, 37.7, 1200);

SELECT Date, StockSymbol, StockPriceOpen, StockPriceClose, StockPriceLow,
StockPriceHigh, StockPriceAdjClose, StockVolume
FROM dbo.DailyPrices
WHERE DailyPrices.StockSymbol = 'FFF';
```

This test returned the new entry correctly and verifies that entries are able to be made into the DailyPrices table.

Entries without StockSymbol cannot be added to StockName

```
INSERT INTO dbo.StockNames (StockName, StockExchange)
VALUES ('Applied Crumbs', 'NYSE');
```

SQL Server disallows the entry to be made due to the StockSymbol column disallowing null values. Test passes.

Entries without StockSymbol or date cannot be added to DailyPrices

```
INSERT INTO DailyPrices (StockSymbol, StockPriceOpen, StockPriceClose, StockPriceLow,
StockPriceHigh, StockPriceAdjClose, StockVolume)
VALUES ('CGG', 34.4, 36.7, 36.7, 32.1, 37.7, 1200);
```

SQL Server disallows the entry to be made due to the date column disallowing null values. Test passes.

```
INSERT INTO DailyPrices (Date, StockPriceOpen, StockPriceClose, StockPriceLow,
StockPriceHigh, StockPriceAdjClose, StockVolume)
VALUES (GETDATE(), 34.4, 36.7, 36.7, 32.1, 37.7, 1200);
```

SQL Server disallows the entry to be made due to the StockSymbol column disallowing null values. Test passes.

All entries in DailyPrices for a specific stock can be returned

```
SELECT *
FROM dbo.DailyPrices
WHERE DailyPrices.StockSymbol = 'AEG';
```

SQL Server returns all entries for AEG, test passes.

All entries in DailyPrices with a closing price of over 25 can be returned

```
SELECT *
FROM dbo.DailyPrices
WHERE DailyPrices.StockPriceClose > 25;
```

SQL Server returns all rows where the StockPriceClose column is over 25, test passes.

All entries in StockSymbol with “America” in the name are returned

```
SELECT *
FROM dbo.StockNames
WHERE StockNames.StockName LIKE '%america%';
```

SQL Server returns all results with “America” in the StockName column, test passes.