NYU | TANDON SCHOOL OF ENGINEERING

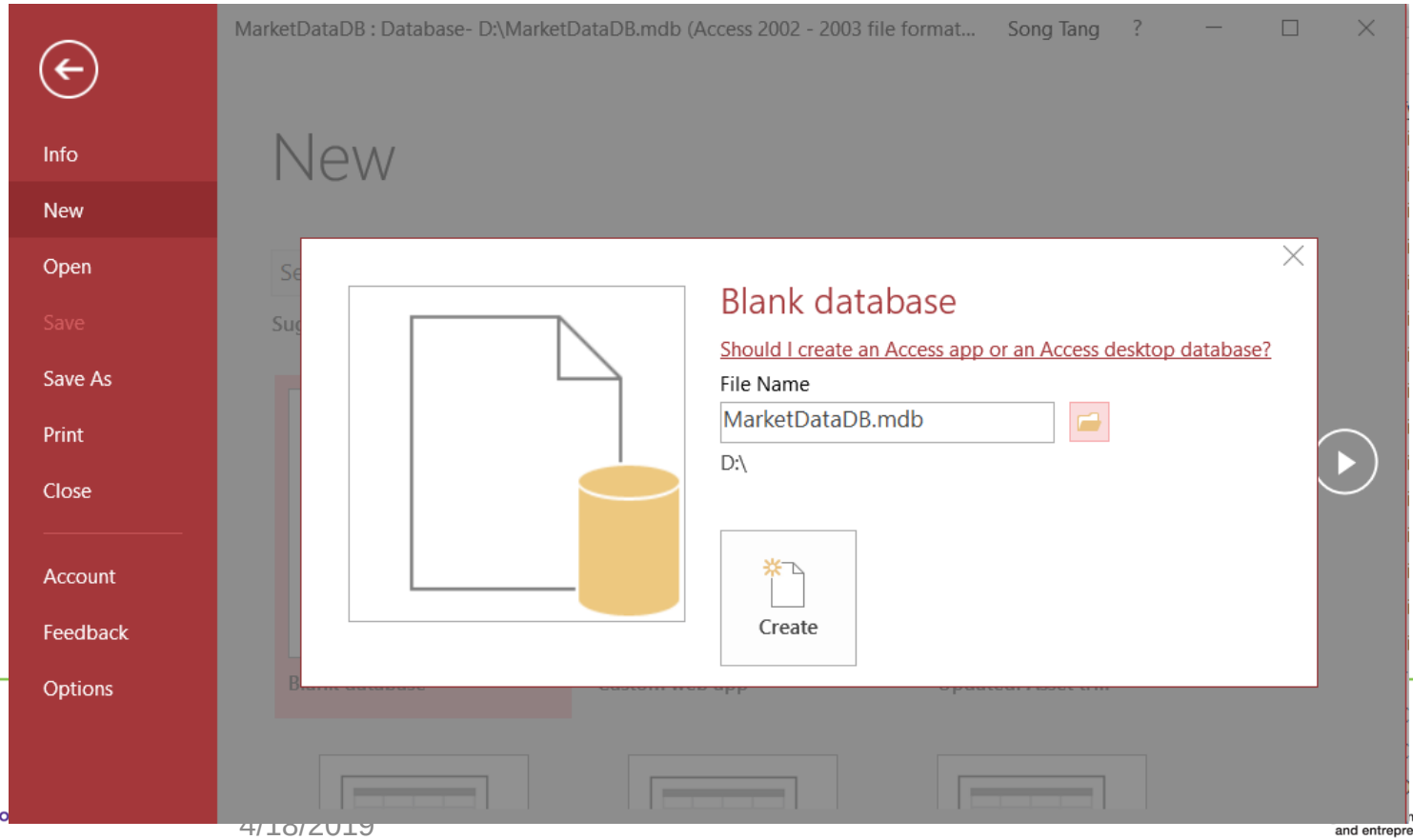
BLPAPI-RDB

RDB-BLPAPI

- Create a relational database in Microsoft Access for market data with the following tables:
  - Markets
  - Issuers
  - Stocks
  - Daily Data
  - Intraday Data
- Retrieve Daily and Intraday Data from BLPAPI to populate the data tables.

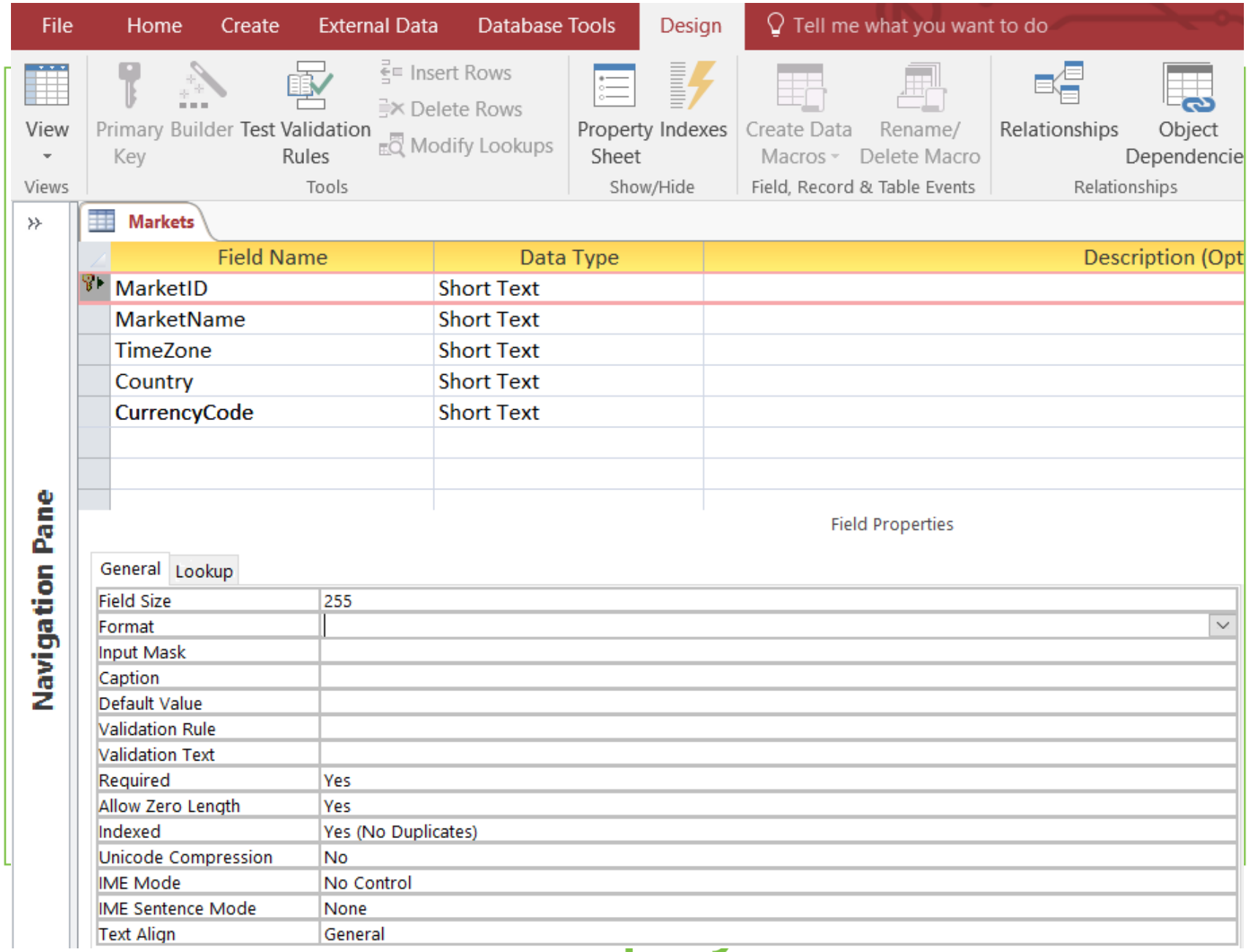
Create Market Data RDB

- Use Microsoft Access Database to create a database called D:\MarketDataDB.mdb
- Open Microsoft Access: File->New->Blank Database



Create Fields in a table

- Select Table Design from Create
- Enter field name
  - Must be unique, but only within the same table
- Select field type from a menu
  - Use date/time for times
  - Use text for phone numbers
- Designate primary key (right mouse button)
- Save the table
  - That's when you get to assign a table name

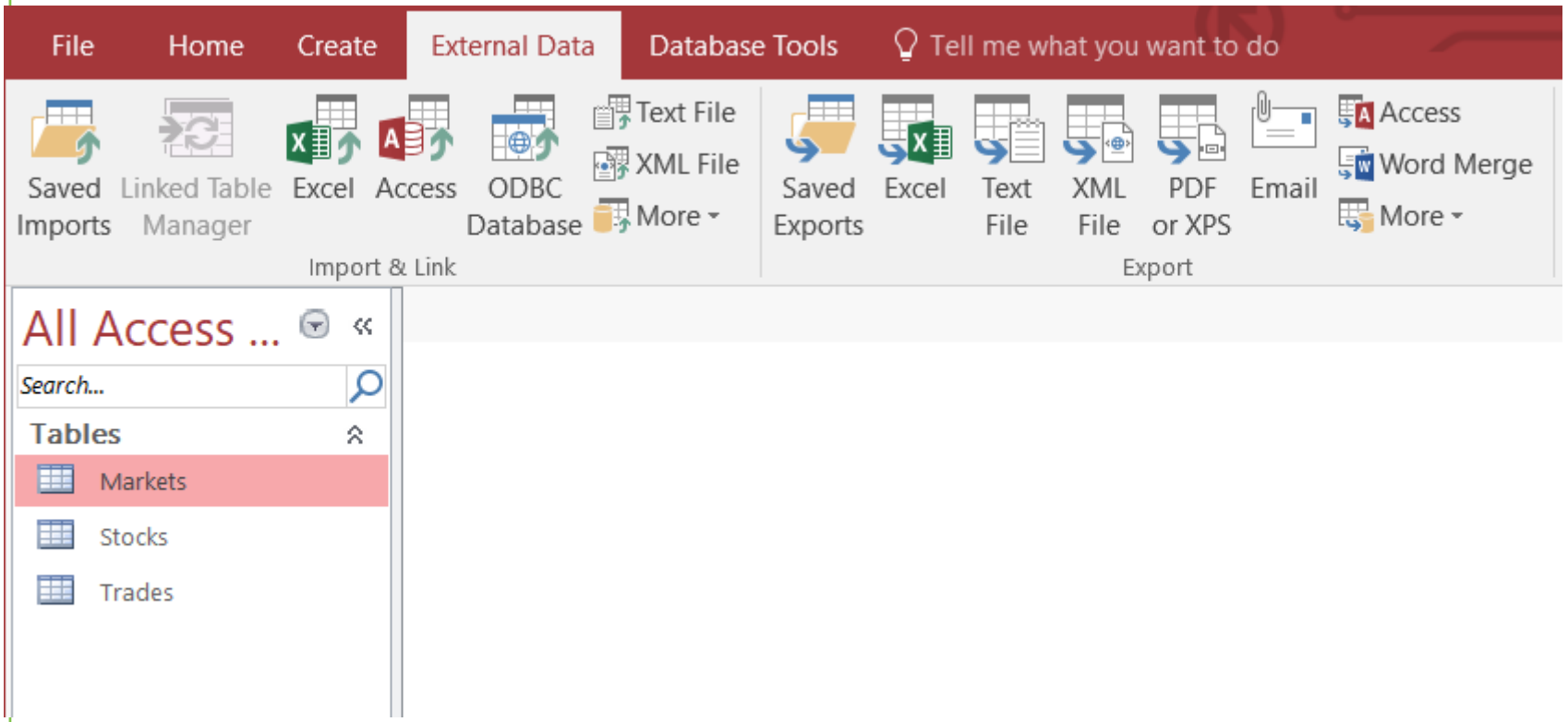

File
Home
Create
External Data
Database Tools
Design
Tell me what you want to do
View
Primary Key
Builder
Test Validation Rules
Insert Rows
Delete Rows
Modify Lookups
Property Sheet
Indexes
Create Data Macros
Rename/ Delete Macro
Relationships
Object Dependencie
Views
Tools
Show/Hide
Field, Record & Table Events
Relationships
Navigation Pane
Markets
Field Name
Data Type
Description (Opt
MarketID
Short Text
MarketName
Short Text
TimeZone
Short Text
Country
Short Text
CurrencyCode
Short Text
Field Properties
General
Lookup
Field Size
255
Format
Input Mask
Caption
Default Value
Validation Rule
Validation Text
Required
Yes
Allow Zero Length
Yes
Indexed
Yes (No Duplicates)
Unicode Compression
No
IME Mode
No Control
IME Sentence Mode
None
Text Align
General

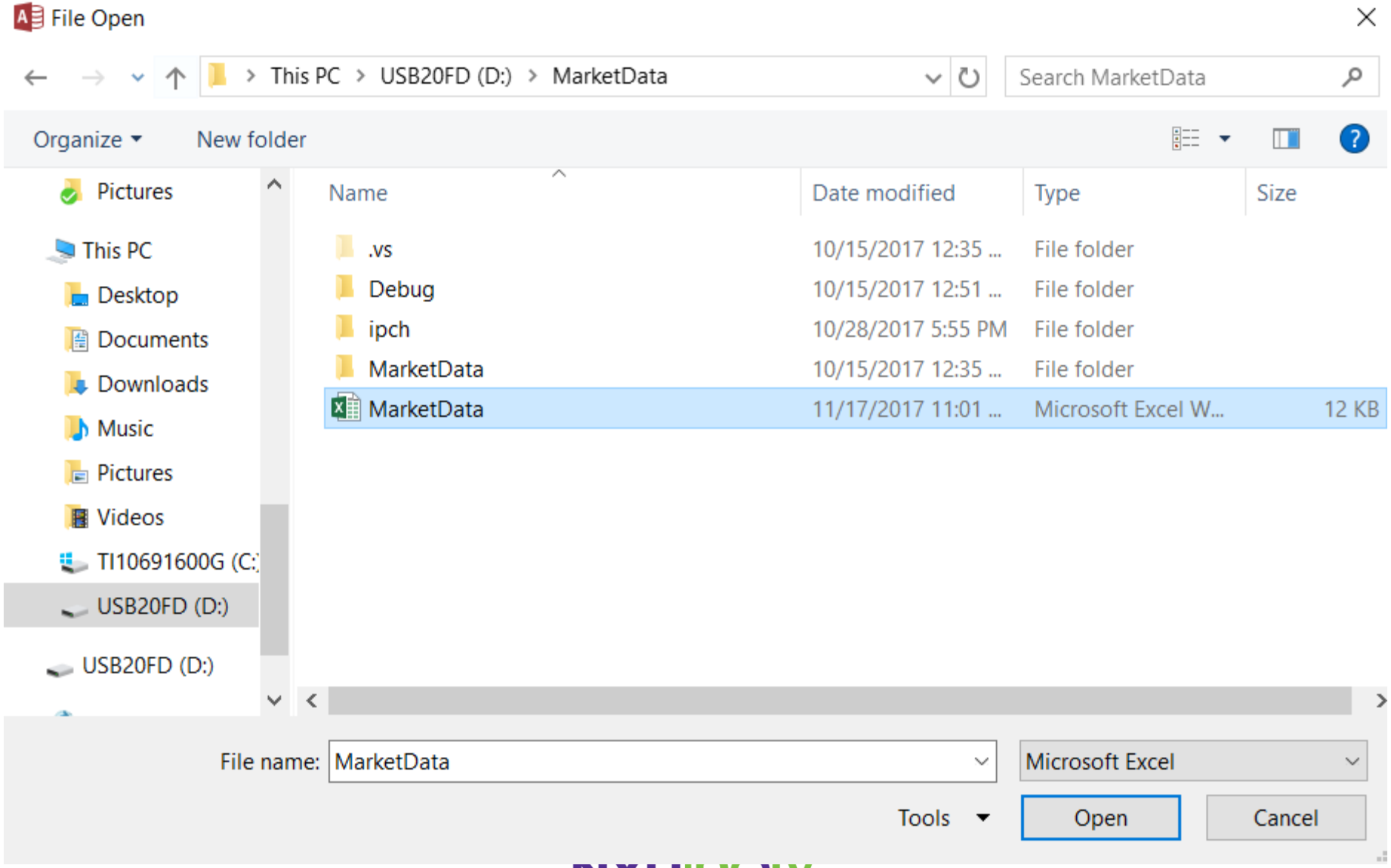
Enter Data

- Open the table
  - Double-click on the icon
- Enter new data in the bottom row
  - A new (blank) bottom row will appear
- Close the table
  - No need to “save” – data is stored automatically
- Import data from an Excel sheet

Input Data from Excel Sheets

- Close your table first and select Excel as External Data for Market table.




File Open
This PC
USB20FD (D:)
MarketData
Search MarketData
Organize
New folder
Pictures
This PC
Desktop
Documents
Downloads
Music
Pictures
Videos
TI10691600G (C:)
USB20FD (D:)
USB20FD (D:)
Name
Date modified
Type
Size
.vs
10/15/2017 12:35 ...
File folder
Debug
10/15/2017 12:51 ...
File folder
ipch
10/28/2017 5:55 PM
File folder
MarketData
10/15/2017 12:35 ...
File folder
MarketData
11/17/2017 11:01 ...
Microsoft Excel W...
12 KB
File name:
MarketData
Microsoft Excel
Tools
Open
Cancel

Get External Data - Excel Spreadsheet

Select the source and destination of the data

Specify the source of the definition of the objects.

File name: C:\Users\Song\Documents\NYU-Poly\PolyNYU2018\FRE7831_Spring2018\MarketData\MarketData.xlsx Browse...

Specify how and where you want to store the data in the current database.

- ○ **Import the source data into a new table in the current database.**
  If the specified table does not exist, Access will create it. If the specified table already exists, Access might overwrite its contents with the imported data. Changes made to the source data will not be reflected in the database.
- ◉ **Append a copy of the records to the table:** Markets
  If the specified table exists, Access will add the records to the table. If the table does not exist, Access will create it. Changes made to the source data will not be reflected in the database.
- ○ **Link to the data source by creating a linked table.**
  Access will create a table that will maintain a link to the source data in Excel. Changes made to the source data in Excel will be reflected in the linked table. However, the source data cannot be changed from within Access.

OK Cancel

Import Spreadsheet Wizard

Your spreadsheet file contains more than one worksheet or range. Which worksheet or range would you like?

- (•) Show Worksheets
- ( ) Show Named Ranges

Stocks
Markets
Issuers

Sample data for worksheet 'Markets'.

1	MarketID	MarketName	TimeZone	Country	CurrencyCode
2	HK	Hong Kong Stock Exchange	HKT	China	HKD
3	IN	Nationa Stock Exchange	IST	India	INR
4	N	New York Stock Exchange	EST	United States	USD
5	O	NASDAQ Stock Exchange	EST	United States	USD
6	OTCM	OTC Markets Group Inc	EST	United States	USD
7	SG	Singapore Exchange	SST	Singapore	SGD
8	SH	Shanghai Stock Exchange	CST	China	CNY
9	SZ	Shenzhen Stock Exchange	CST	China	CNY
10	T	Toronto Stock Exchange	EST	Canada	CAD
11	TO	Tokyo Stock Exchange	JST	Japan	JPY

Cancel | < Back | Next > | Finish

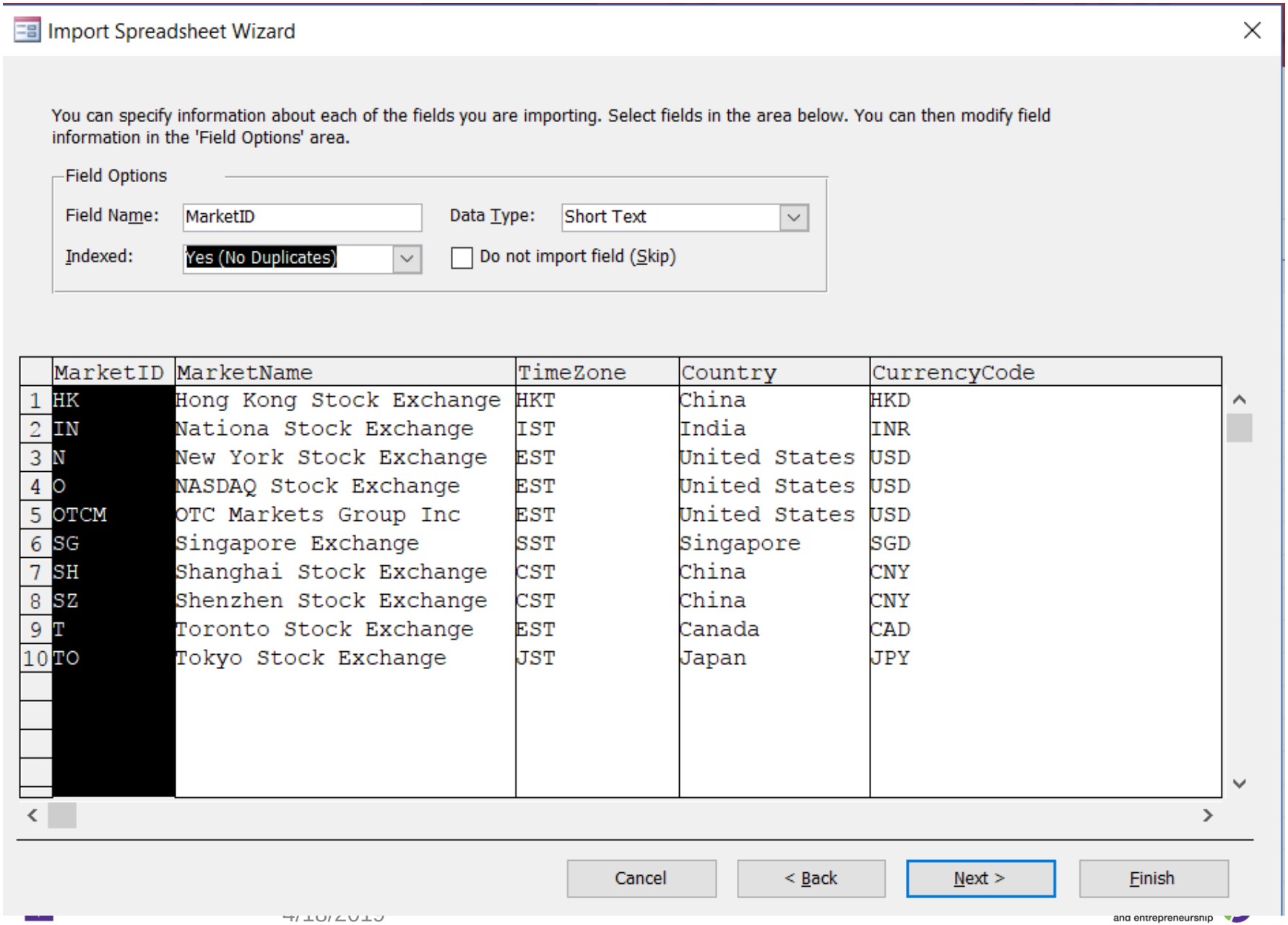
Import Spreadsheet Wizard

Microsoft Access can use your column headings as field names for your table. Does the first row specified contain column headings?

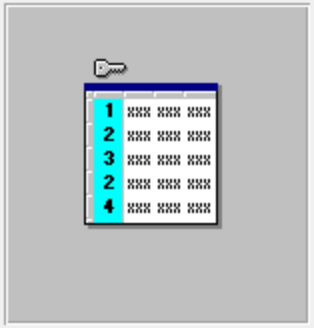
☑ First Row Contains Column Headings

	MarketID	MarketName	TimeZone	Country	CurrencyCode
1	HK	Hong Kong Stock Exchange	HKT	China	HKD
2	IN	Nationa Stock Exchange	IST	India	INR
3	N	New York Stock Exchange	EST	United States	USD
4	O	NASDAQ Stock Exchange	EST	United States	USD
5	OTCM	OTC Markets Group Inc	EST	United States	USD
6	SG	Singapore Exchange	SST	Singapore	SGD
7	SH	Shanghai Stock Exchange	CST	China	CNY
8	SZ	Shenzhen Stock Exchange	CST	China	CNY
9	T	Toronto Stock Exchange	EST	Canada	CAD
10	TO	Tokyo Stock Exchange	JST	Japan	JPY

Cancel | < Back | Next > | Finish


Import Spreadsheet Wizard
You can specify information about each of the fields you are importing. Select fields in the area below. You can then modify field information in the 'Field Options' area.
Field Options
Field Name: MarketID
Data Type: Short Text
Indexed: Yes (No Duplicates)
Do not import field (Skip)
MarketID MarketName TimeZone Country CurrencyCode
1 HK Hong Kong Stock Exchange HKT China HKD
2 IN Nationa Stock Exchange IST India INR
3 N New York Stock Exchange EST United States USD
4 O NASDAQ Stock Exchange EST United States USD
5 OTCM OTC Markets Group Inc EST United States USD
6 SG Singapore Exchange SST Singapore SGD
7 SH Shanghai Stock Exchange CST China CNY
8 SZ Shenzhen Stock Exchange CST China CNY
9 T Toronto Stock Exchange EST Canada CAD
10 TO Tokyo Stock Exchange JST Japan JPY
Cancel
< Back
Next >
Finish

Import Spreadsheet Wizard



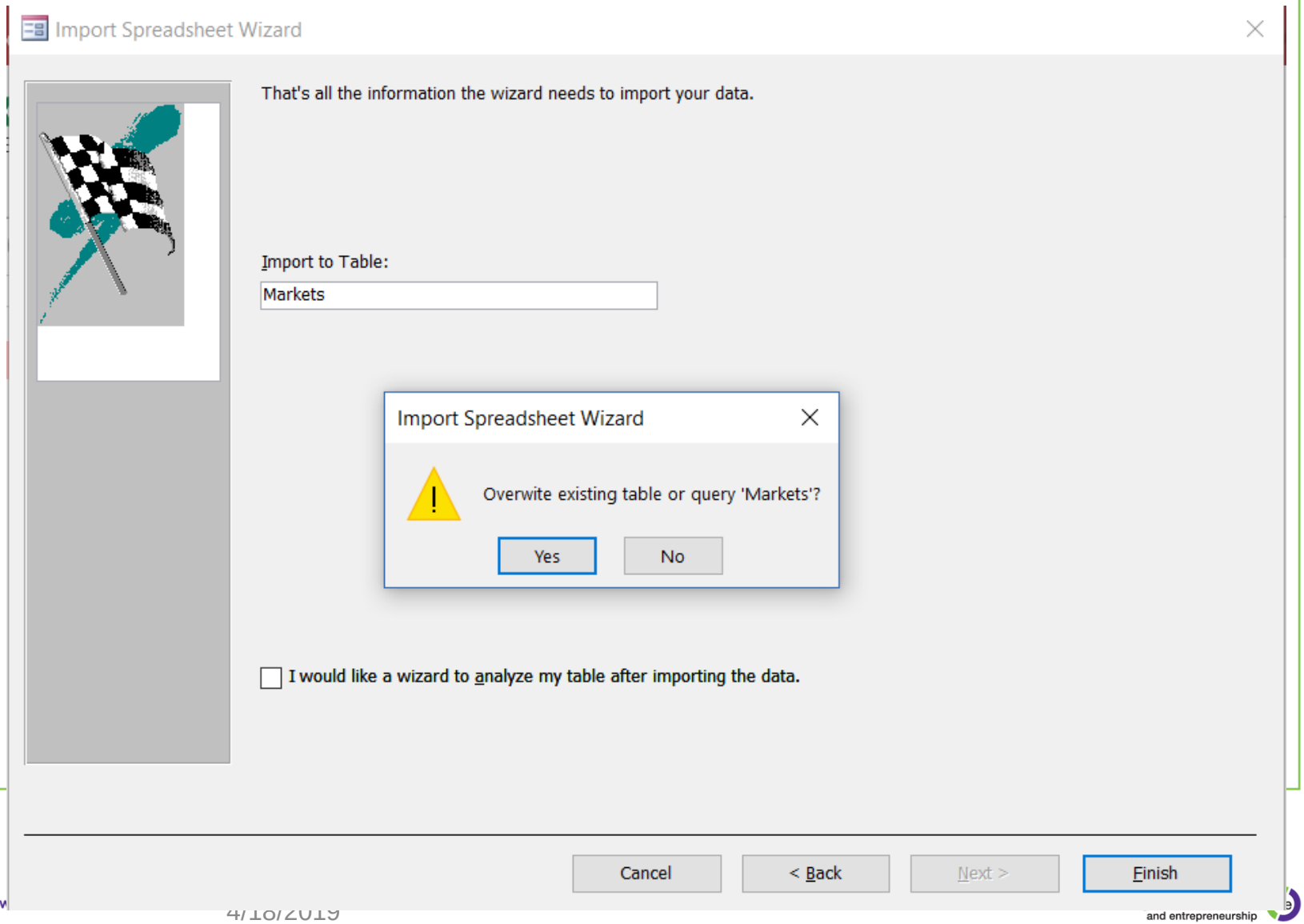
Microsoft Access recommends that you define a primary key for your new table. A primary key is used to uniquely identify each record in your table. It allows you to retrieve data more quickly.

- ○ Let Access add primary key.
- ◉ Choose my own primary key. MarketID
- ○ No primary key.

	MarketID	MarketName	TimeZone	Country	CurrencyCode
1	HK	Hong Kong Stock Exchange	HKT	China	HKD
2	IN	Nationa Stock Exchange	IST	India	INR
3	N	New York Stock Exchange	EST	United States	USD
4	O	NASDAQ Stock Exchange	EST	United States	USD
5	OTCM	OTC Markets Group Inc	EST	United States	USD
6	SG	Singapore Exchange	SST	Singapore	SGD
7	SH	Shanghai Stock Exchange	CST	China	CNY
8	SZ	Shenzhen Stock Exchange	CST	China	CNY
9	T	Toronto Stock Exchange	EST	Canada	CAD
10	TO	Tokyo Stock Exchange	JST	Japan	JPY

Cancel | < Back | Next > | Finish

Click Yes to import the data from Excel sheet



Import Spreadsheet Wizard

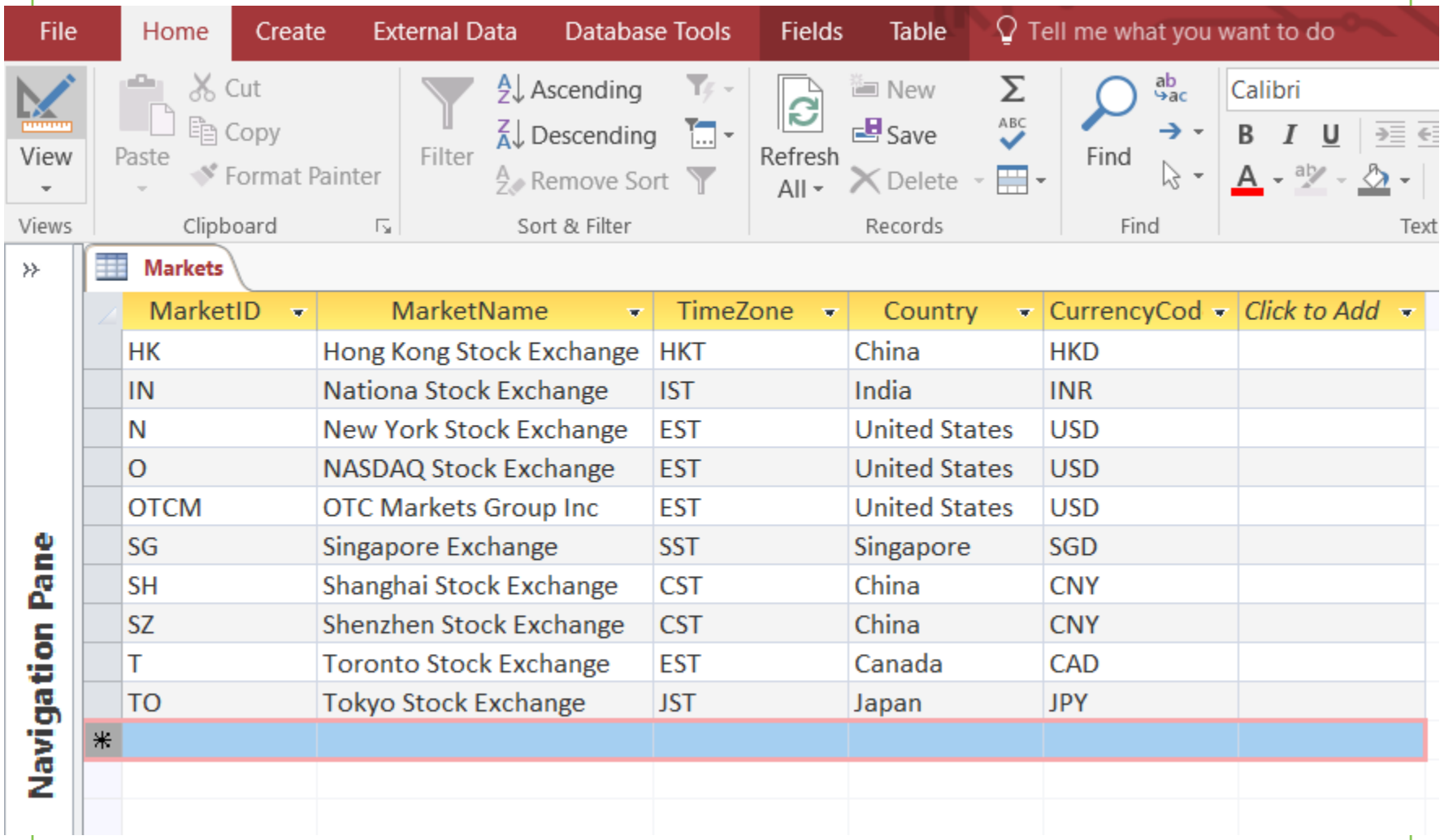
That's all the information the wizard needs to import your data.

Import to Table:

Markets

I would like a wizard to analyze my table after importing the data.

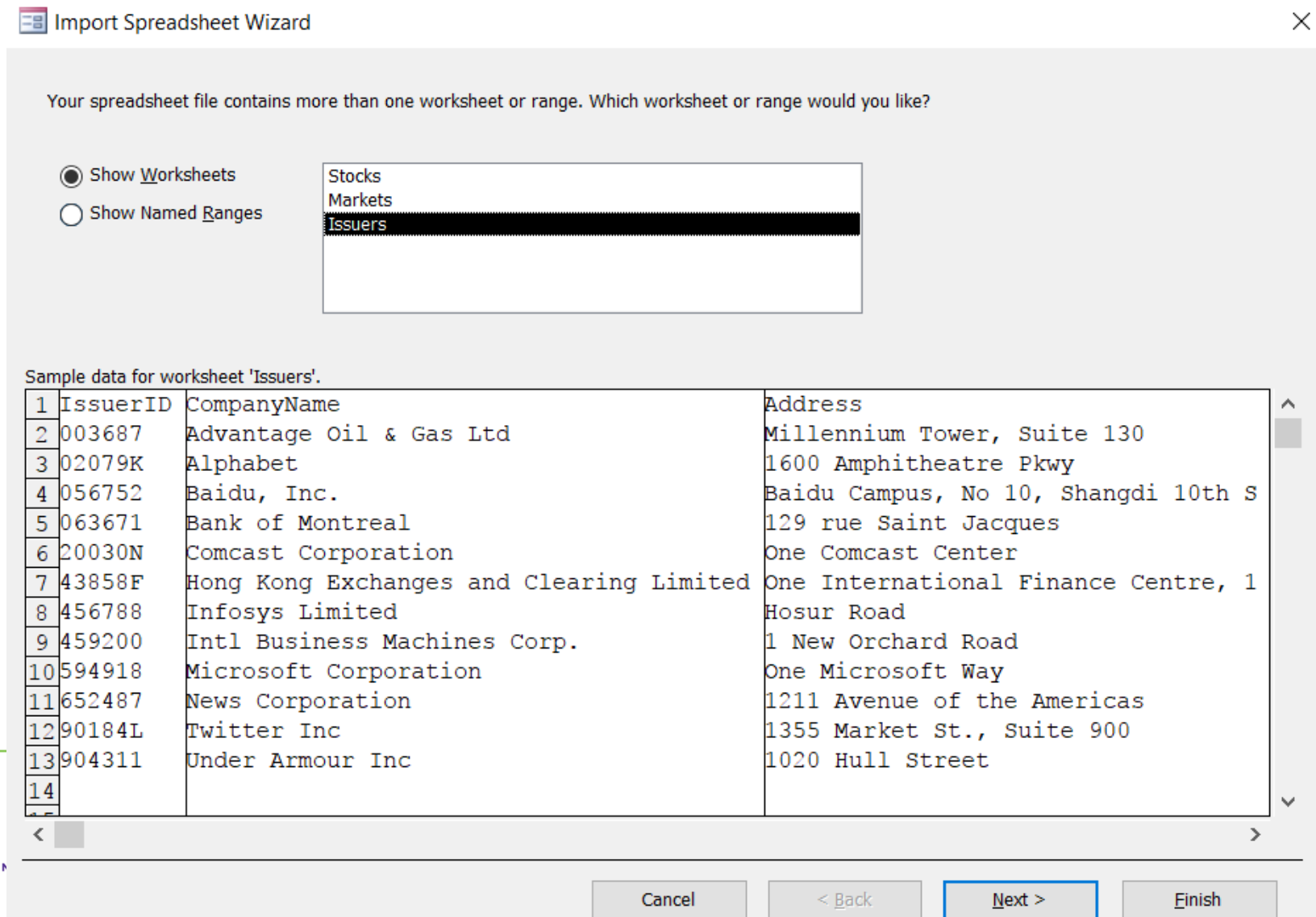
Cancel | < Back | Next > | Finish



MarketID	MarketName	TimeZone	Country	CurrencyCod	Click to Add
HK	Hong Kong Stock Exchange	HKT	China	HKD	
IN	Nationa Stock Exchange	IST	India	INR	
N	New York Stock Exchange	EST	United States	USD	
O	NASDAQ Stock Exchange	EST	United States	USD	
OTCM	OTC Markets Group Inc	EST	United States	USD	
SG	Singapore Exchange	SST	Singapore	SGD	
SH	Shanghai Stock Exchange	CST	China	CNY	
SZ	Shenzhen Stock Exchange	CST	China	CNY	
T	Toronto Stock Exchange	EST	Canada	CAD	
TO	Tokyo Stock Exchange	JST	Japan	JPY	

Repeat the Same Steps for the Table Issuers

- Import Excel sheet Issuers without creating an empty table in Access



Imported table from Excel Sheet Issuers

Markets | Issuers

IssuerID	CompanyName	Address	City	State	ZipCode	Country
003687	Advantage Oil & Gas Ltd	Millennium Tower, Suite 1	Calgary	AB	T2P 5E9	Canada
02079K	Alphabet	1600 Amphitheatre Pkwy	Mountain View	CA	94043	United States
056752	Baidu, Inc.	Baidu Campus, No 10, Sha	Beijing		100085	China
063671	Bank of Montreal	129 rue Saint Jacques	Montreal	QC	H2Y 1L6	Canada
20030N	Comcast Corporation	One Comcast Center	Philadelphia	PA	19103	United States
43858F	Hong Kong Exchanges and Clea	One International Finance	Hong Kong			China
456788	Infosys Limited	Hosur Road	Electronics City	Bengaluru	560100	India
459200	Intl Business Machines Corp.	1 New Orchard Road	Armonk	NY	10504	United States
594918	Microsoft Corporation	One Microsoft Way	Redmond	WA	98052	United States
652487	News Corporation	1211 Avenue of the Ameri	New York	NY	10036	United States
90184L	Twitter Inc	1355 Market St., Suite 900	San Francisco	CA	94103	United States
904311	Under Armour Inc	1020 Hull Street	Baltimore	MD	21230	United States

Add Restriction to Table Fields

- Verify IssuerID is the primary key and its length is 6

Markets | Issuers

Field Name	Data Type	Description (Optiona
IssuerID	Short Text	
CompanyName	Short Text	
Address	Short Text	
City	Short Text	
State	Short Text	
ZipCode	Short Text	
Country	Short Text	

Field Properties

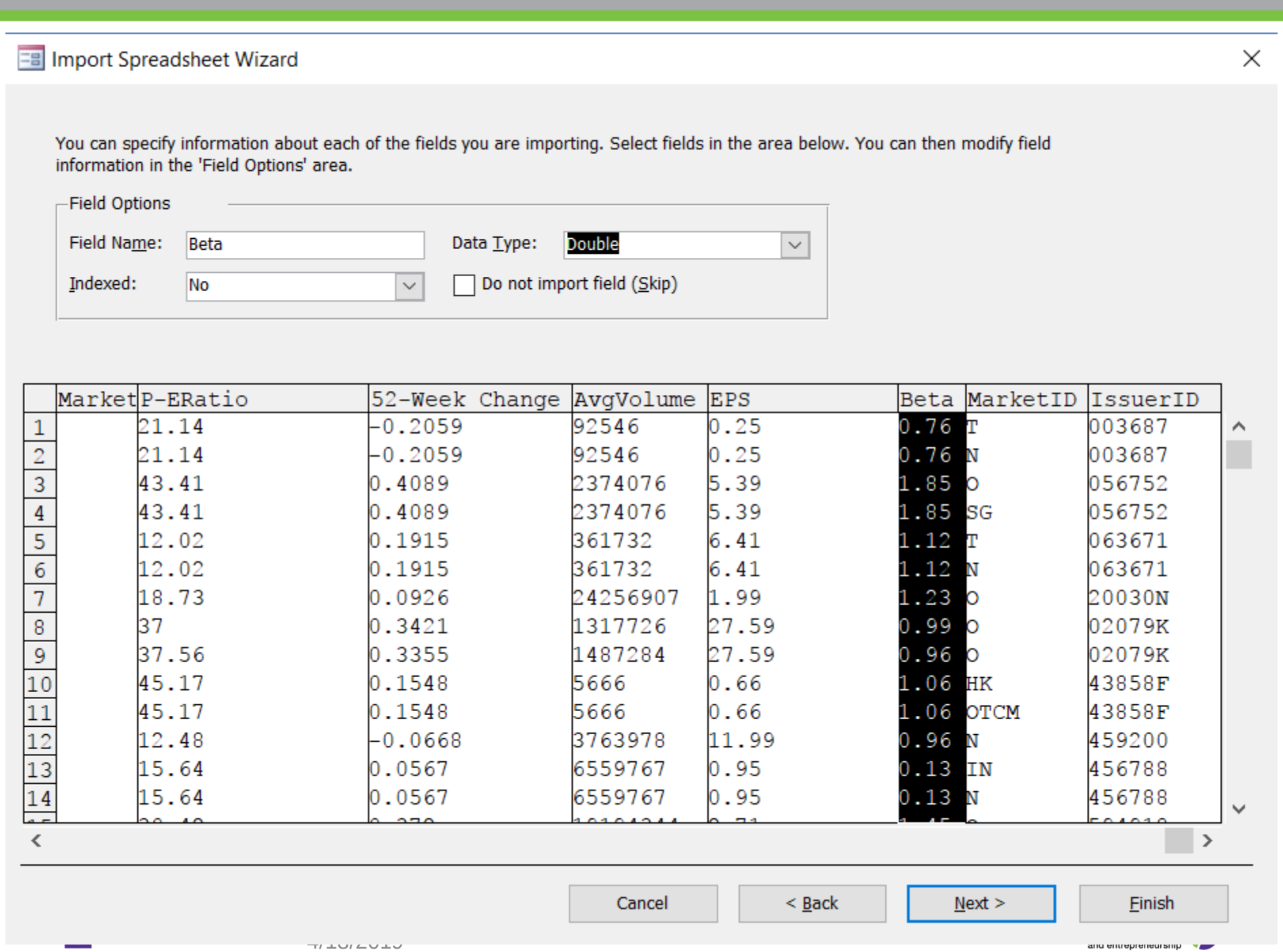
General | Lookup

Field Size	6
Format	@
Input Mask	
Caption	
Default Value	
Validation Rule	
Validation Text	
Required	Yes
Allow Zero Length	Yes
Indexed	Yes (No Duplicates)
Unicode Compression	No
IME Mode	No Control
IME Sentence Mode	None
Text Align	General

Continue with Stock Table

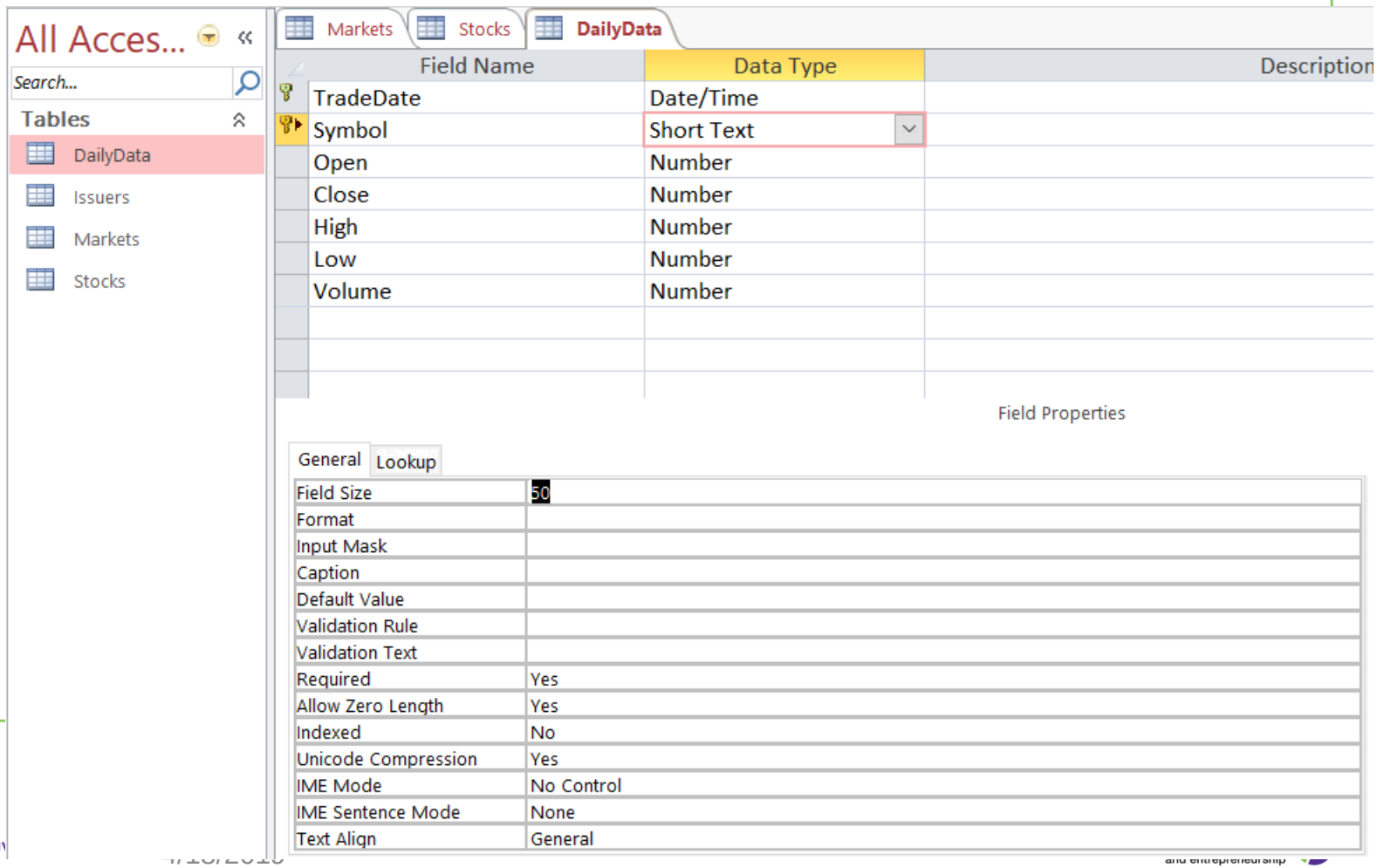
- Create Stock table and Make sure the date type of each field matches the following before importing data from the Excel sheet.

Field Name	Data Type
Symbol	Short Text
Cusip	Short Text
MarketCap	Currency
P-ERatio	Number
52-Week Change	Number
AvgVolume	Number
EPS	Currency
Beta	Number
MarketID	Short Text
IssuerID	Short Text


Import Spreadsheet Wizard
You can specify information about each of the fields you are importing. Select fields in the area below. You can then modify field information in the 'Field Options' area.
Field Options
Field Name: Beta
Data Type: Double
Indexed: No
Do not import field (Skip)
Market P-ERatio 52-Week Change AvgVolume EPS Beta MarketID IssuerID
1 21.14 -0.2059 92546 0.25 0.76 T 003687
2 21.14 -0.2059 92546 0.25 0.76 N 003687
3 43.41 0.4089 2374076 5.39 1.85 O 056752
4 43.41 0.4089 2374076 5.39 1.85 SG 056752
5 12.02 0.1915 361732 6.41 1.12 T 063671
6 12.02 0.1915 361732 6.41 1.12 N 063671
7 18.73 0.0926 24256907 1.99 1.23 O 20030N
8 37 0.3421 1317726 27.59 0.99 O 02079K
9 37.56 0.3355 1487284 27.59 0.96 O 02079K
10 45.17 0.1548 5666 0.66 1.06 HK 43858F
11 45.17 0.1548 5666 0.66 1.06 OTCM 43858F
12 12.48 -0.0668 3763978 11.99 0.96 N 459200
13 15.64 0.0567 6559767 0.95 0.13 IN 456788
14 15.64 0.0567 6559767 0.95 0.13 N 456788
Cancel
< Back
Next >
Finish

Create DailyData Table

- Hold Ctl key to select both TradeData and Symbol as the composite Primary Key.



Create IntradyDayData table

Markets | Issuers | Stocks | DailyData | **IntradayData**

Field Name	Data Type	Description (Optional)
TradeTime	Number	
BidPrice	Number	
BitQty	Number	
AskPrice	Number	
AskQty	Number	
TradePrice	Number	
TradeQty	Number	
TradeDate	Date/Time	
Symbol	Short Text	

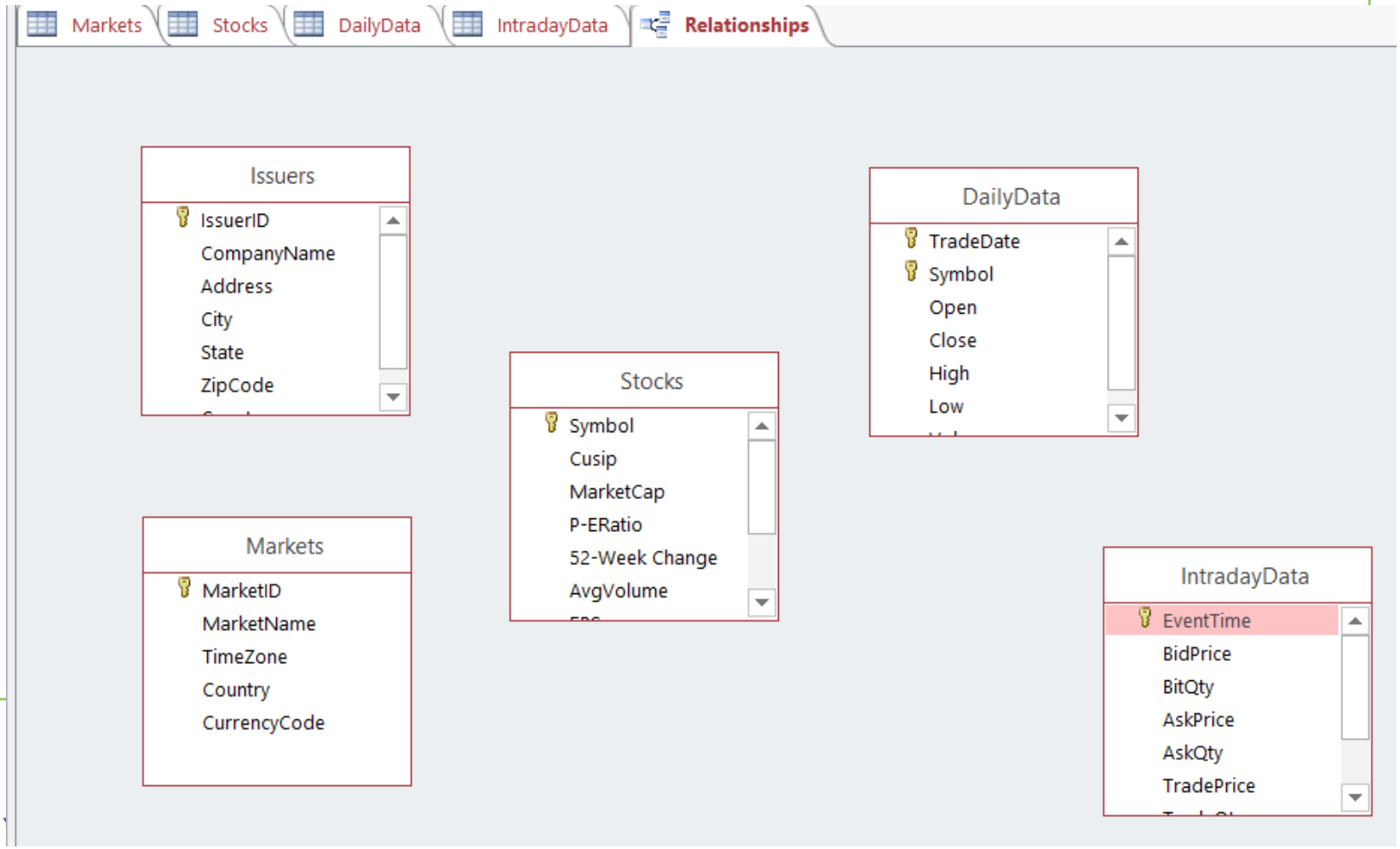
Field Properties

General | Lookup

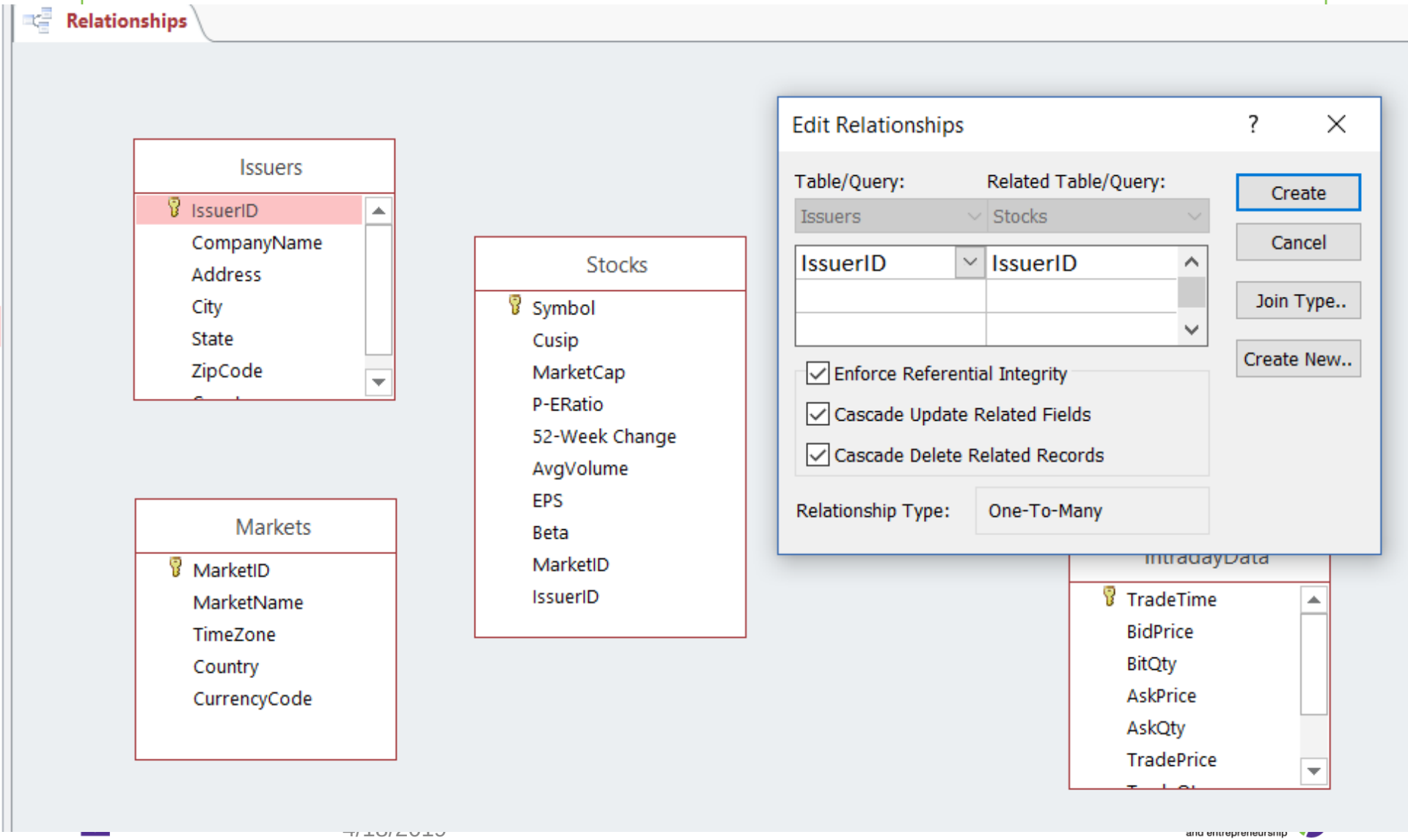
Field Size	Long Integer
Format	
Decimal Places	0
Input Mask	
Caption	
Default Value	0
Validation Rule	
Validation Text	
Required	Yes
Indexed	No
Text Align	General

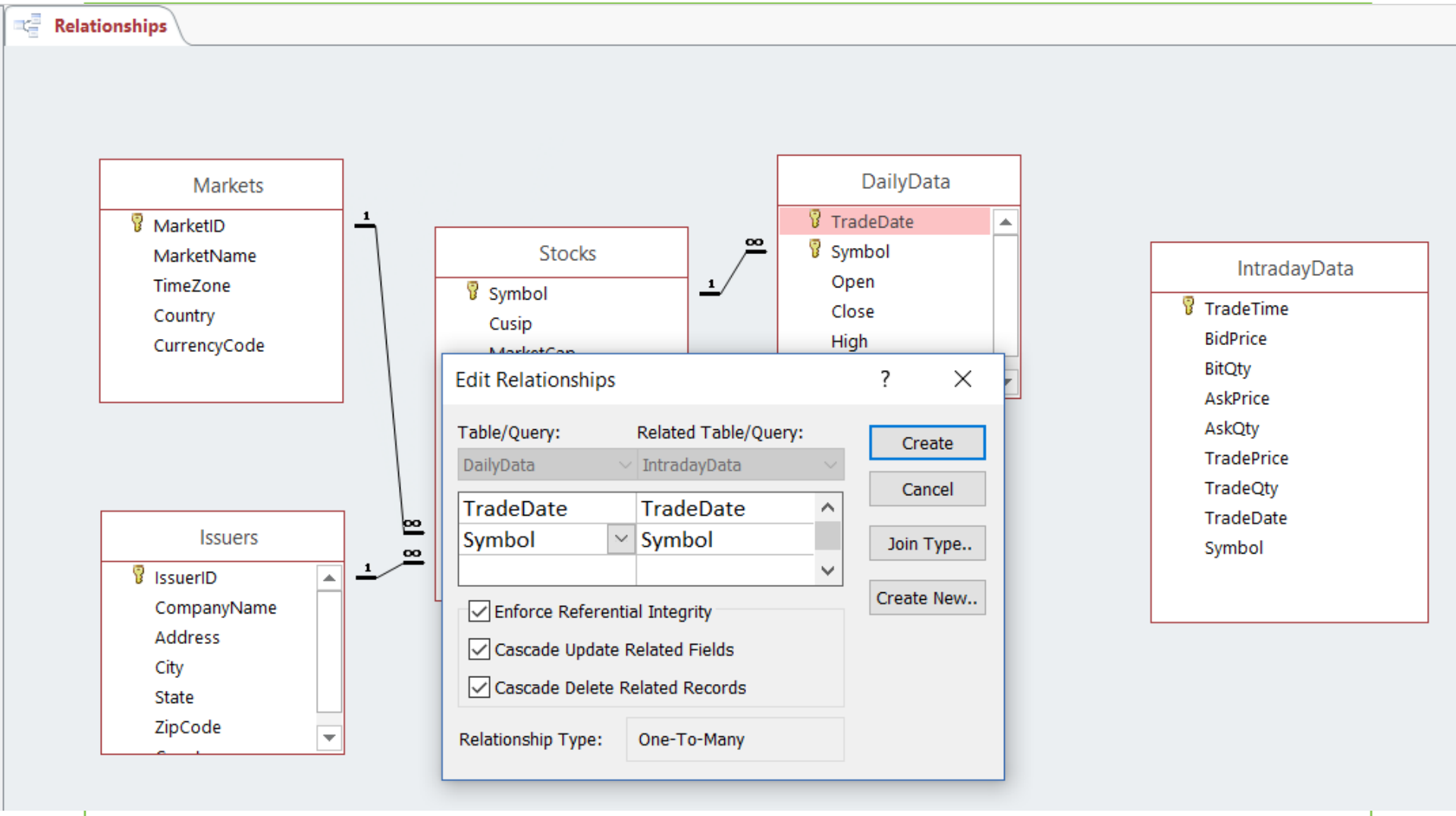
Entity-Relationship Diagram

- Click Database Tools and Select Relationships. Close/Save all the tables
- Drag all the tables in the diagram

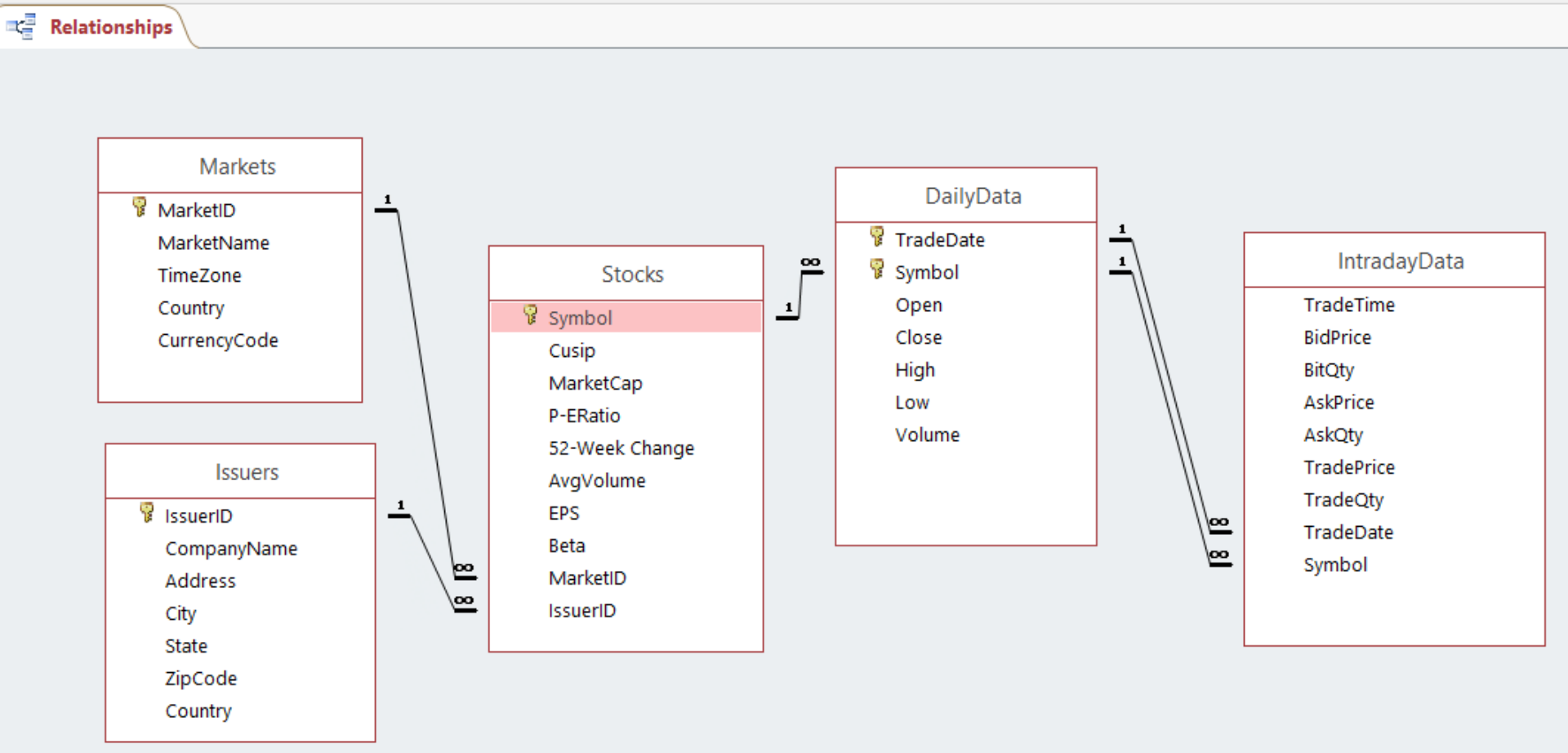


Create Relationship between Tables




Relationships
Markets
MarketID
MarketName
TimeZone
Country
CurrencyCode
Stocks
Symbol
Cusip
DailyData
TradeDate
Symbol
Open
Close
High
IntradayData
TradeTime
BidPrice
BitQty
AskPrice
AskQty
TradePrice
TradeQty
TradeDate
Symbol
Issuers
IssuerID
CompanyName
Address
City
State
ZipCode
Edit Relationships
Table/Query:
Related Table/Query:
DailyData
IntradayData
TradeDate
TradeDate
Symbol
Symbol
Create
Cancel
Join Type..
Create New..
Enforce Referential Integrity
Cascade Update Related Fields
Cascade Delete Related Records
Relationship Type:
One-To-Many

Completed E-R Diagram



Access Microsoft Access Database in C++ Program

- Microsoft JET Engine
- Data Access Technologies
  - OLE DB
  - ADO
  - DAO
  - ODBC

Microsoft JET Engine

- A database engine is the underlying software component that a database management system (DBMS) uses to create, read, update and delete (CRUD) data from a database.
- Microsoft Access uses JET as its underlying database engine.
- We will use the data access application programming interface (API) in our C++ program to interact directly with JET without going through the user interface of the DBMS.

Data Access Technologies

- ***OLE DB*** is a system-level programming interface for accessing data, and is the underlying technology for ADO.
- ***ADO*** provides a COM-based application-level interface for OLE DB data providers. ADO supports a variety of development needs, including the creation of front-end database clients and middle-tier business objects using live connections to data in relational databases.
- ***DAO*** provides access to JET (Access) databases. This API can be used from Microsoft Visual Basic, Microsoft Visual C++, and scripting languages.
- ***ODBC*** is a low-level, high-performance C programming interface designed specifically for relational databases.

Data Access Technologies

- The interaction between your C++ program and the ACCESS database, MarketDataDB is achieved by using ADO API interface. ADO consists of a series of objects that are used to invoke SQL query on the database from your C++ program directly.

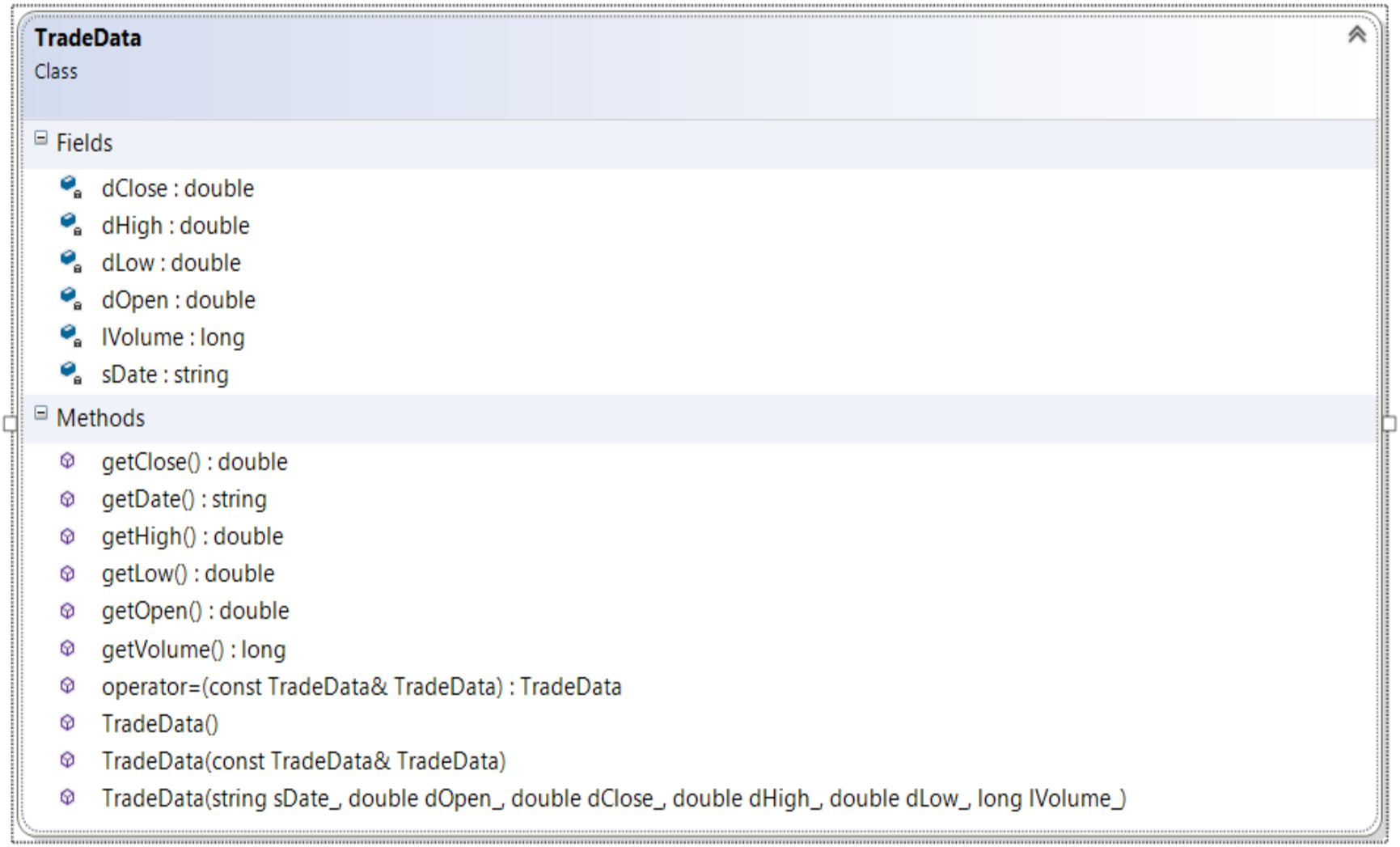
Populate Data from BLPAPI

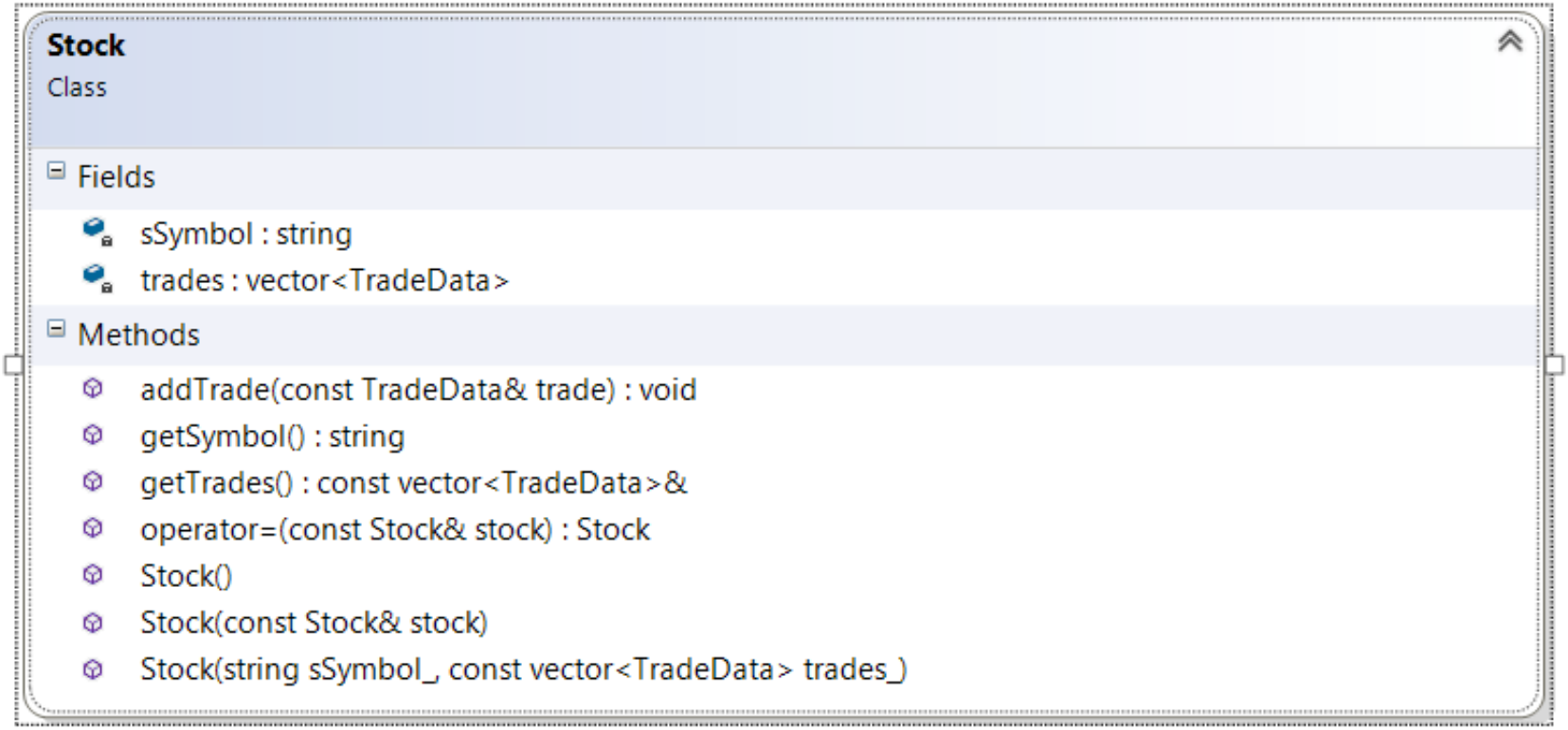
- The program MarketData
  - Connect BLPAPI to retrieve history data and intraday data for a list of stocks
  - Connect to the relational database, MarketDataDB.mdb via ADO/OLE-DB

```
#import <C:\\Program Files\\Common Files\\System\\ado\\msado15.dll>  \
    rename( "EOF", "AdoNSEOF" )

_bstr_t bstrConnect = "Provider=Microsoft.Jet.OLEDB.4.0; Data Source=E:\\MarketData\\MarketDataDB.mdb;";
```

  - Parse the data from BLPAPI and populate the data tables in the database


TradeData
Class
Fields
dClose : double
dHigh : double
dLow : double
dOpen : double
lVolume : long
sDate : string
Methods
getClose() : double
getDate() : string
getHigh() : double
getLow() : double
getOpen() : double
getVolume() : long
operator=(const TradeData& TradeData) : TradeData
TradeData()
TradeData(const TradeData& TradeData)
TradeData(string sDate_, double dOpen_, double dClose_, double dHigh_, double dLow_, long lVolume_)


Stock
Class
Fields
sSymbol : string
trades : vector<TradeData>
Methods
addTrade(const TradeData& trade) : void
getSymbol() : string
getTrades() : const vector<TradeData>&
operator=(const Stock& stock) : Stock
Stock()
Stock(const Stock& stock)
Stock(string sSymbol_, const vector<TradeData> trades_)


MarketData
Class
Fields
iPort : int
sHost : string
stockMap : vector<Stock>
Methods
parseCommandLine(int argc, char** argv) : bool
PrintComError(_com_error& e) : void
PrintStockMap() : void
printUsage() : void
run(int argc, char** argv) : void
UpdateTradeDBTable() : int

MarketData::Run() - Request Data for a list of stocks

```
Service refDataService = session.getService("//blp/refdata");
Request request =
        refDataService.createRequest("HistoricalDataRequest");
request.getElement("fields").appendValue("OPEN");
request.getElement("fields").appendValue("LAST_TRADE");
request.getElement("fields").appendValue("VOLUME");
request.getElement("fields").appendValue("HIGH");
request.getElement("fields").appendValue("LOW");
request.set("periodicityAdjustment", "ACTUAL");
request.set("periodicitySelection", "DAILY");
request.set("startDate", "20171010");
request.set("endDate", "20171111");
request.set("maxDataPoints", 2000);
```

```
vector<string> stockList;
stockList.push_back("IBM");
stockList.push_back("MSFT");
string sSuffix = " US Equity";
vector<string>::iterator vitr = stockList.begin();
for (; vitr != stockList.end(); vitr++)
{
	string sStock = *vitr + sSuffix;
	request.getElement("securities").appendValue(sStock.c_str());
}
std::cout << "Sending Request: " << request << std::endl;
session.sendRequest(request);
```

MarketData:Run() - Parse the data

```
Element securityData = msg.getElement(SECURITY_DATA);
sSymbol = securityData.getElement("security").getValueAsString();
std::size_t found = sSymbol.find(" ");
if (found != std::string::npos)
        sSymbol = sSymbol.substr(0, found);
Element fieldData = securityData.getElement(FIELD_DATA);
for (int i = 0; i < int(fieldData.numValues()); i++)
{    string sDate = fieldData.getValueAsElement(i).getElement("date").getValueAsString();
double fClose = fieldData.getValueAsElement(i).getElement("LAST_TRADE").getValueAsFloat64();
double fOpen = fieldData.getValueAsElement(i).getElement("OPEN").getValueAsFloat64();
double fHigh = fieldData.getValueAsElement(i).getElement("HIGH").getValueAsFloat64();
double fLow = fieldData.getValueAsElement(i).getElement("LOW").getValueAsFloat64();
long lVolume = (long)fieldData.getValueAsElement(i).getElement("VOLUME").getValueAsInt64();
TradeData aTrade(sDate, fClose, fOpen, fHigh, fLow, lVolume);
trades.push_back(aTrade);    }
}
if (sSymbol.length() > 0)
        stockMap.push_back(Stock(sSymbol, trades));
```

MarketData:UpdateTradeDBTable() – Populate table

```
ADODB::_ConnectionPtr pConnect("ADODB.Connection");
hResult = pConnect->Open(bstrConnect, "admin", "",
ADODB::adConnectUnspecified);
if (SUCCEEDED(hResult))
{
        char sQuery[256];
        memset((void*)sQuery, '\0', 256);
        sprintf_s(sQuery, "DELETE * FROM DailyData;");
        ADODB::_RecordsetPtr pRecSet("ADODB.Recordset");
        hResult = pRecSet->Open(sQuery, _variant_t((IDispatch *)pConnect, true),
                                        ADODB::adOpenUnspecified,
                                        ADODB::adLockUnspecified, ADODB::adCmdText);
```

```
memset((void*)sQuery, '\0', 256);
for (vector<Stock>::iterator mIt = stockMap.begin(); mIt != stockMap.end(); mIt++)
{
        string sSymbol = mIt->getSymbol();
        vector<TradeData> trades = mIt->getTrades();
        for (vector<TradeData>::iterator vIt = trades.begin(); vIt != trades.end(); vIt++)
        {       string sDate = vIt->getDate();
                double dOpen = vIt->getOpen();
                double dClose = vIt->getClose();
                double dHigh = vIt->getHigh();
                double dLow = vIt->getLow();
                long lVolume = vIt->getVolume();
                sprintf_s(sQuery, "INSERT INTO DailyData VALUES('%s','%s',%.2f,%.2f,%.2f,%.2f,%ld);",
                        sDate.c_str(), sSymbol.c_str(), dOpen, dClose, dHigh, dLow, lVolume);
                ADODB::_RecordsetPtr pRecSet("ADODB.Recordset");
                hResult = pRecSet->Open(sQuery, _variant_t((IDispatch *)pConnect, true),
                                        ADODB::adOpenUnspecified,
                                                ADODB::adLockUnspecified, ADODB::adCmdText);        }
}
```

DailyData table populated via BLPAPI

Markets | Stocks | **DailyData**

TradeDate	Symbol	Open	Close	High	Low	Volume
10/10/2017	IBM	148.50	147.71	148.95	147.65	4,032,601
10/10/2017	MSFT	76.29	76.33	76.63	76.14	13,944,545
10/11/2017	IBM	147.62	148.40	148.47	147.28	3,702,238
10/11/2017	MSFT	76.42	76.36	76.46	75.95	15,388,898
10/12/2017	IBM	147.03	147.56	147.89	146.77	3,264,344
10/12/2017	MSFT	77.12	76.49	77.29	76.37	16,876,538
10/13/2017	IBM	147.10	147.48	147.85	146.94	2,506,584
10/13/2017	MSFT	77.49	77.59	77.87	77.29	15,335,742
10/16/2017	IBM	146.83	147.22	147.67	146.51	3,052,091
10/16/2017	MSFT	77.65	77.42	77.81	77.35	12,380,093
10/17/2017	IBM	146.54	146.63	147.12	146.18	6,372,393
10/17/2017	MSFT	77.59	77.47	77.62	77.25	16,823,989
10/18/2017	IBM	159.53	157.12	161.23	156.95	30,490,192
10/18/2017	MSFT	77.61	77.67	77.85	77.37	13,300,701
10/19/2017	IBM	160.90	159.80	160.96	159.09	9,914,169
10/19/2017	MSFT	77.91	77.57	77.93	77.35	15,092,758
10/20/2017	IBM	162.07	161.07	162.48	159.77	7,868,803
10/20/2017	MSFT	78.81	78.32	78.97	78.22	22,866,426
10/23/2017	IBM	159.55	162.05	162.51	159.54	5,779,378
10/23/2017	MSFT	78.83	78.99	79.34	78.76	20,627,173
10/24/2017	IBM	155.88	159.65	159.70	155.17	8,194,690
10/24/2017	MSFT	78.86	78.90	79.20	78.46	17,517,182