EODC usage examples

The following examples are listed below:

0) General usage

The general format of EODC queries is:

```
>> [data, errMsg, limits] = EODC(action, parameter1Name,parameter1Value, parameter2Name,parameter2Value, ...);
```

where:

- `data` – the returned data (usually from from EODHistoricalData's servers)
- `errMsg` – error message (if any)
- `limits` – the updated query limits. These can be used to ensure you do not surpass your license's allowed limits

- `action` – a string that specifies the query type. One of: 'license', 'limits', 'version', 'update', 'revert', 'doc', 'prices', 'fundamentals', 'technicals', 'options', 'dividends', 'splits', 'earnings', 'shorts', 'ipo', 'lookup'
- `parameterName` – a case-insensitive string that specifies the parameter name. Each query action has its own set of acceptable parameter names.

In your very first EODC command, you need to specify the API token that you received from EODHistoricalData.com, via the `'API_Token'` parameter (you can get a free token here [16]). EODC will reuse this token in all subsequent commands, so you only need to specify it once (or when you change your token key). For example:

```
>> data = EODC('prices', ..., 'API_Token','123456.abcdef');  % '123456.abcdef' is a dummy (invalid) token, use your own personal token
```

Note: all parameter names are case in-sensitive. i.e., you can use 'API_Token' or 'API_TOKEN' or 'api_token' as you wish.

List of general parameters that are relevant to all query types:

- `API_Token` – string; no default. This is the API token that is provided by EODHistoricalData.com (you can get a free token here [16])
- `Timeout` – numeric; default=5. Max number of seconds to wait for EOD's response
- `Symbol` or `Symbols` – string or cell-array of strings, e.g., 'IBM' or 'IBM,GOOG' or {'IBM','GOOG'}
- `UseParallel` – logical true/false; default=false (only available in Analyst/Pro licenses, for multiple symbols)
- `Order` – string; default='asc'; either 'asc' or 'desc'. Affects the order of returned data array(s), where relevant

1) Get market data (delayed snapshot)

```
>> data = EODC('prices', 'symbol','IBM', 'datatype','live')
data =
  struct with fields:
          symbol: 'IBM.US'
       timestamp: 1577204520
       gmtoffset: 0
            open: 135.61
            high: 135.62
             low: 134.61
           close: 134.7425
          volume: 593912
   previousClose: 135.55
          change: -0.808
        change_p: -0.596
     datestr_GMT: '24-Dec-2019 16:22:00'
     datenum_GMT: 737783.681944444
```

Available parameters that affect this query (in addition to the standard general parameters [18]):

- `DataType` – string; must be 'live' to get the current (delayed) market snapshot
- `SecType` – string; default='equity'; one of 'equity','index','bond'

2) Get historic/intra-day data

```
>> data = EODC('prices', 'symbol','IBM', 'FromDate',20191203, 'ToDate','2019/12/13', 'DataType','day')
data =
  9×1 struct array with fields:
    symbol
    date
    datenum
    open
    high
    low
    close
    adjusted_close
    volume
```

```
>> data(1)
ans =
  struct with fields:

           symbol: 'IBM.US'
             date: '2019-12-03'
          datenum: 737762
             open: 132
             high: 132.44
              low: 130.69
            close: 132.12
   adjusted_close: 132.12
           volume: 3642500
```

Available parameters that affect this query (in addition to the standard general parameters[18]):

- `DataType` – string; default='day'; one of 'intreday','day','week','month'
- `SecType` – string; default='equity'; one of 'equity','index','bond'
- `FromDate` – integer or string; default=[]; earliest data date (GMT timezone). Examples: 20191203, '2019-12-03', 201912031535, '2019/12/03 15:35'
- `ToDate` – integer or string; default=[]; latest data date (GMT timezone). Examples: 20191203, '2019-12-03', 201912031535, '2019/12/03 15:35'

3) Get fundamental data

```
% Fundamental data for an equity:
>> data = EODC('fundamental', 'symbol','IBM')
data =
  struct with fields:
             symbol: 'IBM.US'
            General: [1×1 struct]
         Highlights: [1×1 struct]
          Valuation: [1×1 struct]
         SharesStats: [1×1 struct]
         Technicals: [1×1 struct]
     SplitsDividends: [1×1 struct]
          ESGScores: [1×1 struct]
   outstandingShares: [1×1 struct]
           Earnings: [1×1 struct]
         Financials: [1×1 struct]

>> data.General
ans =
  struct with fields:
                   Code: 'IBM'
                   Type: 'Common Stock'
                   Name: 'International Business Machines Corporation'
               Exchange: 'NYSE'
           CurrencyCode: 'USD'
           CurrencyName: 'US Dollar'
         CurrencySymbol: '$'
            CountryName: 'USA'
             CountryISO: 'US'
                   ISIN: 'US4592001014'
                  CUSIP: '459200101'
                    CIK: 51143
       EmployerIdNumber: '13-0871985'
          FiscalYearEnd: 'December'
                IPODate: []
   InternationalDomestic: 'International/Domestic'
                 Sector: 'Technology'
               Industry: 'Information Technology Services'
             GicSector: 'Information Technology'
              GicGroup: 'Software & Services'
           GicIndustry: 'IT Services'
        GicSubIndustry: 'IT Consulting & Other Services'
           Description: 'International Business Machines Corporation operates as an integrated technology and services company worldwide...'
               Address: '1 New Orchard Road↵10504,Armonk,USA↵'
                 Phone: '914-499-1900'
                WebURL: 'www.ibm.com'
               LogoURL: 'https://eodhistoricaldata.com/img/logos/US/IBM.png'
       FullTimeEmployees: 350600
               UpdatedAt: '2019-12-24'

>> data.Highlights
ans =
  struct with fields:
        MarketCapitalization: 120048099328
     MarketCapitalizationMln: 120048.0993
                      EBITDA: 16664000512
                     PERatio: 15.7525
                    PEGRatio: 4.92
       WallStreetTargetPrice: 148.3
                   BookValue: 20.275
               DividendShare: 6.48
               DividendYield: 0.0478
               EarningsShare: 8.605
      EPSEstimateCurrentYear: 12.8
         EPSEstimateNextYear: []
      EPSEstimateNextQuarter: 2.17
   EPSEstimateCurrentQuarter: []
           MostRecentQuarter: '2019-09-30'
                ProfitMargin: 0.1
           OperatingMarginTTM: 0.1445
            ReturnOnAssetsTTM: 0.0513
            ReturnOnEquityTTM: 0.4061
                  RevenueTTM: 77130997760
           RevenuePerShareTTM: 86.587
    QuarterlyRevenueGrowthYOY: -0.039
                GrossProfitTTM: 36936000000
                 DilutedEpsTTM: 8.605
   QuarterlyEarningsGrowthYOY: -0.365

% Fundamental data for a bond (note that we can search by CUSIP or ISIN in the Symbol parameter):
>> data = EODC('fundamental', 'symbol','910047AG4', 'sectype','bond')
```

```
data =
  struct with fields:
                     symbol: '910047AG4'
                       ISIN: 'US910047AG49'
                      CUSIP: '910047AG4'
                       Name: 'UNITED AIRLS HLDGS INC 6% 01Dec2020'
                 UpdateDate: '2019-11-27'
                        WKN: 'A1HS3T'
                      Sedol: 'BFV4Y03'
                       FIGI: 'BBG005J8GHT4'
                   Currency: 'USD'
                     Coupon: 6
                      Price: 103.03
              LastTradeDate: '2019-11-27'
              Maturity_Date: '2020-11-30'
            YieldToMaturity: 2.833
                   Callable: 'No'
                NextCallDate: []
    MinimumSettlementAmount: '1000 USD'
        ParIntegralMultiple: '1000 USD'
         ClassificationData: [1×1 struct]
                     Rating: [1×1 struct]
                  IssueData: [1×1 struct]

>> data.ClassificationData
ans =
  struct with fields:
               BondType: 'Unternehmensanleihen Welt Rest'
               DebtType: 'Senior Unsecured Note'
          IndustryGroup: 'Industrial'
       IndustrySubGroup: 'Transportation'
        SubProductAsset: 'CORP'
    SubProductAssetType: 'Corporate Bond'

>> data.Rating
ans =
  struct with fields:
             MoodyRating: 'Ba3'
    MoodyRatingUpdateDate: '2019-11-27'
                 SPRating: 'BB'
       SPRatingUpdateDate: '2018-04-16'

>> data.IssueData
ans =
  struct with fields:
                 IssueDate: '2013-11-08'
              OfferingDate: '2013-11-01'
            FirstCouponDate: '2014-06-01'
            FirstTradingDay: '2013-11-08'
    CouponPaymentFrequency: []
                    Issuer: 'United Airlines Holdings Inc.'
          IssuerDescription: 'United Continental Holdings Inc. is an airline holding company. The Company owns and operates airlines that transpor
              IssuerCountry: 'USA'
                  IssuerURL: []
```

Available parameters that affect this query (in addition to the standard general parameters[18]):

- `SecType` – string; default='equity'; one of 'equity','index','bond'

4) Get historic and upcoming earning events

```
% Historic earning reports (requested on Dec. 24, 2019):
>> data = EODC('earnings', 'symbol','IBM', 'fromdate',20190101)
data =
  4×1 struct array with fields:
    symbol
    report_date
    date
    datenum
    actual
    estimate
    difference
    percent

>> data(end)
ans =
  struct with fields:
         symbol: 'IBM.US'
    report_date: '2019-10-16'
           date: '2019-09-30'
        datenum: 737698
         actual: 2.68
       estimate: 2.67
     difference: 0.01
        percent: 0.37

% Upcoming earning reports (requested on Dec. 24, 2019):
>> data = EODC('earnings', 'symbol','IBM')
data =
     []

>> data = EODC('earnings', 'symbol','JOB.US')
data =
  struct with fields:
         symbol: 'JOB.US'
    report_date: '2019-12-30'
           date: '2019-12-30'
        datenum: 737789
         actual: []
       estimate: -0.2
     difference: 0
        percent: []

% All upcoming (announced) earnings reports:
```

```
>> data = EODC('earn')
data =
  38×1 struct array with fields:
    symbol
    report_date
    date
    datenum
    actual
    estimate
    difference
    percent

>> struct2table(data)
ans =
  38×8 table
      symbol     report_date        date       datenum      actual        estimate     difference     percent
    __________  ____________  ____________  _______  ____________  ____________  __________  ____________
    '3174.TSE'  '2019-12-24'  '2019-12-24'  737783  {0×0 double}  {0×0 double}      0       {0×0 double}
    '6196.TSE'  '2019-12-24'  '2019-12-24'  737783  {0×0 double}  {0×0 double}      0       {0×0 double}
    'HURC.US'   '2019-12-24'  '2019-12-24'  737783  {0×0 double}  {0×0 double}      0       {0×0 double}
    'JRJC.US'   '2019-12-24'  '2019-12-24'  737783  {0×0 double}  {0×0 double}      0       {0×0 double}
    'MLR.V'     '2019-12-24'  '2019-12-24'  737783  {0×0 double}  {0×0 double}      0       {0×0 double}
    'ROYT.US'   '2019-12-24'  '2019-12-24'  737783  {0×0 double}  {[     0.05]}      0       {0×0 double}
    ...
```

Available parameters that affect this query (in addition to the standard general parameters[18]):

- `FromDate` – integer or string; default=[]; earliest data date (GMT timezone). Examples: 20191203, '2019-12-03'
- `ToDate` – integer or string; default=[]; latest data date (GMT timezone). Examples: 20191203, '2019-12-03'

5) Get historic and upcoming split events

```
% 2 splits for AAPL since 1/1/2000:
>> data = EODC('splits', 'symbol','AAPL', 'fromdate',20010101)
data =
  2×1 struct array with fields:
    symbol
    date
    datenum
    split
    split_ratio

>> struct2table(data)
ans =
  2×5 table
     symbol         date       datenum          split          split_ratio
    _________  ____________  _______  _____________________  ___________
    'AAPL.US'  '2005-02-28'  732371   '2.000000/1.000000'        2
    'AAPL.US'  '2014-06-09'  735759   '7.000000/1.000000'        7

% No splits for IBM since 1/1/2000:
>> data = EODC('splits', 'symbol','IBM', 'fromdate',20100101)
data =
     []

% Upcoming (future) splits:
>> data = EODC('splits')
data =
  60×1 struct array with fields:
    symbol
    split_date
    split_datenum
    optionable
    old_shares
    new_shares
    split_ratio

>> struct2table(data)
ans =
  60×7 table
         symbol         split_date    split_datenum  optionable  old_shares  new_shares      split_ratio
    ________________  ____________  _____________  __________  __________  __________  ___________________
    '5DP.SG'          '2019-12-24'     737783         'N'           2           3                       1.5
    'DCTH.US'         '2019-12-24'     737783         'N'         700           1      0.00142857142857143
    'LZRFY.US'        '2019-12-24'     737783         'N'         100         105                      1.05
    'QKL.F'           '2019-12-24'     737783         'N'          14           1       0.0714285714285714
    'QKL.STU'         '2019-12-24'     737783         'N'          14           1       0.0714285714285714
    'QUIK.US'         '2019-12-24'     737783         'N'          14           1       0.0714285714285714
    'BALMLAWRIE.NSE'  '2019-12-26'     737785         'N'           2           3                       1.5
    '000100.KO'       '2019-12-27'     737786         'N'           1           1                         1
    ...
```

Available parameters that affect this query (in addition to the standard general parameters[18]):

- `FromDate` – integer or string; default=[]; earliest data date (GMT timezone). Examples: 20191203, '2019-12-03'
- `ToDate` – integer or string; default=[]; latest data date (GMT timezone). Examples: 20191203, '2019-12-03'

6) Get historic and upcoming dividend events

```
% Historic dividend events:
>> data = EODC('dividend', 'symbol','IBM', 'fromdate',20190101)
data =
  4×1 struct array with fields:
    symbol
    date
    datenum
    declarationDate
    recordDate
    paymentDate
    value

>> struct2table(data)
ans =
  4×7 table
```

```
    symbol       date      datenum  declarationDate   recordDate   paymentDate   value
    ______  __________  _______  _______________  __________  ___________  _____
    'IBM.US'  '2019-02-07'  737463     '2019-01-29'   '2019-02-08'  '2019-03-09'  1.57
    'IBM.US'  '2019-05-09'  737554     '2019-04-30'   '2019-05-10'  '2019-06-10'  1.62
    'IBM.US'  '2019-08-08'  737645     '2019-07-30'   '2019-08-09'  '2019-09-10'  1.62
    'IBM.US'  '2019-11-07'  737736     '2019-10-29'   '2019-11-08'  '2019-12-10'  1.62

>> data(1)
ans =
  struct with fields:
            symbol: 'IBM.US'
              date: '2019-02-07'
           datenum: 737463
    declarationDate: '2019-01-29'
        recordDate: '2019-02-08'
       paymentDate: '2019-03-09'
             value: 1.57

% Upcoming (future) dividend events:
>> data = EODC('dividend', 'symbol','IBM')
data =
  199×1 struct array with fields:
    symbol
    date
    datenum
    declarationDate
    recordDate
    paymentDate
    value

>> data(1)
ans =
  struct with fields:
            symbol: 'IBM.US'
              date: '1970-05-01'
           datenum: 719649
    declarationDate: []
        recordDate: []
       paymentDate: []
             value: 0.06
```

Available parameters that affect this query (in addition to the standard general parameters[18]):

- `FromDate` – integer or string; default=[]; earliest data date (GMT timezone). Examples: 20191203, '2019-12-03'
- `ToDate` – integer or string; default=[]; latest data date (GMT timezone). Examples: 20191203, '2019-12-03'

7) Get historic IPO events

```
% Historic IPO events:
>> data = EODC('IPO', 'fromdate',20191201)
data =
  132×1 struct array with fields:
    symbol
    name
    exchange
    currency
    start_date
    filing_date
    amended_date
    price_from
    price_to
    offer_price
    shares
    deal_type

>> struct2table(data)
ans =
  132×12 table
     symbol            name             exchange      currency   start_date   filing_date   amended_date   price_from   price_to   offer_price   sha
    ________  ____________________  _____________  ________  ___________  ___________  ____________  ___________  ________  ___________  ____
    'AMX.AU'  'Aerometrex Ltd'      'ASX'          'AUD'     '2019-12-10'  '2019-11-01'  '2019-11-01'            0         0            1  2500
    'N/A'     'Air Baltic Corp AS'  'Riga'         'EUR'     {0×0 double}  '2018-07-25'  '2019-12-09'            0         0            0
    'N/A'     'Akeso Inc'           'HKSE'         'USD'     {0×0 double}  '2019-12-03'  {0×0 double}            0         0            0
    '9966.HK'  'Alphamab Oncology'  'HKSE'         'HKD'     '2019-12-05'  '2018-10-23'  '2019-12-02'            0         0            0
    '9966.HK'  'Alphamab Oncology'  'HKSE'         'HKD'     {0×0 double}  '2018-10-23'  '2019-12-02'            0         0         10.2  8970
    '9966.HK'  'Alphamab Oncology'  'HKSE'         'HKD'     {0×0 double}  '2018-10-23'  '2019-12-02'            0         0         10.2  8970
    '9966.HK'  'Alphamab Oncology'  'HKSE'         'HKD'     '2019-12-05'  '2018-10-23'  '2019-12-02'            0         0            0
    'NA.CN'   'Altum Resource Corp'  'Canadian Sec'  'CAD'     {0×0 double}  '2019-12-12'  {0×0 double}       0.0759    0.0759            0   350
    ...
```

Available parameters that affect this query (in addition to the standard general parameters[18]):

- `FromDate` – integer or string; default=[]; earliest data date (GMT timezone). Examples: 20191203, '2019-12-03'
- `ToDate` – integer or string; default=[]; latest data date (GMT timezone). Examples: 20191203, '2019-12-03'

8) Get historic technical indicators

```
>> data = EODC('technical', 'symbol','IBM', 'fromdate',20190701, 'period',40, 'function','rsi')
data =
  83×1 struct array with fields:
    symbol
    date
    datenum
    rsi

>> data(1)
ans =
  struct with fields:

     symbol: 'IBM.US'
       date: '2019-08-27'
    datenum: 737664
        rsi: 42.92
```

```
>> struct2table(data)
ans =
  83×4 table
     symbol        date        datenum     rsi
    ________    ____________    _______    ______

    'IBM.US'    '2019-08-27'    737664      42.92
    'IBM.US'    '2019-08-28'    737665     44.394
    'IBM.US'    '2019-08-29'    737666     46.291
    'IBM.US'    '2019-08-30'    737667     46.861
    'IBM.US'    '2019-09-03'    737671     45.765
    'IBM.US'    '2019-09-04'    737672     47.712
    'IBM.US'    '2019-09-05'    737673     51.457
    'IBM.US'    '2019-09-06'    737674     51.134
    'IBM.US'    '2019-09-09'    737677      52.68
    'IBM.US'    '2019-09-10'    737678     54.464
    ...
```

Available parameters that affect this query (in addition to the standard general parameters[18]):

- `FromDate` – integer or string; default=[]; earliest data date (GMT timezone). Examples: 20191203, '2019-12-03'
- `ToDate` – integer or string; default=[]; latest data date (GMT timezone). Examples: 20191203, '2019-12-03'

9) Get historic short interests

```
% Historic short interests for AAPL (note the 'desc' order)
>> data = EODC('shorts', 'symbol','AAPL', 'fromdate',20190101, 'todate',20191231, 'order','desc')
data =
  17×1 struct array with fields:
    symbol
    date
    datenum
    short
    volume

>> data(1)
ans =
  struct with fields:

     symbol: 'AAPL.US'
       date: '2019-09-13'
    datenum: 737681
      short: 42651634
     volume: 28651684

>> struct2table(data)
ans =
  17×5 table
     symbol         date         datenum     short       volume
    _________    ____________    _______    ________    ________

    'AAPL.US'    '2019-09-13'    737681     42651634    28651684
    'AAPL.US'    '2019-08-30'    737667     39517330    25536624
    'AAPL.US'    '2019-08-15'    737652     45594258    36615393
    'AAPL.US'    '2019-07-31'    737637     43005960    23525795
    'AAPL.US'    '2019-07-15'    737621     42428971    19134090
    'AAPL.US'    '2019-06-28'    737604     43448528    24985287
    'AAPL.US'    '2019-06-14'    737590     47003511    26629536
    'AAPL.US'    '2019-05-31'    737576     51257104    29778281
    'AAPL.US'    '2019-05-15'    737560     49550348    37444374
    'AAPL.US'    '2019-04-30'    737545     52669476    24561618
    'AAPL.US'    '2019-04-15'    737530     61003851    23755423
    'AAPL.US'    '2019-03-29'    737513     67786010    35017255
    'AAPL.US'    '2019-03-15'    737499     72751656    27347457
    'AAPL.US'    '2019-02-28'    737484     96832513    22065273
    'AAPL.US'    '2019-02-15'    737471     39903215    26937971
    'AAPL.US'    '2019-01-31'    737456     40360782    34248007
    'AAPL.US'    '2019-01-15'    737440     46579709    45191034
```

Available parameters that affect this query (in addition to the standard general parameters[18]):

- `FromDate` – integer or string; default=[]; earliest data date (GMT timezone). Examples: 20191203, '2019-12-03'
- `ToDate` – integer or string; default=[]; latest data date (GMT timezone). Examples: 20191203, '2019-12-03'

10) Search for a symbol across all exchanges

```
>> data = EODC('lookup', 'symbol','IBM')
data =
  18×1 struct array with fields:
    symbol
    Exchange
    Name
    Country
    Currency
    ISIN

>> struct2table(data)
ans =
  18×6 table
    symbol    Exchange                       Name                          Country     Currency         ISIN
    ______    ________    ___________________________________________    _________    ________    ______________

    'IBM'     'US'        'International Business Machines Corporation'    'USA'        'USD'     'US4592001014'
    'IBM'     'XETRA'     'International Business Machines Corporation'    'Germany'    'EUR'     {0×0 double}
    'IBM'     'MX'        'International Business Machines Corporation'    'Mexico'     'MXN'     {0×0 double}
    'IBM'     'F'         'International Business Machines Corporation'    'Germany'    'EUR'     {0×0 double}
    'IBM'     'STU'       'IBM (IBM.SG)'                                   'Germany'    'EUR'     {0×0 double}
    'IBM'     'MU'        'IBM'                                            'Germany'    'EUR'     {0×0 double}
    'IBM'     'HM'        'IBM'                                            'Germany'    'EUR'     {0×0 double}
    'IBM'     'BE'        'IBM'                                            'Germany'    'EUR'     {0×0 double}
    'IBM'     'DU'        'IBM - Dusseldorf Stock Exchang'                 'Germany'    'EUR'     {0×0 double}
    'IBMK'    'US'        'iShares iBonds Dec 2022 Term Muni Bond ETF'     'USA'        'USD'     'US46435G7557'
    'IBMI'    'US'        'iShares iBonds Sep 2020 Term Muni Bond ETF'     'USA'        'USD'     'US46434V5710'
    'IBMJ'    'US'        'iShares iBonds Dec 2021 Term Muni Bond ETF'     'USA'        'USD'     'US46435G7896'
    'IBML'    'US'        'iShares iBonds Dec 2023 Term Muni Bond ETF'     'USA'        'USD'     'US46435G3184'
    'IBMM'    'US'        'iShares iBonds Dec 2024 Term Muni Bond ETF'     'USA'        'USD'     {0×0 double}
    'IBMN'    'US'        'iShares iBonds Dec 2025 Term Muni Bond ETF'     'USA'        'USD'     {0×0 double}
```

```
    'IBMO'    'US'        'iShares iBonds Dec 2026 Term Muni Bond ETF'     'USA'         'USD'      {0×0 double}
    'IBMP'    'US'        'iShares iBonds Dec 2027 Term Muni Bond ETF'     'USA'         'USD'      {0×0 double}
    'IBMQ'    'US'        'iShares iBonds Dec 2028 Term Muni Bond ETF'     'USA'         'USD'      {0×0 double}
```

Available parameters that affect this query (in addition to the standard general parameters[18]):

- `DataType` – string; must be 'symbol' for this query type

11) Search for all symbols in an exchange

```
>> data = EODC('lookup', 'symbol','LSE', 'dataType','exchange')
data =
  3444×1 struct array with fields:
    symbol
    Name
    Country
    Exchange
    Currency
    Type

>> head(struct2table(data))
ans =
  8×6 table
    symbol                                    Name                                      Country   Exchange   Currency       Type
    ______   ________________________________________________________________________   _______   ________   ________   ______________
    '0KBI'   'Knorr-Bremse Aktiengesellschaft'                                           'UK'      'LSE'      'EUR'      'Common Stock'
    '100H'   'MULTI-UNITS LUXEMBOURG - Lyxor FTSE 100 UCITS ETF - Monthly Hedged to USD - Acc'   'UK'   'LSE'   'USD'   'ETF'
    '1MCS'   'WisdomTree FTSE 250 1x Daily Short'                                        'UK'      'LSE'      'GBX'      'ETF'
    '1PAS'   'WisdomTree Palladium 1x Daily Short'                                       'UK'      'LSE'      'USD'      'ETF'
    '2MCL'   'WisdomTree FTSE 250 2x Daily Leveraged'                                    'UK'      'LSE'      'GBX'      'ETF'
    '2PAL'   'WisdomTree Palladium 2x Daily Leveraged'                                   'UK'      'LSE'      'USD'      'ETF'
    '2UKL'   'WisdomTree FTSE 100 2x Daily Leveraged'                                    'UK'      'LSE'      'GBX'      'ETF'
    '2UKS'   'WisdomTree FTSE 100 2x Daily Short'                                        'UK'      'LSE'      'GBX'      'ETF'
```

Available parameters that affect this query (in addition to the standard general parameters[18]):

- `DataType` – string; must be 'exchange' for this query type

12) Get an options chain

```
>> data = EODC('options', 'symbol','IBM')
data =
  struct with fields:
      symbol: 'IBM.US'
        code: 'IBM'
    exchange: 'US'
        data: [15×1 struct]

>> data.data(1)
ans =
  struct with fields:
       expirationDate: '2019-12-27'
    expirationDatenum: 737786
                 puts: [39×1 struct]
                calls: [39×1 struct]

>> struct2table(data.data)
ans =
  15×4 table
    expirationDate  expirationDatenum      puts           calls
    ______________  _________________  _____________  _____________
    {'2019-12-27'}       737786        {39×1 struct}  {39×1 struct}
    {'2020-01-03'}       737793        {38×1 struct}  {38×1 struct}
    {'2020-01-10'}       737800        {35×1 struct}  {35×1 struct}
    {'2020-01-17'}       737807        {55×1 struct}  {55×1 struct}
    ...

>> data.data(1).puts(6)
ans =
  struct with fields:

         contractName: 'IBM191227P00120000'
         contractSize: 'REGULAR'
             currency: 'USD'
                 type: 'PUT'
           inTheMoney: 'FALSE'
    lastTradeDateTime: '2019-12-23 10:01:34'
       expirationDate: '2019-12-27'
    expirationDatenum: 737786
               strike: 120
            lastPrice: 0.01
                  bid: []
                  ask: 0.03
               change: -0.21
        changePercent: -0.9545
               volume: 2
         openInterest: 3
    impliedVolatility: 45.512
                delta: -0.0048
                gamma: 0.0022
                theta: -0.0113
                 vega: 0.002
                  rho: -0.0001
          theoretical: 0.01
       intrinsicValue: 0
            timeValue: 0
            updatedAt: '2019-12-23 19:32:03'

>> struct2table(data.data(1).puts)
ans =
  39×26 table
    ...
```

Available parameters that affect this query (in addition to the standard general parameters[18]):

- `FromExpiryDate` – integer or string; default=[]; earliest data date (GMT timezone). Examples: 20191203, '2019-12-03'
- `ToExpiryDate` – integer or string; default=[]; latest data date (GMT timezone). Examples: 20191203, '2019-12-03'
- `FromTradeDate` – integer or string; default=[]; earliest data date (GMT timezone). Examples: 20191203, '2019-12-03'
- `ToTradeDate` – integer or string; default=[]; latest data date (GMT timezone). Examples: 20191203, '2019-12-03'

13) Get EODC version and query limits

```
>> data = EODC('version')
data =
  struct with fields:
    Version: 1.03
    Release: '24-Dec-2019'
    License: 'Free'
     Expiry: '08-Aug-2020'

>> limits = EODC('limits')
limits =
  struct with fields:
          licenseType: 'Free'
     maxQueriesPerDay: 30
     maxQueriesPerMin: 3
      numQueriesToday: 27
    numQueriesThisMin: 0
       secsToEndOfMin: 0
```

Legal disclaimer