BQL

COMPANY FINANCIALS

PARAMETERS, FUNCTIONS & ASSOCIATIVE COLUMNS

Version: C-2020-11-23

Table of Contents

1. *Introduction*

This document supplements the BQL Raw Query Guide by providing instructions and examples about how Bloomberg Company Financials can be queried via BQL.

Some of the major benefits of Bloomberg Company Financials in BQL include:

- Consistency of the parameters and parameter parameters for actuals and estimates
- Accessibility of actuals and estimates data using a single field
- New fiscal period types such as LTM, Blended Annual and Blended Trailing
- Easy calendarization for aligning data across different companies
- Point in time fundamental data and ratios
- Ability to exclude the impact of the new Operating Lease Accounting Standard

With Bloomberg Company Financials via BQL the user has many more options to retrieve and manipulate the data that Bloomberg Company Financials offers.

The first section of this document provides details about each parameter offered for Bloomberg Company Financials in BQL, the available parameter inputs, as well as the functionality of each of those inputs

The second section offers specifications and examples about BQL functions which are applicable to Company Financials only.

The third section of this document offers many examples about how the different parameters interact with each other and how they can be used to retrieve Bloomberg Company Financials data.

The fourth section provides details about the associative columns available which are useful for understanding the context of each data point returned by a BQL query.

Finally, the final section provides a comparison between the parameters for Company Financials in BQL and the legacy Bloomberg Company Financials parameters, which is useful for converting existing queries which use Bloomberg legacy Company Financials as well as provide some context for the new parameters available in BQL.

2. ***Parameters***

The data parameters offered for Bloomberg fundamental data in BQL are the following:

- FA_PERIOD_TYPE
- FA_PERIOD_TYPE_SOURCE
- FA_PERIOD_OFFSET
- FA_PERIOD_REFERENCE
- AS_OF_DATE
- FA_PERIOD_YEAR_END
- FA_FILING_STATUS
- FA_ADJUSTED
- FA_ACT_EST_DATA
- FA_CONSOLIDATED
- FA_ACCT_STD
- CAPITAL_CHANGES_ADJUST
- FILL
- EST_SOURCE
- CURRENCY
- CURRENCY_CONVERSION_METHOD
- EXCL_OPER_LEASES

The rest of this section offers more details about the meaning of these parameters and the functionality that they offer.

2.1. Period type

Parameter description:	Controls the type of fiscal period returned, which could be a fiscal period reported by the company (such as an Annual, Quarterly or Semiannual fiscal period) or a synthetic fiscal period created by Bloomberg (such as a last-12-months, calendar, or blended period)
Utility:	Enables analysis of fiscal periods of different duration (3, 6, 9 or 12- months) and provides several flavors or 12-month data calculations
Parameter mnemonic:	FA_PERIOD_TYPE
Abbreviated mnemonic(s):	FPT
Default value:	LTM
Example query:	get(IS_EPS(**FPT=A**)) for (['IBM US Equity'])

Parameter available inputs:

Value	Description	Functionality
A	Annual	Returns data for an Annual fiscal period, as reported by the source[1].
Q	Quarterly	Returns data for a Quarterly fiscal period, as reported by the source.
S	Semiannual	Returns data for a Semiannual fiscal period, as reported by the source.
P	Primary	Returns data for the primary periodicity[2] used by the company to report its financials.
LTM	Last 12 months	Returns data for a synthetic fiscal period calculated by Bloomberg that covers a 12-months period[3]. The calculation may use quarterly, semiannual or annual data reported by the company, depending on a company's primary reporting periodicity.
YTD	Year-to-date	Returns data for a synthetic fiscal period calculated by Bloomberg for the last 3, 6, 9 or 12-month period. The

[1] The source could be the company (periodic financial filings) or sell-side research (research reports).

[2] See field PRIMARY_PERIODICITY.

[3] It can be calculated as (i) the sum of 4 quarterly or 2 semiannual periods (depending on company disclosure) for flow accounting concepts (such as Revenue); (ii) average of 4 quarterly or 2 semiannual periods (depending on company disclosure) for average accounting concepts (such as Average Shares Outstanding); (iii) data reported in the latest fiscal period for stock accounting concepts (such as Total Assets); or (iv) data reported at the beginning of the period for any beginning-of-period accounting concepts (such as Cash at the beginning of the period). For any Last-12-months fiscal period coinciding with the Annual fiscal period, the Annual data is returned without any additional calculations.

		calculation may use quarterly, semiannual or annual data reported by the company, depending on a company's primary reporting periodicity.
BA	Blended annual	Returns data for a synthetic fiscal period calculated by Bloomberg for a 12-month period ending on a custom date[4]. The calculation blends data from quarterly, semiannual or annual data reported by the company, depending on a company's primary reporting periodicity, using a time weighted average.
BQ	Blended quarter	Returns data for a synthetic fiscal period calculated by Bloomberg for a 3-month period ending on a custom date[4]. The calculation blends data from quarterly, semiannual or annual data reported by the company, depending on a company's primary reporting periodicity, using a time weighted average.
BS	Blended semiannual	Returns data for a synthetic fiscal period calculated by Bloomberg for a 6-month period ending on a custom date[4]. The calculation blends data from quarterly, semiannual or annual data reported by the company, depending on a company's primary reporting periodicity, using a time weighted average.
BT	Blended trailing	Returns data for a synthetic fiscal period calculated by Bloomberg for a 12-month period ending on a custom date[5]. The calculation blends data from quarterly, semiannual or annual data reported by the company, depending on a company's primary reporting periodicity, using a time weighted average.
Q1	Quarter 1	Returns data for a Q1 fiscal period, as reported by the source.
Q2	Quarter 2	Returns data for a Q2 fiscal period, as reported by the source.
Q3	Quarter 3	Returns data for a Q3 fiscal period, as reported by the source.
Q4	Quarter 4	Returns data for a Q4 fiscal period, as reported by the source.
S1	Semiannual 1	Returns data for a Semiannual 1 fiscal period, as reported by the source.
S2	Semiannual 2	Returns data for a Semiannual 2 fiscal period, as reported by the source.

[4] The default period end date is the latest December 31 date preceding the period end date of the latest fiscal period reported by the company. This can be modified through the FA_PERIOD_YEAR_END parameter.

[5] The default period end date is determined by the AS_OF_DATE parameter.

2.2. *Period type source*

Parameter description:	Determines the source of data used to calculate synthetic fiscal period types
Utility:	Allows different fiscal period types reported by the company to be used in the calculation of synthetic fiscal period types.
Parameter mnemonic:	FA_PERIOD_TYPE_SOURCE
Abbreviated mnemonic(s):	FPTS
Default value:	P
Example query:	get(IS_EPS(**FPTS=A**)) for (['IBM US Equity'])

Parameter available inputs:

Value	**Description**	**Functionality**
P	Primary	Sources the data for the FA_PERIOD_TYPE parameter from the primary fiscal periodicity reported by the company[6].
A	Annual	Sources the data for the FA_PERIOD_TYPE parameter from Annual periods reported by the company.
Q	Quarterly	Sources the data for the FA_PERIOD_TYPE parameter from Quarterly periods reported by the company.
S	Semiannual	Sources the data for the FA_PERIOD_TYPE parameter from Semiannual periods reported by the company.

[6] Field PRIMARY_PERIODICITY returns the primary reporting periodicity of a company.

2.3. Period offset

Parameter description: Controls the number of fiscal periods offset from period 0. Period 0 is defined as the last fiscal period reported by the company.

Utility: Allows a relative offset from period 0, thus providing a dynamic update of the fiscal periods used as the company reports new fiscal periods. It offers the ability to request both a single fiscal period and a range of fiscal periods, always using a relative offset from period 0.

Parameter mnemonic: FA_PERIOD_OFFSET

Abbreviated mnemonic(s): FPO

Default value: 0R

Example query: get(IS_EPS(**FPO='1'**)) for (['IBM US Equity'])
get(IS_EPS(**FPO=range('-3','3')**)) for (['IBM US Equity'])

Parameter available inputs:

Value	Description	Functionality
[Integer]	Number of fiscal periods	A negative integer returns data for fiscal periods prior to period 0 (i.e. fiscal periods in the past), whereas a positive integer returns data for fiscal periods after period 0 (i.e. fiscal period in the future). For example, **FPO='-1'** returns data from the prior fiscal period, whereas **FPO='1'** returns from the next fiscal period. If not specified, the length of 1 offset is determined by the FA_PERIOD_TYPE parameter.
[Integer] + [R]	Number of rolling fiscal periods	**FPO='-1R'** returns data from the prior fiscal period which is determined on a rolling basis. 'G' (generic) is also an alternative input instead of 'R' (rolling).
[Integer] +[F]	Number of fixed[7] fiscal periods	**FPO='-1F'** returns data from the prior fiscal period which is determined on a fixed basis.
[Integer] + [A]	Number of fiscal periods with annual offset length	**FPO='-1A'** returns data from the prior fiscal period with one annual offset from period 0.
[Integer] + [S]	Number of fiscal periods with semiannual offset length	**FPO='-1S'** returns data from the prior fiscal period with one semiannual offset from period 0.

[7] The “fixed” period offset (“F”) is only applicable to a query involving a range in the AS_OF_DATE parameter as illustrated in this sample query which uses the hypothetical data shown in Section 3.

[Integer] + [Q]	Number of fiscal periods with quarterly offset length	**FPO='-1Q'** returns data from the prior fiscal period with one quarter offset from period 0.
[Integer range]	Range between a start and end offset	**FPO=range('-3','3')** returns 7 points of data from fiscal periods -3, -2, -1, 0, 1, 2 and 3. If not specified, the length of 1 offset is determined by the FA_PERIOD_TYPE parameter.
[Integer range] + [A]	Range between a start and end offset with annual offset length	**FPO=range('-3A','3A')** returns 7 points of data from fiscal periods -3A, -2A, -1A, 0, 1A, 2A and 3A where the length of the offset between the fiscal periods is one annual.
[Integer range] + [S]	Range between a start and end offset with semiannual offset length	**FPO=range('-3S','3S')** returns 7 points of data from fiscal periods -3S, -2S, -1S, 0, 1S, 2S and 3S where the length of the offset between the fiscal periods is one semiannual
[Integer range] + [Q]	Range between a start and end offset with quarterly offset length	**FPO=range('-3Q','3Q')** returns 7 points of data from fiscal periods -3Q, -2Q, -1Q, 0, 1Q, 2Q and 3Q where the length of the offset between the fiscal periods is one quarter.
[Integer] + [R]+ [A]	Number of fiscal periods with rolling annual offset length	**FPO='-1RA'** returns data from the prior fiscal period with one annual offset from period 0.
[Integer] + [R] + [S]	Number of fiscal periods with rolling semiannual offset length	**FPO='-1RS'** returns data from the prior fiscal period with one semiannual offset from period 0.
[Integer] + [R]+ [Q]	Number of fiscal periods with rolling quarterly offset length	**FPO='-1RQ'** returns data from the prior fiscal period with one quarter offset from period 0.
[Integer] + [F]+ [A]	Number of fiscal periods with fixed annual offset length	**FPO='-1FA'** returns data from the prior fiscal period with one annual offset from period 0.
[Integer] + [F] + [S]	Number of fiscal periods with fixed semiannual offset length	**FPO='-1FS'** returns data from the prior fiscal period with one semiannual offset from period 0.
[Integer] + [F]+ [Q]	Number of fiscal periods with fixed quarterly offset length	**FPO='-1FQ'** returns data from the prior fiscal period with one quarter offset from period 0.

2.4. Period reference

Parameter description:	Defines the fiscal year and period which is the anchor for offset reference.
Utility:	Allows an absolute reference to a fiscal period and/or to the period end date of a fiscal period. It offers the ability to request both a single fiscal period and a range of fiscal periods, always using an absolute reference to a fiscal period and/or to the period end date of a fiscal period.
Parameter mnemonic:	FA_PERIOD_REFERENCE
Abbreviated mnemonic(s):	FPR
Default value:	None[8]

Example query:
get(IS_EPS(**FPR='2015Q3'**)) for (['IBM US Equity'])
get(IS_EPS(**FPR='2015-09-30'**)) for (['IBM US Equity'])
get(IS_EPS(**FPR=range('2015Q3','2017Q3')**)) for (['IBM US Equity'])
get(IS_EPS(**FPR=range('2015-09-30','2017-09-30')**)) for (['IBM US Equity'])

Parameter available inputs:

Value	**Description**	**Functionality**
Date	Period end date reference	Returns data from the latest fiscal period with a period end date on or before the period end date reference specified. For example, **FPR='12/31/2015'** will return data from the fiscal period ending on December 31, 2015.
Year	Period end date reference from fiscal year	Example: **FPR='2015'** returns data from the latest fiscal period with a period end date on or before the 2015A period end date.
Year + [A]	Period end date reference from annual fiscal period	Example: **FPR='2015A'** returns data from the latest fiscal period with a period end date on or before the 2015A period end date.
Year + [Q1]	Period end date reference from annual fiscal period	Example: **FPR='2015Q1'** returns data from the latest fiscal period with a period end date on or before the 2015Q1 period end date
Year + [Q2]	Period end date reference from annual fiscal period	Example: **FPR='2015Q2'** returns data from the latest fiscal period with a period end date on or before the 2015Q2 period end date
Year + [Q3]	Period end date reference from annual	Example: **FPR='2015Q3'** returns data from the latest fiscal period with a period end date on or before the

[8] A query that does not specify the FA_PERIOD_REFERENCE parameter returns the latest period reported

	fiscal period	2015Q3 period end date
Year + [Q4]	Range between a start and end offset with annual offset length	Example: **FPR='2015Q4'** returns data from the latest fiscal period with a period end date on or before the 2015Q4 period end date
Year + [S1]	Period end date reference from annual fiscal period	Example: **FPR='2015S1'** returns data from the latest fiscal period with a period end date on or before the 2015S1 period end date
Year + [S2]	Period end date reference from annual fiscal period	Example: **FPR='2015S2'** returns data from the latest fiscal period with a period end date on or before the 2015S2 period end date
Date range	Period end date reference range	Returns data for a range of fiscal periods, where the first fiscal period ends on a date on or before the start date defined in the range and the last fiscal period ends on a date on or before the end date defined in the range. For example, **FPR=range('2013-12-31','2017-12-31')** will return data from the latest fiscal period with a period end date on or before 12/31/2013 to the latest fiscal period with a period end date on or before 12/31/2017.

2.5. As of date

Parameter description:	Sets the observation date of an observer with perfect information
Utility:	Allows an observer to move back in time and request publically available data that was observed as of a given date in the past. It is useful for point in time analysis combining market data (such a pricing data) with fundamental data.
Parameter mnemonic:	AS_OF_DATE
Abbreviated mnemonic(s):	DATES
Default value:	Today's date

Example query:
get(IS_EPS(**DATES=2016-09-15**)) for (['IBM US Equity'])
get(IS_EPS(**DATES=range(2016-01-01,2017-12-31)**)) for (['IBM US Equity'])
get(IS_EPS(**DATES=-10D**)) for (['IBM US Equity'])
get(IS_EPS(**DATES=range(-10D,0D)**)) for (['IBM US Equity'])

Parameter available inputs:

Value	Description	Functionality
Date	Absolute As of date	Returns the data that would have been known to an observer having perfect access to public information. For example, **DATES=2016-09-15** will return the data that was publicly available as of September 15, 2016
Date range	Range of absolute as of dates	Returns the data that would have been known to an observer having perfect access to public information between the two observation dates specified. For example, **DATES=range(2016-01-01,2017-12-31)** will return a stream of data that represents the data that was publicly available as of every date from January 1, 2016 to December 31, 2017
[Non positive Integer Range] + [D]	Range of relative as of dates expressed in days	Returns the data that would have been known to an observer having perfect access to public information between the two relative observation dates specified. For example, assuming today's date is December 5, 2016, **DATES=range(-10D,0D)** will return a stream of data that represents the data that was publicly available as of every date from November 25, 2016 to December 5, 2016.
[Non positive Integer Range]	Range of relative as of dates expressed in weeks	Returns the data that would have been known to an observer having perfect access to public information between the two relative observation dates specified. For example, assuming today's date is December 5, 2016,

+ [W]		**DATES=range(-3W,-1W)** will return a stream of data that represents the data that was publicly available as of every date from November 14, 2016 to November 28, 2016.
[Non positive Integer Range] + [M]	Range of relative as of dates expressed in weeks	Returns the data that would have been known to an observer having perfect access to public information between the two relative observation dates specified. For example, assuming today's date is December 5, 2016, **DATES=range(-1M,0M)** will return a stream of data that represents the data that was publicly available as of every date from November 5, 2016 to December 5, 2016.
[Non positive Integer Range] + [Q]	Range of relative as of dates expressed in quarters	Returns the data that would have been known to an observer having perfect access to public information between the two relative observation dates specified. For example, assuming today's date is December 5, 2016, **DATES=range(-4Q,-2Q)** will return a stream of data that represents the data that was publicly available as of every date from December 5, 2015 to June 5, 2016.
[Non positive Integer Range] + [S]	Range of relative as of dates expressed in semiannual periods	Returns the data that would have been known to an observer having perfect access to public information between the two relative observation dates specified. For example, assuming today's date is December 5, 2016, **DATES=range(-1S,0S)** will return a stream of data that represents the data that was publicly available as of every date from June 5, 2016 to December 5, 2016.
[Non positive Integer Range] + [Y]	Range of relative as of dates expressed in years	Returns the data that would have been known to an observer having perfect access to public information between the two relative observation dates specified. For example, assuming today's date is December 5, 2016, **DATES=range(-2Y,-1Y)** will return a stream of data that represents the data that was publicly available as of every date from December 5, 2014 to December 5, 2015.

2.6. Period year end

Parameter description: Realigns fiscal periods to a customized company calendar

Utility: Better alignment of fiscal periods for cross-company analysis

Parameter mnemonic: FA_PERIOD_YEAR_END

Abbreviated mnemonic(s): FYE

Default value: F

Example query: get(IS_EPS(**FYE='F'**)) for (['IBM US Equity'])

Parameter available inputs:

Value	Description	Functionality
F	Fiscal year end	Defines the fiscal periods of a company according to the standard fiscal calendar of that company[9].
C	Calendar year end	Re-defines the fiscal periods of a company based on a calendar year (ending on December 31).
C[MMDD]	Custom year end	Re-defines the fiscal periods of a company based on a fiscal year ending on a custom month and date.

Disclosure example: Company X has a fiscal year ending on September 30 and Company Y has a fiscal year ending on June 30. Both companies report Annual and Quarterly filings.

FYE parameter impact: The fiscal period names for company X and Y will be set as shown in the table below, depending on the value of the FYE parameter. The FYE parameters of 'C' or 'C0531' make fiscal period names consistent for both companies:

	Company X			**Company Y**		
FYE parameter value	**F**	**C**	**C0531**	**F**	**C**	**C0531**
Period ending on March 31	Q2	Q1	Q3	Q3	Q1	Q3
Period ending on June 30	Q3	Q2	Q4/A	Q4/A	Q2	Q4/A
Period ending on September 30	Q4/A	Q3	Q1	Q1	Q3	Q1
Period ending on December 31	Q1	Q4/A	Q2	Q2	Q4/A	Q2

[9] Bloomberg defines the standard fiscal calendar for a company based on the period end date of a company's annual report. If the period end date of a company's annual report falls between 01/15/[YYYY] to 01/14/[YYYY+1], Bloomberg defines that annual report as Fiscal Year [YYYY]. Interim (semiannual or quarterly) reports are then defined based the definition of the annual report that includes the timeframes covered by those interim reports.

2.7. Filing Status

Parameter description: Filters the filing types data can be returned from

Utility: Provides the ability to filter out filings with less complete data, such as preliminary filings.

Parameter mnemonic: FA_FILING_STATUS

Abbreviated mnemonic(s): FS

Default value: MR

Example query: get(IS_EPS(**FS=MR**)) for (['IBM US Equity'])

Parameter available inputs:

Value	Description	Functionality
MR	Most recent	Returns data from the most recent version of a fiscal period, among the different versions of filings available for that fiscal period
MRXP	Most recent excluding preliminary filings	Returns the most recent data for a fiscal period, among the different versions of filings available for that fiscal period, excluding earnings or preliminary records
LR	Least recent	Returns the least recent data for a fiscal period, among the different versions of filings available for that fiscal period (i.e. the first version of data that a company reported for a fiscal period)

2.8. Adjustments for abnormal items

Parameter description:	Defines whether accounting adjustments for abnormal items should apply
Utility:	Allows, where available and applicable, the exclusion of the impact of abnormal items that may impact a company's performance
Parameter mnemonic:	FA_ADJUSTED
Abbreviated mnemonic(s):	ADJ
Default value:	N
Example query:	get(IS_EPS(**ADJ=Y**)) for (['IBM US Equity'])

Parameter available inputs:

Value	**Description**	**Functionality**
N	Unadjusted	Returns data unadjusted for abnormal items
Y	Adjusted	Returns data adjusted to exclude the impact of abnormal items, where available[10].

[10] This parameter only impacts data for companies where ADJUSTED_FINANCIALS_INDICATOR = Y

2.9. Actuals or Estimates

Parameter description:	Defines whether the data returned is actual or estimated data
Utility:	Enables a user to define whether the data returned is actual data reported by the company or estimated data contributed by research analysts
Parameter mnemonic:	FA_ACT_EST_DATA
Abbreviated mnemonic(s):	AE
Default value:	AE
Example query:	get(IS_EPS(**AE=E**)) for (['IBM US Equity'])

Parameter available inputs:

Value	Description	Functionality
AE	Actuals and Estimates	Returns actual data reported by the company for fiscal periods already reported by the company and estimates data for fiscal periods not reported by the company, yet.
A	Actuals	Returns actual data reported by the company for fiscal periods already reported by the company.
E	Estimates	Returns estimates data for both fiscal periods already reported by the company and for fiscal periods not reported by the company, yet.

2.10. Consolidated or Unconsolidated

Parameter description: Defines whether the data returned is consolidated or unconsolidated

Utility: Enables a user to request consolidated or unconsolidated data, where available

Parameter mnemonic: FA_CONSOLIDATED

Abbreviated mnemonic(s): CONSL

Default value: Y

Example query: get(IS_EPS(**CONSL=Y**)) for (['IBM US Equity'])

Parameter available inputs:

Value	**Description**	**Functionality**
Y	Consolidated	Returns consolidated data.
N	Unconsolidated	Returns unconsolidated data, where available[11].

[11] Unconsolidated data is available only for companies where EQY_CONSOL_FUND_AVAIL = Y

2.11. Accounting Standard

Parameter description:	Defines the accounting standard
Utility:	Enables a user to request data in different accounting standards, where available[12]
Parameter mnemonic:	FA_ACCT_STD
Abbreviated mnemonic(s):	STD
Default value:	MIXED
Example query:	get(IS_EPS(**STD=MIXED**)) for (['IBM US Equity'])

Parameter available inputs:

Value	**Description**	**Functionality**
MIXED	Mixed	Returns data in the most widely used accounting standard, depending on data availability for a company. If data sets in more than one accounting standard are available for a fiscal period, the priority order of the accounting standard returned is IFRS, if available, otherwise US GAAP, if available, otherwise Local GAAP.
IFRS	IFRS	Returns the data set reported by company in the IFRS accounting standard, if available.
US	US GAAP	Returns the data set reported by company in the US GAAP accounting standard, if available.
LOCAL	Local GAAP	Returns the data set reported by company in the Local GAAP accounting standard, if available.

[12] Data sets in more than one accounting standard are available for companies where MULTIPLE_ACCOUNTING_STANDARD_IND = Y

2.12. Capital Changes Adjustments

Parameter description: Sets whether adjustments for capital changes should be applied.

Utility: Gives the ability to display the data adjusted or unadjusted for corporate actions that impact the capital represented by one share issued by the company

Parameter mnemonic: CAPITAL_CHANGES_ADJUST

Abbreviated mnemonic(s): CA_ADJ

Default value: SPLITS

Example query: get(IS_EPS(**CAP_CHG_ADJ =Y**)) for (['IBM US Equity'])

Parameter available inputs:

Value	**Description**	**Functionality**
SPLITS	Adjusted for splits	Returns data adjusted for split-like capital changes, namely spin-offs, stock splits (consolidations), stock dividends (bonuses) and rights offerings (entitlements).
RAW	Raw data	Returns data not adjusted for any capital changes

2.13. Fill

Parameter description: Determines what should be used to fill observations with no data available

Utility: Offers the ability to get a continues stream of data even when there is no data available for certain observations

Parameter mnemonic: FILL

Abbreviated mnemonic(s): FILL

Default value: NA

Example query: get(IS_EPS(**FILL =NA**)) for (['IBM US Equity'])

Parameter available inputs:

Value	Description	Functionality
NA	Not available	Returns NA if no data is available
PREV	Previous value	Returns the data from the previous date if no data is available for the current date

2.14. Estimates Source

Parameter description:	Determines the data source
Utility:	Offers the ability to get estimates data from a specific source, such as a specific sell-side research firm
Parameter mnemonic:	EST_SOURCE
Abbreviated mnemonic(s):	
Default value:	BST
Example query:	get(IS_EPS(**EST_SOURCE =BST**)) for (['IBM US Equity'])
Parameter available inputs:	

Value	Description	Functionality
BST	Bloomberg Standard	Returns the estimates data from the Bloomberg standard estimates consensus
BLI	Bloomberg Leading Indicator	Returns the estimates data from the Bloomberg leading indicator estimates consensus which includes only the estimates updated within the last four weeks
BPE	Bloomberg Post Event	Returns the estimates data from the Bloomberg post-event estimates consensus which includes only the estimates updated after the last earnings announcement date
CMPY	Company Reported	Returns estimates data provided by the company (i.e. guidance).
[BBB]	Broker Code	Returns the estimates data from an individual sell-side research contributor[13]
[‘BBB’,’CCC’, …]	List of Broker Codes	Returns the estimates data from a list of sell-side research contributors .
BROKERS_ALL	All Brokers	Returns all brokers who have provided Bloomberg an estimate for the company.
BROKERS_ALL_ACTIVE	All Active Brokers	Returns all brokers who have provided Bloomberg an estimate for the company within the past 500 calendar days of the specified AS_OF_DATE.

[13] Entitlement for the research for the selected contributor is required. To view the entitlements or request entitlement for a particular research contributor, the ENTC <GO> function on the Bloomberg Terminal should be used.

2.15. Currency

Parameter description: Determines the currency the data retrieved should be displayed in

Utility: Offers the ability to compare fundamental data between companies reporting in different currencies

Parameter mnemonic: CURRENCY

Abbreviated mnemonic(s): None

Default value: None[14]

Example query: get(IS_EPS(**CURRENCY =USD**)) for (['IBM US Equity'])

Parameter available inputs:

Value	Description	Functionality
[3-letter code]	ISO currency code	Converts fundamental data to the specified currency

[14] If no currency is specified, the query will return data in the currency in which a given company reports its financial results. Field EQY_FUND_CRNCY displays the currency in which a company currently reports its financial results.

2.16. Currency Conversion Method

Parameter description:	Determines the method used to convert fundamental data from one currency to another
Utility:	
Parameter mnemonic:	CURRENCY_CONVERSION_METHOD
Abbreviated mnemonic(s):	CUR_METH
Default value:	ACC
Example query:	get(IS_EPS(**CUR_METH =ACC**)) for (['IBM US Equity'])

Parameter available inputs:

Value	Description	Functionality
ACC	Accounting Method	Converts fundamental data to the specified currency using the average currency conversion rate during the fiscal period for flow accounting concepts (such as Revenue) and the currency conversion rate as of the end of the fiscal period for stock accounting concepts (such as Assets).
AOD	As-of-date Method	Converts fundamental data to the specified currency using the currency conversion rate applicable to the observers date as specified by the AS_OF_DATE parameter.

2.17. Impact of Capitalized Operating Leases

Parameter description:	Determines whether the data item includes or excludes the impact of the new Operating Lease Accounting Standard (ASC 842 and IFRS 16)
Utility:	
Parameter mnemonic:	EXCL_OPER_LEASES
Abbreviated mnemonic(s):	EXOL
Default value:	N
Example query:	get(NET_DEBT (**EXOL=Y**)) for (['IBM US Equity'])

Parameter available inputs:

Value	Description	Functionality
N	Includes the impact of Operating Leases	Returns data that includes the impact of operating leases
Y	Excludes the impact of Operating Leases	Returns data the excludes the impact of operating leases

2.18. Actuals and Estimates Mapping

Parameter description: Determines whether the estimate data returned is generically or precisely aligned to actuals data.

Utility: Offers the ability to retrieve estimates data that is on the same basis as the company reported actuals.

Parameter mnemonic: ACT_EST_MAPPING

Abbreviated mnemonic(s): None

Default value: GENERIC

Example query: get(SALES_REV_TURN (**ACT_EST_MAPPING=PRECISE**)) for (['IBM US Equity'])

Parameter available inputs:

Value	**Description**	**Functionality**
GENERIC	Preferred Basis Estimates	Returns estimate data which is generically aligned to Actuals following the same methodology for all companies. Provides better coverage and history compared to the PRECISE options, but it does not guarantee the alignment. It is the recommended option for back testing, which requires history and coverage.
PRECISE	Company Basis Estimates	Returns Estimates data which is aligned to Actuals following a company specific methodology. It provides a precise alignment but with less coverage and history since it requires deeper estimates content from sell-side research contributors. It is the recommended option for company modeling.

3. <u>Functions</u>

3.1. Number of contributors (partially supported)[15]

Function description:	Provides a count of the number of contributors
Utility:	Offers the ability to assess whether data such as estimates consensus is robust based on the number of contributors
Function mnemonic:	CONTRIBUTOR_COUNT(DATA_ITEM)

Example query:
get(**CONTRIBUTOR_COUNT(**IS_EPS(FPO=1,FPT=A)**)**) for (['IBM US Equity'])

Values for the DATA_ITEM parameter:

Value	Description	Functionality
[Data Item]	Data item	Any data item and related parameters

[15] Not supported for (i) FA_PERIOD_TYPE values different from A, S or Q; (ii) macro fields (i.e. fields that are calculated through BQL expressions); and (iii) mode=cached_only.

3.2. Contributor Statistics (partially supported)[16]

Function description: Provides statistics about the contributors

Utility:

Function mnemonic: CONTRIBUTOR_STATS (DATA_ITEM,STAT_TYPE)

Example query:
get(**CONTRIBUTOR_STATS(**IS_EPS(FPO=1,FPT=A)**,**MEDIAN**)**) for (['IBM US Equity'])

Values for the DATA_ITEM parameter:

Value	Description	Functionality
[Data Item]	Data item	Any data item and related parameters

Values for the STAT_TYPE parameter:

Value	Description	Functionality
AVG	Average	Average of the values from contributors included in the consensus. This is the default value of the parameter.
MEDIAN	Median	Median of the values from contributors included in the consensus.
MAX	Highest Value	Highest value among contributors included in the consensus.
MIN	Lowest Value	Lowest value among contributors included in the consensus.
STD	Standard Deviation	Standard deviation of values from contributors included in the consensus.

[16] Not supported for (i) FA_PERIOD_TYPE values different from A, S or Q; (ii) macro fields (i.e. fields that are calculated through BQL expressions); and (iii) mode=cached_only.

3.3. Contributor revisions (partially supported)[17]

Function description: Provides information about the contributor revisions

Utility: Offers insights into the sentiment about a given measure

Function mnemonic: CONTRIBUTOR_REVISIONS (DATA_ITEM,REVISION_TYPE,WINDOW)

Example query:
get(**CONTRIBUTOR_REVISIONS**(IS_EPS(FPO=1,FPT=A)**,**NETUP**,**13W**)**) for (['IBM US Equity'])

Values for the DATA_ITEM parameter:

Value	**Description**	**Functionality**
[Data Item]	Data item	Any data item and related parameters

Values for the REVISION_TYPE parameter:

Value	**Description**	**Functionality**
NUMUP	Number of upgrades	Number of upgrade events from contributors included in the consensus during the period specified in WINDOW.
NUMDN	Number of downgrades	Number of downgrade events from contributors included in the consensus during the period specified in WINDOW.
NUMCONF	Number of confirmations	Number of confirmation events from contributors included in the consensus during the period specified in WINDOW.
NUMADD	Number of additions	Number of coverage initiation events from contributors included in the consensus during the period specified in WINDOW.
NUMDROP	Number of drops	Number of coverage drop events from contributors included in the consensus during the period specified in WINDOW.
NUMCHG	Number of changes	Number of revision events changing a value from contributors included in the consensus during the period specified in WINDOW.
NUMUNCHG	Number of no changes	Number of days for which each contributor included in the consensus did not change any values during the period specified in WINDOW.
NETUP	Number of net upgrades	Number of contributors included in the consensus whose estimate as of the end of the WINDOW was

[17] Not supported for (i) FA_PERIOD_TYPE values different from A, S or Q; (ii) macro fields (i.e. fields that are calculated through BQL expressions); and (iii) mode=cached_only.

		higher than the estimate as of the beginning of the WINDOW.
NETDN	Number of net downgrades	Number of contributors included in the consensus whose estimate as of the end of the WINDOW was lower than the estimate as of the beginning of the WINDOW.
NETCONF	Number of net confirmations	Number of contributors included in the consensus who confirmed that their estimate as of the end of the WINDOW was the same as of the estimate as the beginning of the WINDOW.
NETADD	Number of net additions	Number of contributors included in the consensus as of the end of the WINDOW who were not covering the security as of the beginning of the WINDOW.
NETDROP	Number of net drops	Number of contributors included in the consensus as of the beginning of the WINDOW who were no longer covering the security as of the end of the WINDOW.
NETCHG	Number of net changes	Number of contributors included in the consensus whose estimate as of the end of the WINDOW was different from the estimate as of the beginning of the WINDOW.
NETUNCHG	Number of net no changes	Number of contributors included in the consensus whose estimate as of the end of the WINDOW was the same as the estimate as of the beginning of the WINDOW.

Values for the WINDOW parameter:

Value	Description	Functionality
[Integer]+D	Number of days	Specifies the number of days for contributor revisions
[Integer]+W	Number of weeks	Specifies the number of weeks for contributor revisions

4. ***Data sources discoverability***

4.1. Estimate Source Code

A user can then get the list of sell-side research firms which have ever contributed estimates for a company by running one of the following queries:

get(**EST_SOURCE_CODE**) for (['IBM US Equity'])

get(**EST_SOURCE_CODE(EST_SOURCE=BROKERS_ALL)**) for (['IBM US Equity'])

A user can also get the list of sell-side research firms which are actively[18] contributing estimates for a company, by running the following query:

get(**EST_SOURCE_CODE(EST_SOURCE=BROKERS_ALL_ACTIVE)**)
for (['IBM US Equity'])

An example output of the queries above (assuming the AS_OF_DATE is 2020-09-10) is shown below:

AS_OF_DATE	**REVISION_DATE**	**FIRM_NAME**	**EST_SOURCE_CODE**
2020-09-10	2020-09-07	BofA Securities	BAM
2020-09-10	2020-07-21	BMO Capital Markets	BMO
...	…	...	…

[18] Active is defined as any sell-side research firm that has provided an estimate revision within 2 years of the user specified AS_OF_DATE

5. ***Data retrieval for segments***

Please note that only the members of the SPX as of November 23rd 2020 are supported.

The data retrieval for segments follows the same pattern of data retrieval for a company, the same data item is used and the data items take the same parameters as when retrieving the data for a company. The main difference between the data retrieval for a segment and a company is the object the data is requested for.

For example, the following are the BQL queries for requesting the annual Revenue data for Apple and iPhone (one of Apple Inc's segments) for 2018:

get(SALES_REV_TURN(FPR=2018,FPT=A)) for ('**AAPL US Equity**')

get(SALES_REV_TURN(FPR=2018,FPT=A)) for ('**SEG0000155493 Segment**')

Except for the identifier (highlighted in bold), the rest of the syntax is the same.

BQL offers two new objects for a user to be able to discover how a company breaks itself down into segments and the structure of such segments (namely if a segment is the child of some other segment):

Hierarchies represent the different methods that a company could use to break down its metrics to the different parts of the company. Such break downs could be by operating segments, business units, products, locations, end users, etc. The Segment Hierarchies section below provides the details on how a user can retrieve and examine the different break downs available for a given company.

Segments represent a part of company, such as a business unit, product, a location, an end user, etc and are organized into the Hierarchies. The Segment Details section below provides the details on how a user can retrieve and examine the Segments available for a company and/or how they are organized into a Hierarchy.

Finally for users which are already familiar with a given company, the Segment Data Retrieval section provides details on how a user can directly request the segment data without having to first discover what Hierarchies and/or Segments are available for a given company.

Hierarchies, Segments and Segment data in BQL are all supported for point-in-time data retrieval, similarly to the rest of the Company Financials data.

5.1. Segment Hierarchies

Hierarchies represent the different methods that a company can use to break itself down into parts. For example, the company can break itself down by business unit, operating segments, locations, products, etc. A hierarchies() universe function is offered for users to discover the hierarchies of a company.

Function description:	Retrieves the hierarchies available for a company
Utility:	Provides details about the different methods that a company uses to break itself down into parts
Function signature:	hierarchies(<SYMBOLS>,<DATES>)

Example query:

get(HIERARCHY_NAME) for (**hierarchies**(‘BTU US Equity’))
get(HIERARCHY_NAME) for (**hierarchies**('BTU US Equity',DATES=2018-01-01))

Function parameters:

Value	Description	Functionality
SYMBOLS	Equity Ticker	Allows a user to specify the equity ticker of the company the user wants to retrieve the hierarchies for. A Symbol entry is required.
DATES	Observation date	Specifies the date of the observation of the hierarchy for point-in-time retrieval of the hierarchies available as of a given date since segments could be added or removed from a given hierarchy over time. The DATES parameter is not required and the default value, if none is specified, is today’s date.

The field HIERARCHY_NAME will return the following associative columns:

Value	Description	Functionality
ID	Identifier	Identifier of the hierarchy starting with the letter “H” and followed by an integer. The ID of a Hierarchy is unique within a company, but it is not unique across companies.
HIERARCHY_REPORTABLE	Reportable Hierarchy	Returns Y or N to indicate whether the Hierarchy contains the reportable segments of the company, which is the

		main method that the company uses to organize itself and break itself into parts. As any given time, there is at least one and only one reportable hierarchy for which this associative column will return Y. Usually the reportable Hierarchy is the one that contains most of the segment data disclosure by the company
FUNDAMENTAL_TICKER	Company’s ticker	Returns the ticker of the company the hierarchy is part of
HIERARCHY_REVISION_DATE	Revision Date	Returns the date the Hierarchy was last revised, i.e. a segment was added or removed from the hierarchy and/or the structure of the hierarchy changed.
AS_OF_DATE	Observer’s Date	The as-of-date of the observer
HIERARCHY_NAME	Name of the Hierarchy	Short description of the hierarchy which indicates the method that the company user to break down itself into parts.

5.2. *Segment Details*

Segments represent the different parts of a company. For example, a company could have business units, products, locations, assets, etc. A segments() universe function is offered for users to discover the segments of a company.

Function description: Retrieves the segments available for a company

Utility: Provides details about the segments of a company

Function signature: segments(<SYMBOLS>,<TYPE>,<DATES>)

Example query:

```
get(SEGMENT_NAME) for (segments(‘BTU US Equity’))
get(SEGMENT_NAME) for (segments('BTU US Equity',TYPE=REPORTABLE_HIERARCHY))
get(SEGMENT_NAME) for (segments('BTU US Equity',TYPE=REPORTABLE_HIERARCHY,DATES=2018-01-01))
```

Universe function parameters:

Value	**Description**	**Functionality**
SYMBOLS	Equity Ticker	Allows a user to specify the equity ticker of the company the user wants to retrieve the segments for. A Symbol entry is required.
TYPE	Segments universe type	Allows a user to specify the desired set of segments. The available types are as follows: REPORTABLE_HIERARCHY (default value) Returns the segments which are part of the reportable hierarchy (which is the hierarchy where the HIERARCHY_REPORTABLE=Y), including the hierarchical structure of the segments in the reportable hierarchy. SEGMENT_CATALOG Returns the flat list of unique segments available for the company, not organized in any hierarchical structure REPORTED_HIERARCHIES Returns the list of segments, including the company reported hierarchy(ies) they are part of. A segment could appear more than once with this type selected, since a segment can be part of one or more company reported hierarchies BICS_HIERARCHY (*not supported yet*)

		Returns the list of standardized (BICS) segments, including the hierarchical structure of the BICS segments in the Bloomberg Industry Classification System (BICS) GEO_HIERARCHY (*not supported yet*) Returns the list of standardized (GEO) segments, including the hierarchical structure of the GEO segments in the Bloomberg Geographic Classification System (GEO)
DATES	Observation date	Specifies the date of the observation of the segments for point-in-time retrieval of the segments available as of a given date. The DATES parameter is not required and the default value, if not specified, is today’s date.

The field SEGMENT_NAME will return the following associative columns:

Value	**Description**	**Functionality**
ID	Identifier	Identifier of the segment starting with the letter “SEG” and followed by a string on numbers. The ID of a Segment is unique across all companies.
SEGMENT_REPORTABLE	Reportable Segment	Returns Y or N to indicate whether the Segment is reportable which generally means that it represents more than 10% of the total revenue, profit or assets of the company and it is specifically disclosed by the company as being a reportable segment or equivalent.
FUNDAMENTAL_TICKER	Company’s ticker	Returns the ticker of the company the segment is part of
SEGMENT_REVISION_DATE	Revision Date	Returns the date the Segment was last revised (for example the Segment label could change over time)
AS_OF_DATE	Observer’s Date	The as-of-date of the observer
SEGMENT_NAME	The label of the segment	Short description of what the Segment is
SEGMENT_TYPE	Type of the segment	Returns one of the following segment types: - OPERATING_SEGMENT - BUSINESS_UNIT - PRODUCT - SERVICE - BRAND

		- CUSTOMER - END_USER - LOCATION (destination) - DISTRIBUTION_CHANNEL - PAYER_SOURCE - CORPORATE - ASSET - INDUSTRY - SUBSIDIARY - LOCATION_ORIGINATION - GEO (*not supported yet*) - BICS (*not supported yet*)
SEGMENT_ACTIVE	Active or Inactive	Returns Y for a Segment which is active on the AS_OF_DATE, and N for a segment which is inactive (defunct) on the AS_OF_DATE
HIERARCHY_ID	ID of the Hierarchy	Returns the ID of the Hierarchy the Segment is part of (it will return null if TYPE = SEGMENT_CATALOG)
HIERARCHY_NAME	Name of the Hierarchy	Returns the Name of the Hierarchy the Segment is part of (it will return null if TYPE = SEGMENT_CATALOG)
HIERARCHY_LEVEL	Level in the Hierarchy	Returns a non-negative integer that shows the Hierarchy level the Segment is in (it will return null if TYPE = SEGMENT_CATALOG)
PARENT_SEGMENT_ID	ID of the Parent Segment	Returns the ID of the Segment which is the parent of the requested Segment in the hierarchy structure (it will return null if TYPE = SEGMENT_CATALOG or if HIERARCHY_LEVEL = 0)
PARENT_SEGMENT_NAME	Name of the Parent Segment	Returns the Name of the Segment which is the parent of the requested Segment in the hierarchy structure (it will return null if TYPE = SEGMENT_CATALOG or if HIERARCHY_LEVEL = 0)
SEGMENT_ORDER	Order of the Segment in Hierarchy	Returns the sort order of the Segment in the hierarchy structure (it will return null if TYPE = SEGMENT_CATALOG)

5.3. Segment Data Retrieval

In order to retrieve the Segment data (i.e. the data for a given data item for a Segment), there are two different paths that a user can follow:

1. Discover the hierarchies and/or segments available for the company as described in the Segment Hierarchies and Segment Details section and then request the data for the desired data item(s) for the desired segment(s); or

2. Request directly the desired data item(s) for the desired hierarchy

The rest of this section describes each of the above approaches in detail.

1. Discover the hierarchies/segments and then request the data

This approach is most useful for a user that wishes to have full control on how to model the segments of a company, without having to follow the hierarchical structures of segments designed by Bloomberg. In this approach a user uses the hierarchies() and segments() universe queries to understand what segments are available for a company and how the segments are organized in order to select the segments that the user is interested in to build a model for the company. For example, a user can run the following queries:

get(HIERARCHY_NAME) for (hierarchies(‘BTU US Equity’))

which will return the following:

ID	HIERARCHY_REPORTABLE	AS_OF_DATE	HIERARCHY_ACTIVE	HIERARCHY_REVISION_DATE	HIERARCHY_NAME
H1	Y	2020-09-10	Y	2017-03-22	By Operating Segment, Location, Service, Asset
H4	N	2020-09-10	Y	2017-03-22	By Location
...	…	…	...		

A user can then get the list of segments available by running one of the following queries:

get(SEGMENT_NAME) for (segments(‘BTU US Equity’))

If the reportable hierarchy is what the user is looking for, than the query above is sufficient to provide the segments in the reportable hierarchy.

But if the user desires to see the segments available in all hierarchies, the following query achieves that.

get(SEGMENT_NAME) for (segments('BTU US Equity',TYPE=REPORTED_HIERARCHIES))

If the user desires to focus on the segments of one of the hierarchies, the user can then filter in the hierarchy that they desire:

```
get(SEGMENT_NAME) for (filter(segments('BTU US
Equity',TYPE=REPORTED_HIERARCHIES),HIERARCHY_ID=H1))
```

The purpose of all the queries above is to enable the user to narrow down the list of the segments to the segments that the user desires to use in modeling the company. Once the user has identified the desired segments, the user can retrieve the data for those segments exactly the same way the data is retrieved for the company as described in the Parameters section. For example:

```
get(SALES_REV_TURN(FPR=2018,FPT=A)) for (‘SEG0000155493 Segment’)
```

The associative columns returned from the query above are the same as the associative columns described in the Associative Columns section with the addition of the associative columns described in the Segments Detail section above.

2. Request the data for the segments in a hierarchy directly

This approach is more useful for a user who wishes to rely on the hierarchy structure(s) already designed by Bloomberg. In this approach, the user can simply request the data item as follows:

```
get(SALES_REV_TURN(FPT=A,FPR=2018)) for (segments(‘BTU US Equity’))
```

The query above will return the revenue data for the 2018 Annual period for all the segments of BTU US which are included in the reportable hierarchy, together the information necessary to describe the structure of the segments in that hierarchy. The query above will return all the associative columns described in the Associative Columns section with the addition of the associative columns described in the Segments Detail section above.

If the user desires to get the segment data from all hierarchies, instead of just the reportable hierarchy, the user can run the following query:

```
get(SALES_REV_TURN(FPR=2018,FPT=A))
for (segments('BTU US Equity',TYPE=REPORTED_HIERARCHIES))
```

For a complete list of all the types of segments sets that the user can request the data for, please see the TYPES explanation in the Segment Details section above.

5.4. Segment data for universes of companies

The hierarchy, segment, and segment data queries are also supported for more than one company.

For example, a user can request the list of hierarchies, segments, or segment data for two companies, as follows:

```
get(HIERARCHY_NAME) for (hierarchies([‘BTU US Equity’, ‘AAPL US Equity’]))
```

```
get(SEGMENT_NAME) for (segments([‘BTU US Equity’, ‘AAPL US Equity’]))
```

```
get(SALES_REV_TURN(FPR=2018,FPT=A))
for (segments([‘BTU US Equity’, ‘AAPL US Equity’]))
```

In addition, the user can also request the list of hierarchies, segments, or segment data for universes of companies, such as indices, as follows:

```
get(HIERARCHY_NAME) for (hierarchies(members(‘INDU Index’)))
```

```
get(SEGMENT_NAME) for (segments(members(‘INDU Index’)))
```

```
get(SALES_REV_TURN(FPR=2018,FPT=A))
for (segments(members(‘INDU Index’)))
```

6. ***Interaction between parameters***

This section explains how different parameters interact with each other. For a more detailed description of each parameter and the available parameter inputs, please refer to the Parameters section.

The following example with be used throughout this section to illustrate the behavior of the several parameters when they interact with each other. The table below shows hypothetical data available for company XYZ US for field IS_EPS. Another assumption is that today's date is December 5, 2016.

FISCAL PERIOD	PERIOD END DATE	REVISION DATE	VALUES FOR IS_EPS(FPT='Q')	VALUES FOR IS_EPS(FPT=LTM)
2015 Q3	2015-09-30	2015-11-11	10	
2015 Q4	2015-12-31	2016-02-25	15	
2016 Q1	2016-03-31	2016-05-16	20	
2016 Q2	2016-06-30	2016-08-10	25	70
2016 Q3	2016-09-30	2016-08-31	29	89
2016 Q3	2016-09-30	2016-11-12	30	90
2016 Q4	2016-12-31	2016-08-31	33	107
2016 Q4	2016-12-31	2016-11-20	34	109
2016 Q4	2016-12-31	2016-12-02	35	110
2017 Q1	2017-03-31	2016-08-31	38	116
2017 Q1	2017-03-31	2016-11-20	39	128
2017 Q1	2017-03-31	2016-12-02	40	130
2017 Q2	2017-06-30	2016-08-31	43	139
2017 Q2	2017-06-30	2016-11-20	44	147
2017 Q2	2017-06-30	2016-12-02	45	150
2017 Q3	2017-09-30	2016-08-31	48	158
2017 Q3	2017-09-30	2016-11-20	49	166
2017 Q3	2017-09-30	2016-12-02	50	170
2017 Q4	2017-12-31	2016-08-31	53	182
2017 Q4	2017-12-31	2016-11-20	54	186
2017 Q4	2017-12-31	2016-12-02	55	190

The latest fiscal period reported by the company as of today is 2016 Q3 and therefore that is the FPO period 0. The data highlighted in grey is data that has been reported by the company, whereas the data non-highlighted, represents estimates data provided by the sell-side research.

6.1. Period offset & Period type

The following examples illustrate the interaction between the Fiscal period offset (FPO) and Fiscal period type (FPT) parameters.

Example query 1: get(IS_EPS) for (['XYZ US Equity'])

Since no value is specified in the query for the FPT and FPO parameters, the default values of FPT=LTM and FPO=0 are implied. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-11-12	2016-12-05	2016-09-30	90

Example query 2: get(IS_EPS(**FPT=Q**)) for (['XYZ US Equity'])

Since no value is specified in the query for the FPO parameter, the default value of FPO='0' is implied. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-11-12	2016-12-05	2016-09-30	30

Example query 3: get(IS_EPS(**FPO='0'**)) for (['XYZ US Equity'])

Since no value is specified in the query for the FPT parameter, the default value of FPT=LTM is implied. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-11-12	2016-12-05	2016-09-30	90

Example query 4: get(IS_EPS(**FPO='1'**)) for (['XYZ US Equity'])

Since the FPO value requested does not specify the length of the offset step, than the value of the FPT parameter determines what the length of the offset step is (in addition to determining the fiscal period type). Since no FPT value is specified in the query, the default FPT=LTM is implied. Therefore the query above will result in an offset of 1 LTM period from period 0. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-12-02	2016-12-05	2017-09-30	170

Example query 5: get(IS_EPS(**FPO='1',FPT=Q**)) for (['XYZ US Equity'])

Since the FPO value requested does not specify the length of the offset step, than the value of the FPT parameter determines what the length of the offset step is (in addition to determining the fiscal period type). Since FPT=Q, the query above will result in an offset of 1 Q period from period 0. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-12-02	2016-12-05	2016-12-31	35

Example query 6: get(IS_EPS(**FPO='1',FPT=LTM**)) for (['XYZ US Equity'])

Since the FPO value requested does not specify the length of the offset step, than the value of the FPT parameter determines what the length of the offset step is (in addition to determining the fiscal period type). Since FPT=LTM, the query above will result in an offset of 1 LTM period from period 0. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-12-02	2016-12-05	2017-09-30	170

Example query 7: get(IS_EPS(**FPO='1Q',FPT=LTM**)) for (['XYZ US Equity'])

Since the FPO value requested determines the length of the offset step as 1Q, and the FPT value requested determines the fiscal period type as LTM, the query above will result in an offset of 1 Q from the LTM period 0. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-12-02	2016-12-05	2016-12-31	110

Example query 8: get(IS_EPS(**FPO=range('-1','1')**)) for (['XYZ US Equity'])

Since the FPO range requested does not specify the length of the offset steps, than the value of the FPT parameter determines what the length of the offset steps is (in addition to determining the fiscal period type). Since no FPT value is specified in the query, the default FPT=LTM is implied. Therefore the query above will result in offset steps of 1 LTM period relative to period 0. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
	2016-12-05	2015-09-30	
2016-11-12	2016-12-05	2016-09-30	90
2016-12-02	2016-12-05	2017-09-30	170

Example query 9: get(IS_EPS(**FPO=range('-1','1'),FPT=Q**)) for (['XYZ US Equity'])

Since the FPO range requested does not specify the length of the offset steps, than the value of the FPT parameter determines what the length of the offset steps is (in addition to determining the fiscal period type). Since FPT=Q the query above will result in offset steps of 1 Q period relative period 0. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-08-10	2016-12-05	2016-06-30	25
2016-11-12	2016-12-05	2016-09-30	30
2016-12-02	2016-12-05	2016-12-31	35

Example query 10: get(IS_EPS(**FPO=range('-1','1'),FPT=LTM**)) for (['XYZ US Equity'])

Since the FPO range requested does not specify the length of the offset steps, than the value of the FPT parameter determines what the length of the offset steps is (in addition to determining the fiscal period type). Since FPT=LTM the query above will result in offset steps of 1 LTM period relative period 0. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
	2016-12-05	2015-09-30	
2016-11-12	2016-12-05	2016-09-30	90
2016-12-02	2016-12-05	2017-09-30	170

Example query 11:
get(IS_EPS(**FPO=range('-1Q','1Q'),FPT=LTM**)) for (['XYZ US Equity'])

Since the FPO range requested determines the length of the offset steps as 1Q, and the FPT value requested determines the fiscal period type as LTM, the query above will result in offset steps of 1 Q relative to period 0. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-08-10	2016-12-05	2016-06-30	70
2016-11-12	2016-12-05	2016-09-30	90
2016-12-02	2016-12-05	2016-12-31	110

6.2. Period reference & Period type

The following examples illustrate the interaction between the Fiscal period reference (FPR) and Fiscal period type (FPT) parameters.

Example query 1: get(IS_EPS) for (['XYZ US Equity'])

Since no value is specified in the query for the FPR parameter, the default FPT=LTM is implied and the query will return the latest LTM fiscal period reported. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-11-12	2016-12-05	2016-09-30	90

Example query 2: get(IS_EPS(**FPR='2016Q4'**)) for (['XYZ US Equity'])
get(IS_EPS(**FPR='2016-12-31'**)) for (['XYZ US Equity'])

Since no value is specified in the query for the FPT parameter, the default value of FPT=LTM is implied. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-12-02	2016-12-05	2016-12-31	110

Example query 3: get(IS_EPS(**FPT=LTM**)) for (['XYZ US Equity'])

Since no value is specified in the query for the FPR parameter, the latest LTM fiscal period reported will be returned. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-11-12	2016-12-05	2016-09-30	90

Example query 4: get(IS_EPS(**FPR='2016Q4',FPT=Q**)) for (['XYZ US Equity'])
get(IS_EPS(**FPR='2016-12-31',FPT=Q**)) for (['XYZ US Equity'])

The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-12-02	2016-12-05	2016-12-31	35

Example query 5: get(IS_EPS(**FPR='2017Q3',FPT=LTM**)) for (['XYZ US Equity']) get(IS_EPS(**FPR='2017-09-30',FPT=LTM**)) for (['XYZ US Equity'])

The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-12-02	2016-12-05	2017-09-30	170

Example query 6:
get(IS_EPS(**FPR=range('2015','2016')**)) for (['XYZ US Equity'])

Since the FPR range only specifies a fiscal year and does not specify any fiscal period type an annual fiscal period is implied[19]. Since the FPT parameter is not specified, the default value of FPT=LTM is implied. Therefore the query above will return LTM data in steps of 1 LTM period. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
	2016-12-05	2015-12-31	
2016-12-02	2016-12-05	2016-12-31	110

Example query 7:
get(IS_EPS(**FPR=range('2015-12-31','2016-12-31')**)) for (['XYZ US Equity'])

Since the FPR range only specifies a fiscal period end date and does not specify any fiscal period type, the length of the steps is determined by the FPT parameter. Since the FPT parameter is not specified, the default value of FPT=LTM is implied. Therefore the query above will return LTM data in steps of 1 LTM period. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
	2016-12-05	2015-12-31	
2016-12-02	2016-12-05	2016-12-31	110

Example query 8:
get(IS_EPS(**FPR=range('2015','2016'),FPT=Q**)) for (['XYZ US Equity'])

Since the FPR range only specifies a fiscal year and does not specify any fiscal period type, an annual fiscal period is implied[20]. The implied annual fiscal period also determines the length of the steps. Therefore the query above will return quarterly data in steps of 1 annual period. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS

[19] FPR=range(2015,2016) is therefore equivalent to FPR=range(2015A,2016A)

[20] FPR=range(2015,2016) is therefore equivalent to FPR=range(2015A,2016A)

2016-02-25	2016-12-05	2015-12-31	15
2016-12-02	2016-12-05	2016-12-31	35

Example query 9:
get(IS_EPS(**FPR=range('2015-12-31','2016-12-31'),FPT=Q**)) for (['XYZ US Equity'])

Since the FPR range only specifies a fiscal period end date and does not specify any fiscal period type, the length of the steps is determined by the FPT parameter. Therefore the query above will return quarterly data in steps of 1 quarterly period. The result of the query is shown below:

REVISION_DATE	**AS_OF_DATE**	**PERIOD_END_DATE**	**IS_EPS**
2016-02-25	2016-12-05	2015-12-31	15
2016-05-16	2016-12-05	2016-03-31	20
2016-08-10	2016-12-05	2016-06-30	25
2016-11-12	2016-12-05	2016-09-30	30
2016-12-02	2016-12-05	2016-12-31	35

Example query 10:
get(IS_EPS(**FPR=range('2016Q2','2016Q4'),FPT=LTM**)) for (['XYZ US Equity'])

Since the FPR range requested specifies the length of the steps as 1 quarter, and the value of the FPT parameter determines that LTM periods should be returned, the query above will result in steps of 1 quarter. The result of the query is shown below:

REVISION_DATE	**AS_OF_DATE**	**PERIOD_END_DATE**	**IS_EPS**
2016-08-10	2016-12-05	2016-06-30	70
2016-11-12	2016-12-05	2016-09-30	90
2016-12-02	2016-12-05	2016-12-31	110

Example query 11:
get(IS_EPS(**FPR=range('2016-06-30','2017-06-30'),FPT=LTM**)) for (['XYZ US Equity'])

Since the FPR range requested does not specify the length of the steps, the value of the FPT parameter determines the length of the steps (in addition to determining the fiscal period type). Therefore the query above will result in steps of 1 LTM period. The result of the query is shown below:

REVISION_DATE	**AS_OF_DATE**	**PERIOD_END_DATE**	**IS_EPS**
2016-08-10	2016-12-05	2016-06-30	70
2016-12-02	2016-12-05	2017-06-30	150

6.3. Period offset, Period reference & Period type

The following examples illustrate the interaction between the Fiscal period reference (FPR), Fiscal period offset (FPO) and Fiscal period type (FPT) parameters.

Example query 1: get(IS_EPS) for (['XYZ US Equity'])

Since no value is specified in the query for the FPR, FPO and FPT parameters, the default FPO='0' and FPT=LTM are implied and the query will return the latest LTM fiscal period reported. The result of the query is shown below:

REVISION_DATE	**AS_OF_DATE**	**PERIOD_END_DATE**	**IS_EPS**
2016-11-12	2016-12-05	2016-09-30	90

Example query 2: get(IS_EPS(**FPT=Q**)) for (['XYZ US Equity'])

Since no value is specified in the query for the FPR and FPO parameters, the default value of FPO=0 is implied. The result of the query is shown below:

REVISION_DATE	**AS_OF_DATE**	**PERIOD_END_DATE**	**IS_EPS**
2016-11-12	2016-12-05	2016-09-30	30

Example query 3: get(IS_EPS(**FPR='2016Q4'**)) for (['XYZ US Equity'])
get(IS_EPS(**FPR='2016-12-31'**)) for (['XYZ US Equity'])

Since no value is specified in the query for the FPO and FPT parameters, the default values of FPO=0 and FPT=LTM are implied. The result of the query is shown below:

REVISION_DATE	**AS_OF_DATE**	**PERIOD_END_DATE**	**IS_EPS**
2016-12-02	2016-12-05	2016-12-31	110

Example query 4: get(IS_EPS(**FPO='1'**)) for (['XYZ US Equity'])

Since the FPO value requested does not specify the length of the offset step, than the value of the FPT parameter determines what the length of the offset step is (in addition to determining the fiscal period type). Since no FPT value is specified in the query, the default FPT=LTM is implied. Therefore the query above will result in an offset of 1 LTM period from period 0. Since the value of the FPR parameter is not specified, it is implied that the period 0 references the latest fiscal period reported (in this case 2016 Q3). The result of the query is shown below:

REVISION_DATE	**AS_OF_DATE**	**PERIOD_END_DATE**	**IS_EPS**

2016-12-02	2016-12-05	2017-09-30	170

Example query 5: get(IS_EPS(**FPR='2016Q1',FPO='1')**) for (['XYZ US Equity'])
get(IS_EPS(**FPR='2016-03-31',FPO='1')**) for (['XYZ US Equity'])

Since the FPO value requested does not specify the length of the offset step, than the value of the FPT parameter determines what the length of the offset step is (in addition to determining the fiscal period type). Since no FPT value is specified in the query, the default FPT=LTM is implied. Therefore the query above will result in an offset of 1 LTM period from period 0. Since the value of the FPR parameter is specified as 2016Q1, the period 0 references 2015Q2 as the period the offset of 1 LTM period should be applied from. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-12-02	2016-12-05	2017-03-31	130

Example query 6:
get(IS_EPS(**FPR='2016Q1',FPO='1',FPT=Q)**) for (['XYZ US Equity'])
get(IS_EPS(**FPR='2016-03-31',FPO='1',FPT=Q)**) for (['XYZ US Equity'])

Since the FPO value requested does not specify the length of the offset step, than the value of the FPT parameter determines what the length of the offset step is (in addition to determining the fiscal period type). Therefore the query above will result in an offset of 1 Q period from period 0. Since the value of the FPR parameter is specified as 2016Q1, the period 0 references 2016Q1 as the period the offset of 1 Q period should be applied from. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-08-10	2016-12-05	2016-06-30	25

Example query 7:
get(IS_EPS(**FPR='2017Q1',FPO=range('-1','1'),FPT=Q)**) for (['XYZ US Equity'])
get(IS_EPS(**FPR='2017-03-31',FPO=range('-1','1'),FPT=Q)**) for (['XYZ US Equity'])

Since the FPO value requested does not specify the length of the offset steps in the range, than the value of the FPT parameter determines what the length of the offset steps is (in addition to determining the fiscal period type). Therefore the query above will result in offset steps of 1 Q period from period 0. Since the value of the FPR parameter is specified as 2017Q1, the period 0 references 2017Q1 as the period the offset steps of 1 Q period are relative to. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-12-02	2016-12-05	2016-12-31	35
2016-12-02	2016-12-05	2017-03-31	40
2016-12-02	2016-12-05	2017-06-30	45

Example query 8:
get(IS_EPS(**FPR='2017Q1',FPO=range('-1Q','1Q'),FPT=LTM)**) for (['XYZ US Equity'])
get(IS_EPS(**FPR='2017-03-31',FPO=range('-1Q','1Q'),FPT=LTM)**) for (['XYZ US Equity'])

Since the FPO value requested specifies the length of the offset steps in the range as 1 Q, the query above will result in offset steps of 1 Q period from period 0. Since the value of the FPR parameter is specified as 2017Q1, the period 0 references 2017Q1 as the period the offset steps of 1 Q period are relative to. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-12-02	2016-12-05	2016-12-31	110
2016-12-02	2016-12-05	2017-03-31	130
2016-12-02	2016-12-05	2017-06-30	150

Example query 9:
get(IS_EPS(**FPR=range('2015','2016'),FPO='1',FPT=LTM)**) for (['XYZ US Equity'])
get(IS_EPS(**FPR=range('2015-12-31','2016-12-31'),FPO='1',FPT=LTM)**) for (['XYZ US Equity'])

Since the FPR range only specifies a fiscal year and does not specify any fiscal period type, an annual fiscal period is implied[21]. The implied annual fiscal period determines the length of the steps in the range. Since the FPO value requested does not specify the length of the offset step, the FPT=LTM determines the length of the offset step as 1 LTM from period 0. Since the implied values from the FPR range are 2015A and 2016A, there are two periods 0 (2015A and 2016A) the offset step of 1 LTM period is relative to. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-12-02	2016-12-05	2016-12-31	110
2016-12-02	2016-12-05	2017-12-31	190

Example query 10:
get(IS_EPS(**FPR=range('2015','2016'),FPO='2Q',FPT=LTM)**) for (['XYZ US Equity'])
get(IS_EPS(**FPR=range('2015-12-31','2016-12-31'),FPO='2Q',FPT=LTM)**) for (['XYZ US Equity'])

Since the FPR range only specifies a fiscal year and does not specify any fiscal period type, an

[21] FPR=range(2015,2016) is therefore equivalent to FPR=range(2015A,2016A)

annual fiscal period is implied[22]. The implied annual fiscal period determines the length of the steps in the range. Since the FPO value requested specifies the length of the offset steps as 2 Q, the query above will result in offset steps of 2 Q periods from period 0. Since the implied values from the FPR range are 2015A and 2016A, there are two periods 0 (2015A and 2016A) the offset step of 2 Q periods is relative to. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-08-10	2016-12-05	2016-06-30	70
2016-12-02	2016-12-05	2017-06-30	150

Example query 11:
get(IS_EPS(**FPR=range('2015Q4','2016Q4'),FPO='2Q',FPT=LTM)**) for (['XYZ US Equity'])

Since the FPR range specifies a quarterly fiscal period, the length of the steps in the range is quarterly. Since the FPO value requested specifies the length of the offset steps as 2 Q, the query above will result in offset steps of 2 Q periods from period 0. Since the implied values from the FPR range are 2015Q4, 2016Q1, 2016Q2, 2016Q3 and 2016Q4, there are five periods 0 (2015Q4, 2016Q1, 2016Q2, 2016Q3 and 2016Q4) the offset step of 2 Q periods is relative to. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-08-10	2016-12-05	2016-06-30	70
2016-11-12	2016-12-05	2016-09-30	90
2016-12-02	2016-12-05	2016-12-31	110
2016-12-02	2016-12-05	2017-03-31	130
2016-12-02	2016-12-05	2017-06-30	150

Example query 12:
get(IS_EPS(**FPR=range('2015Q4','2016Q4'),FPO=range('-2Q','1Q'),FPT=LTM)**) for (['XYZ US Equity'])

The query above is invalid since both the FPR and the FPO parameters specify a range.

[22] FPR=range(2015,2016) is therefore equivalent to FPR=range(2015A,2016A)

6.4. Period offset, Period reference, Period type & As of date

The following examples illustrate the interaction between the Fiscal period reference (FPR), Fiscal period offset (FPO), Fiscal period type (FPT) and As of date (DATES) parameters.

Example query 1: get(IS_EPS) for (['XYZ US Equity'])

Since no value is specified in the query for the FPR, FPO, FPT and DATES parameters, the default FPO='0', FPT=LTM and DATES=2016-12-05 (today's date) values are implied and the query will return the latest LTM fiscal period reported as of December 5, 2016. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-11-12	2016-12-05	2016-09-30	90

Example query 2: get(IS_EPS(**DATES=2016-09-01**)) for (['XYZ US Equity'])

Since no value is specified in the query for the FPR, FPO and FPT parameters, the default FPO=0, and FPT=LTM values are implied and the query will return the latest LTM fiscal period reported as of September 1, 2016. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-08-10	2016-09-01	2016-06-30	70

Example query 3: get(IS_EPS(**FPT=Q,DATES=2016-09-01**)) for (['XYZ US Equity'])

Since no value is specified in the query for the FPR and FPO parameters, the default value pf FPO='0' is implied and the query will return the latest quarterly fiscal period reported as of September 1, 2016. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-08-10	2016-09-01	2016-06-30	25

Example query 4: get(IS_EPS(**FPR='2016Q4'**)) for (['XYZ US Equity'])
get(IS_EPS(**FPR='2016-12-31'**)) for (['XYZ US Equity'])

Since no value is specified in the query for the FPO, FPT and DATES parameters, the default values of FPO='0', FPT=LTM and DATES=2016-12-05 (today's date) are implied. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-12-02	2016-12-05	2016-12-31	110

Example query 5:
get(IS_EPS(**FPR='2016Q4',DATES=2016-11-30**)) for (['XYZ US Equity'])
get(IS_EPS(**FPR='2016-12-31',DATES=2016-11-30**)) for (['XYZ US Equity'])

Since no value is specified in the query for the FPO and FPT parameters, the default values of FPO='0' and FPT=LTM are implied and the query above will return an LTM period with a 0 offset from 2016 Q4 as of November 30, 2016. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-11-20	2016-11-30	2016-12-31	109

Example query 6:
get(IS_EPS(**FPR='2016Q4',FPO='-1Q',DATES=2016-09-01**)) for (['XYZ US Equity'])
get(IS_EPS(**FPR='2016-12-31',FPO='-1Q',DATES=2016-09-01**)) for (['XYZ US Equity'])

Since no value is specified in the query for the FPT parameter, the default value of FPT=LTM is implied and the query above will return an LTM period with an offset of 1 quarter from 2016 Q4 as of September 1, 2016. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-08-31	2016-09-01	2016-09-30	89

Example query 7:
get(IS_EPS(**FPT=Q,FPO='1',DATES=range(2016-09-03,2016-12-03),FILL=PREV**)) for (['XYZ US Equity'])

Since the FPO value requested does not specify the length of the offset step, than the value of the FPT parameter determines what the length of the offset step is (in addition to determining the fiscal period type). Also, since 'R' of 'F' is not specified for the FPO parameter, 'R' (rolling) is implied. Since FPT=Q the query above will result in an offset of 1 Q period from period 0 as of each date between September 3 and December 3, 2016. The result of the query is shown below:

REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
2016-08-31	2016-09-03	2016-09-30	29
2016-08-31	2016-09-04	2016-09-30	29
(omitted…)			
2016-08-31	2016-11-11	2016-09-30	29
2016-08-31	2016-11-12	2016-12-31	33
2016-08-31	2016-11-13	2016-12-31	33
(omitted…)			
2016-08-31	2016-11-19	2016-12-31	33
2016-11-20	2016-11-20	2016-12-31	34
2016-11-20	2016-11-21	2016-12-31	34

(omitted…)			
2016-11-20	2016-12-01	2016-12-31	34
2016-12-02	2016-12-02	2016-12-31	35
2016-12-02	2016-12-03	2016-12-31	35

Example query 8:
get(IS_EPS(**FPT=Q,FPO='1F',DATES=range(2016-09-03,2016-12-03),FILL=PREV**)) for (['XYZ US Equity'])

Since the FPO value requested does not specify the length of the offset step, than the value of the FPT parameter determines what the length of the offset step is (in addition to determining the fiscal period type). Since FPT=Q the query above will result in an offset of 1 Q period from fixed period 0 as of today, as of each date between September 3 and December 3, 2016. The result of the query is shown below:

REVISION_DATE	**AS_OF_DATE**	**PERIOD_END_DATE**	**IS_EPS**
2016-08-31	2016-09-03	2016-12-31	33
2016-08-31	2016-09-04	2016-12-31	33
(omitted…)			
2016-08-31	2016-11-11	2016-12-31	33
2016-08-31	2016-11-12	2016-12-31	33
2016-08-31	2016-11-13	2016-12-31	33
(omitted…)			
2016-08-31	2016-11-19	2016-12-31	33
2016-11-20	2016-11-20	2016-12-31	34
2016-11-20	2016-11-21	2016-12-31	34
(omitted…)			
2016-11-20	2016-12-01	2016-12-31	34
2016-12-02	2016-12-02	2016-12-31	35
2016-12-02	2016-12-03	2016-12-31	35

Example query 9:
get(IS_EPS(**FPR='2017Q1',DATES=range(2016-08-03,2016-12-03),FILL=PREV**)) for (['XYZ US Equity'])
get(IS_EPS(**FPR='2017-03-31',DATES=range(2016-08-03,2016-12-03),FILL=PREV**)) for (['XYZ US Equity'])

Since no value is specified in the query for the FPO and FPT parameters, the default values of FPO='0' and FPT=LTM are implied and the query above will return an LTM period with a 0 offset from 2017 Q1 as of each date between August 3 and December 3, 2016. The result of the query is shown below:

REVISION_DATE	**AS_OF_DATE**	**PERIOD_END_DATE**	**IS_EPS**
	2016-08-03	2017-03-31	
	2016-08-04	2017-03-31	
(omitted…)			
	2016-08-30	2017-03-31	

	2016-08-31	2017-03-31	116
	2016-09-01	2017-03-31	116
(omitted…)			
2016-08-31	2016-11-19	2017-03-31	116
2016-11-20	2016-11-20	2017-03-31	128
2016-11-20	2016-11-21	2017-03-31	128
(omitted…)			
2016-11-20	2016-12-01	2017-03-31	128
2016-12-02	2016-12-02	2017-03-31	130
2016-12-02	2016-12-03	2017-03-31	130

7. ***Associative columns***

For any BQL query of Bloomberg Company Financials data, a set of default associative columns is returned and, in addition, there are additional associative columns that can be displayed on demand. For example, the query below will return the results as shown on the table:

get(IS_EPS) for (['IBM US Equity'])

IDs	CURRENCY	REVISION_DATE	AS_OF_DATE	PERIOD_END_DATE	IS_EPS
IBM US Equity	USD	2016-10-25	2016-12-05	2016-09-30	12.3

The first row of the table above shows the labels of the associative columns, whereas the second row shows the value returned for each associative column.

In order to request a specific associative column, the following BQL query can be run:

get(IS_EPS(FPR='2015A').**PERIOD_END_DATE**) for (['IBM US Equity'])

IDs	PERIOD_END_DATE
IBM US Equity	2015-12-31

As shown above, the result of the query will be the period end date for the 2015 Annual fiscal period for IBM US Equity.

The list of the associative columns offered for Bloomberg fundamental data in BQL is shown below:

- VALUE
- CURRENCY
- AS_OF_DATE
- PERIOD
- REVISION_DATE
- PERIOD_END_DATE
- PERIOD_OFFSET
- ACT_EST_DATA
- VALUE_EXISTS
- IN_CONSENSUS_BST
- IN_CONSENSUS_28_DAYS
- IN_CONSENSUS_PE
- ANALYST_NAME
- ANALYST_ID
- FIRM_NAME
- ENTITLED
- ACT_EST_SOURCE
- EST_SOURCE_SPECIFIED

The rest of this section offers more details about the meaning and utility of each of these associate columns and the values returned for them.

7.1. Value

Description: Returns the value of each data item requested in the query without any additional associative columns, except for the ID

Mnemonic: VALUE

Displayed by default: Yes

Example query: get(IS_EPS(FPR='2015A').**VALUE**) for (['IBM US Equity'])

Example values:

IDs	VALUE
IBM US Equity	13.48

Associative column possible values:

Value	Description	Meaning
Value	Value	Value of the data item requested by the query
Null	No Value	Data is not available.

7.2. Currency

Description: Returns the currency of each data item[23] returned by the query

Mnemonic: CURRENCY

Displayed by default: Yes

Example query: get(IS_EPS(FPR='2015A').**CURRENCY**) for (['IBM US Equity'])

Example values:

IDs	CURRENCY
IBM US Equity	USD

Associative column possible values:

Value	Description	Meaning
[3-letter code]	ISO currency code	The currency the data is returned in

[23] Only for data items for which the units of measure is a currency

7.3. As of date

Description: Returns the observation date of the observer of each data item returned by the query

Mnemonic: AS_OF_DATE

Displayed by default: Yes

Example query: get(IS_EPS(FPR='2015A').**AS_OF_DATE**) for (['IBM US Equity'])

Example values:

IDs	AS_OF_DATE
IBM US Equity	2016-12-05

Associative column possible values:

Value	Description	Meaning
Date	As of date	The date the observer run the query for

7.4. Period

Description: Returns the fiscal year and period each data item returned by the query came from

Mnemonic: PERIOD

Displayed by default: No

Example query: get(IS_EPS(FPR='2015A').**PERIOD**) for (['IBM US Equity'])

Example values:

IDs	PERIOD
IBM US Equity	2015 A

Associative column possible values:

Value	Description	Meaning
[YYYY] [PERIOD]	Fiscal year and period	Fiscal year and period associated with the data point

7.5. Revision date

Description: Returns the date in which each data item return by the query was publicly available

Mnemonic: REVISION_DATE

Displayed by default: Yes

Example query:
get(IS_EPS(FPR='2015A').**REVISION_DATE**) for (['IBM US Equity'])

Example values:

IDs	REVISION_DATE
IBM US Equity	2016-12-05

Associative column possible values:

Value	Description	Meaning
Date	Revision date	The date when the value of the data item became public information
Null	No Value	Data is not available.

7.6. Period end date

Description: Returns the period end date of the fiscal period each data item returned by the query came from

Mnemonic: PERIOD_END_DATE

Displayed by default: Yes

Example query:
get(IS_EPS(FPR='2015A').**PERIOD_END_DATE**) for (['IBM US Equity'])

Example values:

IDs	PERIOD
IBM US Equity	2015-12-31

Associative column possible values:

Value	Description	Meaning
Date	Period end date	The end date of the fiscal period

7.7. Period offset

Description: Returns the offset from fiscal period 0 of each data item returned by the query

Mnemonic: PERIOD_OFFSET

Displayed by default: No

Example query: get(IS_EPS(FPR='2015Q4').**PERIOD_OFFSET)** for (['IBM US Equity'])

Example values:

IDs	PERIOD_OFFSET
IBM US Equity	-7 Q

Associative column possible values:

Value	Description	Meaning
[Integer] + Q	Number of quarters	Number of quarters offset from period 0, with a positive number indicating periods not reported yet by the company and a negative number indicating periods already reported by the company.
[Integer] + S	Number of semiannual	Number of semiannuals offset from period 0, with a positive number indicating periods not reported yet by the company and a negative number indicating periods already reported by the company.
[Integer] + A	Number of annuals	Number of annuals offset from period 0, with a positive number indicating periods not reported yet by the company and a negative number indicating periods already reported by the company.

7.8. Actual or Estimated data

Description: Indicates whether each data item returned by the query is data actually reported by a company, or estimated

Mnemonic: ACT_EST_DATA

Displayed by default: No

Example query:
get(IS_EPS(FPR='2015A').**ACT_EST_DATA)** for (['IBM US Equity'])

Example values:

IDs	ACT_EST_DATA
IBM US Equity	A

Associative column possible values:

Value	Description	Meaning
A	Actual	The data item is an actual value reported by the company
E	Estimate	The data item is an estimated value
Null	No Value	Data is not available.

7.9. Value Exists

Description: Indicates whether an estimate data point exists for the specified sell-side research contributor.

Mnemonic: VALUE_EXISTS

Displayed by default: No, only displays when a broker code is specified in the EST_SOURCE parameter.

Example query:

get(IS_EPS(FPR='2020A', AE=E, EST_SOURCE=JPM).**VALUE_EXISTS)** for (['IBM US Equity'])

Example values:

IDs	**VALUE_EXISTS**
IBM US Equity	Y

Associative column possible values:

Value	**Description**	**Meaning**
Y	Value Exists	An estimate data point exists for the specified sell-side research contributor.
N	No Value Exists	An estimate data point does not exist for the specified sell-side research contributor.

7.10. In Bloomberg Standard Consensus

Description: Indicates whether an estimate data from the specified sell-side research contributor is included in the Bloomberg Standard Consensus (BST).

Mnemonic: IN_CONSENSUS_BST

Displayed by default: No, only displays when a broker code is specified in the EST_SOURCE parameter.

Example query:

get(IS_EPS(FPR='2020A', AE=E, EST_SOURCE=JPM).**IN_CONSENSUS_BST)**
for (['IBM US Equity'])

Example values:

IDs	IN_CONSENSUS_BST
IBM US Equity	Y

Associative column possible values:

Value	Description	Meaning
Y	Value Included	The estimate data from the specified sell-side research contributor is included in the Bloomberg Standard Consensus.
N	Value Excluded	The estimate data from the specified sell-side research contributor is excluded from the Bloomberg Standard Consensus.
Null	No Value	Estimates data is not available.

7.11. In Bloomberg 28-Days Consensus

Description: Indicates whether an estimate data point for the specified sell-side research contributor is included in the Bloomberg 28-Days Consensus (BLI).

Mnemonic: IN_CONSENSUS_28_DAYS

Displayed by default: No, only displays when a broker code is specified in the EST_SOURCE parameter.

Example query:

get(IS_EPS(FPR='2020A', AE=E, EST_SOURCE=JPM).**IN_CONSENSUS_28_DAYS)**
for (['IBM US Equity'])

Example values:

IDs	**IN_CONSENSUS_28_DAYS**
IBM US Equity	Y

Associative column possible values:

Value	**Description**	**Meaning**
Y	Value Included	The estimate data from the specified sell-side research contributor is included in the Bloomberg 28-Days Consensus.
N	Value Excluded	The estimate data from the specified sell-side research contributor is excluded from the Bloomberg 28-Days Consensus.
Null	No Value	Estimates data is not available.

7.12. In Bloomberg Post-Event Consensus

Description: Indicates whether an estimate data point for the specified sell-side research contributor is included in the Bloomberg Post-Event Consensus (BPE).

Mnemonic: IN_CONSENSUS_PE

Displayed by default: No, only displays when a broker code is specified in the EST_SOURCE parameter.

Example query:

get(IS_EPS(FPR='2020A', AE=E, EST_SOURCE=JPM).**IN_CONSENSUS_PE)**
for (['IBM US Equity'])

Example values:

IDs	IN_CONSENSUS_PE
IBM US Equity	Y

Associative column possible values:

Value	Description	Meaning
Y	Value Included	The estimate data from the specified sell-side research contributor is included in the Bloomberg Post-Event Consensus.
N	Value Excluded	The estimate data from the specified sell-side research contributor is excluded from the Bloomberg Post-Event Consensus.
Null	No Value	Estimates data is not available.

7.13. Analyst Name

Description:	Name of the analyst with the sell-side research contributor providing the estimates data.
Mnemonic:	ANALYST_NAME
Displayed by default:	No, only displays when a broker code is specified in the EST_SOURCE parameter.

Example query:

get(IS_EPS(FPR='2020A', AE=E, EST_SOURCE=JPM).**ANALYST_NAME)**
for (['IBM US Equity'])

Example values:

IDs	ANALYST_NAME
IBM US Equity	John Doe

Associative column possible values:

Value	Description	Meaning
[Text String]	Analyst First and Last Name	Name of the analyst with the sell-side research contributor providing the estimates data.
Null	No Value	Estimates data is not available.

7.14. Analyst ID

Description: ID of the analyst with the sell-side research contributor providing the estimates data.

Mnemonic: ANALYST_ID

Displayed by default: No, only displays when a broker code is specified in the EST_SOURCE parameter.

Example query:

get(IS_EPS(FPR='2020A', AE=E, EST_SOURCE=JPM).**ANALYST_ID)**
for (['IBM US Equity'])

Example values:

IDs	ANALYST_ID
IBM US Equity	12345

Associative column possible values:

Value	Description	Meaning
[Identifier]	Analyst ID	ID of the analyst with the sell-side research contributor providing the estimates data.
Null	No Value	Estimates data is not available.

7.15. Firm Name

Description: Name of the sell-side research firm contributing the estimates data.

Mnemonic: FIRM_NAME

Displayed by default: No, only displays when a broker code is specified in the EST_SOURCE parameter.

Example query:

get(IS_EPS(FPR='2020A', AE=E, EST_SOURCE=JPM).**FIRM_NAME)**
for (['IBM US Equity'])

Example values:

IDs	FIRM_NAME
IBM US Equity	JP Morgan

Associative column possible values:

Value	Description	Meaning
[Text String]	Firm Name	Name of the sell-side research firm contributing the estimates data.

7.16. Entitled

Description: Indicates whether the user is entitled for the estimates data provided by the sell-side research contributor specified.

Mnemonic: ENTITLED

Displayed by default: No, only displays when a broker code is specified in the EST_SOURCE parameter.

Example query:

get(IS_EPS(FPR='2020A', AE=E, EST_SOURCE=JPM).**ENTITLED)**
for (['IBM US Equity'])

Example values:

IDs	ENTITLED
IBM US Equity	N

Associative column possible values:

Value	Description	Meaning
Y	Entitled	Indicates the user is entitled for the estimates data provided by the sell-side research contributor specified.
N	Not Entitled	Indicates the user is not entitled for the estimates data provided by the sell-side research contributor specified.
Null	No Value	Estimates data is not available.

7.17. Actuals or Estimates Source

Description:	Indicates the source of the data returned.
Mnemonic:	ACT_EST_SOURCE
Displayed by default:	No, only displays when a broker code is specified in the EST_SOURCE parameter.

Example query:

get(IS_EPS(FPR='2019A', **AE=E**, EST_SOURCE=JPM).**ACT_EST_SOURCE)**
for (['IBM US Equity'])

IDs	ACT_EST_SOURCE
IBM US Equity	JPM

get(IS_EPS(FPR='2019A', **AE=A**).**ACT_EST_SOURCE)** for (['IBM US Equity'])

IDs	ACT_EST_SOURCE
IBM US Equity	CMPY

Associative column possible values:

Value	Description	Meaning
CMPY	Company	Source of the data point is the company filings.
[BBB]	Broker Code	Source of the data point is a sell-side research contributor.

7.18. Estimate Source Specified

Description: Indicates the estimate source requested in the query. This allows to distinguish the time series of data returned for different estimates sources which are using the same Actuals data for the reported periods (all Actuals data is sourced from the company filings).

Mnemonic: EST_SOURCE_SPECIFIED

Displayed by default: No, only displays when a broker code is specified in the EST_SOURCE parameter.

Example query:

```
get(IS_EPS(FPR='2020A', AE=E,EST_SOURCE=JPM).EST_SOURCE_SPECIFIED)
for (['IBM US Equity'])
```

IDs	EST_SOURCE_SPECIFIED
IBM US Equity	JPM

This is useful for queries requesting data from multiple sources for both reported and future periods. For example, the following query:

```
get(IS_EPS(FPO=RANGE(-1,1),FPT=A, EST_SOURCE=['JPM','BAM'])
for (['IBM US Equity'])
```

will return the following data[24]:

IDs	PERIOD_OFFSET	ACT_EST_DATA	ACT_EST_SOURCE	**EST_SOURCE_SPECIFIED**	Value
IBM US Equity	-1A	A	CMPY	**JPM**	V1
IBM US Equity	0	A	CMPY	**JPM**	V2
IBM US Equity	1A	E	JPM	**JPM**	V3
IBM US Equity	-1A	A	CMPY	**BAM**	V1
IBM US Equity	0	A	CMPY	**BAM**	V2
IBM US Equity	1A	E	BAM	**BAM**	V4

[24] The table is only showing select associative columns for illustration pruposes. For a list of all the associative columns support, see the Associative columns section.

8. ***BQL vs Legacy Company Financials parameters***

The table below provides examples of common queries in Legacy Company Financials functions BDP() and BDH() and how the same queries can be run via BQL Company Financials

Description of query	**Legacy Company Financials**	**BQL Company Financials**
Get total assets data for IBM for the latest quarter	=BDP("IBM US Equity", "BS_TOT_ASSET","FUND_PER=Q")	=BQL("IBM US Equity","BS_TOT_ASSET","FA_PERIOD_TYPE=Q")
Get EPS data for Vodafone for the latest semiannual period	=BDP("VOD LN Equity", "IS_EPS","FUND_PER=S")	=BQL("VOD LN Equity","IS_EPS","FA_PERIOD_TYPE=S")
Get the latest LTM EPS data for Vodafone	=BDP("IBM US Equity","TRAIL_12M_EPS_AFTER_XO","FUND_PER=S")	=BQL("VOD LN Equity","IS_EPS","FA_PERIOD_TYPE=LTM")
Get the estimated EPS GAAP data for the current fiscal year for Siemens	=BDP("SIE GR Equity","BEST_EPS_GAAP","BEST_FPERIOD_OVERRIDE=1FY")	=BQL("SIE GR Equity","IS_EPS","FA_PERIOD_TYPE=A","FA_PERIOD_OFFSET='1A'")
Get the estimated EPS Adj data for next fiscal year for Siemens	=BDP("SIE GR Equity","BEST_EPS","BEST_FPERIOD_OVERRIDE=2FY")	=BQL("SIE GR Equity","IS_EPS","FA_PERIOD_TYPE=A","FA_PERIOD_OFFSET='2A'","FA_ADJUSTED=Y")
Get the estimated EPS Adj data for the next 12 months for Siemens	=BDP("SIE GR Equity","BEST_EPS","BEST_FPERIOD_OVERRIDE=1TY")	=BQL("SIE GR Equity","IS_EPS","FA_PERIOD_TYPE=LTM","FA_PERIOD_OFFSET='1A'","FA_ADJUSTED=Y")
Get the estimated EPS Adj data for the next 24 months for Siemens	Not supported	=BQL("SIE GR Equity","IS_EPS","FA_PERIOD_TYPE=LTM","FA_PERIOD_OFFSET='2A'","FA_ADJUSTED=Y")
Get the EPS GAAP guidance data for IBM for fiscal year 2017	=BDP("IBM US Equity","CEST_EPS_GAAP","BEST_FPERIOD_OVERRIDE=17Y"	=BQL("IBM US Equity","IS_EPS","FA_PERIOD_TYPE=A","FA_PERIOD_REFERENCE='2017'","EST_SOURCE=CGD")
Get annual EPS Adj actual data for the last 3 years and estimates data for the next 2 years	Not supported as a single query. The following steps are needed: 1. For the actuals part, run the following query: =BDH("IBM US EQUITY","IS_DIL_EPS_CONT_OPS","-2FY","-0FY","FUND_PER=Y") 2. For the estimates part, the =BDH() function can't be used to obtain fiscal period data for future fiscal period, but each individual data point can be obtained through the =BDP() function: Data Point 1: =BDP("IBM US Equity","BEST_EPS","BEST_FPERIOD_OVERRIDE=1FY") Data point 2: =BDP("IBM US Equity","BEST_EPS","BEST_FPERIOD_OVERRIDE=2FY") 3. Populate the appropriate cells in Excel with the formulas from step 1 and step 2 to create a single data stream	=BQL("IBM US Equity","IS_EPS","FA_PERIOD_TYPE=A","FA_PERIOD_OFFSET=RANGE(-'2','2')","FA_ADJUSTED=Y")
Get annual EPS Adj estimates data from for	Not supported as a single query. Five separate queries are needed for each of the data points	=BQL("IBM US Equity","IS_EPS","FA_PERIOD_TYPE=A","FA_P

the last 3 and for the next 2 years for IBM	requested: Data point 1: =BDP("IBM US EQUITY","BEST_EPS","BEST_FPERIOD_OVERRIDE=-2FY") … Data point 5: =BDP("IBM US EQUITY","BEST_EPS","BEST_FPERIOD_OVERRIDE=2FY")	ERIOD_OFFSET=RANGE(-'2','2')","FA_ADJUSTED=Y","FA_ACT_EST_DATA=E")
Get current year's EPS Adj estimates data as of every day from December 1, 2016 to January 31, 2017 for IBM	=BDH("IBM US EQUITY","BEST_EPS","12/01/2016","01/31/2017","BEST_FPERIOD_OVERRIDE=1GY","DAYS=ALL","FILL=P") *Note: The "current year" will roll over on 12/31/2016 which is the period end date of IBM's fiscal year*	=BQL("IBM US EQUITY","IS_EPS","FA_ADJUSTED=Y","FA_PERIOD_TYPE=A","FA_PERIOD_OFFSET='1A'","DATES=RANGE(2016-12-01,2017-01-31)","FILL=PREV") *Note: The "current year" will roll over on 01/19/2017 when IBM reported the earnings for the fiscal year ending on 12/31/2016*
Get the 1 year blended forward EPS Adj estimates as of every day from December 1, 2016 to January 31, 2017 for IBM	=BDH("IBM US EQUITY","BEST_EPS","12/01/2016","01/31/2017","BEST_FPERIOD_OVERRIDE=1BF","DAYS=ALL","FILL=P")	=BQL("IBM US EQUITY","IS_EPS","FA_ADJUSTED=Y","FA_PERIOD_TYPE=BT","FA_PERIOD_OFFSET='1'","DATES=RANGE(2016-12-01,2017-01-31)","FILL=PREV")
Get the 1 year blended trailing EPS Adj actuals data as of every day from December 1, 2016 to January 31, 2017 for IBM	Not supported	=BQL("IBM US EQUITY","IS_EPS","FA_ADJUSTED=Y","FA_PERIOD_TYPE=BT","DATES=RANGE(2016-12-01,2017-01-31)","FILL=PREV")
Get the revisions history for the annual EPS Adj estimates data for the current year as of every day from December 1, 2016 to January 31, 2017 for IBM	=BDH("IBM US EQUITY","BEST_EPS","12/01/2016","01/31/2017","BEST_FPERIOD_OVERRIDE=1FY","DAYS=ALL","FILL=P")	=BQL("IBM US EQUITY","IS_EPS","FA_ADJUSTED=Y","FA_PERIOD_TYPE=A","FA_PERIOD_OFFSET='1FA'","DATES=RANGE(2016-12-01,2017-01-31)","FILL=PREV")
Get the EPS Adj actuals LTM data as of every day from December 1, 2016 to January 31, 2017 for IBM	Not supported as a single query. A separate query is needed for every day: Data point 1: =BDP("IBM US EQUITY","T12M_DIL_EPS_CONT_OPS","FUNDAMENTAL_PUBLIC_DATE=20161201") … Data point 62: =BDP("IBM US EQUITY","T12M_DIL_EPS_CONT_OPS","FUNDAMENTAL_PUBLIC_DATE=20170131")	=BQL("IBM US EQUITY","IS_EPS","FA_ADJUSTED=Y","DATES=RANGE(2016-12-01,2017-01-31)","FILL=PREV")