



BMLL Data Feed

Level 3 - Futures

Date: 2025-11-18

Version: 2.7

1.1. Introduction

The Level 3 dataset offers an order by order view of the order book. Showing every single update to the order book, it provides an in-depth understanding of historical market activity, enabling users to make more informed trading decisions.

1.2. Dataset schema

Basic Types

Type	Content	Text representation
BOOLEAN	A True or False boolean value.	1,0
INT32	A 32 bit signed integer.	12345
INT64	A 64 bit signed integer.	12345
DOUBLE	IEEE 64-bit floating point value.	123.4567
BYTE_ARRAY	An arbitrarily long byte array.	"VODAFONE GROUP"
NUMBER	A number with defined precision (total digits allowed) and scale (number of digits allowed to the right of the decimal point). Unless otherwise specified, the default is precision 38 and scale 0.	12345
VARCHAR	Variable length string up to 500 characters. May contain quotes and special characters. All text will be utf8-encoded unless indicated differently. Ticker and ISO codes should be ASCII compatible.	"VODAFONE GROUP"
TIMESTAMP_NTZ	Nanoseconds since 1970 in UTC. Timestamp will be represented in ISO format in text format.	20220302-11:23.57.12345789
FLOAT	IEEE 64-bit floating point value.	123.4567
DATE	A calendar date consisting of a year, month, and day, without including any time or time zone components.	2025-01-01

1.3. Schema

One row in the dataset corresponds to a single update of the limit order book during the trading day. An update consists generally of the addition of an order to the limit order book, the removal of an order book of an update on an existing order. A single trading order can cause multiple updates on the limit order book - for instance, a single aggressive sweep order may clear multiple orders that were passively resting on the book.

Whenever we receive that information, we will group updates of the order book related to the same trading order to a unique event number - and this is designed to identify the state of the order book that is actually being interacted with.

Unless preceded by "Old", columns will always represent the value after the event.

Field Name	Type	Description
MIC	BYTE_ARRAY	The Market Identifier Code (MIC) is used to uniquely identify the exchange for which the L1-book is reconstructed.
Ticker	BYTE_ARRAY	Ticker as provided by the reporting venue.
ListingId	INT64	The BMLL identifier represents the listing
InstrumentId	INT64	If not 0, the BMLL identifier represents the instrument to which the listing belongs. BMLL Instrument IDs are different if the primary identifier is different, the currency or the region is different.
ProductCode	BYTE_ARRAY	Product Code as supplied by the exchange.
ContractType	BYTE_ARRAY	Normalised Type of the Listing. Mappings are provided per dataset. Can take values: <ul style="list-style-type: none"> • • Outright • Spread • Inter Exchange • Inter Product Spread • Pack and Bundle • Strip • Butterfly • Condor • Other • Unknown

BMLL Data Feed: L3 - Futures

Field Name	Type	Description
MaturityMonthYear	BYTE_ARRAY	The month and year on which the Listing matures, in format YYYYMM.
TradeDate	INT32	The specific date on which the event occurred - the date is derived from the LocalTimestamp.
TimestampNanoseconds	INT64	The time of the updates as reported by the venue, represented as the number of nanoseconds since 1970-01-01 00:00:00.0 UTC.
ReceiveTimestampNanoseconds	INT64	The time of message receipt by the capture device, represented as the number of nanoseconds since 1970-01-01 00:00:00.0 UTC.
TZOffset	INT32	The offset between the local timestamp and UTC in seconds. Note: This is based on the original listing location of the primary instrument. Format was +/- HHMM . This will be adjusted during the DST if such a change occurs.
EventNo	INT64	An increasing ID per symbol, used to identify atomic updates to the orderbook. Multiple rows can share one EventNo - these should be treated as one update when rebuilding an order book.
Side	INT64	Side of the order update 1: Order is updated on the BID side 2: Order is updated on the ASK side
LobAction	INT64	Type of activity on the order book: INSERT: A new order added to the book. REMOVE: An existing order on the book is removed, due to either cancellation or order execution. UPDATE: An existing order on the book is modified. Rows with an unknown value of LOB action should be skipped for the purpose of building the order book
OrderType	INT64	The type of order 0: Unknown 1: Limit 2: Market
Price	DOUBLE	The price of the order after the update. Null if LobAction is REMOVE.

Field Name	Type	Description
Size	INT64	The size of the order after the update. 0 if LobAction is REMOVE.
IsImplied	BOOLEAN	Indicates if the order is a real order on the orderbook, or an implied price level (with a synthetic order id).
OrderId	BYTE_ARRAY	The order ID after the update. Empty if LobAction is REMOVE.
OldPrice	DOUBLE	The price of the order before the update. Null if LobAction is INSERT.
OldSize	INT64	The size of the order before the update. 0 if LobAction is INSERT.
OldOrderId	BYTE_ARRAY	The order ID before the update. Empty if order is inserted
OriginalOrderId	BYTE_ARRAY	The original ID of the order inserted into the order book. Note that in some exchanges, order Ids can change (e.g. due to a priority shift). This ID allows you to track modified and updated orders throughout the day
OrderExecuted	BOOLEAN	Indicates if the order has been removed due to an execution (rather than a cancellation)
ExecutionPrice	DOUBLE	The price that the order executed at. Note this can be different to the price the order was placed at (e.g. during auctions)
ExecutionSize	INT64	The size that the order executed at. Note this can be more than OldSize (e.g. with an Iceberg order)
TradeId	BYTE_ARRAY	The ID of the trade, if provided by the exchange
PriceLevel	INT64	The price order of the level after the instruction (1 = best price) where applicable. For a REMOVE event, equals the OldPriceLevel.
PosChange	BOOLEAN	Indicates whether the order has changed priority with the update
OldPriceLevel	INT64	The level of the order before the instruction (1 = best price) where applicable. For a LobAction of INSERT, this is the price level of the inserted order if the level previously existed, or 0 otherwise. For LobAction of UNKNOWN this is 0.

Field Name	Type	Description
SizeAhead	INT64	For LobAction = INSERT or UPDATE: The total size (quantity) of the orders ahead of the given order in the priority queue, after the event. For LobAction = REMOVE: This is -1.
OrdersAhead	INT64	For LobAction = INSERT or UPDATE: The number of the orders ahead of the given order in the priority queue, after the event. For LobAction = REMOVE: This is -1.
EndOfEvent	BOOLEAN	Indicates whether this is an end of event
MarketState	BYTE_ARRAY	The market state of the update
Printable	BOOLEAN	Indicates whether the execution is printable. Printable is used to stop double counting (e.g. auction executions)
DelOrderIndex	INT64	Used in the order book rebuild. Provides the position of the order in all priority before the order book update -1 for an INSERT
AddOrderIndex	INT64	Used in the order book rebuild. Provides the position of the order in all priority after the order book update. -1 for a REMOVE
MPIDAttribution	BYTE_ARRAY	The market participant id (if provided by the exchange)
OriginalExchangeMessage	BYTE_ARRAY	The message type as provided by the exchange.
ExchangeSequenceNo	INT64	The sequence number of the update, if provided by the exchange.
BMLLSequenceSource	INT64	Source of the data, where multiple order book sources are provided together. Example would be out of hours orderbook and main session (e.g. Borsa Italiana)
BMLLSequenceNo	INT64	A synthetic sequence number as provided by BMLL. This ensures the correct ordering of all messages from a particular source and should be used as the standard to sort and join all messages.

1.4. Market State Normalisation

BMLL provides a consistent normalisation for market state messages, in order to make cross venue analysis quick and easy.

The following normalisation states are available

Market State

Value	Description
AUCTION_ON_DEMAND	Auctions which take place alongside CONTINUOUS_TRADING. Includes order entry and uncrossing periods.
CLOSED	The market is closed so no trading activity can take place.
CLOSING_AUCTION	Includes order entry and uncrossing periods.
CONDITIONAL	An uncrossing state for specific market mechanisms e.g. Turquoise Plato Uncross.
CONTINUOUS_TRADING	Main continuous trading session in which orders can be placed and matched. NOTE: Separate continuous trading occurring before the primary open or after close will be generally indicated with PRE_TRADE and POST_TRADE.
CONTINUOUS_TRADING_PRIMARY_CLOSED	A continuous trading session in which orders can be placed and matched on secondary markets, but the primary market is CLOSED. Used for secondary markets only (e.g. a multilateral trading facility in the EU, or alternative trading system in the USA).
HALTED	Unscheduled trading halts, for example, when a circuit breaker is triggered.
INTRADAY_AUCTION	Scheduled auctions which interrupt CONTINUOUS_TRADING. Includes order entry and uncrossing periods.
NOT_APPLICABLE	Used for venues such as trade reporting facilities where the concept of market phase does not apply, as there are no specific market hours.
OPENING_AUCTION	Includes order entry and uncrossing periods.

Value	Description
POST_TRADE	Market phase following CONTINUOUS_TRADING phase, and generally occurring after the primary CLOSING_AUCTION. This includes trade-at-last phases; such trades can be distinguished using the CLOSING_PRICE bml_trade_type.
PRE_OPEN	Market phase preceding CONTINUOUS_TRADING, and generally prior to OPENING_AUCTION when applicable.
UNKNOWN	Used when the market phase could not be resolved into one of the known states listed here.
UNSCHEDULED_AUCTION	Unscheduled auctions which interrupt CONTINUOUS_TRADING. Includes order entry and uncrossing periods.

1.5. File format and delivery structure

Data is delivered as a single file per market per date, as a parquet file.

- **{YYYY}**: Year
- **{MM}**: Month
- **{DD}**: Date
- **{Product}**: The product you are subscribing
 - **Level 3**: L3
- **{Region}**: The region of the venue, such as APAC, AMERICAS, EMEA and GLOBAL
- **{MIC}**: Market Identifier Code (MIC) is a unique four-character code that identifies stock markets and exchanges for trading and referencing computer systems
- **{Channel}**: For Nymex (XNYM) and CME main market (XCME) we will further subdivide the data by multicast channel; XCEC and XCBOT will be delivered in single files.

File Structure:

{YYYY}/{MM}/{DD}/L3/{Region}/futures-L3-{MIC}-{Channel}-YYYYMMDD.parquet

{YYYY}/{MM}/{DD}/L3/{Region}/futures-L3-{MIC}-YYYYMMDD.parquet

1.6. Channels CME and XNYM

MIC	Channel Number	Channel Name
XCME	310	CME Globex Equity Futures - E-mini S&P 500 futures
XCME	312	CME Globex Interest Rate Futures
XCME	314	CME Globex FX Futures
XCME	316	CME Globex Commodity Futures
XCME	318	CME Globex Equity Futures - excludes E-mini S&P 500
XCME	326	CME Crypto Futures
XNYM	380	NYMEX Globex Emissions Futures
XNYM	382	NYMEX Globex Crude & Crude Refined Futures
XNYM	384	NYMEX Globex Metals, Softs, & Alternative Market Futures
XNYM	386	NYMEX Globex Nat Gas & other Non-Crude Energy Futures

1.7. Data arrival times and coverage

Full product coverage and arrival times are available at <https://data.bmlitech.com>.