BMLL Data Feed

Data Feed: Level 3 - Prediction

Date: 2026-09-02
Version: 1.0

1.1. Introduction

The BMLL Data Feed: Prediction Level 3 dataset provides an order-by-order view of the limit order book for CME prediction market (event contract) instruments. Every insert, update, cancel, and execute is captured, giving an in-depth view of historical market activity.

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

Copyright © 2026 BMLL Technologies. All rights reserved.

1.3. Schema

Field Name (Snowflake / Databricks)	Field Name (Parquet)	Type (Snowflake / Databricks)	Type (Parquet)	Description
MIC	MIC	VARCHAR	BYTE_ARRAY	The Market Identifier Code (MIC) is used to uniquely identify the exchange which is reporting market activity.
TICKER	Ticker	VARCHAR	BYTE_ARRAY	BMLL-format ticker for a listing on its trading venue. This is the customer-facing Ticker used in API, Snowflake, and Databricks materialisations.
UNDERLYING_TICKER	UnderlyingTicker	VARCHAR	BYTE_ARRAY	Ticker of the underlying symbol
UNDERLYING_PRODUCT_CODE	UnderlyingProductCode	VARCHAR	BYTE_ARRAY	Underlying product code.
STRIKE_PRICE	StrikePrice	FLOAT	DOUBLE	Strike Price of the Option
OPTION_TYPE	OptionType	VARCHAR	BYTE_ARRAY	Indicate whether the option is a put or a call.
EXPIRY_DATE	ExpiryDate	DATE	INT32	Last trading date of the future

Field Name (Snowflake / Databricks)	Field Name (Parquet)	Type (Snowflake / Databricks)	Type (Parquet)	Description
CURRENCY_CODE	CurrencyCode	VARCHAR	BYTE_ARRAY	The currency code used for all price information (unless specified differently in specific column descriptions). The Currency code is represented following ISO-4217 codes (see https://en.wikipedia.org/wiki/ISO_4217) . Minor currencies are represented using the major currency code and putting in lower case the last letter of the major currency code.
TRADE_DATE	TradeDate	DATE	INT32	Date associated with the dataset row. Product-level smell records document any legacy product where this date is derived from publication time rather than the venue trade timestamp.
TIMESTAMP_NANOSECONDS	TimestampNanoseconds	NUMBER	INT64	This column contains a representation of the time of the update as the number of nanoseconds since 1970-01-01 00:00:00.0 UTC.
RECEIVE_TIMESTAMP_NANOSECONDS	ReceiveTimestampNanoseconds	NUMBER	INT64	BMLL infrastructure receive timestamp as nanoseconds since 1970-01-01 00:00:00 UTC.

Field Name (Snowflake / Databricks)	Field Name (Parquet)	Type (Snowflake / Databricks)	Type (Parquet)	Description
TZ_OFFSET	TZOffset	NUMBER	INT64	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.
EVENT_NO	EventNo	NUMBER	INT64	Synthetic event id allowing to group actions that occur atomically on the limit order book and during which no other order may have interacted with the book
SIDE	Side	NUMBER	BYTE_ARRAY	Side on which the order is resting
LOB_ACTION	LobAction	NUMBER	BYTE_ARRAY	The type of the instruction as relevant to rebuilding the limit order book state:
ORDER_TYPE	OrderType	VARCHAR	BYTE_ARRAY	Order type: 0 = Unknown; 1 = Limit; 2 = Market.
PRICE	Price	FLOAT	DOUBLE	price at which the order occurred after the event
SIZE	Size	FLOAT	DOUBLE	size of the order on the book after the event
IS_IMPLIED	IsImplied	VARCHAR	BYTE_ARRAY	True if this is a synthetic implied order rather than a real resting order.

Field Name (Snowflake / Databricks)	Field Name (Parquet)	Type (Snowflake / Databricks)	Type (Parquet)	Description
ORDER_ID	OrderId	VARCHAR	BYTE_ARRAY	Order ID currently assigned to the order to which the instruction refers, as provided by the exchange. This is always the ID after this instruction is applied. For a lob_action of remove (or unknown), this is an empty string, ''.
OLD_PRICE	OldPrice	FLOAT	DOUBLE	Price of the order the instruction refers to before it is applied. For a lob_action of remove, update or skip, this is the old price. For insert (or unknown) this is Nan.
OLD_SIZE	OldSize	NUMBER	INT64	Size (quantity) of the order the instruction refers to before it is applied. For a lob_action of remove, update or skip, this is the old size. For insert (or unknown) this is 0.
OLD_ORDER_ID	OldOrderId	VARCHAR	BYTE_ARRAY	Order ID the instruction refers to as provided by the exchange before this instruction is applied. For a lob_action of insert (or unknown), this is an empty string, ''.

Field Name (Snowflake / Databricks)	Field Name (Parquet)	Type (Snowflake / Databricks)	Type (Parquet)	Description
ORIGINAL_ORDER_ID	OriginalOrderId	VARCHAR	BYTE_ARRAY	Order ID originally assigned to the order to which instruction refers. In some cases order_id can change in the order's lifetime; this is always the first ID assigned to the order.
ORDER_EXECUTED	OrderExecuted	BOOLEAN	BOOLEAN	True when the instruction was an execution.
EXECUTION_PRICE	ExecutionPrice	FLOAT	DOUBLE	Execution price when the instruction is an execution. Equals the price except for market orders, and in special circumstances (see the relevant dataset page). When order_executed is False, this is NaN.
EXECUTION_SIZE	ExecutionSize	NUMBER	INT64	Execution size when the instruction is an execution. Equals old_size - size when order_executed is True, and 0 otherwise.
TRADE_ID	TradeId	VARCHAR	BYTE_ARRAY	When available, the Trade ID as provided by the exchange. The ID associated with an execution can be used to link to the Trades Table described below. Each value corresponds to a unique row in the trades table, if provided.

Field Name (Snowflake / Databricks)	Field Name (Parquet)	Type (Snowflake / Databricks)	Type (Parquet)	Description
PRICE_LEVEL	PriceLevel	NUMBER	INT64	The level of the order after the instruction (1 = best price) where applicable. For a lob_action of remove, this is the price level of the removed order if the level still exists, or 0 otherwise. For lob_action of unknown this is 0.
POS_CHANGE	PosChange	VARCHAR	BYTE_ARRAY	True if the order's queue priority position changed as a result of this update.
OLD_PRICE_LEVEL	OldPriceLevel	NUMBER	INT64	The level of the order before the instruction (1 = best price) where applicable. For a lob_action of insert, this is the price level of the inserted order if the level previously existed, or 0 otherwise. For lob_action of unknown this is 0.
SIZE_AHEAD	SizeAhead	NUMBER	INT64	For lob_action = insert or update, the total size (quantity) of the orders ahead of the given order in the priority queue, after the event. For lob_action = remove, this is -1.
ORDERS_AHEAD	OrdersAhead	NUMBER	INT64	Number of orders ahead

Field Name (Snowflake / Databricks)	Field Name (Parquet)	Type (Snowflake / Databricks)	Type (Parquet)	Description
END_OF_EVENT	EndOfEvent	BOOLEAN	BOOLEAN	Many rows (instructions, i.e. changes to orders) may occur for one event (i.e. transition from one valid order book state to another). This flag is true if this is the last instruction in the same event (see above).
MARKET_STATE	MarketState	VARCHAR	BYTE_ARRAY	Standardised market state information. See below for description of possible market states.
PRINTABLE	Printable	BOOLEAN	BOOLEAN	Printable value as computed by BMLL parser allowing to filter out duplicate (non-printable) trade reports. These can be synthetic trade report, or trade reports describing a same execution from the view of the counterparty.
DEL_ORDER_INDEX	DelOrderIndex	NUMBER	INT64	Zero-based index of the order to remove from the book array when reconstructing as an ordered list. -1 if not applicable.
ADD_ORDER_INDEX	AddOrderIndex	NUMBER	INT64	Zero-based index at which to insert the updated order into the book array. -1 if not applicable.

Field Name (Snowflake / Databricks)	Field Name (Parquet)	Type (Snowflake / Databricks)	Type (Parquet)	Description
MPID_ATTRIBUTION	MPIDAttribution	VARCHAR	BYTE_ARRAY	Market Participant ID (MPID) of the firm attributed to this order or trade event, as reported by the exchange.
ORIGINAL_EXCHANGE_MESSAGE	OriginalExchangeMessage	VARCHAR	BYTE_ARRAY	Raw exchange message string, provided for diagnostic use.
EXCHANGE_SEQUENCE_NO	ExchangeSequenceNo	NUMBER	INT64	The sequence number as provided by the exchange. Note that some exchange may not provide a sequence number in this case the ExchangeSequenceNo will always be 0.
BMLL_SEQUENCE_SOURCE	BMLLSequenceSource	NUMBER	INT64	An integer used to identify the upstream feed from which the sequence numbers are generated. This is designed to avoid the creation of non-existent order between events that come from different message streams

Field Name (Snowflake / Databricks)	Field Name (Parquet)	Type (Snowflake / Databricks)	Type (Parquet)	Description
BMLL_SEQUENCE_NO	BMLLSequenceNo	NUMBER	INT64	The sequence no of the trade in the upstream feed as inferred by BMLL - this sequence number will respect the order of messages in upstream protocol and will be populated even if the upstream venue does not provide a sequence number - where the source of the feed is indicated by BMLLSequenceSource.