BMLL Data Feed

Trades - Prediction

Date: 2026-04-28
Version: 1.0

1.1. Introduction

The BMLL Data Feed: Prediction Trades dataset provides details of every reported trade execution for predictions, including strategy (multi-leg) fills and off-book reports. One row is produced per trade. The dataset includes regulatory reporting classification fields aligned with the MiFID II MMT standard.

Trade amendments are always preceded by a trade cancellation. Trade cancellations carry a negative Size field. When computing volume aggregates, filter to rows where Printable = True to avoid double-counting duplicate or synthetic trade reports.

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
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.
LISTING_ID	ListingId	NUMBER	INT64	The BMLL ListingId is used to identify a specific instrument on a market venue (trading venue or reporting venue). BMLL listings identify uniquely every ticker on the venue and remain attached to the instrument even when a ticker change occurs — though changes of primary instrument identifier (such as ISIN changes) can cause a new listing_id to be issued.

Field Name (Snowflake / Databricks)	Field Name (Parquet)	Type (Snowflake / Databricks)	Type (Parquet)	Description
INSTRUMENT_ID	InstrumentId	NUMBER	INT64	The BMLL identifier of the BMLL instrument. BMLL instruments are designed to connect together listings of a designated underlying asset when they are issued with similar condition (i.e. same quoting currency, geographical, legislation region, primary reference venue).
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.
TRADE_TIMESTAMP	TradeTimestamp	TIMESTAMP_NTZ	INT64	Time at which the trade was executed. Timezone is UTC. Generally delivered at microsecond precision.
PUBLICATION_TIMESTAMP	PublicationTimestamp	TIMESTAMP_NTZ	INT64	Time at which the trade was reported. Timezone is UTC. Generally delivered at microsecond precision.
LOCAL_TRADE_TIMESTAMP	LocalTradeTimestamp	TIMESTAMP_NTZ	INT64	

Field Name (Snowflake / Databricks)	Field Name (Parquet)	Type (Snowflake / Databricks)	Type (Parquet)	Description
LOCAL_PUBLICATION_TIMESTAMP	LocalPublicationTimestamp	TIMESTAMP_NTZ	INT64	Time at which the trade was reported. Time Zone is the local timezone. Generally delivered at microsecond precision.
TRADE_TIMESTAMP_NANOSECONDS	TradeTimestampNanoseconds	NUMBER	INT64	Time at which the trade was executed. Timezone is UTC. Generally delivered at nanosecond precision.
PUBLICATION_TIMESTAMP_NANOSECONDS	PublicationTimestampNanoseconds	NUMBER	INT64	Time at which the trade was reported. Timezone is UTC. Generally delivered at nanosecond precision.
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.
TRADE_SUMMARY_ID	TradeSummaryId	VARCHAR	BYTE_ARRAY	An ID representing the Trade summary. This is provided directly from the exchange feed. This will be repeated for each trade in the trade summary.

Field Name (Snowflake / Databricks)	Field Name (Parquet)	Type (Snowflake / Databricks)	Type (Parquet)	Description
TRADE_SUMMARY_PRICE	TradeSummaryPrice	FLOAT	DOUBLE	The price indicated by the trade summary message. This is provided directly from the exchange feed. This will be repeated for each trade in the trade summary.
TRADE_SUMMARY_SIZE	TradeSummarySize	FLOAT	DOUBLE	The size indicated by the trade summary message. This is provided directly from the exchange feed. This will be repeated for each trade in the trade summary.
TRADE_SUMMARY_AGGRESSOR_SIDE	TradeSummaryAggressorSide	NUMBER	INT64	Side of the aggressive trade summary: 2: Quotes from the BID side are aggressed by an aggressive sell order. 1: Quotes from the ASK side are aggressed by an aggressive buy order. 0: Quotes where the a
END_OF_TRADE_SUMMARY	EndOfTradeSummary	BOOLEAN	BOOLEAN	Indicates if this trade is the last linked to one trade summary

Field Name (Snowflake / Databricks)	Field Name (Parquet)	Type (Snowflake / Databricks)	Type (Parquet)	Description
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.
LINKED_TRADE_ID	LinkedTradeId	VARCHAR	BYTE_ARRAY	Any trade ID that is provided to link different trades. Example would be combinationTradeId is ASX24
BUY_PARTICIPANT_ID	BuyParticipantId	VARCHAR	BYTE_ARRAY	An ID to indicate the participant on the buy side of the trade
SELL_PARTICIPANT_ID	SellParticipantId	VARCHAR	BYTE_ARRAY	An ID to indicate the participant on the sell side of the trade
AGGRESSOR_SIDE	AggressorSide	VARCHAR	BYTE_ARRAY	Side of the aggressive trade
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
BEST_ASK_PRICE	BestAskPrice	FLOAT	DOUBLE	Best ask price at the time of the event. NaN if no ask orders.
BEST_BID_PRICE	BestBidPrice	FLOAT	DOUBLE	The best bid price if the book is not crossed during the auction

Field Name (Snowflake / Databricks)	Field Name (Parquet)	Type (Snowflake / Databricks)	Type (Parquet)	Description
BEST_ASK_SIZE	BestAskSize	FLOAT	DOUBLE	The size available at the best ask price on the Consolidated/National or Primary exchange (as determined by BMLL for region/country - see section 1.5) as of the trade timestamp (prior to trade). For
BEST_BID_SIZE	BestBidSize	FLOAT	DOUBLE	Size (quantity) at the best bid price at the time of the event.
FROM_IMPLIED	FromImplied	BOOLEAN	BOOLEAN	Indicates if the trade execution originated from an implied price.
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.
EXECUTION_VENUE	ExecutionVenue	VARCHAR	BYTE_ARRAY	MIC of the venue of which the trade was executed. If executed off-venue the execution venue will be XOFF. If it is executed on a systematic internaliser, the execution venue will be SINT

Field Name (Snowflake / Databricks)	Field Name (Parquet)	Type (Snowflake / Databricks)	Type (Parquet)	Description
MARKET_STATE	MarketState	VARCHAR	BYTE_ARRAY	Standardised market state information. See below for description of possible market states.
BMLL_TRADE_TYPE	BMLLTradeType	VARCHAR	BYTE_ARRAY	Normalised Trade Type according to BMLL normalisation
ORIGINAL_TRADE_TYPE	OriginalTradeType	VARCHAR	BYTE_ARRAY	A concatenation of flags relating to trade type as reported by the exchange in their string representation. Documentation of this field is exchange dependent. See appendices table for more information about this feld.
MARKET_MECHANISM	MarketMechanism	VARCHAR	BYTE_ARRAY	MMT Flag
TRADING_MODE	TradingMode	VARCHAR	BYTE_ARRAY	MMT Flag
TRANSACTION_CATEGORY	TransactionCategory	VARCHAR	BYTE_ARRAY	MMT Flag - Transactiion Category
NEGOTIATED_TRADE	NegotiatedTrade	VARCHAR	BYTE_ARRAY	MMT Flag - Negotiation Indicator or Pre-Trade Transparency Waiver
CROSSING_TRADE	CrossingTrade	VARCHAR	BYTE_ARRAY	MMT Flag - used to indicate if the trade was the result of a cross executed by the broker.
MODIFICATION_INDICATOR	ModificationIndicator	VARCHAR	BYTE_ARRAY	Indicator specifying if a trade has been amended or cancelled.
BENCHMARK_INDICATOR	BenchmarkIndicator	VARCHAR	BYTE_ARRAY	MMT Flag

Field Name (Snowflake / Databricks)	Field Name (Parquet)	Type (Snowflake / Databricks)	Type (Parquet)	Description
SPECIAL_DIVIDEND_INDICATOR	SpecialDividendIndicator	VARCHAR	BYTE_ARRAY	MMT Flag - Denotes trades that are either: executed during the ex-dividend period where the dividend or other form of distribution accrues to the buyer instead of the seller; or executed during the cum-dividend period where the dividend or other form of distribution accrues to the seller instead of the buyer.
OFF_BOOK_AUTOMATED_INDICATOR	OffBookAutomatedIndicator	VARCHAR	BYTE_ARRAY	MMT Flag
PRICE_FORMATION	PriceFormation	VARCHAR	BYTE_ARRAY	MMT Flag
ALGORITHMIC_TRADE	AlgorithmicTrade	VARCHAR	BYTE_ARRAY	MMT Flag
PUBLICATION_MODE	PublicationMode	VARCHAR	BYTE_ARRAY	MMT Flag
DEFERRAL_TYPE	DeferralType	VARCHAR	BYTE_ARRAY	MMT Flag
DUPLICATIVE_INDICATOR	DuplicativeIndicator	VARCHAR	BYTE_ARRAY	MMT Flag
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.

Field Name (Snowflake / Databricks)	Field Name (Parquet)	Type (Snowflake / Databricks)	Type (Parquet)	Description
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
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.
LOADED_AT	LoadedAt	TIMESTAMP_NTZ	INT64	

1.4. Notes

Basic Types One row per trade execution event. Events include order book executions, new trade reports, trade amendments, and trade cancellations. The Ticker field contains the exchange-provided ticker for the contract. ListingId and InstrumentId are present in the file but are currently reserved fields and should not be relied upon for analytical purposes. Aggressor Side The aggressor side is only populated for executions that can be deterministically mapped to a side of the order book. It is not inferred for trades where there is no clear aggressor (e.g. off-book trades). On some

venues, the exchange does not publish aggressor information, in which case AggressorSide will always be 0. MMT Flags Multiple fields are taken from the FIX Market Model Typology (MMT) standard (see https://www.fixtrading.org/mmt/). Where a venue provides MMT flags natively they are carried as-is. For venues where MMT is not provided, BMLL normalises equivalent fields into the MMT structure. Prevailing Quote Fields BestBidPrice, BestAskPrice, BestBidSize, and BestAskSize capture the prevailing top-of-book at the moment of the trade. Availability of these fields varies by venue; consult per-venue documentation for clarification.

1.5. Option Contract Identification

Options are identified by a composite key built from several fields that together uniquely identify a single tradeable contract. This key links records across all Options Data Feed files.

Level	Fields	What It Defines
Venue	MIC	The exchange (e.g. XEUR, XCME)
Product	MIC + UnderlyingTicker	All options on a given underlying
Expiry group	MIC + UnderlyingTicker + ExpiryDate	All strikes and option types for one expiry
Contract	MIC + UnderlyingTicker + ExpiryDate + StrikePrice + OptionType	A single tradeable option contract

1.6. Market State Normalisation

BMLL provides a consistent normalisation for market state messages to make cross-venue analysis straightforward. Market State Values

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.
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_OPEN and POST_TRADE.
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.
POST_TRADE	Market phase following CONTINUOUS_TRADING phase, and generally occurring after the primary CLOSING_AUCTION.
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.

1.7. BMLL Trade Type

Value	Description
BENCHMARK_PRICE	Trades executed at a publicly available reference price. This includes VWAP, TWAP, and other trades marked BENC under MiFID II.
CLOSING_PRICE	A trade executed at the closing price resulting from a Market-on-Close (MOC) order.
DARK	Dark trades result from matches on a dark order book which has either limited pre-trade transparency or is exempt from pre-trade transparency disclosure.
EXCHANGE_FOR_RELATED_POSITION	Arrangement between two parties whereby a futures or options position is exchanged for a related position or vice versa.
LIT	Lit trades result from a match on a lit order book with full pre-trade transparency. This group also includes executions of hidden orders that cannot be distinguished from standard lit executions, such as iceberg orders.
OTC	An over-the-counter (OTC) trade agreed bilaterally between two parties and reported to a trade reporting facility.
REQUEST_FOR_QUOTE	A trade resulting from a request for quote (RFQ).
SPECIAL_PRICE	Trades which execute at a special price. This includes negotiated trades, trades marked as not contributing to price discovery, and other technical trades.
UNCROSSING	Uncrossing trades occur when paired quantities are matched at the end of an auction market phase.
UNKNOWN	Used for trades which could not be resolved into one of the known trade types listed here.