BMLL Data Feed

Market State - Prediction

Date: 2026-04-28
Version: 1.0

1.1. Introduction

The BMLL Data Feed: Prediction Market State dataset records every market phase transition for every option contract. One row is produced per state change. BMLL provides a consistent normalisation for market state messages in order to make cross-venue analysis straightforward. There are 13 distinct normalised states. Transitions are recorded per individual contract rather than per venue. On a typical trading day, an options venue will generate a large number of state change rows covering all listed contracts.

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.
TIMESTAMP	Timestamp	TIMESTAMP_NTZ	INT64	The UTC timestamp indicates the date and time at which the update occurred in the timezone of the primary record associated with the venue, with microsecond precision.
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.

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.
EXCHANGE_MARKET_STATE	ExchangeMarketState	VARCHAR	BYTE_ARRAY	Flags related to market state as implemented by exchange. This field contains an exchange specific values that represent the latest exchange-specific flags associated with a change of market state on the main order lit order book.
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.
LOADED_AT	LoadedAt	TIMESTAMP_NTZ	INT64	

1.4. Notes

One row per market state transition per contract. 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.

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 Definitions

The following are the BMLL normalised Market State definitions. The exchange-native state code is always preserved in the ExchangeMarketState field.

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

Value	Description
	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.
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 bmll_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.