Kaiko

MIGRATION GUIDE

Futures & Perpetual Trades

HTTP REST

Amberdata → Kaiko

2026

www.kaiko.com

1 At a glance

What this endpoint gives you. The raw, tick-level trade history for a single exchange and futures contract — dated (expiring) futures and perpetual futures alike — with price, size, side, trade ID and timestamp, over a date range you choose.

What changes for you

- **One Amberdata endpoint becomes two Kaiko instrument classes.** Amberdata's single `/futures/trades` endpoint covers both dated and perpetual futures; Kaiko requires you to know which one you're asking for and put it in the URL as `future` or `perpetual-future` (see §2 and §3). Your client needs new logic to classify an instrument before building the Kaiko request.
- `exchange` and `instrument` move from query parameters to the **URL path**, same as Spot Trades.
- **Instrument code construction differs by contract type** — unlike Spot's simple separator swap, see §6.
- `volume` / `amount` here means **number of contracts**, not underlying asset quantity — same on both sides, but different from what Spot Trades' same-named field means (see §5).
- The response envelope and the same highest-risk field adaptations as Spot Trades apply here too (trade-side polarity, price/amount type change, status/result envelope change) — see §7.

Coverage & distribution

Data type: Tick-level (per-trade) futures trade records
Contract types: Dated (expiring) futures and perpetual futures
Source: One CEX derivatives exchange + instrument per request
Granularity: Per-trade (no aggregation)
Distribution: REST API

2 Endpoint & authentication

	Amberdata (today)	Kaiko (new)
Method	GET	GET
Base URL	`https://api.amberdata.com`	`https://us.market-api.kaiko.io` (or `eu.market-api.kaiko.io`)
Path	`/markets/futures/trades/{instrument}`	`/v4/data/exchanges/{exchange}/{instrument_class}/{instrument}/trades`
Auth	API key header `x-api-key`	API key header `X-Api-Key` (same mechanism, new key)

`{instrument_class}` is `future` for dated/expiring contracts or `perpetual-future` for perpetuals — confirmed via real calls: `.../exchanges/binc/perpetual-future/btc-usd/trades` and `.../exchanges/binc/future/btcusd261225/trades`.

Kaiko API family: Classic (versioned)

This endpoint belongs to Kaiko's existing versioned API. Its URL structure and response envelope differ from Kaiko's New API grammar endpoints (`market-data`), such as OHLCV Batch and Tickers.

3 Request parameters

Amberdata param	Kaiko param	Change	Notes
—	`{instrument_class}` (path, required)	NEW	**No Amberdata equivalent.** Amberdata's single endpoint covers both contract types; Kaiko requires you to pick `future` or `perpetual-future` upfront. Wrong value likely means the instrument isn't found, not a silent error.
`{instrument}` (path, required)	`{instrument}` (path, required)	RENAMED	Same position (URL path) on both sides, but the code itself is constructed differently per contract type — see §6, not a simple separator swap like Spot.
`exchange` (query, required; only 1 allowed)	`{exchange}` (path, required)	RENAMED	Moves from a query parameter to a **URL path segment**, same as Spot Trades. Exchange codes also change (see §6).
`startDate` (optional)	`start_time` (optional)	RENAMED	Amberdata accepts epoch seconds, epoch milliseconds, or ISO 8601; Kaiko accepts **ISO 8601 only**. Inclusive lower bound on both Amberdata and Kaiko.
`endDate` (optional)	`end_time` (optional)	RENAMED	Same format restriction as `start_time`. Exclusive upper bound on both Amberdata and Kaiko.
`timeFormat` (optional)	—	REMOVED	Kaiko always returns `timestamp` as unix nanoseconds; no alternate format option.
`sortDirection` (optional)	`sort` (optional)	RENAMED	Same `asc` / `desc` values; Kaiko defaults to `desc`.
—	`page_size`, `continuation_token` (optional)	NEW	Pagination controls; no Amberdata equivalent param for this endpoint.

4 Response envelope & metadata

Identical envelope mechanics to Spot Trades — see that guide for the full breakdown. Summary below.

Amberdata field	Kaiko field	Change	Notes
`status` / `title` / `description`	`result`	RENAMED	**Adaptation required.** Single string (e.g. `"success"`) replaces the HTTP-style status object. Error-handling code must be rewritten, not just renamed.
`payload.data`	`data`	RENAMED	**Array** holding the trade records (see §5), now at the top level.
`payload.metadata.api-version`	`query.data_version`	RENAMED	Data version surfaced inside the echoed `query` object.
`payload.metadata.next`	`continuation_token` + `next_url`	RENAMED	Pagination mechanism differs — rewrite pagination logic, don't just rename.
—	`time`, `timestamp`	NEW	Server response time. No Amberdata equivalent.

5 Response - data fields

Amberdata field	Kaiko field	Type	Change	Notes
`instrument`	(path) / `query.instrument`	string	REMOVED	Not repeated per record, but recoverable from the request path or the echoed `query.instrument`.
`exchange`	(path) / `query.exchange`	string	REMOVED	Not repeated per record, but recoverable from the request path or the echoed `query.exchange`.
`exchangeTimestamp` + `exchangeTimestampNanoseconds`	`timestamp`	string+int → integer	RENAMED	Merged into one unix-nanosecond integer, same as Spot Trades. No precision lost.
`isBuySide`	`taker_side_sell`	boolean	RENAMED	**Adaptation required.** Same inverted convention confirmed for Spot Trades (§7 there): `isBuySide = NOT taker_side_sell`.
`price`	`price`	number → string	IDENTICAL NAME	**Adaptation required.** Same field name, JSON type changes from number to string.
`volume`	`amount`	integer → string	RENAMED	**Different unit than Spot Trades.** For futures/perpetuals this is the **number of contracts** traded, not an underlying asset quantity — true on both sides, but easy to misread if you've already migrated Spot Trades. Also changes type to string.
`tradeId`	`trade_id`	string	RENAMED	Casing only.
`sequence`	`sequence`	string/number → null	IDENTICAL NAME	Both `null` on this product. Kaiko's is reserved for future use regardless (see Spot Trades §5).
`quoteVolume`	`quote_volume`	number (nullable) → string	RENAMED	Same pattern as Spot Trades: Kaiko **computes** it and populates it even where Amberdata returns `null`.

6 Exchange & instrument names

Kaiko identifies venues and instruments with its own normalized codes, which differ from Amberdata's names. Codes can also differ by market for the same venue (e.g. Binance spot and options).

Reference

The mapping rules and the full code lists apply to every endpoint and are described in the **Exchange, Asset & Instrument Mapping** migration guide (`exchange-instrument-asset-codes.pdf`).

7 Will your values change?

Short answer

Same answer as Spot Trades: the underlying trade facts are the exchange's raw executed trades on both sides, tick-for-tick. The same encoding differences apply here — confirmed again on real futures and perpetual data — plus one new structural requirement specific to this product (choosing `instrument_class` upfront).

Highest-risk adaptations (same as Spot Trades, reconfirmed here)

- **Trade side is inverted.** `isBuySide = NOT taker_side_sell`. A rename-only migration flips every trade side.
- `price` **and** `amount` **/** `volume` **change JSON type** (number → string).
- **Status/error handling changes shape** (`status` / `title` / `description` → single `result` string).

- **Instrument classification is a new, required step.** Amberdata needs no up-front distinction between a dated and a perpetual future; Kaiko does (§2, §3). Get it wrong and the request fails loudly (not a silent data error), but it's new logic your client doesn't have today.
- `amount` **/** `volume` **= contracts, not asset quantity.** True on both sides for this product; don't confuse with Spot Trades' same-named field, which is an underlying asset quantity.
- **Instrument code is not mechanically derivable.** Unlike Spot's separator swap, building the Kaiko code for futures/perpetuals requires parsing base/quote/expiry from the Amberdata side first (see §6).
- `quote_volume`**.** Kaiko always computes it; expect it populated even where Amberdata's `quoteVolume` is `null` (confirmed again in these samples).

8 Before / after example

Perpetual future — Request

Amberdata

```
GET /markets/futures/trades/BTCUSD_PERP
    ?exchange=binance
x-api-key: <your-key>
```

Kaiko

```
GET /v4/data/exchanges/binc/perpetual-future/btc-
usd/trades
X-Api-Key: <your-key>
```

Perpetual future — Response (real, captured)

Amberdata


```
{
  "description": "Successful request",
  "payload": {
    "data": [
      {
        "instrument": "BTCUSD_PERP",
        "exchange": "binance",
        "exchangeTimestamp": "2026-10-07 08:49:25
494",
        "exchangeTimestampNanoseconds": 0,
        "isBuySide": true,
        "price": 83939.3,
        "volume": 1,
        "tradeId": "1157621243",
        "sequence": null,
        "quoteVolume": null
      }
    ]
  }
}
```


Kaiko


```
{
  "result": "success",
  "time": "2026-10-08T08:57:23Z",
  "timestamp": 1791449843038,
  "query": {
    "exchange": "binc",
    "instrument": "btc-usd",
    "instrument_class": "perpetual-future",
    "start_time": "2010-01-01T00:00:00Z",
    "end_time": "2026-10-08T08:57:22.967379156Z",
    "page_size": "100",
    "sort": "desc"
  },
  "data": [
    {
      "timestamp": 1791449831258000000,
      "trade_id": "1157815337",
      "price": "82779.1",
      "amount": "176",
      "taker_side_sell": false,
      "quote_volume": "14569121.6",
      "sequence": null
    }
  ]
}
```


The two calls use different trades (real, independently captured examples, not a matched pair) — shown for envelope and field-mapping purposes, not a side-by-side value comparison.

Dated future — Request (real)

Amberdata

```
GET /markets/futures/trades/BTCUSD_261225
    ?exchange=binance
x-api-key: <your-key>
```

Kaiko

```
GET /v4/data/exchanges/binc/future/btcusd261225/
trades
X-Api-Key: <your-key>
```

Dated future — Response (real, captured)

Amberdata


```
{
  "description": "Successful request",
  "payload": {
    "data": [
      {
        "instrument": "BTCUSD_261225",
        "exchange": "binance",
        "exchangeTimestamp": "2026-10-07 09:18:58
595",
        "exchangeTimestampNanoseconds": 0,
        "isBuySide": true,
        "price": 84605.1,
        "volume": 1,
        "tradeId": "789651",
        "sequence": null,
        "quoteVolume": null
      }
    ]
  }
}
```


Kaiko


```
{
  "result": "success",
  "time": "2026-10-08T09:01:27Z",
  "timestamp": 1791450087726,
  "query": {
    "exchange": "binc",
    "instrument": "btcusd261225",
    "instrument_class": "future",
    "start_time": "2010-01-01T00:00:00Z",
    "end_time": "2026-10-08T09:01:27.685935024Z",
    "page_size": "100",
    "sort": "desc"
  },
  "data": [
    {
      "timestamp": 1791450077945000000,
      "trade_id": "800164",
      "price": "83725.1",
      "amount": "1",
      "taker_side_sell": false,
      "quote_volume": "83725.1",
      "sequence": null
    }
  ]
}
```