Datomic Cloud Documentation

Search

What is Datomic?

- Data Model
- Architecture
- Supported Operations
- Programming with Data and EDN

Local Dev and CI

Cloud Setup

- Start a System
- Configure Access
- Get Connected

Tutorial

- Client API
- Assertion
- Read
- Accumulate
- Read Revisited
- Retract
- History

Client API

Videos

- AWS Setup
- Edn
- Datalog
- Datoms
- HTTP Direct
- CLI tools

Schema

- Defining Attributes
- Schema Reference
- Changing Schema
- Data Modeling
- Schema Limits

Transactions

- Processing Transactions
- Transaction Data Reference
- Transaction Functions
- ACID
- Client Synchronization

Query and Pull

- Executing Queries
- Query Data Reference
- Pull
- Index-pull
- Raw Index Access
  Datomic IndexesDatoms APIIndex-range API

Time in Datomic

- Log API
- Time Filters

Ions

- Ions Tutorial
- Ions Reference
- Monitoring Ions

Analytics Support

- Configuration
- Connecting
- Metaschema
- SQL CLI
- Troubleshooting

Analytics Tools

- Metabase
- R
- Python
- Jupyter
- Superset
- JDBC

Hide All Examples

Raw Index Access

Most Datomic programs will not access the indexes directly, instead taking advantage of Datomic's declarative Datalog query engine.

However, raw index access is available for integrations. For example, you might have use the indexes to make Datomic data available to a map/reduce framework spanning multiple data sources of different types.

Datomic Indexes

Datomic automatically maintains four indexes. Each index sorts datoms in a specific order. These indexes are named based on the sort order used:

Index	Sort Order	Contains
EAVT	entity / attribute / value / tx	all datoms
AEVT	attribute / entity / value / tx	all datoms
AVET	attribute / value / entity / tx	all datoms
VAET	value / attribute / entity / tx	all reference datoms

EAVT Index

The EAVT index provides efficient access to everything about a given entity. Conceptually this is very similar to row access style in a SQL database, except that entities can possess arbitrary attributes rather then being limited to a predefined set of columns.

The example below shows all of the facts about entity 42 grouped together:

E	A	V	Tx	Op
41	:release/name	“Abbey Road”	1100	assert
42	:release/name	“Magical Mystery Tour”	1007	assert
42	:release/year	1967	1007	assert
42	:release/artistCredit	“The Beatles”	1007	assert
43	:release/name	“Let It Be”	1234	assert

EAVT is also useful in master/detail lookups, since the references to detail entities are just ordinary Vs alongside the scalar attributes of the master entity. Better still, Datomic assigns entity ids so that when master and detail records are created in the same transaction, they will be colocated in EAVT.

Datomic's EAVT and VAET indexes can automatically navigate entity relationships in both directions, so you do not need to (and should not) create two attributes that model the same relationship but from different directions.

AEVT Index

The AEVT index provides efficient access to all values for a given attribute, comparable to traditional column access style. In the table below, notice how all `:release/name` attributes are grouped together. This allows Datomic to efficiently query for all values of the `:release/name` attribute, because the values reside next to one another in the index.

A	E	V	Tx	Op
:release/artistCredit	42	“The Beatles”	1007	assert
:release/name	41	“Abbey Road”	1007	assert
:release/name	42	“Magical Mystery Tour”	1007	assert
:release/name	43	“Let It Be”	1234	assert
:release/year	42	1967	1007	assert

AVET Index

The AVET index provides efficient access to particular combinations of attribute and value, much like a key-value store. The example below shows a portion of the AVET index allowing lookup by `:release/name`.

A	V	E	Tx	Op
:release/name	“Abbey Road”	41	1199	assert
:release/name	“Let It Be”	43	1234	assert
:release/name	“Let It Be”	55	2367	assert
:release/name	“Magical Mystery Tour”	42	1007	assert
:release/year	1967	42	1007	assert
:release/year	1984	55	2367	assert

The AVET index also supports the `index-range` operation, which returns all attribute values in a particular range. For more information about `index-range`, refer to the Datomic API documentation for your programming language.

VAET Index

The VAET index contains all and only datoms with `:db/valueType` attributes whose value is `:db.type/ref`. This is also known as the *reverse index*, since it allows efficient navigation of relationships in reverse. This is similar to graph-style access.

If The Beatles are entity 100, you can see in the following table how their releases would be grouped together in this index.

Datomic's EAVT and VAET indexes can automatically navigate entity relationships in both directions, so you do not need to (and should not) create two attributes that model the same relationship but from different directions.

V	A	E	Tx	Op
100	:release/artists	41	1007	assert
100	:release/artists	42	1007	assert
100	:release/artists	43	1007	assert
200	:release/artists	55	2367	assert

Datoms API

The datoms API lets you specify a Datomic index, plus a vector of values in the same order as the index. The example excerpted below searches the `:avet` index for datoms whose `a` component is `:inv/sku`

```
(->> (d/datoms db
               {:index :avet
                :components [:inv/sku]})
     (take 3)
     (map :v))
;; => ("SKU-0" "SKU-1" "SKU-10")
```

Index-range API

The index-range API lets you specify an attribute, and returns all datoms sorted by that attribute, optionally limited by start and end values. The example excerpted below returns all the SKUs between "SKU-42" inclusive and "SKU-44" exclusive.

```
(->> (d/index-range
      db
      {:attrid :inv/sku
       :start "SKU-42"
       :end "SKU-44"})
     (map :v))
=> ("SKU-42" "SKU-43")
```