# Referencing Record Fields

Many builtin processors can be configured to work on a specific field in an OpenCDC record . That is done throughfield references , strings that describe the path to a field in a record. That can be a field within an OpenCDC record (such as the metadata or the payload), but it can also be anested field .

To reference one of the OpenCDC record fields, you can use a similar notation to accessing fields in a Go template executed on anopencdc.Record value. For example,.Metadata will reference the field namedMetadata .

Why do record fields start with an uppercase letter? The main record fields start with an uppercase letter, because they are public fields in the Go typeopencdc.Record which is used to resolve field references (e.g.Metadata ,Position ,Operation ,Key ,Payload ).

## Accessing Nested Fields

Nested fields can be accessed using two different notations: thedot notation and thebracket notation :

- Thedot notation
- is used to access fields containing only alphanumeric
- characters. For example, the reference.Metadata.foo
- will access the
- field namedfoo
- in theMetadata
- field.
- Thebracket notation
- is used to access fields containing non-alphanumeric
- characters. For example, the reference.Metadata["opencdc.readAt"]
- will
- access the field namedopencdc.readAt
- in theMetadata
- field.

## Examples

Below is an example OpenCDC record (left) and the field references that can be used to access the fields in the record (right):

Example OpenCDC record { "position" :

"c3RhbmRpbmc=" , "operation" :

"update" , "metadata" :

{ "foo" :

"bar" , "opencdc.readAt" :

"1663858188836816000" } , "key" :

"cGFkbG9jay1rZXk=" , "payload" :

{ "before" :

"eWVsbG93" , "after" :

{ "boolField" :

true , "nested" :

{ "non-alphanumeric key!" :

"baz" } } } } Field references per line . .Position .Operation .Metadata .Metadata.foo / .Metadata [ "foo" ] .Metadata [ "opencdc.readAt" ]

.Key .Payload .Payload.Before .Payload.After .Payload.After.boolField / .Payload.After [ "boolField" ] .Payload.After.nested / .Payload.After [ "nested" ] .Payload.After.nested [ "non-alphanumeric key!" ] / .Payload.After [ "nested" ] [ "non-alphanumeric key!" ] A few things to note:

- .

- references the entire record (i.e. the JSON object above).
- Fields.Key
- and.Payload.Before
- contain raw data (i.e. byte arrays), which
- is represented as a base64-encoded string. We can not reference nested fields
- in raw data. However, if the raw data is first parsed into structured data (e.g.
- if it's a JSON string we can use the json.decode
- processor), then we can
- reference the fields in the structured data.
- .Metadata["opencdc.readAt"]
- references the metadata fieldopencdc.readAt
- using the bracket notation. The dot notation (i.e..Metadata.opencdc.readAt
- )
- cannot be used here because the referenced keyopencdc.readAt
- contains a
- non-alphanumeric character (the dot).
- Note that references to fields nested inside.Key
- ,.Payload.Before
- and.Payload.After
- do not
- start with an uppercase letter, because these
- fields are not part of theopencdc.Record
- type. They are referenced by their actual names, as they appear in JSON.Edit this page Previous Referencing Processors Next Processor Concurrency