Table of Contents

Chapter 1. Relationship hydration with expand(path)

This section documents the new relationship traversal and hydration capability added to the BIAPI query language. It preserves backward compatibility and introduces no SQL-like syntax.

1.1. What is expand(path)?

Use expand(path) to hydrate properties that are modeled as EntityReference or ontology-annotated relations. The path is dot-based and may include array wildcards [*] between segments.

- Single reference: expand(customer)
- Array of references: expand(items[*].product)
- Nested arrays: expand(patient.visits**[].diagnoses[**])

Expansion is a query-time operation: the related document(s) are materialized and embedded at the specified path in the result.

1.2. Filtering related entities (inline)

Continue to use your existing filter primitives in the same query string to filter the root collection. Filtering on related entities will be supported via either:

- Inline path blocks: path { key:value, key2:^[v1,v2] }
- Or filters inside expand(…): expand(customer, status:active)

Note: In v1, parsing of expand(path) is enabled; related-entity filters inside expand and inline path blocks will be introduced with the planner/compiler work.

1.3. Projection with fields:[+/-]

A unified projection syntax can be used both at the root and per expansion.

- Inclusion mode: fields:[+_id,+total,+customer.name]
- Exclusion mode: fields:[-internalNotes,-secret]
- Inside expand: expand(customer, fields:[+name,+tier,-ssn])

Rules: - If any + appears → include only those paths, then apply - as carve-outs. - If only - appears → include all by default and remove listed paths. - _id keeps current default behavior unless explicitly set.

1.4. Examples

- Hydrate a single reference and keep a few fields

```
q = "realm:acme && expand(customer, fields:[+name,+tier]) &&
fields:[+_id,+total,+customer.name]"
```

- Hydrate array of product references inside line items, filter to active products, project a couple of fields

```
q = "expand(items[*].product, active:true, fields:[+sku,+title]) &&
fields:[+_id,+items.product.sku]"
```

- Filter by a related product without projecting it (inline block form)

```
q = "items[*].product{ active:true, sku:^[A123,B456] } && fields:[+_id,+total]"
```

1.5. Semantics and limits

- Backward compatible: if no expand(…) or related-path filters are present, queries run as they do today (single-collection).
- Depth for nested paths is bounded; on mixed array/object hops, traversal stops at the first non-reference boundary.
- Unknown projection paths are hard errors (the request fails with a clear message).

See also: Planner and QueryGateway