— CHAPTER 09 · SEARCH OPERATIONS · BEAT 1 — NOW WE ASK

Every chapter built resources. This one asks them questions.

FHIR search is a query language that rides on plain URL parameters, and it goes far deeper than most people ever use. We compose queries across the parameter types and modifiers, traverse references with chaining and includes, and — the lesson that bites — keep all that power from blowing up your server.

What you will be able to do

- Params & modifiers — types, prefixes, refinements
- Chains & includes — query the reference graph
- The blowup — bound _include and $everything

— COMPARISON · EVERY PARAMETER HAS A TYPE — AND THE TYPE DECIDES THE MATCH

Nine types, three families. Get the type right, search falls into place.

Get it wrong and your query silently matches nothing. Group them: a value-shape family, an ordered family, and composite — the combiner.

— VALUE-SHAPE FAMILY

string · token · reference · uri

string — case- and accent-insensitive prefix match by default; "smi" matches "Smith". token — a coded value as system|code, the Coding shape. reference — a literal reference, the links we built in Ch 03. uri — an absolute address.

— ORDERED FAMILY

date · number · quantity

Values with a natural ordering — so they accept prefixes like greater-than and less-than. quantity also carries a unit. composite combines two values in one expression — for example a component code paired with its value.

- string = prefix match, no case
- token = system|code
- ordered types take prefixes

— COMPARISON · APPEND A COLON AND A KEYWORD TO REFINE THE MATCH

Most missed-result bugs are a forgotten — or wrong — modifier.

Land the universal one first: :missing works on every type. After that, modifiers behave differently per type. Reach for one deliberately — the default is almost always a prefix or exact-code match.

:missing

UNIVERSAL · ALL TYPES

gender:missing=true finds resources where gender was never set. true or false.

string

:EXACT · :CONTAINS · :TEXT

:exact turns off the prefix match. :contains does a substring. :text searches the display text.

token

:NOT · :ABOVE · :BELOW

:not excludes a code. :above and :below walk a terminology hierarchy — :below pulls in the more-specific children.

- :missing — the one that works everywhere
- :below — subsumption, the Ch 04 hierarchies
- Wrong modifier = silent zero results

— DIAGRAM · DATES AND NUMBERS NEED MORE THAN EQUALS

Nine prefixes — and they apply to the ordered types only.

One rule to remember: prefixes never apply to string, token, or uri — only date, number, and quantity. eq is the default.

— WORKED EXAMPLE — BORN ON OR AFTER

GET Patient?birthdate=ge2010-01-01

ge is greater-than-or-equal, so this returns patients born on or after the first of January, twenty-ten.

— THE EVERYDAY SIX

eq · ne · gt · lt · ge · le

eq equals (the default), ne not-equals, gt and lt for greater-than and less-than, ge and le for the or-equal versions.

— THE THREE PEOPLE FORGET

sa · eb · ap

sa starts-after and eb ends-before — both for ranges. ap is approximately, a fuzzy window around the value.

— WALKTHROUGH · CROSS A REFERENCE INSIDE A SINGLE QUERY

The dot hops from the Observation into the Patient it points at.

— FORWARD CHAIN — ONE REQUEST, NO ROUND TRIPS

GET Observation?subject:Patient.name=Smith

Observations whose subject resolves to a Patient whose name is Smith. subject is the reference, :Patient types it because subject can point at several resource kinds, and .name reaches into that patient.

Ava

That is the whole reference graph, queryable from a URL.

Marcus · mechanism

Don't fetch the patients first — chain straight through. One request.

— WALKTHROUGH · THE INVERSE OF CHAINING — WHO POINTS BACK AT ME

Forward chain follows a reference out. _has finds who points at you.

— REVERSE CHAIN — READ IT BACKWARDS

GET Patient?_has:Observation:patient:code=1234-5

Give me patients that are referenced — through the patient field — by some Observation whose code is one-two-three-four-dash-five. The arrow points backward, from the Observation to the Patient.

Ava

Reach for _has when the question is "which parents have a matching child".

Marcus · the subtlety

One trap — stack separate criteria and each chain evaluates independently. They need not be satisfied by the same target resource.

— CONCEPT · THE BLOWUP, PART 1 — WHEN ONE QUERY DRAGS IN THOUSANDS

_include pulls the referenced. _revinclude pulls the referencing.

— _INCLUDE — REFERENCED RIDE ALONG

Observation?_include=Observation:subject

The referenced patients arrive in the same bundle. _revinclude is the mirror — pull resources that reference your matches, like Provenance. Included entries carry search.mode=include, not match, so you can tell what you asked for from what came along.

— THE BLOWUP — _INCLUDE=* / :ITERATE

_include=*&_include:iterate=*

Wildcard every reference, or :iterate to recurse, and one innocent query drags in thousands of resources — a bundle that hammers the server and the client. The fix is discipline: name the exact includes you need, and always cap with _count.

12

BOUNDED · NAMED INCLUDE

One explicit _include plus _count=10 — a small, predictable bundle.

8,400

UNBOUNDED · _INCLUDE=* :ITERATE

Wildcard recursion sweeps the reference graph — the server pays for every hop.

_count

THE BRAKE · ALWAYS PRESENT

Power, bounded. Name the includes, cap the page, and the blowup never happens.

— CONCEPT · THE BLOWUP, PART 2 — THE WHOLE COMPARTMENT IN ONE CALL

$everything is wonderful for an export and brutal as a casual query.

It returns the entire patient compartment — every resource that references the patient. That is a whole-compartment sweep, not a targeted search. On a patient with years of history it is a very large response.

— WHOLE-COMPARTMENT OPERATION

GET Patient/123/$everything?_count=50

HAPI bounds it with _count and paging so it does not fall over. Same lesson as _include: convenience plus a count limit equals safe. Convenience alone equals an incident.

1

CALL

$everything is a single convenient request.

ALL

COMPARTMENT RESOURCES RETURNED

Every Observation, Encounter, Condition, MedicationRequest that references the patient.

_count

PAGING · THE ONLY SAFE WAY

Bound the page size and walk the next-links — never pull it unbounded in a live query.

— WALKTHROUGH · WHEN THE BUILT-IN PARAMETERS DON'T COVER YOUR FIELD

Define your own SearchParameter — then reindex, or get zero results.

— SEARCHPARAMETER RESOURCE — FOUR THINGS MATTER

{ "resourceType": "SearchParameter", "code": "ejection-fraction", "base": ["Observation"], "type": "number", "expression": "Observation.valueQuantity.value" }

code — the name you query with. base — which resource types it applies to. type — one of the nine. expression — a FHIRPath that points at the element to index.

Ava

New writes, yes — they get indexed as they come in.

Marcus · production scar

I shipped a custom param once, got zero results, and panicked. The rows were there — just not indexed for the new path. New param … then reindex … then it works.

— DIAGRAM · WHY A NEW PARAMETER NEEDS A WHOLE REINDEX

Index at write. Query the index — never the stored JSON.

On the JPA server we toured in Ch 06, search never reads your stored document at query time. The mental model is what matters; the exact table layout is a version-specific implementation detail, so don't memorize it.

Write path — HAPI extracts, then indexes

One indexed entry per searchable parameter value

↓

↓

3 **Index entries written — organized by type**
Strings in one place, tokens in another, dates in another. References resolve through a link relationship between resources.

↓

4 **Search hits the index**
A query reads those compact indexes, not the full document — that is why search is fast. A new parameter means new index entries, which is exactly what reindex builds.

— TRY IT YOURSELF · SEE MATCH AND INCLUDE SIDE BY SIDE

One chained query, one include — read back the search modes.

On the public HAPI sandbox, run a chained query with an include and ask for just the search modes. You will see match and include side by side, exactly as Marcus described.

— RUN THIS IN YOUR TERMINAL

curl 'https://hapi.fhir.org/baseR4/Observation?subject:Patient.name=Smith&_include=Observation:subject&_count=5' | jq '.entry[].search.mode'

The matched Observations come back as match; the included Patients come back as include — the distinction, live.

Recap

- 9 parameter types
- 7 modifiers
- 9 ordered prefixes
- chain + _has
- _include + the blowup