JUDICIAL INTELLIGENCE

Judge and Court Intelligence

Research how a judge or court has ruled on a category of motion, and turn the result into an argument-ready briefing.

HOW TO USE THIS DOCUMENT

Upload this PDF to the AI assistant you use with the Docket Navigator connector and ask it to create a skill from it. The pages that follow are the complete skill: the numbered sections are its instructions, and its frontmatter is listed at the end. The assistant needs the Docket Navigator MCP connector connected and authenticated before the skill will run.

Produce empirically grounded answers about how a judge decides a motion type, grounded in coded events and verified against source documents. The failure mode this skill prevents is a fluent, confident briefing built on the wrong denominator or on inferred reasoning.

When to use

Triggers include: "how does Judge X rule on motions to Y", "what's their grant rate", "which orders should I read before the hearing", "what did the court actually say", "is this district different from this judge", "narrow to the ones where Z was true".

Do not use for substantive legal research on doctrine, or for questions answerable without docket analytics.

Library independence

Targets are library-prefixed: `{lib}_events`, `{lib}_motion_success_by_year`, and so on, where `{lib}` is `patent`, `trademark`, `copyright`, or another library. Never assume a target or criterion name exists in a library. Call `docnav_discover_search_options` with the library and a target hint,

and read the returned criteria, join paths, and field names before composing a search. Vocabulary IDs are namespaced per library and are not portable between them.

Core discipline

Five rules. Violating any one produces a confidently wrong answer.

1. Resolve every ID. Never guess one.

Judges, courts, document types, results, and issues all resolve through `docnav_lookup_search_values`. An ID recalled from a prior session belongs to that session's library and may be wrong here.

2. Verify the join path.

A criterion can accept more than one join path, and the paths mean different things. Judge criteria in particular distinguish the judge who decided the document from judges associated with the case. The second returns rulings by other judges and events predating the target judge's commission. Check the `description` block the search returns: it renders the human-readable label for the path actually applied. If the label does not say what you intended, the count is measuring something else. A quick sanity check: if the earliest event predates the judge's appointment, the path is case-scoped and wrong.

3. Specify the document type explicitly.

Planner-selected categories bundle many action IDs. Resolve the exact action you need and confirm it appears in the composed criteria. A category labeled for a family of motions may not contain the specific motion at issue.

4. Name the unit of count.

Results are coded document-type events, not cases and not orders. One case can contribute several events; one order can be coded twice when it disposes of two motions. Say "coded rulings" and never "cases".

5. Separate what the database says from what you concluded.

Annotations are the database's record of the court's reasoning. Patterns across annotations are your interpretation over a sample. Label them differently.

Workflow

Step 1 · Resolve scope

Call `docnav_plan_search_workflow` for orientation, then resolve each entity with `docnav_lookup_search_values`:

- Judge or court, using `autocomplete`
- Document type (the motion), using `tree` or `autocomplete`
- Result values, using `list`

Confirm the judge resolved to exactly one person. Confirm the document type is the specific motion, not a parent category.

Step 2 · Establish the denominator

Run the scope plus document type with no result filter against `{lib}_events`. This total is the denominator for everything downstream. Note the date range of the returned events and check it against the judge's tenure.

Step 3 · Break out results

Query each result family separately and reconcile to the total.

- **Grants:** the grant result plus, if relevant, partial grants
- **Non-grants:** denied, denied without prejudice, denied as moot, denied in part granted in part
- **Residual:** anything left over, commonly deferred rulings or further-briefing orders

If grants plus non-grants do not equal the total, query the remaining result IDs and identify the difference. Never present a rate that silently drops unreconciled events. State the rate with its denominator and say which events it excludes.

Soft dispositions matter. A denial without prejudice preserves refiling. A denial as moot is not a merits ruling. A denial followed by a grant in the same case months later is not a hostile judge. Read the dates before characterizing the record.

Step 4 · Retrieve reasoning

Re-run the non-grant query (and the grant query) requesting the annotation field along with action name, result, filing date, case name and ID, and document GUID.

Annotations attach only to editorially annotated orders, usually a minority. Report the annotated ones as reasoned authority and the rest as uncoded. **Never fill an uncoded row by inference from the caption, the parties, or the outcome.** That is the single most damaging failure available in this workflow.

Extract from each annotation: the coded issues (these are the factors the court weighed), the quoted language, and the page number.

Step 5 · Verify language against the source

Before any quotation reaches a brief or a partner, retrieve the filing text with `docnav_read_filing_text` using the document GUID. Check three things:

- **Whether the language is the court's own.** A sentence the judge quotes from an earlier decision is not that judge's holding. Look for internal quotation marks and citations in the retrieved text. Report the original source when you find one.
- **What the annotation compressed.** Annotations use ellipses. The omitted material is sometimes more probative than what was kept, particularly on procedural posture.
- **Whether text is available at all.** Scanned orders and oral orders return no extractable text. Say so explicitly and attribute any characterization to the annotation rather than the order. Do not let a scanned order be quoted as if verified.

Extracted text carries OCR noise. Treat substance as reliable and citation formatting as unreliable, and say which is which.

Step 6 · Benchmark

Run the same document type against the court with no judge filter. Report the judge's rate and the court's rate together.

Two caveats are mandatory. First, the court series usually starts long before the judge's appointment, so bound the court query to the judge's tenure and give both figures. Second, characterize the gap proportionately. A handful of percentage points across differing windows

supports "broadly in line", not "markedly more receptive".

Watch composition. A high grant rate can be driven by one repeat plaintiff or one class of case. Check the case names behind a cluster before treating the rate as a general disposition.

Step 7 · Time series

Bucket the events by year and reconcile the sum to the total from Step 2. Show grants and non-grants distinctly.

Be honest about small numbers. A few decisions a year cannot support a directional trend claim in either direction. If the volume drops off, check whether the judge's docket was reassigned before concluding that motion practice changed. Empty recent years are worth verifying against the underlying dockets rather than reported as fact.

Deriving a posture the data does not code

Some questions turn on a state that is not a coded field. The pattern is: take the order date, pull the parallel record, and resolve the state **as of that date**.

The worked example below is the patent-library case of PTAB institution status at the time of a stay order. In other libraries, apply the same shape against whatever parallel-proceeding target the library exposes, discovered rather than assumed.

PTAB institution posture as of a district-court stay order

1. From the district-court case profile, take the stay order date and the asserted patent numbers.
2. Query `patent_ptab_institutions` with `criterion_name: patent_number`, `joinpath_name: patent_number`, value = the full patent list. This returns per-ground rows with `unpatentability_result`, `claim_numbers`, `case_cite` (the PTAB proceeding), and `document_filing_date`.
3. Do not treat `document_filing_date` on that target as the institution decision date. Verify it against the PTAB proceeding's own docket.

4. Open the PTAB case profile from `case_cite.case_id` and read the Pleadings & Orders tab. The coded action types give an authoritative timeline: `PTAB Petition for Inter Partes Review`, `PTAB Patent Owner Preliminary Response`, `PTAB Scheduling Order`. A scheduling order marks institution, since the Board issues one when it institutes.
5. Compare to the stay order date and return the posture:
   - **Instituted.** An institution decision predates the stay order.
   - **Institution denied.** A denial predates the stay order.
   - **Pending.** No institution decision predates it. Return the petition date and the interval to the stay order.
   - **No petition.** No PTAB record at all. This is a distinct posture and a different argument, not a variety of pending.
6. Resolve per patent. A multi-patent case can be mixed. Characterize the order by whether any institution decision preceded it.

Guardrails. Measure as of the order date, never as of today; current status silently inverts the answer for older orders. Under §314(b) institution runs roughly six months from the petition, so a stay order inside that window should come back pending, and a contrary result deserves a second look. Where institution was discretionarily denied there may be no scheduling order at all, so key the timeline on the institution decision rather than on the scheduling order alone. Where the asserted patent list cannot be retrieved, report the posture as undeterminable rather than guessing.

Disclose that the posture is derived. It is a computed comparison of two dates, not a Docket Navigator classification. Say so, and say which orders you verified and which you did not.

Deliverables

Binder

Binder creation is mutating and typically cannot be undone. State what will be saved and confirm before calling `docnav_create_binder`.

Build one tab per analytical slice, with the requested breakout in its own tab. Give each tab the full criteria including library and target. After creating, call `docnav_open_binder` and `docnav_inspect_binder_tab` on at least the key tab to confirm the criteria survived the save and the count matches what you reported.

Put method notes in the binder description, since those are what get lost when the binder circulates: which join path was used, which document-type ID was specified explicitly, and the as-of date for the counts. Tabs are live saved searches and will drift as new orders are coded, so any figure quoted in a memo needs an as-of date.

Briefing

Lead with the disposition breakdown and its denominator. Then the factors the court actually weighed, taken from coded issues. Then the orders worth reading, each with a source link. Then the benchmark. Close with what you could not verify.

State coverage limits without being asked. If you verified nine of thirty-eight events, say which nine and why those.

What fails

- A rate without its denominator, or one that drops unreconciled events
- "Cases" where the unit is coded events
- Any characterization of an uncoded order's reasoning
- A quoted sentence presented as the judge's own when the judge was quoting someone else
- A quotation from a scanned or oral order presented as verified
- A trend claim from a handful of decisions a year
- Silently widening from judge to court, or from a specific motion to a category, across follow-up turns
- Inventing a taxonomy value for a state the data does not code
- Attributing arguments to counsel, or inferring brief contents, from an order's coded record
- Presenting case outcomes, party outcomes, or determinations as motion results

Scope preservation

Across follow-up turns, judge scope and document-type scope persist unless the user changes them. Before each new query, confirm both criteria are still present. Silent widening is the most common regression in a multi-turn session and is invisible in the output.

Skill frontmatter

These fields are the skill's YAML frontmatter. Everything before this section is the skill body.

name	`judge-and-court-intelligence`
description	Research how a specific judge or court has ruled on a category of motion using Docket Navigator, and turn the result into an argument-ready briefing. Use when asked how a judge handles a motion type, what their grant rate is, which orders to read before a hearing, what the court actually said in an order, how one judge compares to their district, or when preparing for oral argument on a contested motion. Also use when asked to derive a posture the data does not code, such as whether a parallel proceeding had reached a decision point as of an order date. Works across all Docket Navigator libraries.