Decile one, not ticker by ticker.

The most common quant question is the same shape every time. *”Which names are in the top decile of [some metric] today?”*

The honest answer for most APIs is also the same. *”Loop over the universe, pull the metric per ticker, sort client-side, take the top 10%.”*

That answer scales like the universe. Three thousand names, one HTTP call each. Ten metrics, three windows, three cohorts — and you’re paying for ninety thousand calls to ask one cross-sectional question.

There is a better question shape, and an API surface that matches it.

One call, one cross-section

`POST /api/rankings/screen` takes a **metric, cohort, window** triple and a **filter** (percentile, decile, or sector), and returns up to **500 rows** from the universe — already ranked, already filtered, ready to act on.

```
POST /api/rankings/screen
{
  "metric":  "subsector_residual",
  "cohort":  "subsector",
  "window":  "21d",
  "decile":  1,
  "limit":   50
}
```

That’s the entire request. The response is a sorted list of the top-decile 21-day subsector residual names — what stat-arb desks call “the long book before the carry filter.” Same call shape works for market-cap leadership inside a sector, for short-window outperformers, for any of the seven indexed metrics across any of the three cohorts and four windows.

$0.02 per call. One round-trip.

Why this shape exists

The ranking is precomputed. `ds_rankings_{etf}_{universe}.zarr` holds 96 ranking variables per (teo, symbol) — every metric × cohort × window combination already sorted and percentile-tagged by the ERM3 pipeline at 1d cadence. The screen endpoint isn't doing the math; it's doing **server-side selection** on a panel that's already ranked.

The alternative — pulling each ticker's row and ranking client-side — recomputes what the pipeline already computed, costs N times more, and asks for N times more network round-trips. The screen endpoint replaces N with 1.

What it changes

Before	After
Loop 3,000 tickers through `/rankings` to find decile 1	One call returning the 300 names in decile 1
Fetch metric column, sort in pandas, take top 5%	`min_percentile: 95, limit: 150` server-side
Subset by sector after the fact	`sector_filter: "XLK"` in the request body
Pay per-ticker for a question that's cross-sectional	Pay per **screen**

The mechanic isn't surprising once you've seen it. The product change is that the screening is **a first-class request shape on the API**, not a workflow you have to assemble.

Use cases

STAT-ARB RESIDUAL SCREEN

The classic. Subsector residual return, top decile, 21-day window. Stat-arb long book candidates.

```
from riskmodels import RiskModelsClient
client = RiskModelsClient.from_env()

longs = client.screen_rankings(
    metric="subsector_residual",
    cohort="subsector",
    window="21d",
    decile=1,           # 1 = best
    limit=100,
    as_dataframe=True,
)
```

CAP-RANK LEADERSHIP INSIDE A SECTOR

Which financials are the biggest by market cap today, ranked within their sector cohort.

```
top_xlf = client.screen_rankings(
    metric="mkt_cap",
    cohort="sector",
    window="1d",
    sector_filter="XLF",
    limit=20,
)
```

CROSS-WINDOWS MOMENTUM CHECK

Same metric, two windows, one decile. Names in top decile of 252d AND top decile of 21d are the persistent winners.

```
persistent = (
    client.screen_rankings(metric="gross_return", cohort="universe",
window="21d", decile=1, limit=300)
    .merge(
        client.screen_rankings(metric="gross_return", cohort="universe",
window="252d", decile=1, limit=300),
        on="ticker",
        suffixes=("_21d", "_252d"),
    )
)
```

Where it sits next to the rest of the stack

The rank screen is the **cross-sectional gateway**. Pair it with:

- **`POST /api/batch/lstar`** — feed the screen's ticker list into a batch Lstar history pull for the names that passed. The combination is "what's the residual-clean Lstar history for today's top-decile screen?" — two HTTP calls instead of N + N.
- **`GET /api/industry-panel`** — the macro view (industries × β) sitting alongside the micro view (tickers × rank). Cross-reference: high within-industry `beta_variance` × top-decile residual screen = "industries where stock-picking is paying off and the picks are running."
- **`POST /decompose`** — for each name that passes the screen, the four-bet hedge map is one more call away.

The principle

> Don't loop a cross-sectional question. Ask it cross-sectionally.

The ERM3 pipeline already ranks the universe daily. The screen endpoint just lets you query that rank like a database — one filter, one response, one bill.

Try the rankings screen on the API → · Full endpoint docs →