Elasticsearch Data Generator — User Manual

Overview

A browser-based tool to generate synthetic data, preview and insert into Elasticsearch, query using Elasticsearch SQL, compare index schemas, and delete by query — all with robust tables and a clean tabbed UI.

- Tabs: Connections, Schema Generator, Elasticsearch Editor (Using SQL Query), Compare Schemas, Delete By Query
- Works with local or remote Elasticsearch (HTTP)
- Tables support horizontal/vertical scroll, sticky headers, and pagination

Prerequisites

- Elasticsearch running and reachable (local example below)
- Node.js installed
- Project commands:
  - `npm run dev` — start the app
  - `npm run build` — production build

Start Local Elasticsearch (optional)

- Example command already configured in your environment:
  - `elasticsearch -E discovery.type=single-node -E xpack.security.enabled=false ...`
- Alternatively use Docker: `docker run -p 9200:9200 -e discovery.type=single-node docker.elastic.co/elasticsearch/elasticsearch:8`

Getting Started

1. In a terminal: `npm run dev`
2. Open the app in your browser (e.g., `http://localhost:5173/`)
3. Use the tab bar at the top to navigate features

 Tab Bar

---

Connections

Create and manage Elasticsearch connections. Saved connections live inside this tab.

 Connections Form

Add a Connection

- Name: any label (e.g., "Local ES")
- URL: e.g., `http://localhost:9200`
- Auth Type:
  - Basic: enter username/password
  - API Key: paste your Base64-encoded API key

- Click `Save Connection`
- Click `Add Local ES` to auto-create a local entry

Use Saved Connections

- Select a saved connection from the dropdown
- Update URL or Auth Type in-place
- `Test Connection` checks cluster health
- `Delete` removes the selected connection

Notes

- Browsers enforce TLS; for self-signed certs, trust the certificate in the OS

Schema Generator

Generate sample documents based on an index mapping and custom per-field rules, then bulk-insert.

 Schema Generator

Steps

1. Select `Index`
2. Set `Count` and `Chunk Size` for bulk inserts
3. Choose `Range Start (ISO)` and `Range End (ISO)` or use a `Range Preset`
4. Configure `Unit for Count` (second/minute/hour/day)
5. Define field rules, then click `Preview` to generate sample docs
6. Switch views: JSON/Text, JSON/Tree, or Table
7. Adjust table `Page Size` and use `Prev/Next` to paginate
8. Click `Confirm & Generate + Insert` for bulk upload with progress

Views

- JSON Text: raw JSON with `Copy` /scroll
- JSON Tree: expandable nodes with `Expand All`, `Collapse All`, and `Filter keys`
- Table: sticky headers, horizontal/vertical scroll, pagination controls

Rules — How to Apply with Cases

For each field, choose a rule based on its mapping type. Examples:

- Date (`date`)
  - Case: generate timestamps in a range
  - Steps: select `date`, set start/end ISO; preview shows ISO strings
  - Example result: `"2025-12-07T10:22:31.000Z"`
- Geo Point (`geo_point`)
  - Case: random points in bounds
  - Steps: choose `geo_point` or `geohash` or `geo_city`
  - `geo_point`: set lat/lon bounds
  - Example: `{ lat: 37.7749, lon: -122.4194 }`
- IP (`ip`)

    - Case: IPv4 or IPv6
    - Steps: choose IP version; preview displays valid addresses
    - Example: `"192.168.1.42"` or `"2001:db8::1"`
- Numeric (`integer` , `short` , `long` , `float` , `double`)
    - Case: ranges or max-bound
    - Steps: choose `num_range` with min/max, or `num_max` with upper bound
    - Example: `42` (within 0..100)
- String (`keyword` / `text`)
    - `prefix` — fixed prefix with random tail: e.g., `"ORD-98321"`
    - `phone` — generated phone-like string
    - `string_list` — pick from a provided list: e.g., `"NYC"` , `"SFO"`
    - `image_path` — local path strings for images (no upload)
- Manual (`any type`)
    - Case: force a constant value regardless of mapping
    - Steps: select `manual` , set value
    - Example: `"CONST"`

Tips

- Combine multiple rules across fields for realistic docs (dates + geo + numeric)
- Use `Compact Mode` to reduce table cell padding

Bulk Insert Progress

- Displays processed/total, succeeded/failed, and chunk progress
- Cancel with `Cancel`

Elasticsearch Editor (Using SQL Query)

Query indices/data streams/patterns with Elasticsearch SQL.

 SQL Editor

Steps

1. Source Type: `Index` , `Data Stream` , or `Pattern`
2. Pick a source value (e.g., `flights`)
3. Default SQL uses `LIMIT` from `Fetch Size` ; edit as needed
4. `Translate` to see underlying DSL (optional)
5. `Run` to execute

Results

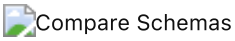
- Table view: sticky headers, pagination `Page Size` , `Prev/Next`
- JSON view: text or tree with `Filter keys` , `Expand/Collapse`
- `Download CSV` : saves the full current result set to a CSV file

Examples

- Basic: `SELECT * FROM "flights" LIMIT 50`
- Aggregation: `SELECT carrier, COUNT(*) FROM "flights" GROUP BY carrier`
- Pattern: `SELECT * FROM "logs-*" LIMIT 100`

Compare Schemas

See added, removed, and type-changed fields between two indices.

 Compare Schemas

Steps

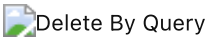
1. Choose `Old Index` and `New Index`
2. Review lists of `Added` and `Removed` fields
3. View `Type Changes` table (`Field` , `Old` , `New`)

Use Cases

- Validate a migration or reindex
- Detect breaking changes before bulk inserts

Delete By Query

Preview and execute deletion tasks by query.

 Delete By Query

Steps

1. Select `Index`
2. Set `Preview Size`
3. Enter `Query JSON` (examples available from dropdown)
4. `Preview` to view matching docs (JSON/Tree/Table with pagination)
5. `Run Delete` to start the task, track `%` progress and status

Safety Tips

- Always preview before deleting
- Start with small `Preview Size` and verify

Pagination & Table UX

- `Page Size` controls available in Preview, SQL results, and Delete Preview tables
- `Prev/Next` navigate pages
- Sticky headers keep columns visible during scroll
- Horizontal and vertical scroll keep layout within page boundaries
- Compact mode reduces padding for dense datasets

Troubleshooting

- Connection errors: verify URL and auth; use `Test Connection`
- CORS/TLS: configure server CORS or trust cert at OS level
- Empty results: check index selection and SQL `LIMIT`
- Slow tables: lower `Page Size` or tighten `LIMIT`

Screenshots

Place images under `es-data-generator/docs/images/` with these names to match the manual:

- `tabbar.png`
- `connections_form.png`
- `schema_generator.png`
- `sql_editor.png`
- `compare_schemas.png`
- `delete_by_query.png`

You can capture these directly from the running app:

- Start the app: `npm run dev`
- Open in browser and use OS screenshot tools

Appendix — Quick Reference

- Start app: `npm run dev`
- Lint: `npm run lint`
- Type check: `npm run typecheck`
- Build: `npm run build`