# QueryShield Monorepo (v1)

![CI](https://github.com/your-org/your-repo/actions/workflows/queryshield.yml/badge.svg)

Note: replace `your-org/your-repo` with your GitHub path after pushing this repo.

This repo contains:

- `queryshield-probe` (Python): Django DB instrumentation + heuristics

- `queryshield` (Python CLI): analyze, budget-check, baseline, verify-patch

- JetBrains plugin (PyCharm/IntelliJ) skeleton with inlay + tool window

- `sample-django-app`: minimal project to demo N+1 and improvements

## Quick Start

- Install: `pip install -e queryshield/probe -e queryshield/cli Django PyYAML typer rich`

- Sample app:

- `cd queryshield/sample-django-app`

- `export DJANGO\_SETTINGS\_MODULE=sample\_django\_app.settings` (Windows: `set`)

- `python manage.py migrate`

- `queryshield analyze --runner=django --output .queryshield/queryshield\_report.json`

- `queryshield budget-check --budgets ../queryshield.yml --report .queryshield/queryshield\_report.json`

## CLI

- `queryshield analyze` options:

- `--runner=django|pytest` (v1: django)

- `--explain/--no-explain` (default on for Postgres)

- `--nplus1-threshold` (default 5)

- `--explain-timeout-ms` (default 500)

- `--explain-max-plans` (default 50)

- `--output` (default `.queryshield/queryshield\_report.json`)

- `queryshield budget-check`:

- Exit codes: 0 OK, 2 violations, 3 invalid config

- `--json-out` prints a JSON line with violations

- Writes suggested index DDL to `.queryshield/ddl-suggestions.txt` (top 5)

## Report Schema (version "1")

- Top-level keys: `version`, `project\_root`, `timestamp`, `framework`, `db`, `run`, `tests`

- `tests[]`:

- `name` (string): fully qualified test name

- `duration\_ms` (float)

- `queries\_total` (int)

- `queries\_p95\_ms` (float)

- `problems[]` (list) with stable keys:

- `id` (string): stable problem id

- `type` (string): one of `N+1`, `MISSING\_INDEX`, `SORT\_WITHOUT\_INDEX`, `SELECT\_STAR\_LARGE`

- `evidence` (object): problem-specific facts (e.g., cluster\_count, relation)

- `suggestion` (object): suggested fix, `kind` and `args`

- `explain` (object|null): explain node info when applicable

- `queries[]` (truncated list): normalized SQL, durations, stack, tags

- `db.vendor`: `postgresql` or `sqlite` etc.

- `run` includes: `mode`, `budgets\_file`, `explain`, `explain\_timeout\_ms`, `explain\_max\_plans`, `nplus1\_threshold`, `duration\_ms`, `explain\_runtime\_ms`.

Schema file: see `schema/queryshield\_report\_v1.json`. Forward-compat: unknown keys are ignored by consumers.

## Budgets YAML

```

defaults:

max\_queries: 5

max\_total\_db\_time\_ms: 200

forbid:

- type: "N+1"

tests:

"package.module::test\_fn":

max\_queries: 3

ignore:

- "n+1:\*" # glob patterns (match problem id or type)

```

## Compatibility

- Python: 3.10–3.12 (CI uses 3.11)

- Django: 3.2 / 4.x / 5.x

- Postgres: 14–16 (EXPLAIN safe-mode only)

- SQLite: N+1 detection only

## Roadmap

- v1.0: Django runner + EXPLAIN heuristics, JetBrains plugin UI (inlay + tool window)

- Next: pytest runner, MySQL support, PR-bot (v1.1)

## Safety & Privacy

- Parameters are redacted (types/shapes only)

- No telemetry; no network calls

- EXPLAIN runs in safe mode (no ANALYZE) with 500ms timeout and capped 50 plans

- Set `QUERYSHIELD\_DEBUG=1` for verbose logs (to be expanded)

## CI

See `.github/workflows/queryshield.yml` for a ready-to-use GitHub Actions job. CI blocks merges when budgets fail.

Smoke scripts:

- Linux/macOS: `scripts/smoke\_sample.sh`

- Windows (PowerShell): `scripts/smoke\_sample.ps1`

## Before/After Proof (sample app, Postgres + EXPLAIN)

Commands used:

```

cd queryshield/sample-django-app

export DJANGO\_SETTINGS\_MODULE=sample\_django\_app.settings

docker compose up -d

python -m pip install -e ../probe -e ../cli psycopg2-binary

python manage.py migrate

# Before (N+1 route)

queryshield analyze --runner=django --output .queryshield/before.json

queryshield budget-check --budgets ../queryshield.yml --report .queryshield/before.json || echo "budgets failed as expected"

# After (optimized with select\_related)

queryshield analyze --runner=django --output .queryshield/after.json

queryshield budget-check --budgets ../queryshield.yml --report .queryshield/after.json

```

Numbers (filled by CI evidence commit after first run):

- Before: Queries total = <fill>, p95 DB ms = <fill>, Overhead = <fill>%

- After: Queries total = <fill>, p95 DB ms = <fill>, Overhead = <fill>%

Both reports are stored under `sample-django-app/.queryshield/`.

## 60s Getting Started (your project)

1) Install: `pip install queryshield queryshield-probe` (or editable from this repo)

2) Set env: `export DJANGO\_SETTINGS\_MODULE=yourproj.settings`

3) Create budgets `queryshield.yml`:

```

defaults:

max\_queries: 10

max\_total\_db\_time\_ms: 500

forbid: [{ type: "N+1" }]

```

4) Run: `queryshield analyze --runner=django` then `queryshield budget-check`

5) In CI: copy `.github/workflows/queryshield.yml` and tune Python/DB versions

### How budgets work

Budgets file `queryshield.yml` supports defaults, per-test overrides, forbids and ignores.

```

defaults:

max\_queries: 10

max\_total\_db\_time\_ms: 500

forbid:

- type: "N+1"

tests:

"package.module::test\_list":

max\_queries: 5

ignore:

- "explain:select\_star\_large:\*" # ignore this explain rule by id

- "SORT\_WITHOUT\_INDEX" # ignore all problems of a type

```

### Report size guard

- Max 500 queries stored per test; SQL truncated to 2KB per item. Problem list is deduplicated per normalized SQL.

### Exit codes

- 0: OK, 2: Budget violations, 3: Invalid config, 1: Runtime errors

### Timeout strategy

- Each EXPLAIN uses 500ms statement\_timeout; if several consecutive plans return None, plan collection is curtailed early; max 50 plans per run.

- Overhead capped in practice by 50 plans × ~500ms each; typical overhead on the sample app is under 20%. The CLI prints total duration, explain\_runtime\_ms, and overhead %.