QA SOP - Safe Version

Demo-safe SOP Generated 2026-02-07 19:34
Distance risk QA — runner SOP (demo-safe)
Audience: Operations associate • Last updated: YYYY-MM-DD • Generated: YYYY-MM-DD HH:MM

1. What this SOP is for

Use this SOP to create the standard Distance Risk QA Excel deliverable. You will (1) attach the period's source Excel extract and (2) attach the versioned runbook Markdown file. The runbook contains the exact rules; your job is to execute it and save the evidence artifacts.

2. Why are we doing this? (Plain-language)

We run this report to protect a GL department from using the wrong location address.

- The ERP/GL system has an address tied to an organizational unit for accounting and reporting.
- The subledger has the address that typically represents the property attributes. Even when two addresses are close together (the input is already filtered to < 1.5 miles), they can still be different buildings, parcels. If we blindly keep the ERP address, we may assign property to the wrong region, which can cause errors in reporting. This deliverable flags the cases that look "close but not truly the same" so a reviewer can quickly focus on the records most likely to need investigation.

3. Who this is written for

This is written for an operations associate. You do not need to know Python or how the calculations work. You do need to follow the steps exactly and confirm the output passes the checks.

4. Before you start (prerequisites)

Have these two files ready:

- Input source Excel (.xlsx): formatted like the approved input template (e.g., Distance_Risk_Input_Template.xlsx).
- Runbook Markdown (.md): Distance_Risk_Runbook_vX.Y.Z.md (or the current approved version). Work environment: Use a tool-enabled environment that can run Python and create .xlsx files. Important assumption: The input is expected to already be pre-filtered to records where the distance is < 1.5 miles. The process must not delete or filter out rows; it only classifies and summarizes.

5. Run steps (end-to-end)

Follow these steps in order:

- Step 1 — Start clean: Start a new run context (recommended). This keeps the run auditable.

- Step 2 — Attach both files: Attach (a) the source .xlsx and (b) the runbook .md.
- Step 3 — Paste the launcher prompt: Paste the launcher prompt exactly as written below (do not shorten it).
- Step 4 — Download the output: When the run completes, download the generated .xlsx.
- Step 5 — Archive evidence: Save the input .xlsx, the runbook .md, and the output .xlsx together in the period folder.

6. Launcher prompt (copy/paste)

Paste exactly (edit only the optional period suffix line if you want a dated filename):

```
Open and follow the attached runbook `Distance_Risk_Runbook_vX.Y.Z.md`
EXACTLY. Use Python to generate the Excel deliverable per the runbook's
Deliverable Contract, including sheets `ExceptionRegister`, `Summary_Bucket`,
and `log`. Read the attached source Excel file and return the generated
`.xlsx` as the final output artifact. (Optional) Use this output filename
suffix: (example: 2026Q1)
```

7. Acceptance criteria (pass/fail checks)

Run these checks before you send the file to reviewers:

A) Workbook structure — Hard fail if not met

- Output is a downloadable .xlsx.
- Exactly three sheets, in this order: ExceptionRegister, Summary_Bucket, log.

B) ExceptionRegister — Hard fail if not met

- Includes all source columns in original order.
- Appends computed columns with the exact names/order in the runbook.

C) Summary_Bucket — Hard fail if not met

- Exactly two columns: RiskBucket, Count.
- Contains only the approved risk labels (even if a count is zero).
- Counts reconcile to ExceptionRegister[RiskBucket].

D) Log sheet — Hard fail if not met

Log must include at minimum: timestamp, input/output filenames, row counts (must match), whether distance was provided/computed, missingness counts, constants used, bucket counts, density quantiles, and any warnings.

8. Common issues & fast fixes

- Issue 1 — The model summarizes instead of executing: Re-run and use the launcher prompt verbatim. Include the phrases “follow EXACTLY” and “use Python to generate”.
- Issue 2 — Missing/renamed columns in the source extract: Confirm the extract matches the template column names. If upstream changed, fix the extract mapping (preferred) or update the runbook with a new version.
- Issue 3 — Output row count differs from input: Hard fail. Re-run. The runbook forbids dropping rows.
- Issue 4 — Summary_Bucket does not reconcile: Hard fail. Re-run and inspect log warnings for null distance handling or missing computed fields.

9. Reviewer notes (how to use the output)

Buckets are risk indicators for QA review within the assumed < 1.5 miles population — not definitive truth. Prioritize review of higher-risk buckets first. Use DensityBand to ensure sampling spans Low/Medium/High density contexts.

10. Versioning / change control

Any change to thresholds, bucket labels, computed column names/order, or sheet structure requires a version bump and a changelog entry. Always archive the runbook version used with the period evidence.