PharmaSUG 2026 - Paper RW-429

Biostatistical Foundations 201: Privacy-Preserving Patient Linkage Across Real-World Data Sources

Anbu Damodaran, Alexion, AstraZeneca Rare Disease;

ABSTRACT

Real-world data (RWD) integration increasingly depends on linking electronic health records (EHRs), claims, and registries to build longitudinal patient trajectories[1,2]. Absent universal identifiers and under strict privacy constraints, linkage must balance privacy and accuracy. This paper consolidates deterministic and probabilistic linkage[3,4], privacy-preserving record linkage (PPRL) methods such as tokenization and Bloom filters[5,6]. and machine learning[7] approaches into a reproducible blueprint geared to statistical programmers. We detail match-quality estimation, precision–recall tuning, propagation of linkage error into downstream analyses, and post-linkage convergence diagnostics (overlap, coverage, temporal alignment, concordance). Operational guidance covers data standardization, blocking, scoring, thresholding with clerical review, continuous QA monitoring, and governance - including salts/keys, auditability, and bias assessment. We explicitly situate PPRL within HIPAA's de-identification pathways[8], emphasizing that PPRL can support Expert Determination but is not itself Safe Harbor de-identification, and provide a threat model and hardening matrix for Bloom-filter encodings. Oncology-specific considerations illustrate application in small-sample, rare-disease contexts.

INTRODUCTION

Building longitudinal patient views by linking disparate RWD sources is now central to integrated evidence generation, pharmacovigilance, and outcomes research. However, the absence of universal patient identifiers and strict privacy requirements precludes naïve joins. Statistical programmers work at the intersection of methodology and implementation. They design linkage pipelines, enforce privacy-preserving transformations, quantify match quality, and assess how linkage errors propagate into downstream analyses and causal inference. This paper presents a pragmatic, code-focused blueprint that integrates deterministic and probabilistic matching, modern tokenization and Bloom filters, and machine learning, alongside governance and reproducibility guidance. We also clarify how PPRL fits within HIPAA de-identification and provide concrete recommendations for security hardening.

REAL-WORLD DATA INTEGRATION

Understanding statistical paradoxes is not just an academic pursuit; it's a practical necessity for anyone making decisions in a data-rich, yet often misleading, world.

1. Source Characteristics

Electronic Health Records (EHRs): Rich in clinical detail (labs, vitals, notes) but idiosyncratic and often health-system bound.

Administrative Claims: Broad coverage of adjudicated, billable events across care settings with wide longitudinal visibility, but limited clinical nuance and potential gaps (e.g., out-of-network or cash-pay).

Disease Registries: Curated depth within a condition area (e.g., oncology) but narrow breadth and variable capture of non-registry events.

2. Convergence and Overlap Metrics

After linkage, quantify what converges versus diverges across sources. Track routine metrics such as patient overlap (A only, B only, and A ∩ B with a Jaccard index), coverage concordance (per-patient encounters, diagnoses, and procedures), temporal alignment (lead–lag and gaps), and concept concordance on sentinel events and variables.

3. Practical Tools

Open-source tools such as Splink (Fellegi–Sunter[9,10] with EM at scale) and Dedupe (active learning with rich similarity features) provide production-ready linkage capabilities. These tools are typically wrapped in orchestrated jobs (e.g., Spark/SQL, Airflow) with parameter registries, versioned data contracts, and automated reports.

PRIVACY-PRESERVING PATIENT LINKAGE

PPRL methods enable matching while suppressing or transforming direct identifiers, supporting minimum-necessary principles and enabling Expert Determination for HIPAA de-identification when residual risk is attested as very small by a qualified expert. PPRL is not, by itself, Safe Harbor de-identification.

1. Record Linkage Techniques

Deterministic linkage relies on exact or canonicalized agreement across robust quasi-identifiers (e.g., DOB, gender, ZIP, phonetic name tokens). Probabilistic linkage computes match weights from agreement patterns across multiple fields—often using string similarities (Jaro–Winkler, Levenshtein) and estimating m/u probabilities via EM, yielding calibrated scores and tunable thresholds for match, clerical review, and non-match.

2. Tokenization and Hash-Based Schemes

Single-party tokenization: Hash normalized attribute combinations (e.g., name+DOB+gender) with a shared secret salt; ensure strict key management and rotation.

Third-party token broker: A neutral intermediary ingests plaintext, issues blinded tokens, and destroys the raw data, reducing cross-party exposure.

Bloom filter encoding[11,12]: Break strings into q-grams and set bits in a fixed-length array via multiple hash functions; compare arrays with Dice/Jaccard to approximate string similarity without revealing raw text. Include field-specific salts and monitor bit density to reduce susceptibility to frequency/pattern-mining attacks.

3. Machine Learning and Clustering[7]

Supervised learning trains classifiers on engineered features (token agreements, edit distances, phonetic matches, geographic proximity), enabling explicit precision–recall optimization and subgroup fairness assessment. When labels are scarce, unsupervised clustering on vectorized representations with careful blocking can surface duplicates, followed by post-hoc quality checks.

BIOSTATISTICAL FOUNDATIONS

Match quality estimation starts with m/u-probability estimation (EM or supervised calibration) and continues with cross-validated precision, recall, F1, AUCPR, and calibration diagnostics. Thresholds should be use-case driven (e.g., recall-focused for cohort assembly, precision-focused for validation). Consider structural zeros induced by blocking when calibrating FS models.

Linkage error propagates into estimands and uncertainty. False merges attenuate effects and can misalign exposures and outcomes. Missed links reduce power and can induce informative missingness. Mitigate through probabilistic linkage analyses (weighting or multiple imputation over linkage draws), sensitivity analyses across thresholds, and calibration versus trusted benchmarks.

Linkage bias arises when quasi-identifier completeness/quality varies across subgroups (e.g., language, SES, mobility). Report performance by subgroup and mitigate via reweighting, subgroup-aware thresholds, and feature augmentation (e.g., encounter co-occurrence patterns).

EVALUATING CONVERGENCE AND DIVERGENCE

Overlap metrics partition patients into A-only, B-only, and A∩B with counts, proportions, and Jaccard indices; for probabilistic scores, compute posterior-weighted overlaps or present ranges across thresholds.

Coverage concordance compares per-patient counts of encounters, diagnoses, procedures, and fills across sources; use scatter plots and Bland–Altman summaries to detect systematic biases.

Temporal alignment reconstructs timelines and assesses lags and gaps; quantify the proportion of events observed within ±d days across sources and identify divergence windows.

Concordance rates for shared events (e.g., hospitalizations, surgeries) measure agreement on occurrence and metadata; compute positive/negative agreement and kappa (with prevalence caveats), and audit mismatches.

TOOLS AND PLATFORMS

Splink: Scalable Fellegi–Sunter with EM estimation, blocking, and thresholding across Python/DuckDB and big-data backends (e.g., Spark, Athena)[13].

Febrl: Classic toolkit for data cleaning and probabilistic linkage; useful for teaching and prototyping.

Dedupe: Active-learning record linkage with string similarity features and human-in-the-loop labeling[14].

recordlinkage (Python): Research-focused toolkit with indexing, comparison, and classifiers, suitable for small/medium jobs and rapid experiments[15].

Commercial platforms: Integrated tokenization/linkage within governed data flows (e.g., end-to-end RWD suites).

Custom Spark/SQL: Bespoke implementations combining blocking, similarity UDFs, EM estimation, and calibrated thresholding under orchestrated, auditable pipelines.

STATISTICAL PROGRAMMING BLUEPRINT: IMPLEMENTATION AND GOVERNANCE

A. Data Standardization & Feature Engineering

Normalize names (casefolding, Unicode normalization, nickname dictionaries), standardize addresses, clean DOBs and genders, and derive phonetic encodings and q-grams. Track provenance and quality flags for each field.

B. Blocking & Candidate Generation

Use multi-pass blocking (e.g., ZIP3+YOB+last-name initial; month-of-birth+state+phonetic last name) with canopies to boost recall; monitor block sizes, candidate inflation, and per-pass recall lift, especially in PPRL settings.

C. Similarity, Scoring, and Model Estimation

Combine exact token matches, edit distances, Jaro–Winkler, phonetic agreements, and geographic distances with field-specific m/u weighting. In PPRL with Bloom filters, compute Dice or cosine similarity between bit arrays and calibrate via labeled samples.

D. Thresholding, Clerical Review, and Calibration

Choose operating thresholds by explicit PR objectives; define a clerical review band; use adjudicated samples to recalibrate probabilities (isotonic or Platt); version thresholds by data vintage and re-estimate upon drift. Sample clerical review cases stratified by score band and key subgroups for

unbiased estimates.

E. Evaluation & Monitoring

Maintain QA dashboards with PR estimates from clerical audits (bootstrap CIs), subgroup fairness metrics, overlap/coverage/temporal concordance summaries, and drift detectors on input field distributions; use canary blocks for early warnings.

F. Error Propagation into Analysis

Incorporate probabilistic linkage via weighting or multiple imputation of link status; run sensitivity grids across thresholds and report robustness bands; compare to benchmarks where available.

G. Privacy, Security, and Compliance

Enforce least-privilege access; manage salts/keys in HSMs or managed secret stores with rotation; prefer field-specific salts; prevent BF saturation via adequate L and monitoring; document threat models, residual risks, and change control. PPRL supports Expert Determination; Safe Harbor requires removal of 18 identifiers and is a distinct pathway[16].

EXAMPLE METRICS & REPORTING TEMPLATES

Replace with program-specific numbers; maintain this structure for reproducibility and audit.

Model Performance Estimation

Metric	Value
Precision (PPV)	0.969 (95% CI: 0.963–0.974)
Recall (Sensitivity)	0.912 (95% CI: 0.901–0.922)
F0.5 (precision-weighted)	0.955
AUCPR	0.94

Overlap (A=EHR, B=Claims): A-only 28.1%; B-only 24.7%; Both 47.2%; Jaccard 0.54.
Concordance: Encounters per patient mean diff (B–A) = +0.6 (95% LOA −1.2, +2.4); Hospitalization date agreement: 86% same-day, 95% within ±3 days.

SPECIAL CONSIDERATIONS IN ONCOLOGY AND RARE DISEASE RWD

While the foundational mechanics of privacy-preserving record linkage (PPRL) apply universally across therapeutic areas, integrating data for oncology and rare disease research introduces domain-specific complexities. Building longitudinal trajectories in these spaces requires statistical programming pipelines to go beyond simple record matching, demanding rigorous clinical harmonization and temporal logic.

1. Index-Date Alignment and Temporal Anchoring Establishing a reliable clinical anchor is a primary challenge when integrating disparate RWD sources. Standardizing the index date is crucial, requiring programmers to explicitly define and harmonize triggering events, such as pathology confirmation from an EHR, the administration of a first systemic therapy, or an initial claim of diagnosis. Because these events are often recorded with varying latencies across systems, linkage algorithms must employ temporal logic to group related index events into a single clinical episode, preventing the artificial duplication of patient journeys.

2. Algorithmic Regimen Construction Accurately mapping lines of therapy (LoT) necessitates the synthesis of both structured and semi-structured medication data. When constructing treatment regimens, data pipelines must intelligently combine National Drug Codes (NDC) and Healthcare Common Procedure Coding System (HCPCS) codes from administrative claims with Medication

Administration Records (MAR) and flowsheets derived from EHRs. Because oncology treatments are subject to delays due to toxicity or scheduling, algorithms must be designed to tolerate treatment gaps by utilizing programmatically defined grace windows. Statistical programmers should parameterize these grace periods to allow for dynamic sensitivity analyses regarding regimen duration and switching logic.

3. Biomarker Concordance and Genomic Integration Precision medicine heavily relies on molecular profiling, yet laboratory data is notoriously idiosyncratic. For robust biomarker concordance, programming pipelines must develop crosswalks that map discrete laboratory values using Logical Observation Identifiers Names and Codes (LOINC) and Entity-Attribute-Value (EAV) schemas. Furthermore, data models must encode the idiosyncrasies inherent to different lab vendors and accurately reconcile tumor Next-Generation Sequencing (NGS) results alongside varying panel naming conventions. Discrepancies in biomarker status across linked sources should trigger automated QA flags for clinical review rather than automated overwrites.

4. Staging Alignment and Mortality Ascertainment Accurate survival analysis depends on precise baseline staging and reliable mortality endpoints. Staging alignment demands the rigorous reconciliation of Tumor, Node, Metastasis (TNM) staging or summary staging guidelines across disparate clinical sources. Programmers must build hierarchical rules to select the most authoritative stage when conflicts arise between registry data and unstructured EHR notes. Finally, for accurate overall survival (OS) calculations, it is necessary to clearly document and clarify the linkage methodology applied to external death data (such as the National Death Index or commercial obituaries) where it is available. Linkage error here directly impacts Kaplan-Meier estimates, making probabilistic sensitivity analyses essential.

CONCLUSION

Privacy-preserving patient linkage is now a core competency for statistical programmers working with RWD. Delivering accurate, ethical, and reproducible linkage requires a synthesis of deterministic and probabilistic methods, PPRL (tokenization and Bloom filters), and modern ML, wrapped in rigorous evaluation and governance. Precision–recall tradeoffs must be explicit and use-case tailored; convergence diagnostics and temporal alignment anchor analytic validity; and linkage bias must be measured and mitigated across subgroups. With the right tools and disciplined operations, teams can operationalize linkage at scale while maintaining HIPAA compliance and cryptographic hygiene.

REFERENCES

1. Damodaran A. Operationalizing Real World Data for External Control Arms: An End to End Framework for Rare Disease and Oncology Trials. PharmaSUG 2026, RW-435.
2. Damodaran A, Medapati S. Architecting Trust: Making Real-World Data Submission-Ready for Rare Disease and Oncology Development. PhUSE Single Day Event (SDE); Mar 05, 2026; Radnor, PA.
3. Murray JS. Probabilistic record linkage and deduplication after indexing, blocking, and filtering. Journal of Privacy and Confidentiality. 2015;7. doi:10.29012/jpc.v7i1.643
4. Fortini M. An improved Fellegi-Sunter framework for probabilistic record linkage between large data sets. Journal of Official Statistics. 2020;36:803-825. doi:10.2478/jos-2020-0039
5. Schnell R, Bachteler T, Reiher J. Privacy-preserving record linkage using Bloom filters. BMC Med Inform Decis Mak. 2009;9:41. doi:10.1186/1472-6947-9-41
6. Niedermeyer F, Steinmetzer S, Kroll M, Schnell R. Cryptanalysis of basic Bloom filters used for privacy-preserving record linkage. Journal of Privacy and Confidentiality. 2014;6. doi:10.29012/jpc.v6i2.640
7. Damodaran A. Advanced Data Analytics in Rare Disease Clinical Trials Using R and Python. PhUSE Single Day Event (SDE); July 23, 2025; West Windsor, NJ.
8. US Department of Health and Human Services, Office for Civil Rights. Guidance regarding methods

for de-identification of protected health information in accordance with the HIPAA Privacy Rule. Accessed March 21, 2026. https://www.hhs.gov/hipaa/for-professionals/special-topics/de-identification/index.html

9. Fellegi IP, Sunter AB. A theory for record linkage. J Am Stat Assoc. 1969;64(328):1183-1210. doi:10.1080/01621459.1969.10501049
10. Winkler WE. Using the EM algorithm for weight computation in the Fellegi–Sunter model. US Census Bureau Research Report Series; 2000. Accessed March 21, 2026. https://www.census.gov/srd/papers/pdf/rr2000-05.pdf
11. Ranbaduge T, Schnell R. Securing Bloom filters for PPRL. In: Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20). 2020. doi:10.1145/3340531.3412105
12. Chen Y, Schnell R, Armknecht F, Heng Y. Salting as a countermeasure against attacks on privacy preserving record linkage techniques. In: Proceedings of the 15th International Joint Conference on Biomedical Engineering Systems and Technologies. 2022. doi:10.5220/0010787200003123
13. Ministry of Justice Analytical Services. Splink (v4) [software]. Version 4. Accessed March 21, 2026. https://moj-analytical-services.github.io/splink/
14. dedupe.io. Dedupe [software]. Accessed March 21, 2026. https://github.com/dedupeio/dedupe
15. Record Linkage Toolkit. recordlinkage [Python package]. Accessed March 21, 2026. https://pypi.org/project/recordlinkage/
16. Electronic Code of Federal Regulations. 45 CFR §164.514—Other requirements relating to uses and disclosures of protected health information. Accessed March 21, 2026. https://www.ecfr.gov/current/title-45/subtitle-A/subchapter-C/part-164/subpart-E/section-164.514

DISCLAIMER

This paper is intended solely for informational purposes and should not be interpreted as professional advice or guidance. The views and opinions expressed are those of the author and do not necessarily represent the views or policies of AstraZeneca.

AI ASSISTANCE DISCLOSURE

Microsoft Copilot was used for non-substantive editorial assistance (grammar, sentence refinement, and clarity). The author reviewed and approved the final wording. All interpretations, conclusions, and any remaining errors are the author's responsibility.

APPENDIX A: SIMILARITY FUNCTIONS (AT A GLANCE)

Names: Jaro–Winkler (prefix-sensitive), Levenshtein, Damerau–Levenshtein, Double Metaphone agreement.

Addresses: token Jaccard, USPS-standardized equality, ZIP/centroid distance.

Dates: exact/full, year-of-birth match, month–year match, absolute day difference.

APPENDIX B: PORTABLE SQL FOR MULTI-PASS BLOCKING (SKETCH)

```
-- Pass 1: strict block on YOB + first letter of last name + ZIP3

CREATE TABLE candidates_p1 AS

SELECT l.person_id AS person_id_l, r.person_id AS person_id_r
```

```
FROM ehr l
JOIN claims r
  ON l.yob = r.yob
AND substr(l.last_name_norm,1,1) = substr(r.last_name_norm,1,1)
AND l.zip3 = r.zip3;

-- Pass 2: looser block on month of birth + state + metaphone(last name)
-- Note: metaphone_last should be precomputed or provided via a UDF for your SQL engine.
CREATE TABLE candidates_p2 AS
SELECT l.person_id AS person_id_l, r.person_id AS person_id_r
FROM ehr l
JOIN claims r
 ON l.month_of_birth = r.month_of_birth
AND l.state = r.state
AND l.metaphone_last = r.metaphone_last;

-- Union distinct candidate pairs across passes
CREATE TABLE candidates AS
SELECT DISTINCT person_id_l, person_id_r FROM candidates_p1
UNION
SELECT DISTINCT person_id_l, person_id_r FROM candidates_p2;
```

APPENDIX C: BLOOM FILTER SKETCH (PSEUDOCODE)

```
def bloom_encode(s: str, q=2, k=6, L=2048, salt=b'secret_field_specific'):
bits = [0] * L
grams = [s[i:i+q] for i in range(len(s)-q+1)]
for g in grams:
 for j in range(k):
     h = hmac_sha256(salt + g.encode() + bytes([j]))
     idx = int.from_bytes(h[:4], 'big') % L
     bits[idx] = 1
return bits
def dice_sim(a, b):
inter = sum(1 for i in range(len(a)) if a[i] and b[i])
sa = sum(a); sb = sum(b)
return 2*inter / (sa + sb + 1e-9)
```

Defaults: q=2–3, k=6, L=2048 with target bit density ~15–35%.

Use field-specific salts stored in HSM/secret store; rotate periodically.

APPENDIX D: THREAT MODEL & HARDENING MATRIX

Adversaries: honest-but-curious counterparties, semi-honest broker, external attacker; potential collusion between parties.

Attack surfaces: frequency/pattern mining against BFs; dictionary attacks; broker compromise; key disclosure; BF saturation.

Hardening measures: field-specific HMAC salts; adequate L (≥1024–2048) and monitored bit density; k=4–8; q=2–3; optional perturbations (sliding-window/XOR); per-deployment pepper; strict key custody (HSM/managed secrets, rotation); access logging and anomaly detection; canary comparisons; minimize data retention at the broker.

APPENDIX E: REPRODUCIBILITY & QA CHECKLIST

• Versioned data contracts and parameter registry (blocking keys, thresholds, salts, L/q/k).

• Blocking diagnostics: block size distributions, candidate inflation, per-pass recall lift.

• Clerical audit plan: stratified sampling by score band and key subgroups; bootstrap CIs for PR metrics.

• Drift monitoring: input field distributions, q-gram frequencies, BF bit density, score distributions.

• Governance: change control, key rotation logs, security attestations (e.g., SOC 2 for brokers).

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Anbu Damodaran
Alexion, AstraZeneca Rare Disease
anbu.damodaran@alexion.com