DATA QUALITY PIPELINE: A PRACTICAL FRAMEWORK

Teaching Systematic Approaches to the GIGO Principle

Author: Sai Shashank Janke
Course: INFO 7390 - Advanced Data Science and Architecture
Institution: Northeastern University
December 2025

1. ABSTRACT

Reliable analysis starts with trustworthy data. When datasets contain errors, missing information, or formatting inconsistencies, even sophisticated analytical methods produce misleading results. This tutorial develops systematic approaches for identifying and correcting quality problems before they compromise decision-making.

Working through these materials, learners gain practical experience detecting six common problem categories: incomplete records, duplicate entries, implausible values, inconsistent formats, constraint violations, and non-uniform representations. Each receives dedicated coverage through detection principles, visualization approaches, and correction techniques.

The framework applies controlled defects to clean datasets, letting learners observe exactly what breaks and verify their corrections work properly. Methods demonstrated here transfer directly to healthcare records, financial transactions, customer databases, and sensor measurements. Students finishing this work will systematically scan datasets for problems, build automated detection tools, select appropriate remediation approaches, and measure whether corrections actually improved quality.

2. INTRODUCTION: UNDERSTANDING DATA QUALITY

2.1 Why Quality Assessment Matters

Analysis quality cannot exceed input quality. A hospital developed patient readmission predictions achieving 92% test accuracy but failing catastrophically in production. Investigation revealed training data contained age typos showing patients as 200 years old and duplicate records inflating certain patient populations. The model learned patterns from corrupted inputs, producing unreliable predictions despite sound statistical methods.

Financial forecasting at a major retailer predicted 500% demand increases. The actual cause involved negative prices in transaction logs from incorrectly coded refunds, duplicate

order entries, and inconsistent date formats causing temporal misalignment. Revenue projections based on this data led to massive overordering and significant losses.

An e-commerce recommendation engine repeatedly suggested products customers already owned. Product identifiers appeared inconsistently—"PROD-123", "prod123", and " PROD-123 " treated as different items despite representing identical products. Revenue declined while complaints increased, all from unconsidered formatting variations.

These failures share common characteristics. Technical implementations were correct. Statistical approaches were appropriate. Problems originated in data appearing acceptable at first glance but containing subtle defects breaking downstream work.

2.2 The Six Quality Dimensions

Quality assessment addresses six distinct problem categories:

Completeness examines whether required information exists. Missing values might indicate sensor failures, customer choices not to respond, or processing errors. A blank medical test result could mean the test wasn't ordered (clinically meaningful) or results weren't recorded (data problem). Different causes require different responses.

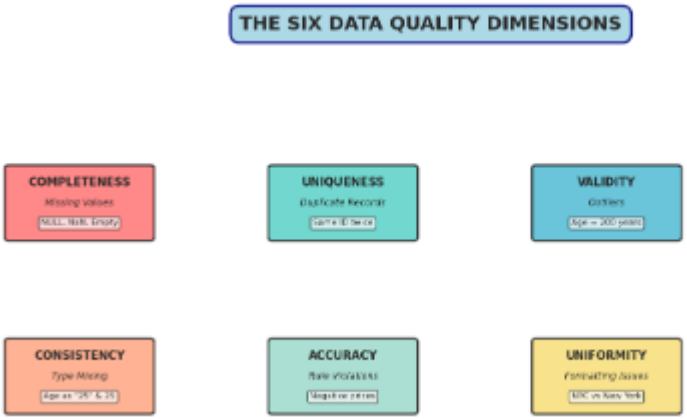
Uniqueness verifies each real-world entity appears once. Duplicates inflate counts, bias statistics toward replicated characteristics, and break relationships with other datasets. Exact duplicates typically result from data entry errors. Near-duplicates—records differing slightly—might represent the same entity with variations or genuinely different entities sharing attributes.

Validity checks whether values fall within reasonable ranges. An age of 150 years is impossible. An age of 95 is unusual but possible. A negative price is invalid for purchases but valid for refunds needing separate handling. Context determines what constitutes valid versus invalid values beyond pure statistical criteria.

Consistency ensures uniform representations throughout datasets. Mixed data types—storing some ages as numbers, others as text—prevent mathematical operations. Consistency also requires uniform units, precision, and encoding. Temperature data mixing Celsius and Fahrenheit produces meaningless statistics without conversion.

Accuracy validates compliance with business rules and logical constraints. If graduation dates precede birth dates, at least one field contains errors. If order totals don't equal quantity times unit price, calculations failed somewhere. These checks require domain knowledge about what relationships should hold.

Uniformity standardizes text formatting enabling reliable matching and aggregation. City names appearing as "New York", "new york", "NYC", or " New York " represent identical locations but won't match in standard comparisons. Dates, identifiers, and product codes need consistent formatting or analysis fragments related records.



2.3 Tutorial Approach

This framework uses controlled corruption of clean datasets. Starting with the Titanic passenger dataset, we systematically introduce defects mirroring real-world problems. Learners see exactly what breaks, implement detection methods, apply corrections, and verify improvements. This controlled approach builds understanding unavailable when encountering organic messiness.

The progression follows diagnostic workflow: detect problems, quantify severity, implement corrections, validate improvements. Each dimension receives coverage through conceptual explanation, detection methodology, correction strategies, and hands-on practice. Students build reusable tools rather than memorizing specific fixes.

3. THEORETICAL FOUNDATIONS

3.1 Completeness: Missing Data Mechanisms

Missing data isn't random. Three mechanisms create different analytical challenges:

Missing Completely At Random (MCAR) occurs when missingness has no relationship to observed or unobserved variables. Sensor failures happening randomly across all conditions create MCAR patterns. Statistical analyses remain unbiased when data is MCAR, though power decreases from reduced sample size. Imputation is safe because missing values don't differ systematically from observed ones.

Missing At Random (MAR) describes situations where missingness depends on observed variables but not the missing value itself. Younger patients more likely to skip optional survey questions creates MAR if missingness relates to age (observed) but not the answer they would have provided (unobserved). Imputation can work if models account for the observed variables predicting missingness.

Missing Not At Random (MNAR) means missingness correlates with unobserved values. High-income individuals declining to report salary creates MNAR—missingness directly relates to what the missing value would reveal. Simple imputation introduces bias because

missing values systematically differ from observed ones. MNAR requires sophisticated handling acknowledging the selection process creating missingness.

Distinguishing these mechanisms guides appropriate responses. MCAR permits straightforward imputation. MAR requires model-based approaches accounting for observed predictors of missingness. MNAR demands acknowledging that imputation may introduce systematic bias.

3.2 Uniqueness: Duplicate Detection Approaches

Exact duplicates share identical values across all fields. Hash-based detection efficiently identifies perfect matches by comparing cryptographic hashes rather than field-by-field comparison. This approach scales well to large datasets but misses near-duplicates differing in minor ways.

Near-duplicates present greater challenges. Records might represent the same entity with slight variations—typos, abbreviations, formatting differences—or genuinely different entities with similar characteristics. Probabilistic matching techniques assign similarity scores based on how many fields match and how distinctive those matches are. Two records both named "John Smith" in New York provides less evidence of duplication than two records sharing an uncommon name plus identical address.

Deduplication thresholds balance false positives against false negatives. Conservative thresholds minimize incorrectly merging distinct entities but miss legitimate duplicates. Aggressive thresholds catch more duplicates but risk combining separate entities. Domain knowledge guides appropriate threshold selection based on costs of each error type.

3.3 Validity: Statistical Outlier Detection

Interquartile Range (IQR) methods define outliers as values beyond Q_1 - 1.5×IQR or above Q_3 + 1.5×IQR, where Q_1 and Q_3 represent first and third quartiles. This approach handles skewed distributions better than methods assuming normality. The 1.5 multiplier balances sensitivity against specificity—smaller values flag more potential outliers, larger values only catch extreme cases.

Z-score approaches flag values more than three standard deviations from the mean. This works well for normally distributed data but performs poorly with skewed distributions where standard deviation poorly represents spread. Outliers themselves can distort means and standard deviations, creating recursive problems where outliers make other values appear more normal.

Domain knowledge must override purely statistical definitions. A salary of $200,000 might be a statistical outlier in a dataset of entry-level positions but represents valid data requiring retention. An age of 200 years is impossible regardless of statistical criteria. Context determines whether unusual values represent errors requiring correction or important observations demanding careful handling.

3.4 Consistency: Type System Validation

Mixed types emerge when importing from systems using different schemas or concatenating datasets collected under varying standards. Pandas may store numbers as strings if any non-numeric values appear in a column, preventing mathematical operations without explicit conversion.

Type checking requires examining actual values beyond declared types. A column might show type "object" (Pandas' string designation) but contain mostly numbers plus a few text values. Automated conversion using `pd.to_numeric()` with `errors='coerce'` transforms valid numbers while marking unconvertible values as missing. This approach reveals how many values truly are numbers versus text.

Precision and unit consistency matter beyond type matching. Storing some measurements in meters, others in feet creates problems even if all values are numeric. Mixing 32-bit and 64-bit floating point representations can introduce rounding discrepancies. Scientific notation versus fixed decimal notation affects string-based operations even when numeric operations work correctly.

3.5 Accuracy: Business Rule Validation

Business rules encode domain knowledge that statistical methods cannot capture. Prices must be non-negative unless representing refunds requiring separate transaction types. Admission dates must precede discharge dates. Age must increase monotonically over time in longitudinal data. Product quantities must be positive integers, not negative or fractional values.

Rule complexity varies. Simple range checks validate individual fields. Relational constraints verify consistency across fields—end dates after start dates, totals equaling sums of components. Temporal rules ensure logical sequencing in time-series data. Cross-table rules validate referential integrity when datasets reference each other.

Violations indicate either data errors or rule exceptions requiring special handling. A negative quantity might represent returns needing different processing than purchases. A future-dated transaction might represent scheduled payments rather than errors. Investigation determines whether violations indicate problems requiring correction or valid exceptions needing separate treatment.

3.6 Uniformity: Text Standardization

Text variations fragment logically identical values. Case differences create separate categories—"Boston" differs from "boston" in string matching. Whitespace—leading spaces, trailing spaces, multiple internal spaces—causes matching failures. Abbreviations introduce inconsistency—"Street" versus "St." versus "street".

Standardization approaches vary by field type. For proper nouns (cities, names), title case often works: capitalize first letters, lowercase the rest. For codes and identifiers, uppercase or lowercase consistently. For free-text fields, consider preserving original entries while creating standardized versions for matching.

Accent handling and character encoding add complexity in international datasets. "José" might appear as "Jose", "JosÃ©", or other variants depending on encoding. Unicode normalization converts different representations of accented characters to standard forms. This matters for names, addresses, and any text potentially containing non-ASCII characters.

4. ASSESSMENT METHODOLOGY

4.1 Systematic Detection Framework

Quality assessment follows structured workflow: scan each dimension, quantify problems found, visualize patterns, document findings. This systematic approach catches problems manual inspection misses.

Completeness Assessment counts missing values per column and calculates percentages of total records. Visualization through heatmaps reveals whether missing values cluster in specific rows (suggesting systematic collection problems) or distribute randomly. Bar charts comparing missing percentages across columns identify which fields need attention.

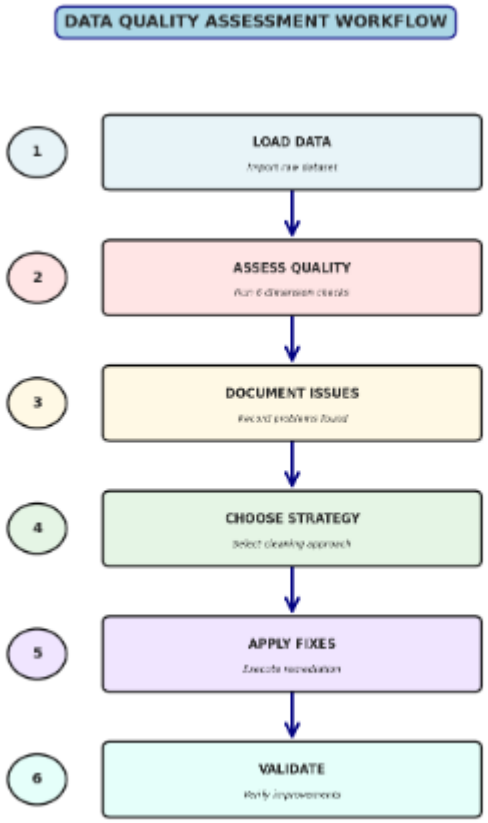
Uniqueness Detection identifies exact duplicates through hash comparison or row-by-row matching. Grouping duplicates shows how many times specific records appear. Examining which columns vary among duplicates reveals partial duplication—same customer with different addresses might represent moves rather than errors.

Validity Checking applies statistical methods identifying outliers plus domain rules flagging impossible values. Box plots visualize distributions and outliers. Scatter plots reveal relationships between variables helping distinguish unusual but valid combinations from impossible ones.

Consistency Validation examines declared types against actual values. Attempting type conversion reveals how many values successfully convert versus throwing errors. Summary statistics on supposedly numeric columns containing text values highlight mixed-type problems.

Accuracy Verification applies business rules checking logical constraints. Range checking ensures values fall within permitted bounds. Relationship checking validates that calculated fields match their components. Temporal checking ensures proper sequencing in time-ordered data.

Uniformity Inspection counts unique values in categorical fields. Unexpectedly high counts suggest formatting variations. Grouping by lowercase or whitespace-stripped versions reveals whether variations represent distinct categories or formatting inconsistencies.



4.2 Interpretation Guidelines

Raw counts and percentages require interpretation through domain lenses. Five percent missing values might be acceptable for exploratory analysis but inadequate for regulatory reporting requiring complete records. Duplicate rates below two percent often represent data entry errors, while rates above ten percent suggest systemic collection or processing problems.

Outlier interpretation depends on domain knowledge and analytical goals. For descriptive statistics summarizing typical cases, excluding extreme values makes sense. For fraud detection or rare event prediction, outliers may be precisely what matters. Context guides whether outliers require removal, investigation, or special handling.

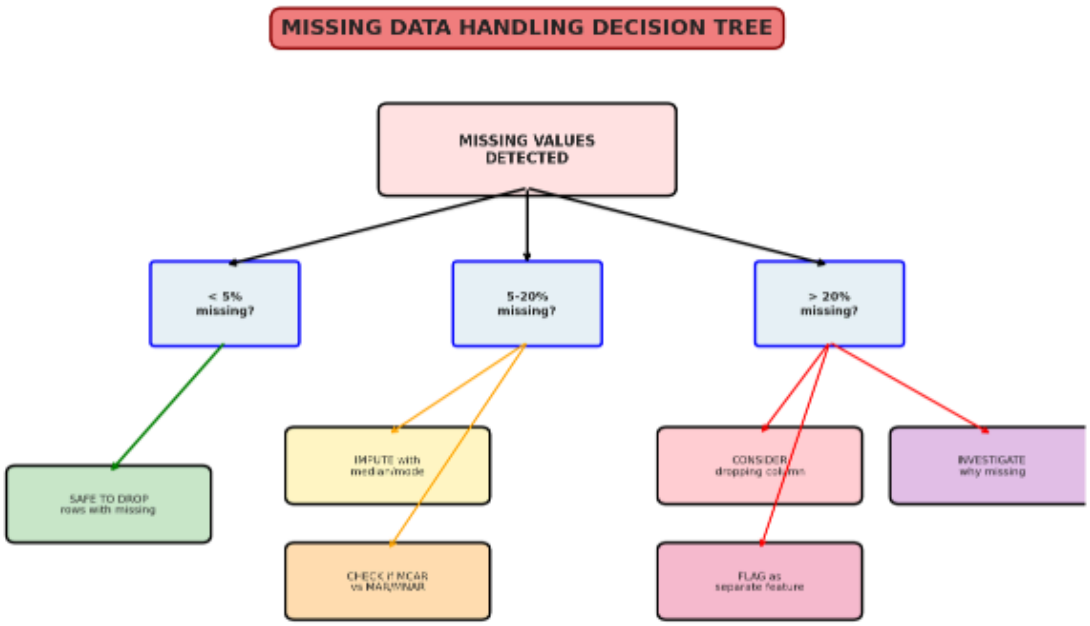
Business rule violations require investigation before automatic correction. Some violations represent errors—future dates in historical data clearly wrong. Others represent valid exceptions—zero prices for promotional items, negative quantities for returns. Investigation distinguishes errors requiring fixes from valid cases needing appropriate handling.

4.3 Documentation Requirements

Assessment findings require documentation enabling informed cleaning decisions. Record which checks were run, what problems were found, and how severe each problem appears. This documentation guides prioritization—fix critical problems first, defer minor issues if they don't affect analysis.

Quantitative summaries provide objective measurement. Report exact counts and percentages rather than vague descriptions like "lots of missing data." Compare quality metrics across different periods or data sources revealing whether quality is improving or degrading over time.

Visual documentation communicates findings to stakeholders who may not understand technical details. Annotated charts showing where problems cluster, before-after comparisons demonstrating improvement, and trend lines tracking quality metrics over time make quality concrete rather than abstract.



5. REMEDIATION STRATEGIES

5.1 Missing Value Handling

Deletion removes incomplete records or columns with excessive missingness. Row deletion works when missing data represents small fraction of total records and is MCAR. Column deletion makes sense when fields are missing for most records and aren't essential for analysis. Deletion is simple but permanently loses information.

Imputation fills missing values with estimates based on observed data. Mean or median imputation replaces missing values with central tendency measures. This preserves sample size but reduces variance and can bias relationships between variables. Forward or backward fill uses adjacent values in time-series data, appropriate when values change gradually. Model-based imputation predicts missing values from other variables, capturing relationships but requiring additional modeling.

Flagging creates indicator variables showing which values were missing. This preserves the signal that something was missing while allowing analysis to proceed. Helpful when

missingness itself carries meaning—patients declining certain tests might differ systematically from those completing all tests.

Strategy selection considers why data is missing, how much is missing, and what analysis requires. Less than five percent missing often permits deletion. Between five and twenty percent suggests imputation. Above twenty percent requires careful consideration of whether the field should be used at all.

5.2 Duplicate Resolution

Removal deletes duplicate records keeping only one instance. Choosing which instance to keep matters. "First" keeps the original, assuming the first occurrence is most likely correct. "Last" keeps the most recent, assuming later entries incorporate corrections. Either approach works for exact duplicates resulting from processing errors.

Merging combines information from duplicates when they contain complementary information. If one duplicate has address information and another has phone numbers, merging creates complete records. This requires logic determining which fields to preserve when both duplicates contain conflicting information for the same field.

Investigation examines duplicates manually when automated rules cannot confidently determine whether records represent errors or legitimate repeated observations. Same customer placing two orders on the same day might be duplicate entry or valid repeated purchase. Domain knowledge guides investigation.

5.3 Outlier Management

Capping limits extreme values to specified thresholds. Setting maximum at 99th percentile prevents extreme values from dominating calculations while retaining most data. Capping preserves record count but changes value distributions.

Removal deletes records with outlying values. Appropriate when outliers represent impossible values—negative ages, future dates in historical data—clearly indicating errors. Removal risks losing important information when outliers represent rare but valid observations.

Transformation applies mathematical operations reducing outlier influence. Logarithmic transformation compresses large values more than small ones, useful for right-skewed distributions. Square root transformation provides intermediate compression. Transformations preserve all data but change interpretation of results.

Segmentation analyzes outliers separately from main data. Creates distinct models or summaries for extreme cases rather than forcing one model to fit all data. Appropriate when outliers represent genuinely different populations requiring separate treatment.

5.4 Type Correction

Conversion transforms values to appropriate types. `pd.to_numeric()` converts strings to numbers where possible. Date parsing converts string representations to datetime objects

enabling temporal operations. Boolean conversion maps text values like "Yes"/"No" to True/False.

The `errors='coerce'` parameter determines how conversion handles unconvertible values. 'Coerce' marks them as missing, revealing how many values truly cannot convert. 'Ignore' leaves unconvertible values as original type, creating mixed-type columns. 'Raise' stops execution on first error, useful for strict validation.

After conversion, examine which values became missing to understand what was unconvertible. A column supposedly containing ages but with many values converting to missing suggests text values like "unknown" or "N/A" requiring different handling than numeric ages.

5.5 Rule Enforcement

Filtering removes records violating business rules. Straightforward when violations clearly indicate errors—ages exceeding human lifespan, prices exceeding any reasonable amount. Less clear when violations might represent valid exceptions requiring investigation.

Correction fixes violations when appropriate values can be inferred. Extra zeros in prices—$100,000 instead of $1,000—might be correctable by dividing by appropriate factors. Date format errors might be fixable by trying multiple parsing approaches. Correction requires confidence about what the correct value should be.

Flagging marks violations for review without automatic correction. Creates columns indicating which records violated which rules, enabling separate analysis of problematic data while preserving original values. Appropriate when violations need human judgment about whether they're errors or valid exceptions.

6. PRACTICAL APPLICATIONS

6.1 Healthcare Data Considerations

Medical records present unique challenges. Missing test results might indicate tests weren't ordered (clinical decision) rather than data loss (quality problem). Treating all missing values identically through imputation could hide important patterns in what gets tested.

Vital sign outliers require careful interpretation. Blood pressure of 180 is concerning but possible in hypertensive crisis. Blood pressure of 300 is physically impossible, clearly indicating sensor or entry error. Domain expertise distinguishes unusual but valid measurements from impossible ones requiring correction.

Duplicate patient records create serious problems. Same patient appearing multiple times inflates disease prevalence estimates and can lead to duplicate treatments. Record linkage in healthcare requires careful handling of protected information, making automated deduplication challenging without proper patient identifiers.

6.2 E-commerce Transaction Data

Transaction data brings specific quality issues. Negative prices might represent refunds requiring separate categorization from purchases. Zero prices could indicate promotions, data errors, or internal transfers. Understanding transaction context determines appropriate handling.

Duplicate orders need investigation distinguishing legitimate repeat purchases from processing errors. Same customer ordering the same product twice on the same day might represent error or valid behavior. Different delivery addresses suggest separate legitimate orders. Identical addresses increase likelihood of duplication error.

Price-quantity relationships enable validation. Total price should equal quantity times unit price. Violations indicate calculation errors, data entry mistakes, or applied discounts not properly recorded. Cross-field validation catches errors pure range checking misses.

6.3 Sensor Data Quality

Time-series sensor data introduces temporal dependencies. Missing values creating gaps in continuous monitoring need interpolation preserving temporal continuity rather than simple mean imputation ignoring time structure.

Sensor drift causes gradual calibration errors. Values slowly diverge from true measurements over time. Detecting drift requires comparing against expected baselines or other sensors measuring the same phenomena. Correction requires understanding drift characteristics—linear, exponential, or more complex patterns.

Stuck sensors read identical values repeatedly when hardware fails. Detecting stuck sensors requires checking variance over sliding windows. Zero or near-zero variance indicates problems. Correction might involve carrying last good value forward until sensor recovers or marking the period as unreliable.

6.4 Domain-Specific Adaptation

Quality frameworks must adapt to domain constraints. Healthcare demands high accuracy since treatment depends on data reliability. Financial services require precise timestamps for transaction sequencing and regulatory compliance. Scientific research needs extensive documentation of collection methods and quality checks for reproducibility.

Risk tolerance varies across domains. Removing twenty percent of records might be acceptable in marketing analysis where approximate patterns suffice but unacceptable in medical diagnosis where complete information matters. Domain expertise guides acceptable tradeoffs between quality and retention.

Business context determines which quality dimensions matter most. E-commerce prioritizes uniqueness—duplicates directly inflate revenue metrics. Healthcare prioritizes accuracy—impossible values could harm patients. Sensor systems prioritize validity—values outside physical bounds indicate equipment failure.

7. EXERCISES AND ASSESSMENT

7.1 Comprehension Check

Conceptual Questions:

1. Explain why missing values in medical test results might carry meaningful information rather than represent pure data quality problems.
2. Describe a scenario where statistical outliers should be retained rather than removed. What distinguishes this case from situations where outlier removal is appropriate?
3. Why does median imputation introduce less bias than mean imputation for skewed distributions? Use a concrete example to illustrate.
4. Under what circumstances would you choose to delete duplicate records versus investigating them further? What factors guide this decision?

Application Problems:

1. You discover a customer database where 15% of email addresses are missing. The missing values concentrate in records created before 2020. What missing data mechanism does this suggest? What implications does this have for imputation versus deletion?
2. A sales dataset contains negative prices in 2% of records. Describe your investigation process for determining whether these represent refunds requiring separate handling or data entry errors requiring correction.
3. An age column contains mix of numbers and text values like "unknown" and "declined". Outline your approach for handling this mixed-type situation, including validation of your solution.

7.2 Hands-On Practice

Dataset: Employee HR records with introduced quality issues including missing salaries, duplicate employee entries, impossible ages, mixed salary formats, and inconsistent department naming.

Tasks:

1. **Assessment Phase:** Run systematic quality checks across all six dimensions. Document what problems exist, their severity, and which areas need priority attention.
2. **Strategy Selection:** For each identified problem, choose an appropriate remediation approach. Justify your choices based on problem characteristics and business requirements.
3. **Implementation:** Apply your chosen strategies to clean the dataset. Document what you changed and why.
4. **Validation:** Compare cleaned data against original. Calculate quality scores before and after. Verify that cleaning improved quality without introducing new problems or changing data meaning.
5. **Business Impact:** Calculate how quality issues affected business metrics. For example, how did duplicate employee records inflate reported headcount? How did missing salary data bias compensation analysis?

7.3 Critical Thinking Extension

Scenario: You're analyzing customer retention for a subscription service. The dataset shows some customers with negative account ages (created_date after current_date), duplicate customer IDs with different subscription tiers, and missing payment information for 30% of active customers.

Develop a quality assessment and remediation plan addressing:

- Which problems require immediate correction versus investigation
- How to handle the 30% missing payment data given regulatory requirements for complete financial records
- Whether duplicate customer IDs with different subscription tiers represent errors or legitimate family accounts
- How to validate that your cleaning preserved business-relevant patterns

This scenario requires balancing technical quality concerns against business requirements and regulatory constraints, mirroring real-world complexity.

8. CONCLUSION

8.1 Core Principles

Data quality determines analytical reliability. Sophisticated methods applied to flawed inputs produce misleading outputs regardless of technical correctness. Systematic assessment across completeness, uniqueness, validity, consistency, accuracy, and uniformity dimensions catches problems manual inspection misses.

Quality work requires judgment beyond mechanical rule application. Context guides whether missing values need imputation or carry meaningful signal. Domain knowledge determines whether outliers represent errors or important observations. Business requirements define acceptable tradeoffs between quality and data retention.

Documentation enables reproducibility and stakeholder communication. Recording what problems existed, which strategies were applied, and why specific decisions made sense allows future validation and debugging. Quantitative metrics provide objective quality measurement enabling tracking over time.

8.2 Professional Practice

Production systems require continuous quality monitoring rather than one-time cleaning. Data quality degrades as source systems change, business processes evolve, and new edge cases emerge. Automated pipelines flag anomalies, track metrics over time, and alert when quality drops below acceptable thresholds.

Different domains impose specific requirements. Healthcare demands strict accuracy for patient safety. Financial services require precise timestamps for regulatory compliance.

Scientific research needs extensive documentation for reproducibility. Generic approaches must adapt to domain-specific constraints.

Quality assessment represents substantial effort in professional data science despite receiving minimal academic attention. Industry surveys consistently find cleaning consuming 60-80% of analyst time. Skills developed here—systematic assessment, strategic cleaning, rigorous validation—prove essential throughout data science careers.

8.3 Continuing Development

This tutorial provides foundational frameworks. Advanced topics worth exploring include automated quality monitoring in production systems, machine learning approaches for anomaly detection, quality considerations in streaming data, and sophisticated missing data techniques beyond simple imputation.

Quality expertise opens professional opportunities. Every analytics team, ML engineering group, and data platform struggles with messy data. Candidates demonstrating systematic quality assessment, documented decision-making, and validated improvement become immediately valuable. These skills transfer across industries and domains.

Most importantly, quality work protects real people from real consequences. Medical decisions, financial forecasts, and policy choices depend on reliable data. Attention to quality prevents flawed inputs from becoming flawed decisions affecting lives. The technical skills matter, but their ultimate value lies in enabling better decisions through trustworthy information.

9. REFERENCES

[1] McCallum, Q. E. (2012). Bad Data Handbook: Cleaning Up The Data So You Can Get Back To Work. O'Reilly Media.

[2] Olson, J. E. (2003). Data Quality: The Accuracy Dimension. Morgan Kaufmann Publishers.

[3] McKinney, W. (2022). Python for Data Analysis (3rd ed.). O'Reilly Media.

[4] Wang, R. Y., & Strong, D. M. (1996). Beyond Accuracy: What Data Quality Means to Data Consumers. Journal of Management Information Systems, 12(4), 5-33.

[5] Rahm, E., & Do, H. H. (2000). Data Cleaning: Problems and Current Approaches. IEEE Data Engineering Bulletin, 23(4), 3-13.

[6] Batini, C., Cappiello, C., Francalanci, C., & Maurino, A. (2009). Methodologies for Data Quality Assessment and Improvement. ACM Computing Surveys, 41(3), Article 16.

[7] Redman, T. C. (1998). The Impact of Poor Data Quality on the Typical Enterprise. Communications of the ACM, 41(2), 79-82.

[8] Kim, W., Choi, B. J., Hong, E. K., Kim, S. K., & Lee, D. (2003). A Taxonomy of Dirty Data. Data Mining and Knowledge Discovery, 7(1), 81-99.

[9] Müller, H., & Freytag, J. C. (2005). Problems, Methods, and Challenges in Comprehensive Data Cleansing. Technical Report HUB-IB-164, Humboldt-Universität zu Berlin.

[10] Hellerstein, J. M. (2008). Quantitative Data Cleaning for Large Databases. United Nations Economic Commission for Europe.

[11] Van den Broeck, J., Cunningham, S. A., Eeckels, R., & Herbst, K. (2005). Data Cleaning: Detecting, Diagnosing, and Editing Data Abnormalities. PLOS Medicine, 2(10), e267.

[12] Kandel, S., Paepcke, A., Hellerstein, J., & Heer, J. (2011). Wrangler: Interactive Visual Specification of Data Transformation Scripts. Proceedings of CHI 2011.

[13] Sculley, D., et al. (2015). Hidden Technical Debt in Machine Learning Systems. Advances in Neural Information Processing Systems 28.

[14] Schelter, S., et al. (2018). Automating Large-Scale Data Quality Verification. Proceedings of the VLDB Endowment, 11(12), 1781-1794.

[15] Polyzotis, N., Roy, S., Whang, S. E., & Zinkevich, M. (2017). Data Management Challenges in Production Machine Learning. SIGMOD 2017.