IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 3, July 2026
Impact Factor: 8.2

A Review of Open Source Data Migration Tools for MySQL to MongoDB Migration

Mr. Bhushan R. Mankar[1] and Dr. Akshay Dhande[2]
PhD. Scholar Department of Electronics and Telecommunication Engineering[1],
Asst. Prof., Department of Electronics and Telecommunication Engineering[2],
P. R. Pote Patil College of Engineering and Management Amravati, India
akshaydhande126@gmail.com[2], https://orcid.org/0000 0002 7340 9806

Abstract: *Organizations increasingly migrate operational data from relational database management systems to NoSQL document stores to obtain horizontal scalability and schema flexibility. Migrating from a relational source such as MySQL to a document store such as MongoDB is non trivial: the two data models differ fundamentally, and a faithful migration requires deciding how related tables map onto nested (embedded) or referenced document structures. A range of open source data migration and extract–load–transform (ELT) tools is now widely used, yet the literature lacks a structured, side by side comparison of these tools on the relational to NoSQL task. This paper presents such a comparison, based on the official documentation, source repositories, and vendor reported benchmarks of four represent open source tools Airbyte, Meltano, Apache NiFi, and Mongify—for the migration of a relational schema from MySQL to MongoDB. We compare the tools along three axes: declared capabilities, expected data modelling fidelity, and reported performance and scalability. Our analysis shows a clear division: general purpose ELT tools (Airbyte, Meltano, NiFi) provide mature change data capture and high throughput ingestion but map relational tables to flat, one collection per entity document structures, whereas only the purpose built migrator (Mongify) exposes explicit embedding and referencing controls. In all cases the semantic decision of how to shape the target document model is left to the engineer. We argue that this gap motivates a new class of migration systems in which large language models (LLMs) and retrieval augmented generation (RAG) inform schema mapping and transformation, and we outline that design space as future work. This study provides the documentation grounded baseline against which such AI driven methods should be evaluated. Empirical benchmarking of the tools is identified as the next step of this research program.*

Keywords: Data migration, NoSQL, MongoDB, MySQL, ETL, schema mapping, open source software, large language models, retrieval augmented generation.

I. INTRODUCTION

THE volume, variety, and velocity of data managed by modern information systems have grown rapidly, prompt ing many organizations to move from relational database management systems (RDBMS) toward NoSQL data stores that offer horizontal scalability, flexible schemas, and high write throughput [1], [2]. Document oriented stores such as MongoDB are a common target because their nested document model aligns well with application level objects. As a result, relational to NoSQL data migration—moving both schema and data from a source such as MySQL to a target such as MongoDB—has become a recurring engineering task [3], [4]. The task is harder than it appears. The relational model organizes data into normalized tables connected by foreign keys, whereas the document model fevers denormalization, with related entities either embedded as sub documents or referenced across collections [5], [8]. Choosing between embedding and referencing is not a purely mechanical decision: it depends on the cardinality of relationships, the expected query workload, and update patterns. A migration that preserves table structure verbatim yields a technically correct but idiomatically poor document model, while a good model requires semantic judgment

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 3, July 2026
Impact Factor: 8.2

about the data and its intended use [6], [7]. In practice, engineers rely on general purpose open source data migration and ELT tools for example Airbyte, Meltano, and Apache NiFi or on purpose built migration utilities such as Mongify. These tools differ substantially in architecture: connector catalog ELT, Singer based pipelines, visual dataflow programming, and special purpose migrators. Despite their widespread use, the research literature offers little in the way of a structured side by side comparison of such tools on the relational to NoSQL task. Existing studies typically propose and evaluate a single migration algorithm [1], [2], [5], or compare database engines rather than the tools that move data between them [9]. This gap makes it difficult for practitioners to choose a tool on the basis of evidence, and it obscures a more fundamental question: how much of the semantic mapping problem do today's tools actually solve. This paper addresses that gap through a documentation based comparative analysis. Using the official documentation, public source repositories, and vendor reported benchmarks of four representative tools, we characterize and compare their capabilities, the data modelling fidelity their default behaviour implies, and their reported performance for MySQL to MongoDB migration. Our contributions are: 1) A capability framework and comparison for open source relational to NoSQL migration tools, covering schema extraction, type mapping, relationship handling (embed ding vs. referencing), incremental/change data capture (CDC) migration, and validation. 2) A structured comparison of Airbyte, Meltano, Apache NiFi, and Mongify along capabilities, expected modelling fidelity, and reported performance, grounded in primary documentation. 3) An analysis of the semantic gap left by current tools, showing where mechanical mapping is insufficient, and a discussion of how LLM and RAG based techniques could inform embedding/referencing and transformation decisions in a future AI driven migration system. The remainder of this paper is organized as follows. Section II reviews relational to NoSQL migration, open-source migration tooling, and emerging AI approaches to schema mapping. Section III describes the research questions, tools, scenario, and analysis method. Section IV defines the com parison criteria. Section V reports the comparison, Section VI discusses the gap toward AI driven migration and threats to validity, and Section VII concludes.

II. BACKGROUND AND RELATED WORK

A. Relational to NoSQL Migration

Migrating relational data to NoSQL document stores has been studied for over a decade. Early work established map ping strategies from MySQL to MongoDB and analysed the trade-offs of denormalization [1], [2]. Subsequent work reverse engineers the relational schema and generates a target document model from metadata and constraints, including parallel snapshot and live stream migration models [4]. More recent work incorporates the expected query workload to decide between embedding and referencing, arguing that schema only information is insufficient for a high quality mapping [5]. Unified metamodels bridging relational and NoSQL schemas [8], post migration data validation across heterogeneous models [6], and schema evolution spanning relational and NoSQL systems [7] complement this line. Comparative and review studies survey these approaches but stop short of a structured comparison of the tools practitioners actually use [3].

B. Open Source Data Migration and ELT Tooling

A parallel, largely industrial ecosystem of open source data movement tools has matured. Connector catalog ELT plat forms expose large libraries of source/destination connectors; Singer based frameworks emphasize modular, version controlled pipelines; visual dataflow systems model migration as a graph of processors; and purpose built migrators translate a relational schema directly into a document model with user specified embedding and referencing. These tools are widely deployed, but their behaviour on the relational to document task particularly the quality of the document model they produce by default has not been systematically compared. This paper treats these tools as the objects of study.

C. AI and LLMs for Schema Matching and Transformation

A rapidly growing body of work applies pre trained and large language models to data integration subtasks. Pre trained

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 3, July 2026
Impact Factor: 8.2

language models have been used for schema matching [10], and experimental studies characterize the strengths and limits of LLM based matching from element names and descriptions alone [11]. Retrieval augmented approaches such as Rematch ground the model in candidate context to improve matching [12], while self improving and compositional LLM programs [13] and hybrid small/large model systems [14] push accuracy further. Knowledge compliant, fine tuning free frameworks reduce dependence on labelled data [15], and instruction tuned models target broader data preparation [16]. Beyond matching, LLM based code generation has been used to synthesize data transformations automatically. These results suggest that the semantic decisions current migration tools avoid may be tractable with LLM and RAG based methods the direction this paper's broader research program pursues. The present study establishes the non AI baseline against which such methods should be measured.

Table I Selected Migration Tools and Represented Paradigms

Tool	Version †	Paradigm
Airbyte	1.x (2026)	Connector catalog ELT
Meltano	3.x (2026)	Singer based / CLI DataOps
Apache NiFi	2.x (2026)	Visual flow based dataflow
Mongify	1.x (2026)	Purpose built RDBMS → MongoDB migrator

† Latest documented release family consulted; fix the exact build before empirical evaluation.

III. METHOD

A. Research Questions The analysis is organized around three research questions: • RQ1 (Capabilities). Which relational to NoSQL migration capabilities schema extraction, type mapping, relationship handling, CDC/incremental migration, and validation do current open source tools support according to their documentation. • RQ2 (Modelling fidelity). What target document model does each tool produce by default, and how does that model represent relational relationships. • RQ3 (Performance). What performance and scalability characteristics do the tools report, and how do their processing models differ. B. Tools Under Comparison Tools were selected for architectural diversity so that findings generalize beyond a single design paradigm rather than for popularity alone. Table I lists the selected tools and the paradigm each represents. dbt was deliberately excluded as it is a transformation layer rather than a migration tool. All tools are open source and support a MongoDB target either natively or through a maintained connector. C. Migration Scenario The migration scenario is fixed as MySQL → MongoDB, the most widely used relational to document pair in the relocation literature [1], [2], [5]. For empirical follow up work (Section VII) we adopt the TPC H benchmark as source dataset, because it is an industry standard relational schema with explicit foreign key relationships making the embedding vs referencing decision meaningful and is generated at control label scale factors. Fig. 1 illustrates the migration scenario and the modelling decision at its core. D. Analysis Method and Sources The comparison is based exclusively on primary sources: each tool's official documentation, its public source repository, and vendor published benchmark reports where available. For each tool we recorded declared support for the capabilities in Section IV, the default target structure it produces, and any reported performance figures, together with the source consulted. Where a capability is achievable only through manual scripting or configuration rather than as a first class feature, it is recorded as partial. This documentation based method establishes a verifiable baseline; controlled empirical measurement is deferred to future work.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 3, July 2026
Impact Factor: 8.2

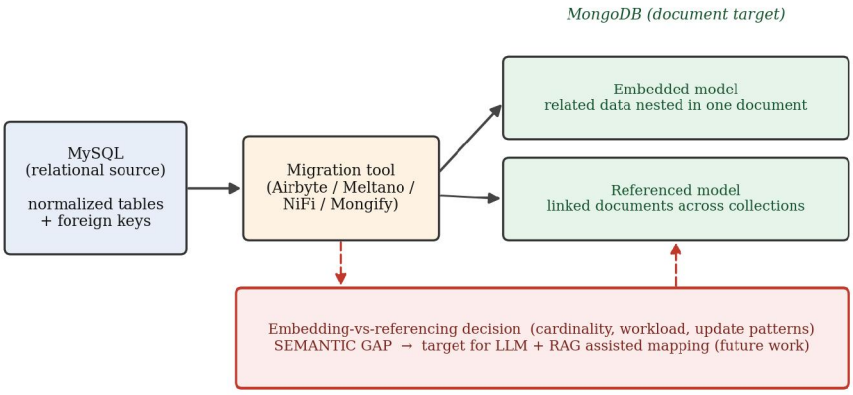

Fig. 1. MySQL-to-MongoDB migration and the embedding-vs-referencing decision. General-purpose tools map tables to flat collections; a good document model requires a semantic decision, the target for future LLM/RAG-assisted mapping.

IV. COMPARISON CRITERIA

Capabilities (RQ1) are recorded as a feature matrix (Table II) covering: relational schema extraction; primitive and complex type mapping; relationship handling (embedding, referencing, or flat); primary/foreign key preservation; incremental or CDC migration; and built in post migration vali dation. Modelling fidelity (RQ2) is characterized qualitatively (Table III) by the default target structure, how relationships are represented, and how source types are coerced. Performance (RQ3) is characterized (Table IV) by the processing model, parallelism/scaling approach, and any reported throughput. All cells are attributed to primary documentation; no values are the result of controlled experiments in this study.

V. RESULTS

A. Capability Coverage (RQ1) Table II summarizes declared capabilities. The general purpose ELT tools provide mature schema discovery, type mapping, and CDC/incremental migration, but none exposes embedding or referencing as a first class feature: by default, they create one target collection per source entity. Airbyte, as a MongoDB destination, creates a collection per stream with an inferred document schema. NiFi can achieve embedding only through manual transformation processors. In contrast, Mongify provides explicit embedding and references directives in its translation file, making relationship shaping a declared capability, at the cost of native CDC.

Table II Capability Coverage (● Supported, ○ Partial/Manual, Not Supported)

Capability	Airbyte	Meltano	NiFi	Mongify
Schema extraction	●	●	○	●
Type mapping	●	●	●	●
Embedding support	—	—	○	●
Referencing support	—	—	○	●
PK/FK preservation	○	○	○	●
Incremental / CDC	●	●	●	○
Built in validation	○	—	○	—

○ for NiFi embedding/referencing denotes achievable only via manual transformation processors, not a first class feature.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 3, July 2026
Impact Factor: 8.2

B. Modelling Fidelity (RQ2) Table III contrasts the default document model each tool produces. The ELT tools reproduce the relational structure faithfully but flatly: foreign keys are carried as ordinary fields rather than resolved into embedded or linked documents. Only Mongify produces a genuinely document oriented model, and only because the engineer specifies the relationships explicitly. No tool autonomously decides embedding vs. referencing from the data or workload.

C. Performance and Scalability (RQ3) Table IV summarizes reported performance characteristics. Only Airbyte publishes concrete MongoDB throughput figures—sustained rates above 10 MB/s and support for col lections beyond 1 TB with resumable, checkpointed syncs. NiFi is engineered for high throughput streaming with back pressure and horizontal clustering, but no directly com parable MySQL to MongoDB figure is published. Mongify is a single process bulk loader without a published large scale benchmark. Because these figures are gathered under differing conditions, they are reported as vendor claims, not as a controlled comparison; producing such a comparison is the motivation for the empirical follow up.

Table III Reported Performance and Scalability

Aspect	Airbyte	Meltano / NiFi	Mongify
Processing model	Batched connector sync	Batch / streaming dataflow	Single process bulk load
Scaling approach	Connector level, resumable	Clustered (NiFi); tap/target (Meltano)	Limited (single process)
Reported throughput	> 10 MB/s; 1 TB+	High throughput design; not published	Not published
CDC / incremental	●	●	○

VI. DISCUSSION

A. The Gap Toward AI Driven Migration The comparison delineates a semantic gap. General purpose tools excel at the mechanical aspects of migration connectivity, type coercion, CDC, and throughput but leave the embedding vs referencing and type refinement decisions, which most influence document model quality, entirely to the engineer. The one tool that shapes the document model (Mongify) does so only from manually authored directives. In other words, no current open source tool reasons about relationships or workload to choose a good target model. This is precisely the class of problem where LLM and RAG based methods have shown promise for related data integration subtasks [11] [13]. A future AI driven migration system could use retrieval to ground a language model in the source schema, sample data, and workload, and generate a target document model and transformation logic while still relying on the mature ingestion and CDC machinery of existing tools for execution, and on established validation methods [6] to guarantee correctness. This study provides the documentation grounded baseline such a system must improve upon.

B. Threats to Validity Construct: the comparison relies on vendor documentation, which may overstate capabilities or omit limitations; we mitigate this by citing primary sources and marking manually achievable features as partial. Internal: tool versions evolve, so capability claims are time stamped to the consulted release family. External: the analysis targets a single source target pair (MySQL → MongoDB); other pairs may shift the balance among tools. Absence of measurement: this study does not report controlled performance or fidelity measurements these are explicitly deferred to the empirical study in future work, and no reported figure should be read as a result of a controlled experiment here.

DOI: 10.48175/IJARSCT-37838

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 3, July 2026
Impact Factor: 8.2

VII. CONCLUSION AND FUTURE WORK

We presented a documentation based comparative analysis of four open source tools for MySQL to MongoDB migration, comparing capabilities, modelling fidelity, and reported performance. The analysis shows that current open source tools solve the mechanical aspects of migration well schema discovery, type mapping, CDC, and high throughput ingestion but leave the semantic document modelling decisions to the engineer; only a purpose built migrator exposes explicit embed ding and referencing, and none decides them autonomously. This gap motivates our broader research program: an AI driven migration system in which large language models and retrieval augmented generation inform schema mapping and transformation.

Future work will (i) execute a controlled empirical benchmark of these tools on TPC H across scale factors, measuring fidelity and performance; (ii) extend the comparison to additional source target pairs; and (iii) de sign and evaluate the proposed AI driven migration approach against the baseline established here.

REFERENCES

1. El Alami and M. Bahaj, "Migration of a relational databases to NoSQL: The way forward," in Proc. 5th Int. Conf. Multimedia Comput. Syst. (ICMCS), IEEE, 2016, pp. 18–23. Electronic ISBN:978 1 5090 5146 5
2. F. Yassine and M. A. Awad, "Migrating from SQL to NoSQL database: Practices and analysis," in Proc. 13th Int. Conf. Innov. Inf. Technol. (IIT), IEEE, 2018, pp. 58–62, doi: 10.1109/INNOVATIONS.2018.8606019. Electronic ISBN: 978 1 5386 6673 9
3. Saadouni, K. El Bouchti, O. M. Reda, and S. Ziti, "Data migration from relational to NoSQL database: Review and comparative study," in Proc. Int. Conf. Adv. Intell. Syst. Sustain. Dev., Springer, 2023, doi: 10.1007/978 3 031 26384 2 22.
4. Namdeo and U. Suman, "A model for relational to NoSQL database migration: Snapshot live stream DB migration model," in Proc. 7th Int. Conf. Adv. Comput. Commun. Syst. (ICACCS), IEEE, 2021, pp. 199–204. DOI: 10.1109/ICACCS51430.2021.9441829
5. Wilson Tandya; Fazat Nur Azizah "Migration of relational database to NoSQL document oriented database," in Proc. IEEE Int. Conf., 2023, Electronic ISBN:979 8 3503 8138 2 doi: 10.1109/ICoDSE59534.2023.10291584.
6. S. Ramzan, I. S. Bajwa, B. Ramzan, and W. Anwar, "Intelligent data engineering for migration to NoSQL based secure environments," IEEE Access, vol. 7, pp. 69042–69057, 2019, doi: 10.1109/AC CESS.2019.2916912.
7. H. Chill´on, M. Klettke, D. S. Ruiz, and J. G. Molina, "A generic schema evolution approach for NoSQL and relational databases," IEEE Trans. Knowl. Data Eng., vol. 36, no. 7, pp. 2774–2789, 2024. DOI: 10.1109/TKDE.2024.3362273
8. J. Fern´andez Candel, D. Sevilla Ruiz, and J. J. Garc´ ıa Molina, "A unified metamodel for NoSQL and relational databases," Inf. Syst., vol. 104, art. 101898, 2022, doi: 10.1016/j.is.2021.101898.
9. D. Damodaran, S. Salim, and S. M. Vargese, "Performance evaluation of MySQL and MongoDB databases," Int. J. Cybern. Inform., vol. 5, pp. 387–394, 2016. Y. Zhang et al., "Schema matching using pre trained language models," in Proc. IEEE 39th Int. Conf. Data Eng. (ICDE), 2023, pp. 1558–1571. DOI: 10.5121/ijci.2016.5241
10. M. Parciak, B. Vandevoort, F. Neven, L. M. Peeters, and S. Vansum meren, "Schema matching with large language models: An experimental study," arXiv:2407.11852, 2024. DOI: 10.1109/ICDE55515.2023.00123
11. Sheetrit, M. Brief, M. Mishaeli, and O. Elisha, "ReMatch: Retrieval enhanced schema matching with LLMs," arXiv:2403.01567, 2024. doi: 10.48550/arXiv.2407.11852
12. N. Seedat and M. van der Schaar, "Matchmaker: Self improving large language model programs for schema matching," arXiv:2410.24105, 2024. DOI: 10.48550/arXiv.2403.01567

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 3, July 2026
Impact Factor: 8.2

13. Y. Liu, E. Pena, A. Santos, E. Wu, and J. Freire, "Magneto: Combining small and large language models for schema matching," arXiv:2412.08194, 2024. DOI: 10.48550/arXiv.2410.24105
14. Y. Xu, H. Li, K. Chen, and L. Shou, "KcMF: A knowledge compliant framework for schema and entity matching with fine tuning free LLMs," arXiv:2410.12480, 2024. DOI: 10.48550/arXiv.2412.08194
15. H. Zhang, Y. Dong, C. Xiao, and M. Oyamada, "Jellyfish: Instruction tuning local large language models for data pre processing," in Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP), 2024, pp. 8754–8782. DOI: 10.48550/arXiv.2410.12480
16. Sharma et al., "DataMorpher: Automatic data transformation using LLM based zero shot code generation," in Proc. IEEE 41st Int. Conf. Data Eng. (ICDE), 2025, pp. 4536–4539. DOI: 10.18653/v1/2024.emnlp main.497